Guide de l’utilisateur

AWS Modernisation du mainframe

AWS Modernisation du mainframe: Guide de l’utilisateur

Table of Contents

Qu'est-ce que la modernisation AWS du mainframe ?

AWS La modernisation du mainframe vous aide à moderniser vos applications mainframe pour les adapter à des environnements d'exécution AWS gérés. Il fournit des outils et des ressources pour vous aider à planifier et à implémenter la migration et la modernisation. Vous pouvez analyser vos applications mainframe existantes, les développer ou les mettre à jour à l'aide de COBOL ou PL/I, et mettre en œuvre un pipeline automatisé pour l'intégration continue et la livraison continue (CI/CD) des applications. Vous pouvez choisir entre des modèles de refactoring automatique ou de replateforme, en fonction des besoins de vos clients. Si vous êtes consultant et que vous aidez un client à migrer ses charges de travail sur le mainframe, vous pouvez utiliser les outils de modernisation du AWS mainframe pour toutes les phases du processus de migration et de modernisation, de la planification initiale aux opérations cloud après la migration.

Vous pouvez utiliser la modernisation du AWS mainframe pour créer et gérer efficacement l'environnement d'exécution de vos applications mainframe, ainsi que AWS pour gérer et surveiller vos applications modernisées.

Rubriques

> **Note**
>
> Avez-vous fait appel à des partenaires compétents en matière de migration de AWS mainframe ou à des services AWS professionnels pour votre projet de modernisation de mainframe ? Si ce n'est pas le cas, nous vous recommandons vivement de faire appel à des experts pour votre projet.
>
> - AWS Partenaires compétents en matière de modernisation des ordinateurs centraux

- AWS Professional Services

Les fonctionnalités et les cas d'utilisation de la modernisation des AWS mainframes soutiennent une approche de modernisation évolutive, qui offre des avantages à court terme en améliorant l'agilité et en offrant de nombreuses opportunités d'optimisation et d'innovation par la suite. Pour de plus amples informations, veuillez consulter Approche de modernisation.

Caractéristiques de la modernisation des AWS ordinateurs centraux

AWS Les fonctionnalités de modernisation du mainframe prennent en charge les cas d'utilisation suivants :

- Évaluation : La fonctionnalité d'évaluation de AWS Mainframe Modernization peut vous aider à évaluer, définir et planifier un projet de migration et de modernisation.
- Refactor : grâce à AWS Blu Age, vous pouvez utiliser le refactoring pour convertir les anciens langages de programmation d'applications, pour créer des macroservices ou des microservices et pour moderniser les interfaces utilisateur (UI) et les piles de logiciels d'application.

  AWS Blu Insights est désormais disponible AWS Management Console via l'authentification unique. Vous n'avez plus besoin de gérer des informations d'identification AWS Blu Insights distinctes. Vous pouvez accéder aux fonctionnalités de la base de code AWS AWS Blu Age et du centre de transformation directement depuis le AWS Management Console.
- Replateforme : grâce à la solution Micro Focus Enterprise, vous pouvez porter l'application sur laquelle une grande partie du code source de l'application est recompilée sans modification.
- IDE pour développeurs : AWS Mainframe Modernization propose un environnement de développement intégré (IDE) à la demande qui permet aux développeurs d'écrire du code plus rapidement grâce à une édition et un débogage intelligents, à une compilation instantanée du code et à des tests unitaires.
- Temps d'exécution géré : L'environnement d'exécution géré AWS Mainframe Modernization surveille en permanence vos clusters afin de garantir le bon fonctionnement des charges de travail de l'entreprise grâce à des calculs autoréparants et à une mise à l'échelle automatisée.
- Intégration et livraison continues (CI/CD) : La fonctionnalité CI/CD de AWS Mainframe Modernization aide les équipes de développement d'applications à apporter des modifications de

code plus fréquemment et de manière plus fiable, ce qui accélère la vitesse de migration, améliore la qualité et contribue à réduire time-to-market le nombre de nouvelles fonctions commerciales disponibles.

- Intégrations avec d'autres AWS services : la modernisation AWS du mainframe permet AWS CloudFormation AWS PrivateLink, et AWS Key Management Service pour un déploiement reproductible et une sécurité et une conformité accrues.
- Disponibilité étendue : la modernisation des AWS ordinateurs centraux est désormais disponible dans l'est des États-Unis (Ohio), l'ouest des États-Unis (Californie du Nord), l'Asie-Pacifique (Mumbai), l'Asie-Pacifique (Séoul), l'Asie-Pacifique (Singapour), l'Asie-Pacifique (Tokyo), l'Europe (Londres) et l'Europe (Paris).

Pour plus d'informations sur les fonctionnalités de modernisation AWS du mainframe, consultezhttps://aws.amazon.com/mainframe-modernization/features/.

Modèles

Le modèle de refactorisation automatisée, développé par AWS Blu Age, vise à accélérer la modernisation en convertissant l'ensemble des applications existantes et sa couche de données en une application Java moderne tout en préservant l'équivalence fonctionnelle. Au cours de cette transformation automatisée, il crée une application à plusieurs niveaux avec un front-end basé sur Angular, un backend Java compatible avec les API et une couche de données accédant aux magasins de données modernes. Le processus de refactorisation fournit des fonctionnalités équivalentes à celles de l'ancienne solution afin d'accroître l'automatisation des projets, ce qui se traduit par une rapidité, une qualité et une réduction des coûts, ce qui permet d'obtenir des avantages commerciaux plus rapidement. Pour plus d'informations, consultez la section AWS Mainframe Modernization Automated Refactor.

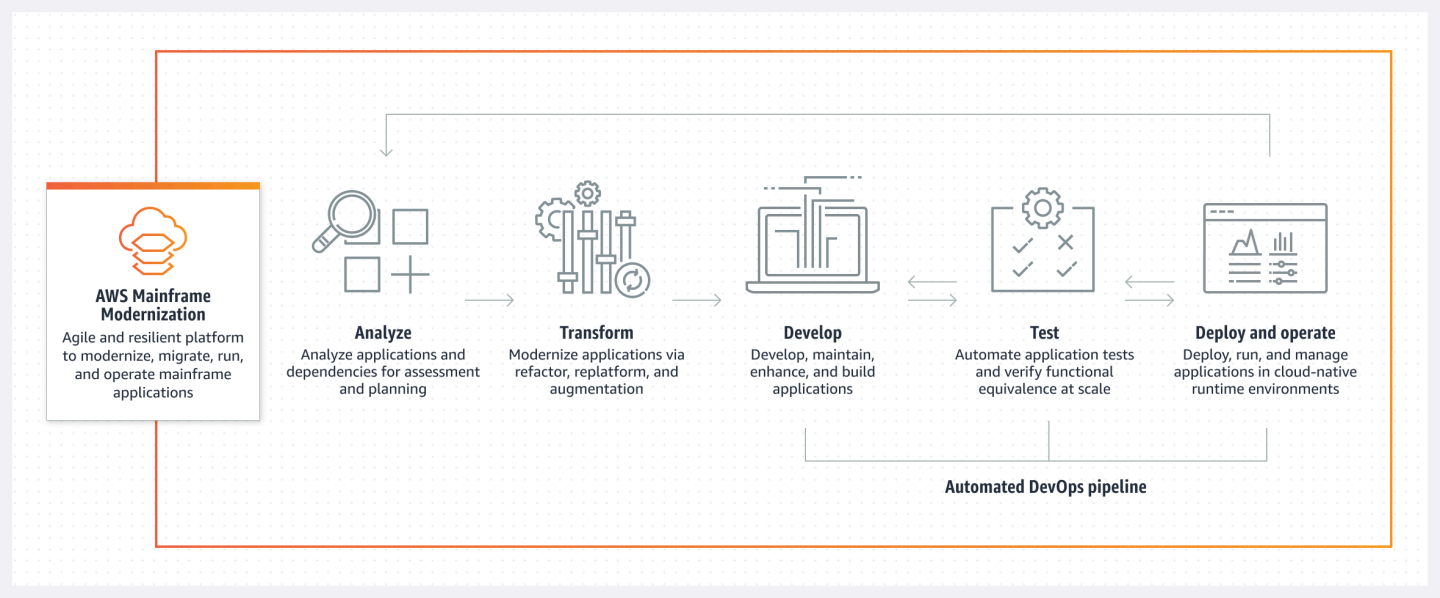
Le modèle de replateforme, basé sur la suite Micro Focus Enterprise, vise à préserver le langage, le code et les artefacts de l'application afin de minimiser l'impact sur les actifs et les équipes de l'application. Il aide les clients à maintenir leurs connaissances et leurs compétences en matière d'applications. Bien que les modifications apportées aux applications soient limitées, ce modèle facilite également la modernisation de l'infrastructure et des processus. L'infrastructure est remplacée par un service géré moderne basé sur le cloud, tandis que les processus sont modifiés pour suivre les meilleures pratiques en matière de développement d'applications et d'opérations informatiques. Pour plus d'informations, consultez la section AWS Mainframe Modernisation Replatform.

Comment démarrer avec la modernisation du AWS mainframe

À vous d'essayer ! Nous proposons des didacticiels et des exemples d'applications pour vous aider à vous faire une idée de ce qu'offre la modernisation des AWS mainframes. Choisissez le Tutoriel : Managed Runtime pour AWS Blu Age ou le Tutoriel : environnement d'exécution géré pour Micro Focus pour un step-by-step didacticiel complet.

Si vous êtes intéressé par le refactoring automatique, consultez les outils AWS Blu Age sur. BluInsights Vous pouvez également configurer la AppStream version 2.0 pour accéder à l'IDE AWS Blu Age Developer ou aux outils Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer.

Les didacticiels et les exemples d'applications ne vous donnent qu'une idée des avantages de la modernisation des AWS mainframes. Lorsque vous êtes prêt à démarrer un projet de modernisation, consultez Approche de modernisation pour plus de détails sur les étapes et les tâches d'un projet de modernisation.



Services connexes

Outre Blu Insights pour le refactoring automatisé, vous pouvez utiliser les AWS services suivants avec AWS Mainframe Modernization.

- Amazon RDS pour héberger vos bases de données migrées.
- Amazon S3 pour le stockage des fichiers binaires et des fichiers de définition des applications.
- Amazon FSx ou Amazon EFS pour le stockage des données d'application.

- Amazon AppStream pour accéder aux outils Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer.
- AWS CloudFormation pour le DevOps pipeline automatisé que vous pouvez utiliser pour configurer le CI/CD pour vos applications migrées.
- AWS Migration Hub
- AWS DMS pour la migration de vos bases de données.

Accès à la AWS modernisation du mainframe

Actuellement, vous pouvez accéder à la modernisation AWS du mainframe via la console à l'adresse https://console.aws.amazon.com/m2/. Pour obtenir la liste des régions dans lesquelles la modernisation des AWS mainframes est disponible, consultez la section Points de terminaison et quotas de modernisation des AWS mainframes dans le. Référence générale d'Amazon Web Services

Utilisez-vous la solution AWS Mainframe Modernisation pour la première fois ?

Si vous utilisez AWS Mainframe Modernization pour la première fois, nous vous recommandons de commencer par lire les sections suivantes :

- Prise en main
- Configuration

Tarification

AWS La modernisation du mainframe entraîne des frais pour l'utilisation d'instances prenant en charge les environnements d'exécution gérés. En outre, la modernisation AWS du mainframe propose certains outils sans frais supplémentaires. Vous êtes responsable des frais engagés pour les autres AWS services que vous utilisez dans le cadre de la modernisation du AWS mainframe. AWS fournira un préavis de 30 jours avant que toute modification tarifaire ne prenne effet dans le cadre de l'utilisation de la modernisation du AWS mainframe. Pour plus d'informations, consultez la section Modernisation du mainframe avec AWS.

Avec AWS Blu Insights, vous payez pour l'utilisation du centre de transformation. Pour plus d'informations, consultez la section Tarification de la modernisation des AWS mainframes.

Configuration de la modernisation AWS du mainframe

Avant de commencer à utiliser AWS Mainframe Modernization, vous ou votre administrateur devez suivre certaines étapes.

Rubriques

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur root a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur administratif, et à uniquement utiliser l'utilisateur root pour effectuer tâches nécessitant un accès utilisateur root.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l'activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en cliquant sur Mon compte.

Création d'un utilisateur administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l’aide pour vous connecter en utilisant l’utilisateur root, consultez Connexion en tant qu’utilisateur root dans le Guide de l’utilisateur Connexion à AWS .
2. Activez l’authentification multifactorielle (MFA) pour votre utilisateur root.

   Pour obtenir des instructions, voir Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d’un utilisateur administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d’ AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur administratif.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu’utilisateur administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l’URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l’utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Commencer à moderniser le AWS mainframe

Pour démarrer avec AWS Mainframe Modernization, vous pouvez suivre des didacticiels qui vous présentent le service et chaque moteur d'exécution.

Rubriques

- Tutoriel : Managed Runtime pour AWS Blu Age
- Tutoriel : environnement d'exécution géré pour Micro Focus

Pour continuer à apprendre, consultez les didacticiels suivants.

- Tutoriel : Configuration de la version Micro Focus pour l' BankDemo exemple d'application
- Tutoriel : Configuration d'un pipeline CI/CD à utiliser avec Micro Focus Enterprise Developer

Tutoriel : Managed Runtime pour AWS Blu Age

Ce didacticiel explique comment déployer une application modernisée AWS Blu Age dans un environnement d'exécution de modernisation AWS du mainframe.

Rubriques

- Prérequis
- Étape 1 : Téléchargez l'application de démonstration
- Étape 2 : Création de la définition de l'application
- Étape 3 : Création d'un environnement d'exécution
- Étape 4 : Création d'une application
- Étape 5 : Déploiement d'une application
- Étape 6 : démarrer une application
- Étape 7 : Accédez à l'application
- Étape 8 : Tester l'application
- Nettoyage des ressources

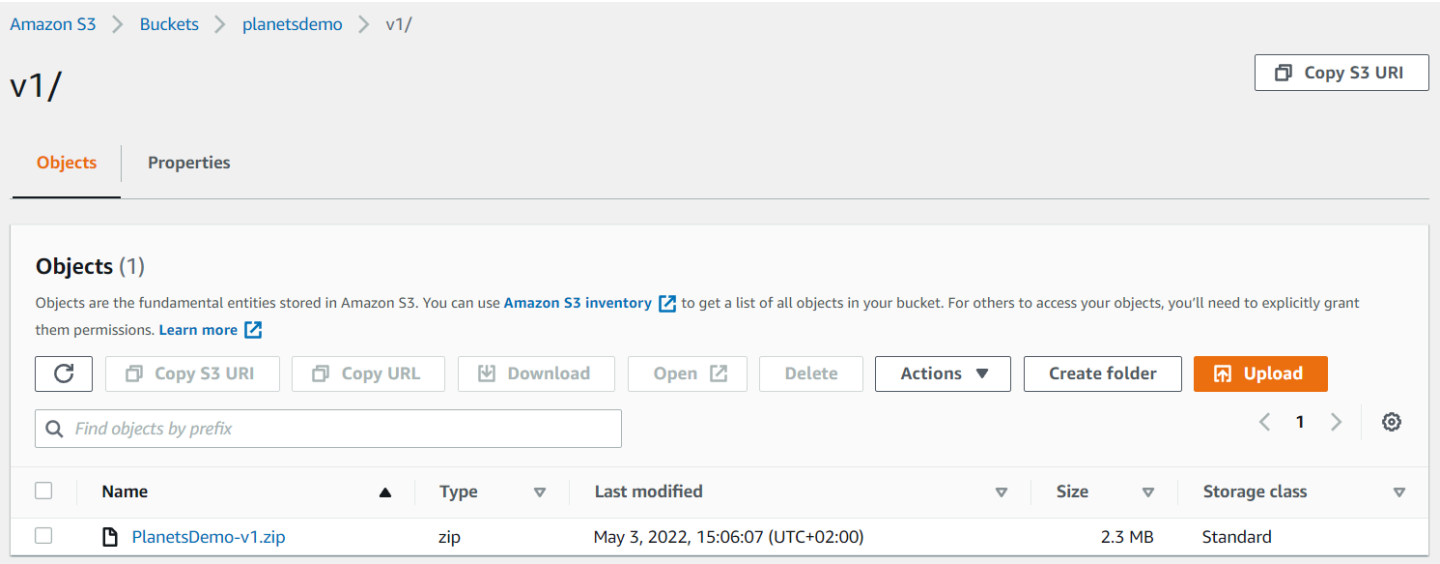
Prérequis

Pour terminer ce didacticiel, téléchargez l'archive PlanetsDemo-v1.zip de l'application de démonstration.

L'application de démonstration en cours d'exécution nécessite un navigateur moderne pour y accéder. Le fait que vous exécutiez ce navigateur depuis votre bureau ou depuis une instance Amazon Elastic Compute Cloud, par exemple au sein du VPC, détermine vos paramètres de sécurité.

Étape 1 : Téléchargez l'application de démonstration

Téléchargez l'application de démonstration dans un compartiment Amazon S3. Assurez-vous que ce compartiment se trouve dans le même compartiment que celui dans Région AWS lequel vous allez déployer l'application. L'exemple suivant montre un bucket nommé planetsdemo, avec un préfixe de clé, ou dossier, nommé v1 et une archive nommée. `planetsdemo-v1.zip`



> **Note**
>
> Le dossier du compartiment est obligatoire.

Étape 2 : Création de la définition de l'application

Pour déployer une application sur le runtime géré, vous avez besoin d'une définition de l'application AWS Mainframe Modernization. Cette définition est un fichier JSON qui décrit l'emplacement et les

paramètres de l'application. L'exemple suivant est une telle définition d'application pour l'application de démonstration :

```
{
    "template-version": "2.0",
    "source-locations": [{
        "source-id": "s3-source",
        "source-type": "s3",
        "properties": {
            "s3-bucket": "planetsdemo",
            "s3-key-prefix": "v1"
        }
    }],
    "definition": {
        "listeners": [{
            "port": 8196,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/PlanetsDemo-v1.zip"
        }
    }
}
```

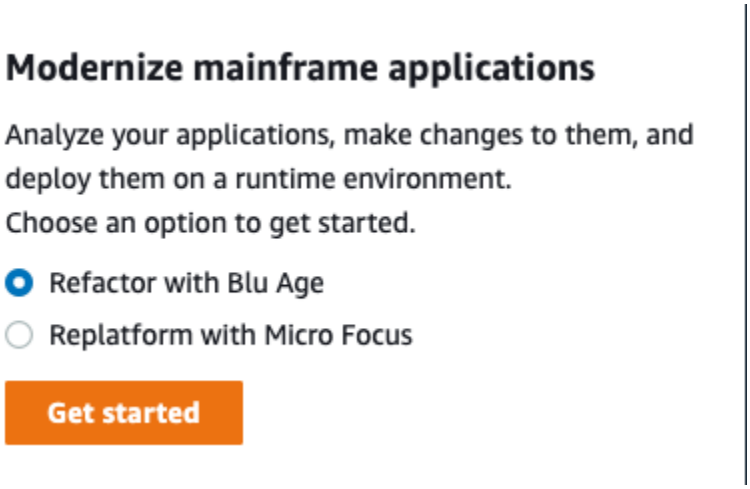
Remplacez l'`s3-bucket`entrée par le nom du compartiment dans lequel vous avez stocké le fichier zip de l'application d'exemple.

Pour plus d'informations sur la définition de l'application, consultezAWS Exemple de définition d'application Blu Age.

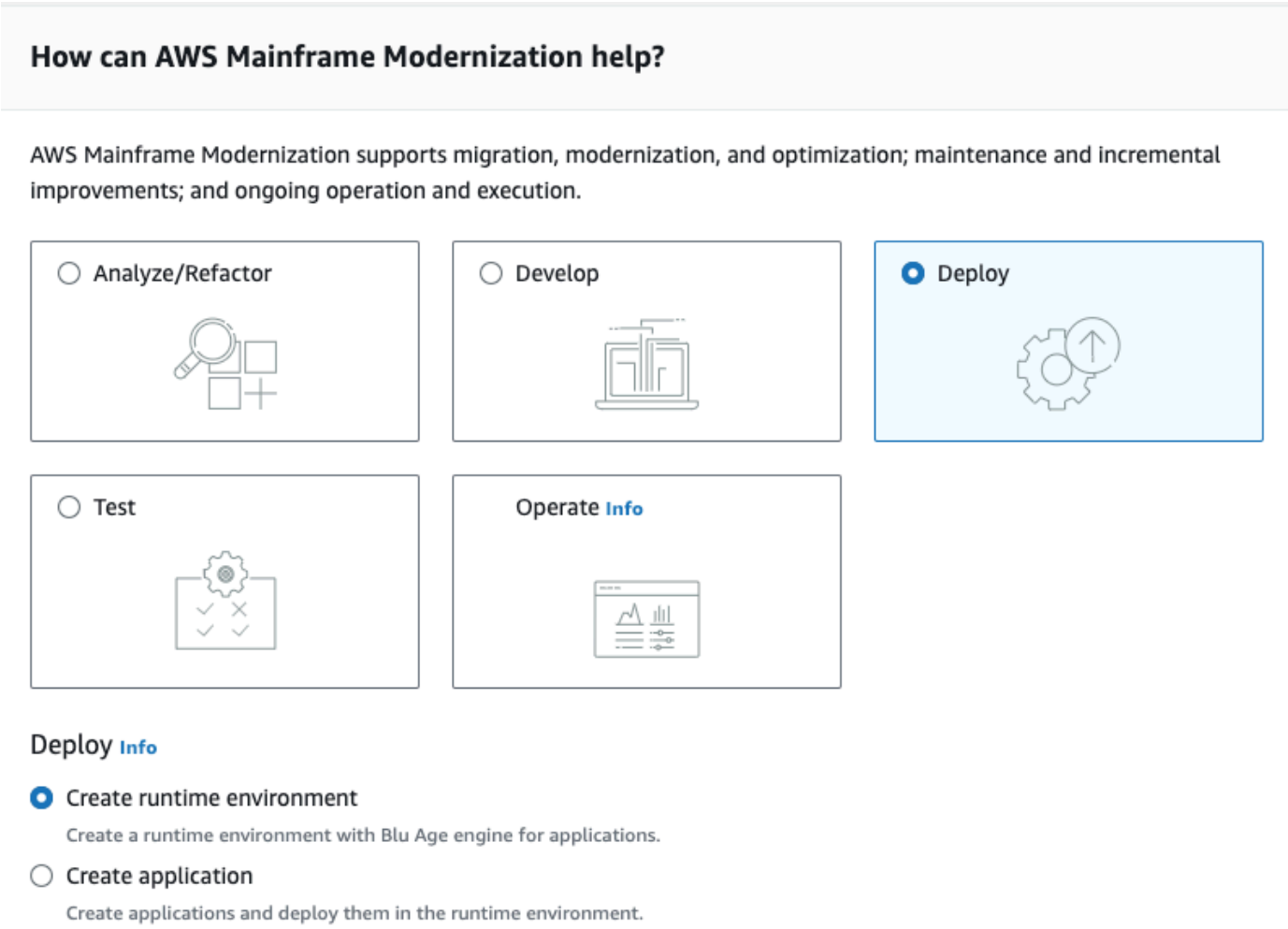
Étape 3 : Création d'un environnement d'exécution

Pour créer l'environnement d'exécution AWS Mainframe Modernization, effectuez les opérations suivantes :

1. Ouvrez la console de modernisation AWS du mainframe.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'environnement. Cela Région AWS doit correspondre à la région dans laquelle vous avez créé le compartiment S3Étape 1 : Téléchargez l'application de démonstration.
3. Sous Moderniser les applications mainframe, choisissez Refactor with Blu Age, puis choisissez Get started.

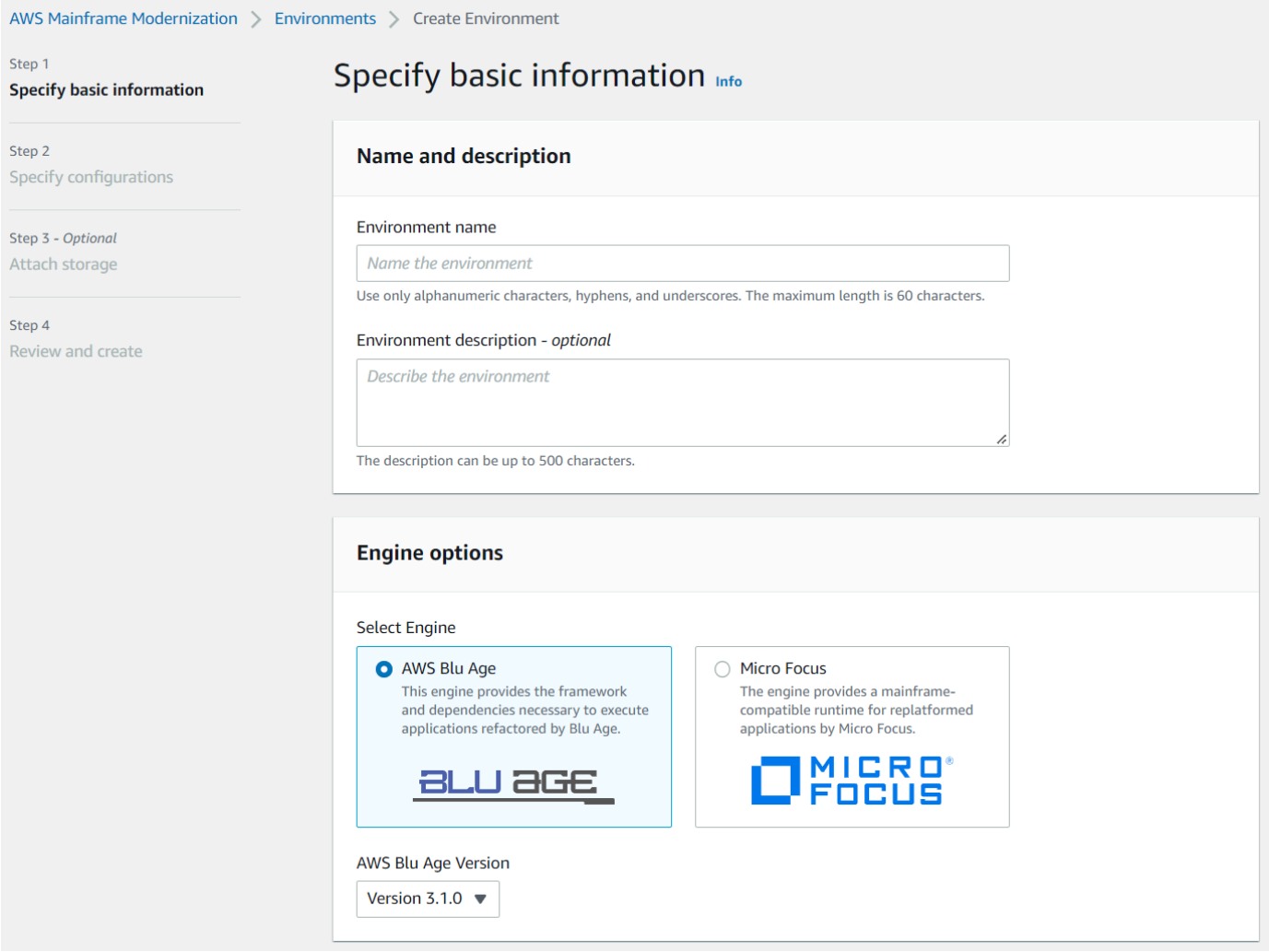


4. Dans la section Comment la modernisation du mainframe AWS peut-elle vous aider, choisissez Déployer et créer un environnement d'exécution.

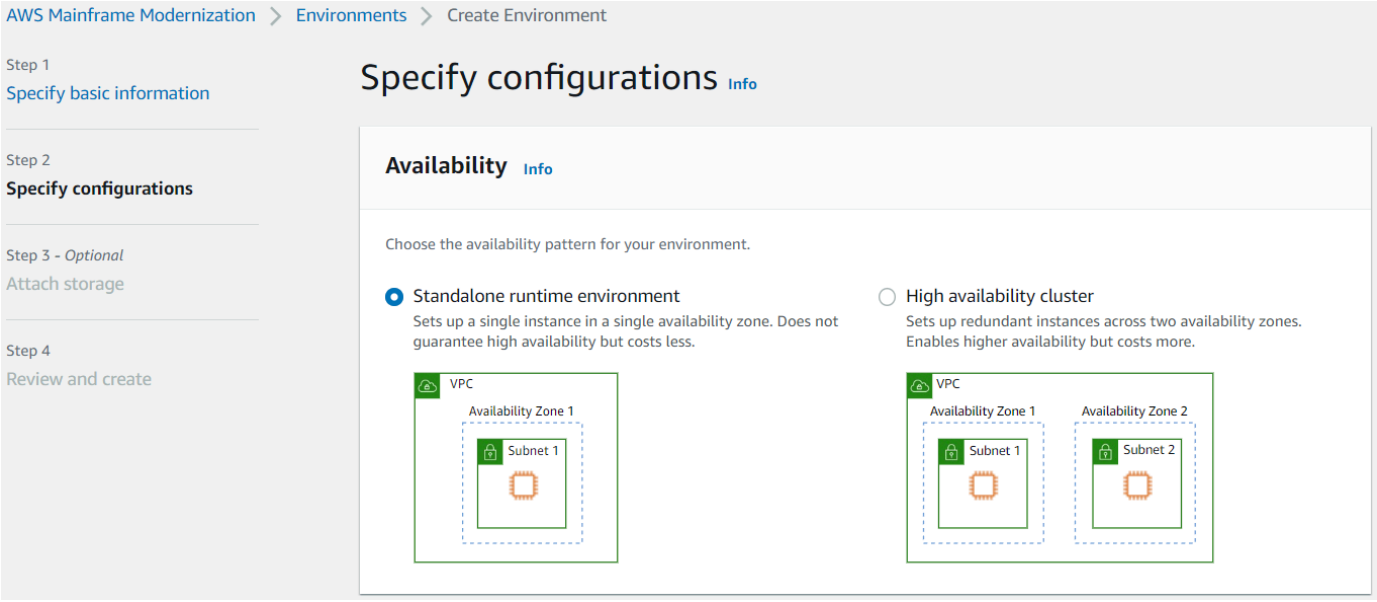


5. Dans le volet de navigation de gauche, choisissez Environments, puis Create environment. Sur la page Spécifier les informations de base, entrez le nom et la description de votre

environnement, puis assurez-vous que le moteur AWS Blu Age est sélectionné. Vous pouvez éventuellement ajouter des balises à la ressource créée. Ensuite, sélectionnez Suivant.

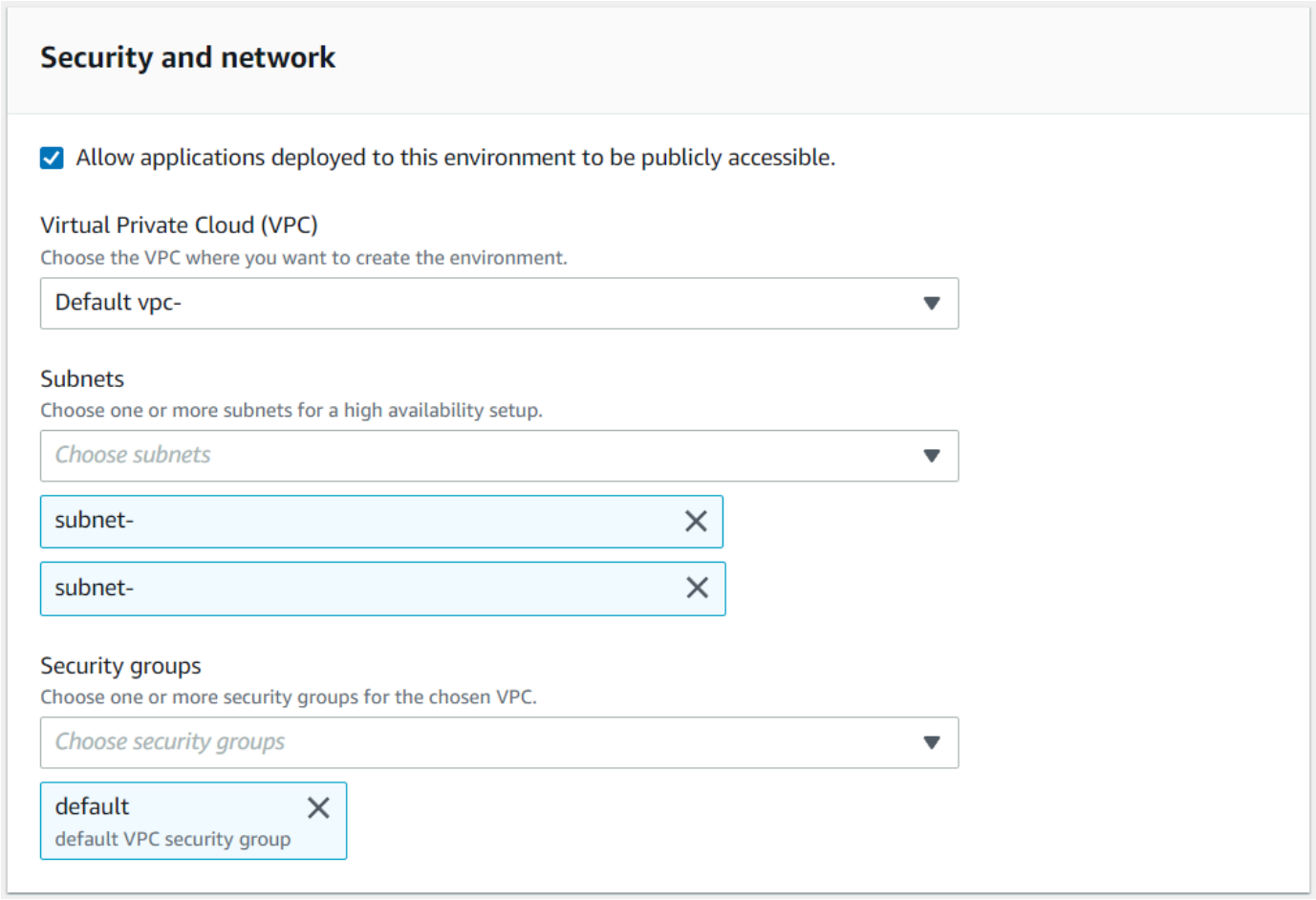


6. Sur la page Spécifier les configurations, sélectionnez Environnement d'exécution autonome.



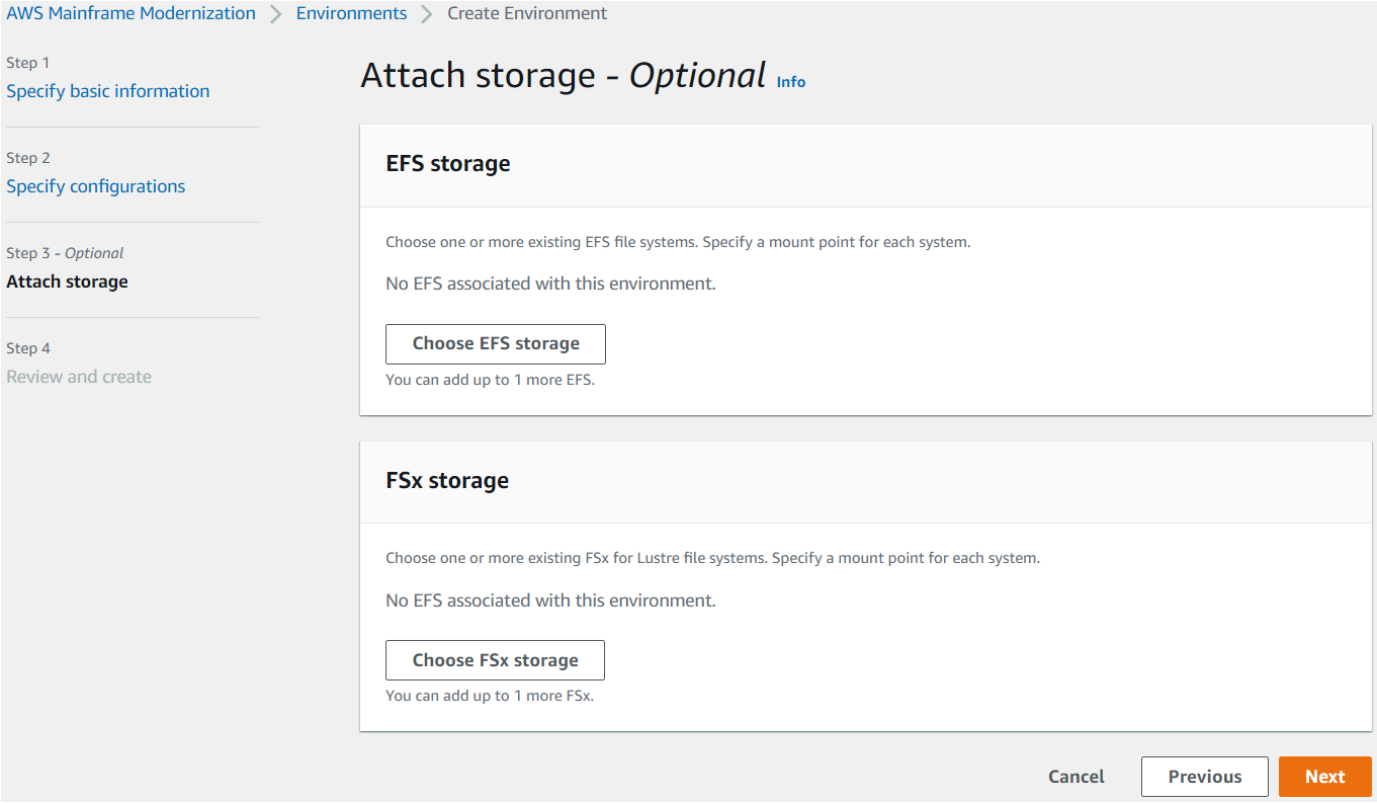
7. Sous Sécurité et réseau, apportez les modifications suivantes :

   - Choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public. Cette option attribue une adresse IP publique à l'application afin que vous puissiez y accéder depuis votre bureau.
   - Choisissez un VPC. Vous pouvez utiliser la valeur par défaut.
   - Choisissez deux sous-réseaux. Assurez-vous que les sous-réseaux autorisent l'attribution d'adresses IP publiques.
   - Sélectionnez un groupe de sécurité. Vous pouvez utiliser la valeur par défaut. Assurez-vous que le groupe de sécurité que vous choisissez autorise l'accès depuis l'adresse IP du navigateur au port que vous avez spécifié dans la `listener` propriété de la définition de l'application. Pour plus d'informations, consultez Étape 2 : Création de la définition de l'application.



Si vous souhaitez accéder à l'application depuis l'extérieur du VPC que vous avez choisi, assurez-vous que les règles de trafic entrant pour ce VPC sont correctement configurées. Pour plus d'informations, consultez Impossible d'accéder à l'URL d'une application.

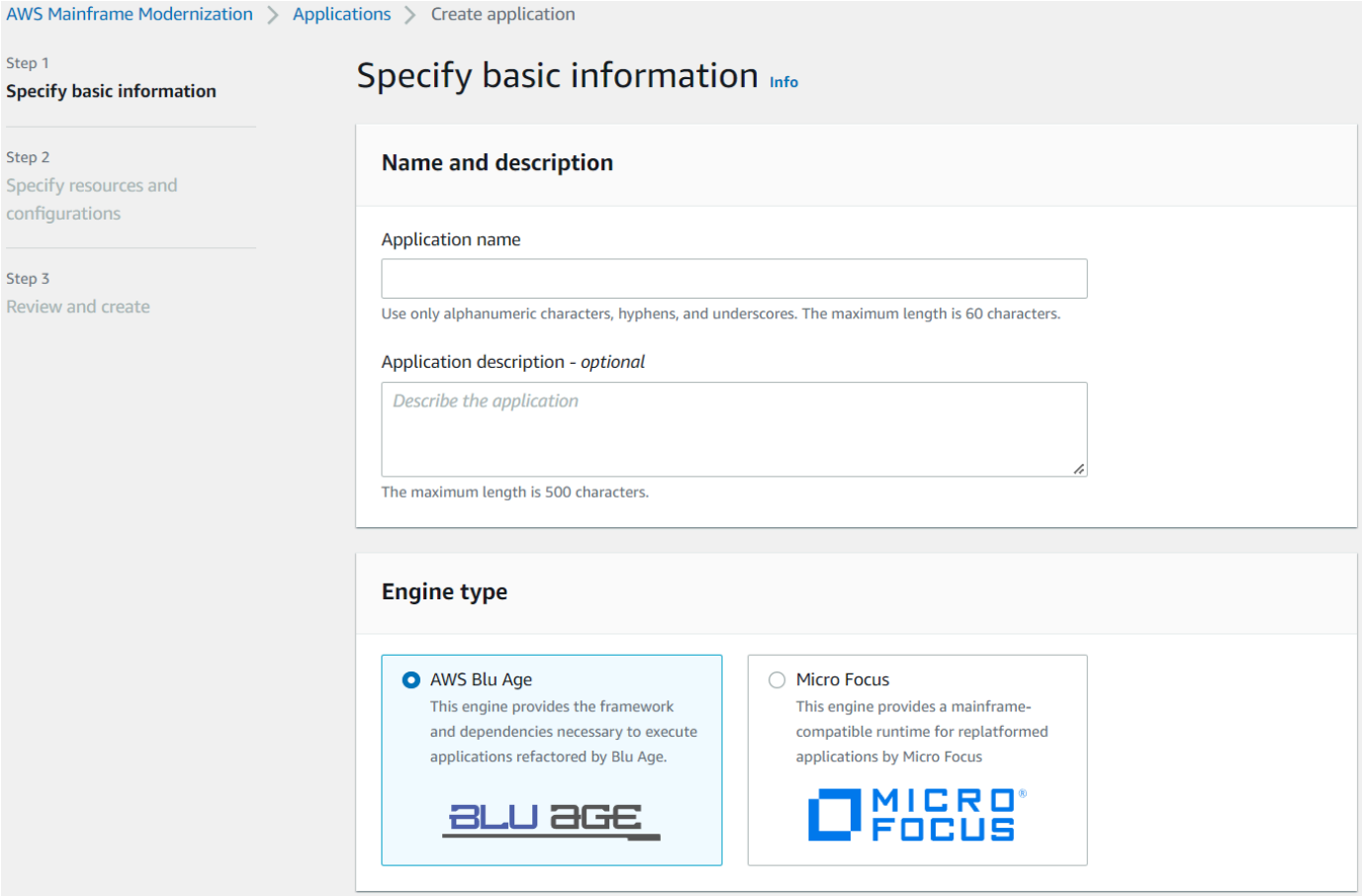
8. Choisissez Suivant.
9. Dans Joindre un espace de stockage - Facultatif, laissez les sélections par défaut et choisissez Suivant.



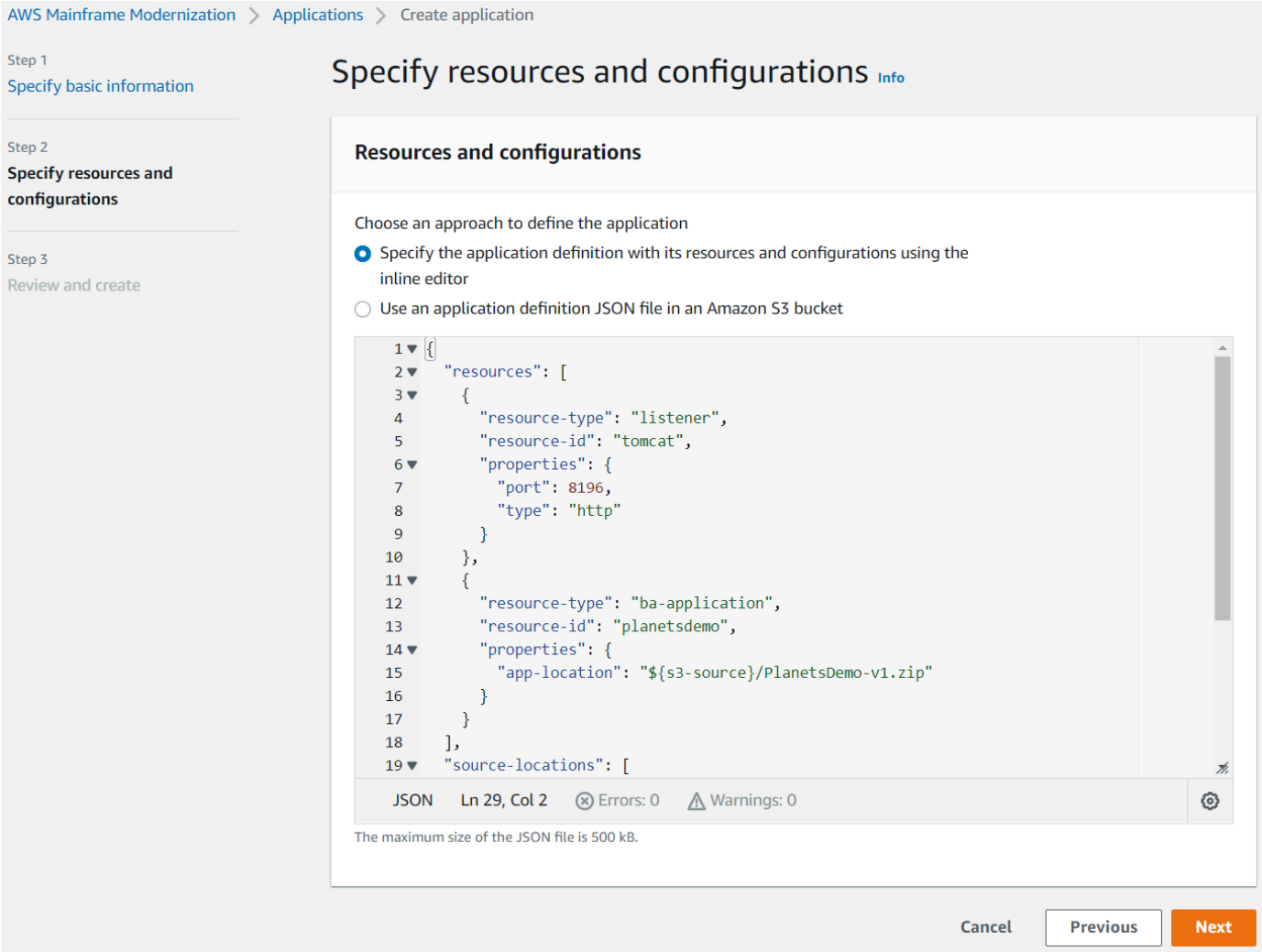
10. Dans Planifier la maintenance, choisissez Aucune préférence, puis cliquez sur Suivant.
11. Dans Révision et création, passez en revue les informations, puis choisissez Créer un environnement.

Étape 4 : Création d'une application

1. Accédez à la modernisation du mainframe AWS dans leAWS Management Console.
2. Dans le volet de navigation, choisissez Applications (Applications), puis Create a new application (Créer une nouvelle application). Sur la page Spécifier les informations de base, entrez le nom et la description de l'application et assurez-vous que le moteur AWS Blu Age est sélectionné. Ensuite, sélectionnez Suivant.



3. Sur la page Spécifier les ressources et les configurations, copiez et collez le JSON de définition d'application mis à jour dans lequel vous l'avez crééethe section called "Étape 2 : Création de la définition de l'application".

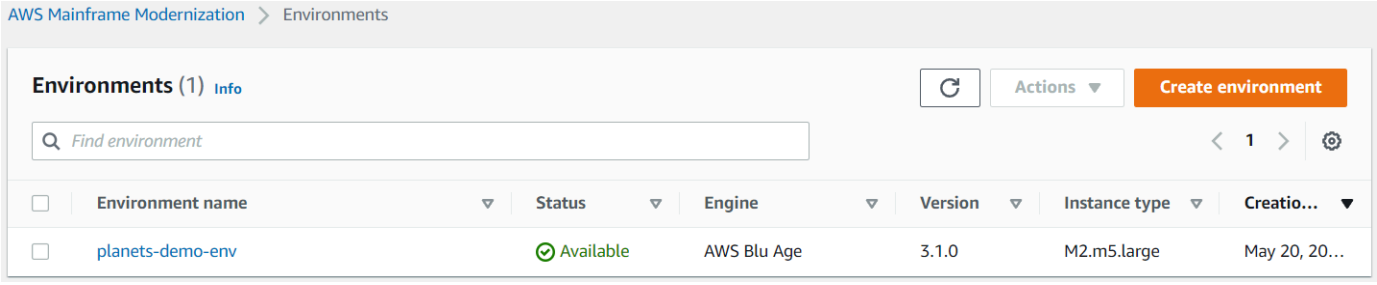


4. Dans Révision et création, passez en revue vos choix, puis choisissez Créer une application.

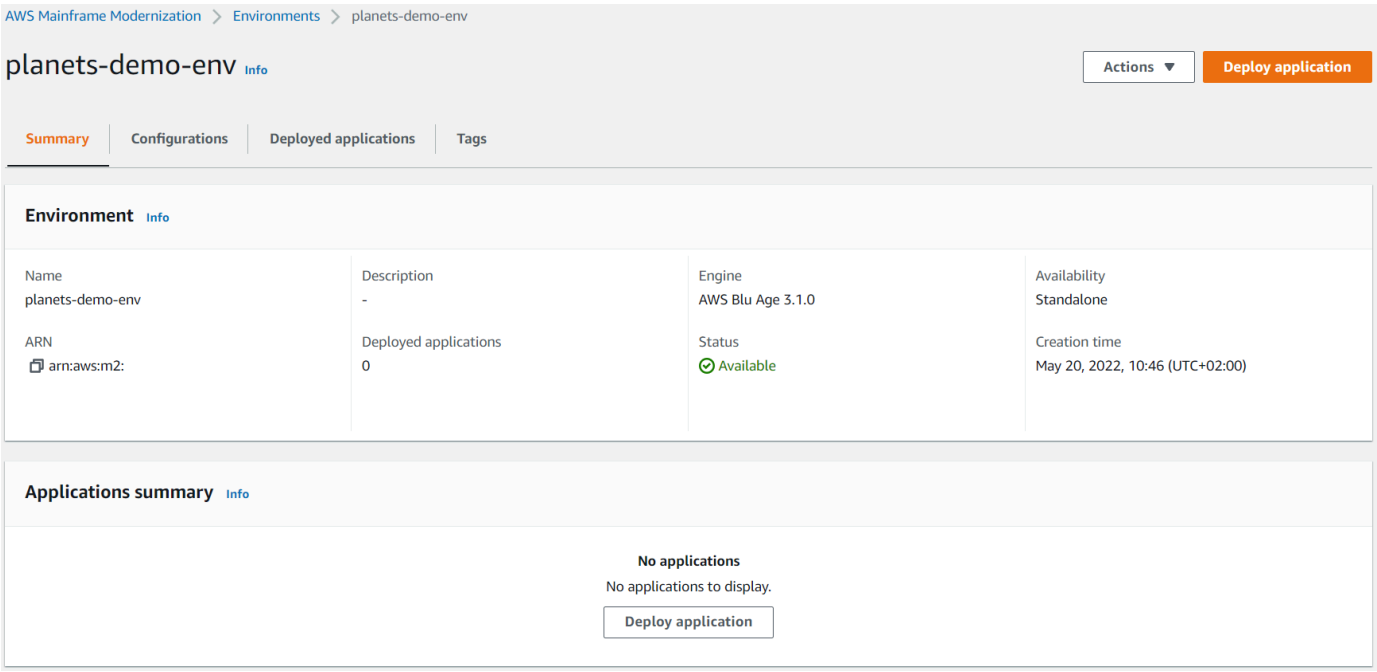
Étape 5 : Déploiement d'une application

Une fois que vous avez créé avec succès l'environnement d'exécution et l'application AWS Mainframe Modernization, et que les deux sont dans l'état Disponible, vous pouvez déployer l'application dans l'environnement d'exécution. Pour y arriver, exécutez les étapes suivantes.

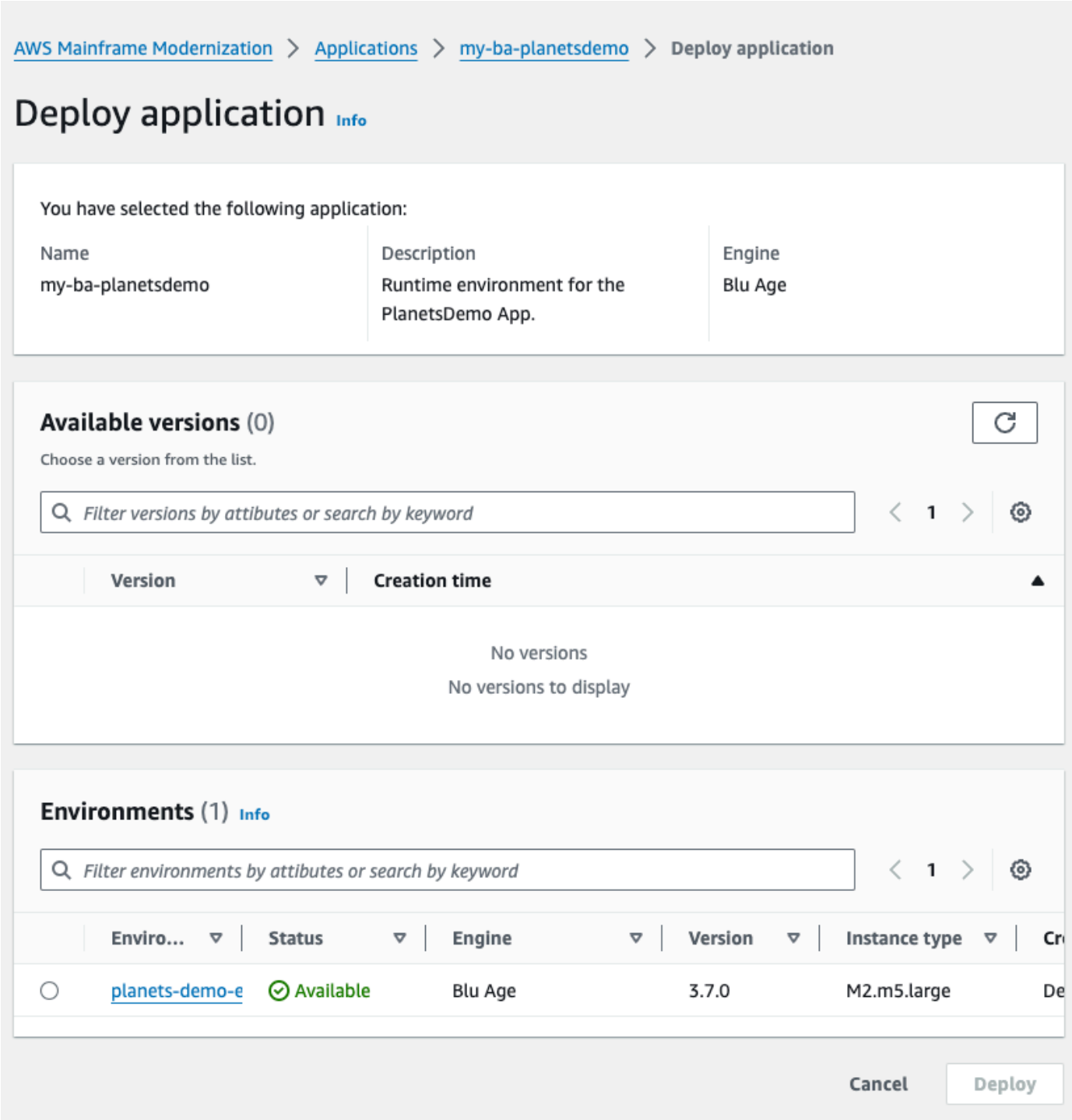
1. Accédez à la modernisation du mainframe AWS dans la console AWS de gestion. Dans le panneau de navigation, choisissez Environments (Environnements). La page de liste des environnements s'affiche.



2. Choisissez l'environnement d'exécution créé précédemment. La page des détails de l'environnement s'affiche.
3. Choisissez Déployer l'application.



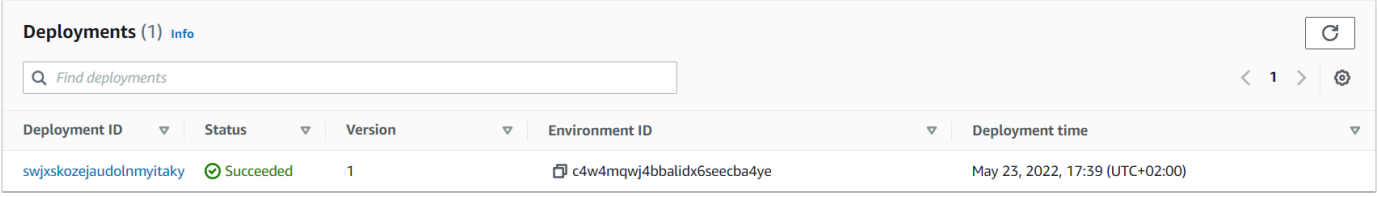
4. Choisissez l'application créée précédemment, puis la version vers laquelle vous souhaitez déployer votre application. Choisissez ensuite Deploy (Déployer).



5. Attendez que l'application ait terminé son déploiement. Vous verrez une bannière avec le message L'application a été déployée avec succès.

Étape 6 : démarrer une application

1. Accédez à AWS Mainframe Modernization dans le AWS Management Console et choisissez Applications.
2. Choisissez votre application, puis accédez à Déploiements. Le statut de la demande doit être Réussi.



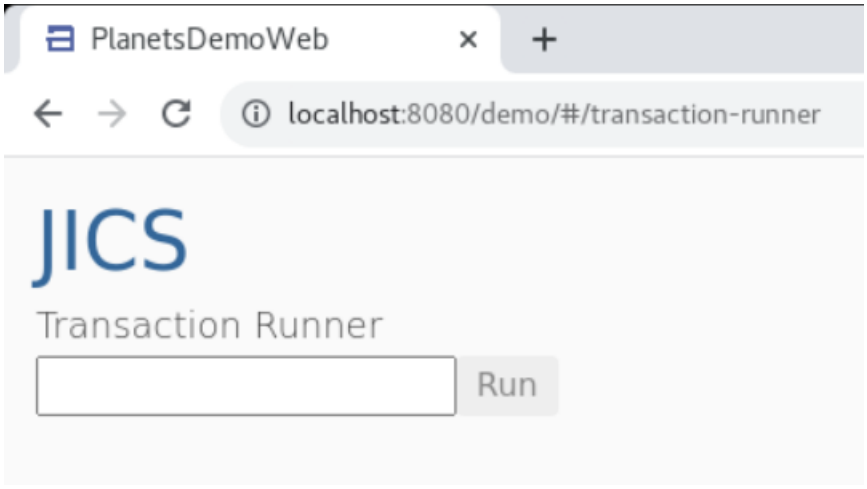
3. Choisissez Actions, puis sélectionnez Démarrer l'application.

Étape 7 : Accédez à l'application

1. Attendez que l'application soit en cours d'exécution. Vous verrez une bannière avec le message L'application a été démarrée avec succès.
2. Copiez le nom d'hôte DNS de l'application. Vous trouverez ce nom d'hôte dans la section Informations sur l'application de l'application.
3. Dans un navigateur, accédez à`http://{hostname}:{portname}/PlanetsDemo-web-1.0.0/`, où :

   - `hostname`est le nom d'hôte DNS copié précédemment.
   - `portname`est le port Tomcat défini dans la définition de l'application que vous avez créée dansÉtape 2 : Création de la définition de l'application.

   L'écran JICS apparaît.



Si vous ne parvenez pas à accéder à l'application, consultezImpossible d'accéder à l'URL d'une application.

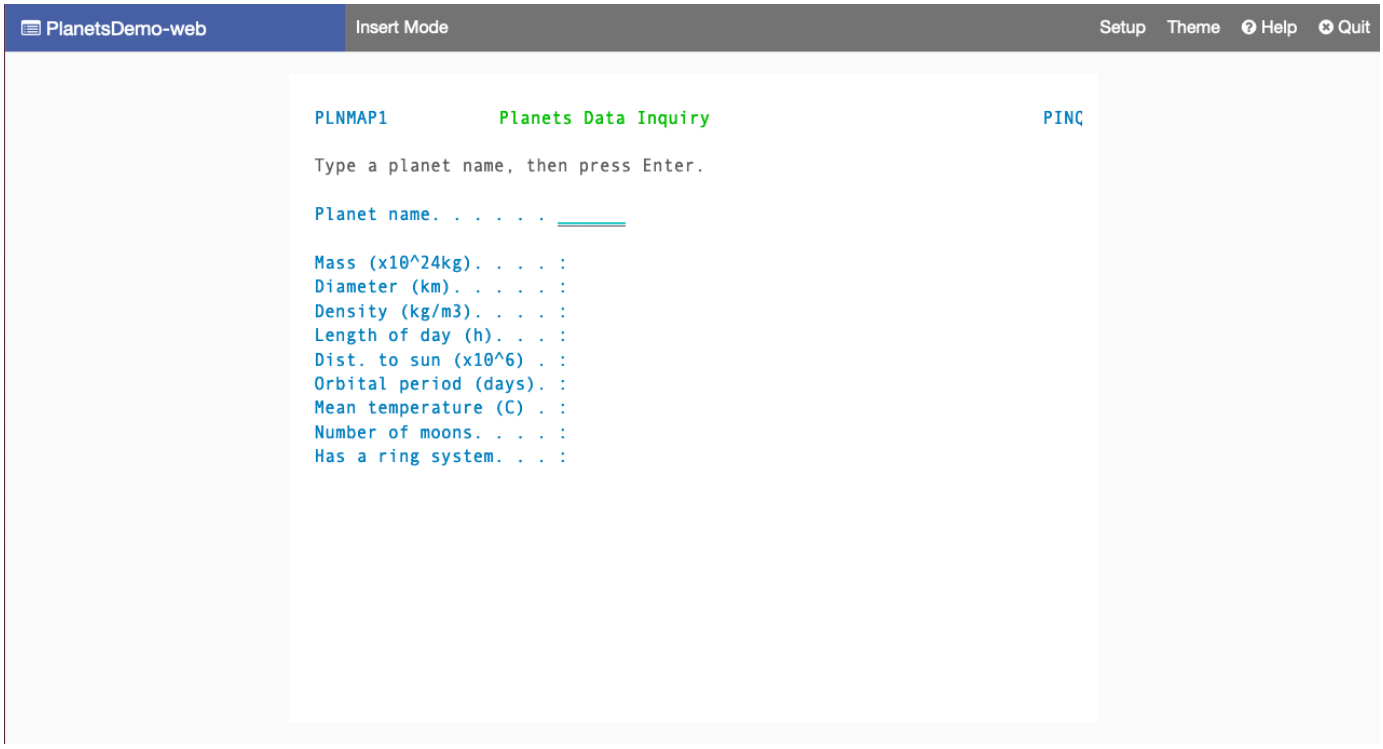
> **Note**
>
> Si l'application n'est pas accessible et que la règle entrante sur le groupe de sécurité indique que « Mon adresse IP » est sélectionnée sur le port 8196, spécifiez la règle pour autoriser le trafic en provenance de LB i/p sur le port 8196.

Étape 8 : Tester l'application

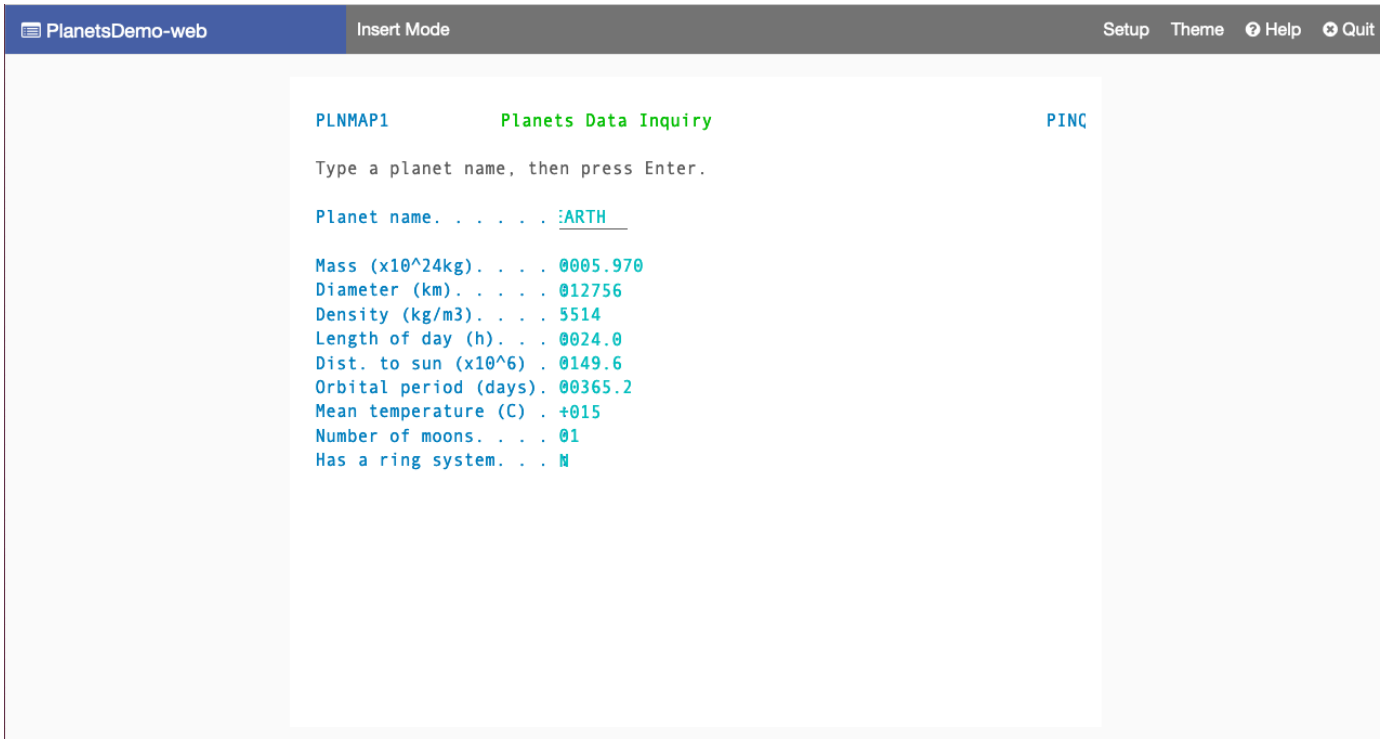
Au cours de cette étape, vous exécutez une transaction dans l'application migrée.

1. Sur l'écran JICS, entrez `PINQ` dans le champ de saisie et choisissez Exécuter (ou appuyez sur Entrée) pour démarrer la transaction de l'application.

   L'écran de l'application de démonstration devrait apparaître.



2. Tapez le nom d'une planète dans le champ correspondant et appuyez sur Entrée.



Vous devriez voir des informations sur la planète.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Si l'application AWS Mainframe Modernization est toujours en cours d'exécution, arrêtez-la.
- Supprimez l'application . Pour plus d'informations, consultez Supprimer une application de modernisation AWS du mainframe.
- Supprimez l'environnement d'exécution. Pour plus d'informations, consultez Supprimer un environnement d'exécution de modernisation du AWS mainframe.

Tutoriel : environnement d'exécution géré pour Micro Focus

Ce didacticiel explique comment déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré AWS Mainframe Modernization avec le moteur d'exécution Micro Focus. L' CardDemo exemple d'application est une application de carte de crédit simplifiée développée pour tester AWS et présenter des technologies partenaires dans le cadre de cas d'utilisation de la modernisation des ordinateurs centraux.

Dans le didacticiel, vous créez des ressources dans d'autresServices AWS. Il s'agit notamment d'Amazon Simple Storage Service, d'Amazon Relational Database Service AWS Key Management Service et. AWS Secrets Manager

Rubriques

- Nettoyage des ressources
- Étapes suivantes

Prérequis

- Assurez-vous d'avoir accès à un émulateur 3270 pour utiliser la connexion CICS. Des émulateurs 3270 gratuits et d'essai sont disponibles sur des sites Web tiers. Vous pouvez également démarrer une instance AWS Mainframe Modernization AppStream 2.0 Micro Focus et utiliser l'émulateur Rumba 3270 (non disponible gratuitement).

  Pour plus d'informations sur la AppStream version 2.0, consultezthe section called "Tutoriel : Configuration de la AppStream version 2.0 pour Enterprise Analyzer et Enterprise Developer".

  > **Note**
  >
  > Lors de la création de la pile, choisissez l'option Enterprise Developer (ED) et non Enterprise Analyzer (EA).

- Téléchargez l'CardDemo exemple d'application et décompressez le fichier téléchargé dans n'importe quel répertoire local. Ce répertoire contiendra un sous-répertoire intitulé`CardDemo`.
- Identifiez un VPC dans votre compte dans lequel vous pourrez définir les ressources créées dans ce didacticiel. Le VPC aura besoin de sous-réseaux dans au moins deux zones de disponibilité. Pour plus d'informations sur Amazon VPC, consultez Comment fonctionne Amazon VPC.

Étape 1 : créer et charger un compartiment Amazon S3

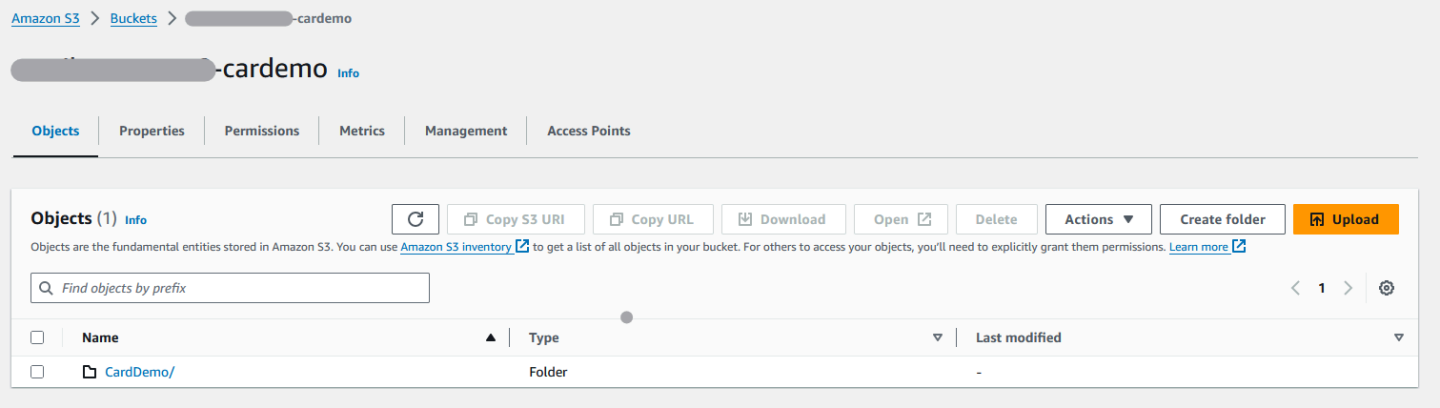
Au cours de cette étape, vous créez un compartiment Amazon S3 et chargez CardDemo des fichiers dans ce compartiment. Plus loin dans ce didacticiel, vous utiliserez ces fichiers pour déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré par Micro Focus AWS Mainframe Modernization.

> **Note**
>
> Il n'est pas nécessaire de créer un nouveau compartiment S3, mais le compartiment que vous choisissez doit se trouver dans la même région que les autres ressources utilisées dans ce didacticiel.

Pour créer un compartiment Amazon S3

1. Ouvrez la console Amazon S3 et choisissez Create bucket.
2. Dans Configuration générale, choisissez la région AWS dans laquelle vous souhaitez créer le AWS Mainframe Modernization Micro Focus Managed Runtime.
3. Entrez un nom de compartiment, par exemple,`yourname-aws-region-carddemo`. Conservez les paramètres par défaut, puis choisissez Create bucket. Vous pouvez également copier les paramètres d'un compartiment Amazon S3 existant, puis choisir Create bucket.
4. Choisissez le bucket que vous venez de créer, puis choisissez Upload.
5. Dans la section Télécharger, choisissez Ajouter un dossier, puis accédez au `CardDemo` répertoire depuis votre ordinateur local.
6. Choisissez Upload pour démarrer le processus de téléchargement. Les temps de téléchargement varient en fonction de la vitesse de votre connexion.
7. Lorsque le téléchargement est terminé, vérifiez que tous les fichiers ont été correctement chargés, puis choisissez Fermer.

Votre compartiment Amazon S3 contient désormais le `CardDemo` dossier.



Pour plus d'informations sur les compartiments S3, consultez Création, configuration et utilisation des compartiments Amazon S3.

Étape 2 : Création et configuration d'une base de données

Au cours de cette étape, vous allez créer une base de données PostgreSQL dans Amazon Relational Database Service (Amazon RDS). Pour le didacticiel, cette base de données contient les ensembles de données que l' CardDemo exemple d'application utilise pour les tâches des clients concernant les transactions par carte de crédit.

Pour créer une base de données dans Amazon RDS

1. Ouvrez la console Amazon RDS.
2. Choisissez la région AWS dans laquelle vous souhaitez créer l'instance de base de données.
3. Dans le volet de navigation, sélectionnez Databases (Bases de données).
4. Choisissez Créer une base de données, puis sélectionnez Création standard.
5. Pour Type de moteur, choisissez PostgreSQL.
6. Choisissez une version du moteur 15 ou supérieure.

> **Note**
>
> Enregistrez la version du moteur car vous en aurez besoin plus tard dans ce didacticiel.

7. Dans Modèles, choisissez Offre gratuite.
8. Remplacez l'identifiant de l'instance de base de données par quelque chose de significatif, par exemple`MicroFocus-Tutorial`.
9. Abstenez-vous de gérer les informations d'identification principales dansAWS Secrets Manager. Entrez plutôt un mot de passe principal et confirmez-le.

> **Note**
>
> Enregistrez le nom d'utilisateur et le mot de passe que vous utilisez pour la base de données. Vous les stockerez en toute sécurité dans les prochaines étapes de ce didacticiel.

10. Sous Connectivité, choisissez le VPC sur lequel vous souhaitez créer l'environnement d'exécution géré AWS Mainframe Modernization.
11. Choisissez Create database (Créer une base de données).

Pour créer un groupe de paramètres personnalisé dans Amazon RDS

1. Dans le volet de navigation de la console Amazon RDS, sélectionnez Groupes de paramètres, puis sélectionnez Créer un groupe de paramètres.
2. Dans la fenêtre Créer un groupe de paramètres, pour Famille de groupes de paramètres, sélectionnez l'option Postgres correspondant à la version de votre base de données.

> **Note**
>
> Certaines versions de Postgres nécessitent un type. Sélectionnez le groupe de paramètres de base de données si nécessaire. Entrez un nom de groupe et une description pour le groupe de paramètres.

3. Choisissez Créer.

Pour configurer le groupe de paramètres personnalisé

1. Choisissez le groupe de paramètres nouvellement créé.
2. Sélectionnez Actions, puis Edit (Modifier).
3. Filtrez `max_prepared_transactions` et remplacez la valeur du paramètre par 100.
4. Choisissez Save Changes (Enregistrer les modifications).

Pour associer le groupe de paramètres personnalisés à la base de données

1. Dans le volet de navigation de la console Amazon RDS, choisissez Databases, puis choisissez l'instance de base de données que vous souhaitez modifier.
2. Sélectionnez Modifier. La page Modifier l'instance de base de données s'affiche.

> **Note**
>
> L'option Modifier n'est pas disponible tant que la création et la sauvegarde de la base de données ne sont pas terminées, ce qui peut prendre plusieurs minutes.

3. Sur la page Modifier une instance de base de données, accédez à Configuration supplémentaire et remplacez le groupe de paramètres de base de données par votre groupe de paramètres. Si votre groupe de paramètres n'est pas disponible dans la liste, vérifiez s'il a été créé avec la bonne version de base de données.
4. Choisissez Continuer, puis consultez le résumé des modifications.
5. Choisissez Appliquer immédiatement pour appliquer les modifications instantanément.
6. Choisissez Modifier l'instance de base de données pour enregistrer vos modifications.

Pour plus d'informations sur les groupes de paramètres, consultez la section Utilisation des groupes de paramètres.

> **Note**
>
> Vous pouvez également utiliser une base de données Amazon Aurora PostgreSQL AWS avec Mainframe Modernization, mais il n'existe pas d'option de niveau gratuit. Pour plus d'informations, consultez la section Utilisation d'Amazon Aurora PostgreSQL.

Étape 3 : Création et configuration d'un AWS KMS key

Pour stocker les informations d'identification de manière sécurisée pour l'instance Amazon RDS, créez d'abord unAWS KMS key.

Pour créer un AWS KMS key

1. Ouvrez la console du service de gestion des clés.
2. Choisissez Create key (Créer une clé).
3. Conservez les valeurs par défaut Symetric pour le type de clé et Chiffrer et déchiffrer pour l'utilisation des clés.
4. Choisissez Suivant.
5. Donnez à la clé un alias `MicroFocus-Tutorial-RDS-Key` et une description facultative.
6. Choisissez Suivant.
7. Désignez un administrateur clé en cochant la case à côté de votre utilisateur ou de votre rôle.
8. Choisissez Next, puis de nouveau Next.
9. Sur l'écran de révision, modifiez la politique clé, puis entrez les informations suivantes :

```
{
     "Sid" : "Allow access for Mainframe Modernization Service",
     "Effect" : "Allow",
        "Principal" : {
           "Service" : "m2.amazonaws.com"
                 },
      "Action" : "kms:Decrypt",
      "Resource" : "*"
},
```

Cette politique accorde à AWS Mainframe Modernization des autorisations de déchiffrement à l'aide de cette politique clé spécifique.

10. Choisissez Terminer pour créer la clé.

Pour plus d'informations, consultez la section Création de clés dans le guide du AWS Key Management Service développeur.

Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données

Stockez maintenant les informations d'identification de la base de données en toute sécurité à l'aide du AWS Secrets Manager etAWS KMS key.

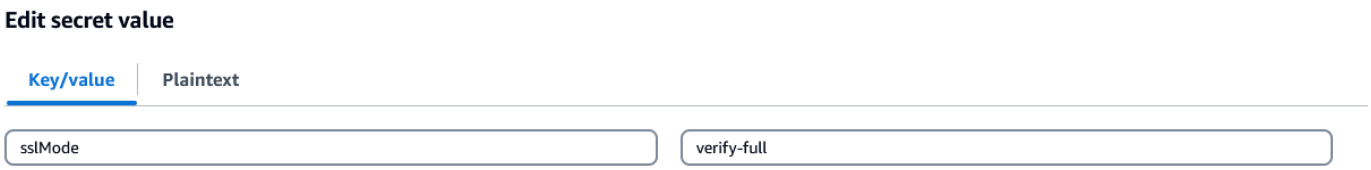
Pour créer et configurer un secret AWS Secrets Manager de base de données

1. Ouvrez la console Secrets Manager.
2. Dans le volet de navigation, sélectionnez Secrets.
3. Dans Secrets, choisissez Enregistrer un nouveau secret.
4. Définissez le type de secret sur Identifiants pour la base de données Amazon RDS.
5. Entrez les informations d'identification que vous avez spécifiées lors de la création de la base de données.
6. Sous Clé de chiffrement, sélectionnez la clé que vous avez créée à l'étape 3.
7. Dans la section Base de données, sélectionnez la base de données que vous avez créée pour ce didacticiel, puis choisissez Next.
8. Sous Nom secret, entrez un nom tel qu'`MicroFocus-Tutorial-RDS-Secret`une description facultative.
9. Dans la section Autorisations relatives aux ressources, choisissez Modifier les autorisations et remplacez le contenu par la politique suivante :

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Principal" : {
```

```
                "Service" : "m2.amazonaws.com"
            },
            "Action" : "secretsmanager:GetSecretValue",
            "Resource" : "*"
        }
    ]
}
```

10. Choisissez Enregistrer.
11. Choisissez Next pour les écrans suivants, puis sélectionnez Store. Actualisez la liste des secrets pour voir le nouveau secret.
12. Choisissez le secret nouvellement créé et notez-le `Secret` ARN car vous en aurez besoin plus tard dans le didacticiel.
13. Dans l'onglet Vue d'ensemble du secret, choisissez Extraire la valeur du secret.
14. Choisissez Modifier, puis Ajouter une ligne.
15. Ajoutez une clé pour `sslMode` avec une valeur de `verify-full` :

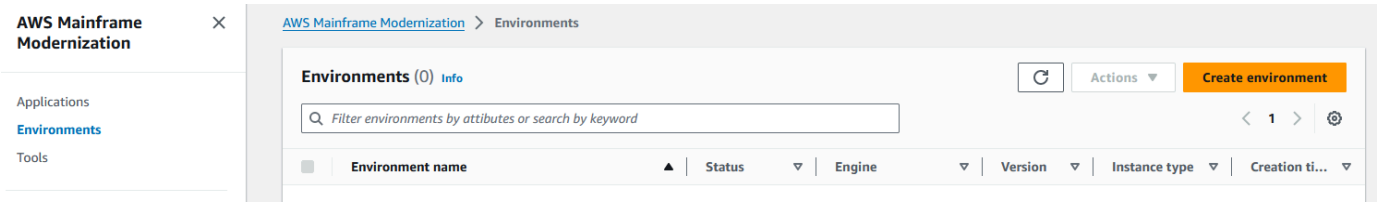


16. Choisissez Enregistrer.

Étape 5 : Création d'un environnement d'exécution

Pour créer un environnement d'exécution

1. Ouvrez la console de modernisation AWS du mainframe.
2. Dans le panneau de navigation, choisissez Environments (Environnements). Choisissez ensuite Create environment.



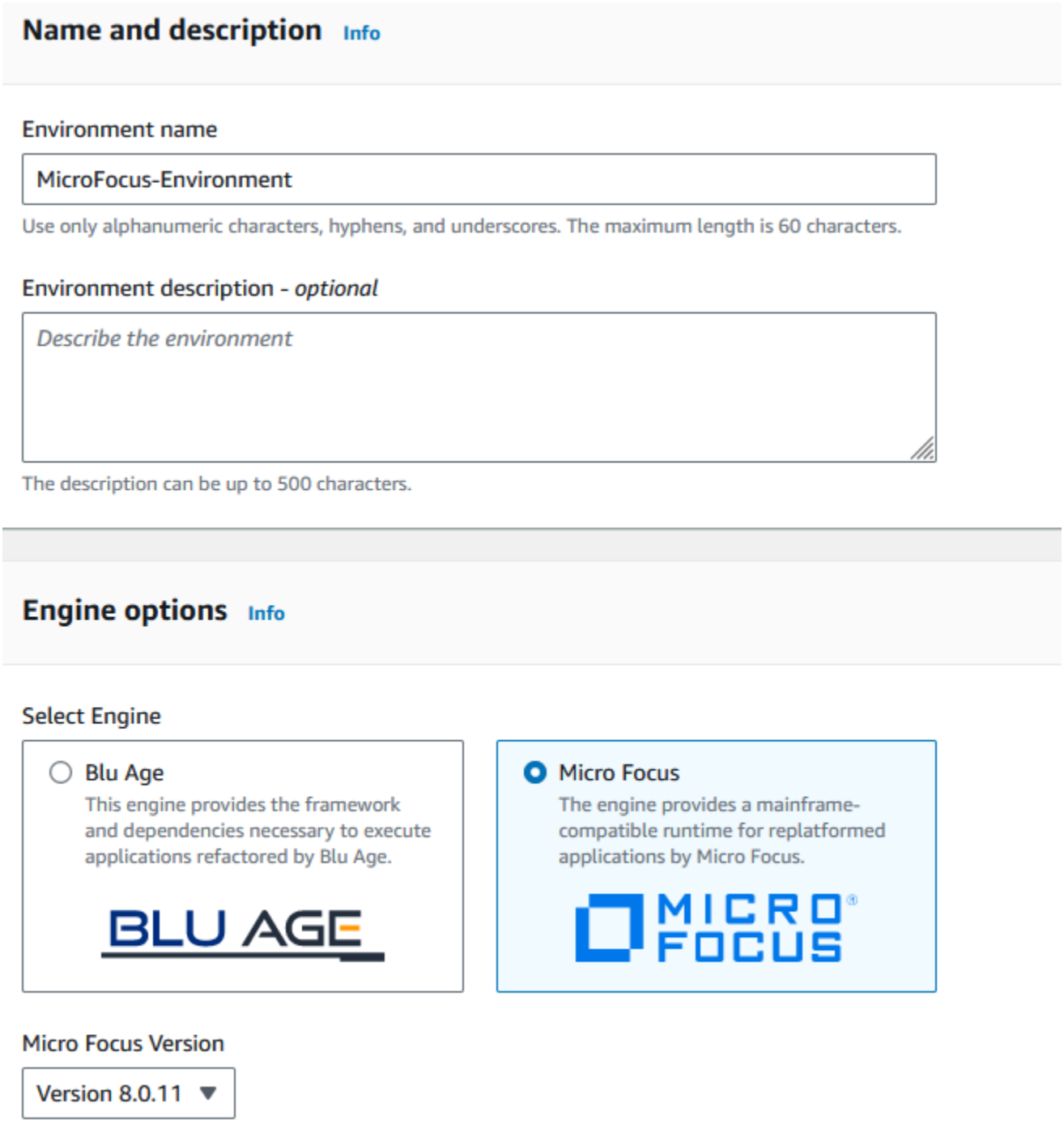
3. Sous Spécifier les informations de base,

    a. Entrez `MicroFocus-Environment` le nom de l'environnement.

b. Dans les options du moteur, assurez-vous que Micro Focus est sélectionné.

c. Choisissez la dernière version de Micro Focus.

d. Choisissez Suivant.



4. Configuration de l'environnement

a. Sous Disponibilité, choisissez High Availability Cluster.

b. Sous Ressources, choisissez l'un `M2.c5.large` ou `M2.m5.large` l'autre type d'instance, ainsi que le nombre d'instances que vous souhaitez. Spécifiez jusqu'à deux instances.

c. Sous Sécurité et réseau, choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public et choisissez au moins deux sous-réseaux publics.

d. Choisissez Suivant.

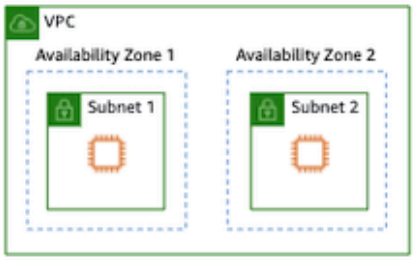
Specify configurations Info

Availability Info

Choose the availability pattern for your environment.

- Standalone runtime environment
  Sets up a single instance in a single availability zone. Does not guarantee high availability but costs less.

- High availability cluster
  Sets up redundant instances across two availability zones. Enables higher availability but costs more.



Resources

Instance type
Choose the instance type for your high availability cluster.

M2.m5.large

Desired capacity
Specify the desired number of instances.

2

Security and network

Allow applications deployed to this environment to be publicly accessible.

Virtual Private Cloud (VPC)
Choose the VPC where you want to create the environment.

Default vpc-15

Subnets
Choose one or more subnets for a high availability setup.

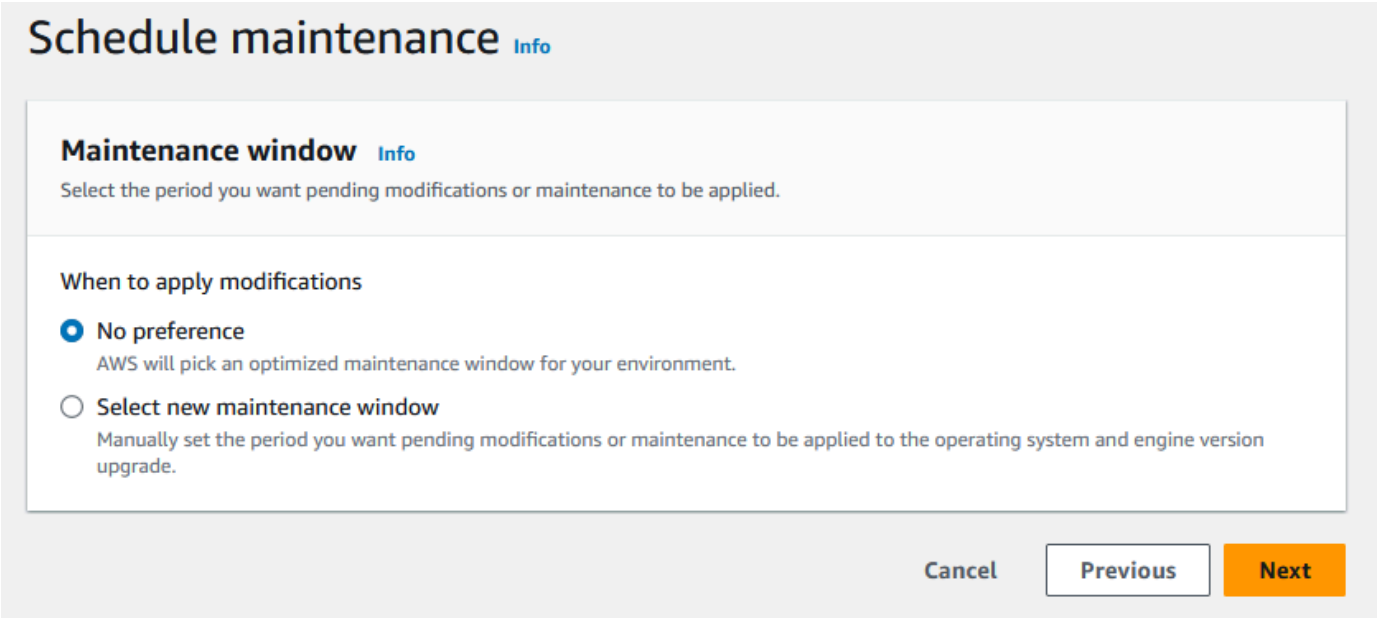
Choose subnets

subnet-56f1a | us-west-2a

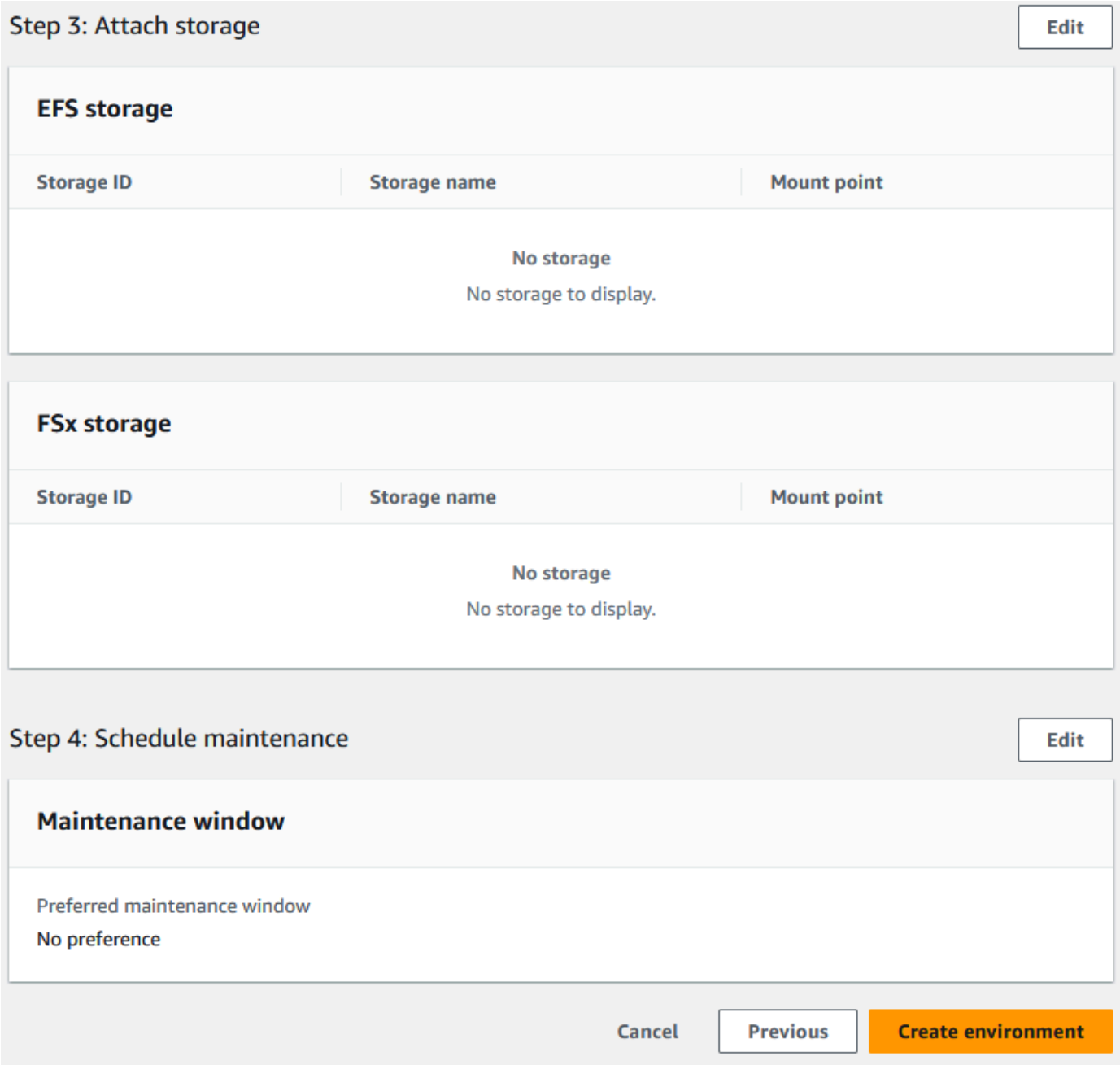
subnet-6685 | us-west-2b

Security groups
Choose one or more security groups for the chosen VPC.

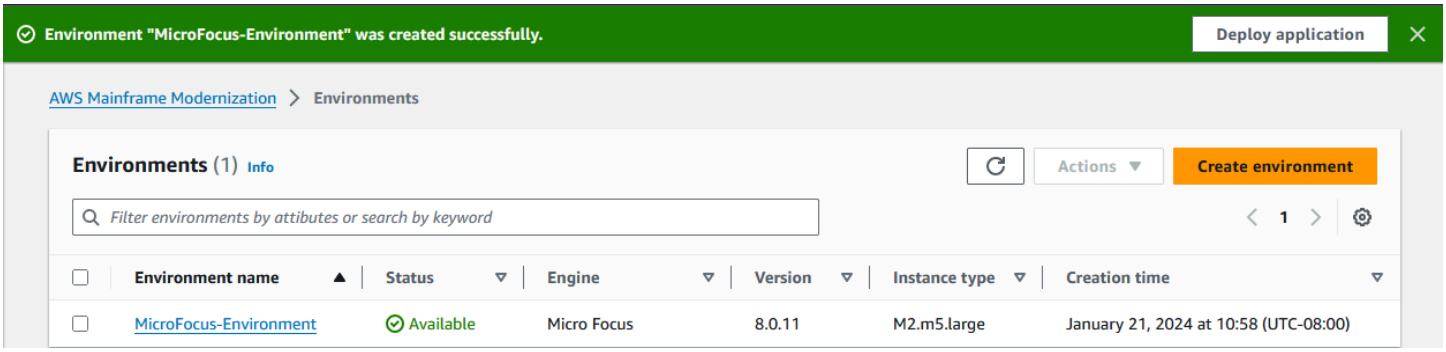
5. Sur la page Joindre un espace de stockage, choisissez Next.
6. Sur la page Planifier la maintenance, choisissez Aucune préférence, puis cliquez sur Suivant.



7. Sur la page Réviser et créer, passez en revue toutes les configurations que vous avez fournies pour l'environnement d'exécution, puis choisissez Créer un environnement.



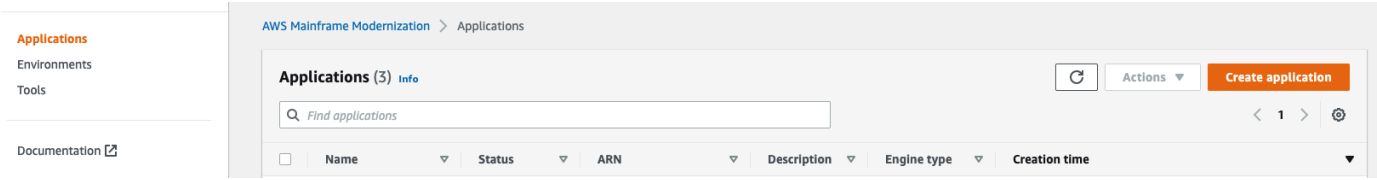
Lorsque vous avez créé votre environnement, une bannière indiquant « `Environment` *name* `was created successfully` Disponible » apparaît et le champ État devient « Disponible ». Le processus de création de l'environnement prend plusieurs minutes, mais vous pouvez passer aux étapes suivantes pendant son exécution.



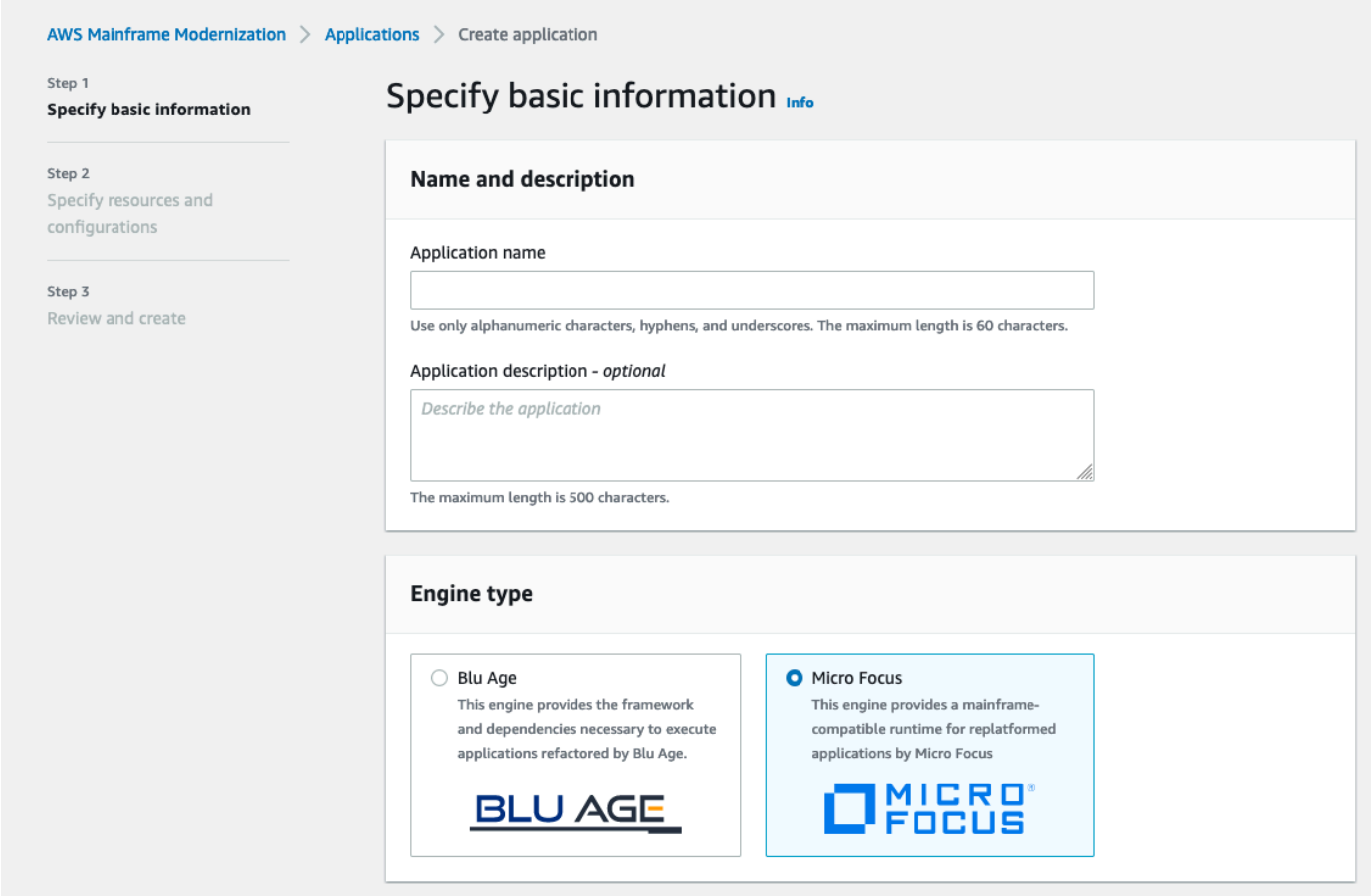
Étape 6 : Création d'une application

Pour créer une application

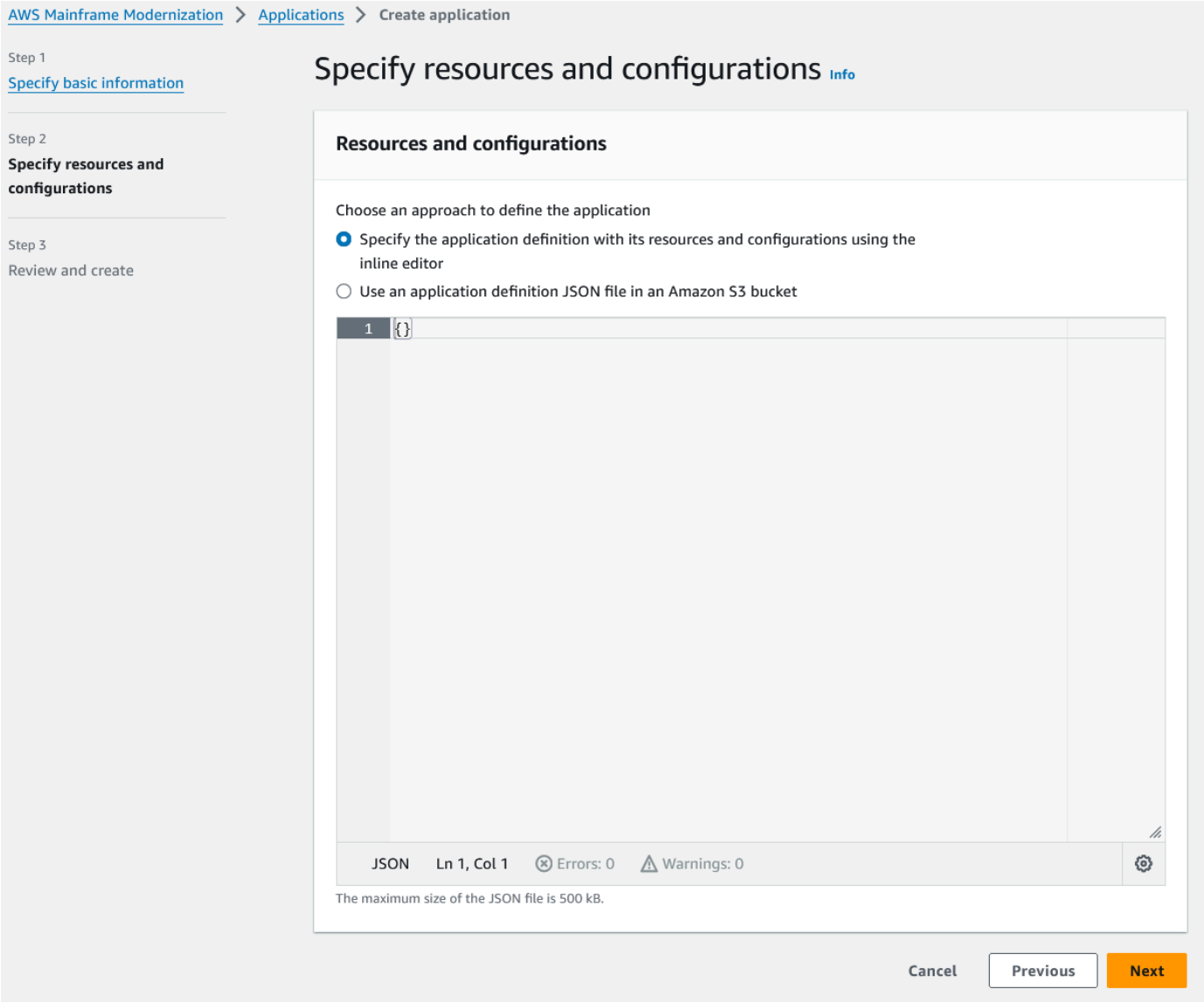
1. Dans le volet de navigation, choisissez Applications. Choisissez ensuite Créer une application.



2. Sur la page Créer une application, sous Spécifier les informations de base, entrez `MicroFocus-CardDemo` le nom de l'application et sous Type de moteur, assurez-vous que Micro Focus est sélectionné. Ensuite, sélectionnez Suivant.



3. Sous Spécifier les ressources et les configurations, choisissez l'option permettant de spécifier la définition de l'application avec ses ressources et ses configurations à l'aide de l'éditeur en ligne.



Entrez la définition d'application suivante dans l'éditeur :

```
{
  "template-version": "2.0",
  "source-locations": [
    {
      "source-id": "s3-source",
      "source-type": "s3",
      "properties": {
        "s3-bucket": "yourname-aws-region-carddemo",
        "s3-key-prefix": "CardDemo"
      }
    }
```

```
  ],
  "definition": {
    "listeners": [
      {
        "port": 6000,
        "type": "tn3270"
      }
    ],
    "dataset-location": {
      "db-locations": [
        {
          "name": "Database1",
          "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx"
        }
      ]
    },
    "batch-settings": {
      "initiators": [
        {
          "classes": [
               "A",
               "B"
            ],
          "description": "initiator_AB...."
        },
        {
          "classes": [
                "C",
                "D"
             ],
          "description": "initiator_CD...."
        }
      ],
      "jcl-file-location": "${s3-source}/catalog/jcl"
    },
    "cics-settings": {
      "binary-file-location": "${s3-source}/loadlib",
      "csd-file-location": "${s3-source}/rdef",
      "system-initialization-table": "CARDSIT"
    },
    "xa-resources": [
      {
```

```
        "name": "XASQL",
        "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx",
        "module": "${s3-source}/xa/ESPGSQLXA64.so"
      }
    ]
  }
}
```

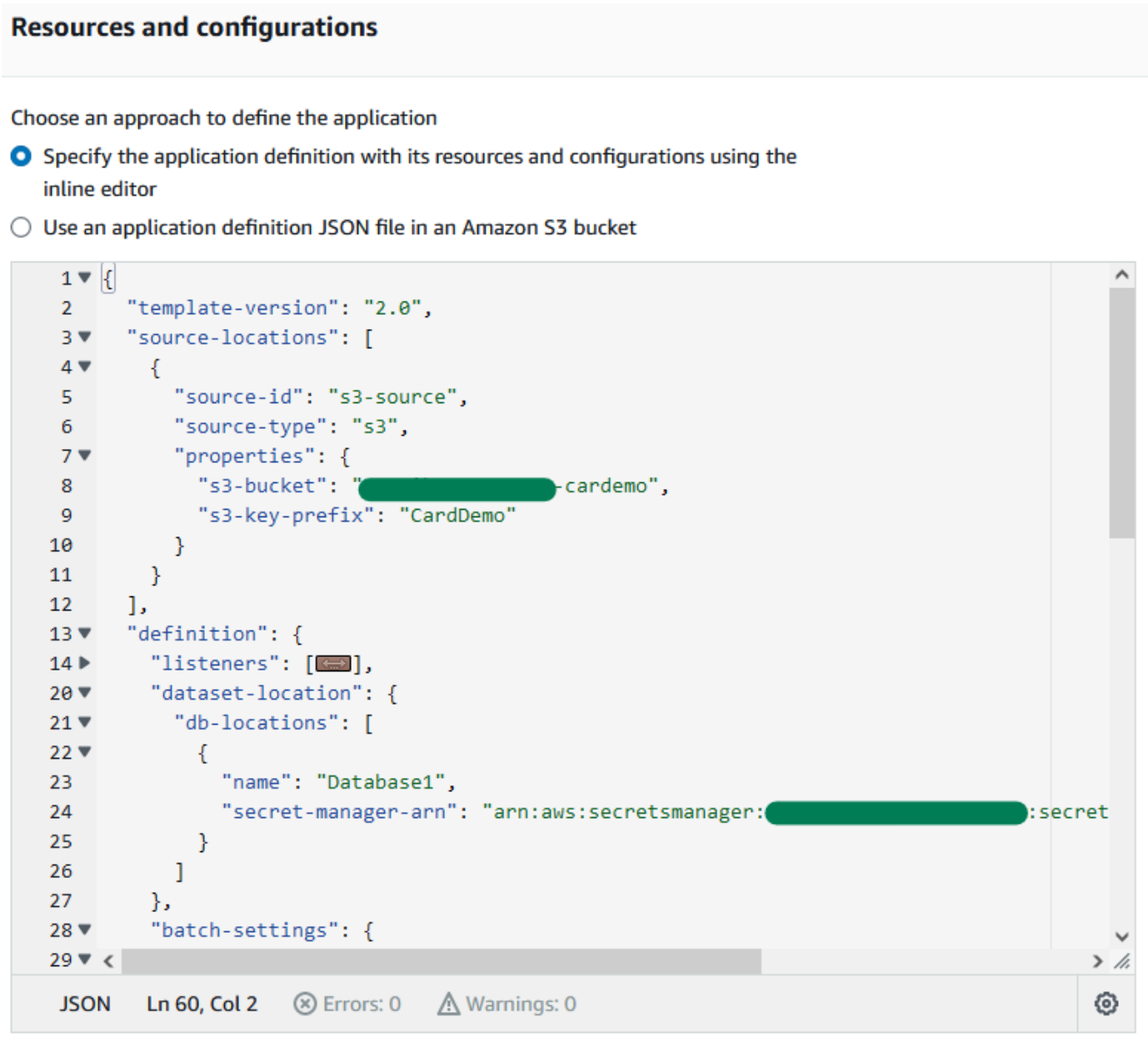
> **Note**
>
> Ce fichier est sujet à modification.

4. Modifiez le JSON de l'application dans l'objet de propriétés de source-locations comme suit :

   a. Remplacez la valeur pour `s3_bucket` par le nom du compartiment Amazon S3 que vous avez créé à l'étape 1.

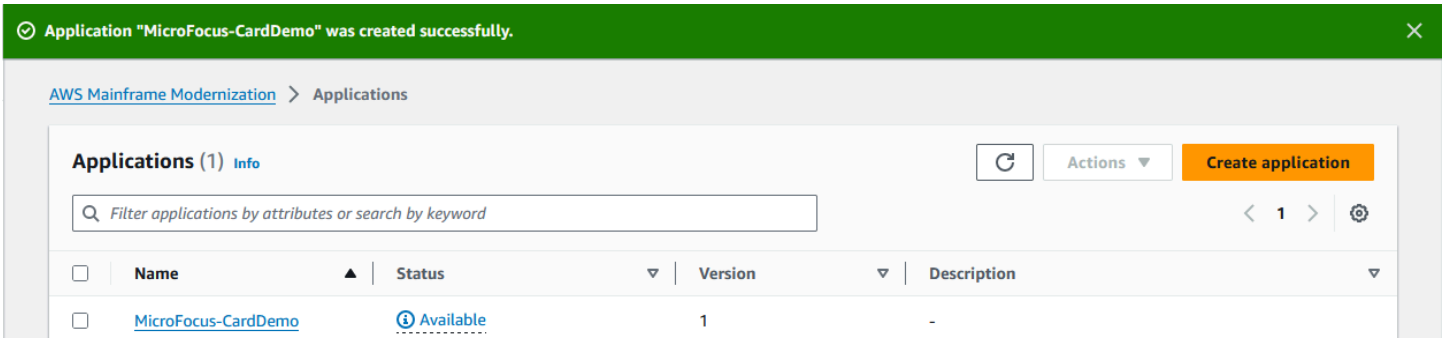
   b. Remplacez la valeur pour `s3-key-prefix` par le dossier (préfixe clé) dans lequel vous avez chargé les fichiers CardDemo d'exemple. Si vous avez chargé le `CardDemo` répertoire directement dans un compartiment Amazon S3, il `s3-key-prefix` n'est pas nécessaire de le modifier.

   c. Remplacez les deux `secret-manager-arn` valeurs par l'ARN du secret de base de données que vous avez créé à l'étape 4.



Pour plus d'informations sur la définition de l'application, consultezDéfinition de l'application Micro Focus.

5. Choisissez Next (Suivant) pour continuer.
6. Sur la page Réviser et créer, passez en revue les informations que vous avez fournies, puis choisissez Créer une application.

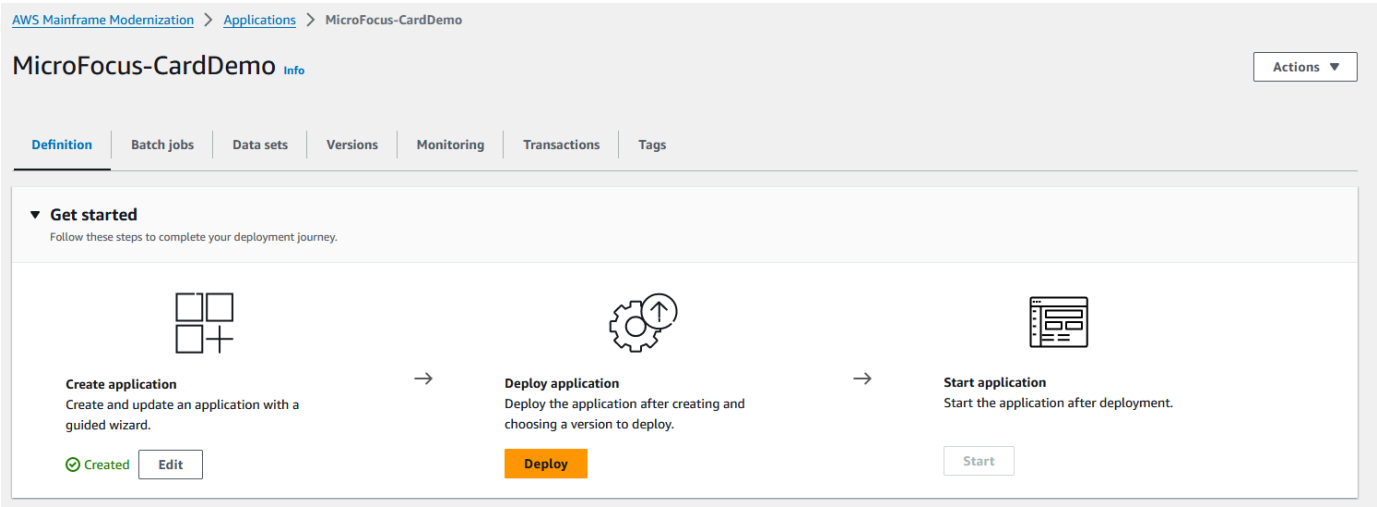


Lorsque vous avez créé votre application, une bannière apparaît indiquant`Application` *`name`* `was created successfully`. Et le champ État devient Disponible.

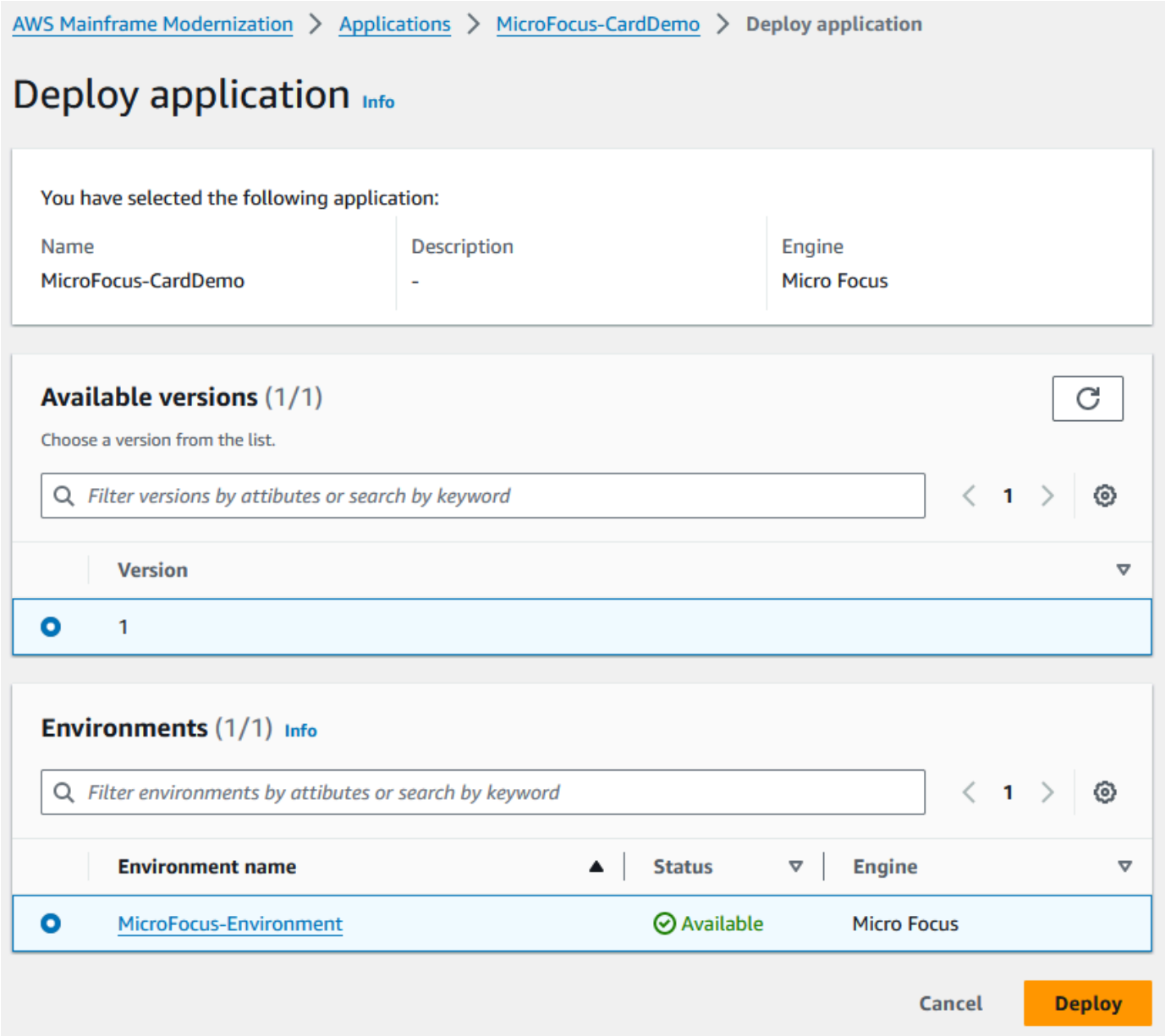
Étape 7 : Déploiement d'une application

Pour déployer une application

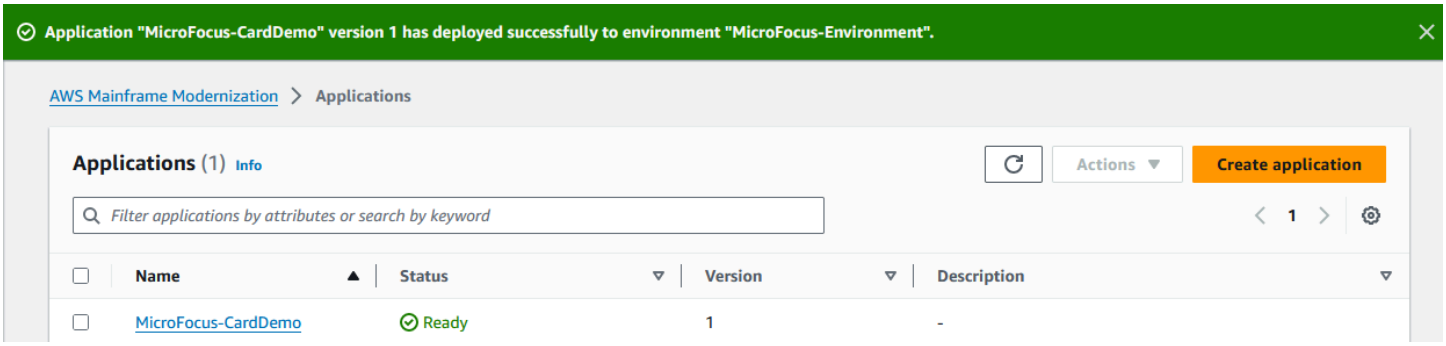
1. Dans le volet de navigation, choisissez Applications, puis choisissez`MicroFocus-CardDemo`.
2. Sous Déployer l'application, choisissez Déployer.



3. Choisissez la dernière version de l'application et l'environnement que vous avez créés précédemment, puis choisissez Déployer.



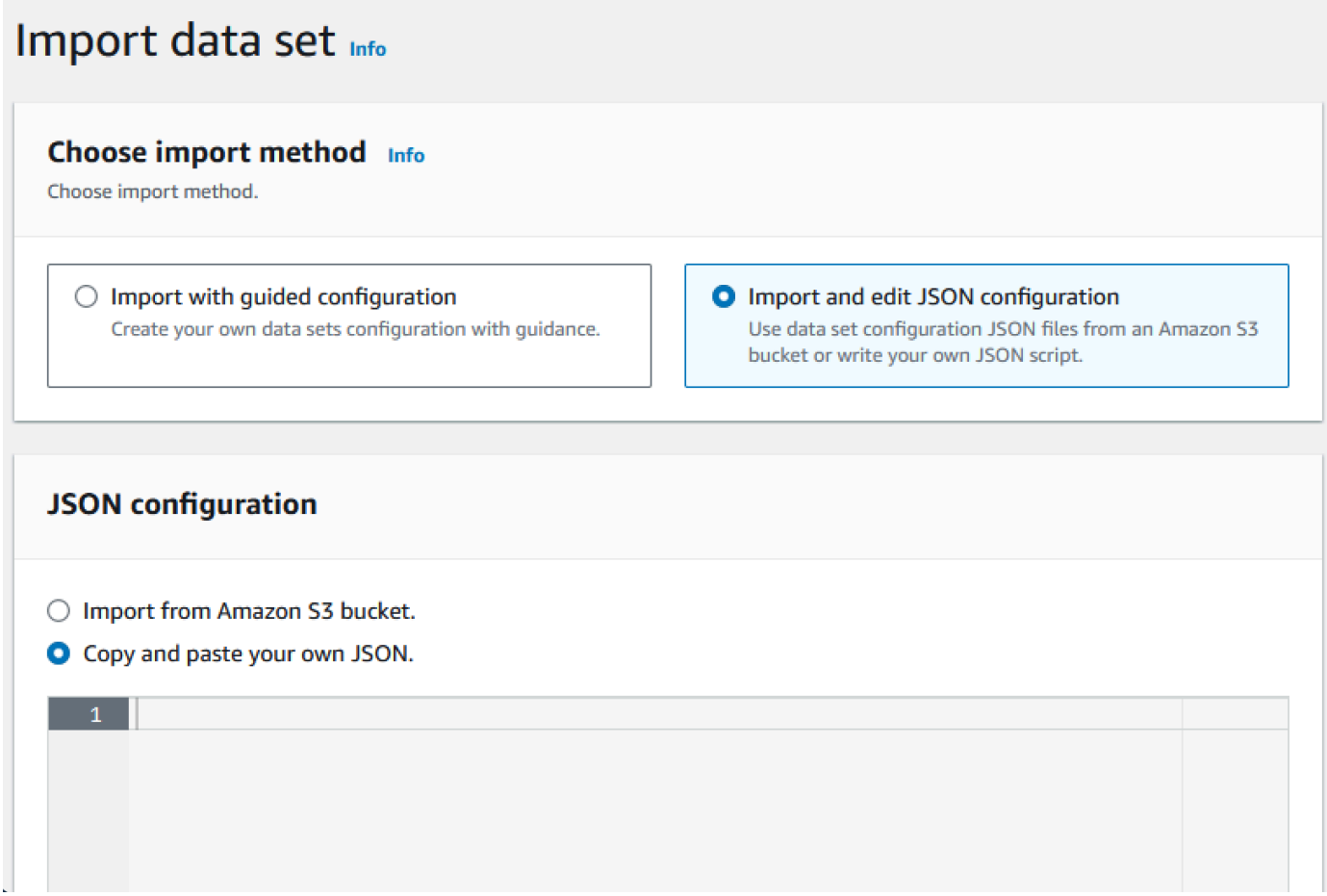
Lorsque l' CardDemo application est déployée avec succès, le statut passe à Prêt.



Étape 8 : Importer des ensembles de données

Pour importer des ensembles de données

1. Dans le volet de navigation, choisissez Applications, puis choisissez l'application.
2. Choisissez l'onglet Ensembles de données. Choisissez ensuite Import (Importer).
3. Choisissez Importer et modifier la configuration JSON, puis choisissez l'option Copier et collez votre propre JSON.



4. Copiez et collez le code JSON suivant, mais ne choisissez pas encore « Soumettre ». Ce JSON contient tous les ensembles de données requis pour l'application de démonstration, mais nécessite les détails de votre compartiment Amazon S3.

```
{
    "dataSets": [
        {
            "dataSet": {
                "storageType": "Database",
```

```
                "datasetName": "AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 300,
                    "max": 300
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 16
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
```

```
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
```

```
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 9,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 500,
                    "max": 500
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
```

```
                        "primaryKey": {
                            "length": 11,
                            "offset": 25
                        }
                    }
                },
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 350,
                    "max": 350
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
```

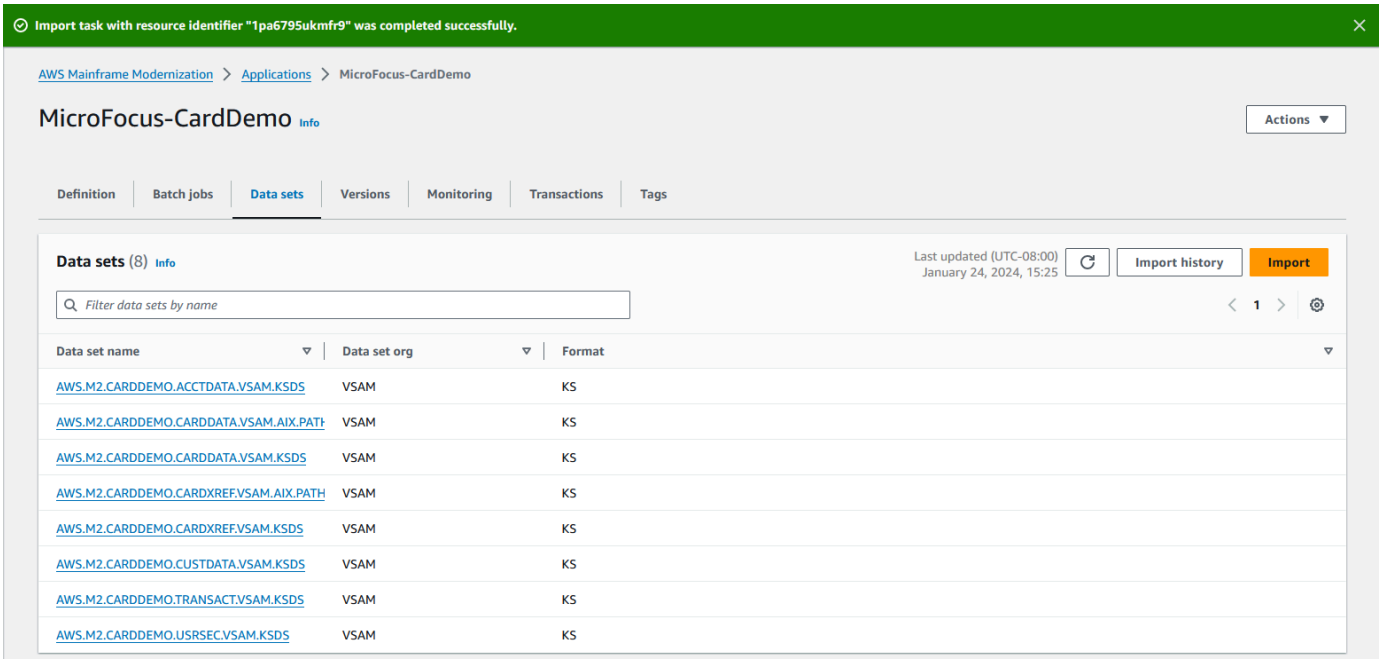
```
                "datasetName": "AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 8,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 80,
                    "max": 80
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS.DAT"
            }
        }
    ]
}
```

5. Remplacez chaque occurrence de `<s3-bucket-name>` (il y en a huit) par le nom du compartiment Amazon S3 qui contient le CardDemo dossier, par exemple,`your-name-aws-region-carddemo`.

   > **Note**
   >
   > Pour copier l'URI Amazon S3 du dossier dans Amazon S3, sélectionnez le dossier, puis choisissez Copier l'URI Amazon S3.

6. Sélectionnez Envoyer.

   Lorsque l'importation est terminée, une bannière apparaît avec le message suivant : `Import task with resource identifier` *`name`* `was completed successfully.` La liste des ensembles de données importés s'affiche.

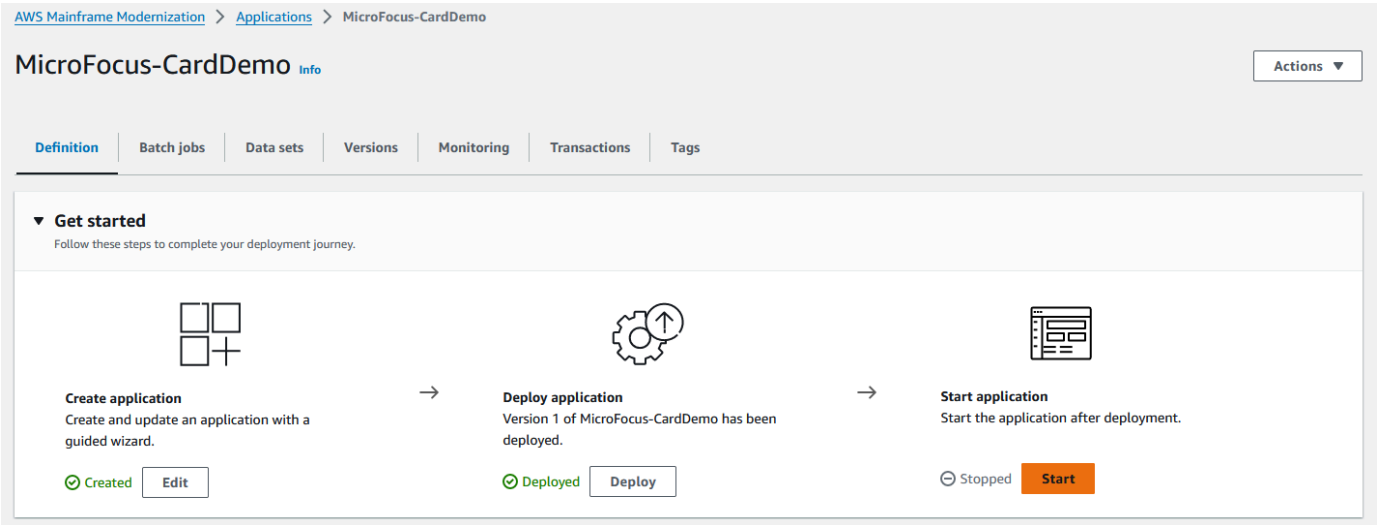


Vous pouvez également consulter l'état de toutes les importations de jeux de données en choisissant Historique des importations dans l'onglet Ensembles de données.

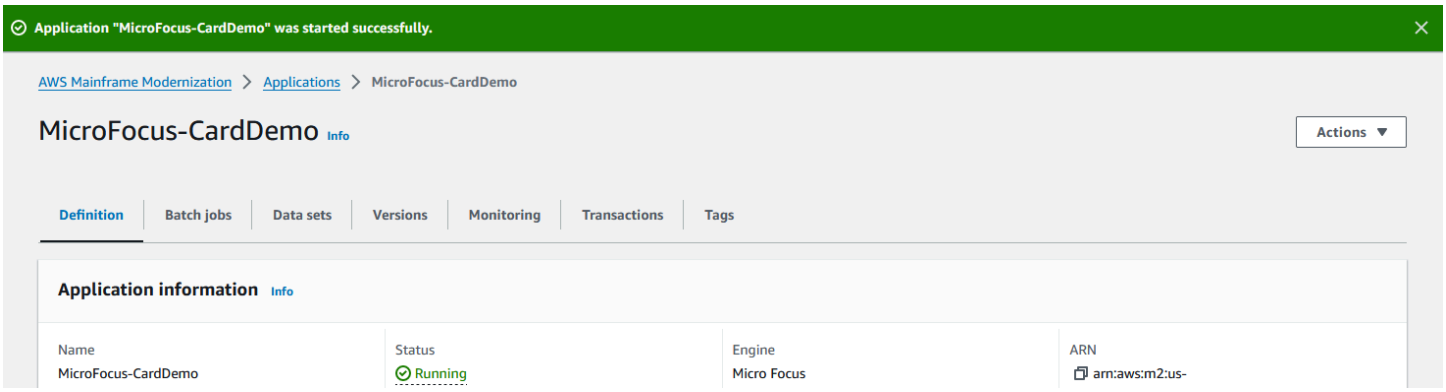
Étape 9 : démarrer une application

Pour démarrer une application

1. Dans le volet de navigation, choisissez Applications, puis choisissez l'application.
2. Choisissez Démarrer l'application.



Lorsque l' CardDemo application démarre avec succès, une bannière apparaît avec le message suivant : `Application` *`name`* `was started successfully` Le champ Status devient Running.



Étape 10 : Connectez-vous à l'application CardDemo CICS

Avant de vous connecter, assurez-vous que le VPC et le groupe de sécurité que vous avez spécifiés pour l'application sont les mêmes que ceux que vous avez demandés pour l'interface réseau à partir de laquelle vous allez vous connecter.

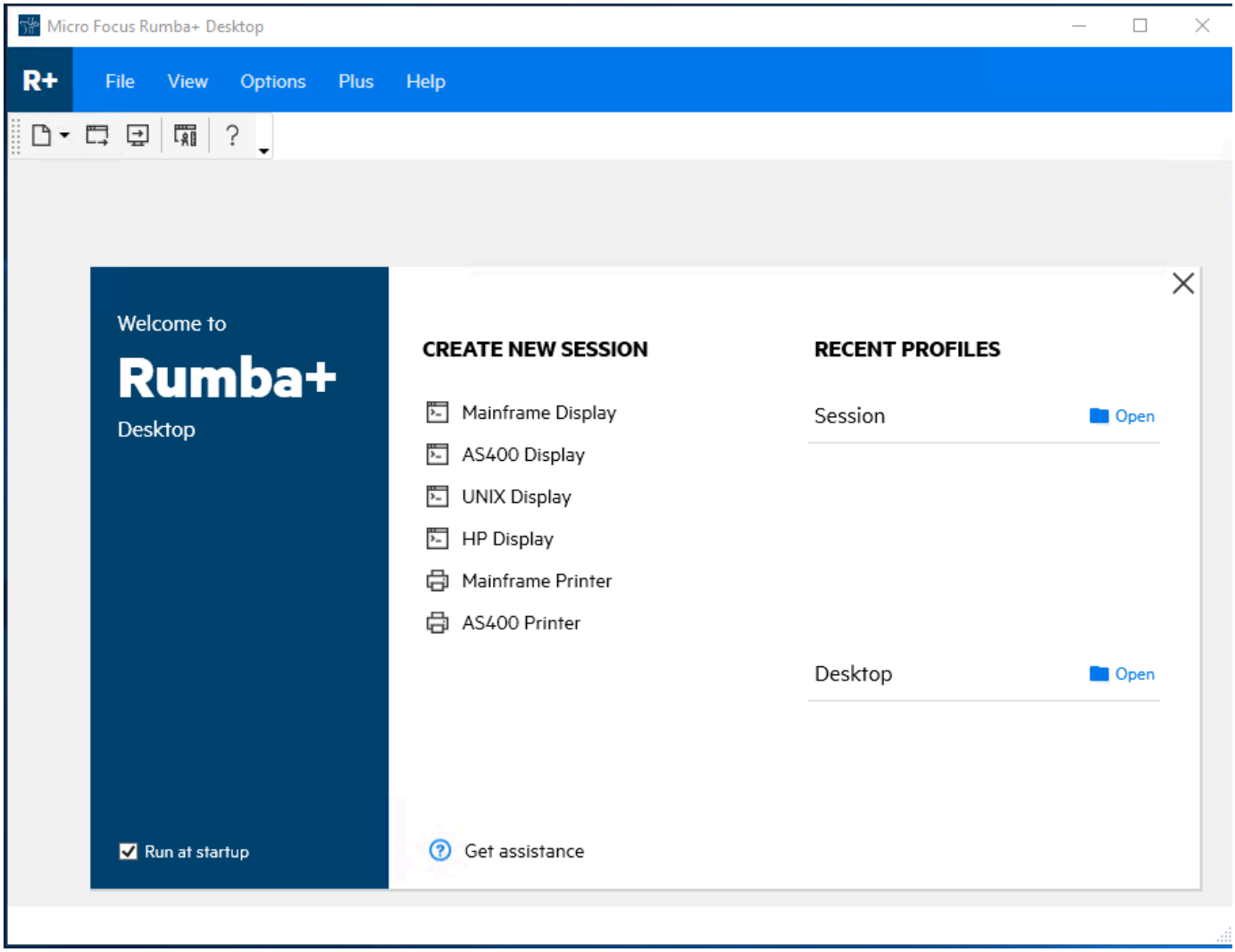
Pour configurer la connexion TN3270, vous avez également besoin du nom d'hôte DNS et du port de l'application.

Pour configurer et connecter une application au mainframe à l'aide de l'émulateur de terminal

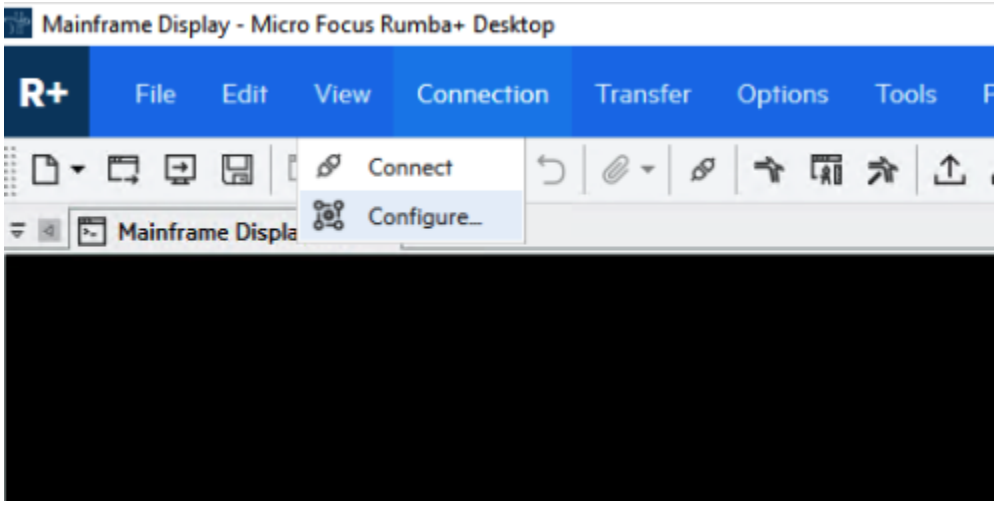
1. Ouvrez la console AWS Mainframe Modernization et choisissez Applications, puis choisissez`MicroFocus-CardDemo`.
2. Cliquez sur l'icône de copie pour copier le nom d'hôte DNS. Assurez-vous également de noter le numéro de port.
3. Démarrez un émulateur de terminal. Ce didacticiel utilise Micro Focus Rumba+.

   > **Note**
   >
   > Les étapes de configuration varient en fonction de l'émulateur.

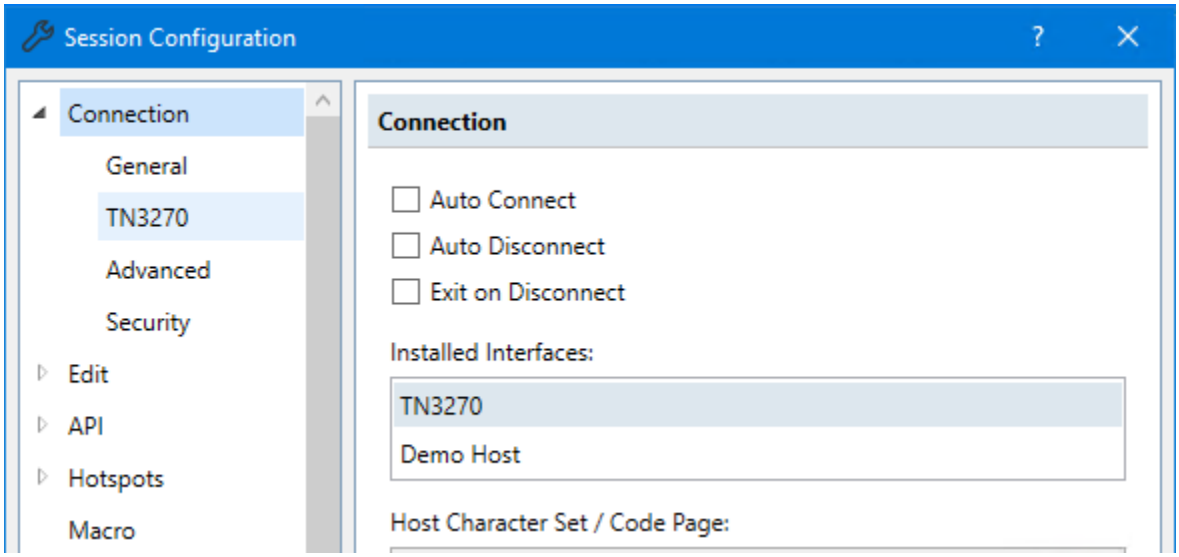
4. Choisissez Mainframe Display.



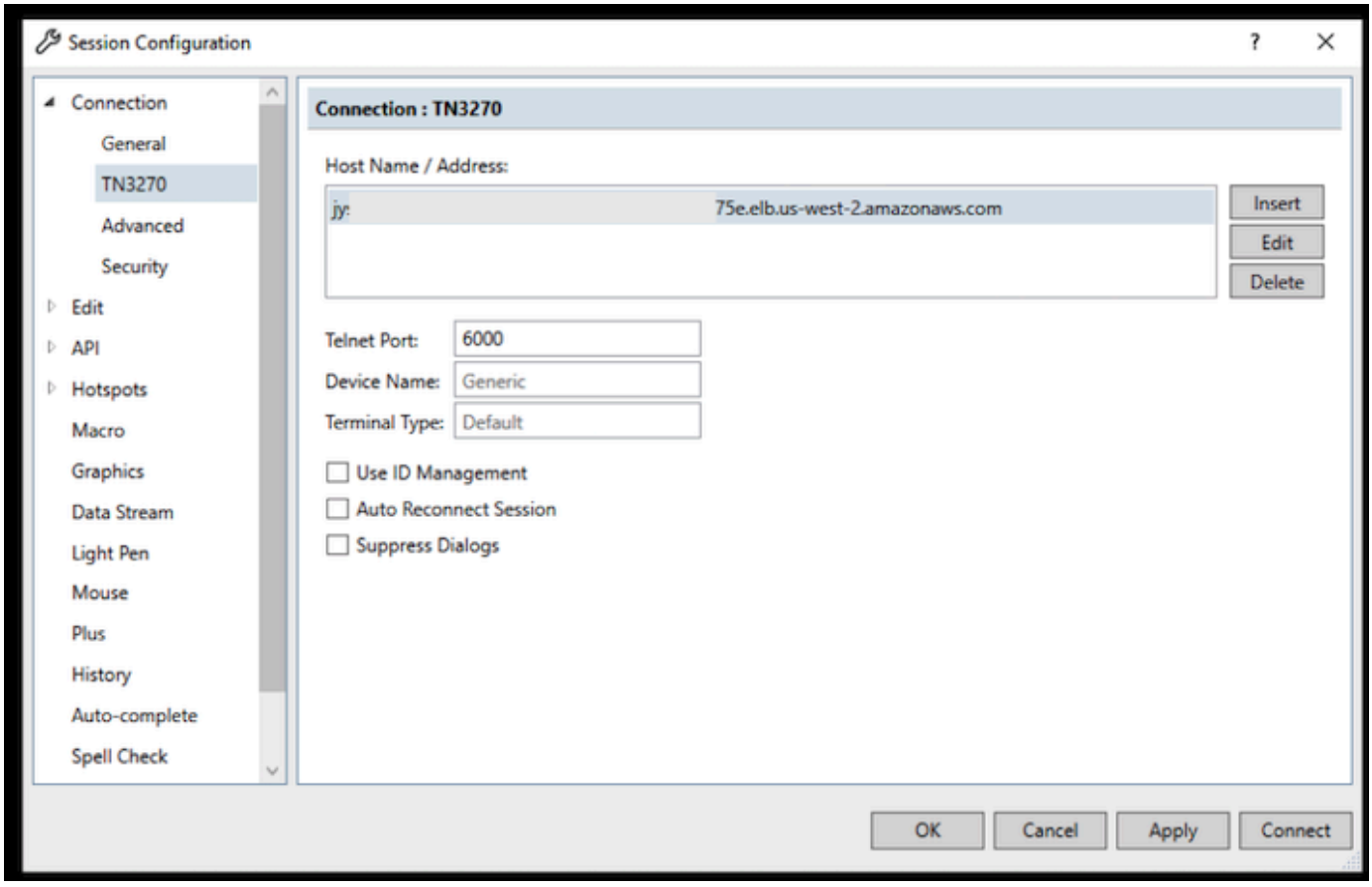
5. Choisissez Connexion, puis Configurer.



6. Sous Interfaces installées, choisissez`TN3270`, puis choisissez `TN3270` à nouveau dans le menu Connexion.



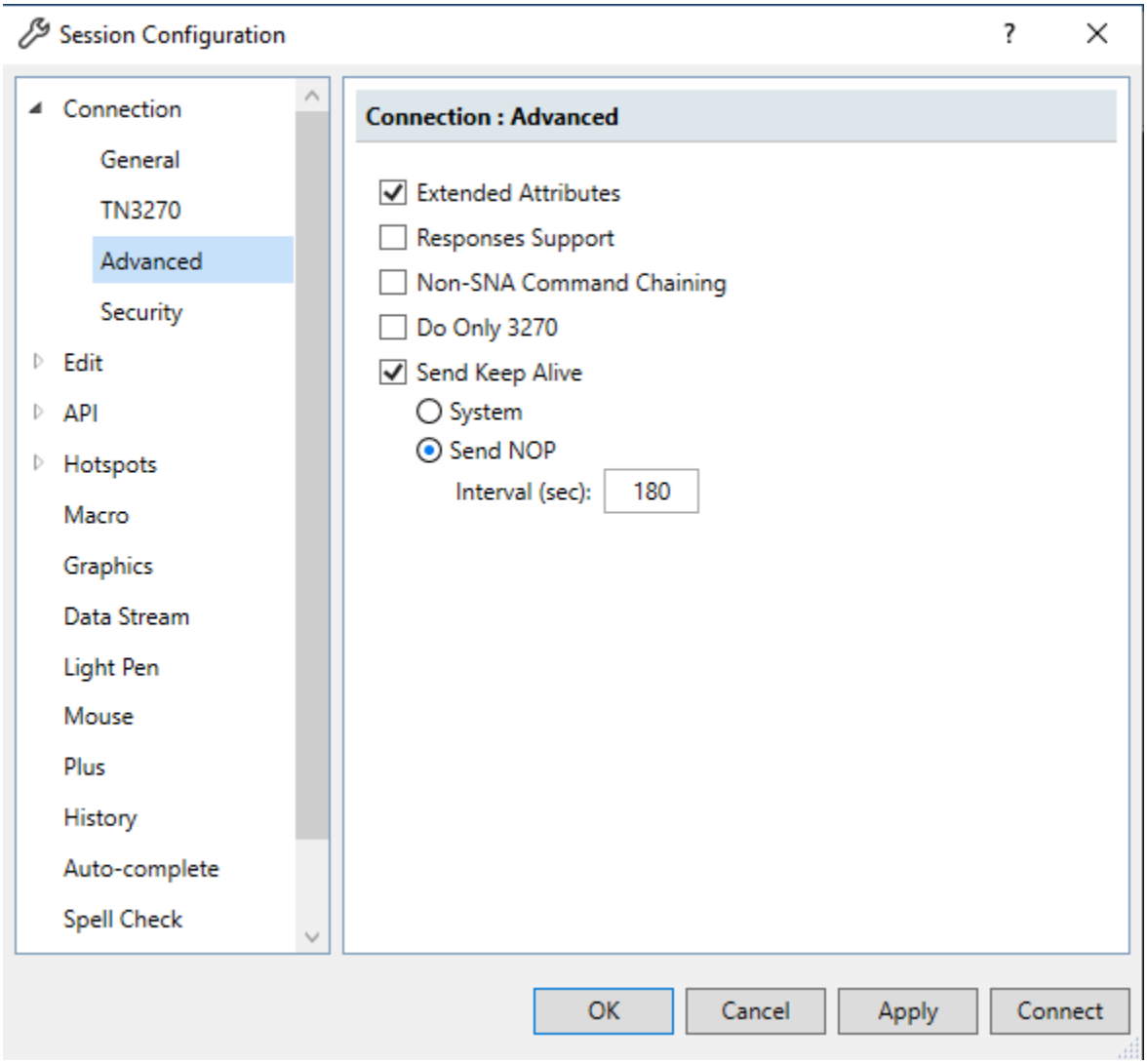
7. Choisissez Insérer, puis collez le `DNS Hostname` pour l'application. Spécifiez `6000` le port Telnet.



> **Note**
>
> Si vous utilisez AWS AppStream 2.0 dans un navigateur et que vous rencontrez des difficultés pour coller des valeurs, reportez-vous à la section Résolution des problèmes liés aux utilisateurs de la AppStream version 2.0.

8. Sous Connexion, choisissez Avancé, puis choisissez Send Keep Alive et Send NOP, puis entrez 180 pour l'intervalle.

> **Note**
>
> La configuration du paramètre Keep Alive sur votre terminal TN3270 à au moins 180 secondes permet de garantir que le Network Load Balancer n'interrompe pas votre connexion.



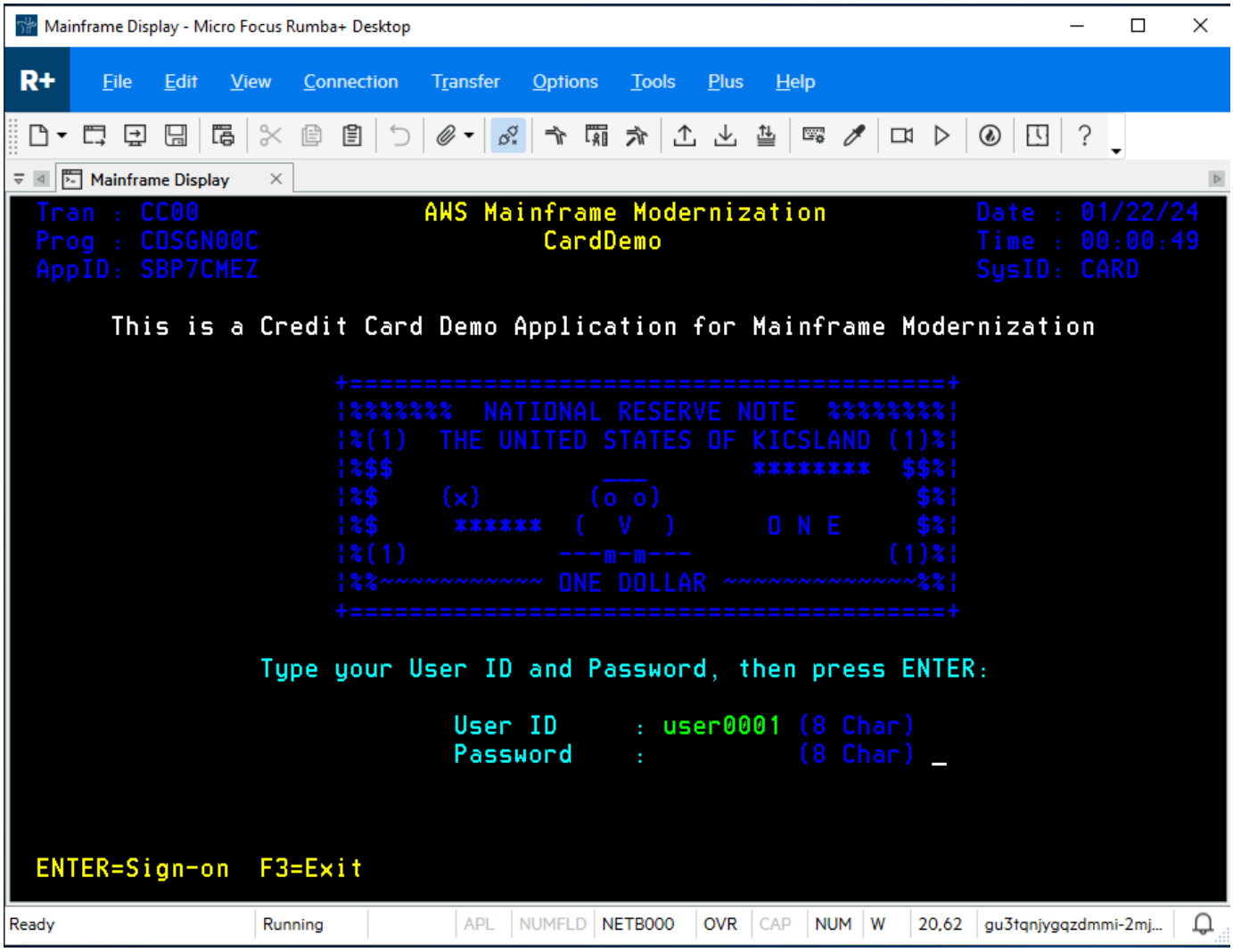
9. Choisissez Se connecter.

> **Note**
>
> En cas d'échec de la connexion :
>
> - Si vous utilisez la AppStream version 2.0, vérifiez que le VPC et le groupe de sécurité spécifiés pour l'environnement de l'application sont identiques à ceux du parc 2.0. AppStream
> - Utilisez le VPC Reachability Analyzer pour analyser la connexion. Vous pouvez accéder à l'Analyzer de Reachability via la console.

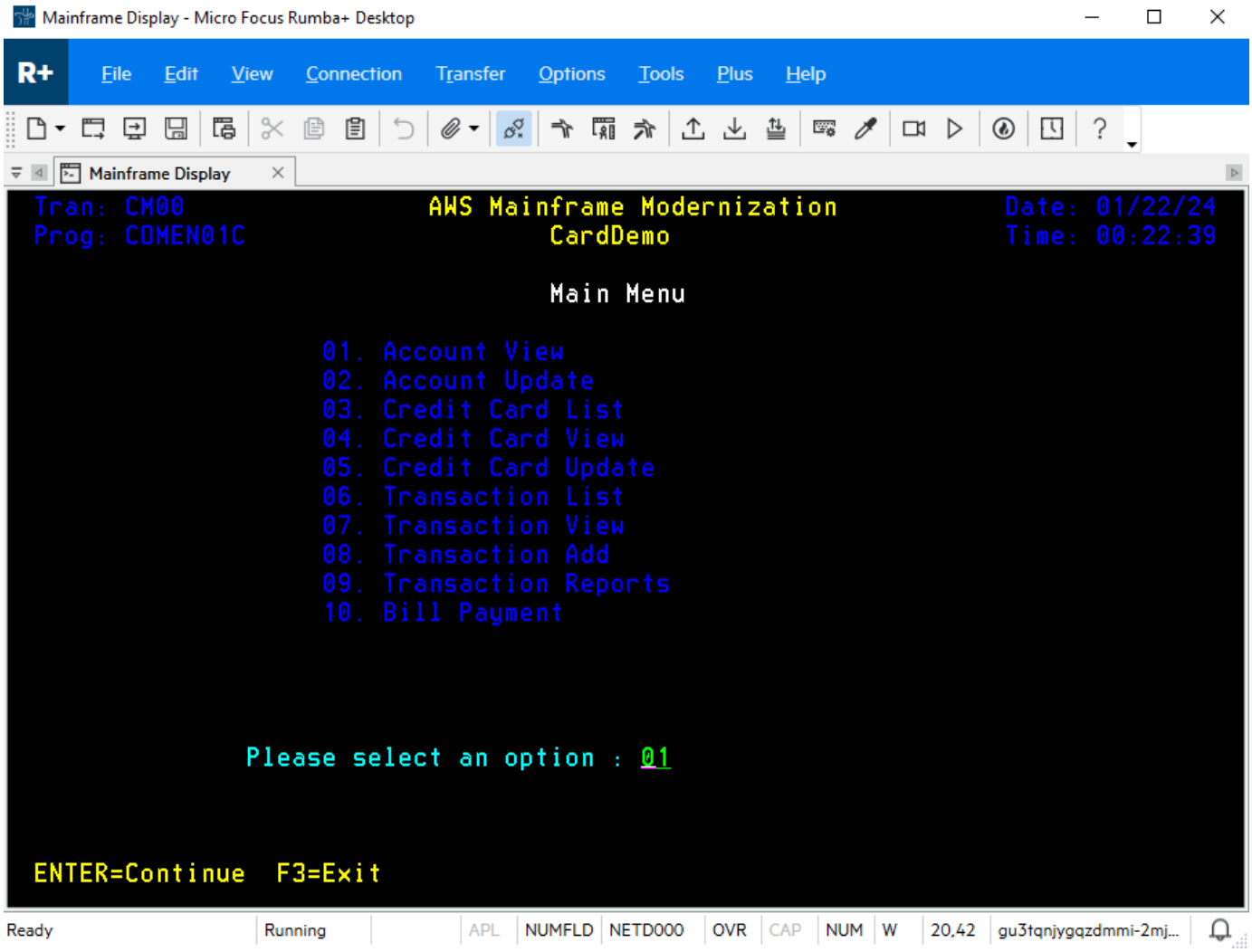
- À titre d'étape de diagnostic, essayez d'ajouter ou de modifier les règles entrantes du groupe de sécurité pour l'application afin d'autoriser le trafic vers le port 6000 depuis n'importe où (c'est-à-dire le bloc CIDR 0.0.0.0/0). Si vous vous connectez avec succès, vous savez que le groupe de sécurité bloquait votre trafic. Remplacez la source du groupe de sécurité par une source plus spécifique. Pour plus d'informations sur les groupes de sécurité, voir Notions de base sur les groupes de sécurité.

10. Entrez `USER0001` le nom d'utilisateur et `password` le mot de passe.

> **Note**
>
> Dans Rumba, la valeur par défaut pour Clear est ctrl-shift-z, et la valeur par défaut pour Reset est ctrl-r.

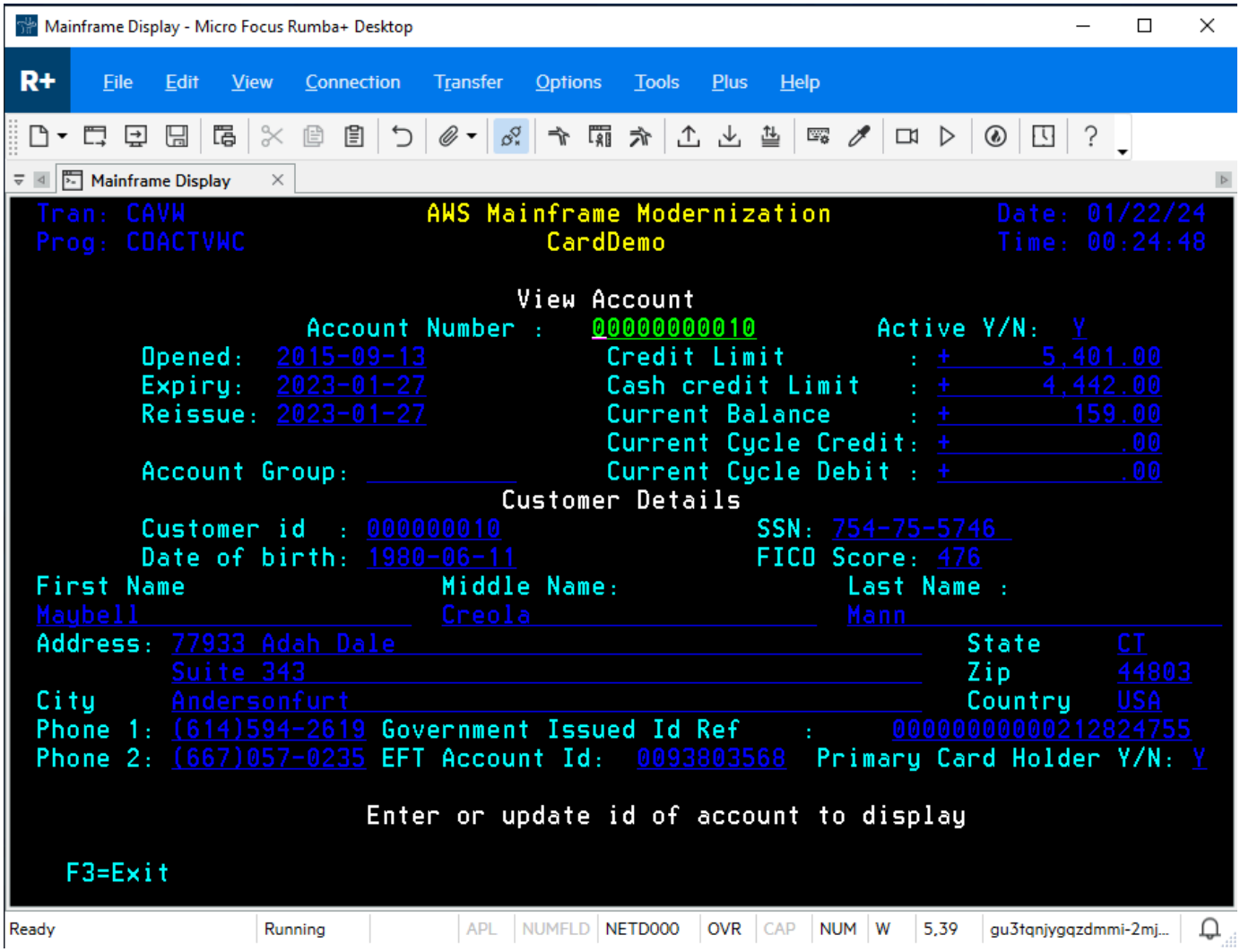


11. Une fois connecté, vous pouvez naviguer dans l' CardDemoapplication.
12. Entrez `01` pour la vue du compte.



13. Entrez le numéro `00000000010` de compte et appuyez sur la touche Entrée de votre clavier.

> **Note**
>
> Les autres comptes valides sont `0000000011` et`00000000020`.



14. Appuyez sur Quitter F3 pour accéder au menu et F3 pour terminer la transaction.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Si nécessaire, arrêtez l'application.
- Supprimez l'application . Pour plus d'informations, consultez Supprimer une application de modernisation AWS du mainframe.
- Supprimez l'environnement d'exécution. Pour plus d'informations, consultez Supprimer un environnement d'exécution de modernisation du AWS mainframe.

- Supprimez les compartiments Amazon S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez la section Suppression d'un compartiment dans le guide de l'utilisateur Amazon S3.
- Supprimez le AWS Secrets Manager secret que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un secret.
- Supprimez la clé KMS que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Supprimer des clés AWS KMS.
- Supprimez la base de données Amazon RDS que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Supprimer l'instance EC2 et l'instance de base de données dans le guide de l'utilisateur Amazon RDS.
- Si vous avez ajouté une règle de groupe de sécurité pour le port 6000, supprimez-la.

Étapes suivantes

Pour savoir comment configurer un environnement de développement pour vos applications modernisées, voir Tutoriel : Configuration AppStream 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer.

Approche de modernisation

La migration est complexe et comporte de nombreuses variables. AWS La modernisation du mainframe propose une approche évolutive qui offre des avantages à court terme en améliorant l'agilité et en offrant de nombreuses opportunités d'optimisation et d'innovation par la suite. En outre, la modernisation du AWS mainframe permet de simplifier le parcours tout en respectant les spécificités de l'entreprise et des activités de votre client. Les deux principales approches prises en charge par la modernisation des AWS mainframes sont le refactoring automatisé ou le replatforming. Le choix dépend de la situation de votre client.

Le refactoring automatisé utilise les outils AWS Blu Age pour convertir automatiquement le code, les données et les dépendances en langage moderne, en banque de données et en frameworks, tout en garantissant l'équivalence fonctionnelle avec les mêmes fonctions métier.

Le replatforming utilise les outils Micro Focus pour transformer les charges de travail du mainframe en services agiles sur. AWS

Vous pouvez envisager le processus de modernisation par étapes. La première étape comprend trois phases : évaluation, mobilisation, migration et modernisation. L'étape suivante comprend la phase d'exploitation et d'optimisation, au cours de laquelle vous pourrez identifier d'autres opportunités d'innovation.

Rubriques

Phase d'évaluation

Au niveau le plus élevé, la phase d'évaluation vise à déterminer si vous êtes prêt à effectuer la migration. Vous définissez une analyse de rentabilisation, puis vous formez votre équipe grâce à des ateliers et à une journée d'immersion (démonstrations et laboratoires) proposés parAWS. Les ateliers et les journées d'immersion abordent différents sujets. Ces tâches sont effectuées en dehors de la modernisation du AWS mainframe.

Phase de mobilisation

Dans la phase de mobilisation, vous démarrez votre projet par un lancement, puis vous exécutez un processus de découverte qui extrait les données de vos applications mainframe et les intègre dans un outil de migration. Vous identifiez les applications que vous souhaitez migrer et vous sélectionnez quelques applications à piloter. Vous affinez votre analyse de rentabilisation, rédigez votre plan de migration et décidez de la manière dont vous souhaitez gérer la sécurité et la conformité, la gouvernance des comptes et votre modèle opérationnel. Vous mettez en place un centre d'excellence dans le cloud avec les bonnes personnes de votre équipe. Vous exécutez les projets pilotes et vous documentez ce que vous avez appris. Vous affinez votre plan de migration et votre analyse de rentabilisation. La plupart de ces tâches sont effectuées en dehors de la modernisation du AWS mainframe.

Phase de migration et de modernisation

La phase de migration et de modernisation s'applique à chaque application et comprend plusieurs tâches, notamment l'affectation de personnes, l'exécution d'une découverte approfondie, la détermination de l'architecture d'application appropriée, la configuration des environnements d'exécution des applicationsAWS, le replatformage ou la refactorisation de votre code, l'intégration à d'autres systèmes et, bien sûr, les tests. À la fin de la phase, vous déployez les applications replateformes ou refactorisées en production et vous passez au nouveau système sur AWS. La plupart ou la totalité de ces tâches sont effectuées dans le cadre de la modernisation du AWS mainframe, dans un autre AWS service ou dans un outil auquel la modernisation du AWS mainframe donne accès.

Si vous souhaitez utiliser le refactoring automatique, consultez Blu Insights. AWS Blu Insights est désormais disponible AWS Management Console via l'authentification unique. Vous n'avez plus besoin de gérer des informations d'identification AWS Blu Insights distinctes. Vous pouvez accéder aux fonctionnalités de la base de code AWS AWS Blu Age et du centre de transformation directement depuis leAWS Management Console.

Pour migrer des données du mainframe versAWS, nous recommandons le AWS SCT et le. AWS Database Migration Service Pour plus d'informations, consultez Qu'est-ce que l'outil AWS Schema Conversion Tool ? dans le guide de l'utilisateur du AWS Schema Conversion Tool et qu'est-ce qu'AWS Database Migration Service ? dans le guide de AWS Database Migration Service l'utilisateur.

Phase d'exploitation et d'optimisation

Dans la phase d'exploitation et d'optimisation, vous vous concentrez sur la surveillance de vos applications déployées, la gestion des ressources et la mise à jour de la sécurité et de la conformité. Vous évaluez également les opportunités d'optimisation des charges de travail migrées.

Concepts

AWS La modernisation du mainframe fournit des outils et des ressources pour vous aider à migrer, à moderniser et à exécuter les charges de travail du mainframe sur celui-ci. AWS

Rubriques

Application

Une charge de travail récurrente dans le cadre de la modernisation du AWS mainframe. Un ensemble de tâches par lots, de transactions interactives (CICS ou IMS) ou d'autres composants constitue une application. Vous définissez le champ d'application. Vous devez définir et spécifier tous les composants ou ressources nécessaires à la charge de travail, tels que les transactions CICS ou les tâches par lots.

Définition de l'application

Définition ou spécification des composants et des ressources nécessaires à une application (charge de travail du mainframe) exécutée dans le cadre de la modernisation du AWS mainframe. Il est important de séparer la définition de l'application elle-même, car il est possible de réutiliser la

même définition pour plusieurs étapes (pré-production, production), représentées par différents environnements d'exécution.

Tâche par lots

Programme planifié configuré pour s'exécuter sans intervention de l'utilisateur. Dans AWS Mainframe Modernization, vous devez stocker à la fois les fichiers JCL des tâches par lots et les fichiers binaires des tâches par lots dans un compartiment Amazon S3, et indiquer leur emplacement dans le fichier de définition de l'application. Lorsque vous exécutez une tâche par lots, AWS Mainframe Modernization indique les valeurs d'état suivantes :

Soumission

Le traitement par lots est en cours de soumission.

En attente

Le traitement par lots est en attente.

Expédition

Le traitement par lots est en cours d'expédition.

En cours d'exécution

Le traitement par lots est actuellement en cours d'exécution.

Annulation

Le traitement par lots est en cours d'annulation.

Annulée

Le traitement par lots est annulé.

Réussi

Le traitement par lots s'est terminé correctement.

Échec

Le traitement par lots a échoué.

Réussite avec avertissement

Le traitement par lots s'est terminé correctement et une erreur mineure a été signalée. Le code de condition de tâche renvoyé dans le cadre de la GetBatchJobExecution réponse indique la cause de l'erreur.

Configuration

Caractéristiques d'un environnement ou d'une application. Les configurations d'environnement comprennent le type de moteur, la version du moteur, les modèles de disponibilité, les configurations de système de fichiers facultatives, etc.

Les configurations d'application peuvent être statiques ou dynamiques. Les configurations statiques ne changent que lorsque vous mettez à jour une application en déployant une nouvelle version. Les configurations dynamiques, qui constituent généralement une activité opérationnelle telle que l'activation ou la désactivation du suivi, changent dès que vous les mettez à jour.

Ensemble de données

Fichier contenant des données destinées à être utilisées par les applications.

Environnement

Combinaison nommée de ressources de AWS calcul, d'un moteur d'exécution et de détails de configuration créée pour héberger une ou plusieurs applications.

Modernisation de l'ordinateur central

Processus de migration des applications d'un environnement mainframe existant vers. AWS

Parcours migratoire

Le end-to-end processus de migration et de modernisation des applications existantes comprend généralement les phases suivantes : évaluation, mobilisation, migration et modernisation, et exploitation et optimisation.

Point de montage

Répertoire d'un système de fichiers qui permet d'accéder aux fichiers stockés dans ce système.

Refactorisation automatisée

Processus de modernisation des artefacts d'applications existants pour les exécuter dans un environnement cloud moderne. Cela peut inclure la conversion de code et de données. Pour plus d'informations, consultez la section AWS Mainframe Modernization Automated Refactor.

Replateforme

Processus de déplacement d'une application et de ses artefacts d'une plateforme informatique vers une autre. Pour plus d'informations, voir Replateforme de modernisation AWS du mainframe

Ressource

Composant physique ou virtuel d'un système informatique.

Moteur d'exécution

Logiciel qui facilite le fonctionnement d'une application.

Refactorisation automatique des applications avec Blu Age AWS

Le refactoring automatisé avec AWS Blu Age fournit une end-to-end solution pour la migration et la modernisation de vos applications mainframe. Les étapes du processus de refactorisation sont les suivantes :

- Analyser l'inventaire
- Analyser les dépendances
- Transformez automatiquement le code
- Capturez et gérez les scénarios de test

Vous pouvez effectuer les étapes précédentes dans l'outil Blu Insights, disponible via l'authentification unique depuis la console AWS Mainframe Modernization. Pour plus d'informations sur Blu Insights, consultez la documentation Blu Insights.

Lorsque vous êtes satisfait du code source transformé, il est temps de passer à AWS, où vous allez effectuer les étapes suivantes :

- Créez et déployez l'application refactorisée.
- Déployez et surveillez votre application dans AWS Mainframe Modernization.

AWS Blu Age Runtime (non géré) est l'une des offres du service de modernisation des AWS mainframes, avec AWS Blu Age Managed. Grâce à AWS Blu Age Managed, vous pouvez déployer votre application modernisée dans un environnement AWS géré qui simplifie votre expérience. Vous n'avez donc pas besoin de gérer l'infrastructure sous-jacente qui exécute votre application modernisée. En revanche, avec AWS Blu Age Runtime (non géré), vous pouvez déployer votre application modernisée dans votre propre AWS compte, afin de gérer votre propre infrastructure. Avec AWS Blu Age Runtime (non géré), vous avez la flexibilité d'utiliser tous les composants techniques nécessaires pour exécuter votre application modernisée comme vous le souhaitez.

AWS Blu Age Runtime (non géré) peut être déployé sur Amazon EC2 et sur Amazon ECS le. AWS Fargate

Rubriques

AWS Notes de mise à jour de Blu Age

Cette section contient les notes de publication de AWS Blu Age Runtime and Modernization Tools à partir de la version 3.5.0, les plus récentes en premier, organisées par numéro de version.

> **Note**
>
> Pour les notes de publication antérieures à ce document, contactez les services de livraison de AWS Blu Age. Pour plus d'informations sur les dernières fonctionnalités de Blu Insights, consultez les versions de Blu Insights.

Rubriques

- Runtime version 3.6.0
- Outils de modernisation, version 3.6.0
- Notes de mise à jour 3.5.0
- Runtime version 3.5.0
- Outils de modernisation, version 3.5.0

Notes de mise à jour 3.10.0

Cette version des outils d'exécution et de modernisation de AWS Blu Age est axée sur les principales mises à niveau et améliorations de base du produit, dans le but d'améliorer les performances et la robustesse à toutes les étapes de transformation et d'exécution. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de version de Java 8 vers Java 17, ce qui augmente la sécurité et les performances, et permet aux clients de déployer et d'exécuter des applications implémentées dans un langage plus moderne et d'utiliser des versions récentes de framework tiers.
- Support supplémentaire pour la gestion de grands espaces de mémoire partagée entre les utilisateurs ou les tâches, ainsi que pour le stockage de données réutilisables après le redémarrage de l'application ou de l'instance.
- Accès plus rapide à de grands ensembles de données dans Blusam grâce à un mécanisme de pagination qui permet de récupérer un sous-ensemble d'enregistrements de manière incrémentielle.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.10.0

Ce runtime est basé sur Java17, Spring2.7 et Angular16.

ZoS

Nouvelles fonctionnalités

- Blusam - Ajout du support pour les grands ensembles de données grâce à un mécanisme paginé dans lequel les index sont stockés et chargés à l'aide de pages

Améliorations

- DataUtils.compare amélioré pour gérer la conversion de priorité inférieure d'une chaîne en nombre
- Ajout du support pour vérifier que non ByteRange est créé avec des valeurs incorrectes via la propriété YML DataSimplifier. byteRangeBoundsVérifiez
- RemoveSosi () amélioré pour prendre en charge l'initialisation d'un caractère vide GraphicAlphanumericType
- Robustesse accrue pour le fonctionnement des tâches et lecture sécurisée de l'état GDG
- Blusam - Ajout du support pour effacer le cache des ensembles de données Blusam via une nouvelle méthode nommée .removeCache () CoreBluesamManager
- Blusam - Amélioration du comportement de suppression/renommage pour les ensembles de données Blusam classiques
- Redis - Support amélioré pour le déverrouillage des ensembles de données et la suppression du verrouillage des enregistrements
- JICS - Amélioration du message d'erreur en cas d'échec des demandes
- JCL - Ajout du support pour la concaténation de variables ControlM basée sur le caractère point
- JCL - Ajout du support pour Write ADVANCING (ADV) pour les fichiers GDG
- JCL - Support amélioré pour le numéro de génération actuel après suppression de tous les fichiers GDG
- JCL - Support amélioré pour la lecture de RDW/RecordSize à partir du catalogue lors de la création du jeu de données
- JCL - Ajout du support pour mettre à jour l'objet de ressource (depuis AbstractSequentialFile) lors de l'ouverture du fichier avec la taille de l'enregistrement de sortie de données
- JCL - Performances IDCAMS améliorées
- JCL - Support amélioré pour PRINT STATEMENT en ajoutant « CHAR » comme alias de « CHARACTER »
- SORT - Support amélioré pour les opérations de copie d'un ensemble de données de longueur fixe Blusam vers un ensemble de données de longueur variable
- SORT - Grammaire de tri améliorée pour gérer certaines instructions spécifiques

AS400

Nouvelles fonctionnalités

- Ajout de la prise en charge des espaces utilisateurs et des API associées
- Ajout de la prise en charge du paramètre TOMSGQ de SNDPGMMSG et mise en œuvre des files d'attente de messages
- CL - Ajout du support pour les paramètres FILE et SPLFNAME pour la commande OVRPRTF
- CL - Ajout du support pour la gestion des bibliothèques pour la table de partition correspondante avec la commande CPYF
- CL - Ajout de la prise en charge de la gestion de la commande CHGCURLIB et de la prise en compte de la bibliothèque actuelle lors de la création de requêtes
- CL - Ajout du support pour la gestion de la commande cl dans le cadre de l'appel stacktrace

Améliorations

- Amélioré MessageHandlingBuilder pour une meilleure gestion de l'entrée de trace de la pile d'appels
- Exécution parallèle améliorée de la fonctionnalité ContextPreconstruct
- Attributs d'affichage améliorés lorsqu'un enregistrement est créé par SFLINZ
- SAVOBJ amélioré pour permettre la gestion de plusieurs fichiers de sortie
- Amélioration de la gestion des programmes groovy en les ajoutant programCallStack lorsqu'ils sont appelés depuis un programme Java
- Détection améliorée du positionnement supérieur du modal d'aide
- Fonctionnalité TopGMQ améliorée lorsque le paramètre TomSGQ est fourni pour SNDPGMMSG
- Récupération améliorée des messages prédéfinis et fonctionnalité du chargeur de messages
- Amélioration de la gestion par CPYTOIMPF des caractères séparateurs dans le contenu
- Verrouillage de déverrouillage amélioré sur l'enregistrement READ

Capacités transversales

Nouvelles fonctionnalités

- Ajout d'une traduction pour les messages système sur le Front-End
- Ajout d'une nouvelle méthode ExecutionContext pour renvoyer la pile d'appels du programme
- Définissez un séparateur de lignes (pour simplifier les données) quel que soit l'environnement réel

- Ajout de la possibilité de configurer le chemin JSON du modèle SQL

Améliorations

- Amélioration de la méthode de comparaison DataUtils. compareAlphInt() lorsqu'il s'agit de rembourrage
- Création d'un drapeau pour autoriser un comportement personnalisé en cas d'exception dans les requêtes de curseur
- Conversion graphique améliorée de la base de données LOWVALUES

Troisième partie

- Mise à niveau pour atténuer les risques CVE-2024-21634, CVE-2023-34055, CVE-2023-34462, IN1-JAVA-ORG/SPRINGFRAMEWORKSECURITY-5905484, CVE-2023-46120, CVE-2023-6481, CVE-2023-6378, CVE-2023-5072)

Outils de modernisation, version 3.10.0

ZoS

Améliorations

- COBOL - Ajout du support pour la fonction ABS
- JCL - Étendue variable améliorée : attachée à STEP au lieu de JOB
- Injection améliorée des paramètres du curseur pour les valeurs faible/élevée
- Analyse CSD améliorée, notamment pour les TRANSACTIONS à distance

AS400

Améliorations

- Vérification en blanc supprimée pour l'indicateur de niveau de contrôle
- Ajout de la prise en charge du nom externe pour les mots clés IMPORT/EXPORT
- Ajout du support pour %LEN sur les champs
- CL - Ajout du support pour les nouveaux opérateurs pour le langage CLLE

- CL - Ajout du support pour les IF imbriqués
- COBOL - Gestion améliorée de la commande START lorsqu'elle est utilisée avec plusieurs touches
- DSPF - Gestion améliorée de la position du curseur avec un numéro d'enregistrement
- DSPF - Amélioration du formatage pour les champs numériques signés, les champs numériques uniquement et les champs à grande échelle
- DSPF - Amélioration de la détermination du titre pour Screen General Help
- DSPF - Support amélioré des spécifications d'entrée/sortie
- DSPF - Gestion améliorée des séparateurs de regroupement lors de la validation d'un champ numérique
- Sortie de mappage/enregistrements DDS améliorés
- Amélioration de la capacité du mot clé REFFLT du fichier d'imprimante à résoudre les champs référencés
- RPG - Support amélioré pour les déclarations « TOUTES gratuites »
- RPG - Analyse des conditions améliorée et prise en charge ajoutée de la gestion de CABXX sans résultat TAG
- RPG - Gestion améliorée des spécifications d'entrée des champs numériques
- RPG - Gestion améliorée des appels de procédure dans les conditions IF/ELSEIF/WHEN
- RPG - Gestion améliorée de la commande READ lorsqu'elle est appelée sur un fichier dspf
- RPG - Améliorer la prise en charge des fichiers faisant référence à un DDS inexistant
- Améliorez la gestion du REFFLD lorsqu'un nom de format d'enregistrement physique est transmis
- Ajout du support pour utiliser « return » comme nom de colonne de base de données

Capacités transversales

Nouvelles fonctionnalités

- Oracle - Permet de définir des utilisateurs plutôt que SYS pour stocker les fonctions intégrées

Améliorations

- Version Java améliorée de v8 à v17
- Condition SQL améliorée avec le nom de colonne du cluster

- Ajout de la prise en charge des clauses ORDER BY depuis la vue

Notes de mise à jour 3.9.0

Cette version de AWS Blu Age Runtime and Modernization Tools met l'accent sur de nombreuses améliorations transversales apportées au produit dans le but d'améliorer les performances des architectures à haute disponibilité, ainsi que sur de nouvelles fonctionnalités permettant d'améliorer l'exécution des tâches à un niveau supérieur. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de la version d'Angular 13 vers Angular 16, renforçant la sécurité et donnant accès à de nouvelles fonctionnalités qui améliorent les performances des applications en ligne des clients.
- Ajoutez la prise en charge des fonctionnalités intertâches dans l'AS400, avec notamment le fait que les tâches peuvent envoyer des messages de demande de manière synchrone entre elles, ce qui permet le découplage dans les tâches modernisées.
- Améliorations des performances liées à l'utilisation de Redis, notamment l'optimisation du pool de connexions, une sécurité de connexion élevée et un mécanisme de verrouillage des ensembles de données amélioré.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.9.0

ZoS

Nouvelles fonctionnalités

- Programme de tri : entrées VSAM mises à jour avec une longueur fixe
- JHDB DB : ajout d'un délai d'attente configurable

Améliorations

- Support amélioré du séparateur de lignes à diffuser en continu s'il est utilisé dans la concaténation de fichiers
- Support amélioré pour ouvrir des fichiers séquentiels concaténés. Initialiser DataSetIndex après ouverture du fichier

- Support amélioré pour le séparateur décimal virtuel lorsque a NumericEditedType est affecté à une valeur numérique
- Support amélioré pour NumericEditedType les valeurs non négatives
- IDCAMS : les cartes SYSIN sont désormais lues à l'aide de la propriété « encoding » définie dans le fichier .yml application-utility-pgm
- IDCAMS : grammaire mise à jour pour prendre en charge l'argument FILE (..) dans l'instruction DEFINE CLUSTER
- INFUTILB : Ajout de la prise en charge de l'argument DFSIGDCB pour remplacer les paramètres DCB de DD SYSREC
- INFUTIL : prise en charge améliorée du paramètre « DFSIGDCB YES »
- SPLICE amélioré pour gérer un énorme fichier d'entrée
- DFSORT : Gestion améliorée des champs de commentaires
- DFSORT : Ajout du support pour le format numérique libre (signé/non signé) (SFF/UFF)
- SORT : Ajout du support d'analyse pour les instructions OPTION PRINT et OPTION ROUTE
- SORT/ICEMAN : Ajout du support pour les opérations de division fermées (champ avec opérateur DIV)
- Support amélioré pour CICS READ à l'aide d'une clé générique
- Correction de la fonction StringUtils .chargraphic pour supprimer le SOSI d'un type graphique
- Améliorez les performances sur DataUtils. isDoubleByteEncodage
- JCL : prise en charge améliorée du mode de disposition KEEP pour un ensemble de données temporaire. Le système change la disposition en PASS
- JCL : gère les paramètres DCB de manière dynamique
- JCL : sorties SUM FIELDS améliorées pour les valeurs incorrectes
- JCL : CommonddUtils : :getContent recherche désormais le RecordSize dans le catalogue
- JCL : lecture des attributs RDW/RecordSize du catalogue lors de la création du jeu de données
- JCL : Ajout du support pour DCB=.MYDD pour copier les paramètres DCB d'un DD dans un autre au cours de la même étape
- JCL : système d'héritage amélioré de la taille des enregistrements
- JCL : ajout d'un verrou de jeu de données exclusif (Redis)
- Redis : ajout du support SSL pour le mode autonome
- Redis : ajout du nombre de verrous Redis synchronisé avec le verrou

- Redis : paramètres de pool pris en charge pour le verrouillage Redis
- Redis : actualisation optimisée des métadonnées avec Redis
- Redis : support amélioré du cluster Redis
- Amélioration des verrous ouverts avec le mode IO
- Les ensembles de données améliorés verrouillent les performances et suppriment les verrous inutilisés
- Chemin amélioré de l'ensemble de données lors du désenregistrement du fichier
- Invalidation améliorée du cache de fenêtre avant la lecture
- Ajout de la prise en charge de l'utilisation des fournisseurs de sources de données utilitaires Thread Safe
- Contrôle de nullité amélioré de DatasetState
- Support amélioré pour ne pas rouvrir les ensembles de données déjà ouverts
- Robustesse accrue pour le fonctionnement final de la tâche
- Support amélioré pour l'ordre des index, des clés, permettant les doublons
- Support amélioré pour l'ordre de sérialisation des listes à ignorer
- Ajout de la prise en charge de la fonction de vidage de bogues pour aider à diagnostiquer les problèmes d'ordre des index
- Support amélioré pour l'actualisation des métadonnées
- Support amélioré pour la lecture en bloc sur Blusam

AS400

Nouvelles fonctionnalités

- Crée un registre du contexte de l'application
- Support du mot clé DSPF CLRL (NO) Support de la surveillance des blocages d'enregistrements
- Support pour keyed DataQueue
- Support pour les messages de demande pour les tâches par lots
- Ajout du support pour le fichier d'imprimante décrit par le programme pour AS400 COBOL
- Gère la commande RMVJOBSCDE cl
- Amélioration pour RUNSQL/DLYJOB

- CHKOBJ : augmentation du code d'erreur existant pour le paramètre LIB
- SNDPGMMSG : prend en charge les paramètres de chaîne
- RTVDTAARA : sous-chaîne améliorée dans LDA
- DSPFD : paramètre FILE pris en charge ajouté pour un nom de fichier spécifique
- RUNQRY : Support du fichier SQL dans QRY PARAM
- CRTDUPOB : Support pour copier les données entre les zones de données
- SBMJOB : Convertit les instructions à utiliser JobQueueManager
- OPNQRYF : Ajout du support pour la bibliothèque Qtemp
- CRTDUPOBJ : Logique améliorée pour copier le contenu d'une partition
- CRTDUPOBJ : Ajout du support pour Qtemp pour les vues
- RTVSYSVAL : Support de la valeur SYSVAL, QDATFMT dans la commande CL
- CHKOBJ : Ajout du support pour OUTQ
- RTVJOBA : Supporte le paramètre SWS
- SNDPGMMSG et RCVMSG : paramètres supplémentaires pris en charge MSGF, MSGFLIB, MSGDTA, MSGTYPE, KEYVAR, MSGKEY, MSGID

Améliorations

- Supports améliorés des cartes d'E/S WORKSTATION
- Gestion améliorée du message défini superposé au message précédent
- Supporte des informations de message supplémentaires sur array-messageline
- Accès amélioré aux wrappers de tableaux autonomes dans EVAL, SortA et figuratives
- Améliorez le nettoyage des DAO lorsque l'application en ligne se termine
- Ajout de la prise en charge de formats de date supplémentaires et amélioration de la gestion des entrées de chaîne
- Amélioration de la gestion CVTDAT de SYSVAL en ajoutant la classe d'assistance aux valeurs du système Decode et des paramètres de construction à partir de la commande CL SbmJob
- Le package com.netfective.bluage.gapwalk.rt.blu4iv a été supprimé de l'analyse des composants gapwalk-cl-command
- Amélioration de la prise en charge des messages prédéfinis pour l'API de file d'attente de messages

- Amélioration du support retrieveSubfileRecord pour les enregistrements écrits dans un autre programme
- Amélioration de la prise en charge des messages immédiats pour l'API de file d'attente de messages
- Gestion améliorée de la zone de données locale lors de la soumission d'une offre d'emploi
- Démarre JobQueues automatiquement au démarrage du serveur
- Utilise la configuration ApplicationContext pour décoder les paramètres de SBMJOB
- Amélioration des messages d'erreur fournis par le système
- Permet à RTVMSG de rechercher des fichiers .properties dans des sous-répertoires imbriqués
- Gère la réinitialisation des entités liées à des pointeurs incorrects/invalides
- Amélioré MessageHandlingBuilder pour afficher MsgID et MsgFile nom sous forme de chaînes pour RCVMSG
- Méthode de withMsgFile nom améliorée de l'API de mise en file d'attente des messages
- Mécanisme de verrouillage de la zone de données amélioré
- RTVMBRD : Support des minuscules et des majuscules pour le paramètre FILE
- CRTDUPOBJ : Gestion améliorée des vues
- CPYTOSTMF : Gestion améliorée de la connexion
- CPYF : amélioration de la gestion du nom du répertoire lors de la copie à partir d'un fichier plat
- RCVF : gère correctement les paramètres DEV/RCDFMT et la transformation de RCDFMT pour Groovy et Java
- RCVF : gère les appels suivants et évite de réinitialiser le curseur
- CPYF : Ajout du support pour l'écriture à partir de fichiers plats
- CRTDUPOBJ : Ajout de la gestion du nouvel obj avec la bibliothèque Qtemp
- CHGDTAARA : augmentation de la longueur maximale de la zone de données de 256 à 2000
- SAVOBJ : Assurez-vous que les enregistrements enregistrés sont dans l'ordre d'insertion
- RTVDTAARA : Valeurs récupérées (à ne pas découper)
- CHKOBJ : renvoie les messages de surveillance corrects lorsque le membre n'existe pas
- RTVDTAARA : Ajout du support de la sous-chaîne LDA
- RTVDTAARA : renvoie des espaces blancs jusqu'à la longueur de la variable spécifiée dans le paramètre RTNVAR

- RTVDTAARA : prend en charge les paramètres entiers pour le début et la longueur et prend en charge le dernier format de transformation
- CHGDTAARA : Ajout de la prise en charge des paramètres qui incluent les limites inférieures et supérieures
- CHKOBJ : gère la valeur VIEW pour le type d'objet paramètre
- CHKOBJ : résultat défini sur vrai quel que soit le membre si la vue existe

Capacités transversales

Nouvelles fonctionnalités

- Gère la génération de rapports dans des fichiers .txt
- Ajout de la propriété de source de données CurrentSchema XA au gestionnaire de secrets
- Ajoutez la propriété yml database.cursor.raise.already.opened.error pour permettre au framework de générer l'erreur SQLCODE 502 lorsque le curseur est déjà ouvert

Améliorations

- Ajout de pompons Gapwalk à l'emballage AWS Blue Age sur Amazon EC2
- Utilise le nouveau paradigme du gestionnaire de signaux par défaut
- Ajout d'un support pour le verrouillage lorsque la disposition est MOD ou ANCIENNE
- Ajout d'un cache pour stocker les modèles de date/heure de la base de données
- Fonction de contrôle améliorée de PackedType
- Améliorez les fonctions DataUtils .setTo pour les enregistrements avec VariableSizeArray
- Gère l'option MQ SYNCPOINT en ce qui concerne l'unité d'exécution
- Framework activé pour définir le SQLCODE lors d'une transaction d'annulation
- Ajout d'un nom de classe de pilote automatique en fonction du secret de la clé du moteur
- Délai d'expiration du programme/de la transaction
- Restaurer la position du curseur après le rollback lors de l'accès au curseur

Troisième partie

- Mettez à niveau les SDK SnakeyAML, Redisson et Amazon, supprimez YamlBeans (atténuez les problèmes CVE-25857, CVE-2023-24621, CVE-2023-42809, CVE-2023-44487)

Outils de modernisation, version 3.9.0

ZoS

Améliorations

- Support amélioré pour XML-TEXT en tant que source pour une cible de type String
- Flux de travail STM vers UML amélioré pour prendre en charge le modèle de division X/ (Y/Z)
- JHDB DB : accepte l'appel ROLLBACK avant toute mise à jour de base de données
- JHDB DB : accepte ROLLBACK même si la transaction est terminée (NOP)
- JCL : fonction de validation des étapes améliorée
- SORT : gère la fonction SUM avec des valeurs négatives décimales de zone
- COBOL : ajoute la prise en charge de l'échappement entre guillemets simples ou doubles dans les chaînes littérales

AS400

Améliorations

- Fonction intégrée %editc améliorée : gestion du code d'édition X en ajoutant des zéros en tête
- Gestion améliorée de la valeur initiale des champs en entrée uniquement
- Ajout de touches d'action pour les boîtes de dialogue d'aide
- Enregistrement du pied de page du tableau dynamique apparaissant en bas
- Commande START gérée sans KEY PHASE pour les fichiers qui spécifient une clé d'enregistrement réelle
- Valeur par défaut ajoutée pour les types float et NumberUtils ::pow
- Ajout du support pour définir une variable à l'aide de LIKE (IN)
- Gestion de la boucle FOR mise à jour pour permettre l'omission d'éléments facultatifs
- Analyse RPG mise à jour pour associer les enregistrements au nom du tableau CTDATA
- Gestion améliorée des indicateurs pour les relevés CabXX
- Supporte le paramètre facultatif sur le mot clé COMMIT
- Support amélioré des mots clés FORMAT dans LF
- Code d'opération LOOKUP géré avec indicateurs élevés et égaux (ou faibles et égaux)
- Nom de clé PF géré déclaré entre guillemets

- Amélioration de la gestion de l'EDTCDE X pour ne pas supprimer les zéros en tête
- Amélioration de la prise en charge de MSGCONE dans le fichier d'imprimante, sans générer d'étiquettes anonymes
- Le CONTENU des champs est partagé par plusieurs structures de données
- Paramètre ERRSFL géré en combinaison avec SFLMSG/SFLMSGID
- Code principal amélioré avant la portée de la déclaration du processus d'un RPG entièrement gratuit
- Spécification de contrôle conditionné par analyse syntaxique ajoutée
- Support amélioré pour la méthode setErrSfl () dans le dataholdermapper
- Résolution de type améliorée pour les variables créées en interne
- Support amélioré pour l'opcode Z-ADD
- Amélioration de la gestion du champ constant avec une valeur DFT
- Améliorez la prise en charge du champ entier dans le statut du programme ds
- Affectation d'indicateurs gérée dans les paramètres ENTRY
- Amélioration du filtre des mots clés propagés via le mot clé ref/reffield
- Structure de DataArea données sans nom prise en charge
- Gestion améliorée du type de données du pointeur
- Eléments manipulés du tableau utilisés pour définir des variables avec le mot clé LIKE : accès au tableau dans le champ de sortie
- Support amélioré pour le numérique signé, affichage uniquement des chiffres
- Relation logique prise en charge sur la carte O
- Cas de test pour %CHAR en caractères alphanumériques
- Spécification de contrôle prise en charge (mot-clé principal)
- EDTCDE avec deux paramètres dans le fichier d'imprimante
- Analyse FullFree RPG améliorée
- Amélioration du tableau dynamique pour garantir le positionnement correct du pied de page
- Ajout du support pour l'initialisation des types numériques avec TOUTES les constantes figuratives
- Gestion améliorée de plusieurs fichiers logiques RPG faisant référence au même fichier physique
- Améliorez la détection des champs modifiés sur un écran moderne
- Synchronisation modale avec des champs dynamiques
- Amélioration de la gestion du champ numérique signé en sortie uniquement

- Améliorez la prise en charge des cartes d'E/S WORKSTATION

Capacités transversales

Nouvelles fonctionnalités

- Outil de migration de données : ajout de la propriété ebcdicFilesWith VarcharIn VB pour permettre de prendre en compte la longueur VARCHAR de 2 octets lors de la lecture d'octets
- Implémentation d'une API commune pour enregistrer les erreurs
- Implémentation BluAgeErrorDictionaryUtils et utilisation d'une API commune pour enregistrer les erreurs et/ou les informations dans COBOL2Model, CycleBuilder RPG, Definitions2Model et FieldsProcessor
- Grammaire SQL améliorée pour prendre en charge différentes définitions de clauses d'isolation

Améliorations

- Version Angular améliorée vers la version 16
- Angular : version ajv améliorée de 6 à 8.9

Troisième partie

- Groovy mis à jour vers la version 2.4.15

Notes de mise à jour 3.8.0

Cette version de AWS Blu Age Runtime and Modernization Tools est axée sur de multiples améliorations transversales du produit afin d'améliorer sa qualité et sa sécurité, ainsi que sur l'amélioration des performances de mise en cache et sur l'unification des supports de commandes dans une distribution unique. Certaines fonctionnalités et modifications clés de cette version sont les suivantes :

- Mise à niveau de version de Spring 2.5 vers Spring 2.7, augmentant le support de maintenance, les performances et la sécurité de la plateforme.
- Unification de la prise en charge de plus de 82 commandes CL dans le cadre de la over-the-counter distribution afin de faciliter l'utilisation et le déploiement d'applications modernisées qui utilisaient auparavant des scripts CL.

- De nouvelles API sont disponibles pour mieux fonctionner et interagir avec les ensembles de données BluSam, telles que l'importation intégrée vers le service géré et la capacité de répertorier les informations de métadonnées des ensembles de données.
- Améliorations des performances et extension de l'utilisation de Redis, y compris la disponibilité en mode cluster, la récupération de données à haute disponibilité, la standardisation de l'utilisation des secrets.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.8.0

ZoS

Nouvelles fonctionnalités

- Gestion de la définition de la clé sous forme de chaîne pour DynamicFileBuilder
- DFSORT : Ajout de la prise en charge des éléments multiples dans OUTFIL TRAILER1 + initialisation de la grammaire DFSORT
- Outil CommonddUtils : gestion de la taille des enregistrements dans les données in-stream
- Fichier indexé : manipulation de l'option GENKEY

Améliorations

- Services de chargement BluSam externalisés dans un fichier jar séparé
- Ajout de la prise en charge de la configuration de l'emplacement pour le stockage des fichiers temporaires
- Mécanismes de cache partagé améliorés pour les cas impliquant plusieurs nœuds
- Utilisation du cache partagé : IDCAMS vérifie l'optimisation
- Améliorez l'injection ROWID pour la sélection intégrée
- JCL : chaque procédure de travail in-stream est désormais générée dans un fichier Groovy distinct
- Assurez une couverture de card-demo-v 2 % sur les cartes IDCAMS JCL
- BluSam : évitez le double WarmUp lors de l'utilisation de plusieurs instances
- Réduction de l'encombrement mémoire lors de l'hydratation du cache

- Support de configuration du pool Jedis
- Séparateur de ligne ajouté au flux s'il est utilisé dans la concaténation de fichiers
- Support des cartes EBCDIC + commentaires de bloc (/*.../) dans l'utilitaire IDCAMS
- Requête de support de base de données : prise en charge des chaînes à double octet lors de la conversion du niveau 49 en SQL
- Grammaire DFSORT : implémente 17 instructions de contrôle + intégration de 2 d'entre elles (OMIT/INCLUDE)
- Améliorez les colonnes GRAPHIC, récupérez INFUTILB
- Support pour la lecture de fichiers avec tableau de taille variable
- Support pour la signature ZonedType par nibble où le premier bit du dernier octet est « E »
- DFSORT/ICETOOL ajoute la prise en charge de l'argument NOMATCH =(..) si un enregistrement ne correspond à aucune des constantes de recherche CHANGE
- Compatibilité avec Redis Cluster
- Gestion du statut du job (échec) en fonction d'un code de sortie groovy
- Support amélioré de CICS SYNCPOINT ROLLBACK.
- Fenêtre de pré-extraction pour optimiser l'utilisation du cache Redis
- JCL/GROOVY : hérite de la propriété IsRDW de l'ensemble de données de l'étape précédente lorsque DISP= (, PASS)
- Gestion de copies partielles de données avec un tableau de taille variable

AS400

Nouvelles fonctionnalités

- Support pour les cartes d'E/S pour l'affichage de fichiers
- Support pour des informations de message supplémentaires pour les mots clés DSPF ERRMSGID et CHKMSGID
- Support pour plusieurs messages d'erreur sur l'écran frontal
- Ajout ou amélioration de la prise en charge de 82 commandes CL dans l' gapwalk-cl-commandapplication

Améliorations

- Support amélioré pour DELETE et READ dans le cadre du contrôle des engagements
- ConvertDate à l'intérieur du %dec intégré
- En-têtes de sécurité XSS renforcés
- Robustesse et cohérence améliorées de la génération STM (meilleure gestion de : ligne de suite dans un rpg en format libre, virgules pour la partie décimale, blocs de forme libre dans la définition/déclaration)
- DataHolderMapper Génération améliorée
- Robustesse et champ de modification accrus dans DataAreaFactory
- Amélioration du changement de focus sur la touche de tabulation
- Performances améliorées lors de la génération de rapports Jasper
- Affichage décimal amélioré avec rembourrage 0
- Support amélioré pour le champ ROW/COL dans INFDS
- Améliorez la prise en charge des champs modifiés depuis l'écran
- Ajout de getters pour le nom et le chemin du rapport généré
- Amélioration de la longueur de la file d'attente de données
- Configuration automatique améliorée des files d'attente de tâches pour répondre aux nouvelles normes de Spring Boot 2.7
- Mises à jour améliorées du poste de travail pour plusieurs sessions simultanées

Capacités transversales

Nouvelles fonctionnalités

- Support de l'absence de tolérance de données non valides pour Packed
- Pagination/filtrage ajoutés pour répertorier les points de terminaison des jeux de données

Améliorations

- Stratégie de transformation des requêtes ORACLE améliorée lors de la comparaison de colonnes par rapport à une chaîne vide
- Gestion de BLOB DB2 avec les programmes utilitaires DSNTEP et INFUTILB. Les BLOB DB2 sont désormais modernisés vers des postgres de type BYTEA.

- Amélioration de la suppression du dernier élément du curseur
- Support amélioré pour supprimer le fichier RRDS
- Performances améliorées de AWS Blusam Secret
- Gestion améliorée des connexions aux bases de données dans le framework SQL
- Clés de gestion de secrets normalisées pour AWS plusieurs sources de données
- Correctifs de régression des performances
- Fonction de vérification améliorée pour PackedType
- Gestion améliorée de LOW-VALUE pour PackedType
- Emballage de sécurité à ressort amélioré pour la connexion Cognito
- Ne pas appliquer le codage et le décodage par point de transfert de code sur les bases de données ciblées DB2

Troisième partie

- Mise à niveau de Spring Boot de la version 2.5 à la version 2.7

Outils de modernisation, version 3.8.0

ZoS

Nouvelles fonctionnalités

- JCL : Gestion du flux avec le retour du chariot « \ r »

Améliorations

- Journalisation améliorée pour empêcher la division par zéro lors de la modernisation d'une clause DIVIDE with ON SIZE ERROR
- JCL : support amélioré pour appeler une procédure dans une procédure
- Support du mot clé OF dans la commande FORMATTIME CICS en cas de champs ambigus
- JCL : prise en charge du caractère Â¥ dans les variables
- JCL : calcul de la RC en fonction des étapes précédentes
- Comparaison d'octets au lieu de chaînes lorsque PL1 SUBSTR est utilisé

- Amélioration de l'initialisation de tableaux multidimensionnels à partir d'une source unique
- Analyse améliorée du COBOL lorsqu'il implique une seule requête SQL dans un bloc IF

AS400

Nouvelles fonctionnalités

- Support de l'instruction IF imbriquée dans CL
- Support amélioré pour l'instruction ENDDO dans RPG Freeform

Améliorations

- Support amélioré pour le niveau de contrôle du conditionnement
- Retour du prototype amélioré avec LIKE
- Support amélioré pour les fonctions de gestion %months, %year, %days
- Support de la fonction d'aide pour l'ensemble de l'écran
- Gestion des BLANKS figuratifs passés en paramètre
- Amélioration de l'expression EVAL avec l'opérateur « »
- Gestion de la commande START sans KEY PHASE
- Amélioration de la gestion du mot clé LIKEREC
- Amélioration des sous-champs anonymes
- Amélioration de la procédure renvoyant un type non signé
- Support amélioré pour le fonctionnement RESET (RPG gratuit), les fonctionnalités intégrées %CHAR et %DEC
- Amélioration de la fonction intégrée %LOOKUPXX
- Support amélioré pour le mot clé LIKEDS sur la procédure sans prototype
- Gestion du type de tableau de mots clés Dim (VAR, AUTO)
- Support amélioré pour XFOOT
- COBOL : prise en charge améliorée des champs RENAMES
- CL : support en condition (vraie)
- Amélioration de la gestion des tableaux autonomes avec le mot clé LIKE
- Amélioration de la fonction intégrée %INT

- Analyse complète et gratuite du RPG amélioré
- Support amélioré pour le tableau dans la liaison
- CL2GROOVY : Déclaration Support Select
- Amélioration du mot clé DSPF « ERRMSGID »
- Amélioration de la gestion de l'initialisation des octets avec des zéros en tête
- Amélioration des valeurs autorisées pour les champs numériques
- Manipulation de l'extendeur H pour une instruction EVAL au format libre
- CL to Groovy : Support de la sous-chaîne LDA
- Support amélioré pour la réinitialisation sur un enregistrement
- Amélioration de la gestion de l'EDTCDE et de l'EDTWRD avec les références
- Mappage des champs de saisie amélioré avec les champs DDS
- Support amélioré pour déplacer un personnage dans le tableau IN
- Amélioration du prototype avec le mot clé LIKEDS
- Support amélioré pour le mot clé DSPF DSPATR
- Analyse améliorée de la carte D avec +/-
- Robustesse accrue dans les appels de programmes
- Robustesse accrue dans le processus de résolution sur le terrain

Capacités transversales

Améliorations

- FrontEnd: simuler un événement de collage pour une entrée IME

Troisième partie

- Mise à niveau de Spring Boot de la version 2.5 à la version 2.7

Notes de mise à jour 3.7.0

Cette version des outils d'exécution et de modernisation de AWS Blu Age inclut principalement des améliorations visant à mieux prendre en charge les commandes et les utilitaires, des fonctionnalités

d'intégration à AWS Secrets Manager et de nouvelles fonctionnalités de surveillance. Certains des principaux changements apportés à cette version sont les suivants :

- Plusieurs composants d'exécution peuvent désormais utiliser AWS Secrets Manager pour améliorer la configuration de sécurité des applications modernisées, principalement liées aux sources de données des services publics, aux files d'attente Redis pour TS, au BluSam cache et aux verrous.
- Point de terminaison de surveillance qui permet de récupérer les métriques des transactions, des lots et de la JVM pour l'optimisation de l'utilisation des ressources et la gestion opérationnelle, telles que le statut, la durée, le volume, etc.
- Nouvelles fonctionnalités pour prendre en charge les appels IBM MQ dans les RPG et augmentation de la couverture de transformation de JCL SORT et IDCAMS.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.7.0

Rubriques

ZoS

Nouvelles fonctionnalités

- Améliorez l'analyse des requêtes impliquées dans les applications utilitaires du programme en utilisant du langage SQL similaire à la grammaire. (V7-9401)
- Gérer un tableau de taille variable indexé lorsqu'il est décalé (V7-9904)
- Support de la colonne INSERT SQL TIME dans DB2 au format 24:00:00 (V7-10023)
- Support de la requête SQL INSERT à partir de tableaux avec les options FOR ROWS et ATOMIC (V7-10105)
- JCL SORT - amélioration TranscodeTool pour prendre en charge OUTREC avec IFTHEN (V7-10124)

- JCL SORT - ajout de la prise en charge du mot clé DATE dans la commande OUTREC (V7-10125)
- JCL - ajout de la prise en charge des procédures In-Stream (V7-10223)

Améliorations

- Un ensemble de données marqué de la disposition « PASS » doit être disponible pour toutes les étapes du travail (V7-9504)
- Support de l'attribut JCL SCHENV (V7-9570)
- Support SEND avec option CTLCHAR (V7-9714)
- COBOL - Gérer différents jeux de caractères séparateurs de lignes dans les instructions ACCEPT (V7-9875)
- Évitez les annulations multiples (V7-9958)
- Autoriser l'utilisation de la disposition MOD à ajouter à la fin des fichiers GDG (V7-10031)
- Optimisation : refactorisation de PutAll (V7-10063)
- PutAll refactorisation : ajout de pagination (V7-10063)
- Rendre le délai de lecture du client Jedis configurable (V7-10063)
- UseSsl prise en charge du mode autonome (V7-10114)
- Support EIBDS après ouverture réussie du fichier (V7-10147)
- Support EIBDS après une demande de contrôle de fichier (V7-10147)
- Améliorer le support de CICS SYNCPOINT (V7-10187)
- BluesamRedisSerializer: problème avec MetadatapPersistence (V7-10202)
- Support de Redis AWS Secrets Manager pour les files d'attente TS (V7-10204)
- Support JCLBCICS sur la personnalisation de la taille du nom du DD (V7-10224)
- Ajoute la prise en charge du chemin absolu dans l'instruction IDCAMS DELETE (V7-10308)

AS400

Nouvelles fonctionnalités

- Implémentation de la fonction d'aide pour les écrans AS400 (V7-9673)

Améliorations

- Nombre d'enregistrements dans l'INFDS (V7-9377)

Capacités transversales

Nouvelles fonctionnalités

- Support pour Runtime on EC2 pour envoyer des journaux à Amazon CloudWatch (D87990246)
- Ajout d'un nouveau point de terminaison pour récupérer les métriques relatives aux lots, aux transactions et à la JVM (D88393832)

Améliorations

- Support des sources de données AWS Secrets Manager pour l'utilitaire pgm (V7-9570)
- Ajout du support DB2 pour DSNUTILB DISCARD (V7-9798)
- Support pour l'écriture dans le logger au lieu du flux de sortie système par défaut dans les fichiers SYSPRINT et SYSPUNCH par défaut (V7-10098)
- Support du cache BluSam Redis et des propriétés de connexion verrouillées dans AWS Secrets Manager (V7-10238)
- Support de la connexion SSL sur Db2 XA AWS secret (V7-10258)
- Metadonnées mises à jour pour IDCAMS REPRO et VERIFY (V7-10281)
- Gestion améliorée du code de retour IDCAMS Abend (V7-10307)

Outils de modernisation, version 3.7.0

Rubriques

ZoS

Nouvelles fonctionnalités

- PLI - Affectation améliorée pour la section transversale des matrices et les matrices bidimensionnelles (V7-9830)

AS400

Nouvelles fonctionnalités

- Manipulation des indicateurs de niveau de commande (V7-9227)
- Support du paramètre EXTNAME *INPUT (V7-9897)
- Réécriture Goto améliorée : Support pour les balises situées dans les instructions SELECT OTHER (V7-9973)
- Support du mot clé REFSHIT DSPF (V7-10049)

Améliorations

- Amélioration de la gestion du mot clé de description de fichier EXTIND (*INux) (V7-7404)
- Transformation de fichiers SQLDDS améliorée (V7-7687)
- Les objets de fichier ne sont plus générés pour les fichiers AS400 (V7-9062)
- Gestion améliorée du mot-clé de description de fichier EXTDESC (V7-9268)
- Gestion améliorée de la version intégrée de %CHAR (V7-9311)
- Support amélioré pour le défilement de page sur le dernier enregistrement sans SFLEND (V7-9322)
- Support amélioré pour les structures de données préfixées (V7-9436)
- Support pour les dimensions définies avec %SIZE (V7-9472)
- Support pour la gestion du nom de champ PF déclaré entre guillemets (V7-9557)
- Fonctionnement des fichiers amélioré, insensible aux majuscules et minuscules (V7-9785)
- Support pour le champ initialisé à *USER (V7-9806)
- Support du type COMP dans l'AS400 (V7-9840)
- Analyse COBOL400 améliorée sur (Non) (V7-9922) InvalidKey
- Gestion améliorée de l'opération SCAN (V7-9971)
- Support amélioré de l'opcode GOTO (V7-9973)
- Gestion améliorée du fonctionnement EXCEPT (V7-9977)
- Support de préfixe amélioré (V7-10000)
- Support pour les appels MQ en RPG (V7-10007)

- Intégration %LOOKUP améliorée (structure de données de tableau à clés) (V7-10022)
- Support pour les opérations Close *All (V7-10036)
- Support de l'instruction UPDATE AS ROW CHANGE SQLDDS (V7-10051)
- Amélioration de la gestion de la valeur littérale de type Long (V7-10073)
- Grammaire RPG améliorée (utilisation du mot-clé INZ comme nom du sous-programme) (V7-10074)
- Grammaire RPG améliorée pour prendre en charge les valeurs numériques avec une partie fractionnaire vide (V7-10077)
- Support amélioré pour les champs partagés entre CL et un fichier externe (V7-10081)
- Support amélioré pour les indicateurs conditionnels DDS (V7-10084)
- Support du type binaire DDS avec les programmes COBOL (V7-10100)
- Amélioration de la collision de noms avec le lien (V7-10109)
- Support pour les procédures de mixage principales et d'exportation (V7-10112)
- Support amélioré DataStructure dans une sous-procédure (V7-10113)
- Support amélioré de CLEAR (V7-10126)
- Support amélioré de la boucle DO (V7-10134)
- Support de SQLTYPE dans un RPG entièrement gratuit (V7-10151)
- Analyse améliorée des conditions sur le mot clé DDS (V7-10155)
- Génération DSL améliorée (V7-10163)
- Amélioration de ProcessIndicators lorsque la condition est une expression binaire. (V7-10164)
- GoTOS amélioré avec condition Else (V7-10168)
- Support pour le type Heure et horodatage dans le DSPF (V7-10173)
- Analyse améliorée de la ligne de continuation pour DDS (V7-10183)
- Support COBOL pour RENAMES FLD OF RECORD (V7-10195)
- Analyse des indicateurs conditionnels améliorée sur les champs DSPF (V7-10221)
- Support de l'analyse syntaxique du mot-clé DDS NOALTSEQ (V7-10288)
- Support : menu d'aide et champs masqués (V7-10314)
- Vérification améliorée de l'intégrité des mots clés d'aide DSPF (V7-10328)
- Ne plus propager tous les mots clés dans le champ Ref (V7-10347)

Capacités transversales

Nouvelles fonctionnalités

- Data Migrator - Gestion des données CLOB (V7-9665)

Améliorations

- Propagation de la propriété JCL SCHENV depuis la définition JOB vers PROC GROOVY via (V7-10225) JobContext
- FrontEnd - Réglage de la taille de la fenêtre en cas d'absence de bordure (V7-10358)

Notes de mise à jour 3.6.0

Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les migrations existantes vers zOS et AS400, principalement destinées à étendre les mécanismes de support CICS, à compléter les fonctionnalités JCL, à optimiser les performances des fonctionnalités simultanées et à volume élevé, et à ajouter des fonctionnalités. multi-data-source Certains des principaux changements apportés à cette version sont les suivants :

- Amélioration de la gestion dynamique des fichiers JCL, extension des instructions actuelles et gestion des ensembles de données concaténés, exécution de plusieurs instructions dans un seul bloc et transfert de données des lots vers les programmes.
- Prise en charge améliorée de plusieurs commandes CICS, y compris la recherche de plusieurs types de ressources CICS.
- La possibilité de disposer de différentes bases de données lors de l'utilisation de Blu Age Runtime Utilities, idéale pour les scénarios dans lesquels les données commerciales sont distribuées entre plusieurs sources.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.6.0

Rubriques

- ZoS

- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- JCL - DynamicFileBuilder - Gestion améliorée des descripteurs de fichiers (V7-9408)
- Conversion de format améliorée sur certaines fonctions SQL DB2 intégrées lors de l'appel de l'utilitaire INFUTILB UNLOAD (V7-9554)
- Affectations de réseaux multidimensionnels PLI améliorées (V7-9592)
- Gestion de la redirection sysout vers un fichier (V7-9992)

Améliorations

- Ajout du déclenchement de procédures stockées pour les SGBDR DB2 (V7-9155)
- SORT gère la conversion au format PDF (V7-9286)
- JCL/GROOVY - Amélioration de l'instruction REPRO pour prendre en charge les ensembles de données DUMMY (V7-9424)
- Améliorer le support CICS UNLOCK (V7-9606)
- Gérer la taille de valeur par défaut pour Union (V7-9648)
- JCL/GROOVY gèrent différentes terminations/dispositions dans les ensembles de données concaténés (V7-9653)
- Rendre PageSize configurable pour les ensembles de données Blusam (V7-9680)
- DSNUTIL - autorise le chargement de 24:00:00 comme heure valide dans DB2LUW (V7-9697)
- Support de comparaison de valeurs élevées (0xff) dans NumberUtils .ne ()/ NumberUtils.eq () (V7-9731)
- JCL/GROOVY - supporte DO... Mots-clés THEN dans les clauses IDCAMS IF-THEN-ELSE pour exécuter plusieurs instructions dans un seul bloc (V7-9750)
- JHDB non valide a appelé un programme en dehors de BatchRunner JHDB (V7-9782)
- Support des espaces blancs dans la carte de contrôle SORT OUTFIL (V7-9808)
- Améliorer le support CICS READ PREV (V7-9845)

- Améliorer l'accès simultané aux index des ensembles de données (V7-9864)
- Améliorer le support CICS REWRITE (V7-9873)
- COBOL : prise en charge des instructions SYSIN multilignes dans ACCEPT pour transmettre des données d'un lot (JCL) à un programme (COBOL) (V7-9875)
- Groovy - Meilleure gestion de l'étape de ConcatenatedFileConfiguration création des fichiers (V7-9876)
- UTILITAIRE IDCAMS - Gestion de l'instruction DEFINE PATH (V7-9878)
- SORT BUILD - Ajustez l'option TRAN et gérez les blancs implicites (V7-9925)
- Améliorez CICS DELETE avec le support des options GENERIC (V7-9939)
- Améliorez le support CICS STARTBR et ENDBR (V7-9952)
- Améliorez les performances de clôture en cas d'accès simultané (V7-9953)
- Améliorer la gestion de l'état des fichiers au démarrage (V7-9991)
- Groovy - Autorise l'appel de GetDisposition ()/()/getNormalTermination() sur getAbnormalTermination ConcatenatedFileConfiguration (V7-10012)

AS400

Nouvelles fonctionnalités

- Support des indicateurs externes sur les mots clés COMMIT (V7-6035)
- Réinitialisez la boucle ReadC après l'écriture SFLCTL (V7-8061)
- Support de l'indicateur LR dans CALL (V7-9250)
- Ajout d'un nouveau type de champ dynamique (fractionné) pour gérer le champ de saisie sur plusieurs lignes (V7-9370)
- Support du fichier primaire/secondaire (V7-9390)
- Les zones de données locales sont désormais transmises à la tâche appelée lors de la soumission d'une tâche (V7-9775)
- Support de QTEMP pour la zone de données et prise en charge de la création de valeur de la zone de données. (V7-9916)
- Contrôle des engagements : support pour activer/désactiver le contrôle des engagements (V7-9956)
- Support des indicateurs externes sur les mots clés COMMIT

Améliorations

- Amélioration de l'affichage de la valeur 0 et de l'EDTWRD (V7-8933)
- Support du mot-clé DSPF « CHKMSGID » (V7-9125)
- Transaction de validation SQL à la fin du lot (V7-9232)
- Améliorer la prise en charge des mots clés EXPORT et IMPORT pour le champ et la structure des données (V7-9265)
- Support en minuscules DateHelper (V7-9461)
- Support de conversion *CYMD en *ISO (numérique) (V7-9488)
- Améliorer la gestion du %len intégré pour un champ variable (côté gauche et côté droit d'une expression) (V7-9733)
- Amélioration de la prise en charge des fonctions intégrées '%LOOKUPXX' XX (« LE », "LT », "GE », "GT ») (V7-10064)

Capacités transversales

Nouvelles fonctionnalités

- CICS - Améliore la transaction de demande de renseignements pour connaître le statut des options (V7-9712)
- JCL - Améliore le chargement pour Sysprint avec le fichier de sortie système (V7-9797)
- CICS - Améliore INQUIRE TSQUEUE (V7-9823)
- CICS - Améliore le terminal de demande pour l'option userid (V7-9906)

Améliorations

- Améliorez le maniement de la comparaison avec le blanc (V7-8047)
- Améliorer la journalisation pour Jics et BluSam (V7-8847)
- Support des attributs étendus SOSI du BMS et du symbole programmé F8 pour les champs dynamiques (V7-8857)
- Gérer le dépassement de la mémoire tampon dans les paramètres du programme (V7-9138)
- Améliorer la simultanéité des threads et des écritures pour le registre des verrous Blusam (V7-9505)

- Support de configuration de sources de données multiples pour Utility-PGM (V7-9570)
- Mode de verrouillage du niveau d'enregistrement Blusam uniquement (V7-9626)
- Assurez-vous que la persistance des métadonnées résiste au redémarrage du serveur (V7-9748)
- Améliorer le nettoyage du DAO en cas d'exception (fermeture du navigateur) (V7-9790)
- Support DummyFile pour INFUTILB SYSPUNCH (V7-9799)
- Améliorer la prise en charge des valeurs négatives sur NumericEditedType (V7-9935)

Outils de modernisation, version 3.6.0

Rubriques

ZoS

Nouvelles fonctionnalités

- JCL - Amélioration de la journalisation pour la fin de la procédure (V7-8509)
- PL1 - Amélioration de la génération de sacs pour le type de données PakedLong (V7-8917)
- JCL - Améliore la journalisation pour la fin de la procédure lorsque le fichier contient le marqueur « fin »//(V7-9509)
- PL1 - Amélioration de la prise en charge de GET EDIT avec Fixed-point et SYSIN stream (V7-9593)
- DB2 - Support amélioré pour le type VARGRAPHIC DB2 (V7-9809)
- CICS - Améliore la sécurité des requêtes de commande pour l'option LOGMESSAGE (V7-9969)
- PL1 - Améliore la génération de sacs pour l'intégration de charge/Chargraphic (V7-9989)

Améliorations

- PL1- Améliorer la prise en charge du mot clé INCLUDEX (V7-9588)
- PL/I - Gère le mot clé CHARGRAPHIC comme paramètre valide de tout appel de méthode (V7-9589)

- Amélioration de la résolution des variables hôtes PL1 lorsqu'elles sont nommées avec des caractères spécifiques @ # $ §. (V7-9654)
- COBOL - Support des mots clés C01... C12 et S01... S05 en tant que paramètre de l'instruction WRITE ADVANCING à l'étape d'analyse (V7-9669)

AS400

Nouvelles fonctionnalités

- Support de la transformation SQL-DDS dans Analyzer (V7-7687)
- Automatiser la détection des fichiers SQL-DDS (V7-7687)
- Implémentation du prétraitement SQL-DDS (V7-7687)
- Support du mot clé ALIGN (V7-9254)
- Support du ExtName DSPF et de la matrice multi-DIM (V7-9663)
- InvalidKey Déclarations de support sur COBOL WRITE (V7-9793)

Améliorations

- Amélioration de l'opcode TESTB (V7-8865)
- Améliorer le support du DECFMT on focus (V7-8933)
- Gestion de l'indicateur résultant sur MOVE (V7-9224)
- Améliorer la prise en charge du modèle de mot clé pour le champ et la structure de données (V7-9278)
- Amélioration de LIKEDS (le DS défini à l'aide de LIKEDS est automatiquement qualifié) (V7-9302)
- COBOL - Améliorer la génération de la structure des indicateurs (V7-9423)
- Le paramètre Const du prototype n'est pas en lecture seule (V7-9437)
- Améliorez le mot clé EDTCDE avec le code d'édition « Y » (V7-9443)
- Support de génération du champ *ROUTINE dans PSDS et INFDS (V7-9487)
- Améliorer le champ de réécriture XXX pour le rendre autonome (la valeur par défaut est perdue lors de la réécriture) (V7-9522)
- Support amélioré des mots clés DSPF (V7-9658)
- Gestion de la valeur par défaut ZEROES sur le binaire (V7-9666)
- Support du pointeur implicite (V7-9719)

- Améliorez la gestion de l'appel intégré %size avec un seul paramètre (V7-9730)
- Améliorez le traitement des références de structure de données dans les appels intégrés (%ELEM) (V7-9736)
- Améliorer la gestion de la longueur signée pour le champ avec une référence LIKE dans la spécification de définition (V7-9738)
- Amélioration de REWRITE (V7-9791)
- Amélioration de la génération d'index à partir de fichiers DDS (V7-9803)
- Améliorez la robustesse des mappeurs avec une valeur numérique non valide (V7-9813)
- Améliorer la génération de fichiers SQLModel et AllIndexes (V7-9818)
- Améliorez le support DS qualifié (V7-9863)
- Amélioration de la prise en charge de LOOKUP (avec un champ autonome COMME un DS en paramètre) (V7-9961)
- Améliorer le LIKE sur l'indicateur (V7-9985)
- Gestion de l'indicateur résultant sur le MVR (V7-9995)
- Support du caractère N avec tilde (V7-10021)
- Améliorez la génération de fichiers DDL modernes à partir des anciens fichiers SQLDDS (V7-10067)

Capacités transversales

Nouvelles fonctionnalités

- Personnaliser l'emplacement des ressources avec une propriété yml (D88816105)
- COBOL - Support de l'instruction EXIT PERFORM pour quitter un PERFORM en ligne sans utiliser de GO TO/PERFORM... VIA (V7-9582)
- Spécifier le codage existant par défaut à prendre en compte dans les métadonnées globales. (V7-9883)

Améliorations

- Améliorer la génération de masques (V7-9602)
- Améliorer l'échauffement du contexte (V7-9621)
- Sécurisez le thread Charset CUSTOM930. (V7-9674)

- Amélioration apportée à MOVEA (V7-9773)

Notes de mise à jour 3.5.0

Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les migrations existantes vers zOS et AS400, principalement orientées vers les ensembles de données et l'optimisation de la messagerie, ainsi que des fonctionnalités Java étendues en tant qu'atout du processus de transformation. Certains des principaux changements apportés à cette version sont les suivants :

- Possibilité de migrer des programmes CL vers Java en plus de la fonctionnalité préexistante de scripts groovy, afin de faciliter son intégration avec d'autres programmes modernisés et de simplifier la courbe d'apprentissage des clients en unifiant le langage de programmation qui en résulte.
- Réduction du temps et optimisation des performances des chargements de jeux de données dans Redis grâce à la nouvelle fonctionnalité de masse de données.
- Capacité à exploiter et à transmettre des ensembles de données au cours des étapes de travail afin de moderniser les comportements traditionnels des ensembles de données.
- Extension de la migration SQL pour prendre en charge les fichiers d'entrée VB et migration simplifiée avec Java 11.
- Plusieurs nouveaux mécanismes pour une intégration plus rapide avec IBM MQ, notamment des en-têtes supplémentaires, un support GET/PUT étendu et la récupération automatique des métadonnées des files d'attente.
- Point de terminaison REST pour les métadonnées des ensembles de données et l'importation de jeux de données à partir de compartiments S3.

Pour plus d'informations sur les modifications incluses dans cette version, consultez les sections suivantes.

Runtime version 3.5.0

Rubriques

- ZoS
- AS400
- Capacités transversales

ZoS

Nouvelles fonctionnalités

- JCL SORT - Gérer la nouvelle superposition de mots clés (V7-9409)
- ZOS COBOL - amélioration de la prise en charge des caractères flottants (V7-9404)
- Port de RedisJics TSqueue vers RedisTemplate & ListOperations (V7-9212)
- ZOS JCL - améliore le chemin du répertoire temporaire avec le répertoire des fichiers s'il est défini via UserDefinedParameters (V7-9012)
- Manipulez la FONCTION ORD-MAX avec TOUS (tous les éléments de la gamme) (V7-9366)
- Des clés préfixées et lisibles par l'homme sont désormais utilisées lors du stockage des files d'attente TS dans Redis (V7-9212)
- Ajouter le point de terminaison Get Dataset pour l'API Blusam
- JCL - AJOUT du support pour les tâches par lots dont le nom comporte un caractère spécial tel que # (V7-9136)
- La récupération de TsModel est désormais effectuée de manière robuste à la demande (V7-9212)

Améliorations

- Support INCLUDE non versionné dans les fichiers LNK (V7-6022)
- MQ - Support d'encodage amélioré (V7-9652)
- Amélioration de la prise en charge des octets doubles ou des jeux de caractères mixtes pour différents types de caractères (V7-9596)
- JCL - Support de la configuration du répertoire de fichiers dans IDCAMS (suppression des instructions NONVSAM) (V7-9609)
- Support du mode masse pour le chargement d'ensembles de données ESDS et RDS à partir de fichiers (V7-8639)
- Gérez l'ouverture d'un ESDS vide en mode entrée. (V7-9287)
- Améliorez l'instruction DEFINE CLUSTER avec la prise en charge des abréviations ORD/UNORD (V7-9451)
- BluSam Améliorations des performances de Redis Lock (V7-8639)
- Améliorez l'instruction DEFINE CLUSTER pour prendre en charge RECORDSIZE fourni dans la portée de l'argument DATA () (V7-9337)

- Ajoute la prise en charge des attributs BUFFERSPACE/UNIQUE sur les instructions DEFINE CLUSTER (V7-9419)
- Améliorez les opérations de BluSam lecture pour les ensembles de données d'enregistrements de longueur variable. (V7-9391)
- L'ADRESSE CICS représente correctement le CWA manquant comme nul (V7-9491)
- Supprimer les écritures inutiles aux extrémités des verrous (V7-8639)
- Gérer l'injection de modèles de cache Redis dans le cache (V7-9510)
- Décode correctement le paramètre BPXWDYN (V7-9417)
- Amélioration de la consommation d'exportations LISTCAT (V7-9201)
- Prise en charge des caractères non imprimables dans le nom BluSam des files d'attente TS (V7-9212)
- Gérer la création de cartes de réception pour le champ avec mapset null (V7-9486)
- Améliorez BluesamRelativeFile les opérations de suppression et de réécriture pour le mode d'accès dynamique. (V7-8989)

AS400

Nouvelles fonctionnalités

- Ajout d'une fonctionnalité permettant de générer des fichiers CL sous forme de programmes Java via le pivot DS/STM standard (V7-9427)
- Support du fichier d'entrée avec le mode ADD (V7-9378)
- Amélioration de la gestion de l'ordre de tri et de la récupération pour prendre en charge la commande cl OPNQRYF (Open Query File) et ajout de la prise en charge du paramètre SHARE dans. OverrideItem (V7-9364)

Améliorations

- Support SFLNXTCHG sur (V7-8061) UpdateSubfile
- Modifier la portée du contexte CL lors de l'exécution de la commande CL (V7-9624)
- Gérer le code de retour pour le programme BPXWDYN (V7-9417)
- Effacez les moniteurs locaux. (V7-9624)
- Support du mot clé DSPF RTNCSRLOC (V7-9389)

- setOnGreaterOrEqual() ne définit pas la valeur égale à 1 (V7-9342)
- Mettre à jour le cache des champs sur UpdateSubfileRecord (V7-9376)
- Améliorer le support SFLNXTCHG (V7-8061)

Capacités transversales

Nouvelles fonctionnalités

- Ignorez le préfixe G sur la chaîne graphique littérale. (V7-9420)
- ZOS COBOL - Amélioration de la prise en charge de Fiedl.initialize () pour certaines structures spéciales (V7-9485)
- Autoriser l'initialisation du contexte de manière asynchrone pour améliorer les performances de démarrage du programme (V7-9446)
- SQL Release explicitement l'instruction prepare ouverte et ResulSet. (V7-9422)
- Améliorez JMS MQ - support MQRFH2 pour MQ PUT/V7-7085 - support du gestionnaire de files d'attente par défaut (V7-9400)
- SQL Management - Activer les conversions Lambda sur les paramètres des commandes SET (V7-9492)
- ZOS MQ JMS - Ajout du support à MQCOMIT et MQBACK (V7-9399)
- ZOS IBMMQ - Amélioration de la prise en charge de MQINQ (V7-9544)
- Gérez l'opération CONCAT avec un octet au lieu d'une chaîne lors de l'utilisation d'un codage à double octet. (V7-8932)
- ZOS IBMMQ - Amélioration de la prise en charge de la commande PUT avec les options SET_ALL_CONTEXT (V7-9544)

Améliorations

- Gérer les noms de fichiers gdg avec le caractère $ (V7-9066)
- SQL Diagnostic renvoie 1 sous forme de clause NUMBER lorsque l'instruction SQL précédente est réussie. (V7-9410)
- Schéma d'un champ dont la longueur n'est pas nulle (V7-7536)
- Support de la fonction GRAPHIQUE PL1 intégrée (V7-9245)
- MQ - Ajout du support de version pour le réglage des champs MQGMO (V7-9500)

- JMS MQ GET - Amélioration de la longueur des données renvoyées par le message (V7-9502)
- Définissez sqlerrd (3) avec le nombre d'éléments récupérés dans le contexte ROWSET. (V7-9371)

Outils de modernisation, version 3.5.0

Rubriques

ZoS

Nouvelles fonctionnalités

- ZOS PLI - Support de l'index astérisque lors de l'assignation avec une expression binaire (V7-9178)
- JCL to BatchScript - Un «//» marque la fin de l'exécution de la tâche (V7-9304)
- ZOS PLI - amélioration de la prise en charge des caractères flottants et des signatures en type numérique édité (V7-8982)
- COBOL - Support de la fonction SUM intégrée (V7-9367)
- JCL- éventuellement, commentez le code mort après une instruction nulle (//) (V7-9202)
- JCL- Support de l'opérateur « | » dans la déclaration de condition (V7-9499)
- PL/I - Commentaire des directives de précompilation lors de l'étape de prétraitement pour empêcher les exceptions d'analyse syntaxique (V7-9507)

Améliorations

- Gérer la définition du flux avec un délimiteur (V7-9615)
- Amélioration de la gestion des exportations LISTCAT. (V7-9201)
- PL/I - Amélioration pour prendre en charge les arguments « nuls » implicites (V7-9204)

AS400

Nouvelles fonctionnalités

- Support du mot clé DDS CONCAT (V7-9439)
- Refactorisez le code Java généré pour les mots clés DSPF. (V7-7700)
- Support : mots clés variables dans les champs d'une définition de structure de données (V7-9029)

Améliorations

- Améliorer l'analyse de la relation logique ET/OU (V7-9352)
- COBOL Améliore le mappage entre vo et DSentity (V7-9449)
- Afficher une valeur vide si la saisie numérique est focalisée (V7-9374)
- Variable locale dans le curseur de déclaration SQL (V7-9456)
- Problème de portée avec un DS vide (V7-9466)
- Tronquer les lignes après col 80 avant l'analyse (V7-9632)
- Améliorez la gestion des références de champ et des appels intégrés dans les mots clés (DIM, LIKE,...) dans la spécification de définition (V7-9358)
- Support des commentaires SQL (--) (V7-9632)
- FullFree analyse, tapez Date/Heure/Horodatage (V7-9542)
- Inclure le SQLCA à partir de l' FullFree analyse syntaxique (V7-9333)
- Support amélioré du niveau de contrôle. (V7-9610)
- Comparaison de Handle DS avec *BLANKS (V7-9668)
- Améliorer la prise en charge de plusieurs indicateurs dans le DDS (V7-9318)
- Améliorer la prise en charge de plusieurs programmes DSPF (V7-9657)
- Améliorez la gestion des champs avec LIKE (cas d'une structure de données aimée et cas d'une structure de données aimée dans un tableau) (V7-9213)
- RPG gratuit, continuez à gérer au pied de la lettre (V7-9686)
- Support amélioré des dossiers de fin de programme (V7-9452)
- Support de la phrase LINKAGE dans l'instruction CALL. (V7-9685)
- Code d'opération CASXX (CASBB sans groupe CASXX) (V7-9357)
- Améliorer l'analyse FullFree des RPG (V7-9457)
- Le %LEN intégré ne prend pas en charge DS comme argument (V7-9267)
- Améliorations de MOVEA lorsque le facteur 2 est *ALL'X... ' (V7-9228)

- Support : attribution avec champ RENAME (V7-9385)

Capacités transversales

Nouvelles fonctionnalités

- Outil SQL Migrator : ajoutez une option OID pour une longueur d'enregistrement variable lors de l'étape de chargement d'ebcdic. (V7-9380)
- Outil SQL Migrator - Support de Java 11 sur l'option OID (V7-9599)

Améliorations

- Amélioration de la prise en charge des baies imbriquées (V7-9595)
- Remplacez Â¬ caractère par ! dans le cas de Â¬ est supporté par le codage d'origine. (V7-9465)
- JCL - Support de la terminaison normale du PASS pour partager des ensembles de données entre les étapes de travail (V7-9504)
- Appliquez ON NULL à la définition de colonne sur ORACLE lorsqu'il s'agit de VARCHAR et de type de colonne de base de données nullable. (V7-9681)
- Améliorez la conformité des injections à ressort (V7-9635)

AWS Concepts d'exécution de Blu Age

Comprendre les concepts de base du AWS Blu Age Runtime peut vous aider à comprendre comment vos applications sont modernisées grâce au refactoring automatique.

Rubriques

- AWS Architecture de haut niveau Blu Age Runtime
- AWS Structure Blu Age d'une application modernisée
- Simplificateur de données

AWS Architecture de haut niveau Blu Age Runtime

Faisant partie de la solution AWS Blu Age pour la modernisation des programmes existants vers Java, le AWS Blu Age Runtime fournit un point d'entrée unifié basé sur REST pour les

applications modernisées, ainsi qu'un cadre d'exécution pour ces applications, par le biais de bibliothèques fournissant des structures héritées et d'une standardisation de l'organisation du code des programmes.

Ces applications modernisées sont le résultat du processus AWS Blu Age Automated Refactor visant à moderniser les programmes centraux et de milieu de gamme (appelés « anciens » dans le document suivant) vers une architecture basée sur le Web.

Les objectifs de AWS Blu Age Runtime sont la reproduction du comportement des programmes existants (isofonctionnalité), les performances (en termes de temps d'exécution des programmes et de consommation de ressources) et la facilité de maintenance des programmes modernisés par les développeurs Java, grâce à l'utilisation d'environnements et d'expressions familiers tels que tomcat, Spring, getters/setters, API fluides...

Rubriques

AWS Composants d'exécution Blu Age

L'environnement d'exécution AWS Blu Age est composé de deux types de composants :

- Ensemble de bibliothèques Java (fichiers JAR) souvent référencées sous le nom de « dossier partagé » et fournissant des constructions et des instructions héritées.
- Ensemble d'applications Web (fichiers de guerre) contenant des applications Web basées sur Spring fournissant un ensemble commun de cadres et de services aux programmes modernisés.

Les sections suivantes décrivent en détail le rôle de ces deux composants.

AWS Bibliothèques Blu Age

Les bibliothèques AWS Blu Age sont un ensemble de fichiers jar stockés dans un `shared/` sous-dossier ajouté au classpath standard de Tomcat, afin de les rendre accessibles à tous les programmes Java modernisés. Leur objectif est de fournir des fonctionnalités qui ne sont ni nativement ni facilement disponibles dans l'environnement de programmation Java, mais typiques

des environnements de développement existants. Ces fonctionnalités sont exposées d'une manière aussi familière que possible aux développeurs Java (getters/setters, API fluides basées sur des classes). Un exemple important est la bibliothèque Data Simplifier, qui fournit aux programmes Java des structures de configuration et de manipulation de mémoire héritées (présentes dans les langages COBOL, PL1 ou RPG). Ces fichiers JAR sont une dépendance essentielle du code Java modernisé généré à partir de programmes existants. Pour plus d'informations sur le simplificateur de données, consultezSimplificateur de données.

Application web

Les archives d'applications Web (WAR) constituent un moyen standard de déployer du code et des applications sur le serveur d'applications Tomcat. Ceux fournis dans le cadre du runtime AWS Blu Age visent à fournir un ensemble de frameworks d'exécution reproduisant les environnements existants et les moniteurs de transactions (lots JCL, CICS, IMS...), ainsi que les services requis associés.

Le plus important est `gapwalk-application` (souvent abrégé en « Gapwalk »), qui fournit un ensemble unifié de points d'entrée basés sur REST pour déclencher et contrôler l'exécution des transactions, des programmes et des lots. Pour de plus amples informations, veuillez consulter AWS API d'exécution Blu Age.

Cette application Web alloue des fils d'exécution Java et des ressources pour exécuter des programmes modernisés dans le contexte pour lequel ils ont été conçus. Des exemples de tels environnements reproduits sont détaillés dans la section suivante.

D'autres applications Web ajoutent à l'environnement d'exécution (plus précisément, au « Registre des programmes » décrit ci-dessous) des programmes émulant ceux disponibles et appelables depuis les anciens programmes. Les deux catégories importantes de ce type sont les suivantes :

- Émulation de programmes fournis par le système d'exploitation : les lots pilotés par JCL s'attendent notamment à pouvoir appeler une variété de programmes de manipulation de fichiers et de bases de données dans le cadre de leur environnement standard. Les exemples incluent`SORT`/`DFSORT`ou`IDCAMS`. À cette fin, des programmes Java reproduisant un tel comportement sont fournis et peuvent être appelés en utilisant les mêmes conventions que les anciens programmes.
- Les « pilotes », qui sont des programmes spécialisés fournis par le framework d'exécution ou le middleware comme points d'entrée. Par exemple`CBLTDLI`, les programmes COBOL qui s'exécutent dans l'environnement IMS dépendent pour accéder aux services liés à IMS (base de données IMS, dialogue utilisateur via MFS, etc.).

Registre des programmes

Pour participer à ces constructions, frameworks et services et en tirer parti, les programmes Java modernisés à partir des anciens programmes adhèrent à une structure spécifique documentée dans. AWS Structure Blu Age d'une application modernisée Au démarrage, le AWS Blu Age Runtime collectera tous ces programmes dans un « registre des programmes » commun afin qu'ils puissent être invoqués (et s'appeler mutuellement) par la suite. Le registre des programmes fournit un couplage souple et des possibilités de décomposition (étant donné que les programmes qui s'appellent entre eux n'ont pas besoin d'être modernisés simultanément).

Environnements d'exécution

Les environnements et chorégraphies traditionnels fréquemment rencontrés sont disponibles :

- Les lots pilotés par JCL, une fois modernisés en programmes Java et en scripts Groovy, peuvent être démarrés de manière synchrone (blocage) ou asynchrone (détachée). Dans ce dernier cas, leur exécution peut être surveillée via des points de terminaison REST.
- Un sous-système AWS Blu Age fournit un environnement d'exécution similaire à CICS grâce à :
  - un point d'entrée utilisé pour démarrer une transaction CICS et exécuter les programmes associés tout en respectant la chorégraphie des « niveaux de course » du CICS,
  - un stockage externe pour les définitions de ressources,
  - un ensemble homogène d'API Java fluentes reproduisant des `EXEC CICS` instructions,
  - un ensemble de classes enfichables reproduisant les services CICS, tels que les files d'attente de stockage temporaires, les files de données temporaires ou l'accès aux fichiers (plusieurs implémentations sont généralement disponibles, telles qu'Amazon Managed Service pour Apache Flink, Amazon Simple Queue Service ou RabbitMQ pour les files d'attente TD),
  - pour les applications destinées aux utilisateurs, le format de description d'écran BMS est modernisé pour devenir une application Web angulaire, et le dialogue « pseudo-conversationnel » correspondant est pris en charge.
- De même, un autre sous-système fournit une chorégraphie basée sur des messages IMS et prend en charge la modernisation des écrans d'interface utilisateur au format MFS.
- En outre, un troisième sous-système permet l'exécution de programmes dans un environnement de type iSeries, y compris la modernisation des écrans spécifiés au format DSPF (Display File).

Tous ces environnements s'appuient sur des services communs au niveau du système d'exploitation tels que :

- l'émulation de l'allocation et de la disposition de la mémoire héritées (Data Simplifier),
- reproduction basée sur un thread Java du mécanisme d'exécution des « unités d'exécution » COBOL et de transmission de paramètres (`CALL`instruction),
- émulation d'organisations plates, concaténées, VSAM (via le jeu de bibliothèques Bluesam) et GDG Data Set,
- accès aux magasins de données, tels que les RDBMS (`EXEC SQL`instructions).

Apatridie et gestion des sessions

L'une des fonctionnalités importantes du AWS Blu Age Runtime est de permettre des scénarios de haute disponibilité (HA) et d'évolutivité horizontale lors de l'exécution de programmes modernisés.

La pierre angulaire de cette situation est l'apatridie, dont un exemple important est la gestion des sessions HTTP.

Gestion des sessions

Tomcat étant basé sur le Web, un mécanisme important pour cela est la gestion des sessions HTTP (telle que fournie par Tomcat et Spring) et la conception apatride. En tant que telle, la conception de l'apatridie repose sur les éléments suivants :

- les utilisateurs se connectent via HTTPS,
- les serveurs d'applications sont déployés derrière un équilibreur de charge,
- lorsqu'un utilisateur se connecte pour la première fois à l'application, celle-ci est authentifiée et le serveur d'applications crée un identifiant (généralement dans un cookie)
- cet identifiant sera utilisé comme clé pour enregistrer et récupérer le contexte utilisateur vers/depuis un cache externe (magasin de données).

La gestion des cookies est effectuée automatiquement par le framework AWS Blu Age et le serveur Tomcat sous-jacent, ce qui est transparent pour l'utilisateur. Le navigateur Internet de l'utilisateur gérera cela automatiquement.

L'application Web Gapwalk peut stocker l'état de la session (le contexte) dans différents magasins de données :

- Amazon ElastiCache pour Redis
- Cluster Redis

- dans la carte mémoire (uniquement pour les environnements de développement et autonomes, ne convient pas à HA).

Haute disponibilité et apatridie

Plus généralement, l'un des principes de conception du framework AWS Blu Age est l'apatridie : la plupart des états non transitoires nécessaires pour reproduire le comportement des programmes existants ne sont pas stockés sur les serveurs d'applications, mais partagés via une « source unique de vérité » externe commune.

Les files d'attente de stockage temporaire ou les définitions de ressources de CICS sont des exemples de tels états, et les stockages externes typiques pour ces derniers sont les serveurs compatibles Redis ou les bases de données relationnelles.

Cette conception, combinée à l'équilibrage de charge et aux sessions partagées, permet de distribuer la plupart des dialogues destinés aux utilisateurs (OLTP, « Traitement transactionnel en ligne ») entre plusieurs « nœuds » (ici, les instances Tomcat).

En effet, un utilisateur peut exécuter une transaction sur n'importe quel serveur sans se soucier de savoir si le prochain appel de transaction est effectué sur un autre serveur. Ensuite, lorsqu'un nouveau serveur est créé (à cause du dimensionnement automatique ou pour remplacer un serveur non sain), nous pouvons garantir que tout serveur joignable et sain peut exécuter la transaction comme prévu avec les bons résultats (valeur renvoyée attendue, changement de données attendu dans la base de données, etc.).

AWS Structure Blu Age d'une application modernisée

Ce document fournit des détails sur la structure des applications modernisées (à l'aide des outils de refactoring de modernisation du AWS mainframe), afin que les développeurs puissent accomplir diverses tâches, telles que :

- navigation fluide dans les applications.
- développer des programmes personnalisés qui peuvent être appelés à partir des applications modernisées.
- refactoriser en toute sécurité des applications modernisées.

Nous supposons que vous possédez déjà des connaissances de base dans les domaines suivants :

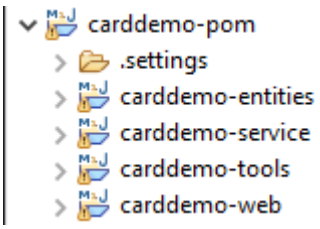
- les anciens concepts de codage courants, tels que les enregistrements, les ensembles de données et leurs modes d'accès aux enregistrements (indexés, séquentiels), le VSAM, les unités d'exécution, les scripts jcl, les concepts CICS, etc.
- codage Java à l'aide du framework Spring.
- Tout au long du document, nous l'utilisons `short class names` pour des raisons de lisibilité. Pour plus d'informations, voir AWS Mappages de noms entièrement qualifiés Blu Age pour récupérer les noms complets correspondants pour les éléments d'exécution de AWS Blu Age et Mappages de noms entièrement qualifiés par des tiers pour récupérer les noms complets correspondants pour les éléments tiers.
- Tous les artefacts et échantillons sont extraits des résultats du processus de modernisation de l'exemple d'application COBOL/CICSCardDemo .

Rubriques

Organisation des artefacts

AWS Les applications modernisées de Blu Age sont regroupées sous forme d'applications Web Java (.war), que vous pouvez déployer sur un serveur JEE. Généralement, le serveur est une instance Tomcat qui intègre le runtime AWS Blu Age Velocity, qui est actuellement basé sur les frameworks Springboot et Angular (pour la partie interface utilisateur).

La guerre regroupe plusieurs artefacts de composants (.jar). Chaque fichier jar est le résultat de la compilation (à l'aide de l'outil maven) d'un projet Java dédié dont les éléments sont le résultat du processus de modernisation.



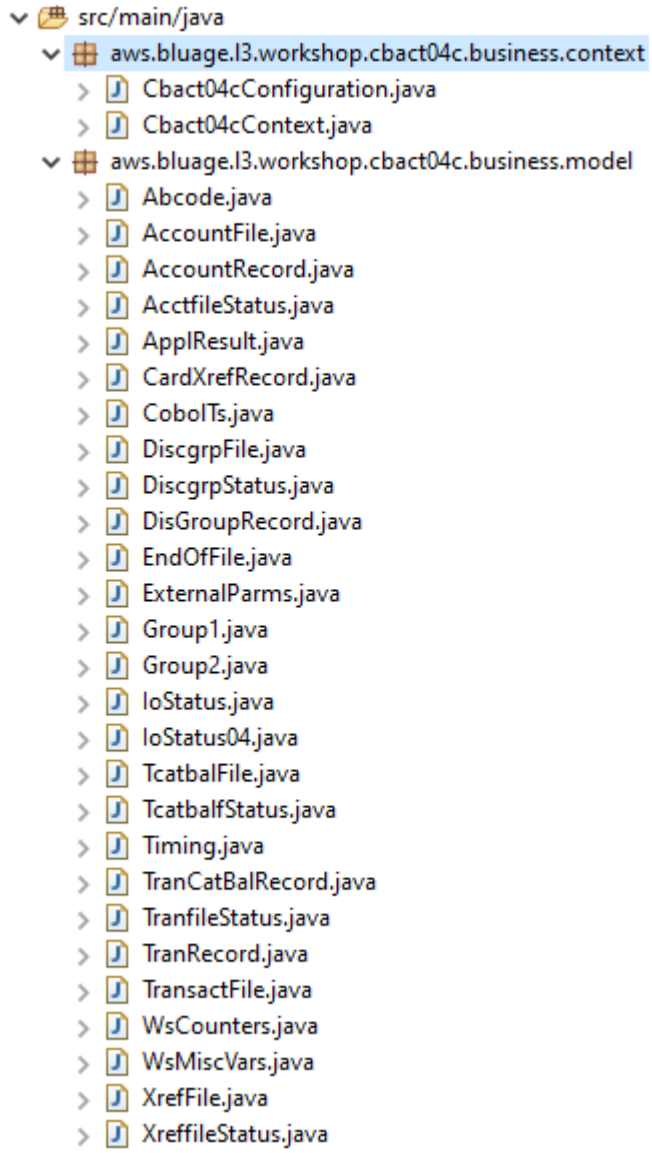
L'organisation de base repose sur la structure suivante :

- Projet Entities : contient des éléments de modèle commercial et de contexte. Le nom du projet se termine généralement par « -entities ». Généralement, pour un ancien programme COBOL donné, cela correspond à la modernisation de la section E/S (ensembles de données) et de la division des données. Vous pouvez avoir plusieurs projets d'entités.
- Projet de service : contient des éléments de modernisation de la logique métier héritée. Il s'agit généralement de la division des procédures d'un programme COBOL. Vous pouvez avoir plusieurs projets de service.
- Projet utilitaire : contient des outils et utilitaires communs partagés, utilisés par d'autres projets.
- Projet Web : contient la modernisation des éléments liés à l'interface utilisateur, le cas échéant. Non utilisé pour les projets de modernisation par lots uniquement. Ces éléments d'interface utilisateur peuvent provenir de cartes CICS BMS, de composants IMS MFS et d'autres sources d'interface utilisateur du mainframe. Vous pouvez avoir plusieurs projets Web.

Entités, contenu du projet

> **Note**
>
> Les descriptions suivantes s'appliquent uniquement aux sorties de modernisation COBOL et PL/I. Les sorties de modernisation des RPG sont basées sur une disposition différente.

Avant toute refactorisation, l'organisation des packages dans le projet d'entités est liée aux programmes modernisés. Vous pouvez y parvenir de différentes manières. La méthode préférée consiste à utiliser la boîte à outils Refactoring, qui fonctionne avant que vous ne déclenchiez le mécanisme de génération de code. Il s'agit d'une opération avancée, qui est expliquée dans les BluAge formations. Pour plus d'informations, consultez la section Atelier de refactorisation. Cette approche vous permet de conserver la possibilité de régénérer le code Java ultérieurement, afin de bénéficier de nouvelles améliorations dans le futur, par exemple). L'autre méthode consiste à effectuer des refactorisations Java régulières, directement sur le code source généré, en utilisant n'importe quelle approche de refactorisation Java que vous souhaiteriez appliquer, à vos risques et périls.



Cours liés au programme

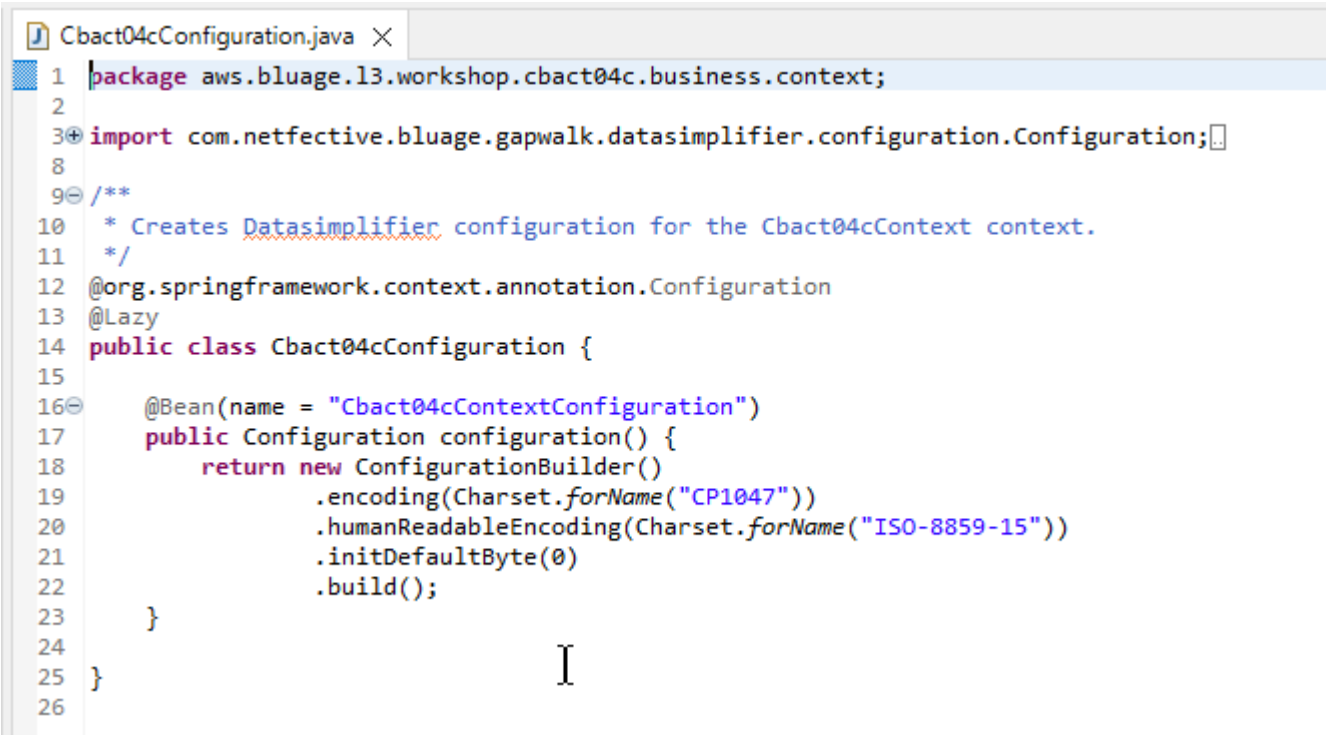
Chaque programme modernisé est associé à deux packages, un package business.context et un package business.model.

- *`base package`*`.`*`program`*`.business.context`

  Le sous-package business.context contient deux classes, une classe de configuration et une classe de contexte.

  - Une classe de configuration pour le programme, qui contient les détails de configuration spécifiques au programme donné, tels que le jeu de caractères à utiliser pour représenter les éléments de données basés sur des caractères, la

valeur d'octet par défaut pour le remplissage des éléments de structure de données, etc. Le nom de la classe se termine par « Configuration ». Il est marqué par l'`@org.springframework.context.annotation.Configuration`annotation et contient une méthode unique qui doit renvoyer un `Configuration` objet correctement configuré.



- Une classe de contexte, qui sert de pont entre les classes de service du programme (voir ci-dessous) et les structures de données (`Record`) et les ensembles de données (`File`) du sous-package du modèle (voir ci-dessous). Le nom de classe se termine par « Context » et est une sous-classe de la `RuntimeContext` classe.

Cbact04cContext.java

```java
@Component("aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext")
@Import({
    aws.bluage.l3.workshop.cbact04c.business.model.TcatbalFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.XrefFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.AccountFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.TransactFile.class
})
@Lazy
@Scope("prototype")
public class Cbact04cContext extends JicsRuntimeContext {

    @Autowired
    private TcatbalFile tcatbalFile;

    @Autowired
    private XrefFile xrefFile;

    @Autowired
    private DiscgrpFile discgrpFile;

    @Autowired
    private AccountFile accountFile;

    @Autowired
    private TransactFile transactFile;

    private IndexedFile tcatbalFileFile;

    private IndexedFile xrefFileFile;

    private IndexedFile discgrpFileFile;

    private IndexedFile accountFileFile;

    private SequentialFile transactFileFile;

    private TranCatBalRecord tranCatBalRecord;
    private TcatbalfStatus tcatbalfStatus;
    private CardXrefRecord cardXrefRecord;
```

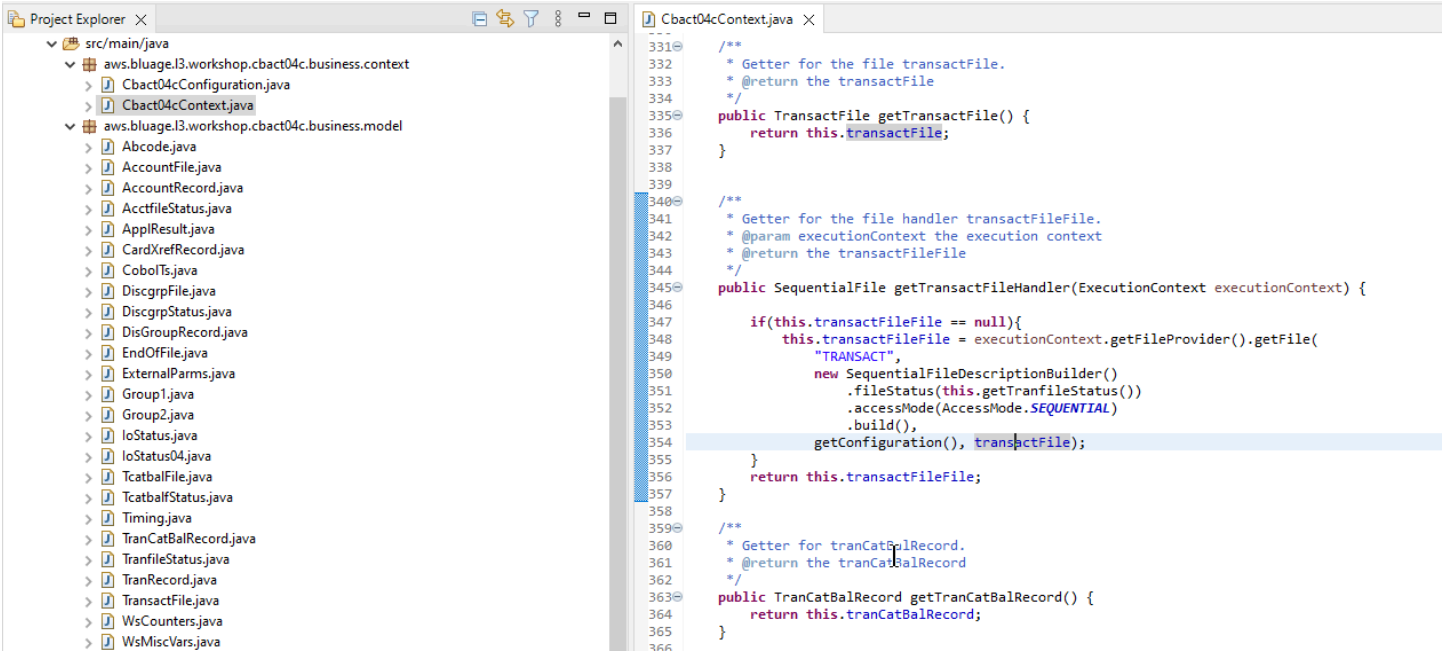
- *base package*.*program*.business.model

  Le sous-package du modèle contient toutes les structures de données que le programme donné peut utiliser. Par exemple, toute structure de données COBOL de niveau 01 correspond à une classe du sous-package du modèle (les structures de données de niveau inférieur sont les propriétés de leur propre structure de niveau 01). Pour plus d'informations sur la façon dont nous modernisons les structures de données 01, consultezSimplificateur de données.

DiscgrpFile.java

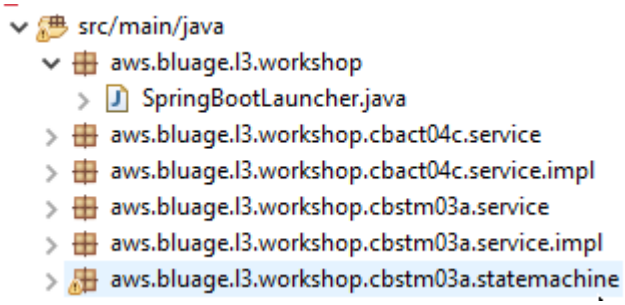
```java
package aws.bluage.l3.workshop.cbact04c.business.model;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group;
import com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

/**
 * Data simplifier file DiscgrpFile.
 *
 * <p>About 'fdDiscgrpRec' field, <br>uml entity: aws.bluage.l3.workshop.cbact04c.business.model.FdDiscgrpRec
 * <br></p>
 *
 */
@Component("aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile")
@Lazy
@Scope("prototype")
public class DiscgrpFile extends RecordEntity {

    private final Group root = new Group(getData());
    private final Group fdDiscgrpRec = new Group(root);
    private final Group fdDiscgrpKey = new Group(fdDiscgrpRec);
    private final Elementary fdDisAcctGroupId = new Elementary(fdDiscgrpKey,new AlphanumericType(10));
    private final Elementary fdDisTranTypeCd = new Elementary(fdDiscgrpKey,new AlphanumericType(2));
    private final Elementary fdDisTranCatCd = new Elementary(fdDiscgrpKey,new ZonedType(4, 0, false));
    private final Elementary fdDiscgrpData = new Elementary(fdDiscgrpRec,new AlphanumericType(34));
```

Toutes les classes étendent la `RecordEntity` classe, qui représente l'accès à une représentation d'enregistrement commercial. Certains enregistrements ont un objectif spécial, car ils sont liés à un`File`. La liaison entre a `Record` et a `File` est établie dans les FileHandler méthodes * correspondantes trouvées dans la classe de contexte lors de la création de l'objet fichier. Par exemple, la liste suivante montre comment le TransactfileFile `File` est lié au TransactFile `Record` (à partir du sous-package du modèle).



Contenu du projet de service

Chaque projet de service est accompagné d'une application Springboot dédiée, qui est utilisée comme colonne vertébrale de l'architecture. Cela se matérialise par le biais de la classe nomméeSpringBootLauncher, située dans le package de base des sources Java du service :



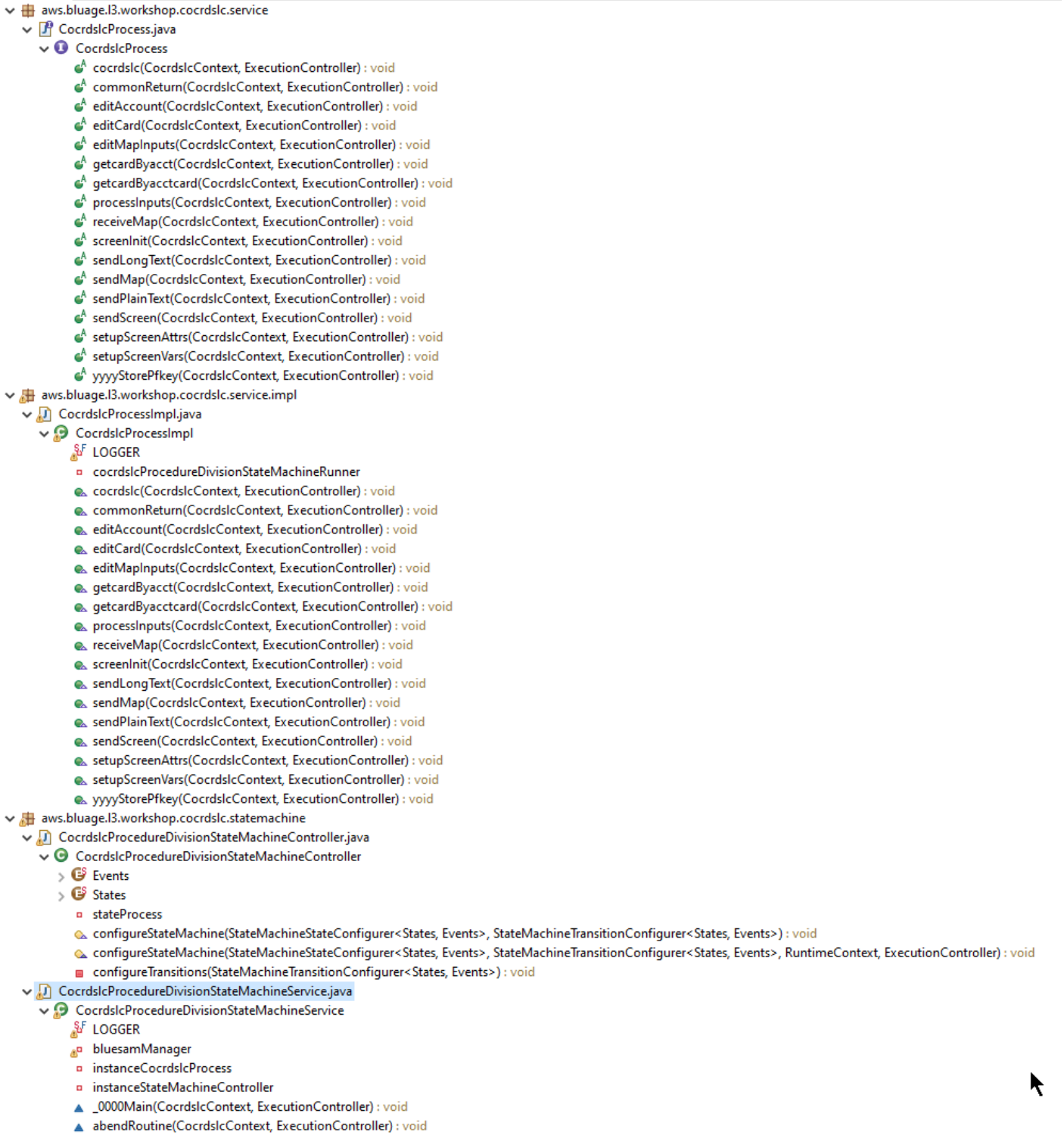
Cette classe est notamment chargée de :

- faire le lien entre les classes du programme et les ressources gérées (sources de données/ gestionnaires de transactions/mappages d'ensembles de données/etc...).
- fournissant un ou `ConfigurableApplicationContext` deux programmes.
- découvrir toutes les classes marquées comme spring components (`@Component`).
- s'assurer que les programmes sont correctement enregistrés dans le `ProgramRegistry` -- voir la méthode d'initialisation en charge de cet enregistrement.

```
/**
 * Initialization method called when the spring application is ready.
 * Register all programs and services to the gapwalk shared context.
 * @param event the application ready event
 */
@EventListener
public void initialize(ApplicationReadyEvent event) {
    Map<String, ProgramContainer> programContainers = event.getApplicationContext().getBeansOfType(ProgramContainer.class);
    programContainers.values().forEach(ProgramRegistry::registerProgram);
    Map<String, ServiceContainer> serviceContainers = event.getApplicationContext().getBeansOfType(ServiceContainer.class);
    serviceContainers.values().forEach(ServiceRegistry::registerService);
}
```

Artefacts liés au programme

Sans refactorisation préalable, les résultats de la modernisation de la logique métier sont organisés en deux ou trois packages par programme existant :



Le cas le plus exhaustif comportera trois packages :

- *base package*`.`*program*`.service`: contient une interface nommée Program Process, qui dispose de méthodes métier pour gérer la logique métier, en préservant le flux de contrôle d'exécution existant.
- *base package*`.`*program*`.service.impl`: contient une classe nommée ProgramProcessImpl, qui est l'implémentation de l'interface Process décrite précédemment. C'est ici que les anciennes instructions sont « traduites » en instructions Java, en s'appuyant sur le framework AWS Blu Age :

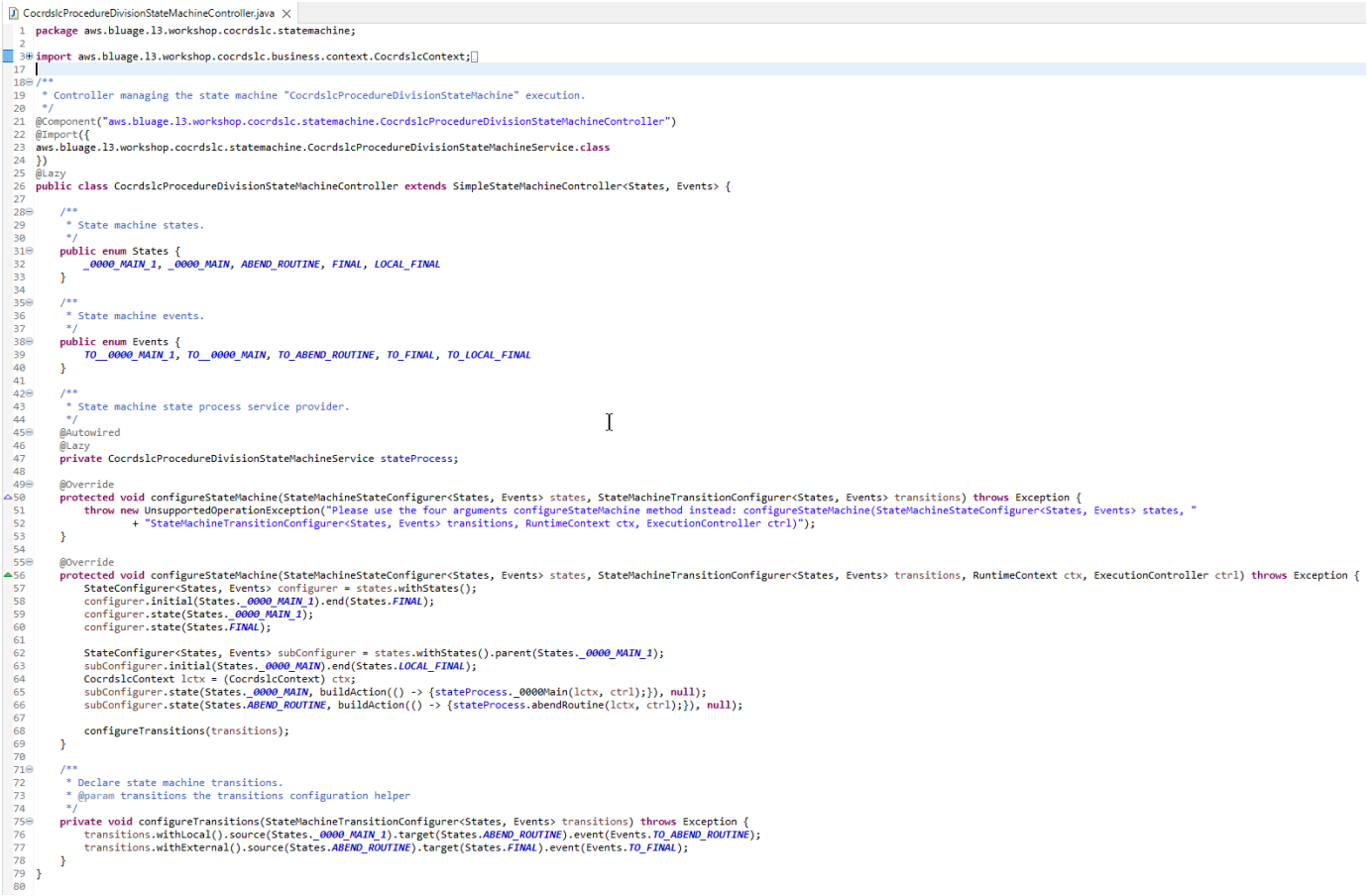
CocrdslcProcessImpl.java

```java
    /**
     * Process operation sendScreen.
     *
     * @param ctx
     * @param ctrl
     */
    @Override
    public void sendScreen(final CocrdslcContext ctx, final ExecutionController ctrl) {
        ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
        ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
        ctx.getCarddemoCommarea().setCdemoPgmReenter(true);
        SendMapBuilder.newInstance(ctx.getDfheiblk(), ctx)
        .withMap(ctx.getCcWorkAreas().getCcardNextMap())
        .withMapset(ctx.getCcWorkAreas().getCcardNextMapset())
        .withData(ctx.getGroup1().getCcrdslaoReference())
        .withCursor()
        .withErase()
        .withFreeKB()
        .execute();
        ctx.getWsMiscStorage().setWsRespCd(ctx.getDfheiblk().getEibresp());
    }

    /**
     * Process operation processInputs.
     *
     * @param ctx
     * @param ctrl
     */
    @Override
    public void processInputs(final CocrdslcContext ctx, final ExecutionController ctrl) {
        receiveMap(ctx, ctrl);
        editMapInputs(ctx, ctrl);
        ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
        ctx.getCcWorkAreas().setCcardNextProg(ctx.getWsLiterals().getLitThispgm());
        ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
        ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
    }

```

- *base package*`.`*program*`.statemachine`: ce package n'est peut-être pas toujours présent. Cela est nécessaire lorsque la modernisation de l'ancien flux de contrôle doit utiliser une approche basée sur une machine à états (notamment en utilisant le StateMachine framework Spring) pour couvrir correctement le flux d'exécution existant.

  Dans ce cas, le sous-package statemachine contient deux classes :

- *Program*`ProcedureDivisionStateMachineController`: une classe qui étend une classe implémentant les interfaces `StateMachineController` (définition des opérations nécessaires pour contrôler l'exécution d'une machine à états) et `StateMachineRunner` (définition des opérations requises pour exécuter une machine à états), utilisées pour piloter la mécanique des machines à états Spring ; par exemple, `SimpleStateMachineController` comme dans le cas d'exemple.



```
CocrdslcProcedureDivisionStateMachineController.java
package aws.bluage.l3.workshop.cocrdslc.statemachine;

import aws.bluage.l3.workshop.cocrdslc.business.context.CocrdslcContext;

/**
 * Controller managing the state machine "CocrdslcProcedureDivisionStateMachine" execution.
 */
@Component("aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineController")
@Import({
aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineService.class
})
@Lazy
public class CocrdslcProcedureDivisionStateMachineController extends SimpleStateMachineController<States, Events> {

    /**
     * State machine states.
     */
    public enum States {
        _0000_MAIN_1, _0000_MAIN, ABEND_ROUTINE, FINAL, LOCAL_FINAL
    }

    /**
     * State machine events.
     */
    public enum Events {
        TO__0000_MAIN_1, TO__0000_MAIN, TO_ABEND_ROUTINE, TO_FINAL, TO_LOCAL_FINAL
    }

    /**
     * State machine state process service provider.
     */
    @Autowired
    @Lazy
    private CocrdslcProcedureDivisionStateMachineService stateProcess;

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        throw new UnsupportedOperationException("Please use the four arguments configureStateMachine method instead: configureStateMachine(StateMachineStateConfigurer<States, Events> states, "
                + "StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl)");
    }

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl) throws Exception {
        StateConfigurer<States, Events> configurer = states.withStates();
        configurer.initial(States._0000_MAIN_1).end(States.FINAL);
        configurer.state(States._0000_MAIN_1);
        configurer.state(States.FINAL);

        StateConfigurer<States, Events> subConfigurer = states.withStates().parent(States._0000_MAIN_1);
        subConfigurer.initial(States._0000_MAIN).end(States.LOCAL_FINAL);
        CocrdslcContext lctx = (CocrdslcContext) ctx;
        subConfigurer.state(States._0000_MAIN, buildAction(() -> {stateProcess._0000Main(lctx, ctrl);}), null);
        subConfigurer.state(States.ABEND_ROUTINE, buildAction(() -> {stateProcess.abendRoutine(lctx, ctrl);}), null);

        configureTransitions(transitions);
    }

    /**
     * Declare state machine transitions.
     * @param transitions the transitions configuration helper
     */
    private void configureTransitions(StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        transitions.withLocal().source(States._0000_MAIN_1).target(States.ABEND_ROUTINE).event(Events.TO_ABEND_ROUTINE);
        transitions.withExternal().source(States.ABEND_ROUTINE).target(States.FINAL).event(Events.TO_FINAL);
    }
}
```

Le contrôleur de machine à états définit les différents états possibles et les transitions entre eux, qui reproduisent le flux de contrôle d'exécution existant pour le programme donné.

Lors de la création de la machine à états, le contrôleur fait référence aux méthodes définies dans la classe de service associée située dans le package de machine à états et décrite ci-dessous :

```
subConfigurer.state(States._0000_MAIN, buildAction(() ->
 {stateProcess._0000Main(lctx, ctrl);}), null);
subConfigurer.state(States.ABEND_ROUTINE, buildAction(() ->
 {stateProcess.abendRoutine(lctx, ctrl);}), null);
```

- *Program*ProcedureDivisionStateMachineService: cette classe de service représente une certaine logique métier qui doit être liée à la machine d'état créée par le contrôleur de machine d'état, comme décrit précédemment.

  Le code des méthodes de cette classe utilise les événements définis dans le contrôleur State Machine :

CocrdslcProcedureDivisionStateMachineService.java

```
59    /**
60     * State process operation _0000Main.
61     *
62     * @param ctx
63     * @param ctrl
64     */
65    void _0000Main(CocrdslcContext ctx, ExecutionController ctrl) {
66        ctx.getDfheiblk().bind(ArgUtils.get(ctx, 0));
67        ctx.getDfhcommarea().bind(ArgUtils.get(ctx, 1));
68
69        /*
70        *****************************************************************
71        Program:     COCRDSLC.CBL                                      *
72        Layer:       Business logic                                    *
73        Function:    Accept and process credit card detail request     *
74        *****************************************************************
75        Copyright Amazon.com, Inc. or its affiliates.
76        All Rights Reserved.
77        Licensed under the Apache License, Version 2.0 (the \"License\").
78        You may not use this file except in compliance with the License.
79        You may obtain a copy of the License at
80        http://www.apache.org/licenses/LICENSE-2.0
81        Unless required by applicable law or agreed to in writing,
82        software distributed under the License is distributed on an
83        \"AS IS\" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
84        either express or implied. See the License for the specific
85        language governing permissions and limitations under the License
86        *****************************************************************
87        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:16:00 CDT */
88        instanceStateMachineController.registerSignalHandler(Events.TO_ABEND_ROUTINE, "!ABEND");
89        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).execute().handleException();
90        ctx.getCcWorkAreas().getCcWorkAreaReference().getField().initialize();
91        ctx.getWsMiscStorage().getField().initialize();
92        DataUtils.initialize(ctx.getWsCommarea().getWsCommareaReference());
93
```

CocrdslcProcedureDivisionStateMachineService.java

```
     *
     * @param ctx
     * @param ctrl
     */
    void abendRoutine(CocrdslcContext ctx, ExecutionController ctrl) {
        if (DataUtils.isLowValue(ctx.getAbendData().getAbendMsgReference())) {
            ctx.getAbendData().setAbendMsg("UNEXPECTED ABEND OCCURRED.");
        }
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        SendTextBuilder.newInstance(ctx.getDfheiblk(), ctx)
        .withData(ctx.getAbendData())
        .withLength(134)
        .execute();
        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).cancel().execute().handleException();
        AbendBuilder.newInstance(ctx.getDfheiblk(), ctx).withAbendCode("9999").execute().handleException();

        /*
        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:12:33 CDT */
        instanceStateMachineController.sendEvent(Events.TO_FINAL);

    }
}

```

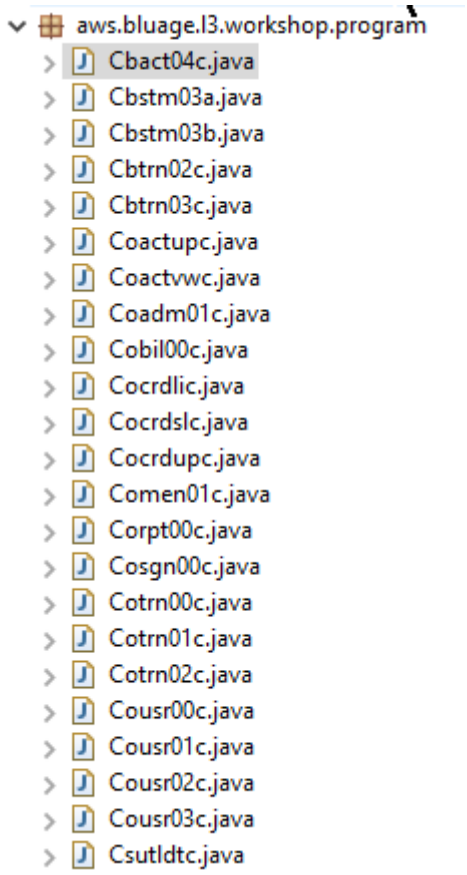
Le service statemachine appelle également l'implémentation du service de processus décrite précédemment :

CocrdslcProcedureDivisionStateMachineService.java

```
        /*
        ************************************************************
        COMING FROM CREDIT CARD LIST SCREEN
        SELECTION CRITERIA ALREADY VALIDATED
        ************************************************************ */
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter() && DataUtils.compare(ctx.getCarddemoCommarea().getCdemoFromProgramReference(), ctx.getWsLiterals().getLitCclistpgmReference()) == 0) {
        ctx.getWsMiscStorage().setInputOk(true);
        ctx.getCcWorkAreas().setCcAcctIdN(ctx.getCarddemoCommarea().getCdemoAcctId());
        ctx.getCcWorkAreas().setCcCardNumN(ctx.getCarddemoCommarea().getCdemoCardNum());
        instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter()) {

        /*
        ************************************************************
        COMING FROM SOME OTHER CONTEXT
        SELECTION CRITERIA TO BE GATHERED
        ************************************************************ */
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmReenter()) {
        instanceCocrdslcProcess.processInputs(ctx, ctrl);
        if (ctx.getWsMiscStorage().isInputError()) {
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        } else {
            instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        }
    } else {
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        ctx.getAbendData().setAbendCode("0001");
        DataUtils.setToBlank(ctx.getAbendData().getAbendReasonReference());
        ctx.getWsMiscStorage().setWsReturnMsg("UNEXPECTED DATA SCENARIO");
        instanceCocrdslcProcess.sendPlainText(ctx, ctrl);
    }

    /*
    If we had an error setup error message that slipped through
    Display and return */
    if (ctx.getWsMiscStorage().isInputError()) {
        ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    }
    instanceCocrdslcProcess.commonReturn(ctx, ctrl);
```

De plus, un package nommé *`base package`*`.program` joue un rôle important, car il regroupe une classe par programme, qui servira de point d'entrée au programme (plus de détails à ce sujet

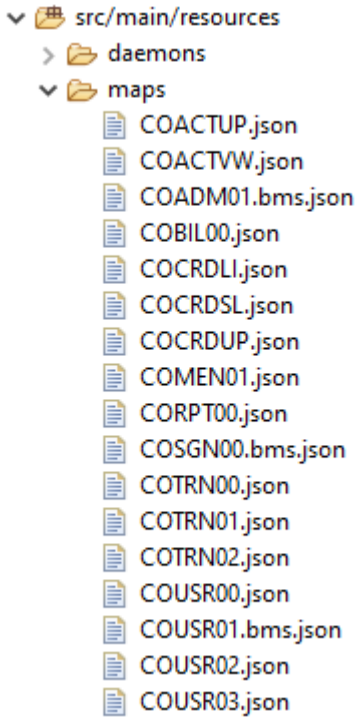
ultérieurement). Chaque classe implémente l'`Program`interface, marqueur d'un point d'entrée du programme.



Autres artéfacts

- Compagnons BMS MaPS

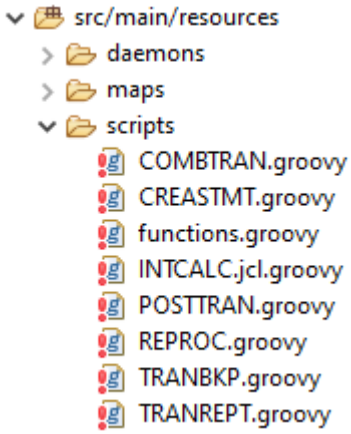
  Outre les artefacts liés au programme, le projet de service peut contenir d'autres artefacts à des fins diverses. Dans le cas de la modernisation d'une application en ligne CICS, le processus de modernisation produit un fichier json et place dans le dossier map du dossier /src/main/resources :



Le moteur d'exécution Blu Age utilise ces fichiers json pour lier les enregistrements utilisés par l'instruction SEND MAP aux champs d'écran.

- Scripts géniaux

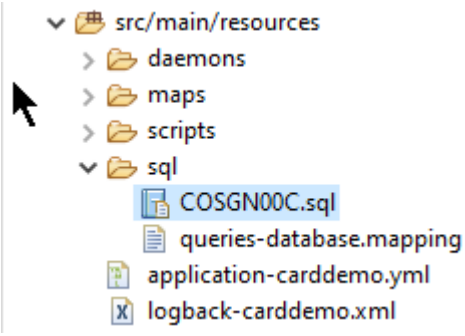
  Si l'ancienne application comportait des scripts JCL, ceux-ci ont été modernisés en scripts groovy, stockés dans le dossier /src/main/resources/scripts (nous reviendrons sur cet emplacement spécifique ultérieurement) :



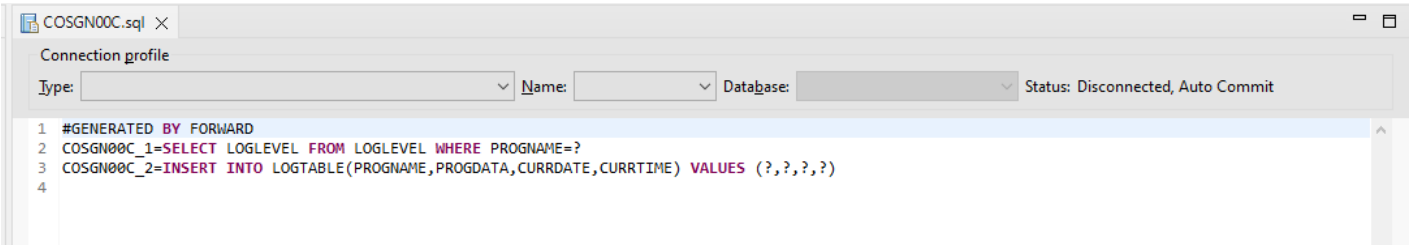
  Ces scripts sont utilisés pour lancer des tâches par lots (charges de travail de traitement de données dédiées, non interactives et gourmandes en ressources processeur).

- fichiers SQL

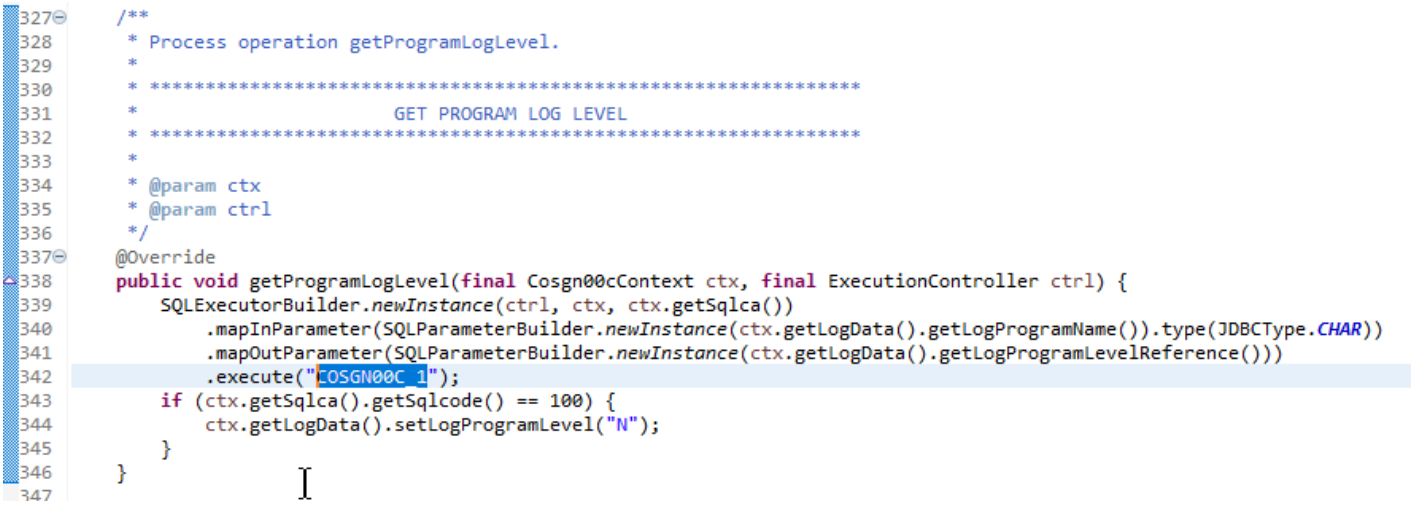
Si l'ancienne application utilisait des requêtes SQL, les requêtes SQL modernisées correspondantes ont été rassemblées dans des fichiers de propriétés dédiés, avec le modèle de dénomination program .sql, où program est le nom du programme utilisant ces requêtes.



Le contenu de ces fichiers SQL est une collection d'entrées (clé=requête), où chaque requête est associée à une clé unique, que le programme modernisé utilise pour exécuter la requête donnée :

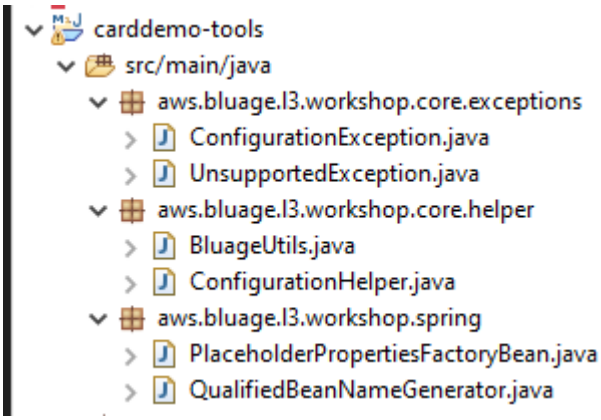


Par exemple, le programme COSGN00C exécute la requête avec la clé « COSGN00C_1 » (la première entrée du fichier SQL) :



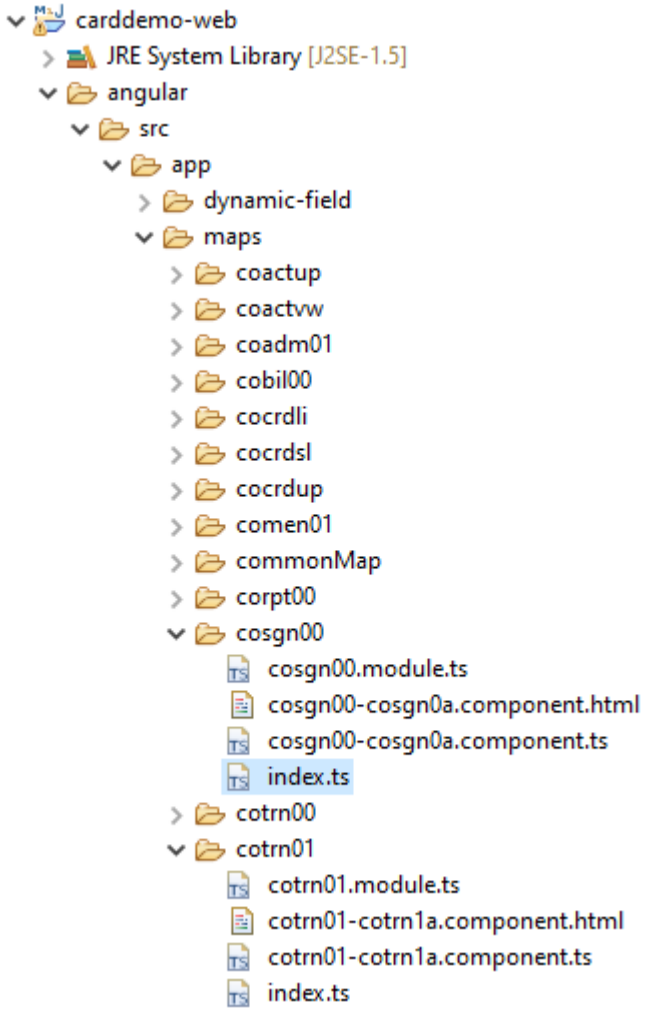
Contenu du projet Utilities

Le projet d'utilitaires, dont le nom se termine par « -tools », contient un ensemble d'utilitaires techniques qui peuvent être utilisés par tous les autres projets.



Contenu du (des) projet (s) Web

Le projet Web n'est présent que lors de la modernisation des anciens éléments de l'interface utilisateur. Les éléments d'interface utilisateur modernes utilisés pour créer le front-end d'application modernisé sont basés sur Angular. L'exemple d'application utilisé pour présenter les artefacts de modernisation est une application COBOL/CICS exécutée sur un ordinateur central. Le système CICS utilise des cartes pour représenter les écrans de l'interface utilisateur. Les éléments modernes correspondants seront, pour chaque carte, un fichier html accompagné de fichiers Typescript :



Le projet Web ne prend en charge que l'aspect frontal de l'application. Le projet de service, qui repose sur les projets d'utilitaires et d'entités, fournit les services principaux. Le lien entre le frontend et le backend est établi via l'application Web nommée Gapwalk-Application, qui fait partie de la distribution d'exécution standard de Blu Age. AWS

Programmes en cours d'exécution et d'appel

Sur les anciens systèmes, les programmes sont compilés sous forme d'exécutables autonomes qui peuvent s'appeler eux-mêmes via un mécanisme CALL, tel que l'instruction COBOL CALL, en transmettant des arguments lorsque cela est nécessaire. Les applications modernisées offrent les mêmes fonctionnalités mais utilisent une approche différente, car la nature des artefacts concernés est différente de celle des anciens artefacts.

Du côté modernisé, les points d'entrée du programme sont des classes spécifiques qui implémentent l'`Program`interface, sont des composants Spring (@Component) et sont situés dans des projets de service, dans un package nommé*`base package`*`.program`.

Inscription aux programmes

Chaque fois que le serveur Tomcat hébergeant les applications modernisées est démarré, le service Springboot est également lancé, ce qui déclenche l'enregistrement des programmes. Un registre dédié nommé `ProgramRegistry` est rempli d'entrées de programme, chaque programme étant enregistré à l'aide de ses identifiants, une entrée par identifiant de programme connu, ce qui signifie que si un programme est connu sous plusieurs identifiants différents, le registre contient autant d'entrées qu'il y a d'identifiants.

L'enregistrement d'un programme donné repose sur la collection d'identifiants renvoyés par la méthode getProgramIdentifiers () :

Cbact04c.java

```java
package aws.bluage.l3.workshop.program;

import aws.bluage.l3.workshop.SpringBootLauncher;

/**
 * Reference the spring application of program CBACT04C.
 *  Provides an access to the contained program for the run unit.
 */
@Component
@Import({
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cConfiguration.class,
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext.class,
    aws.bluage.l3.workshop.cbact04c.service.impl.Cbact04cProcessImpl.class
})
public class Cbact04c implements Program {
    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("CBACT04C").collect(Collectors.toSet()));

    /**
     * Main program identifier for the contained program.
     */
    private static final String programIdentifier = "CBACT04C";
    @Autowired
    PlatformTransactionManager transactionManager;

    @Autowired
    Map<String, DataSource> datasources;
    @Autowired
    BeanFactory beanFactory;
    /**
     * {@inheritDoc}
     */
    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public void updateExecutionContext(ExecutionContext executionContext) {
        executionContext.setDatasources(datasources);
        executionContext.setDatabaseSupport(ExecutionContext.DatabaseSupport.POSTGRE);
        executionContext.setSqlcaVersion(ExecutionContext.SqlcaVersion.getEnum("ansi-comp5"));
        executionContext.setTransactionManager(transactionManager);
        executionContext.setUseSQLDateNewParadigm(true);
        executionContext.setUseSQLTrimStringType(false);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Dans cet exemple, le programme est enregistré une seule fois, sous le nom « CBACT04C » (regardez le contenu de la collection ProgramIdentifiers). Les journaux Tomcat indiquent chaque enregistrement de programme. L'enregistrement du programme dépend uniquement des identificateurs de programme déclarés et non du nom de classe de programme lui-même (bien que les identifiants de programme et les noms de classe de programme soient généralement alignés).

Le même mécanisme d'enregistrement s'applique aux programmes utilitaires fournis par les différentes applications Web utilitaires AWS Blu Age, qui font partie de la distribution d'exécution

AWS Blu Age. Par exemple, l'application Web Gapwalk-Utility-Pgm fournit les équivalents fonctionnels des utilitaires système z/OS (IDCAMS, ICEGENER, SORT, etc.) et peut être appelée par des programmes ou des scripts modernisés. Tous les programmes utilitaires disponibles enregistrés au démarrage de Tomcat sont enregistrés dans les journaux Tomcat.

Enregistrement des scripts et des démons

Un processus d'enregistrement similaire, au démarrage de Tomcat, se produit pour les scripts groovy situés dans la hiérarchie des dossiers /src/main/resources/scripts. La hiérarchie des dossiers de scripts est parcourue et tous les scripts groovy découverts (à l'exception du script réservé special functions.groovy) sont enregistrés dans le`ScriptRegistry`, en utilisant leur nom court (la partie du nom du fichier de script située avant le premier point) comme clé de récupération.

> **Note**
>
> - Si plusieurs scripts ont des noms de fichiers qui produiront la même clé d'enregistrement, seule la dernière est enregistrée, remplaçant ainsi toute inscription précédemment rencontrée pour cette clé donnée.
> - Compte tenu de ce qui précède, faites attention lorsque vous utilisez des sous-dossiers, car le mécanisme d'enregistrement aplatit la hiérarchie et peut entraîner des remplacements inattendus. La hiérarchie ne compte pas dans le processus d'enregistrement : en général, /Scripts/a/MyScript.Groovy et /Scripts/B/MyScript.Groovy entraînent le remplacement de /Scripts/B/MyScript.Groovy par /Scripts/a/MyScript.Groovy.

Les scripts groovy du dossier /src/main/resources/daemons sont gérés un peu différemment. Ils sont toujours enregistrés en tant que scripts classiques, mais en plus, ils ne sont lancés qu'une seule fois, directement au moment du démarrage de Tomcat, de manière asynchrone.

Une fois les scripts enregistrés dans le`ScriptRegistry`, un appel REST peut les lancer, en utilisant les points de terminaison dédiés exposés par l'application Gapwalk. Pour plus d'informations, consultez la documentation correspondante.

Programmes, programmes d'appel

Chaque programme peut appeler un autre programme en tant que sous-programme et lui transmettre des paramètres. Pour ce faire, les programmes utilisent une implémentation de l'`ExecutionController`interface (la plupart du temps, il s'agira d'une

`ExecutionControllerImpl` instance), ainsi qu'un mécanisme d'API fluide appelé le `CallBuilder` pour générer les arguments d'appel du programme.

Toutes les méthodes des programmes prennent à la fois a `RuntimeContext` et an `ExecutionController` comme arguments de méthode, de sorte que an `ExecutionController` est toujours disponible pour appeler d'autres programmes.

Voir, par exemple, le schéma suivant, qui montre comment le programme CBST03A appelle le programme CBST03B en tant que sous-programme, en lui transmettant des paramètres :

Cbstm03aProcessImpl.java

```
67      /**
68       * Process operation xreffileGetNext.
69       *
70       * ----------------------------------------------------------------*
71       *
72       * @param ctx
73       * @param ctrl
74       */
75      @Override
76      public void xreffileGetNext(final Cbstm03aContext ctx, final ExecutionController ctrl) {
77          ctx.getWsM03bArea().setWsM03bDd("XREFFILE");
78          ctx.getWsM03bArea().setM03bRead(true);
79          DataUtils.setToZeroes(ctx.getWsM03bArea().getWsM03bRcReference());
80          DataUtils.setToBlank(ctx.getWsM03bArea().getWsM03bFldtReference());
81          ctrl.callSubProgram("CBSTM03B", CallBuilder.newInstance()
82              .byReference(ctx.getWsM03bArea())
83              .getArguments(), ctx);
84          if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "00") == 0) {
85
86              /*
87              Do nothing */
88          } else if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "10") == 0) {
89              ctx.getMiscVariables().setEndOfFile("Y");
90          } else {
91              if (LOGGER.isInfoEnabled()) LOGGER.info("ERROR READING XREFFILE");
92              if (LOGGER.isInfoEnabled()) LOGGER.info("{}{}" , "RETURN CODE: " , ctx.getWsM03bArea().getWsM03bRc());
93              abendProgram(ctx, ctrl);
94          }
95          ctx.getCardXrefRecord().setBytes(ctx.getWsM03bArea().getWsM03bFldtReference().getBytes());
96      }
97
```

- Le premier argument de `ExecutionController.callSubProgram` est un identifiant du programme à appeler (c'est-à-dire l'un des identifiants utilisés pour l'enregistrement du programme, voir paragraphes ci-dessus).
- Le deuxième argument, qui est le résultat de la construction sur le`CallBuilder`, est un tableau de`Record`, correspondant aux données transmises de l'appelant à l'appelé.
- Le troisième et dernier argument est l'`RuntimeContext`instance de l'appelant.

Les trois arguments sont obligatoires et ne peuvent pas être nuls, mais le second argument peut être un tableau vide.

L'appelé ne pourra traiter les paramètres transmis que s'il a été initialement conçu pour le faire. Pour un ancien programme COBOL, cela signifie avoir une section LINKAGE et une clause USING pour la division des procédures afin d'utiliser les éléments LINKAGE.

Par exemple, consultez le fichier source COBOL CBSTM03B.CBL correspondant :

github.com/aws-samples/aws-mainframe-modernization-carddemo/blob/main/app/cbl/CBSTM03B.CBL

```
 98
 99          LINKAGE SECTION.
100          01  LK-M03B-AREA.
101              05  LK-M03B-DD          PIC X(08).
102              05  LK-M03B-OPER        PIC X(01).
103                88  M03B-OPEN       VALUE 'O'.
104                88  M03B-CLOSE      VALUE 'C'.
105                88  M03B-READ       VALUE 'R'.
106                88  M03B-READ-K     VALUE 'K'.
107                88  M03B-WRITE      VALUE 'W'.
108                88  M03B-REWRITE    VALUE 'Z'.
109              05  LK-M03B-RC          PIC X(02).
110              05  LK-M03B-KEY         PIC X(25).
111              05  LK-M03B-KEY-LN      PIC S9(4).
112              05  LK-M03B-FLDT        PIC X(1000).
113
114          PROCEDURE DIVISION USING LK-M03B-AREA.
115
```

Le programme CBSTM03B prend donc un seul `Record` comme paramètre (un tableau de taille 1). C'est ce qu'ils sont en train de construire, en utilisant les méthodes de chaînage byReference () et getArguments (). `CallBuilder`

La classe d'API `CallBuilder` Fluent dispose de plusieurs méthodes pour remplir le tableau d'arguments à transmettre à un appelé :

- AsPointer (RecordAdaptable) : ajoute un argument de type pointeur, par référence. Le pointeur représente l'adresse d'une structure de données cible.
- ByReference (RecordAdaptable) : ajoute un argument par référence. L'appelant verra les modifications qu'il effectue.
- ByReference (RecordAdaptable...) : variante varargs de la méthode précédente.
- ByValue (Object) : ajoute un argument, transformé en a`Record`, par valeur. L'appelant ne verra pas les modifications effectuées par l'appelé.

- ByValue (RecordAdaptable) : identique à la méthode précédente, mais l'argument est directement disponible sous forme de`RecordAdaptable`.
- byValueWithBounds (Object, int, int) : ajoutez un argument, transformé en a, en extrayant la partie du tableau d'octets définie par les limites données, par valeur. `Record`

Enfin, la méthode getArguments collectera tous les arguments ajoutés et les renverra sous forme de tableau de`Record`.

> **Note**
>
> Il est de la responsabilité de l'appelant de s'assurer que le tableau d'arguments a la taille requise, que les éléments sont correctement ordonnés et compatibles, en termes de disposition de la mémoire, avec les dispositions attendues pour les éléments de liaison.

Scripts appelant des programmes

L'appel de programmes enregistrés à partir de scripts groovy nécessite l'utilisation d'une instance de classe implémentant l'`MainProgramRunner`interface. Habituellement, l'obtention d'une telle instance est obtenue grâce à l' ApplicationContext utilisation de Spring :

REPROC.groovy

```groovy
// Import
import com.netfective.bluage.gapwalk.rt.provider.ScriptRegistry
import com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
import com.netfective.bluage.gapwalk.io.support.FileConfigurationUtils
import com.netfective.bluage.gapwalk.rt.job.support.DefaultJobContext
import com.netfective.bluage.gapwalk.rt.utils.GroovyUtils
import com.netfective.bluage.gapwalk.rt.io.support.FileConfiguration
import com.netfective.bluage.gapwalk.rt.shared.AbendException
import com.netfective.bluage.gapwalk.rt.call.exception.GroovyExecutionException
// Variables
mpr = applicationContext.getBean("com.netfective.bluage.gapwalk.rt.call.ExecutionController", MainProgramRunner.class)
TreeMap mapTransfo = [:]
```

Une fois qu'une `MainProgramRunner` interface est disponible, utilisez la méthode RunProgram pour appeler un programme et transmettre l'identifiant du programme cible en tant que paramètre :

REPROC.groovy

```
//*******************************************************************
//*                              STEPS                              *
//*******************************************************************
// STEP PRC001 - PGM - IDCAMS*****************************************
def stepPRC001(Object shell, Map params, Map programResults){
    shell.with {
        if (checkValidProgramResults(programResults)) {
            return execStep("PRC001", "IDCAMS", programResults, {
                mpr
                    .withFileConfigurations(new FileConfigurationUtils()
                        .systemOut("SYSPRINT")
                        .output("*")
                        .build()
                        .bluesam("FILEIN")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .bluesam("FILEOUT")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .fileSystem("SYSIN")
                        .path("&CNTLLIB(REPROCT)")
                        .disposition("SHR")
                        .build()
                        .getFileConfigurations(fcmap))
                    .withParameters(params)
                    .runProgram("IDCAMS")
                })
        }
    }
}
```

Dans l'exemple précédent, une étape de travail appelle IDCAMS (programme utilitaire de gestion de fichiers), fournissant un mappage entre les définitions réelles des ensembles de données et leurs identifiants logiques.

Lorsqu'il s'agit d'ensembles de données, les anciens programmes utilisent principalement des noms logiques pour identifier les ensembles de données. Lorsque le programme est appelé à partir d'un script, celui-ci doit associer les noms logiques aux ensembles de données physiques réels. Ces ensembles de données peuvent se trouver sur le système de fichiers, dans un stockage Blusam ou même définis par un flux en ligne, la concaténation de plusieurs ensembles de données ou la génération d'un GDG.

Utilisez withFileConfiguration cette méthode pour créer une carte logique-physique des ensembles de données et la mettre à la disposition du programme appelé.

Écrivez votre propre programme

L'écriture de votre propre programme pour des scripts ou d'autres programmes modernisés à appeler est une tâche courante. Généralement, dans le cadre de projets de modernisation, vous écrivez vos propres programmes lorsqu'un ancien programme exécutable est écrit dans un langage que le processus de modernisation ne prend pas en charge, ou lorsque les sources ont été perdues (oui, cela peut arriver), ou lorsque le programme est un utilitaire dont les sources ne sont pas disponibles.

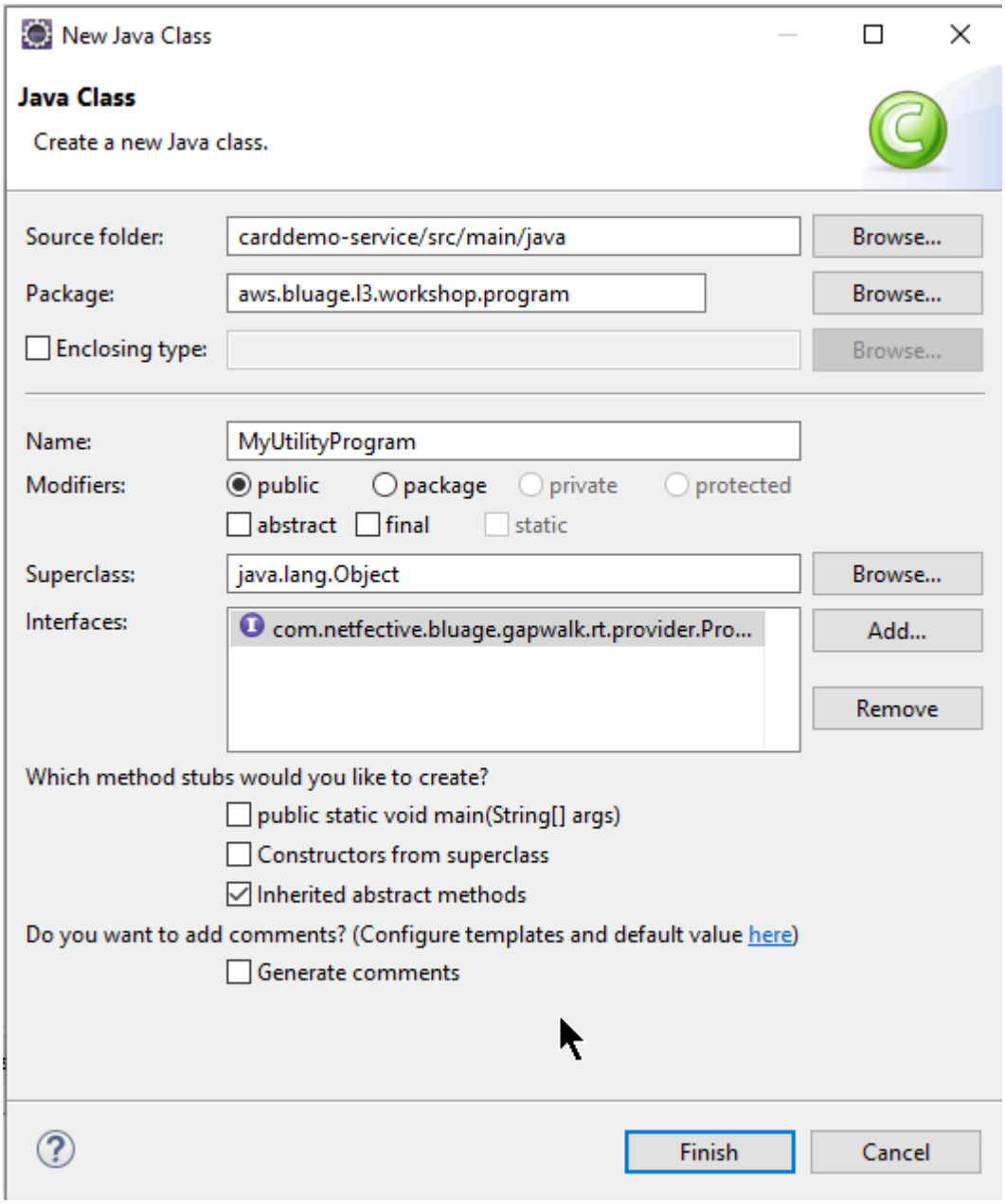
Dans ce cas, vous devrez peut-être écrire vous-même le programme manquant, en Java (en supposant que vous sachiez suffisamment quel devrait être le comportement attendu du programme, la disposition en mémoire des arguments du programme, le cas échéant, etc.) Votre programme Java doit respecter les mécanismes du programme décrits dans ce document, afin que d'autres programmes et scripts puissent l'exécuter.

Pour vous assurer que le programme est utilisable, vous devez effectuer deux étapes obligatoires :

- Écrivez une classe qui implémente correctement l'`Program`interface, afin qu'elle puisse être enregistrée et appelée.
- Assurez-vous que votre programme est correctement enregistré, afin qu'il soit visible depuis d'autres programmes/scripts.

Rédaction de l'implémentation du programme

Utilisez votre IDE pour créer une nouvelle classe Java qui implémente l'`Program`interface :



L'image suivante montre l'IDE Eclipse, qui se charge de créer toutes les méthodes obligatoires à implémenter :

MyUtilityProgram.java

```java
package aws.bluage.l3.workshop.program;

import java.util.Set;

public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Context getContext() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public void run(ExecutionController ctrl) {
        // TODO Auto-generated method stub

    }

}
```

Intégration Spring

Tout d'abord, la classe doit être déclarée en tant que composant Spring. Annotez la classe avec l'`@Componentannotation` suivante :

```java
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.rt.call.ExecutionController;
import com.netfective.bluage.gapwalk.rt.context.Context;
import com.netfective.bluage.gapwalk.rt.provider.Program;

import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
public class MyUtilityProgram implements Program {
```

Ensuite, implémentez correctement les méthodes requises. Dans le contexte de cet exemple, nous avons ajouté le `MyUtilityProgram` package qui contient déjà tous les programmes

modernisés. Cet emplacement permet au programme d'utiliser l'application Springboot existante pour fournir les éléments requis `ConfigurableApplicationContext` pour l'implémentation de la getSpringApplication méthode :

```
public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }
```

Vous pouvez choisir un autre emplacement pour votre propre programme. Par exemple, vous pouvez localiser le programme en question dans un autre projet de service dédié. Assurez-vous que le projet de service donné possède sa propre application Springboot, qui permet de récupérer le ApplicationContext (qui devrait être un`ConfigurableApplicationContext`).

Donner une identité au programme

Pour pouvoir être appelé par d'autres programmes et scripts, le programme doit recevoir au moins un identifiant, qui ne doit entrer en collision avec aucun autre programme enregistré existant dans le système. Le choix de l'identifiant peut être dicté par la nécessité de couvrir le remplacement d'un ancien programme existant ; dans ce cas, vous devrez utiliser l'identifiant attendu, tel que défini dans les occurrences CALL trouvées dans l'ensemble des anciens programmes. La plupart des identifiants de programme comportent 8 caractères dans les anciens systèmes.

La création d'un ensemble non modifiable d'identifiants dans le programme est un moyen d'y parvenir. L'exemple suivant montre comment choisir « MYUTILPG » comme identifiant unique :

```
@Component
public class MyUtilityProgram implements Program {

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));


    public ConfigurableApplicationContext getSpringApplication() {

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Associer le programme à un contexte

Le programme a besoin d'une `RuntimeContext` instance associée. Pour les programmes modernisés, AWS Blu Age génère automatiquement le contexte associé, en utilisant les structures de données qui font partie de l'ancien programme.

Si vous écrivez votre propre programme, vous devez également écrire le contexte associé.

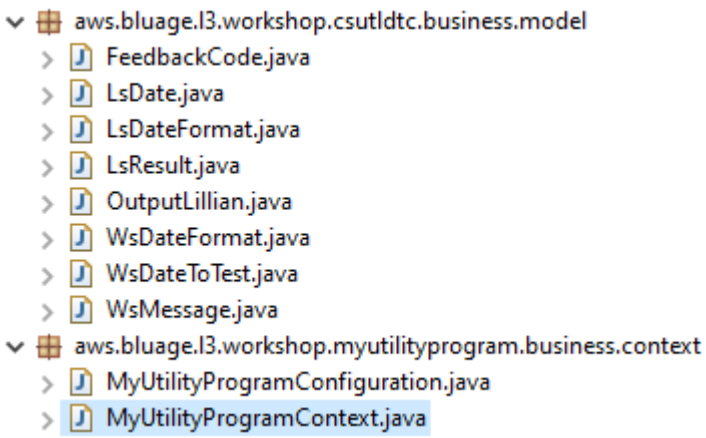
En référence à ce qui suitCours liés au programme, vous pouvez constater qu'un programme nécessite au moins deux cours complémentaires :

- une classe de configuration.
- une classe de contexte qui utilise la configuration.

Si le programme utilitaire utilise une structure de données supplémentaire, elle doit également être écrite et utilisée par le contexte.

Ces classes doivent se trouver dans un package faisant partie d'une hiérarchie de packages qui sera analysée au démarrage de l'application, afin de s'assurer que le composant contextuel et la configuration seront gérés par le framework Spring.

Écrivons une configuration et un contexte minimaux, dans le *`base package`*`.myutilityprogram.business.context` package, fraîchement créé dans le projet entities :



Voici le contenu de la configuration. Il utilise une version de configuration similaire à celle d'autres programmes (modernisés) situés à proximité. Vous devrez probablement le personnaliser en fonction de vos besoins spécifiques.

MyUtilityProgramConfiguration.java

```java
package aws.bluage.l3.workshop.myutilityprogram.business.context;

import java.nio.charset.Charset;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Lazy;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder;

/**
 * Creates Datasimplifier configuration for the MyUtilityProgram context.
 */
@org.springframework.context.annotation.Configuration
@Lazy
public class MyUtilityProgramConfiguration {

    @Bean(name = "MyUtilityProgramContextConfiguration")
    public Configuration configuration() {
        return new ConfigurationBuilder()
                .encoding(Charset.forName("CP1047"))
                .humanReadableEncoding(Charset.forName("ISO-8859-15"))
                .initDefaultByte(0)
                .build();
    }
}
```

Remarques :

- La convention de dénomination générale est ProgramNameConfiguration.
- Il doit utiliser les annotations @org .springframework.context.annotation.Configuration et @Lazy.
- Le nom du haricot suit généralement la ProgramNameContextConfiguration convention, mais cela n'est pas obligatoire. Assurez-vous d'éviter les collisions de noms de beans dans le projet.
- La seule méthode à implémenter doit renvoyer un `Configuration` objet. Utilisez l'API `ConfigurationBuilder` Fluent pour vous aider à en créer une.

Et le contexte associé :

MyUtilityProgramContext.java

```java

import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.rt.context.RuntimeContext;

@Component("aws.bluage.l3.workshop.mytuilityprogram.business.context.MyUtilityProgramContext")
@Lazy
@Scope("prototype")
public class MyUtilityProgramContext extends RuntimeContext{

    protected MyUtilityProgramContext(@Qualifier("MyUtilityProgramContextConfiguration") Configuration configuration) {
        super(configuration);
    }

    @Override
    public void cleanUp() {
        // TODO implement clean-up of associated data structures if any
    }

    @Override
    protected void doReset() {
        // TODO implement reset of associated data structures if any
    }

}

```

Remarques

- La classe de contexte doit étendre une implémentation d'`Context`interface existante (`RuntimeContext`soit`JicsRuntimeContext`, soit améliorée `RuntimeContext` avec des éléments spécifiques à JICS).
- La convention de dénomination générale est ProgramNameContext.
- Vous devez le déclarer en tant que composant du prototype et utiliser l'annotation @Lazy.
- Le constructeur fait référence à la configuration associée en utilisant l'annotation @Qualifier pour cibler la classe de configuration appropriée.
- Si le programme utilitaire utilise des structures de données supplémentaires, celles-ci doivent être :
  - écrit et ajouté au *`base package`*`.business.model` package.
  - référencé dans le contexte. Examinez d'autres classes de contexte existantes pour voir comment référencer les classes de structures de données et adapter les méthodes de contexte (constructeur/nettoyage/réinitialisation) selon les besoins.

Maintenant qu'un contexte dédié est disponible, laissez le nouveau programme l'utiliser :

MyUtilityProgram.java

```java
import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
@Import({
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramConfiguration.class,
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class
})
public class MyUtilityProgram implements Program {

    @Autowired
    BeanFactory beanFactory;

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));

    private static final String programIdentifier = "MYUTILPG";

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public String getProgramMainIdentifier() {
        return programIdentifier;
    }


    @Override
    public Context getContext() {
        return ProgramContextStore.getOrCreate(
        getProgramMainIdentifier(),
        aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class,
        beanFactory);
    }

```

Remarques :

- La méthode GetContext doit être implémentée strictement comme indiqué, en utilisant une délégation à la getOrCreate méthode de la `ProgramContextStore` classe et le Spring câblé automatiquement. `BeanFactory` Un identifiant de programme unique est utilisé pour stocker le contexte du programme dans le `ProgramContextStore` ; cet identifiant est référencé comme étant « l'identifiant principal du programme ».
- La configuration associée et les classes de contexte doivent être référencées à l'aide de l'annotation `@Import` Spring.

Mise en œuvre de la logique métier

Lorsque le squelette du programme est terminé, implémentez la logique métier du nouveau programme utilitaire.

Faites-le dans la `run` méthode du programme. Cette méthode sera exécutée chaque fois que le programme est appelé, que ce soit par un autre programme ou par un script.

Bon codage !

Gestion de l'enregistrement du programme

Enfin, assurez-vous que le nouveau programme est correctement enregistré dans le`ProgramRegistry`. Si vous avez ajouté le nouveau programme au package qui contient déjà d'autres programmes, il n'y a plus rien à faire. Le nouveau programme est sélectionné et enregistré auprès de tous les programmes voisins au démarrage de l'application.

Si vous avez choisi un autre emplacement pour le programme, vous devez vous assurer qu'il est correctement enregistré au démarrage de Tomcat. Pour vous inspirer sur la manière de procéder, examinez la méthode d'initialisation des SpringbootLauncher classes générées dans le ou les projets de service (voirContenu du projet de service).

Consultez les journaux de démarrage de Tomcat. Chaque inscription au programme est enregistrée. Si votre programme est enregistré avec succès, vous trouverez l'entrée de journal correspondante.

Lorsque vous êtes certain que votre programme est correctement enregistré, vous pouvez commencer à itérer sur le codage logique métier.

Mappages de noms entièrement qualifiés

Cette section contient des listes de mappages de noms complets de AWS Blu Age et de tiers à utiliser dans vos applications modernisées.

AWS Mappages de noms entièrement qualifiés Blu Age

Nom court	Nom entièrement qualifié
`CallBuilder`	`com.netfective.bluage.gapwalk.runtime.statements.CallBuilder`
`Configuration`	`com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration`

Nom court	Nom entièrement qualifié
ConfigurationBuilder	com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder
ExecutionController	com.netfective.bluage.gapwalk.rt.call.ExecutionController
ExecutionControllerImpl	com.netfective.bluage.gapwalk.rt.call.internal.ExecutionControllerImpl
File	com.netfective.bluage.gapwalk.rt.io.File
MainProgramRunner	com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
Program	com.netfective.bluage.gapwalk.rt.provider.Program
ProgramContextStore	com.netfective.bluage.gapwalk.rt.context.ProgramContextStore
ProgramRegistry	com.netfective.bluage.gapwalk.rt.provider.ProgramRegistry
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordEntity	com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity
RuntimeContext	com.netfective.bluage.gapwalk.rt.context.RuntimeContext

Nom court	Nom entièrement qualifié
SimpleStateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.SimpleStateMachineController
StateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineController
StateMachineRunner	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineRunner

Mappages de noms entièrement qualifiés par des tiers

Nom court	Nom entièrement qualifié
@Autowired	org.springframework.beans.factory.annotation.Autowired
@Bean	org.springframework.context.annotation.Bean
BeanFactory	org.springframework.beans.factory.BeanFactory
@Component	org.springframework.stereotype.Component
ConfigurableApplicationContext	org.springframework.context.ConfigurableApplicationContext
@Import	org.springframework.context.annotation.Import
@Lazy	org.springframework.context.annotation.Lazy

Simplificateur de données

Sur les systèmes mainframe et de milieu de gamme (appelés systèmes « anciens » dans la rubrique suivante), les langages de programmation fréquemment utilisés tels que COBOL, PL/I ou RPG fournissent un accès de bas niveau à la mémoire. Cet accès se concentre sur la disposition de la mémoire accessible via des types natifs tels que zonée, compressée ou alphanumérique, éventuellement agrégée via des groupes ou des tableaux.

Un mélange d'accès à un élément de mémoire donné, à la fois via des champs dactylographiés et sous forme d'accès direct aux octets (mémoire brute), coexiste dans un programme donné. Par exemple, les programmes COBOL transmettent des arguments aux appelants sous forme d'ensembles d'octets contigus (LINKAGE) ou lisent et écrivent des données à partir de fichiers de la même manière (enregistrements), tout en interprétant ces plages de mémoire à l'aide de champs typés organisés dans des cahiers.

Ces combinaisons d'accès brut et structuré à la mémoire, le recours à une disposition précise de la mémoire au niveau des octets et les types existants, tels que zoné ou compressé, sont des fonctionnalités qui ne sont ni nativement ni facilement disponibles dans l'environnement de programmation Java.

Faisant partie de la solution AWS Blu Age pour moderniser les anciens programmes vers Java, la bibliothèque Data Simplifier fournit de telles constructions aux programmes Java modernisés et les expose d'une manière aussi familière que possible aux développeurs Java (getters/setters, tableaux d'octets, basés sur des classes). Il s'agit d'une dépendance essentielle du code Java modernisé généré à partir de tels programmes.

Pour des raisons de simplicité, la plupart des explications suivantes sont basées sur des constructions COBOL, mais vous pouvez utiliser la même API pour la modernisation de la mise en page des données PL1 et RPG, car la plupart des concepts sont similaires.

Rubriques

Classes principales

Pour faciliter la lecture, ce document utilise les noms abrégés Java des interfaces et des classes de l'API AWS Blu Age. Pour de plus amples informations, veuillez consulter FQN des types Java discutés.

Représentation de la mémoire de bas niveau

Au niveau le plus bas, la mémoire (une plage contiguë d'octets accessible de manière rapide et aléatoire) est représentée par l'`Record`interface. Cette interface est essentiellement une abstraction d'un tableau d'octets de taille fixe. En tant que tel, il fournit des setters et des getters capables d'accéder aux octets sous-jacents ou de les modifier.

Représentation de données structurées

Pour représenter des données structurées, telles que « 01 éléments de données » ou « 01 copybooks », comme on le trouve dans COBOL DATA DIVISION, des sous-classes de la `RecordEntity` classe sont utilisées. Ils ne sont généralement pas écrits à la main, mais générés par les outils de modernisation de AWS Blu Age à partir des anciennes constructions correspondantes. Il est toujours utile de connaître leur structure principale et leur API, afin de comprendre comment le code d'un programme modernisé les utilise. Dans le cas de COBOL, ce code est généré par Java à partir de leur DIVISION PROCEDURE.

Le code généré représente chaque « 01 élément de données » avec une `RecordEntity` sous-classe ; chaque champ élémentaire ou agrégat qui le compose est représenté sous la forme d'un champ Java privé, organisé sous forme d'arbre (chaque élément a un parent, à l'exception de la racine).

À des fins d'illustration, voici un exemple d'élément de données COBOL, suivi du code généré par AWS Blu Age correspondant qui le modernise :

```
01 TST2.
 02 FILLER PIC X(4).
 02 F1     PIC 9(2) VALUE 42.
 02 FILLER PIC X.
 02        PIC 9(3) VALUE 123.
 02 F2     PIC X VALUE 'A'.
```

```
public class Tst2 extends RecordEntity {

    private final Group root = new Group(getData()).named("TST2");
```

```
    private final Filler filler = new Filler(root,new AlphanumericType(4));
    private final Elementary f1 = new Elementary(root,new ZonedType(2, 0, false),new
BigDecimal("42")).named("F1");
    private final Filler filler1 = new Filler(root,new AlphanumericType(1));
    private final Filler filler2 = new Filler(root,new ZonedType(3, 0, false),new
BigDecimal("123"));
    private final Elementary f2 = new Elementary(root,new
AlphanumericType(1),"A").named("F2");



    /**
     * Instantiate a new Tst2 with a default record.
     * @param configuration the configuration
     */
    public Tst2(Configuration configuration) {
        super(configuration);
        setupRoot(root);
    }
    /**
     * Instantiate a new Tst2 bound to the provided record.
     * @param configuration the configuration
     * @param record the existing record to bind
     */
    public Tst2(Configuration configuration, RecordAdaptable record) {
        super(configuration);
        setupRoot(root, record);
    }

    /**
     * Gets the reference for attribute f1.
     * @return the f1 attribute reference
     */
    public ElementaryRangeReference getF1Reference() {
        return f1.getReference();
    }

    /* *
     * Getter for f1 attribute.
     * @return f1 attribute
     */
    public int getF1() {
        return f1.getValue();
    }
```

```
    /**
     * Setter for f1 attribute.
     * @param f1 the new value of f1
     */
    public void setF1(int f1) {
        this.f1.setValue(f1);
    }
    /**
     * Gets the reference for attribute f2.
     * @return the f2 attribute reference
     */
    public ElementaryRangeReference getF2Reference() {
        return f2.getReference();
    }

    /**
     * Getter for f2 attribute.
     * @return f2 attribute
     */
    public String getF2() {
        return f2.getValue();
    }

    /**
     * Setter for f2 attribute.
     * @param f2 the new value of f2
     */
    public void setF2(String f2) {
        this.f2.setValue(f2);
    }
}
```

Domaines élémentaires

Les champs de classe `Elementary` (ou`Filler`, lorsqu'ils ne sont pas nommés) représentent une « feuille » de l'ancienne structure de données. Ils sont associés à une plage contiguë d'octets sous-jacents (« plage ») et ont généralement un type (éventuellement paramétré) indiquant comment interpréter et modifier ces octets (en « décodant » et en « codant » respectivement une valeur depuis/vers un tableau d'octets).

Tous les types élémentaires sont des sous-classes de. `RangeType` Les types courants sont les suivants :

Type COBOL	Type de simplificateur de données
`PIC X(n)`	`AlphanumericType`
`PIC 9(n)`	`ZonedType`
`PIC 9(n) COMP-3`	`PackedType`
`PIC 9(n) COMP-5`	`BinaryType`

Champs d'agrégation

Les champs d'agrégation organisent la disposition en mémoire de leur contenu (autres agrégats ou champs élémentaires). Ils n'ont pas eux-mêmes de type élémentaire.

`Group`les champs représentent des champs contigus en mémoire. Chacun de leurs champs contenus est disposé dans le même ordre en mémoire, le premier champ étant `0` décalé par rapport à la position du champ de groupe en mémoire, le second champ étant décalé`0 + (size in bytes of first field)`, etc. Ils sont utilisés pour représenter des séquences de champs COBOL sous le même champ conteneur.

`Union`les champs représentent plusieurs champs accédant à la même mémoire. Chacun de leurs champs contenus est disposé de manière `0` décalée par rapport à la position du champ d'union en mémoire. Ils sont par exemple utilisés pour représenter la construction COBOL « REDEFINES » (les premiers enfants de l'Union étant l'élément de données redéfini, les seconds étant sa première redéfinition, etc.).

Les champs matriciels (sous-classes de`Repetition`) représentent la répétition, en mémoire, de la disposition de leur champ enfant (qu'il s'agisse d'un agrégat lui-même ou d'un élément élémentaire). Ils mettent en mémoire un certain nombre de mises en page pour enfants de ce type, chacune étant `index * (size in bytes of child)` décalée. Ils sont utilisés pour représenter les constructions COBOL « OCCURRENTS ».

Primitives

Dans certains cas de modernisation, les « primitives » peuvent également être utilisées pour présenter des éléments de données « racines » indépendants. Leur utilisation est très similaire, `RecordEntity` mais ils ne proviennent pas de celui-ci et ne sont pas basés sur le code généré. Au lieu de cela, ils sont directement fournis par le moteur d'exécution AWS Blu Age en tant que

sous-classes de l'`Primitive`interface. Des exemples de ces cours fournis sont `Alphanumeric` ou`ZonedDecimal`.

Liaison des données et accès

L'association entre les données structurées et les données sous-jacentes peut se faire de différentes manières.

Une interface importante à cette fin est `RecordAdaptable` celle qui est utilisée pour obtenir une `Record` « vue inscriptible » des données `RecordAdaptable` sous-jacentes. Comme nous le verrons ci-dessous, plusieurs classes sont implémentées`RecordAdaptable`. Réciproquement, les API AWS Blu Age et le code manipulant de la mémoire de bas niveau (tels que les arguments des programmes, les enregistrements d'E/S de fichiers, les virgules CICS, la mémoire allouée...) s'attendent souvent à un `RecordAdaptable` identifiant pour cette mémoire.

Dans le cas de la modernisation du COBOL, la plupart des éléments de données sont associés à une mémoire qui sera corrigée pendant la durée d'exécution du programme correspondant. À cette fin, les `RecordEntity` sous-classes sont instanciées une seule fois dans un objet parent généré (le programme Context) et se chargeront d'instancier leur sous-jacent`Record`, en fonction de la taille en octets. `RecordEntity`

Dans d'autres cas COBOL, tels que l'association d'éléments LINKAGE à des arguments de programme ou la modernisation de la construction SET ADDRESS OF, une `RecordEntity` instance doit être associée à un `RecordAdaptable` À cette fin, deux mécanismes existent :

- si l'`RecordEntity`instance existe déjà, la `RecordEntity.bind(RecordAdaptable)` méthode (héritée de`Bindable`) peut être utilisée pour que cette instance « pointe » vers celle-ci`RecordAdaptable`. Tout getter ou setter appelé sur le `RecordEntity` sera ensuite sauvegardé (octets en lecture ou en écriture) par les octets sous-jacents`RecordAdaptable`.
- si le `RecordEntity` doit être instancié, un constructeur généré acceptant a `RecordAdaptable` est disponible.

Inversement, les données `Record` actuellement liées aux données structurées sont accessibles. Pour cela`RecordAdaptable`, `RecordEntity` implémente donc `getRecord()` n'importe quelle instance de ce type.

Enfin, de nombreux verbes COBOL ou CICS nécessitent l'accès à un seul champ, à des fins de lecture ou d'écriture. La `RangeReference` classe est utilisée pour représenter un tel

accès. Ses instances peuvent être obtenues à partir de `getXXXReference()` méthodes `RecordEntity` générées (`XXX`étant le champ accédé) et transmises aux méthodes d'exécution. `RangeReference`est généralement utilisé pour accéder à l'ensemble `RecordEntity` ou`Group`, tandis que sa sous-classe `ElementaryRangeReference` représente les accès aux `Elementary` champs.

Notez que la plupart des observations ci-dessus s'appliquent aux `Primitive` sous-classes, car elles visent à implémenter un comportement similaire à celui fourni par le moteur d'exécution AWS Blu Age (au lieu du code généré). `RecordEntity` À cette fin, toutes les sous-classes d'`Primitive`implémentent `ElementaryRangeReference` et `RecordAdaptable` les `Bindable` interfaces de manière à être utilisables à la fois à la place des `RecordEntity` sous-classes et des champs élémentaires.

FQN des types Java discutés

Le tableau suivant indique les noms complets des types Java décrits dans cette section.

Nom court	Nom entièrement qualifié
`Alphanumeric`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.Alphanumeric`
`AlphanumericType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType`
`BinaryType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.BinaryType`
`Bindable`	`com.netfective.bluage.gapwalk.datasimplifier.data.Bindable`
`Elementary`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary`

Nom court	Nom entièrement qualifié
ElementaryRangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference
Filler	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Filler
Group	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group
PackedType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.PackedType
Primitive	com.netfective.bluage.gapwalk.datasimplifier.elementary.Primitive
RangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference
RangeType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.RangeType
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordAdaptable	com.netfective.bluage.gapwalk.datasimplifier.data.RecordAdaptable

Nom court	Nom entièrement qualifié
`RecordEntity`	`com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity`
`Repetition`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Repetition`
`Union`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Union`
`ZonedDecimal`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.ZonedDecimal`
`ZonedType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType`

AWS Fichiers de configuration et de configuration de Blu Age Runtime

Le framework Velocity et le code client sont des applications Web utilisant le framework Spring Boot. Il tire parti des fonctionnalités de Spring pour fournir une configuration, avec plusieurs emplacements possibles et règles de priorité. Il existe également des règles de priorité similaires pour fournir de nombreux autres fichiers, tels que les scripts Groovy, SQL, etc.

Le framework Velocity contient également des applications Web facultatives supplémentaires, qui peuvent être activées si nécessaire.

Rubriques

- Principes de base de configuration des applications
- Priorité des applications
- JNDI pour bases de données

- Utiliser des AWS secrets
- Autres fichiers (groovy, sql, etc.)
- Application Web supplémentaire
- Propriétés habilitantes
- Configurer l'authentification pour les applications Gapwalk

Principes de base de configuration des applications

La méthode par défaut pour gérer la configuration des applications consiste à utiliser des fichiers YAML dédiés à fournir dans le `config` dossier du serveur d'applications. Il existe deux principaux fichiers de configuration YAML :

- `application-main.yaml`
- `application-`*`profile`*`.yaml`(où *`profile`* la valeur est configurée lors de la génération de l'application).

Le premier fichier configure le framework, c'est-à-dire`Gapwalk-application.war`, tandis que le second est destiné à des options supplémentaires spécifiques à l'application cliente. Cela fonctionne avec l'utilisation de profils Spring : l'application Gapwalk utilise le `main` profil, tandis que l'application cliente utilise le *`profile`* profil.

L'exemple suivant montre un fichier YAML principal typique.

```
##########################################
##### JICS datasource configuration #####
##########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver
    url: jdbc:postgresql://localhost/jics
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource

#####################################################
##### Embedded Bluesam datasource configuration #####
#####################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

##########################################
##### Embedded Bluesam configuration #####
##########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
```

L'exemple suivant montre un fichier YAML client typique.

```
# Logback context logger integration.
logging.config : classpath:logback-[redacted].xml
# Limits Spring logger output.
logging.level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN
logging.level.org.springframework.statemachine : WARN
# If the datasource support mode is not static-xa, spring JTA transactions autoconfiguration must me disabled
spring.jta.enabled : false

spring:
  aws:
    client:
      datasources:
        names: primary
        primary:
          secret: arn:aws:secretsmanager:[redacted]

spring.jta.atomikos.datasource.primary.unique-resource-name: primary
spring.jta.atomikos.datasource.primary.xa-data-source-class-name: org.postgresql.xa.PGXADataSource
spring.jta.atomikos.datasource.primary.maxPoolSize: 20
spring.jta.atomikos.datasource.primary.autoCommit: false
```

Pour plus d'informations sur le contenu des fichiers YAML, consultezPropriétés habilitantes.

Priorité des applications

Pour ces fichiers de configuration, les règles de priorité Spring s'appliquent. Notamment :

- Le fichier `application-main` YAML apparaît dans le fichier war principal de Gapwalk avec les valeurs par défaut, et celui du `config` dossier le remplace.
- La même chose doit être faite pour la configuration de l'application cliente
- Des paramètres supplémentaires peuvent être transmis sur la ligne de commande au moment du lancement du serveur. Ils remplaceraient ceux de YAML.

Pour plus d'informations, consultez la documentation officielle de Spring Boot.

JNDI pour bases de données

La configuration de la base de données peut être fournie avec JNDI dans le fichier context.xml de Tomcat. Toute configuration de ce type remplacerait celle de YAML. Mais attention, son utilisation ne permettra pas d'encapsuler vos informations d'identification dans un gestionnaire secret (voir ci-dessous).

L'exemple suivant montre des exemples de configurations pour les JICS et les BluSam bases de données.

```
<Resource auth="Container" driverClassName="org.postgresql.Driver" initialSize="0"
 maxIdle="5"
   maxOpenPreparedStatements="-1" maxTotal="10" maxWaitMillis="-1" name="jdbc/jics"
   poolPreparedStatements="true" testOnBorrow="false" type="javax.sql.DataSource"
   url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX"
 password="XXXX" />
```

jdbc/jics

Ce serait `jdbc/jics` pour la base de données JICS et `jdbc/bluesam` (attention au « e ») pour la base de données blusam.

url="jdbc:postgresql : //xxxx.rds.amazonaws.com:5432/XXXX » Nom d'utilisateur = « XXXX » Mot de passe = « XXXX »

URL, nom d'utilisateur et mot de passe de la base de données.

Utiliser des AWS secrets

Certaines configurations de ressources contenant des informations d'identification peuvent être davantage sécurisées à l'aide de AWS secrets. L'idée est de stocker les données critiques dans un AWS secret et d'avoir une référence au secret dans la configuration YAML afin que le contenu secret soit sélectionné à la volée au démarrage de Tomcat.

Secrets pour Aurora

La configuration de la base de données Aurora (pour jics, blusam, customer db, etc.) utilisera le secret de base de données intégré, qui remplira automatiquement tous les champs pertinents à partir de la base de données correspondante.

> **Note**
>
> La `dbname` clé est facultative, selon la configuration de votre base de données, elle entrera dans le secret ou non. Vous pouvez l'y ajouter manuellement ou en fournissant le nom au fichier YAML.

Autres secrets

Les autres secrets concernent les ressources dotées d'un mot de passe unique (notamment les caches Redis protégés par mot de passe). Dans ce cas, l'autre type de secret doit être utilisé, avec une seule `password` clé.

Références YAML aux secrets

Ils `application-main.yaml` peuvent référencer l'ARN secret pour diverses ressources. Les plus importants sont les suivants :

- Informations d'identification de base de données JICS avec `spring.aws.jics.db.secret`
- JICS TS met en file d'attente les informations d'identification Redis avec `spring.aws.client.jics.queues.ts.redis.secret`
- Informations d'identification de la base de données Blusam avec `spring.aws.client.bluesam.db.secret`
- Mot de passe du cache Blusam avec `spring.aws.client.bluesam.redis.secret`

- Blusam verrouille le mot de passe du cache avec `spring.aws.client.bluesam.locks.redis.secret`

L'exemple suivant montre comment déclarer ces secrets dans un fichier YAML.

```
spring:
  aws:
    client:
      bluesam:
        locks:
          redis:
            secret: arn:aws:secretsmanager:XXXX
        db:
          dbname: bluesam
          secret: arn:aws:secretsmanager:XXXX
        redis:
          secret: arn:aws:secretsmanager:XXXX
      jics:
         queues:
          ts:
           redis:
            secret: arn:aws:secretsmanager:XXXX
    jics:
      db:
        secret: arn:aws:secretsmanager:XXXX
```

nom de base de données : bluesam

> Dans cet exemple, le nom de la base de données ne figure pas dans le secret et est fourni ici à la place.

Le client `application-`*`profile`*`.yaml` peut référencer l'ARN secret de la base de données client. Cela nécessite une propriété supplémentaire pour répertorier les sources de données, comme illustré dans l'exemple ci-dessous :

```
spring:
  aws:
    client:
      datasources:
        names: primary,host
        primary:
```

```
          secret: arn:aws:secretsmanager:XXXX
        host:

          secret: arn:aws:secretsmanager:XXXX
```

noms : principal, hôte

Exemple avec deux sources de données clientes nommées primary et host, chacune avec sa base de données et ses informations d'identification.

nom de base de données : mydb

Dans cet exemple, le nom de la base de données « hôte » ne figure pas dans le secret et est fourni ici à la place, tandis que pour la base de données « principale », il figure dans le secret.

Aucune clé secrète prise en charge par XA

- moteur (postgres/oracle/db2/mssql)
- port
- dbname
- Schéma actuel
- nom d'utilisateur
- mot de passe
- url

Car `postgres` seule la clé `sslMode` secrète valuée (`disable/allow/prefer/require/verify-ca/verify-full`) en plus de la propriété `spring.aws.rds.ssl.cert-path` yml permet de se connecter avec SSL.

Clés secrètes prises en charge par XA

Si la base de données client utilise XA, les propriétés subxa sont prises en charge par le biais de valeurs secrètes.

- hôte
- port
- dbname

- Schéma actuel
- nom d'utilisateur
- mot de passe
- url
- Connexion SSL (vrai/faux)

Cependant, pour les autres propriétés x (par exemple `maxPoolSize` ou`driverType`), la clé YAML normale `spring.jta.atomikos.datasource.XXXX.unique-resource-name` doit toujours être fournie.

La valeur secrète remplace les propriétés YAML.

Autres fichiers (groovy, sql, etc.)

Les autres fichiers utilisés par le projet client utilisent des règles de priorité similaires à celles de la configuration Spring. Exemples :

- Les scripts Groovy sont des `.groovy` fichiers contenus dans le `scripts` dossier ou les sous-dossiers.
- Les scripts SQL sont des `.sql` fichiers contenus dans le `sql` dossier ou les sous-dossiers.
- Les scripts Daemon sont des `.groovy` fichiers contenus dans le `daemons` dossier ou les sous-dossiers.
- Les fichiers de mappage de base de données de requêtes sont `queries-database.mapping` des fichiers nommés fichiers dans les sous-dossiers du `sql` dossier.
- Les modèles Jasper sont des `.jrxml` fichiers contenus dans le `templates` dossier ou les sous-dossiers.
- Les catalogues de jeux de données sont `.json` des fichiers contenus dans le `catalog` dossier.
- Les fichiers Lnk sont `.json` des fichiers du `lnk` dossier.

Tous ces emplacements peuvent être remplacés par le biais d'une propriété système ou d'une propriété yml du client.

- Pour les scripts Groovy : `configuration.scripts`
- Pour les scripts SQL : `configuration.sql`
- Pour les scripts Daemon : `configuration.daemons`

- Pour le fichier de mappage de base de données de requêtes : `configuration.databaseMapping`
- Pour les modèles Jasper : `configuration.templates`
- Pour les catalogues de jeux de données : `configuration.catalog`
- Pour les fichiers Lnk : `configuration.lnk`

Si la propriété n'est pas trouvée, les fichiers seront extraits de l'emplacement par défaut mentionné ci-dessus. La recherche sera d'abord effectuée avec le répertoire de travail Tomcat en tant que racine, puis dans le fichier war de l'application.

Application Web supplémentaire

Le framework Velocity contient des applications Web supplémentaires dans son `webapps-extra` dossier. Ces applications ne sont pas desservies par défaut par le serveur Tomcat.

L'activation de ces applications Web dépend du projet de modernisation et s'effectue en déplaçant le fichier war souhaité du `webapps-extra` dossier vers le `webapps` dossier. Après cela, la guerre sera servie par le serveur Tomcat au prochain démarrage.

Certaines configurations supplémentaires spécifiques au projet peuvent également être ajoutées dans un fichier de configuration YAML pour chaque guerre supplémentaire, comme cela est fait dans le `application-main.yml` fichier et expliqué ci-dessus. Les guerres supplémentaires sont les suivantes :

- `gapwalk-utility-pgm.war`: contient le support pour les programmes utilitaires ZOS et l'utilise `application-utility-pgm.yaml` comme configuration.
- `gapwalk-cl-command.war`: contient le support pour les programmes utilitaires AS/400 et l'utilise `application-cl-command.yaml` comme configuration.
- `gapwalk-hierarchical-support.war`: contient le support des transactions IMS/MFS et l'utilise comme configuration `application-jhdb.yaml`

Propriétés habilitantes

Dans les applications Spring Boot `application-main.yml` se trouve le fichier de configuration dans lequel nous définissons différents types de propriétés telles que le port d'écoute, la connectivité à la base de données, etc.

Rubriques

Notation YML

Dans la documentation suivante, une propriété telle que celle-ci `parent.child1.child2=true` est écrite comme suit au format YAML.

```
parent:
 child1:
   child2: true
```

Démarrage rapide/Cas d'utilisation

Les cas d'utilisation suivants présentent des exemples de clés et de valeurs applicables.

- Application-main.yml par défaut

```
----
#### DEFAULT APPLICATION-MAIN.YML FILE      #####
#### SHOWING USEFUL CONFIGURATION ELEMENTS #####
#### SHOULD BE OVERRIDDEN AND EXTERNALIZED   #####

###################################
##### Logging configuration #####
###################################
logging:
  config: classpath:logback-main.xml
  level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN

###################################
##### Spring configuration #####
###################################
spring:
  quartz:
    auto-startup: false
    scheduler-name: Default
```

```
    properties:
      org.quartz.threadPool.threadCount: 1
  jta:
    enabled: false
    atomikos.properties.maxTimeout : 600000
    atomikos.properties.default-jta-timeout : 100000
  jpa:
# DISABLE OpenEntityManagerInViewInterceptor
    open-in-view: false
    # Fix Postgres JPA Error:
    # Method org.postgresql.jdbc.PgConnection.createClob() is not yet implemented.
    properties.hibernate.temp.use_jdbc_metadata_defaults : false
#####################################
##### Jics tables configuration #####
#####################################

    # The dialect should match the jics datasource choice
    database-platform : org.hibernate.dialect.PostgreSQLDialect #
 org.hibernate.dialect.PostgreSQLDialect, org.hibernate.dialect.SQLServerDialect

    # those properties can be used to create and initialize jics tables
 automatically.
#    properties:
#      hibernate:
#      globally_quoted_identifiers: true
#        hbm2ddl:
#           import_files_sql_extractor :
 org.hibernate.tool.hbm2ddl.MultipleLinesSqlCommandExtractor
#           import_files : file:./setup/initJics.sql
#           auto : create

#########################
###### Level 2 cache #####
#########################
#        cache:
#          use_second_level_cache: true
#          use_query_cache: true
#          region:
#            factory_class: org.hibernate.cache.ehcache.EhCacheRegionFactory
#      javax:
#        persistence:
#          sharedCache:
#            mode: ENABLE_SELECTIVE
#########################
```

```
###### Redis settings #####
##########################
  session:
    store-type: none #redis

########################################
##### JICS datasource configuration #####
########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url: jdbc:postgresql://localhost/jics # jdbc:postgresql://localhost:5433/jics,
 jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

#####################################################
##### Embedded Bluesam datasource configuration #####
#####################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url : jdbc:postgresql://localhost/bluesam # jdbc:postgresql://localhost:5433/
jics, jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

########################################
##### Embedded Bluesam configuration #####
########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
```

```
  write-behind:
    enabled: true
  pgsql :
    dataSource : bluesamDs

#########################
##### Jics settings #####
#########################
rabbitmq.host: localhost
jics:
  cache: false #redis
  resource-definitions.store-type: jpa # default value: jpa, other possible value:
 redis
  redis.hostname: 127.0.0.1 # Redis server host.
  redis.password: redis # Login password of the redis server.
  redis.port: 6379 # Redis server port.
  redis.username: # Redis username
  redis.mode: standalone # Redis mode. Possible values: standalone, cluster
jics.disableSyncpoint : false
#jics.initList:
#jics.parameters.datform: DDMMYY
#jics.parameters.applid: VELOCITY
#jics.parameters.sysid: CICS
#jics.parameters.eibtrmid: TERM
#jics.parameters.userid: MYUSERID
#jics.parameters.username: MYUSERNAME
#jics.parameters.opid: XXX
#jics.parameters.cwa.length: 0
#jics.parameters.netname: MYNETNAME
#jics.parameters.jobname: MJOBNAME
#jics.parameters.sysname: SYSNAME

###############################################
##### Jics RunUnitLauncher pool settings #####
###############################################
#jics.runUnitLauncherPool.enable: false
#jics.runUnitLauncherPool.size: 20
#jics.runUnitLauncherPool.validationInterval: 1000

#########################
##### Jhdb settings #####
#########################
#jhdb.lterm: LTERMVAL
#jhdb.identificationCardData: SomeIDData
```

```
###################################
##### DateHelper configuration ####
###################################
#forcedDate: "2013-08-26T12:59:58+01:57"

##############################
##### Sort configuration ####
##############################
#externalSort.threshold: 256MB

###################################
##### Server timeout (10 min)  ####
###################################
spring.mvc.async.request-timeout: 600000

#################################
##### DATABASE STATISTICS  ####
#################################
databaseStatistics : false

######################
##### CALLS GRAPH ####
######################
callGraph : false

#################################
##### SQL SHIFT CODE POINT ####
#################################
# Code point 384 match unicode character \u0180
sqlCodePointShift : 384

#################################
##### LOCK TIMEOUT RECORD  ####
#################################
# Blu4IV record lock timeout
lockTimeout : 100

#################################
##### REPORTS OUTPUT PATH ####
#################################
reportOutputPath: reports

#################################
```

```
##### TASK EXECUTOR       ####
##############################
taskExecutor:
  corePoolSize: 5
  maxPoolSize: 10
  queueCapacity: 50
  allowCoreThreadTimeOut: false

##############################
##### PROGRAM NOT FOUND   ####
##############################
stopExecutionWhenProgNotFound: false

#########################################################
#####  DISP DEFAULT VALUE (to be removed one day) ####
#########################################################
defaultKeepExistingFiles: true

#####################################
#####  JOBQUEUE CONFIGURATION    ####
#####################################
jobqueue:
  api.enabled: false
  impl: none # possible values: quartz, none
  schedulers:  # list of schedulers
    -
      name: queue1
      threadCount: 5
    -
      name: queue2
      threadCount: 5

#############################################################################
##### QUERY BUILDING                                                     ##
# useConcatCondition : false by default
# if true, in the query, the where condition is build with key concatenation  ##
#############################################################################
# query.useConcatCondition: true
----
```

- Utiliser des fichiers de longueur variable avec les commandes LISTCAT

```
[**/*.*]
encoding=IBM930
reencoding=false

[global]
listcat.variablelengthpreprocessor.enabled=true
listcat.variablelengthpreprocessor.type=rdw
# use "rdw" if your .listcat file contains a set of records (RDW)
# use "bdw" if your .listcat file contains a set of blocks (bdw)
```

- Fournir une valeur d'indicateur d'octet nul dans l'utilitaire LOAD/UNLOAD

```
# Unload properties
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntax to specify the byte value
# - When the value is null in database : the value dumped to the file is filled by
 low value characters and the NBI is
# equal to the byte 6F (the ? character)
# - When the value is not null in database and the column is nullable: the NBI is
 equal to the byte 00 (low value) and NOT
# equal to the byte 40 (space)
unload:
  sqlCodePointShift: 0
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
```

Propriétés disponibles pour l'application principale

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-main.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont également disponibles.
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa, la configuration automatique des transactions Spring JTA doit être désactivée.
`datasource.jicsDs` + `-driver-class-name` + `-url` + `-username` + `-password` + `-type`	Source de données Spring standard avec sous-clés		Contient les informations de connexion à la base de données Jics. L'utilisation des secrets AWS est également fortement encouragée, comme expliqué dans xref :.. / configuration/configuration.adoc [Configuration].
`datasource.bluesamDs` + `-driver-class-name` + `-url`	Source de données Spring standard avec sous-clés		Contient les informations de connexion à la base de données Blusam. L'utilisa

Clé	Type	Valeur par défaut	Description
+ `-username` + `-password` + `-type`			tion des secrets AWS est également fortement encouragé e, comme expliqué dans xref :.. /configur ation/configuratio n.adoc [Configur ation].
`bluesam.d isabled`	boolean	false	S'il faut désactive r complètement le blusam.
`bluesam.cache`	chaîne		S'il n'est pas défini, le cache blusam ne sera pas utilisé. Les valeurs possibles (implémentations du cache) sont ehcache et redis.
`forcedDate`	chaîne		Force la date à la date indiquée s'il y en a une.
`frozenDate`	boolean	true	Spécifie s'il faut geler la date. S'appliqu e uniquement s'`forcedDate` il est également défini.
`externalS ort.threshold`	taille des données (par ex. 12 Mo)		Le seuil de tri : quand passer au tri externe (fusion).

Clé	Type	Valeur par défaut	Description
`jics.parameters.datform`	chaîne	MMDDYY	Le formulaire de date.
`jics.initList``	chaîne		La liste des jics d'initialisation, séparée par des virgules. Le cas échéant, il définit les noms de listes séparés par des virgules à activer au démarrage de Tomcat parmi les listes CICS. Exemple de valeur :`$UUU,DFH$IVPL,PEZ1` . Cela se répercutera sur les groupes contenus dans ces listes et leurs définitions de ressources sous-jacentes, qui seront ensuite visibles par le moteur d'exécution. Vide par défaut.
`jics.parameters.applid`	chaîne	VÉLOCITÉ	Le nom appliqué à pour identifier l'application dans JICS (au moins 4 caractères, pas de longueur maximale).
`jics.parameters.sysid`	chaîne	CICS	L'identification du système (SYSID).

Clé	Type	Valeur par défaut	Description
`jics.parameters.eibtrmid`	chaîne	TERME	L'identifiant du terminal (4 caractères maximum, 1 minimum).
`jics.parameters.userid`	chaîne		Le nom d'utilisateur (8 caractères maximum, pas de minimum). Lorsqu'aucune valeur n'est fournie (vide par défaut), l'identifiant de session HTTP est utilisé comme identifiant utilisateur.
`jics.parameters.username`	chaîne	MON NOM D'UTILISATEUR	Le nom d'utilisateur (10 caractères maximum, 1 minimum).
`jics.parameters.netname`	chaîne	MYNETNAME	Le nom du réseau (8 caractères maximum, 1 minimum).
`jics.parameters.opid`	chaîne	XXX	L'identification de l'opérateur à 3 caractères.
`jics.parameters.jobname`	chaîne	NOM DU POSTE	Le nom du poste.
`jics.parameters.sysname`	chaîne	SYSNAME	Le nom du système AS400 (nom système).

Clé	Type	Valeur par défaut	Description
`jics.parameters.cwa.length`	nombre	0	La longueur de la zone de travail commune (cwa).
`jics.parameters.charset`	chaîne	CP037	Jeu de caractères utilisé dans le monde entier par JICS.
`jics.parameters.tsqimpl`	chaîne	bluesam	Implémentation de la file d'attente de stockage temporaire (TSQ) JICS (les valeurs autorisées sont`bluesam`/`memory`) `redis`
`jics.queues.ts.redis.hostname`	chaîne	127,0.0.1	Le nom d'hôte du serveur Redis du cache jics.
`jics.queues.ts.redis.port`	nombre	6379	Le port du serveur Redis du cache jics.
`jics.queues.ts.redis.password`	chaîne	redis	Le mot de passe du serveur Redis du cache jics.
`jics.queues.ts.redis.username`	chaîne		Nom d'utilisateur du serveur Redis du cache jics. La valeur par défaut est vide (aucun nom d'utilisateur).

Clé	Type	Valeur par défaut	Description
`jics.queues.ts.redis.mode`	chaîne	autonome	Le mode cache du jics. Les valeurs possibles sont `standalone` ou`cluster`. La valeur par défaut est`standalone` .
`lockTimeout`	nombre	500	Le délai d'expiration du verrouillage, en millisecondes.
`sqlCodePointShift`	nombre		Facultatif. Le changement de point de code SQL. Déplace le point de code pour les caractères de contrôle que nous pouvons rencontrer lors de la migration des données d'un SGBDR existant vers un SGBDR moderne. Par exemple, vous pouvez spécifier de `384` faire correspondre le caractère Unicode`\u0180`.

Clé	Type	Valeur par défaut	Description
`sqlIntegerOverflowAllowed`	boolean	false	Spécifie s'il faut autoriser le dépassement des nombres entiers SQL, c'est-à-dire s'il est permis de placer des valeurs plus importantes dans la variable hôte.
`database.cursor.overflow.allowed`	boolean	true	Spécifie s'il faut autoriser le dépassement du curseur. Réglez sur `true` pour effectuer un appel suivant sur le curseur, quelle que soit sa position. Réglez `false` sur pour vérifier si le curseur est à la dernière position avant d'effectuer un prochain appel sur le curseur. Activez uniquement si le curseur est SCROLLABLE (SENSITIVE ou INSENSITIVE).
`reportOutputPath`	chaîne	`/reports`	Le chemin de sortie du rapport.

Clé	Type	Valeur par défaut	Description
`spring.session.store-type`	chaîne	none	Le cache de session pour les environnements à haute disponibilité. Les valeurs possibles sont `none` ou `redis`. La valeur par défaut est `none`.
`stopExecutionWhenProgNotFound`	boolean	true	Spécifie s'il faut arrêter l'exécution si aucun programme n'est trouvé. S'il est défini sur `true`, interrompt l'exécution si aucun programme n'est trouvé.
`forceHR`	boolean	false	Spécifie s'il faut utiliser SYSPRINT lisible par l'homme, soit sur la console, soit en sortie de fichier.
`rollbackOnRTE`	boolean	false	Spécifie s'il faut annuler la transaction d'unité d'exécution implicite sur les exceptions d'exécution.
`sctThreadLimit`	long	5	La limite de threads pour le déclenchement de scripts.

Clé	Type	Valeur par défaut	Description
`dataSimplifier.onInvalidNumericData`	chaîne	rejeter	Comment réagir lors du décodage de données numériques non valides. Les valeurs autorisées sont `reject`/`tolerates paces` /`tolerates paceslowv alues` /`toleratem ost` . La valeur par défaut est `reject`.
`filesDirectory`	chaîne		Le répertoire des fichiers d'entrée/sortie par lots.
`ims.messages.extendedSize`	boolean	false	Spécifie s'il faut définir la valeur ExtendedSize pour les messages ims.
`defaultKeepExistingFiles`	boolean	false	Spécifie s'il faut définir la valeur précédente par défaut de l'ensemble de données.

Clé	Type	Valeur par défaut	Description
`jics.db.ddlScriptLocation`	chaîne		L'emplacement du script Jics DDL. Permet de lancer le schéma de base de données jics à l'aide d'un script .sql. Vide par défaut. Par exemple, `./jics/sql/jics.sql` .
`jics.db.schemaTestQueryLocation`	chaîne		Emplacement du fichier SQL qui doit contenir une requête unique renvoyant le nombre d'objets du schéma jics (le cas échéant).
`jics.db.dataScriptLocation`	chaîne		Emplacement du script initJics.sql, préparé par Analyzer à partir de l'analyse des exportations CSD depuis le mainframe.

Clé	Type	Valeur par défaut	Description
`jics.db.dataTestQueryLocation`	chaîne		Emplacement d'un script SQL contenant une seule requête SQL censée renvoyer un nombre d'objets (par exemple : compter le nombre d'enregistrements dans la table du programme jics). Si le nombre est égal à 0, la base de données sera chargée à l'aide du `jics.db.dataScriptLocation` script, sinon le chargement de la base de données sera ignoré.
`jics.data.dataJsonInitLocation`	chaîne		
`jics.xa.agent.timeout`	nombre		
`query.useConcatCondition`	boolean	false	Spécifie si la condition clé est créée par concaténation de clés ou non.
`system.qdecfmt`	chaîne		

Clé	Type	Valeur par défaut	Description
`dispositi on.checke xistence`	boolean	false	Spécifie s'il convient de vérifier l'existen ce du fichier pour Dataset avec DISP SHR ou OLD.
`useContro lMVariable`	boolean	false	Spécifie s'il faut utiliser la spécifica tion Control-M pour le remplacement des variables.
`card.encoding`	chaîne	CP1145	Encodage de la carte : à utiliser avec `useContro lMVariable`.
`mapTransf o.prefixes`	chaîne	&,@,%%	Liste des préfixes à utiliser lors de la transformation des variables ControlM. Chacun d'eux est séparé par une virgule.
`checkinpu tfilesize`	boolean	false	Spécifie s'il faut lancer une vérification si la taille du fichier est un multiple de la taille de l'enregistrement.
`stepFailW henAbend`	boolean	true	Spécifie s'il faut déclencher un abend en cas d'échec ou de fin d'exécution d'une étape.

Clé	Type	Valeur par défaut	Description
`bluesam.fileLoading.commitInterval`	nombre	100 000	L'intervalle de validation du bluesam.
`uppercaseUserInput`	boolean	true	Spécifie si les données saisies par l'utilisateur doivent être en majuscules.
`jhdb.lterm`	chaîne		Permet de forcer un identifiant de terminal logique commun dans le cas d'une émulation IMS. S'il n'est pas défini, SessionId est utilisé.
`jhdb.identificationCardData`	chaîne		Utilisé pour coder en dur certaines « données de la carte d'identification de l'opérateur » dans le champ MID désigné par le paramètre CARD. Vide par défaut, aucune restriction de saisie.
`encoding`	chaîne	ASCII	L'encodage utilisé dans les projets (pas dans les fichiers groovy). Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8`

Clé	Type	Valeur par défaut	Description
`cl.configuration.context.encoding`	chaîne	CP297	L'encodage des fichiers CL. Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8` La valeur par défaut est `CP297`
`cl.zonedMode`	chaîne	EBCDIC_STRICT	Mode d'encodage ou de décodage des commandes du langage de contrôle (CL). Les valeurs autorisées sont`EBCDIC_STRICT /EBCDIC_MODIFIED /AS400`.
`ims.programs`	chaîne		Liste des programmes IMS à utiliser. Séparez chaque paramètre par un point-virgule (`;`) et chaque transaction par une virgule (). `,` Exemples :`PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`

Clé	Type	Valeur par défaut	Description
`jhdb.configuration.context.encoding`	chaîne	CP297	Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047`,`IBM930`,`ASCII`,
`jhdb.metadata.extrapath`	chaîne	fichier :. /configuration/	Paramètre de configuration qui spécifie un dossier racine supplémentaire spécifique à l'exécution pour les dossiers psbs et dbds.

Clé	Type	Valeur par défaut	Description
`jhdb.checkpointPersistence`	chaîne	none	Le mode de persistance du point de contrôle. Les valeurs autorisées sont`none`/`add`/`end`. `add`À utiliser pour conserver les points de contrôle lorsqu'un nouveau point est créé et ajouté au registre. Utilisez un point de contrôle `end` trop persistant lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir des effets sur les performances.

Clé	Type	Valeur par défaut	Description
`jhdb.checkpointPath`	chaîne	fichier :. /configuration/	Si ce n'`jhdb.checkpointPersistence` est pas le `casnone`, ce paramètre vous permet de configurer le chemin de persistance du point de contrôle (emplacement de stockage du fichier checkpoint.dat). Toutes les données des points de contrôle contenues dans le registre sont sérialisées et sauvegardées dans un fichier (checkpoint.dat) situé dans le dossier fourni. Notez que seules les données du point de contrôle (ScriptID, StepID, position de la base de données et zone du point de contrôle) sont concernées par cette sauvegarde.

Clé	Type	Valeur par défaut	Description
`jhdb.navigation.cachenexts`	nombre	5000	Durée du cache (en millisecondes) utilisée dans la navigation hiérarchique pour un SGBDR.
`jhdb.use-db-prefix`	boolean	true	Spécifie s'il faut activer un préfixe de base de données dans la navigation hiérarchique pour un SGBDR.
`jhdb.query.limitJoinUsage`	boolean	true	Spécifie s'il faut utiliser le paramètre d'utilisation limite des jointures sur les graphes RDBMS.
`taskExecutor.corePoolSize`	nombre	5	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille du pool principal.

Clé	Type	Valeur par défaut	Description
`taskExecutor.maxPoolSize`	nombre	10	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille maximale du pool (nombre maximal de threads parallèles).
`taskExecutor.queueCapacity`	nombre	50	Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille de la file d'attente. (= nombre maximum de transactions en attente lorsqu'il `taskExecutor.maxPoolSize` est atteint)

Clé	Type	Valeur par défaut	Description
`taskExecutor.allowCoreThreadTimeOut`	boolean	false	Spécifie s'il faut autoriser les threads principaux à expirer dans JCIS. Cela permet une croissance et une réduction dynamiques, même en combinaison avec une file d'attente différente de zéro (étant donné que la taille maximale du pool n'augmentera que lorsque la file d'attente sera pleine).
`jics.runUnitLauncherPool.enable`	boolean	false	Spécifie s'il faut activer le pool de lanceurs d'unités exécutées dans JICS.
`jics.runUnitLauncherPool.size`	nombre	20	Taille du pool de lanceurs d'unités exécutées dans JICS.
`jics.runUnitLauncherPool.validationInterval`	nombre	1 000	Intervalle de validation du pool de lanceurs d'unités exécutées dans JICS, exprimé en millisecondes.

Clé	Type	Valeur par défaut	Description
`spring.aws.application.credentials`	chaîne	null	Chargez les informations d'identification AWS à partir du fichier de profils d'identification dans JICS.
`jics.queues.sqs.region`	chaîne	eu-west-1	La région AWS pour le service AWS Simple Queue, utilisé dans JICS.
`mq.queues.sqs.region`	chaîne	eu-west-3	La région AWS pour le service AWS SQS MQ.
`quartz.scheduler.stand-by-if-error`	boolean	false	Spécifie s'il faut déclencher l'exécution des tâches si le planificateur de tâches est en mode veille. Si vrai, lorsque cette option est activée, l'exécution de la tâche n'est pas déclenchée.
`databaseStatistics`	boolean	false	Spécifie s'il faut autoriser les générateurs SQL à collecter et à afficher des informations statistiques.

Clé	Type	Valeur par défaut	Description
`dbDateFormat`	chaîne	yyyy-MM-dd	Le format de date cible de la base de données.
`dbTimeFormat`	chaîne	HH : mm : SS	Le format d'heure cible de la base de données.
`dbTimestampFormat`	chaîne	YYYY-MM-DD HH:MM:SS.ssssss	Le format d'horodatage cible de la base de données.
`dateTimeFormat`	chaîne	ISO	dateTimeFormat décrit comment répartir la date, l'heure et le type d'horodatage de la base de données dans des entités simplificatrices de données. Les valeurs autorisées sont`ISO/EUR//EUR/USA/LOCAL`
`localDateFormat`	chaîne		Liste des formats de date locaux. Séparez chaque format par. \
`localTimeFormat`	chaîne		Liste des formats d'heure locale. Séparez chaque format par \

Clé	Type	Valeur par défaut	Description
`localTimestampFormat`	chaîne		Liste des formats d'horodatage locaux. Séparez chaque format par\.
`pgmDateFormat`	chaîne	yyyy-MM-dd	Le format de la date et de l'heure.
`pgmTimeFormat`	chaîne	HH.mm.ss	Le format d'heure utilisé pour l'exécution de pgm (programmes).
`pgmTimestampFormat`	chaîne	YYYY-MM-DD-HH.MM.SS.SSSSSSS	Le format d'horodatage.
`cacheMetadata`	boolean	true	Spécifie s'il faut mettre en cache les métadonnées de base de données.
`forceDisableSQLTrimStringType`	boolean	false	Spécifie s'il faut désactiver le découpage de tous les paramètres de chaîne SQL.

Clé	Type	Valeur par défaut	Description
`fetchSize`	nombre		La valeur FetchSize pour les curseurs. À utiliser lors de la récupération de données à l'aide de fragments par des utilitaires de chargement/déchargement.
`check-groovy-file`	boolean	true	Spécifie s'il faut vérifier le contenu des fichiers groovy avant de les enregistrer.
`qtemp.uuid.length`	nombre	9	La longueur de l'identifiant unique QTEMP.
`qtemp.dblog`	boolean	false	S'il faut activer la journalisation de la base de données QTEMP.
`qtemp.cleanup.threshold.hours`	nombre	0	Pour spécifier quand `qtemp.dblog` est activé. Durée de vie de la partition de base de données (en heures).
`sort.function`	chaîne		Nom de la fonction de tri pour la base de données blu4iv.

Propriétés disponibles pour les applications Web facultatives

En fonction de votre application modernisée, vous devrez peut-être configurer une ou plusieurs applications Web facultatives prenant en charge des dépendances telles que z/OS, AS/400 ou IMS/MFS. Les tableaux suivants contiennent la liste des paramètres clé/valeur disponibles pour configurer chaque application Web facultative.

gapwalk-utility-pgm.guerre

Cette application Web facultative prend en charge les programmes utilitaires Z/OS.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalisation standard sont également disponibles.
`spring.jta.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa, la configuration automatique des transactions Spring JTA doit être désactivée.
`spring.datasource.primary.jndi-name`	chaîne	jdbc/primaire	Le nom jndi (Java Naming And Directory Interface) de la source de données principal

Clé	Type	Valeur par défaut	Description
			e, si vous utilisez JNDI.
`primary.datasource + -driver-class-name + -url + -username + -password`	Source de données Spring standard avec sous-clés		Contient les informations de connexion pour la base de données de l'application, si vous n'utilisez pas JNDI. Doit avoir la même configuration que dans le fichier yml de l'application modernisée. L'utilisation des secrets AWS est également fortement encouragée, comme expliqué dans xref :.. / configuration/configuration.adoc [Configuration].
`encoding`	chaîne	ASCII	Le codage utilisé dans les programmes utilitaires. Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8`
`sysPunchEncoding`	chaîne	ASCII	Le jeu de caractères de codage Syspunch. Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8`

Clé	Type	Valeur par défaut	Description
`zonedMode`	chaîne	EBCDIC_STRICT	Mode de codage ou de décodage des types de données zonés. Les valeurs autorisées sont`EBCDIC_STRICT` /`EBCDIC_MODIFIED` /`AS400`.
`unload.chunkSize`	nombre	0	Taille du morceau utilisée pour l'utilitaire de déchargement.
`unload.sqlCodePointShift`	nombre	0	L'utilitaire SQL Pointshift for Unload. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible de DB2 est Postgresql.
`unload.columnFiller`	chaîne	espace	Le remplisseur de colonnes utilitaire de déchargement.
`unload.varCharIsNull`	boolean	false	Utilisez ce paramètre dans le programme INFTILB. S'il est défini sur, tous les champs non nullables contenant des valeurs vides (espaces) renvoient une chaîne vide. `true`

Clé	Type	Valeur par défaut	Description
`unload.useDatabaseConfiguration`	boolean	false	Spécifie s'il faut utiliser la configuration de date ou d'heure de application-main.yml dans l'utilitaire de déchargement.
`unload.format.date`	chaîne	MM/dd/yyyy	Si cette option `unload.useDatabaseConfiguration` est activée, le format de date à utiliser dans l'utilitaire de déchargement.
`unload.format.time`	chaîne	HH.mm.ss	Si cette option `unload.useDatabaseConfiguration` est activée, le format d'heure à utiliser dans l'utilitaire de déchargement.
`unload.format.timestamp`	chaîne	YYYY-MM-DD-HH.MM.SS.SSSSSSS	Si cette option `unload.useDatabaseConfiguration` est activée, le format d'horodatage à utiliser dans l'utilitaire de déchargement.

Clé	Type	Valeur par défaut	Description
`unload.nbi.whenNull`	hexadécimal	6F	La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données est nulle.
`unload.nbi.whenNotNull`	hexadécimal	00	La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données n'est pas nulle.
`unload.nbi.writeNullIndicator`	boolean	false	Spécifie s'il faut écrire l'indicateur nul dans le fichier de sortie de déchargement.
`unload.fetchSize`	nombre	0	Vous permet de régler la taille de lecture lorsque vous manipulez des curseurs dans l'utilitaire de déchargement.
`treatLargeNumberAsInteger`	boolean	false	Spécifie s'il faut traiter les grands nombres comme`Integer`. Ils sont traités comme `BigDecimal` par défaut.
`load.batchSize`	nombre	0	Taille du lot de l'utilitaire de chargement.

Clé	Type	Valeur par défaut	Description
`load.format.localDate`	chaîne	dd.mm.yyyy \ dd/mm/yyyy \ yyyy-mm-dd	Le format de date local de l'utilitaire de chargement à utiliser.
`load.format.localTime`	chaîne	HH : mm : SS \ HH.mm.ss	Format d'heure locale de l'utilitaire de chargement à utiliser.
`load.format.dbDate`	chaîne	yyyy-MM-dd	Format de base de données de l'utilitaire de chargement à utiliser.
`load.format.dbTime`	chaîne	HH : mm : SS	Durée d'utilisation de la base de données de l'utilitaire de chargement.
`load.sqlCodePointShift`	nombre	0s	L'utilitaire SQL Pointshift for Load. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible de DB2 est Postgresql.
`forcedDate`	chaîne		Force la date à la date indiquée s'il y en a une.
`frozenDate`	boolean	true	Spécifie s'il faut geler la date. S'appliqu e uniquement s'`forcedDate` il est également défini.

Clé	Type	Valeur par défaut	Description
`jcl.type`	chaîne	mvs	Type de fichier .jcl. Les valeurs autorisées sont`jcl`/`vse`. Les commandes PRINT/ REPRO de l'utilitaire IDCAMS renvoient 4 si le fichier est vide pour un jcl non vse.
`hasGraphic`	boolean	false	Si l'utilitaire INFUTILB doit gérer les colonnes GRAPHIC DB2.

gapwalk-cl-command.guerre

Cette application Web optionnelle prend en charge les programmes utilitaires AS/400.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalis ation standard sont également disponibl es.
`spring.jt a.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas

Clé	Type	Valeur par défaut	Description
			static-xa, la configuration automatique des transactions Spring JTA doit être désactivée.
`spring.datasource.primary.jndi-name`	chaîne	jdbc/primaire	Le nom jndi (Java Naming And Directory Interface) de la source de données principale, si vous utilisez JNDI.
`primary.datasource` + `-driver-class-name` + `-url` + `-username` + `-password`	Source de données Spring standard avec sous-clés		Contient les informations de connexion pour la base de données de l'application, si vous n'utilisez pas JNDI. Doit avoir la même configuration que dans le fichier yml de l'application modernisée. L'utilisation des secrets AWS est également fortement encouragée, comme expliqué dans xref :... / configuration/configuration.adoc [Configuration].

Clé	Type	Valeur par défaut	Description
`encoding`	chaîne	ASCII	Le codage utilisé dans les programme s utilitaires. Exige un encodage valide `CP1047IBM930,,ASCII,UTF-8`
`zonedMode`	chaîne	EBCDIC_STRICT	Mode de codage ou de décodage des types de données zonés. Les valeurs autorisée s sont`EBCDIC_ST RICT /EBCDIC_MO DIFIED /AS400`.
`commands-off`	chaîne		Liste des commandes à désactiver, séparées par des virgules. Les valeurs autorisée s sont `PGM_BASIC RCVMSG,SNDRCVF,CHGVAR,QC` Utile lorsque vous souhaitez désactive r ou remplacer un programme existant. `PGM_BASIC` est un programme de vélocité spécifique conçu à des fins de débogage.

gapwalk-hierarchical-support.guerre

Cette application Web optionnelle prend en charge les transactions IMS/MFS.

Ce tableau fournit une vue exhaustive des paramètres clés/valeurs pour cette application.

Clé	Type	Valeur par défaut	Description
`logging.config`	Chemin	chemin de classe : logback-utility.xml	Clé standard pour la référence au fichier de configuration du logback. D'autres clés de journalis ation standard sont également disponibl es.
`spring.jt a.enabled`	boolean	false	Clé standard Si le mode de prise en charge de la source de données n'est pas static-xa, la configura tion automatique des transactions Spring JTA doit être désactivée.
`jhdb.conf iguration .context. encoding`	chaîne		Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047`,`IBM930`,`ASCII`,
`jhdb.chec kpointPer sistence`	chaîne	none	Le mode de persistan ce du point de contrôle. Les valeurs autorisées sont`none`/`add`/`end`. `add`À utiliser pour conserver les points de contrôle lorsqu'un

Clé	Type	Valeur par défaut	Description
			nouveau point est créé et ajouté au registre. Utilisez un point de contrôle end trop persistant lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir des effets sur les performances.

Configurer l'authentification pour les applications Gapwalk

Cette section décrit comment configurer l'authentification OAuth2 pour les applications Gapwalk à l'aide d'un fournisseur d'identité (IdP) tel que Cognito, Azure AD, Keycloak, etc.

Rubriques

Prérequis

Dans ce didacticiel, nous utiliserons Amazon Cognito comme IdP et PlanetDemo comme projet modernisé.

Vous pouvez utiliser n'importe quel autre fournisseur d'identité externe. Les ClientRegistration informations doivent être obtenues auprès de votre IdP et sont requises pour l'authentification Gapwalk. Pour plus d'informations, consultez la documentation officielle de votre IdP.

Les ClientRegistration informations :

identificateur du client

l'identifiant du ClientRegistration, dans notre exemple ce sera PlanetDemo.

client-secret

le secret de votre client.

point final d'autorisation

L'URI du point de terminaison d'autorisation pour le serveur d'autorisation.

point de terminaison symbolique

L'URI du point de terminaison du jeton pour le serveur d'autorisation.

point de terminaison jwks

L'URI utilisé pour obtenir la clé Web JSON (JWK) contenant les clés de validation de la signature Web JSON émise par le serveur d'autorisation.

URI de redirection

URI vers lequel le serveur d'autorisation redirige l'utilisateur final si l'accès est accordé.

Configuration d'Amazon Cognito

Nous allons d'abord créer et configurer un groupe d'utilisateurs et un utilisateur Amazon Cognito que nous utiliserons avec notre application Gapwalk déployée à des fins de test.

> **Note**
>
> Si vous utilisez un autre IdP, vous pouvez ignorer cette étape.

Création d'un groupe d'utilisateurs

1. Accédez à Amazon Cognito dans le AWS Management Console et authentifiez-vous à l'aide de vos informations d'identification. AWS
2. Choisissez Groupes d'utilisateurs.
3. Sélectionnez Create a user pool.
4. Dans Configurer l'expérience de connexion, conservez le type de fournisseur par défaut du groupe d'utilisateurs Cognito. Vous pouvez choisir une ou plusieurs options de connexion au groupe d'utilisateurs de Cognito ; pour l'instant, choisissez Nom d'utilisateur, puis Suivant.
5. Dans Configurer les exigences de sécurité, conservez les valeurs par défaut et désactivez l'authentification multifactorielle en choisissant Pas de MFA, puis en choisissant Suivant.
6. Désactivez Activer l'enregistrement automatique par mesure de sécurité et choisissez Suivant.
7. Choisissez Envoyer un e-mail avec Cognito. Choisissez Suivant.
8. Dans Intégrer votre application, choisissez un nom pour votre groupe d'utilisateurs. Dans les pages d'authentification hébergées, choisissez Utiliser l'interface utilisateur hébergée de Cognito.
9. Pour des raisons de simplicité, dans Domaine, choisissez Utiliser un domaine Cognito et entrez un préfixe de domaine, par exemple,. `https://planetsdemo` L'application de démonstration doit être ajoutée en tant que client.

    1. Dans Client d'application initial, choisissez Client confidentiel. Entrez le nom du client de l'application, par exemple `` `planetsdemo` `` et choisissez Generate a client secret.
    2. Dans URL de rappel autorisée, entrez l'URL vers laquelle rediriger l'utilisateur après l'authentification. Une URL http://localhost:8080/planetsdemo temporaire peut également être utilisée, puis modifiée ultérieurement.
    3. Conservez les valeurs par défaut dans les sections Paramètres avancés du client de l'application et Autorisations de lecture et d'écriture des attributs.
    4. Choisissez Suivant.
10. Dans Vérifier et créer, vérifiez vos choix, puis choisissez Créer un groupe d'utilisateurs.

Pour plus d'informations, voir Création d'un groupe d'utilisateurs.

Création d'utilisateurs

L'enregistrement automatique étant désactivé, créez un utilisateur Amazon Cognito. Accédez à Amazon Cognito dans le. AWS Management Console Choisissez le groupe d'utilisateurs que vous avez créé, puis dans Utilisateurs, choisissez Créer un utilisateur.

Dans Informations utilisateur, choisissez Envoyer une invitation par e-mail, entrez un nom d'utilisateur et une adresse e-mail, puis choisissez Générer un mot de passe. Choisissez Create user (Créer un utilisateur).

Intégration d'Amazon Cognito dans l'application Gapwalk

Maintenant que votre groupe d'utilisateurs et vos utilisateurs Amazon Cognito sont prêts, accédez au `main-application.yml` fichier de votre application modernisée et ajoutez le code suivant :

```
spring:
  security:
    oauth2:
      client:
        registration:
          cognito:
            client-id: client-id
            client-name: client-name
            client-secret: client-secret
            provider: cognito
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "{baseUrl}/login/oauth2/code/{registrationId}"
        provider:
          cognito:
            issuerUri: ${gapwalk-application.security.issuerUri}
            authorization-uri: ${gapwalk-application.security.domainName}/oauth2/
authorize
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/
jwks.json
            token-uri: ${gapwalk-application.security.domainName}/oauth2/token
            user-name-attribute: cognito:username
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/jwks.json

gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
```

```
gapwalk-application.security.issuerUri: https://cognito-idp.region.amazonaws.com/pool-
id
gapwalk-application.security.domainName: your-cognito-domain
```

Remplacez les espaces réservés suivants comme décrit :

1. Accédez à Amazon Cognito dans le AWS Management Console et authentifiez-vous à l'aide de vos informations d'identification. AWS
2. Choisissez Groupes d'utilisateurs et choisissez le groupe d'utilisateurs que vous avez créé. Vous pouvez trouver votre identifiant de *pool dans ID du* groupe d'utilisateurs.
3. Choisissez l'intégration des applications où vous pouvez trouver votre *your-cognito-domain*application, accédez à Clients et analyses des applications et choisissez votre application.
4. Dans App client : YourApp, vous pouvez trouver le ***nom du client, l'identifiant du client et le secret du client*** (Afficher le secret du ***client***).
5. *region-id* correspond à l'identifiant de région dans lequel vous avez créé votre utilisateur et votre groupe d'utilisateurs Amazon Cognitoeu-west-3, par exemple.
6. Pour redirect-uri, entrez l'URI vers lequel rediriger l'utilisateur après l'authentification.

Vous pouvez déployer votre application Gapwalk dès maintenant et utiliser l'utilisateur créé précédemment pour vous connecter à votre application.

> **Note**
>
> Si, lors de la connexion, vous obtenez une erreur avec l'exception « Attribut 'cognito:username' manquant dans les attributs », supprimez la ligne suivante : cognito:username user-name-attribute

AWS API d'exécution Blu Age

Le AWS Blu Age Runtime utilise plusieurs applications Web pour exposer les points de terminaison REST, fournissant ainsi des moyens d'interagir avec les applications modernisées à l'aide de clients REST (par exemple en appelant des tâches à l'aide d'un planificateur).

Le but de ce document est de répertorier les points de terminaison REST disponibles, en fournissant des détails sur :

- Leur rôle
- La façon de les utiliser correctement

La liste des points de terminaison est organisée en catégories, en fonction de la nature du service fourni et de l'application Web exposant les points de terminaison.

Nous partons du principe que vous avez déjà une connaissance de base de l'utilisation des points de terminaison REST (à l'aide d'outils dédiés tels que POSTMAN, Thunder Client, CURL, navigateurs Web, etc.) ou en écrivant votre propre code pour effectuer un appel d'API.

Rubriques

Création d'URL

Chaque application Web ci-dessous définit un chemin racine, partagé par tous les points de terminaison. Chaque point de terminaison ajoute ensuite son propre chemin dédié. L'URL résultante à utiliser est le résultat de la concaténation des chemins. Par exemple, en considérant le premier point de terminaison de l'application Gapwalk, nous avons :

- `/gapwalk-application`pour le chemin de l'application Web racine.
- `/scripts`pour le chemin du point de terminaison dédié.

L'URL résultante à utiliser sera `http://server:port/gapwalk-application/scripts`

serveur

pointe vers le nom du serveur (celui hébergeant l'application Web donnée).

port

le port exposé par le serveur.

Application Gapwalk

Les points de terminaison de l'application Web Gapwalk utilisent le chemin racine. `/gapwalk-application`

Rubriques

Points de terminaison liés aux tâches par lots (JCL modernisés et similaires)

Les tâches Batch peuvent être exécutées de manière synchrone ou asynchrone (voir les détails ci-dessous). Les tâches par lots sont exécutées à l'aide de scripts groovy issus de la modernisation des anciens scripts (JCL).

Rubriques

Répertorier les scripts déployés

- Méthode prise en charge : GET

- Trajectoire : `/scripts`
- Arguments : aucun
- Ce point de terminaison renvoie la liste des scripts groovy déployés sur le serveur, sous forme de chaîne. Ce point de terminaison est principalement destiné à être utilisé à partir d'un navigateur Web, car la chaîne qui en résulte est une page HTML, avec des liens actifs (un lien par script pouvant être lancé, voir l'exemple ci-dessous).

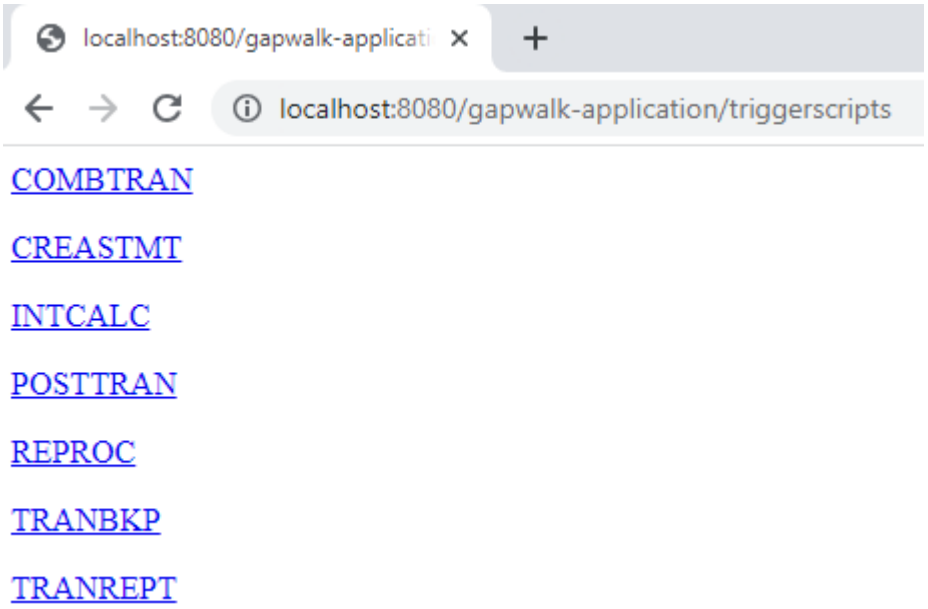
Exemple de réponse :

```
<p><a href=./script/COMBTRAN>COMBTRAN</a></p><p><a href=./script/CREASTMT>CREASTMT</
a></p><p><a href=./script/INTCALC>INTCALC</a></p><p><a href=./script/
POSTTRAN>POSTTRAN</a></p><p><a href=./script/REPROC>REPROC</a></p><p><a href=./script/
TRANBKP>TRANBKP</a></p><p><a href=./script/TRANREPT>TRANREPT</a></p><p><a href=./
script/functions>functions</a></p>
```

> **Note**
>
> Les liens représentent l'URL à utiliser pour lancer chaque script répertorié de manière synchrone.

- Méthode prise en charge : GET
- Trajectoire : `/triggerscripts`
- Arguments : aucun
- Ce point de terminaison renvoie la liste des scripts groovy déployés sur le serveur, sous forme de chaîne. Ce point de terminaison est principalement destiné à être utilisé à partir d'un navigateur Web, car la chaîne qui en résulte est une page HTML, avec des liens actifs (un lien par script pouvant être lancé, voir l'exemple ci-dessous).

  Contrairement à la réponse précédente du point de terminaison, les liens représentent l'URL à utiliser pour lancer chaque script répertorié de manière asynchrone.



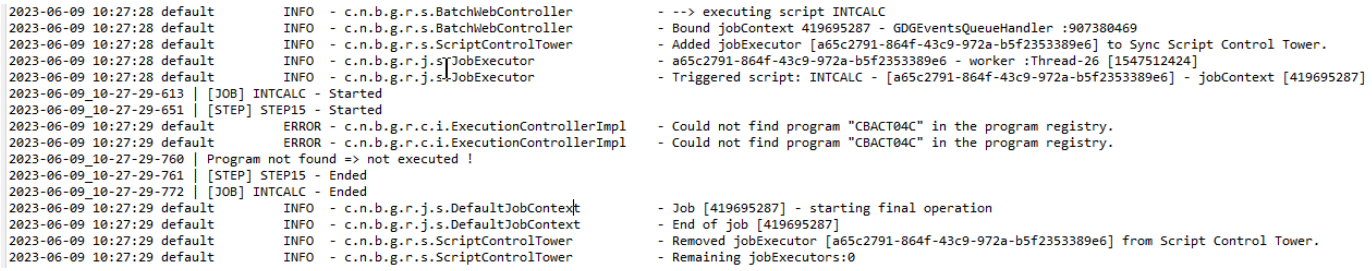
Lancer un script de manière synchrone

Ce point de terminaison possède deux variantes avec des chemins dédiés pour l'utilisation de GET et POST (voir ci-dessous).

- Méthode prise en charge : GET
- Trajectoire : `/script/{scriptId:.+}`
- Méthode prise en charge : POST
- Trajectoire : `/post/script/{scriptId:.+}`
- Arguments :
  - identifiant du script à lancer
  - optionnellement : paramètres à transmettre au script, en utilisant les paramètres de requête (considérés comme un`Map<String,String>`). Les paramètres donnés seront automatiquement ajoutés aux liaisons du script groovy invoqué.
- L'appel lancera le script avec l'identifiant donné, en utilisant des paramètres supplémentaires s'ils sont fournis, et attendra la fin de l'exécution du script avant de renvoyer un message (`String`) qui sera soit :
  - « C'est fait. » (si l'exécution de la tâche s'est bien déroulée).
  - Un message d'erreur JSON contenant des détails sur ce qui s'est mal passé lors de l'exécution de la tâche. De plus amples informations peuvent être extraites des journaux du serveur afin de comprendre ce qui s'est mal passé lors de l'exécution de la tâche.

```
{
    "exitCode": -1,
    "stepName": "STEP15",
    "program": "CBACT04C",
    "status": "Error"
}
```

En consultant les journaux du serveur, nous pouvons découvrir qu'il s'agit d'un problème de déploiement (le programme attendu n'a pas été correctement déployé, il est donc introuvable, ce qui entraîne l'échec de l'exécution de la tâche) :



```
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - --> executing script INTCALC
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - Bound jobContext 419695287 - GDGEventsQueueHandler :907380469
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Added jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] to Sync Script Control Tower.
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - a65c2791-864f-43c9-972a-b5f2353389e6 - worker :Thread-26 [1547512424]
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - Triggered script: INTCALC - [a65c2791-864f-43c9-972a-b5f2353389e6] - jobContext [419695287]
2023-06-09_10-27-29-613 | [JOB] INTCALC - Started
2023-06-09_10-27-29-651 | [STEP] STEP15 - Started
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09_10-27-29-760 | Program not found => not executed !
2023-06-09_10-27-29-761 | [STEP] STEP15 - Ended
2023-06-09_10-27-29-772 | [JOB] INTCALC - Ended
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - Job [419695287] - starting final operation
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - End of job [419695287]
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Removed jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] from Script Control Tower.
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Remaining jobExecutors:0
```

> **Note**
>
> Les appels synchrones doivent être réservés aux tâches de courte durée. Les tâches exécutées pendant de longues périodes devraient plutôt être lancées de manière asynchrone (voir point de terminaison dédié ci-dessous).

Lancer un script de manière asynchrone

- Méthodes prises en charge : GET/POST
- Trajectoire : `/triggerscript/{scriptId:.+}`
- arguments :
  - identifiant du script à lancer
  - optionnellement : paramètres à transmettre au script, en utilisant les paramètres de requête (considérés comme un`Map<String,String>`). Les paramètres donnés seront automatiquement ajoutés au https://docs.groovy-lang.org/latest/html/api/groovy/lang/Binding.html[bindings] du script groovy invoqué.

- Contrairement au mode synchrone ci-dessus, le point de terminaison n'attend pas la fin de l'exécution de la tâche pour envoyer une réponse. L'exécution de la tâche est lancée immédiatement, si un thread disponible est trouvé pour le faire, et une réponse est immédiatement envoyée à l'appelant, avec l'identifiant d'exécution de la tâche, un identifiant unique représentant l'exécution de la tâche, qui peut être utilisé pour demander l'état d'exécution de la tâche ou forcer l'arrêt d'une exécution de tâche censée mal fonctionner. Le format de la réponse est le suivant :

```
Triggered script <script identifier> [unique job execution id] @ <date and time>
```

- Étant donné que l'exécution asynchrone de la tâche repose sur un nombre limité de threads, l'exécution de la tâche risque de ne pas être lancée si aucun thread disponible n'est trouvé. Dans ce cas, le message renvoyé ressemblera plutôt à :

```
Script [<script identifier>] NOT triggered - Thread limit reached (<actual thread
 limit>) - Please retry later or increase thread limit.
```

  Consultez le `settriggerthreadlimit` point de terminaison ci-dessous pour savoir comment augmenter la limite de threads.

Exemple de réponse :

```
Triggered script INTCALC [d43cbf46-4255-4ce2-aac2-79137573a8b4] @ 06-12-2023 16:26:15
```

L'identifiant unique d'exécution des tâches permet de récupérer rapidement les entrées de journal associées dans les journaux du serveur si nécessaire. Il est également utilisé par plusieurs autres points de terminaison détaillés ci-dessous.

Liste des scripts déclenchés

- Méthodes prises en charge : GET
- Chemins :`/triggeredscripts/{status:.+}`, `/triggeredscripts/{status:.+}/{namefilter}`
- Arguments :
  - État (obligatoire) : le statut des scripts déclenchés à récupérer. Les valeurs possibles sont les suivantes :
    - `all`: affiche tous les détails d'exécution des tâches, que celles-ci soient toujours en cours d'exécution ou non.

  - `running`: affiche uniquement les détails des tâches en cours d'exécution.
  - `done`: affiche les détails des jobs uniquement pour les jobs dont l'exécution est terminée.
  - `killed`: affiche uniquement les détails des tâches dont l'exécution a été interrompue de force à l'aide du point de terminaison dédié (voir ci-dessous).
  - `triggered`: affiche uniquement les détails des tâches qui ont été déclenchées mais qui n'ont pas encore été lancées.
  - `failed`: affiche uniquement les détails des tâches dont l'exécution a été marquée comme ayant échoué.
  - _namefilter (facultatif) _ : récupère uniquement les exécutions pour l'identifiant de script donné.
- Renvoie une collection de détails sur les exécutions de tâches au format JSON. Pour de plus amples informations, veuillez consulter Structure du message Job Execution Details.

Exemple de réponse :

```
[
    {
      "scriptId": "INTCALC",
      "caller": "127.0.0.1",
      "identifier": "d43cbf46-4255-4ce2-aac2-79137573a8b4",
      "startTime": "06-12-2023 16:26:15",
      "endTime": "06-12-2023 16:26:15",
      "status": "DONE",
      "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
 \"CBACT04C\", \"status\": \"Error\" }",
      "executionMode": "ASYNCHRONOUS"
    }
  ]
```

Récupération des détails d'exécution d'une tâche

- Méthode prise en charge : GET
- Trajectoire : `/getjobexecutioninfo/{jobexecutionid:.+}`
- Arguments :
  - jobexecutionid (obligatoire) : identifiant unique d'exécution de la tâche pour récupérer les détails d'exécution de la tâche correspondants.

- Renvoie : une chaîne JSON représentant les détails d'exécution d'une seule tâche (voirStructure du message Job Execution Details) ou une réponse vide si aucun détail d'exécution de tâche n'a pu être trouvé pour l'identifiant donné.

Liste des scripts lancés de manière asynchrone qui peuvent être supprimés

- Méthode prise en charge : GET
- Trajectoire : `/killablescripts`
- Renvoie une collection d'identifiants d'exécution de tâches lancées de manière asynchrone qui sont toujours en cours d'exécution et peuvent être supprimées de force (voir le `/kill` point de terminaison ci-dessous).

Liste des scripts lancés de manière synchrone qui peuvent être supprimés

- Méthode prise en charge : GET
- Trajectoire : `/killablesyncscripts`
- Renvoie une collection d'identifiants d'exécution de tâches qui ont été lancées de manière synchrone, sont toujours en cours d'exécution et peuvent être supprimées de force (voir le `/kill` point de terminaison ci-dessous).

Annulation de l'exécution d'une tâche donnée

- Méthode prise en charge : GET
- Trajectoire : `/kill/{identifier:.+}`
- argument : identifiant d'exécution de la tâche (obligatoire) : identifiant unique d'exécution de la tâche qui doit être supprimée de force.
- Retours : un message textuel détaillant le résultat de la tentative d'arrêt de l'exécution de la tâche ; le message contiendra l'identifiant du script, l'identifiant unique de l'exécution de la tâche ainsi que la date et l'heure auxquelles l'arrêt de l'exécution s'est produit. Si aucune exécution de tâche en cours n'a pu être trouvée pour l'identifiant donné, un message d'erreur sera renvoyé à la place.

⚠ Warning

- Le moteur d'exécution fait de son mieux pour arrêter correctement l'exécution de la tâche cible. Ainsi, la réponse du point de terminaison /kill peut mettre un certain temps à atteindre l'appelant, car le moteur d'exécution de AWS Blu Age essaiera de minimiser l'impact commercial d'une suppression de la tâche.
- L'arrêt forcé d'une exécution de travail ne doit pas être fait à la légère, car cela peut avoir des conséquences commerciales directes, notamment une perte ou une corruption de données. Il doit être réservé aux cas où l'exécution d'une tâche donnée a échoué et où les moyens de correction des données sont clairement identifiés.
- La suppression d'un emploi devrait donner lieu à de nouvelles enquêtes (analyse post mortem) afin de déterminer ce qui n'a pas fonctionné et de prendre les mesures correctives appropriées.
- Dans tous les cas, toute tentative d'arrêt d'une tâche en cours d'exécution sera consignée dans les journaux du serveur avec des messages de niveau d'avertissement.

Répertorier les points de contrôle existants pour la redémarrabilité

La capacité de redémarrage des tâches repose sur la capacité des scripts à enregistrer des points de contrôle dans le `CheckpointRegistry` afin de suivre la progression de l'exécution des tâches. Si l'exécution d'une tâche ne se termine pas correctement et que des points de contrôle de redémarrage ont été enregistrés, il suffit de redémarrer l'exécution de la tâche à partir du dernier point de contrôle enregistré connu (sans avoir à exécuter les étapes situées au-dessus du point de contrôle).

- Méthode prise en charge : GET
- Trajectoire : `/restarts`
- Renvoie la liste des points de redémarrage existants, qui peuvent être utilisés pour redémarrer une tâche dont l'exécution ne s'est pas terminée correctement, sous forme de page html. Si aucun point de contrôle n'a été enregistré par aucun script, le contenu de la page sera « Aucun point de contrôle enregistré ».

Redémarrer une tâche (de manière synchrone)

- Méthode prise en charge : GET
- Trajectoire : `/restart/{hashcode}`

- Arguments : hashcode (entier - obligatoire) : redémarrez l'exécution d'une tâche précédemment abandonnée, en utilisant le hashcode fourni comme valeur de point de contrôle (voir le point de `/restarts` terminaison ci-dessus pour savoir comment récupérer une valeur de point de contrôle valide).
- Retours : voir la description des `script` retours ci-dessus.

Redémarrer une tâche (de manière asynchrone)

- Méthode prise en charge : GET
- Trajectoire : `/triggerrestart/{hashcode}`
- Arguments : hashcode (entier - obligatoire) : redémarrez l'exécution d'une tâche précédemment abandonnée, en utilisant le hashcode fourni comme valeur de point de contrôle (voir le point de `/restarts` terminaison ci-dessus pour savoir comment récupérer une valeur de point de contrôle valide).
- Retours : voir la description des `triggerscript` retours ci-dessus.

Définition de la limite de threads pour les exécutions de tâches asynchrones

L'exécution asynchrone de la tâche repose sur un pool de threads dédié dans la JVM. Ce pool a une limite fixe en ce qui concerne le nombre de threads disponibles. L'utilisateur a la possibilité d'ajuster la limite en fonction des capacités de l'hôte (nombre de processeurs, mémoire disponible, etc...). Par défaut, la limite de threads est fixée à 5 threads.

- Méthode prise en charge : GET
- Trajectoire : `/settriggerthreadlimit/{threadlimit:.+}`
- Argument (entier) : nouvelle limite de thread à appliquer. Doit être un entier strictement positif.
- Renvoie un message (`String`) indiquant la nouvelle limite de thread et la précédente, ou un message d'erreur si la valeur de limite de thread fournie n'est pas valide (il ne s'agit pas d'un entier strictement positif).

Exemple de réponse :

```
Set thread limit for Script Tower Control to 10 (previous value was 5)
```

Le comptage des tâches en cours a déclenché des exécutions de tâches

- Méthode prise en charge : GET
- Trajectoire : `/countrunningtriggeredscripts`
- Renvoie un message indiquant le nombre de tâches en cours lancées de manière asynchrone et la limite de threads (c'est-à-dire le nombre maximum de tâches déclenchées pouvant être exécutées simultanément).

Exemple de réponse :

```
0 triggered script(s) running (limit =10)
```

> **Note**
>
> Cela peut être utilisé pour vérifier, avant de lancer une tâche, si la limite de threads n'a pas été atteinte (ce qui empêcherait le lancement de la tâche).

Purger les informations relatives aux exécutions de tâches

Les informations relatives à l'exécution des tâches restent dans la mémoire du serveur tant que celui-ci est actif. Il peut être pratique de purger les informations les plus anciennes de la mémoire, car elles ne sont plus pertinentes ; c'est le but de ce point de terminaison.

- Méthode prise en charge : GET
- Trajectoire : `/purgejobinformation/{age:.+}`
- Arguments : une valeur entière strictement positive représentant l'âge en heures des informations à purger.
- Renvoie un message contenant les informations suivantes :
  - Nom du fichier de purge dans lequel les informations d'exécution des tâches purgées sont stockées à des fins d'archivage.
  - Nombre d'informations d'exécution de tâches purgées.
  - Nombre d'informations d'exécution de tâches restantes dans le mémo

Points de terminaison des métriques

JVM

Ce point de terminaison renvoie les métriques disponibles relatives à la JVM.

- Méthode prise en charge : GET
- Trajectoire : `/metrics/jvm`
- Arguments : aucun
- Renvoie un message contenant les informations suivantes :
  - threadActiveCount: nombre de threads actifs.
  - jvmMemoryUsed: mémoire utilisée activement par la machine virtuelle Java.
  - jvmMemoryMax: Mémoire maximale autorisée pour la machine virtuelle Java.
  - jvmMemoryFree: la mémoire disponible n'est pas actuellement utilisée par la machine virtuelle Java.

Session

Ce point de terminaison renvoie des métriques relatives aux sessions HTTP actuellement ouvertes.

- Méthode prise en charge : GET
- Trajectoire : `/metrics/session`
- Arguments : aucun
- Renvoie un message contenant les informations suivantes :
  - SessionCount : nombre de sessions utilisateur actives actuellement gérées par le serveur.

Par lots

- Méthode prise en charge : GET
- Trajectoire : `/metrics/batch`
- Arguments :
  - StartTimestamp (facultatif, numéro) : horodatage de début pour le filtrage des données.
  - EndTimestamp (facultatif, numéro) : horodatage de fin pour le filtrage des données.
  - page (facultatif, numéro) : numéro de page pour la pagination.
  - PageSize (facultatif, nombre) : nombre d'éléments par page en pagination.

- Renvoie un message contenant les informations suivantes :
  - contenu : liste des mesures d'exécution par lots.
  - PageNumber : numéro de page actuel dans la pagination.
  - PageSize : nombre d'éléments affichés par page.
  - TotalPages : nombre total de pages disponibles.
  - numberOfElements: nombre d'éléments sur la page en cours.
  - last : drapeau booléen pour la dernière page.
  - premier : drapeau booléen pour la première page.

Transaction

- Méthode prise en charge : GET
- Trajectoire : `/metrics/transaction`
- Arguments :
  - StartTimestamp (facultatif, numéro) : horodatage de début pour le filtrage des données.
  - EndTimestamp (facultatif, numéro) : horodatage de fin pour le filtrage des données.
  - page (facultatif, numéro) : numéro de page pour la pagination.
  - PageSize (facultatif, nombre) : nombre d'éléments par page en pagination.
- Renvoie un message contenant les informations suivantes :
  - contenu : liste des mesures d'exécution des transactions.
  - PageNumber : numéro de page actuel dans la pagination.
  - PageSize : nombre d'éléments affichés par page.
  - TotalPages : nombre total de pages disponibles.
  - numberOfElements: nombre d'éléments sur la page en cours.
  - last : drapeau booléen pour la dernière page.
  - premier : drapeau booléen pour la première page.

Autres points de terminaison

Utilisez ces points de terminaison pour répertorier les programmes ou services enregistrés, découvrir leur état de santé et gérer les transactions JICS.

Rubriques

Répertorier les programmes enregistrés

- Méthode prise en charge : GET
- Trajectoire : `/programs`
- Renvoie la liste des programmes enregistrés, sous forme de page html. Chaque programme est désigné par son identifiant principal. Les anciens programmes modernisés et les programmes utilitaires (IDCAMS, IEBGENER, etc...) sont renvoyés dans la liste. Notez que les programmes utilitaires disponibles dépendent des applications Web utilitaires déployées sur votre serveur Tomcat. Par exemple, les programmes de support de l'utilitaire z/OS peuvent ne pas être disponibles pour les actifs iSeries modernisés, car ils ne sont pas pertinents.

Répertorier les services enregistrés

- Méthode prise en charge : GET
- Trajectoire : `/services`
- Renvoie la liste des services d'exécution enregistrés, sous forme de page html. Les services fournis sont fournis par le moteur d'exécution AWS Blu Age sous forme d'utilitaires, qui peuvent être utilisés par exemple dans des scripts groovy. Les services de chargement Blusam (pour créer des ensembles de données Blusam à partir d'anciens ensembles de données) entrent dans cette catégorie.

Exemple de réponse :

```
<p>BluesamESDSFileLoader</p><p>BluesamKSDSFileLoader</p><p>BluesamRRDSFileLoader</p>
```

État de santé

- Méthode prise en charge : GET

- Trajectoire : `/`
- Renvoie un message simple, indiquant que l'application gapwalk est opérationnelle () `Jics application is running.`

Liste des transactions JICS disponibles

- Méthode prise en charge : GET
- Trajectoire : `/transactions`
- Renvoie une page HTML répertoriant toutes les transactions JICS disponibles. Cela n'a de sens que pour les environnements dotés d'éléments JICS (modernisation des éléments CICS existants).

Exemple de réponse :

```
<p>INQ1</p><p>MENU</p><p>MNT2</p><p>ORD1</p><p>PRNT</p>
```

Lancer une transaction JICS

- Méthodes prises en charge : GET, POST
- Trajectoire : `/jicstransrunner/{jtrans:.+}`
- arguments :
  - Identifiant de transaction JICS (chaîne, obligatoire) : identifiant de la transaction JICS à lancer (8 caractères au maximum)
  - obligatoire : données d'entrée supplémentaires à transmettre à la transaction, sous forme de carte<String, Object>. Le contenu de cette carte sera utilisé pour alimenter la COMMAREA qui sera consommée par la transaction JICS. La carte peut être vide si aucune donnée n'est requise pour exécuter la transaction.
  - facultatif : entrées d'en-têtes HTTP, pour personnaliser l'environnement d'exécution pour la transaction donnée. Les clés d'en-tête suivantes sont prises en charge :
    - `jics-channel`: nom du CANAL JICS à utiliser par le programme qui sera lancé lors du lancement de cette transaction.
    - `jics-container`: nom du CONTENEUR JICS à utiliser pour le lancement de cette transaction JICS.

  - `jics-startcode`: le STARTCODE (chaîne de 2 caractères maximum) à utiliser au début de la transaction JICS. Voir STARTCODE pour les valeurs possibles (parcourez la page en bas de la page).
  - `jicxa-xid`: Le XID (structure XID de l'identifiant de transaction X/Open) d'une « transaction globale » (XA), initiée par l'appelant, à laquelle participera le lancement actuel de la transaction JICS.
- Renvoie : une sérialisation `com.netfective.bluage.gapwalk.rt.shared.web.TransactionResultBean` JSON, représentant le résultat du lancement de la transaction JICS.

Pour plus d'informations sur les détails de la structure, consultezStructure des résultats du lancement de la transaction.

Lancer une transaction JICS (alternative)

- méthodes prises en charge : GET, POST
- chemin : `/jicstransaction/{jtrans:.+}`
- arguments :

  Identifiant de transaction JICS (chaîne, obligatoire)

  identifiant de la transaction JICS à lancer (8 caractères au maximum)

  obligatoire : données d'entrée supplémentaires à transmettre à la transaction, sous forme de carte<String, Object>

  Le contenu de cette carte sera utilisé pour alimenter la COMMAREA qui sera consommée par la transaction JICS. La carte peut être vide si aucune donnée n'est requise pour exécuter la transaction.

  facultatif : entrées d'en-têtes HTTP, pour personnaliser l'environnement d'exécution pour la transaction donnée.

  Les clés d'en-tête suivantes sont prises en charge :

  - `jics-channel`: nom du CANAL JICS à utiliser par le programme qui sera lancé lors du lancement de cette transaction.
  - `jics-container`: nom du CONTENEUR JICS à utiliser pour le lancement de cette transaction JICS.

    - `jics-startcode`: le STARTCODE (chaîne de 2 caractères maximum) à utiliser au début de la transaction JICS. Pour les valeurs possibles, voir STARTCODE (parcourez la page en bas de la page).
    - `jicxa-xid`: Le XID (structure XID de l'identifiant de transaction X/Open) d'une « transaction globale » (XA), initiée par l'appelant, à laquelle participera le lancement actuel de la transaction JICS.
- retours : une sérialisation `com.netfective.bluage.gapwalk.rt.shared.web.RecordHolderBean` JSON, représentant le résultat du lancement de la transaction JICS. Les détails de la structure se trouvent dansStructure des résultats de l'enregistrement du lancement de la transaction.

Points de terminaison liés aux files d'attente de tâches

Les files d'attente constituent le support AWS Blu Age pour le mécanisme de soumission de tâches AS400. Les files d'attente de tâches sont utilisées dans l'AS400 pour exécuter des tâches sur des pools de threads spécifiques. Une file de tâches est définie par un nom et un nombre maximum de threads correspondant au nombre maximum de programmes pouvant être exécutés simultanément sur cette file d'attente. Si le nombre de tâches soumises dans la file d'attente est supérieur au nombre maximal de threads, les tâches attendront qu'un fil soit disponible.

Pour obtenir la liste exhaustive du statut d'une tâche dans une file d'attente, consultezÉtat possible d'une tâche dans une file d'attente.

Les opérations sur les files d'attente de tâches sont gérées via les points de terminaison dédiés suivants. Vous pouvez appeler ces opérations depuis l'URL de l'application Gapwalk avec l'URL racine suivante :. `http://server:port/gapwalk-application/jobqueue`

Rubriques

- Liste des files d'attente disponibles
- Démarrer ou redémarrer une file d'attente de tâches
- Soumettre une offre d'emploi pour lancement
- Répertorier tous les travaux planifiés
- Répertoriez tous les emplois « en attente »
- Liste de tous les emplois actifs
- Répertoriez tous les jobs en attente de lancement

- Libérez toutes les tâches « en attente »
- Libérer toutes les tâches « en attente » pour un nom de tâche donné
- Libérer une tâche donnée pour un numéro de tâche

Liste des files d'attente disponibles

- Méthode prise en charge : GET
- Trajectoire : `list-queues`
- Renvoie la liste des files d'attente disponibles ainsi que leur statut, sous forme de liste JSON de valeurs-clés.

Exemple de réponse :

```
{"Default":"STAND_BY","queue1":"STARTED","queue2":"STARTED"}
```

Les statuts possibles d'une file d'attente de tâches sont les suivants :

EN ATTENTE

la file d'attente des tâches attend d'être démarrée.

DÉMARRÉE

la file d'attente des tâches est opérationnelle.

UNKNOWN

l'état de la file d'attente des tâches ne peut pas être déterminé.

Démarrer ou redémarrer une file d'attente de tâches

- Méthode prise en charge : POST
- Trajectoire : `/restart/{name}`
- Argument : nom de la file d'attente à démarrer/redémarrer, sous forme de chaîne - obligatoire.
- Le point de terminaison ne renvoie rien mais s'appuie plutôt sur le statut HTTP pour indiquer le résultat de l'opération de démarrage/redémarrage :

HTTP 200

l'opération de démarrage/redémarrage s'est bien déroulée : la file de tâches donnée est maintenant DÉMARRÉE.

HTTP 404

la file d'attente des tâches n'existe pas.

HTTP 503

une exception s'est produite lors de la tentative de démarrage/redémarrage (les journaux du serveur doivent être inspectés pour déterminer ce qui s'est mal passé).

Soumettre une offre d'emploi pour lancement

- Méthode prise en charge : POST
- Argument : obligatoire en tant que corps de requête, une sérialisation JSON d'un `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objet. Pour de plus amples informations, veuillez consulter Soumettre une offre d'emploi.
- Renvoie : un fichier JSON contenant l'original `SubmitJobMessage` et un journal indiquant si le travail a été soumis ou non.

Répertorier tous les travaux planifiés

- Méthode prise en charge : GET
- Trajectoire : `list-jobs`
- Renvoie : une liste de toutes les tâches planifiées, sous forme de chaîne JSON. Pour un exemple de réponse, voirRéponse à la liste des tâches planifiées.

Répertoriez tous les emplois « en attente »

- Méthode prise en charge : GET
- Trajectoire : `list-jobs-hold`
- Renvoie : une liste de toutes les tâches planifiées, sous forme de chaîne JSON. Pour un exemple de réponse, voirListe des réponses « en attente » concernant les offres d'emploi.

Liste de tous les emplois actifs

- Méthode prise en charge : GET
- Trajectoire : `list-jobs-active`
- Renvoie : une liste de toutes les tâches ayant le statut ACTIVE, sous forme de chaîne JSON. La structure de la réponse est similaire à celle deRéponse à la liste des tâches planifiées.

Répertoriez tous les jobs en attente de lancement

- Méthode prise en charge : GET
- Trajectoire : `list-jobs-waiting`
- Renvoie : une liste de toutes les tâches ayant le statut EXECUTION_WAIT (tâches qui attendent qu'un thread soit disponible pour le lancement), sous forme de chaîne JSON. La structure de la réponse est similaire à celle dethe section called "Réponse à la liste des tâches planifiées".

Libérez toutes les tâches « en attente »

- Méthode prise en charge : POST
- Trajectoire : `release-all`
- Retours : un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :
  - HTTP 200 et un message « Toutes les offres d'emploi ont été publiées avec succès ! » si tous les jobs ont été publiés avec succès.
  - HTTP 503 et un message « Jobs not released. Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Libérer toutes les tâches « en attente » pour un nom de tâche donné

<job name, job number>Pour un nom de tâche donné, plusieurs tâches peuvent être soumises, avec des numéros de tâche différents (l'unicité d'une exécution de tâche est garantie par deux). Le point de terminaison tentera de publier toutes les soumissions de tâches portant le nom de tâche donné, qui sont « en attente ».

- Méthode prise en charge : POST
- Trajectoire : `/release/{name}`

- Arguments : le nom de la tâche à rechercher, sous forme de chaîne. Obligatoire.
- Retours : un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :
  - HTTP 200 et un message « Jobs in group <name>(<number of released jobs>) a été publié avec succès ! » les offres d'emploi ont été publiées avec succès.
  - HTTP 503 et un message « Les tâches du groupe <name>n'ont pas été publiées. Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Libérer une tâche donnée pour un numéro de tâche

Le point de terminaison tentera de publier la soumission de tâche unique qui est « en attente », pour le couple donné <job name, job number>.

- Méthode prise en charge : POST
- Trajectoire : `/release/{name}/{number}`
- Arguments :

  name

  le nom de la tâche à rechercher, sous forme de chaîne. Obligatoire.

  nombre

  le numéro de tâche à rechercher, sous forme de nombre entier. Obligatoire.

  renvoie

  un message indiquant le résultat de l'opération de tentative de publication. Voici deux cas possibles :
  - HTTP 200 et un message « Job <name/number> released with success ! » si le job a été publié avec succès.
  - HTTP 503 et un message « Job <name/number> not released. Une erreur inconnue s'est produite. Consultez le journal pour plus de détails » en cas de problème lors de la tentative de publication.

Points de terminaison REST de la console d'applications Blusam

La console d'application Blusam est une API conçue pour simplifier la gestion des ensembles de données VSAM modernisés. Les points de terminaison de l'application Web Blusam utilisent le chemin racine. `/bac`

Rubriques

Points de terminaison liés aux ensembles de données

Utilisez les points de terminaison suivants pour créer ou gérer un ensemble de données spécifique.

Rubriques

Création d'un ensemble de données

La création d'un point de terminaison d'un ensemble de données permet de créer une définition d'ensemble de données et nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `` `/api/services/rest/bluesamservice/createDataSet` ``

- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.

  type

  (obligatoire, chaîne) : le type de jeu de données. Les valeurs possibles sont les suivantes :`ESDS`,`KSDS`,`RRDS`.

  Taille de l'enregistrement

  (facultatif, chaîne) : taille maximale de chaque enregistrement de l'ensemble de données.

  Longueur fixe

  (facultatif, booléen) : indique si la longueur des enregistrements est fixe.

  compression

  (facultatif, booléen) : indique si le jeu de données est compressé.

  Activer le cache

  (facultatif, booléen) : indique si la mise en cache est activée pour l'ensemble de données.

  Clés alternatives

  (facultatif, liste des clés) :

  - offset (obligatoire, nombre)
  - longueur (obligatoire, nombre)
  - nom (obligatoire, numéro)
- Renvoie un fichier json représentant le nouvel ensemble de données créé.

Demande d'échantillon :

```
POST /api/services/rest/bluesamservice/createDataSet
{
  "name": "DATASET",
  "checked": false,
  "records": [],
  "primaryKey": {
    "name": "PK"
  },
```

```
  "alternativeKeys": [
    {
      "offset": 10,
      "length": 10,
      "name": "ALTK_0"
    }
  ],
  "type": "ESDS",
  "recordSize": 10,
  "compression": true,
  "cacheEnable": true
}
```

Exemple de réponse :

```
{
    "dataSet": {
      "name": "DATASET",
      "checked": false,
      "nbRecords": 0,
      "keyLength": -1,
      "recordSize": 10,
      "compression": false,
      "fixLength": true,
      "type": "ESDS",
      "cacheEnable": false,
      "cacheWarmup": false,
      "cacheEviction": "100ms",
      "creationDate": 1686744961234,
      "modificationDate": 1686744961234,
      "records": [],
      "primaryKey": {
        "name": "PK",
        "offset": null,
        "length": null,
        "columns": null,
        "unique": true
      },
      "alternativeKeys": [
        {
          "offset": 10,
          "length": 10,
          "name": "ALTK_0"
```

```
        }
      ],
      "readLimit": 0,
      "readEncoding": null,
      "initCharacter": null,
      "defaultCharacter": null,
      "blankCharacter": null,
      "strictZoned": null,
      "decimalSeparator": null,
      "currencySign": null,
      "pictureCurrencySign": null
    },
    "message": null,
    "result": true
  }
```

Charger un fichier

Ce point de terminaison permet de télécharger des fichiers sur le serveur. Le fichier est stocké dans un dossier temporaire qui correspond à chaque utilisateur spécifique. Ce point de terminaison doit être utilisé chaque fois qu'un fichier doit être téléchargé. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/upload`
- Arguments :

  dans le fichier

  (obligatoire, multipart/form-data) : Le fichier à télécharger.
- Renvoie une valeur booléenne reflétant le statut du téléchargement

Charger un ensemble de données

Une fois la définition de l'ensemble de données créée à l'aide du point de terminaison de création d'ensemble de données décrit précédemment, vous pouvez charger les enregistrements associés au fichier téléchargé dans un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/loadDataSet`
- Arguments :

name

(obligatoire, chaîne) : le nom de l'ensemble de données.

- Renvoie l'état de la demande et le jeu de données chargé.

Charger un ensemble de données depuis un compartiment Amazon S3

Charge un ensemble de données à l'aide d'un fichier listcat depuis un compartiment Amazon S3.

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/loadDataSetFromS3`
- Arguments :

  Emplacement des fichiers ListCat 3

  (obligatoire, chaîne) : l'emplacement du fichier listcat sur Amazon S3.

  Emplacement des fichiers du jeu de données 3

  (obligatoire, chaîne) : l'emplacement du fichier d'ensemble de données sur Amazon S3.

  region

  (obligatoire, chaîne) : l'Amazon S3 Région AWS où les fichiers sont stockés.
- Renvoie le jeu de données nouvellement créé

Demande d'échantillon :

```
/BAC/api/services/rest/bluesamservice/loadDataSetFromS3?region=us-
east-1&listcatFileS3Location=s3://bucket-name/listcat.json&datasetFileS3Location=s3://
bucket-name/dataset.DAT
```

Exporter un ensemble de données vers un compartiment Amazon S3

Exporte un ensemble de données vers le compartiment Amazon S3 spécifié.

- Méthodes prises en charge : GET
- Trajectoire : /api/services/rest/bluesamservice/exportDataSetToS3
- Arguments :

s3Location

(obligatoire, chaîne) : l'emplacement Amazon S3 vers lequel exporter l'ensemble de données.

datasetName

(obligatoire, chaîne) : le nom du jeu de données à exporter.

region

(obligatoire, chaîne) : Région AWS celui du compartiment Amazon S3.

- Renvoie l'ensemble de données exporté

Demande d'échantillon :

```
/BAC/api/services/rest/bluesamservice/exportDataSetToS3?region=eu-
west-1&s3Location=s3://bucket-name/dump&datasetName=dataset
```

Effacer un ensemble de données

Efface tous les enregistrements d'un ensemble de données. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/clearDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom du jeu de données à effacer.
- Renvoie le statut de la demande.

Supprimer un ensemble de données

Supprime la définition et les enregistrements de l'ensemble de données. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/deleteDataSet`
- Arguments :

name

(obligatoire, chaîne) : le nom du jeu de données à supprimer.

- Renvoie le statut de la demande et le jeu de données supprimé.

Compter les enregistrements d'ensembles de données

Ce point de terminaison renvoie le nombre d'enregistrements associés à un ensemble de données. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/countRecords`
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Retours : le nombre d'enregistrements

Points de terminaison associés aux ensembles de données en masse

Utilisez les points de terminaison suivants pour créer ou gérer plusieurs ensembles de données à la fois.

Rubriques

- Exporter des ensembles de données
- Création de plusieurs ensembles de données
- Répertorier tous les ensembles de données
- Liste directe de tous les ensembles de données
- Supprimer tous les ensembles de données
- Obtenir les définitions des ensembles de données à partir du fichier listcat
- Obtenir les définitions des ensembles de données à partir du fichier list cat téléchargé
- Charger listcat depuis un fichier json

Exporter des ensembles de données

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/exportDataSet`
- Arguments :

  name

  (obligatoire, chaîne) : le nom du jeu de données à supprimer.

  datasetOutputFile

  (facultatif, chaîne) : le chemin où stocker le jeu de données exporté sur le serveur

  rdw

  (facultatif, booléen) : les champs RDW doivent-ils être exportés ?
- renvoie le statut de la demande et le fichier contenant le jeu de données exporté.

Création de plusieurs ensembles de données

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/createAllDataSets`
- Arguments :
  - Liste des ensembles de données

    name

    (obligatoire, chaîne) : le nom de l'ensemble de données.

    type

    (obligatoire, chaîne) : le type de jeu de données. Les valeurs possibles sont les suivantes :`ESDS`,`KSDS`,`RRDS`.

    Taille de l'enregistrement

    (facultatif, chaîne) : taille maximale de chaque enregistrement de l'ensemble de données.

    Longueur fixe

    (facultatif, booléen) : indique si la longueur des enregistrements est fixe.

compression

(facultatif, booléen) : indique si le jeu de données est compressé.

Activer le cache

(facultatif, booléen) : indique si la mise en cache est activée pour l'ensemble de données.

- Retours : le statut de la demande et le nouvel ensemble de données créé.

Répertorier tous les ensembles de données

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/listDataSet`
- Arguments : Aucun
- Retours : le statut de la demande et la liste des ensembles de données.

Liste directe de tous les ensembles de données

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/directListDataSet`
- Arguments : Aucun
- Retours : le statut de la demande et la liste des ensembles de données.

Supprimer tous les ensembles de données

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/removeAll`
- Arguments : Aucun
- Renvoie : un booléen représentant le statut de la demande.

Obtenir les définitions des ensembles de données à partir du fichier listcat

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/getDataSetsDefinitionFromListcat`

- Arguments :

  paramFilePath

  (obligatoire, chaîne) : chemin d'accès au fichier listcat.
- Retours : liste d'ensembles de données

Obtenir les définitions des ensembles de données à partir du fichier list cat téléchargé

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/bluesamservice/getDataSetsDefinitionFromUploadedListcat`
- Arguments : Aucun
- Retours : liste d'ensembles de données

Charger listcat depuis un fichier json

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/loadListcatFromJsonFile`
- Arguments :

  filePath

  (obligatoire, chaîne) : chemin d'accès au fichier listcat.
- Retours : liste d'ensembles de données

Enregistrements

Utilisez les points de terminaison suivants pour créer ou gérer des enregistrements au sein d'un ensemble de données.

Rubriques

- Créer un enregistrement
- Lire un ensemble de données
- Supprimer un enregistrement
- Mettre à jour un enregistrement

- Enregistrer un enregistrement

Créer un enregistrement

Ce point de terminaison permet de créer un nouvel enregistrement. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/createRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  masque

  (obligatoire, masque) : l'objet du masque.
- Renvoie le statut de la demande et l'enregistrement créé.

Lire un ensemble de données

Ce point de terminaison permet de lire un ensemble de données.

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/crud/readDataSet`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données.
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Supprimer un enregistrement

Ce point de terminaison permet de supprimer un enregistrement d'un ensemble de données. Elle nécessite une authentification.

- Méthodes prises en charge : DELETE

- Trajectoire : `/api/services/rest/crud/deleteRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, Enregistrement) : l'enregistrement à supprimer
- Renvoie le statut de la suppression.

Mettre à jour un enregistrement

Ce point de terminaison permet de mettre à jour un enregistrement associé à un ensemble de données. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/updateRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  record

  (obligatoire, enregistrement) : l'enregistrement à mettre à jour
- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Enregistrer un enregistrement

Ce point de terminaison permet d'enregistrer un enregistrement dans un ensemble de données à l'aide d'un masque. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/saveRecord`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

record

(obligatoire, enregistrement) : l'enregistrement à enregistrer

- Renvoie le statut de la demande et le jeu de données contenant les enregistrements.

Masques

Utilisez les points de terminaison suivants pour charger ou appliquer des masques à un ensemble de données.

Rubriques

- Masques de chargement
- Appliquer un masque
- Appliquer un filtre de masque

Masques de chargement

Ce point de terminaison permet de récupérer tous les masques associés à un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/loadMasks`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données
- Renvoie le statut de la demande et la liste des masques.

Appliquer un masque

Ce point de terminaison permet d'appliquer un masque à un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/applyMask`
- Arguments :

dataset

(obligatoire, DataSet) : l'objet du jeu de données

masque

(obligatoire, masque) : l'objet du jeu de données

- Renvoie l'état de la demande et le jeu de données avec le masque appliqué.

Appliquer un filtre de masque

Ce point de terminaison permet d'appliquer un masque et un filtre à un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/rest/crud/applyMaskFilter`
- Arguments :

  dataset

  (obligatoire, DataSet) : l'objet du jeu de données

  masque

  (obligatoire, masque) : l'objet du jeu de données
- Renvoie le statut de la demande et le jeu de données avec le masque et le filtre appliqués.

Autre

Utilisez les points de terminaison suivants pour gérer le cache d'un ensemble de données ou vérifier les caractéristiques d'un ensemble de données

Rubriques

- Vérifiez le cache de préchauffage
- Vérifier le cache activé
- Activer le cache
- Vérifiez le cache RAM alloué
- Vérifier la persistance

- Vérifiez les types d'ensembles de données pris en charge
- Vérifier l'état du serveur

Vérifiez le cache de préchauffage

Vérifie si le cache de préchauffage est activé pour un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : POST
- Trajectoire : `` `/api/services/rest/bluesamservice/warmupCache` ``
- Arguments :

  name

  (obligatoire, chaîne) : le nom de l'ensemble de données.
- Renvoie : vrai si le cache de préchauffage est activé et faux dans le cas contraire.

Vérifier le cache activé

Vérifie si le cache est activé pour un ensemble de données spécifique. Elle nécessite une authentification.

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/isEnableCache`
- Arguments : Aucun
- Renvoie vrai si la mise en cache est activée.

Activer le cache

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/enableDisableCache/{enable}`
- Arguments :

  activer

  (obligatoire, booléen) : s'il est défini sur true, il activera la mise en cache.
- Ne renvoie aucun

Vérifiez le cache RAM alloué

Ce point de terminaison permet de récupérer la mémoire cache RAM allouée. Elle nécessite une authentification.

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/allocatedRamCache`
- Arguments : Aucun
- Renvoie : la taille de la mémoire sous forme de chaîne

Vérifier la persistance

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/persistence`
- Arguments : Aucun
- Retours : la persistance utilisée sous forme de chaîne

Vérifiez les types d'ensembles de données pris en charge

- Trajectoire : `/api/services/rest/bluesamservice/getDataSetTypes`
- Arguments : Aucun
- Renvoie : la liste des types d'ensembles de données pris en charge sous forme de liste de chaînes.

Vérifier l'état du serveur

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/rest/bluesamservice/serverIsUp`
- Arguments : Aucun
- Retours : Aucun

Users

Utilisez les points de terminaison suivants pour gérer les interactions des utilisateurs.

Rubriques

- Connexion
- Vérifier le compte utilisateur
- Connectez-vous
- Répertorier tous les utilisateurs
- Déconnexion

Connexion

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/security/servicelogin/login`
- Arguments :

  nom d'utilisateur

  (obligatoire, chaîne)

  mot de passe

  (obligatoire, chaîne)
- Renvoie le nom d'utilisateur et les rôles de l'utilisateur connecté

Exemple de réponse

```
{"login":"some-user","roles":[{"id":0,"roleName":"ROLE_ADMIN"}]}
```

Vérifier le compte utilisateur

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/security/servicelogin/hasAccount`
- Arguments : Aucun
- Renvoie : vrai si l'utilisateur est déjà connecté

Connectez-vous

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/security/servicelogin/recorduser`

- Arguments : Aucun
- Renvoie : vrai si l'utilisateur est déjà connecté

Répertorier tous les utilisateurs

- Méthodes prises en charge : GET
- Trajectoire : `/api/services/security/servicelogin/listusers`
- Arguments : Aucun
- Retours : liste de tous les utilisateurs

Déconnexion

- Méthodes prises en charge : POST
- Trajectoire : `/api/services/security/servicelogout/logout`
- Arguments : Aucun
- Renvoie : vrai si l'utilisateur s'est déconnecté avec succès.

Console d'application JICS

Le composant JICS est le support AWS Blu Age pour la modernisation des ressources CICS existantes. L'application Web JICS Application Console est dédiée à l'administration des ressources JICS. Les points de terminaison suivants permettent d'effectuer les tâches d'administration sans avoir à interagir avec l'interface utilisateur JAC. Chaque fois qu'un point de terminaison nécessite une authentification, la demande doit inclure les détails de l'authentification (nom d'utilisateur/mot de passe généralement, comme l'exige l'authentification de base). Les points de terminaison de l'application Web JICS Application Console utilisent le chemin racine. `/jac/`

Rubriques

Gestion des ressources JICS

Tous les points de terminaison suivants sont liés à la gestion des ressources du JICS, ce qui permet aux administrateurs du JICS de gérer les ressources au quotidien.

Rubriques

Liste des listes et des groupes JICS

LIST et GROUPS sont les principales ressources de conteneur propriétaires au sein du composant JICS. Toutes les ressources JICS doivent appartenir à un GROUPE. Les groupes peuvent appartenir à des LISTES, mais cela n'est pas obligatoire. Les LISTS peuvent même ne pas exister dans un environnement JICS donné, mais la plupart du temps, les LISTS sont là pour donner une couche supplémentaire d'organisation aux ressources. Pour plus d'informations sur l'organisation des ressources du CICS, consultez les ressources du CICS.

- Méthode prise en charge : GET
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listJicsListsAndGroups`
- Renvoie : une liste d' JicsContainer objets sérialisés, à la fois LISTS et GROUPS, au format JSON.

Exemple de réponse :

```
[
    {
      "name": "Resources",
      "children": [
        {
          "jacType": "JACList",
          "name": "MURACHS",
          "isActive": true,
          "children": [
            {
              "jacType": "JACGroup",
              "name": "MURACHS",
              "isActive": true,
              "children": []
            }
          ]
        },
        {
          "jacType": "JACGroup",
          "name": "TEST",
          "isActive": true,
          "children": []
        }
      ],
      "isExpanded": true
```

```
    }
  ]
```

Liste JICS GROUPS

- Méthode prise en charge : GET
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listJicsGroups`
- Renvoie : une liste d' JicsContainer objets sérialisés (GROUPS) au format JSON. Les GROUPES sont renvoyés sans leurs propres informations LIST.

Exemple de réponse :

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
      "isActive": true,
      "children": []
    },
    {
      "jacType": "JACGroup",
      "name": "TEST",
      "isActive": true,
      "children": []
    }
  ]
```

Lister les groupes JICS pour une liste donnée

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listGroupsForList`
- Arguments : une charge utile JSON, représentant la LISTE JICS dont nous recherchons les GROUPES. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet.

  Demande d'échantillon :

```
{
    "jacType":"JACList",
    "name":"MURACHS",
    "isActive":true
  }
```

- Renvoie : une liste d' JicsContainer objets sérialisés (GROUPS) au format JSON, attachés à la LIST donnée. Les GROUPES sont renvoyés sans leurs propres informations LIST.

  Exemple de réponse :

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
      "isActive": true,
      "children": []
    }
  ]
```

LISTER les ressources JICS pour un GROUPE donné

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listResourcesForGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS dont nous recherchons les ressources. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet. Il n'est pas nécessaire de spécifier tous les champs pour le GROUPE, mais le nom est obligatoire.

  Demande d'échantillon :

```
{
    "jacType":"JACGroup",
    "name":"MURACHS",
    "isActive":true
  }
```

- Renvoie : une liste d' JicsResource objets sérialisés, appartenant au GROUPE donné. Les objets sont renvoyés sans ordre particulier et sont de types différents (PROGRAMME, TRANSACTION, FICHIER, etc...).

LISTER les ressources JICS pour un GROUPE donné (alternative en utilisant un nom)

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/listResourcesForGroupName`
- Arguments : le nom du GROUPE propriétaire des ressources recherchées.
- Renvoie : une liste d' JicsResource objets sérialisés, appartenant au GROUPE donné. Les objets sont renvoyés sans ordre particulier et sont de types différents (PROGRAMME, TRANSACTION, FICHIER, etc...)

Modification des GROUPES détenus de plusieurs LISTES

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/editGroupsList`
- Arguments : une représentation JSON d'une collection de LISTES avec des groupes enfants ;

  Demande d'échantillon :

```
[
  {
        "jacType": "JACList",
        "name": "MURACHS",
        "isActive": true,
        "children": [
          {
            "jacType": "JACGroup",
            "name": "MURACHS",
            "isActive": true,
            "children": []
          },
          {
            "jacType": "JACGroup",
            "name": "TEST",
```

```
            "isActive": true,
            "children": []
          }
        ]
  }
]
```

Avant cette édition, seul le groupe nommé « MURACHS » faisait partie de la LISTE nommée « MURACHS ». Avec cette édition, nous « ajoutons » le groupe nommé « TEST » à la LISTE nommée « MURACHS ».

- Renvoie une valeur booléenne. Si la valeur est « vraie », les modifications LISTS ont été correctement conservées dans le stockage JICS sous-jacent.

Supprimer une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteList`
- Arguments : une charge utile JSON, représentant la LISTE JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de LIST a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du GROUPE a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer une TRANSACTION

- Méthode prise en charge : POST

- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteTransaction`
- Arguments : une charge utile JSON, représentant la transaction JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TRANSACTION a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer un PROGRAMME

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteProgram`
- Arguments : une charge utile JSON, représentant le programme JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du PROGRAMME a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteFile`
- Arguments : une charge utile JSON, représentant le fichier JICS à supprimer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACFile` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression du FICHIER a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteTDQueue`
- Arguments : une charge utile JSON, représentant la JICS TDQUEUE à supprimer. Il s'agit de la sérialisation JSON d'un objet `com.netfective.bluage.jac.entities.JACTDQueue`.

- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TDQUEUE a été correctement effectuée sur le stockage JICS sous-jacent.

Supprimer un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/deleteTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à supprimer. Il s'agit de la sérialisation JSON d'un objet `com.netfective.bluage.jac.entities.JactsModel`.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la suppression de TSMODEL a été correctement effectuée sur le stockage JICS sous-jacent.

Création d'une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createList`
- Arguments : une charge utile JSON, représentant la liste JICS à créer. Il s'agit de la sérialisation JSON d'un objet `com.netfective.bluage.jac.entities.jacList`.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la LISTE a été correctement créée dans le stockage JICS sous-jacent.

> **Note**
>
> La LISTE sera toujours créée vide. L'ajout de GROUPES à la LISTE nécessitera une autre opération.

Création d'un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createGroup`

- Arguments : une charge utile JSON, représentant le GROUPE JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le GROUPE a été correctement créé dans le stockage JICS sous-jacent.

> **Note**
>
> Le GROUPE sera toujours créé vide. L'attachement de RESSOURCES au GROUPE nécessitera des opérations supplémentaires (la création de ressources les associera automatiquement à un GROUPE donné).

Considérations communes relatives à la création de RESSOURCES

Tous les points de terminaison suivants sont liés à la création de JICS RESOURCES et partagent certaines contraintes communes : dans la charge utile de la demande à envoyer au point de terminaison, le `groupName` champ doit être valorisé.

Contrainte de propriété du GROUPE :

Aucune ressource ne peut être créée sans être attachée à un groupe existant, et le point de terminaison utilise le GroupName pour récupérer le groupe auquel cette ressource sera attachée. Le `groupName` doit pointer vers le nom d'un GROUPE existant. Un message d'erreur avec HTTP STATUS 400 sera envoyé s'il ne pointe pas vers un groupe existant dans le stockage sous-jacent JICS. `groupName`

Contrainte d'unicité au sein d'un GROUPE :

Une ressource donnée portant un nom donné doit être unique au sein d'un groupe donné. La vérification de l'unicité sera effectuée par chaque point de terminaison de création de ressources. Si la charge utile donnée ne respecte pas la contrainte d'unicité, le point de terminaison enverra une réponse avec HTTP STATUS 400 (BAD REQUEST). Voir l'exemple de réponse ci-dessous.

Exemple de charge utile : nous essayons de créer la transaction « ARIT » dans le groupe « TEST », mais une transaction portant ce nom existe déjà dans ce GROUPE.

```
{
    "jacType":"JACTransaction",
    "name":"ARIT",
    "groupName":"TEST",
```

```
    "isActive":true
  }
```

Nous avons reçu la réponse d'erreur suivante :

```
{
    "timestamp": 1686759054510,
    "status": 400,
    "error": "Bad Request",
    "path": "/jac/api/services/rest/jicsservice/createTransaction"
  }
```

L'inspection des journaux des serveurs confirmera l'origine du problème :

```
2023-06-14 18:10:54 default         TRACE - o.s.w.m.HandlerMethod
        - Arguments: [java.lang.IllegalArgumentException: Transaction already
 present in the group, org.springframework.security.web.header.HeaderWriterFilter
$HeaderWriterResponse@e34f6b8]
2023-06-14 18:10:54 default         ERROR - c.n.b.j.a.WebConfig                    -
 400
java.lang.IllegalArgumentException: Transaction already present in the group
 at
 com.netfective.bluage.jac.server.services.rest.impl.JicsServiceImpl.createElement(JicsServiceI
```

Création d'une TRANSACTION

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createTransaction`
- Arguments : une charge utile JSON, représentant la TRANSACTION JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TRANSACTION a été correctement créée dans le stockage JICS sous-jacent.

Création d'un PROGRAMME

- Méthode prise en charge : POST

- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createProgram`
- Arguments : une charge utile JSON, représentant le PROGRAMME JICS à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le PROGRAMME a été correctement créé dans le stockage JICS sous-jacent.

Création d'un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createFile`
- Arguments : une charge utile JSON, représentant le FICHIER JICS à créer. Il s'agit de la sérialisation JSON d'un objet `com.netfective.bluage.jac.entities.jacFile`.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le FICHIER a été correctement créé dans le stockage JICS sous-jacent.

Créez une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createTDQueue`
- Arguments : une charge utile JSON, représentant la JICS TDQUEUE à créer. Il s'agit de la sérialisation JSON d'un objet `com.netfective.bluage.jac.entities.JACTDQueue`.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TDQUEUE a été correctement créée dans le stockage JICS sous-jacent.

Créez un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/createTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à créer. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTSModel` objet.

- Renvoie une valeur booléenne. Si la valeur est « vraie », le TSMODEL a été correctement créé dans le stockage JICS sous-jacent.

Mettre à jour une LISTE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateList`
- Arguments : une charge utile JSON, représentant la LISTE JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACList` objet. Il n'est pas nécessaire de fournir les enfants de la LIST, le mécanisme de mise à jour de la LIST n'en tiendra pas compte.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la LISTE a été correctement mise à jour dans le stockage JICS sous-jacent.

La mise à jour de l'indicateur LIST « IsActive » se propagera à tous les éléments appartenant à la LISTE, c'est-à-dire à tous les GROUPES appartenant à la LISTE et à toutes les RESSOURCES détenues par ces GROUPES. Il s'agit d'un moyen pratique de désactiver un grand nombre de ressources en une seule opération, sur plusieurs GROUPES.

Mettre à jour un GROUPE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateGroup`
- Arguments : une charge utile JSON, représentant le GROUPE JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACGroup` objet. Il n'est pas nécessaire de fournir les enfants du GROUPE, le mécanisme de mise à jour du GROUPE n'en tiendra pas compte.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le GROUPE a été correctement mis à jour dans le stockage JICS sous-jacent.

> **Note**
>
> La mise à jour de l'indicateur « IsActive » du GROUPE se propagera à tous les éléments appartenant au GROUPE, c'est-à-dire à toutes les RESSOURCES détenues par le GROUPE. Il s'agit d'un moyen pratique de désactiver un grand nombre de ressources en une seule opération au sein d'un GROUPE donné.

Considérations courantes relatives à la mise à jour

Tous les points de terminaison suivants concernent la mise à jour des RESSOURCES JICS. À l'aide de `groupName` ce champ, vous pouvez modifier le GROUPE propriétaire de n'importe quelle RESSOURCE JICS, à condition que la valeur du champ pointe vers un GROUPE existant dans le stockage JICS sous-jacent (sinon, vous recevrez une réponse BAD REQUEST (HTTP STATUS 400) de la part du point de terminaison).

Mettre à jour une TRANSACTION

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateTransaction`
- Arguments : une charge utile JSON, représentant la transaction JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTransaction` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TRANSACTION a été correctement mise à jour dans le stockage JICS sous-jacent.

Mettre à jour un PROGRAMME

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateProgram`
- Arguments : une charge utile JSON, représentant le PROGRAMME JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACProgram` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le PROGRAMME a été correctement mis à jour dans le stockage JICS sous-jacent.

Mettre à jour un FICHIER

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateFile`
- Arguments : une charge utile JSON, représentant le FICHIER JICS à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACFile` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le FICHIER a été correctement mis à jour dans le stockage JICS sous-jacent.

Mettre à jour une TDQUEUE

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateTDQueue`
- Arguments : une charge utile JSON, représentant le JICS TDQUEUE à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTDQueue` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », la TDQueue a été correctement mise à jour dans le stockage JICS sous-jacent.

Mettre à jour un TSMODEL

- Méthode prise en charge : POST
- Nécessite une authentification
- Trajectoire : `/api/services/rest/jicsservice/updateTSModel`
- Arguments : une charge utile JSON, représentant le JICS TSMODEL à mettre à jour. Il s'agit de la sérialisation JSON d'un `com.netfective.bluage.jac.entities.JACTSModel` objet.
- Renvoie une valeur booléenne. Si la valeur est « vraie », le TSMODEL a été correctement mis à jour dans le stockage JICS sous-jacent.

Points de terminaison de gestion des utilisateurs JAC

Rubriques

- Supprimer un utilisateur

- Connexion d'un utilisateur
- Tester si au moins un utilisateur existe dans le système
- Enregistrer un nouvel utilisateur
- Lister les utilisateurs
- Lister les utilisateurs
- Déconnecter l'utilisateur actuel
- État de santé du serveur Jics

Supprimer un utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et des droits d'administrateur
- Trajectoire : `/api/services/security/servicelogin/deleteuser`
- Arguments : la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet, représentant l'utilisateur à supprimer du stockage.
- Renvoie une valeur booléenne. Si la valeur est « vraie », l'utilisateur a été correctement supprimé du système JICS.

> **Note**
>
> Cette action ne peut pas être annulée, l'utilisateur supprimé ne pourra pas se reconnecter à l'application JAC. À utiliser avec précaution.

Connexion d'un utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et des droits d'administrateur
- Trajectoire : `/api/services/security/servicelogin/login`
- Renvoie la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet, représentant l'utilisateur dont les informations d'identification sont fournies dans la demande en cours. Le mot de passe est masqué dans la vue dans l'objet renvoyé. Les rôles attribués à l'utilisateur sont listés.

Exemple de réponse :

```
{
    "login": "sadmin",
    "password": null,
    "roles": [
      {
        "id": 0,
        "roleName": "ROLE_SUPER_ADMIN"
      }
    ]
  }
```

Tester si au moins un utilisateur existe dans le système

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogin/hasAccount`
- Renvoie une valeur booléenne, dont la valeur est vraie si au moins un utilisateur, autre que l'utilisateur super administrateur par défaut, a été créé dans le système JICS.

Enregistrer un nouvel utilisateur

- Méthode prise en charge : POST
- Nécessite une authentification et des droits d'administrateur
- Trajectoire : `/api/services/security/servicelogin/recorduser`
- Arguments : la sérialisation JSON d'un `com.netfective.bluage.jac.entities.SignOn` objet, représentant l'utilisateur à ajouter au stockage. Les rôles de l'utilisateur doivent être définis, sinon l'utilisateur risque de ne pas être en mesure d'utiliser la fonction JAC et les points de terminaison.

Demande d'échantillon :

```
{
    "login": "simpleuser",
    "password": "simpleuser",
    "roles": [
      {
        "id": 2,
```

```
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Seuls les rôles suivants peuvent être utilisés lors de l'enregistrement d'un nouvel utilisateur :

- ROLE_ADMIN : peut gérer les ressources et les utilisateurs du JICS.
- ROLE_USER : peut gérer les ressources JICS mais pas les utilisateurs.

Lister les utilisateurs

- Méthode prise en charge : GET
- Nécessite une authentification et des droits d'administrateur
- Trajectoire : `/api/services/security/servicelogin/listusers`
- Renvoie une liste de`com.netfective.bluage.jac.entities.SignOn`, sérialisée au format JSON.

Lister les utilisateurs

- Méthode prise en charge : GET
- Nécessite une authentification et des droits d'administrateur
- Trajectoire : `/api/services/security/servicelogin/listusers`
- Renvoie une liste de`com.netfective.bluage.jac.entities.SignOn`, sérialisée au format JSON.

Déconnecter l'utilisateur actuel

- Méthode prise en charge : GET
- Trajectoire : `/api/services/security/servicelogout/logout`
- Renvoie le message JSON suivant : « {" success » :true} » si la déconnexion de l'utilisateur actuel a réussi (la session HTTP associée sera invalidée).

État de santé du serveur Jics

- Méthode prise en charge : GET

- Trajectoire : `/api/services/rest/jicsserver/serverIsUp`
- Indique que le serveur JICS est opérationnel si vous recevez une réponse ayant le statut HTTP 200.

Structures de données

Cette section décrit en détail les différentes structures de données.

Rubriques

Structure du message Job Execution Details

Les détails de l'exécution de chaque tâche comporteront les champs suivants :

ID du script

: l'identifiant du script appelé.

appelant

: Adresse IP de l'appelant.

identifiant

: identifiant unique d'exécution de la tâche.

startTime

: date et heure auxquelles l'exécution de la tâche a commencé.

endTime

: date et heure auxquelles l'exécution de la tâche s'est terminée.

status

un statut pour l'exécution de la tâche. Une valeur possible parmi :

- `DONE`: l'exécution de la tâche s'est terminée normalement.
- `TRIGGERED`: exécution de la tâche déclenchée mais pas encore lancée.
- `RUNNING`: l'exécution de la tâche est en cours.
- `KILLED`: l'exécution du travail a été supprimée.
- `FAILED`: échec de l'exécution de la tâche.

Résultat de l'exécution

un message résumant le résultat de l'exécution de la tâche. Ce message peut être un simple message si l'exécution de la tâche n'est pas encore terminée ou une structure JSON contenant les champs suivants :

- ExitCode : code de sortie numérique ; les valeurs négatives indiquent des situations de défaillance.
- programme : dernier programme lancé par le job.
- status : une valeur possible parmi :
  - `Error`: lorsque ExitCode = -1 ; cela correspond à une erreur (technique) survenue lors de l'exécution du job.
  - `Failed`: lorsque exitcode = -2 ; Cela correspond à une défaillance survenant lors de l'exécution d'un programme de service (comme dans une situation ABEND).
  - `Succeeded`: lorsque ExitCode >= 0 ;
- StepName : nom de la dernière étape exécutée dans le job.

Mode d'exécution

SYNCHRONE ou ASYNCHRONE, selon la manière dont la tâche a été lancée.

Exemple de sortie :

```
{
    "scriptId": "INTCALC",
    "caller": "127.0.0.1",
    "identifier": "97d410be-efa7-4bd3-b7b9-d080e5769771",
    "startTime": "06-09-2023 11:42:41",
    "endTime": "06-09-2023 11:42:42",
```

```
    "status": "DONE",
    "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
\"CBACT04C\", \"status\": \"Error\" }",
    "executionMode": "ASYNCHRONOUS"
  }
```

Structure des résultats du lancement de la transaction

La structure peut contenir les champs suivants :

Résultat

une chaîne représentant le résultat de l'exécution de la transaction. Les valeurs possibles sont :

- `Success`: l'exécution de la transaction s'est terminée correctement.
- `Failure`: l'exécution de la transaction ne s'est pas terminée correctement, certains problèmes sont survenus.

commarée

une chaîne représentant la valeur finale de COMMAREA, sous la forme d'un tableau d'octets codé par 64 octets. Il peut s'agir d'une chaîne vide.

Container Record

(facultatif) une chaîne représentant le contenu de l'enregistrement du CONTENEUR sous la forme d'un tableau d'octets codé sur 64 octets.

Description du serveur

Peut contenir des informations sur le serveur qui a répondu à la demande (à des fins de débogage). Il peut s'agir d'une chaîne vide.

Un code Bend

(facultatif) si le programme référencé par la transaction lancée est modifié, la valeur du code d'abend sera renvoyée sous forme de chaîne dans ce champ.

Exemples de réponses :

Réussite

```
{
```

```
    "outCome": "Success",
    "commarea": "",
    "serverDescription": ""
  }
```

Échec

```
{
    "outCome": "Failure",
    "commarea": "",
    "serverDescription": "",
    "abendCode": "AEIA"
  }
```

Structure des résultats de l'enregistrement du lancement de la transaction

La structure peut contenir les champs suivants :

Enregistrer le contenu

une chaîne représentant le contenu de l'enregistrement de la COMMAREA sous la forme d'un tableau d'octets codé par 64 octets.

Container Record

une chaîne représentant le contenu de l'enregistrement du CONTENEUR sous la forme d'un tableau d'octets codé sur 64 octets.

Description du serveur

Peut contenir des informations sur le serveur qui a répondu à la demande (à des fins de débogage). Il peut s'agir d'une chaîne vide.

Exemples de réponses :

Réussite

```
{
    "recordContent": "",
    "serverDescription": ""
}
```

État possible d'une tâche dans une file d'attente

Dans une file d'attente, les tâches peuvent avoir le statut suivant :

ACTIF

La tâche est actuellement exécutée dans la file d'attente.

EXECUTION_WAIT

La tâche attend qu'un fil de discussion soit disponible.

PLANIFIÉ

Les tâches sont planifiées pour être exécutées à une date et à une heure spécifiques.

HOLD

Job attend d'être publié avant d'être exécuté.

TERMINÉ

Job exécuté avec succès.

ÉCHEC

L'exécution du Job a échoué.

UNKNOWN

Le statut est inconnu.

Soumettre une offre d'emploi

L'entrée de la tâche de soumission est la sérialisation JSON d'un `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objet. L'exemple d'entrée ci-dessous présente tous les champs correspondant à un tel haricot.

Exemple de saisie :

```
{
    "messageQueueName":null,
    "scheduleDate":null,
    "scheduleTime":null,
    "programName":"PTA0044",
```

```
    "programParams":
     {"wmind":"B"},
    "localDataAreaValue":"",
    "userName":"USER1",
    "jobName":"PTA0044",
    "jobNumber":9,
    "jobPriority":5,
    "executionDate":"20181231",
    "jobQueue":"queue1",
    "jobOnHold":false
}
```

Numéro de poste

si le numéro de tâche est 0, le numéro de tâche sera automatiquement généré en utilisant le numéro suivant dans la séquence des numéros de tâche. Cette valeur doit être définie sur 0 (sauf à des fins de test).

Priorité de l'emploi

La priorité de tâche par défaut dans l'AS400 est 5. La plage valide est comprise entre 0 et 9, 0 étant la priorité la plus élevée.

jobOnHold

Si une tâche est soumise en attente, elle ne sera pas exécutée immédiatement, mais uniquement lorsque quelqu'un la « publie ». Une tâche peut être publiée à l'aide de l'API REST (/release ou / release-all).

ScheduleDate et ScheduleTime

Si ces valeurs ne sont pas nulles, la tâche sera exécutée à la date et à l'heure spécifiées.

Date

peut être fourni au format MMDdyy ou DDMmyyyy (la taille de l'entrée déterminera le format utilisé)

Heure

peut être fourni avec le format HHmm ou HHMMss (la taille de l'entrée déterminera le format utilisé)

Paramètres du programme

cela sera transmis au programme sous forme de carte.

Réponse à la liste des tâches planifiées

Il s'agit de la structure du point de terminaison de la file d'attente des tâches list-jobs. Veuillez noter que le message de soumission d'offre d'emploi utilisé pour soumettre cette offre d'emploi fait partie de la réponse. Cela peut être utilisé à des fins de suivi, de test/de soumission à nouveau. Lorsqu'une tâche est terminée, les dates de début et de fin sont également renseignées.


```
[
    {
      "qrtzJobGroup": "PTA0044-PTA0044",
      "qrtzJobName": "PTA0044-168156109957800",
      "status": "HOLD",
      "qrtzDelay": 0,
      "startDate": null,
      "endDate": null,
      "jobName": "PTA0044",
      "userName": "USER1",
      "jobNumber": 9,
      "jobPriority": 5,
      "jobQueue": "queue1",
      "message": {
        "messageQueueName": null,
        "scheduleDate": null,
        "scheduleTime": null,
        "programName": "PTA0044",
        "programParams": {
          "wmind": "B"
        },
        "localDataAreaValue": "",
        "userName": "USER1",
        "jobName": "PTA0044",
        "jobNumber": 9,
        "jobPriority": 5,
        "executionDate": "20181231",
        "jobQueue": "queue1",
        "jobOnHold": true
      }
    },
    {
      "qrtzJobGroup": "PTA0044-PTA0044",
      "qrtzJobName": "PTA0044-166196954877600",
      "status": "COMPLETED",
      "qrtzDelay": 0,
```

```
      "startDate": "2022-10-13T22:48:34.025+00:00",
      "endDate": "2022-10-13T22:52:54.475+00:00",
      "jobName": "PTA0044",
      "userName": "USER1",
      "jobNumber": 9,
      "jobPriority": 5,
      "jobQueue": "queue1",
      "message": {
        "messageQueueName": null,
        "scheduleDate": null,
        "scheduleTime": null,
        "programName": "PTA0044",
        "programParams": {
          "wmind": "B"
        },
        "localDataAreaValue": "",
        "userName": "USER1",
        "jobName": "PTA0044",
        "jobNumber": 9,
        "jobPriority": 5,
        "executionDate": "20181231",
        "jobQueue": "queue1",
        "jobOnHold": true
      }
    }
  ]
```

Liste des réponses « en attente » concernant les offres d'emploi

La structure est similaire à la précédente, sauf que tous les emplois renvoyés seront « en attente ».

```
[
    {
      "qrtzJobGroup": "PTA0044-PTA0044",
      "qrtzJobName": "PTA0044-168156109957800",
      "status": "HOLD",
      "qrtzDelay": 0,
      "startDate": null,
      "endDate": null,
      "jobName": "PTA0044",
      "userName": "USER1",
      "jobNumber": 9,
```

```
      "jobPriority": 5,
      "jobQueue": "queue1",
      "message": {
        "messageQueueName": null,
        "scheduleDate": null,
        "scheduleTime": null,
        "programName": "PTA0044",
        "programParams": {
          "wmind": "B"
        },
        "localDataAreaValue": "",
        "userName": "USER1",
        "jobName": "PTA0044",
        "jobNumber": 9,
        "jobPriority": 5,
        "executionDate": "20181231",
        "jobQueue": "queue1",
        "jobOnHold": true
      }
    }
  ]
```

AWS Configuration de Blu Age Runtime (non géré)

Cette section explique les étapes à suivre pour configurer AWS Blu Age Runtime (non géré) sur votre AWS infrastructure.

Rubriques

- AWS Prérequis pour Blu Age Runtime
- AWS Intégration à Blue Age Runtime
- Exigences de configuration de l'infrastructure pour AWS Blu Age Runtime (non géré)
- AWS Déploiement de Blu Age Runtime sur Amazon ECS géré par AWS Fargate
- AWS Déploiement de Blu Age Runtime sur Amazon EC2

AWS Prérequis pour Blu Age Runtime

AWS Blu Age Runtime (non géré) est disponible en plusieurs versions. Si vous avez des projets de modernisation en cours, vous pourriez avoir besoin de versions incrémentielles du runtime à des fins

de mise en œuvre et de test. Pour définir vos besoins, contactez votre responsable de livraison AWS Blu Age.

Avant de commencer le processus d'intégration (non géré) de AWS Blu Age Runtime, procédez comme suit :

- Assurez-vous d'avoir un AWS compte.
- Assurez-vous de disposer d'une application modernisée refactorisée avec Blu Age. AWS
- Choisissez une AWS région et un ordinateur (Amazon EC2 ou Amazon ECS activé AWS Fargate).
- Choisissez la version de AWS Blu Age Runtime que vous souhaitez utiliser.
- Vérifiez the section called "Exigences de configuration de l'infrastructure" et validez les composants supplémentaires requis pour exécuter le AWS Blu Age Runtime (non géré).

> **Note**
>
> Si vous souhaitez tester les fonctionnalités de AWS Blu Age Runtime (non géré), vous pouvez utiliser l'application de démonstration`Planets Demo`, que vous pouvez télécharger depuis le PlanetsDemofichier -v1.zip.

AWS Intégration à Blue Age Runtime

Pour commencer, créez un AWS Support dossier pour demander l'intégration afin d'accéder à AWS Blu Age Runtime. Incluez dans votre demande votre Compte AWS identifiant, la AWS région que vous souhaitez utiliser, ainsi qu'un choix de calcul et une version d'exécution. Si vous n'êtes pas sûr de la version dont vous avez besoin, contactez votre responsable de livraison AWS Blu Age.

AWS Blu Age Runtime (non géré) sur Amazon EC2

Nous stockons les artefacts AWS Blu Age Runtime (non gérés) dans différents compartiments Amazon S3 par région et par choix de calcul. Pour accéder au bucket correspondant à votre environnement d'exécution Région AWS pour AWS Blu Age (non géré) sur Amazon EC2, utilisez le nom indiqué dans le tableau suivant.

Région AWS	Libération du godet	Construction du godet
USA Est (Ohio)	aws-bluage-runtime-artifacts-055777665268-us-est-2	aws-bluage-runtime-artifacts-dev-055777665268-us-est-2
USA Est (Virginie du Nord)	aws-bluage-runtime-artifacts-139023371234-us-est-1	aws-bluage-runtime-artifacts-dev-139023371234-us-est-1
USA Ouest (Californie du Nord)	aws-bluage-runtime-artifacts-788454048782-us-ouest-1	aws-bluage-runtime-artifacts-dev-788454048782-us-ouest-1
USA Ouest (Oregon)	aws-bluage-runtime-artifacts-836771190483-us-ouest-2	aws-bluage-runtime-artifacts-dev-836771190483-us-ouest-2
Europe (Irlande)	aws-bluage-runtime-artifacts-925278190477-eu-ouest-1	aws-bluage-runtime-artifacts-dev-925278190477-eu-west-1
Europe (Paris)	aws-bluage-runtime-artifacts-673009995881-eu-ouest-3	aws-bluage-runtime-artifacts-dev-673009995881-eu-west-3
Europe (Francfort)	aws-bluage-runtime-artifacts-485196800481-eu-central-1	aws-bluage-runtime-artifacts-dev-485196800481-eu-central-1
Amérique du Sud (São Paulo)	aws-bluage-runtime-artifacts-737536804457-sa-est-1	aws-bluage-runtime-artifacts-dev-737536804457-sa-east-1
Asie-Pacifique (Tokyo)	aws-bluage-runtime-artifacts-445578176276-ap-northeast-1	aws-bluage-runtime-artifacts-dev-445578176276-ap-northeast-1
Asie-Pacifique (Sydney)	aws-bluage-runtime-artifacts-726160321909-ap-southeast-2	aws-bluage-runtime-artifacts-dev-726160321909-ap-southeast-2

AWS Blu Age Runtime (non géré) sur Amazon ECS géré par Fargate

Nous stockons les artefacts AWS Blu Age Runtime (non gérés) dans différents compartiments Amazon S3 par région et par choix de calcul. Pour accéder au bucket correspondant à votre

environnement d'exécution Région AWS pour AWS Blu Age (non géré) sur Amazon ECS géré par Fargate, utilisez le nom indiqué dans le tableau suivant.

Région AWS	Libération du godet	Construction du godet
USA Est (Ohio)	aws-bluage-runtime-fargate-rel-483416914331-us-est-2	aws-bluage-runtime-fargate-dev-483416914331-us-est-2
USA Est (Virginie du Nord)	aws-bluage-runtime-fargate-rel-308472162679-us-est-1	aws-bluage-runtime-fargate-dev-308472162679-us-est-1
USA Ouest (Californie du Nord)	aws-bluage-runtime-fargate-rel-343763094578-us-ouest-1	aws-bluage-runtime-fargate-dev-343763094578-us-ouest-1
USA Ouest (Oregon)	aws-bluage-runtime-fargate-rel-688933007849-us-ouest-2	aws-bluage-runtime-fargate-dev-688933007849-us-ouest-2
Europe (Irlande)	aws-bluage-runtime-fargate-rel-140138033705-eu-west-1	aws-bluage-runtime-fargate-dev-140138033705-eu-west-1
Europe (Paris)	aws-bluage-runtime-fargate-rel-339712948211-eu-west-3	aws-bluage-runtime-fargate-dev-339712948211-eu-west-3
Europe (Francfort)	aws-bluage-runtime-fargate-rel-339712918892-eu-central-1	aws-bluage-runtime-fargate-dev-339712918892-eu-central-1
Amérique du Sud (São Paulo)	aws-bluage-runtime-fargate-rel-767397998881-sa-east-1	aws-bluage-runtime-fargate-dev-767397998881-sa-est-1
Asie-Pacifique (Tokyo)	aws-bluage-runtime-fargate-rel-891377400849-ap-northeast-1	aws-bluage-runtime-fargate-dev-891377400849-ap-northeast-1
Asie-Pacifique (Sydney)	aws-bluage-runtime-fargate-rel-533267435478-ap-southeast-2	aws-bluage-runtime-fargate-dev-533267435478-ap-southeast-2

Utiliser la ligne de commande pour répertorier le contenu du bucket

Une fois l'intégration terminée, vous pouvez répertorier le contenu du bucket en exécutant la commande suivante dans un terminal.

```
aws s3 ls bucket-name
```

Remplacez `bucket-name` par le nom du bucket correspondant Région AWS à votre nom indiqué dans le tableau précédent.

Cette commande renvoie une liste de dossiers correspondant aux différentes versions du moteur d'exécution (non géré) de AWS Blu Age Runtime, tels que

```
PRE 3.3.0.1/
PRE 3.3.0.2/
```

Nous vous recommandons d'utiliser la dernière version disponible. Si cela n'est pas possible, utilisez la version d'exécution qui a été validée lors de la phase de refactorisation de l'application. Pour répertorier les frameworks disponibles pour une version spécifique, exécutez la commande suivante :

```
aws s3 ls s3://bucket-name/version/Framework/
```

`bucket-name`Remplacez-le par le nom du bucket correspondant à votre Région AWS compte et `version` par la version de votre choix. Voici un exemple.

```
aws s3 ls s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/
Framework/
```

La commande renverra une liste de frameworks, tels que :

```
2023-06-05 10:26:52   97960225 aws-bluage-runtime-3.4.0.tar.gz
2023-06-05 10:27:12         45 aws-bluage-runtime-3.4.0.tar.gz.checksumSHA256
2023-06-05 10:27:14  138497123 aws-bluage-webapps-3.4.0.tar.gz
2023-06-05 10:27:44         45 aws-bluage-webapps-3.4.0.tar.gz.checksumSHA256
```

Téléchargez le framework

Vous pouvez télécharger le framework par exemple pour mettre à niveau la version d'exécution Velocity sur une instance Amazon EC2 existante.

```
aws s3 cp s3://bucket-name/version/Framework/ folder-of-your-choice --
recursive
```

Où :

folder-of-your-choice

chemin du dossier dans lequel vous souhaitez télécharger le framework.

Par exemple : `aws s3 cp s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/Framework/ . --recursive`

Cette commande produit le résultat suivant :

```
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/
Framework/aws-bluage-runtime-3.4.0.tar.gz.checksumSHA256 to ./aws-bluage-
runtime-3.4.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/
Framework/aws-bluage-webapps-3.4.0.tar.gz.checksumSHA256 to ./aws-bluage-
webapps-3.4.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/Framework/aws-
bluage-webapps-3.4.0.tar.gz to ./aws-bluage-webapps-3.4.0.tar.gz
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/3.4.0/Framework/aws-
bluage-runtime-3.4.0.tar.gz to ./aws-bluage-runtime-3.4.0.tar.gz
```

Vous pouvez répertorier les fichiers du framework comme suit :

```
ls -l
```

Cette commande produit le résultat suivant :

```
total 230928
-rw-rw-r-- 1 cloudshell-user cloudshell-user  97960225 Jun  5 10:26 aws-bluage-
runtime-3.4.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Jun  5 10:27 aws-bluage-
runtime-3.4.0.tar.gz.checksumSHA256
-rw-rw-r-- 1 cloudshell-user cloudshell-user 138497123 Jun  5 10:27 aws-bluage-
webapps-3.4.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Jun  5 10:27 aws-bluage-
webapps-3.4.0.tar.gz.checksumSHA256
```

Exigences de configuration de l'infrastructure pour AWS Blu Age Runtime (non géré)

Cette rubrique décrit la configuration d'infrastructure minimale requise pour exécuter AWS Blu Age Runtime (non géré). Les procédures suivantes décrivent comment configurer AWS Blu Age Runtime (non géré) sur l'ordinateur de votre choix pour déployer une application modernisée sur le AWS Blu Age Runtime. Les ressources que vous créez doivent se trouver dans un Amazon VPC doté d'un sous-réseau dédié à votre domaine d'application.

Création d'un groupe de sécurité

1. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
2. Dans le volet de navigation de gauche, sous Sécurité, sélectionnez Groupes de sécurité.
3. Dans le volet central, choisissez Create security group.
4. Dans le champ Nom du groupe de sécurité, entrez**`M2BluagePrivateLink-SG`**.
5. Dans la section Règles entrantes, choisissez Ajouter une règle.
6. Pour Type, choisissez HTTPS.
7. Dans Source, entrez votre VPC CIDR.
8. Dans la section Règles sortantes, choisissez Ajouter une règle.
9. Pour Type, choisissez HTTPS.
10. En regard de Destination, entrez **`0.0.0.0/0`**.
11. Sélectionnez Create security group (Créer un groupe de sécurité).

Création d'un point de terminaison Amazon VPC

1. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
2. Dans le volet de navigation de gauche, sous Virtual private cloud, sélectionnez Endpoints.
3. Dans le volet central, choisissez Create endpoint.
4. Dans la section Services, entrez **`SQS`** dans le champ de recherche, puis sélectionnez le service Amazon SQS correspondant à votre région.
5. Dans la section VPC, sélectionnez l'Amazon VPC que vous avez créé à l'étape précédente.
6. Dans la section Sous-réseaux, sélectionnez le sous-réseau que vous avez créé pour votre domaine d'application.

7. Dans la section Groupes de sécurité, sélectionnez M2 BluagePrivateLink -SG.
8. Choisissez Créer un point de terminaison.

Créer une politique IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, sous Gestion des accès, sélectionnez Politiques.
3. Dans le volet central, choisissez Create policy.
4. Dans la section Éditeur de politique, sélectionnez JSON.
5. Remplacez tout le JSON que vous voyez dans l'éditeur par le JSON suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Si vous avez besoin de plus de détails pour personnaliser votre politique, contactez votre responsable de livraison ou votre responsable de compte AWS Blu Age.

6. Choisissez Suivant.
7. Entrez le nom de la stratégie, puis choisissez Create policy.

Créer un rôle IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le volet de navigation de gauche, sous Gestion des accès, sélectionnez Rôles.
3. Dans le volet central, choisissez Create role.
4. Dans la section Cas d'utilisation, choisissez EC2 ou Elastic Container Service (puis Elastic Container Service Task), en fonction de votre choix de calcul.
5. Choisissez Suivant.
6. Dans le champ de recherche, entrez le nom de la politique que vous avez créée précédemment.
7. Cochez la case située à gauche de votre police d'assurance.
8. Choisissez Suivant.
9. Entrez un nom pour le rôle, puis choisissez Créer un rôle.

Exécution de AWS Blu Age Runtime sur Amazon EC2

Création d'une instance Amazon EC2

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Lancer l'instance.
3. Pour Type d'instance, choisissez l'un des types répertoriés dansthe section called "Types d'instances Amazon EC2 pour AWS Blu Age Runtime (sur Amazon EC2)".
4. Dans la section Paire de clés, choisissez une paire de clés existante ou créez-en une nouvelle.
5. Dans la section Paramètres réseau, choisissez Sélectionner un groupe de sécurité existant.
6. Pour les groupes de sécurité communs, choisissez M2 BluagePrivateLink -SG.
7. Développez la section Détails avancés.
8. Pour le profil d'instance IAM, choisissez le rôle IAM que vous avez créé précédemment.
9. Choisissez Launch instance (Lancer une instance).

Lorsque l'état de l'instance Amazon EC2 passe à En cours d'exécution, connectez-vous à l'instance et installez les composants logiciels suivants :

- Environnement d'exécution Java (JRE) 11.
- Apache Tomcat 9.
- AWS Blu Age Runtime (sur Amazon EC2). Installez le moteur d'exécution AWS Blu Age à la racine du dossier d'installation d'Apache Tomcat (certains fichiers seront ajoutés tandis que d'autres seront remplacés).

Si vous souhaitez installer des applications Web supplémentaires, installez-les dans un dossier distinct. Dans ce cas, répétez le processus d'installation d'Apache Tomcat, puis de l'archive par-dessus.

Exécution de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Créez un cluster Amazon ECS et un rôle de tâche à utiliser lors du lancement d'une AWS Fargate tâche ultérieurement. Pour le rôle de tâche, incluez la politique que vous avez créée précédemment. Selon votre scénario, vous devrez peut-être associer des politiques IAM supplémentaires au rôle de tâche.

Types d'instances Amazon EC2 pour AWS Blu Age Runtime (sur Amazon EC2)

Voici une liste des types d'instances Amazon EC2 que vous pouvez utiliser pour AWS Blu Age Runtime (sur Amazon EC2).

```
t3.xlarge
t3.small
t3.large
t2.small
t2.large
r6i.xlarge
r6i.large
r6i.4xlarge
r6i.2xlarge
r5b.xlarge
r5b.large
r5b.2xlarge
r3.xlarge
m6i.xlarge
m6i.large
m6i.8xlarge
m6i.4xlarge
m6i.2xlarge
m6i.16xlarge
m5zn.xlarge
m5zn.large
m5zn.3xlarge
m5zn.2xlarge
m5.xlarge
m5.large
m5.8xlarge
```

```
m5.4xlarge
m5.2xlarge
m5.16xlarge
m5.12xlarge
c6i.xlarge
c6i.large
c6i.8xlarge
c6i.4xlarge
c6i.2xlarge
c6i.16xlarge
c5.xlarge
c5.large
c5.9xlarge
c5.4xlarge
c5.2xlarge
c5.18xlarge
c5.12xlarge
```

AWS Déploiement de Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Les rubriques de cette section décrivent comment configurer AWS Blu Age Runtime sur Amazon ECS géré par Amazon ECS AWS Fargate, comment mettre à jour la version d'exécution, comment surveiller votre déploiement à l'aide des CloudWatch alarmes Amazon et comment ajouter des dépendances sous licence.

Rubriques

- Configuration de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate
- Mise à niveau du AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate
- Amazon CloudWatch Alarms pour AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate
- Configuration des dépendances sous licence dans AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Configuration de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Cette rubrique explique comment configurer et déployer l' PlanetsDemo exemple d'application à l'aide de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate.

AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate est disponible pour Linux/X86.

Rubriques

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- Configurez le AWS CLI en suivant les étapes décrites dans Configuration de l'AWS CLI.
- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "AWS Intégration à Blue Age Runtime".
- Téléchargez le AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate des fichiers binaires. Pour obtenir des instructions, veuillez consulter the section called "AWS Intégration à Blue Age Runtime".
- Téléchargez les fichiers binaires d'Apache Tomcat 9.
- Téléchargez l'archive de PlanetsDemo l'application.
- Créez une base de données Amazon Aurora PostgreSQL pour JICS et exécutez `PlanetsDemo-v1/jics/sql/initJics.sql` la requête dessus. Pour plus d'informations sur la création d'une base de données Amazon Aurora PostgreSQL, consultez Création et connexion à un cluster de bases de données Aurora PostgreSQL.

Configuration

Pour configurer l' PlanetsDemo exemple d'application, procédez comme suit.

1. Après avoir téléchargé les fichiers binaires d'Apache Tomcat, extrayez le contenu et accédez au `conf` dossier. Ouvrez le `catalina.properties` fichier pour le modifier et remplacez la ligne `common.loader` commençant par la ligne suivante.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Compressez le dossier Apache Tomcat en utilisant la commande tar pour créer une archive « tar.gz ».

3. Préparez un Dockerfile pour créer votre image personnalisée en fonction des fichiers binaires d'exécution et des fichiers binaires du serveur Apache Tomcat fournis. Consultez l'exemple de Dockerfile suivant. L'objectif est d'installer Apache Tomcat 9, puis d'installer AWS Blu Age Runtime (pour Amazon ECS géré par AWS Fargate) extrait à la racine du répertoire d'installation d'Apache Tomcat 9, puis d'installer l'exemple d'application modernisée nommé. PlanetsDemo

> **Note**
>
> Le contenu des scripts install-gapwalk.sh et install-app.sh, utilisés dans cet exemple de Dockerfile, est répertorié après le Dockerfile.

```
FROM --platform=linux/x86_64 amazonlinux:2

RUN mkdir -p /workdir/apps
WORKDIR /workdir
COPY install-gapwalk.sh .
COPY install-app.sh .
RUN chmod +x install-gapwalk.sh
RUN chmod +x install-app.sh

# Install Java and AWS CLI v2-y
RUN yum install sudo java-17-amazon-corretto unzip tar -y
RUN sudo yum remove awscli -y
RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
RUN sudo unzip awscliv2.zip
RUN sudo ./aws/install

#·Installation dir
RUN mkdir -p /usr/local/velocity/installation/gapwalk
# Copy PlanetsDemo archive to a dedicated apps dir
COPY PlanetsDemo-v1.zip /workdir/apps/

# Copy resources (tomcat, blu age runtime) to installation dir
COPY tomcat.tar.gz /usr/local/velocity/installation/tomcat.tar.gz
COPY aws-bluage-on-fargate-runtime-3.10.0.15.tar.gz /usr/local/velocity/
installation/gapwalk/gapwalk-bluage-on-fargate.tar.gz

# run relevant installation scripts
RUN ./install-gapwalk.sh
```

```
RUN ./install-app.sh

EXPOSE 8080
EXPOSE 8081
# ...

# Run Command to start Tomcat server
CMD ["sh", "-c", "sudo /bluage-on-fargate/tomcat.gapwalk/velocity/startup.sh
 $ECS_CONTAINER_METADATA_URI_V4 $AWS_CONTAINER_CREDENTIALS_RELATIVE_URI"]
```

Le contenu du fichier install-gapwalk.sh est le suivant.

```
#!/bin/sh

# Vars
TEMP_DIR=/bluage-on-fargate/tomcat.gapwalk/temp

# Install
echo "Installing Gapwalk and Tomcat"
sudo rm -rf /bluage-on-fargate
mkdir -p ${TEMP_DIR}
# Copy Blu Age runtime and tomcat archives to temporary extraction dir
sudo cp /usr/local/velocity/installation/gapwalk/gapwalk-bluage-on-fargate.tar.gz
 ${TEMP_DIR}
sudo cp /usr/local/velocity/installation/tomcat.tar.gz ${TEMP_DIR}
#·Create velocity dir
mkdir -p /bluage-on-fargate/tomcat.gapwalk/velocity
#·Extract tomcat files
tar -xvf ${TEMP_DIR}/tomcat.tar.gz -C ${TEMP_DIR}
# Copy all tomcat files to velocity dir
cp -fr ${TEMP_DIR}/apache-tomcat-9.x.x/* /bluage-on-fargate/tomcat.gapwalk/velocity
# Remove default webapps of Tomcat
rm -f /bluage-on-fargate/tomcat.gapwalk/velocity/webapps/*
# Extract Blu Age runtime at velocity dir
tar -xvf ${TEMP_DIR}/gapwalk-bluage-on-fargate.tar.gz -C /bluage-on-fargate/
tomcat.gapwalk
# Remove temporary extraction dir
sudo rm -rf ${TEMP_DIR}
```

Le contenu du fichier install-app.sh est le suivant.

```
#!/bin/sh
```

```
APP_DIR=/workdir/apps
TOMCAT_GAPWALK_DIR=/bluage-on-fargate/tomcat.gapwalk

unzip ${APP_DIR}/PlanetsDemo-v1.zip -d ${APP_DIR}
cp -r ${APP_DIR}/webapps/* ${TOMCAT_GAPWALK_DIR}/velocity/webapps/
cp -r ${APP_DIR}/config/* ${TOMCAT_GAPWALK_DIR}/velocity/config/
```

4. Fournissez les informations de connexion pour la base de données que vous avez créée dans le cadre des prérequis dans l'extrait suivant du `application-main.yml` fichier, qui se trouve dans le dossier. `{TOMCAT_GAPWALK_DIR}/config` Pour plus d'informations, voir Création et connexion à un cluster de base de données Aurora PostgreSQL.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

5. Créez et transférez l'image dans votre référentiel Amazon ECR. Pour obtenir des instructions, consultez la section Envoyer une image Docker dans le guide de l'utilisateur d'Amazon Elastic Container Registry.
6. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
7. Dans le volet de navigation de gauche, sélectionnez Définitions de tâches.
8. Dans Type de lancement, sélectionnez AWS Fargate.
9. Sélectionnez le rôle de tâche que vous avez créé dans le cadre de celui-cithe section called "Exigences de configuration de l'infrastructure".
10. Joignez votre image au conteneur.
11. Terminez de remplir le formulaire, puis choisissez Créer.
12. Dans le volet de navigation de gauche, choisissez Clusters, puis choisissez votre cluster dans la liste.
13. Sur la page de détails de votre cluster, sous l'onglet Services, choisissez Create.
14. Sélectionnez la définition de tâche.

15. Développez la section Mise en réseau et configurez le VPC, les sous-réseaux et le groupe de sécurité dont vous avez fait partie. the section called "Exigences de configuration de l'infrastructure"
16. Déployez votre service Amazon ECS.

Si le déploiement échoue, consultez les journaux. Pour les trouver, rendez-vous sur la page des tâches dans Amazon ECS managed by AWS Fargate, puis choisissez l'onglet Logs. Si vous trouvez des codes d'erreur commençant par un C suivi d'un chiffre, par exemple CXXXX, notez les messages d'erreur. Par exemple, le code d'erreur C5102 est une erreur courante indiquant une configuration d'infrastructure incorrecte. Vous pouvez également naviguer dans votre tâche en cours d'exécution et exécuter quelques commandes, comme dans le cas de AWS Blu Age Runtime (sur Amazon EC2). Pour plus d'informations, consultez la section Utilisation d'Amazon ECS Exec pour le débogage dans le manuel Amazon Elastic Container Service Developer Guide.

Pour ouvrir un shell interactif, exécutez la commande suivante depuis votre ordinateur local.

```
aws ecs execute-command --cluster your_cluster_name --container your_container_name --
task task_id --interactive --command /bin/sh
```

Testez l' PlanetsDemo application

Pour vérifier l'état de l' PlanetsDemo application déployée, exécutez les commandes suivantes après l'avoir `listener-port` remplacée`load-balancer-DNS-name`, `web-binary-name` avec les valeurs correctes pour votre configuration.

```
curl http://load-balancer-DNS-name:listener-port/gapwalk-application/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche :`Jics application is running`.

Exécutez ensuite la commande suivante.

```
curl http://load-balancer-DNS-name:listener-port/jac/api/services/rest/jicsservice/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche :`Jics application is running`.

```
Jics application is running
```

Si vous avez configuré Blusam, vous pouvez vous attendre à une réponse vide lorsque vous exécutez la commande suivante.

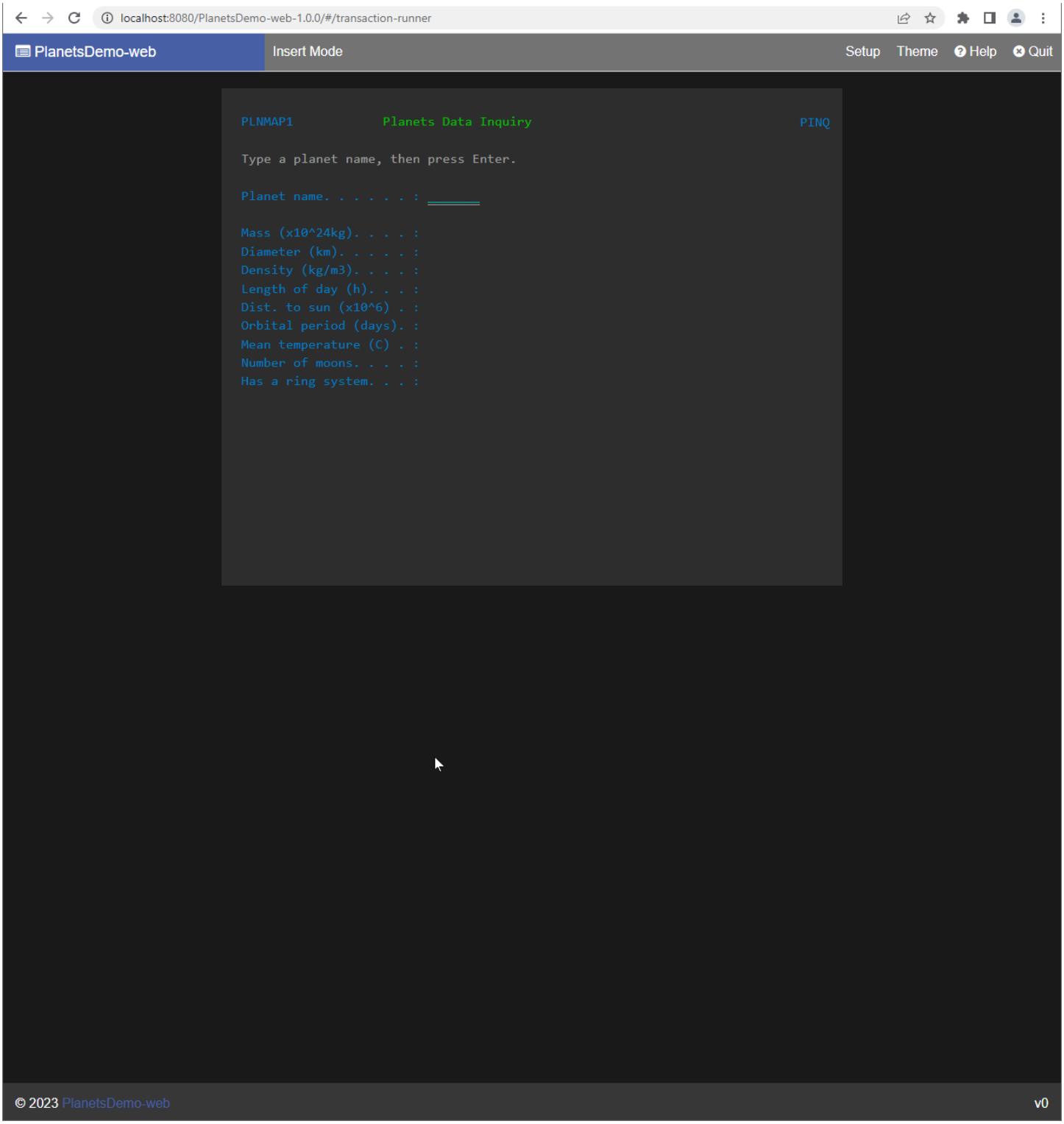
```
curl http://load-balancer-DNS-name:listener-port/bac/api/services/rest/bluesamserver/
serverIsUp
```

Notez le nom du binaire Web (PlanetsDemo-web-1.0.0, s'il est inchangé). Pour accéder à l' PlanetsDemoapplication, utilisez une URL au format suivant.

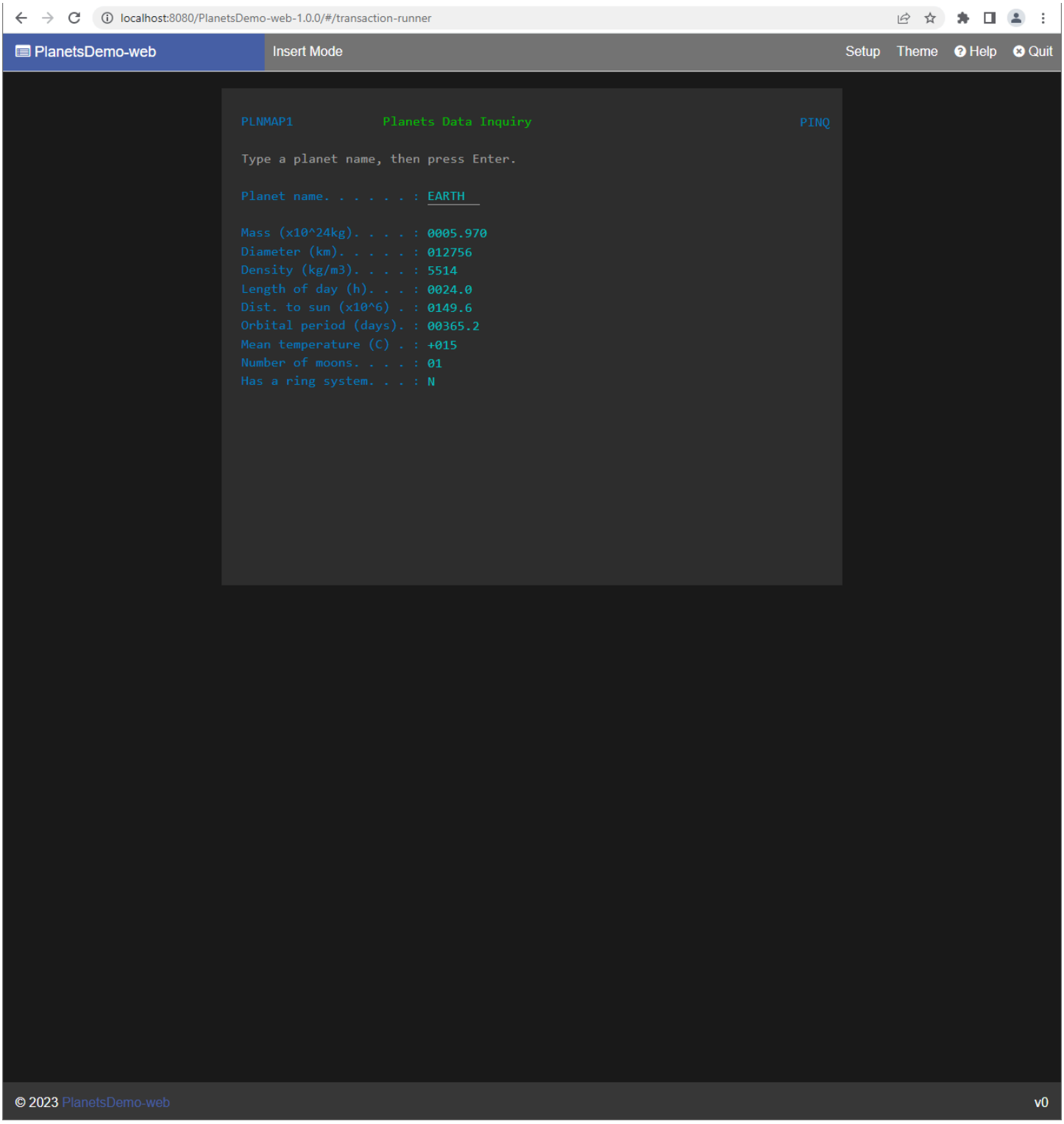
```
https://load-balancer-DNS-name:listener-port/web-binary-name
```

Après le démarrage de PlanetsDemo l'application, la page d'accueil s'affiche.

Entrez PINQ dans la zone de texte, puis appuyez sur Entrée. La page de demande de données s'affiche.



Par exemple, entrez EARTH dans le champ de PlanetsDemo nom, puis appuyez sur Entrée. La page de la planète que vous avez saisie s'affiche.



Mise à niveau du AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Ce guide explique comment mettre à niveau le AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate.

Rubriques

- Prérequis
- Améliorez le runtime Velocity

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "AWS Intégration à Blue Age Runtime".
- Téléchargez la version du AWS Blu Age Runtime vers laquelle vous souhaitez effectuer la mise à niveau. Pour de plus amples informations, veuillez consulter the section called "AWS Intégration à Blue Age Runtime". Le framework se compose de deux fichiers binaires : `aws-bluage-runtime-x.x.x.x.tar.gz` et`aws-bluage-webapps-x.x.x.x.tar.gz`.

Améliorez le runtime Velocity

Procédez comme suit pour mettre à niveau le moteur d'exécution Velocity.

1. Reconstruisez votre image Docker avec la version de AWS Blu Age Runtime souhaitée. Pour obtenir des instructions, veuillez consulter the section called "Configuration de AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate".
2. Transférez votre image Docker dans votre référentiel Amazon ECR.
3. Arrêtez et redémarrez votre service Amazon ECS.
4. Vérifiez les journaux.

Le AWS Blu Age Runtime est correctement mis à niveau.

Amazon CloudWatch Alarms pour AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Pour bénéficier de notifications plus visibles chaque fois que vos applications déployées rencontrent des exceptions, CloudWatch configurez la réception de votre journal des applications et ajoutez une alarme pour vous avertir d'éventuelles erreurs.

Configuration de l'alarme

Avec CloudWatch les journaux, vous pouvez configurer autant de métriques et d'alarmes que vous le souhaitez, en fonction de votre application et de vos besoins.

Plus précisément, vous pouvez configurer des alarmes proactives pour les alertes d'utilisation directement lors de la création de votre cluster Amazon ECS, afin d'être averti en cas d'erreur. Pour mettre en évidence les erreurs de connexion au système de contrôle AWS Blu Age, ajoutez une métrique concernant la chaîne « Error C » dans les journaux. Vous pouvez ensuite définir une alarme qui réagit à cette métrique.

Configuration des dépendances sous licence dans AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate

Cette rubrique explique comment configurer des dépendances sous licence supplémentaires que vous pouvez utiliser avec AWS Blu Age Runtime sur Amazon ECS géré par AWS Fargate.

Rubriques

- Prérequis
- Présentation

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called “AWS Prérequis pour Blu Age Runtime”Complet etthe section called “AWS Intégration à Blue Age Runtime”.
- Obtenez les dépendances suivantes à partir de leur source.

Oracle Database

Fournissez un pilote de base de données Oracle. Par exemple, ojdbc8-19.8.0.0.jar.

Connexion IBM MQ

Fournissez un client IBM MQ. Par exemple, com.ibm.mq.allclient-9.3.0.15.jar.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- javax.jms-api-2.0.1.jar
- json-20080701.jar

Fichiers d'imprimante DDS

Fournissez la bibliothèque de rapports Jasper. Par exemple, jasperreports-6.16.0.jar, mais une version plus récente peut être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar

Présentation

Pour installer les dépendances, procédez comme suit.

1. Copiez l'une des dépendances ci-dessus selon les besoins dans votre dossier de génération d'image Docker.
2. Si votre base de données JICS ou Blusam est hébergée sur Oracle, fournissez le pilote de base de données Oracle dans. *`your-tomcat-path`*`/extra`
3. Sur votre Dockerfile, copiez ces dépendances dans. *`your-tomcat-path`*`/extra`
4. Créez votre image Docker, puis envoyez-la vers Amazon ECR.
5. Arrêtez et redémarrez votre service Amazon ECS.
6. Consultez les journaux.

AWS Déploiement de Blu Age Runtime sur Amazon EC2

Les rubriques de cette section décrivent comment configurer AWS Blu Age Runtime (non géré) sur Amazon EC2, comment mettre à jour la version d'exécution, comment surveiller votre déploiement à l'aide des alarmes CloudWatch Amazon et comment ajouter des dépendances sous licence.

Rubriques

- Configuration de AWS Blu Age Runtime (non géré) sur Amazon EC2
- Mise à niveau du AWS Blu Age Runtime sur Amazon EC2
- AWS Blu Age Runtime (sur Amazon EC2) Amazon Alarmes CloudWatch
- Configuration des dépendances sous licence dans AWS Blu Age Runtime sur Amazon EC2

Configuration de AWS Blu Age Runtime (non géré) sur Amazon EC2

Cette rubrique explique comment configurer et déployer l' PlanetsDemo exemple d'application à l'aide de AWS Blu Age Runtime (non géré) sur Amazon EC2.

Rubriques

- Prérequis
- Configuration
- Testez l' PlanetsDemo application

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- Configurez le AWS CLI en suivant les étapes décrites dans Configuration de l'AWS CLI.
- the section called “AWS Prérequis pour Blu Age Runtime”Complet etthe section called “AWS Intégration à Blue Age Runtime”.
- Créez une instance Amazon EC2 contenant le dernier AWS Blu Age Runtime (sur Amazon EC2). Pour plus d'informations, veuillez consulter Mise en route avec les instances Linux Amazon EC2.
- Assurez-vous de pouvoir vous connecter correctement à l'instance Amazon EC2, par exemple à l'aide de SSM.

- Téléchargez et extrayez AWS Blu Age Runtime (sur Amazon EC2) à l'adresse. *your-tomcat-path*/* Pour obtenir des instructions, veuillez consulter the section called "AWS Intégration à Blue Age Runtime".
- Téléchargez l'archive de PlanetsDemo l'application.
- Décompressez l'archive et chargez l'application dans le compartiment Amazon S3 de votre choix.
- Créez une base de données Amazon Aurora PostgreSQL pour JICS et exécutez `PlanetsDemo-v1/jics/sql/initJics.sql` la requête dessus. Pour plus d'informations sur la création d'une base de données Amazon Aurora PostgreSQL, consultez Création et connexion à un cluster de base de données Aurora PostgreSQL.

Configuration

Pour configurer l' PlanetsDemo exemple d'application, procédez comme suit.

1. Connectez-vous à votre instance Amazon EC2 et accédez au dossier situé sous votre `conf` dossier d'installation d'Apache Tomcat 9. Ouvrez le `catalina.properties` fichier pour le modifier et remplacez la ligne `common.loader` commençant par la ligne suivante.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Accédez au dossier *<your-tomcat-path>*/webapps.
3. Copiez les PlanetsDemo fichiers binaires disponibles dans le dossier PlanetsDemo -v1/webapps/ depuis le compartiment Amazon S3 à l'aide de la commande suivante.

```
aws s3 cp s3://path-to-demo-app-webapps/ . --recursive
```

> **Note**
>
> `path-to-demo-app-webapps`Remplacez-le par l'URI Amazon S3 correct pour le compartiment dans lequel vous avez précédemment décompressé l' PlanetsDemo archive.

4. Copiez le contenu du `PlanetsDemo-v1/config/` dossier dans*<your-tomcat-path>*/`config/`.

5. Fournissez les informations de connexion pour la base de données que vous avez créée dans le cadre des prérequis dans l'extrait de code suivant du fichier. `application-main.yml` Pour plus d'informations, voir Création et connexion à un cluster de base de données Aurora PostgreSQL.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

6. Démarrez votre serveur Apache Tomcat et vérifiez les journaux.

```
your-tomcat-path/startup.sh

tail -f your-tomcat-path/logs/catalina.log
```

Si vous trouvez des codes d'erreur commençant par un C suivi d'un chiffre, par exemple CXXXX, notez les messages d'erreur. Par exemple, le code d'erreur C5102 est une erreur courante indiquant une configuration d'infrastructure incorrecte.

Testez l' PlanetsDemo application

Pour vérifier l'état de l' PlanetsDemo application déployée, exécutez les commandes suivantes après l'avoir `listener-port` remplacée`load-balancer-DNS-name`, `web-binary-name` avec les valeurs correctes pour votre configuration.

```
curl http://load-balancer-DNS-name:listener-port/gapwalk-application/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche : `Jics application is running`

Exécutez ensuite la commande suivante.

```
curl http://load-balancer-DNS-name:listener-port/jac/api/services/rest/jicsservice/
```

Si l'application est en cours d'exécution, le message de sortie suivant s'affiche : `Jics application is running`

```
Jics application is running
```

Si vous avez configuré Blusam, vous pouvez vous attendre à une réponse vide lorsque vous exécutez la commande suivante.

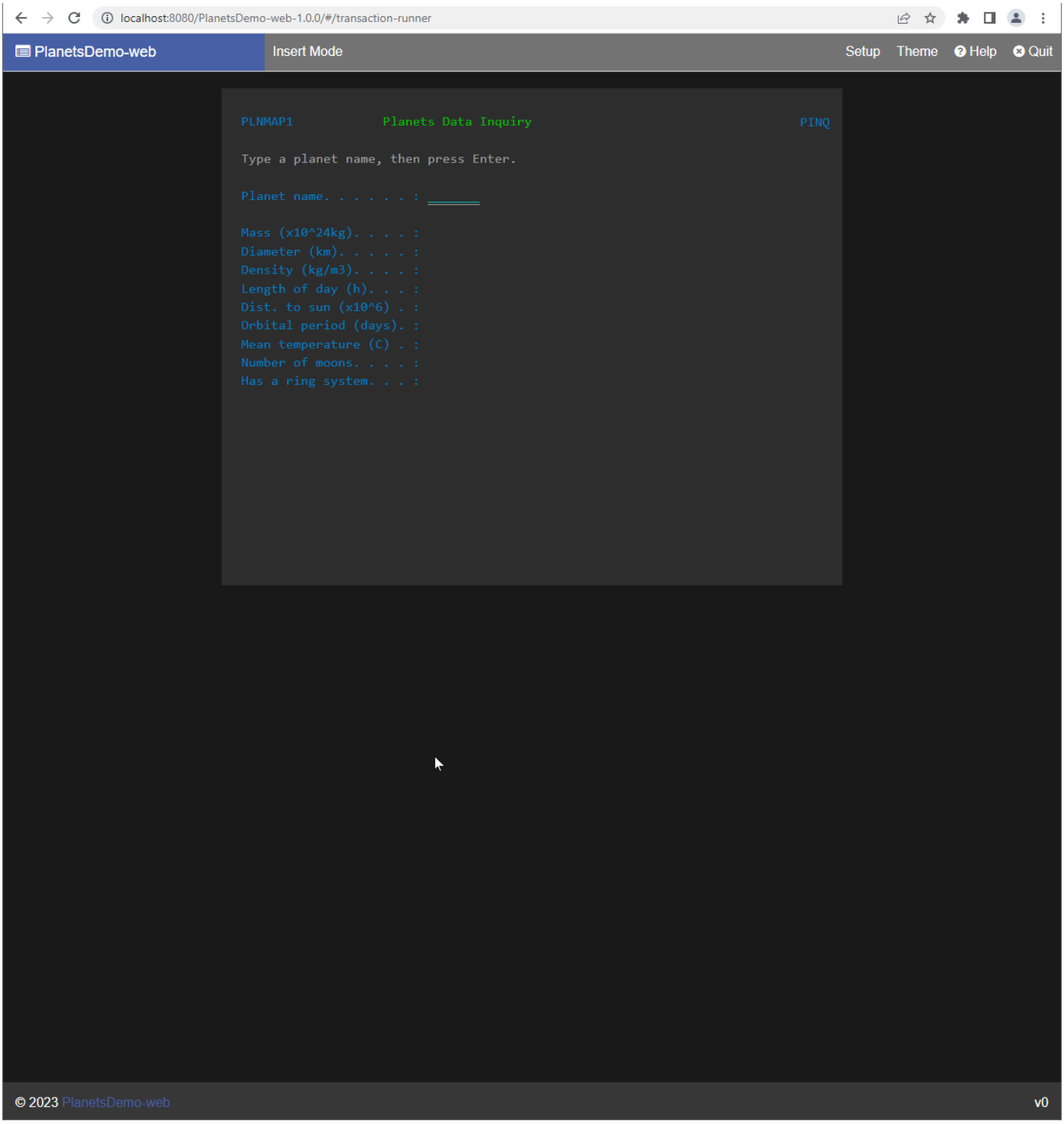
```
curl http://load-balancer-DNS-name:listener-port/bac/api/services/rest/bluesamserver/serverIsUp
```

Notez le nom du binaire Web (PlanetsDemo-web-1.0.0, s'il est inchangé). Pour accéder à l' PlanetsDemo application, utilisez une URL au format suivant.

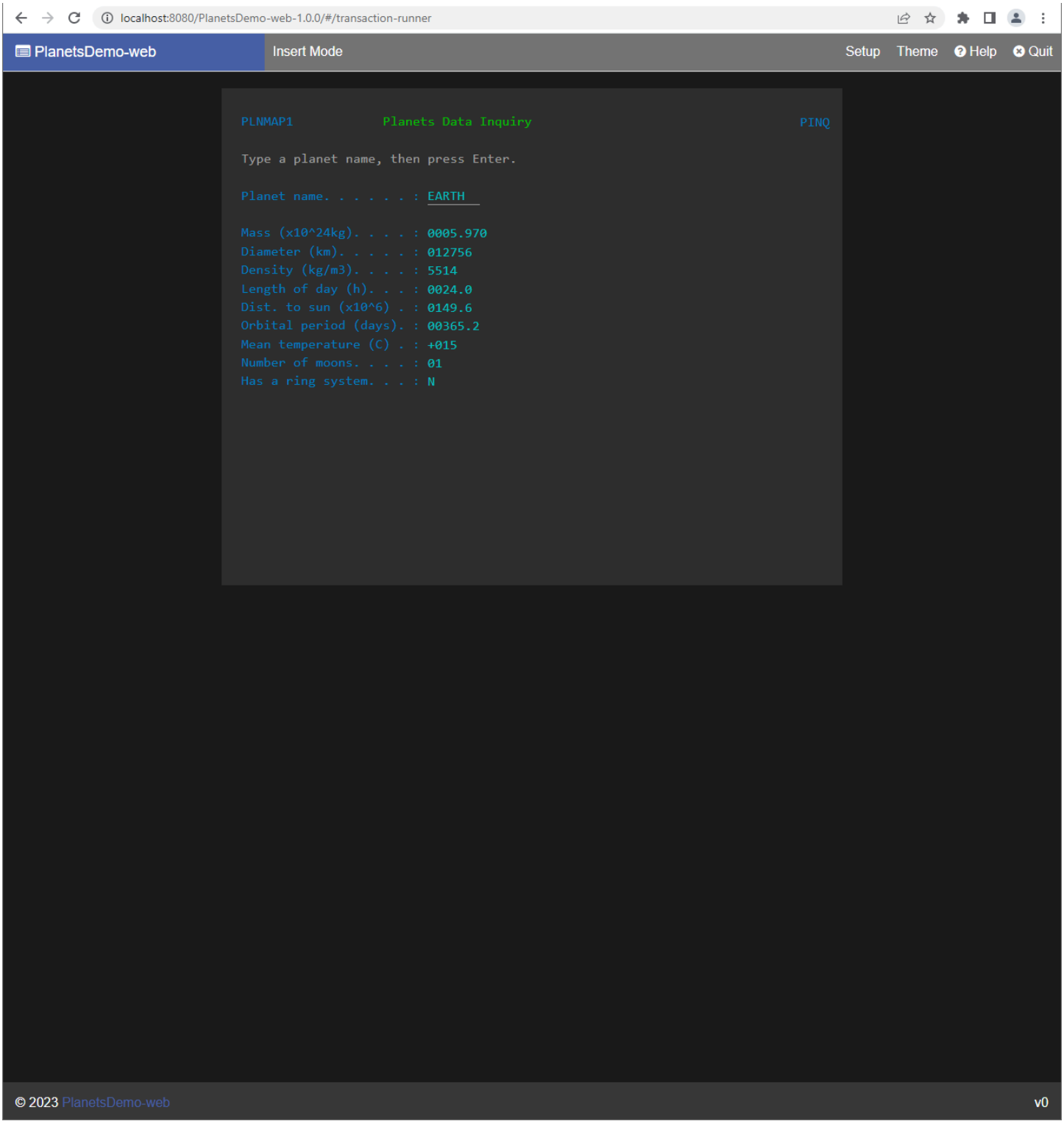
```
https://load-balancer-DNS-name:listener-port/web-binary-name
```

Après le démarrage de PlanetsDemo l'application, la page d'accueil s'affiche.

Entrez PINQ dans la zone de texte, puis appuyez sur Entrée. La page de demande de données s'affiche.



Par exemple, entrez EARTH dans le champ de PlanetsDemo nom, puis appuyez sur Entrée. La page de la planète que vous avez saisie s'affiche.



Mise à niveau du AWS Blu Age Runtime sur Amazon EC2

Ce guide explique comment mettre à niveau le AWS Blu Age Runtime sur Amazon EC2.

Rubriques

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "AWS Intégration à Blue Age Runtime".
- Assurez-vous que vous disposez d'une instance Amazon EC2 qui contient le dernier environnement d'exécution AWS Blu Age. Pour plus d'informations, veuillez consulter Mise en route avec les instances Linux Amazon EC2.
- Assurez-vous de pouvoir vous connecter correctement à l'instance Amazon EC2, par exemple à l'aide de SSM.
- Téléchargez la version du AWS Blu Age Runtime (sur Amazon EC2) vers laquelle vous souhaitez effectuer la mise à niveau. Pour plus d'informations, voir the section called "AWS Configuration de Blu Age Runtime (non géré)" Le framework se compose de deux fichiers binaires : `aws-bluage-runtime-x.x.x.x.tar.gz` et`aws-bluage-webapps-x.x.x.x.tar.gz`.

Présentation

Procédez comme suit pour mettre à niveau le moteur d'exécution Velocity.

1. Connectez-vous à votre instance Amazon EC2 et remplacez l'utilisateur par su en exécutant la commande suivante.

   ```
   sudo su
   ```

   Vous devez disposer du privilège de superutilisateur pour exécuter des commandes dans ce didacticiel.
2. Créez deux dossiers, un pour chaque fichier binaire.
3. Donnez à chaque dossier le même nom que le fichier binaire.
4. Copiez chaque fichier binaire dans le dossier correspondant.

> **Warning**
>
> L'extraction de chaque binaire produit un dossier portant le même nom. Par conséquent, si vous extrayez les deux fichiers binaires au même endroit l'un après l'autre, vous remplacerez le contenu.

5. Pour extraire les fichiers binaires, utilisez les commandes suivantes. Exécutez les commandes dans chaque dossier.

```
tar xvf aws-bluage-runtime-x.x.x.x.tar.gz
tar xvf aws-bluage-webapps-x.x.x.x.tar.gz
```

6. Arrêtez les services Apache Tomcat à l'aide des commandes suivantes.

```
systemctl stop tomcat.service
systemctl stop tomcat-webapps.service
```

7. Remplacez le contenu de `<your-tomcat-path>/shared/` par le contenu de`aws-bluage-runtime-x.x.x.x/velocity/shared/`.
8. Remplacez `<your-tomcat-path>/webapps/gapwalk-application.war` par `aws-bluage-runtime-x.x.x.x/velocity/webapps/gapwalk-application.war`.
9. Remplacez les fichiers war dans`<your-tomcat-path>/webapps/`, à savoir `bac.war` et`jac.war`, par les mêmes fichiers provenant de`aws-bluage-webapps-x.x.x.x/velocity/webapps/`.
10. Démarrez les services Apache Tomcat en exécutant les commandes suivantes.

```
systemctl start tomcat.service
systemctl start tomcat-webapps.service
```

11. Consultez les journaux.

Pour vérifier l'état de l'application déployée, exécutez les commandes suivantes.

```
curl http://localhost:8080/gapwalk-application/
```

Le message suivant doit apparaître.

```
Jics application is running
```

```
curl http://localhost:8181/jac/api/services/rest/jicsservice/
```

Le message suivant doit apparaître.

```
Jics application is running
```

```
curl http://localhost:8181/bac/api/services/rest/bluesamserver/serverIsUp
```

La réponse doit être vide.

Le moteur d'exécution de AWS Blu Age a été correctement mis à niveau.

AWS Blu Age Runtime (sur Amazon EC2) Amazon Alarmes CloudWatch

Afin de bénéficier de notifications plus visibles lorsque vos applications déployées rencontrent des exceptions susceptibles de placer votre application dans une période de grâce, vous pouvez CloudWatch configurer la réception de votre journal d'applications et ajouter une alarme pour vous avertir d'éventuelles erreurs.

Déploiement de la CloudWatch journalisation

Par défaut, le `application-main.yml` fichier inclut une référence à un autre fichier de configuration de journalisation nommé`logback-cloudwatch.yml`.

```
logging:
  config: classpath:logback-cloudwatch.xml
```

Les deux fichiers se trouvent dans le dossier de configuration et c'est ainsi que la CloudWatch journalisation est configurée, comme expliqué dans les sections suivantes.

Configuration de la CloudWatch journalisation

Le contenu `logback-cloudwatch.xml` du fichier par défaut est le suivant.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration>
<configuration>

    <appender name="console" class="ch.qos.logback.core.ConsoleAppender">
```

```
        <encoder>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </encoder>
    </appender>

    <appender name="cloudwatch"
 class="com.netfective.bluage.runtime.cloudwatchlogger.CloudWatchAppender">
        <logGroup>BluAgeRuntimeOnEC2-Logs</logGroup>
        <logStream>%date{yyyy-MM-dd,UTC}.%instanceId.%uuid</logStream>
        <layout>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </layout>
        <appender-ref ref="console" />
    </appender>

    <root level="INFO">
        <appender-ref ref="cloudwatch" />
    </root>
</configuration>
```

Tout ce qui se trouve en dehors de l'<appender name="cloudwatch"/>élément est une configuration de logback standard. Ce fichier contient deux annexes : un annexe de console pour envoyer les journaux à la console et un CloudWatch annexe pour envoyer les journaux. CloudWatch

L'`level`attribut de l'`root`élément indique le niveau de journalisation de l'ensemble de l'application.

Les valeurs requises à l'intérieur de la balise <appender name="cloudwatch"/> sont les suivantes :

- <logGroup/>:Définit le nom du groupe de journaux dans CloudWatch. Si la valeur n'est pas spécifiée, elle est définie par défaut sur`BluAgeRuntimeOnEC2-Logs`. Si le groupe de journaux n'existe pas, il sera créé automatiquement. Ce comportement peut être modifié par le biais de la configuration, décrite ci-dessous.
- <logStream/>: définit le nom du LogStream (à l'intérieur du groupe de journaux) dans. CloudWatch

Valeurs facultatives :

- <region/>: remplace la région dans laquelle le flux de log sera écrit. Par défaut, les journaux sont envoyés dans la même région que l'instance EC2.
- <layout/>: le modèle que les messages du journal utiliseront.

- <maxbatchsize/>: nombre maximal de messages de journal auxquels envoyer CloudWatch par opération.
- <maxbatchtimemillis/>: durée en millisecondes pendant laquelle les CloudWatch journaux peuvent être écrits.
- <maxqueuewaittimemillis/>: durée en millisecondes nécessaire pour essayer d'insérer des demandes dans la file d'attente interne du journal.
- <internalqueuesize/>: taille maximale de la file d'attente interne.
- <createlogdests/>: créez un groupe de journaux et un flux de journaux s'ils n'existent pas.
- <initialwaittimemillis/>: durée pendant laquelle vous souhaitez que le thread soit mis en veille au démarrage. Cette attente initiale permet une accumulation initiale de journaux.
- <maxeventmessagesize/>: taille maximale d'un événement de journal. Les journaux dont la taille est supérieure à cette taille ne seront pas envoyés.
- <truncateeventmessages/>: tronque les messages trop longs.
- <printrejectedevents/>: Activez l'annexe d'urgence.

CloudWatch configuration

Pour que la configuration ci-dessus envoie correctement les journaux vers CloudWatch, mettez à jour votre rôle de profil d'instance Amazon EC2 IAM afin de lui accorder des autorisations supplémentaires pour le groupe de journaux « BluAgeRuntimeOn EC2-Logs` et ses flux de journaux :

- `logs:CreateLogStream`
- `logs:DescribeLogStreams`
- `logs:CreateLogGroup`
- `logs:PutLogEvents`
- `logs:DescribeLogGroups`

Configuration de l'alarme

Grâce aux CloudWatch journaux, vous pouvez ensuite configurer différentes métriques et alarmes, en fonction de votre application et de vos besoins. Plus précisément, vous pouvez configurer des alarmes proactives pour les alertes d'utilisation, afin d'être averti en cas d'erreur susceptible de mettre votre application en période de grâce (et, en fin de compte, de l'empêcher de fonctionner). Pour ce faire, vous pouvez ajouter une métrique concernant la chaîne « Error C5001 » dans les

journaux, qui met en évidence les erreurs de connexion au système de contrôle AWS Blu Age. Vous pouvez ensuite définir une alarme qui réagit à cette métrique.

Configuration des dépendances sous licence dans AWS Blu Age Runtime sur Amazon EC2

Ce guide explique comment configurer des dépendances sous licence supplémentaires que vous pouvez utiliser avec AWS Blu Age Runtime sur Amazon EC2.

Rubriques

Prérequis

Avant de commencer, assurez-vous de remplir les conditions préalables suivantes.

- the section called "AWS Prérequis pour Blu Age Runtime"Complet etthe section called "AWS Intégration à Blue Age Runtime".
- Assurez-vous que vous disposez d'une instance Amazon EC2 contenant la dernière version de AWS Blu Age Runtime (sur Amazon EC2). Pour plus d'informations, veuillez consulter Mise en route avec les instances Linux Amazon EC2.
- Assurez-vous de pouvoir vous connecter correctement à l'instance Amazon EC2, par exemple à l'aide de SSM.
- Obtenez les dépendances suivantes à partir de leurs sources.

Oracle Database

Fournissez un pilote de base de données Oracle. Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec la version ojdbc8-19.8.0.0.jar, mais une version plus récente est peut-être compatible.

Connexion IBM MQ

Fournissez un client IBM MQ. Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec la version com.ibm.mq.allclient-9.3.0.15.jar, mais une version plus récente est peut-être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- javax.jms-api-2.0.1.jar
- json-20080701.jar

Fichiers d'imprimante DDS

Fournissez la bibliothèque de rapports Jasper. Nous avons testé la fonctionnalité AWS Blu Age Runtime (sur Amazon EC2) avec jasperreports-6.16.0.jar, mais une version plus récente est peut-être compatible.

Avec cette version de dépendance, fournissez également les dépendances transitives suivantes :

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar

Présentation

Pour installer les dépendances, procédez comme suit.

1. Connectez-vous à votre instance Amazon EC2 et remplacez l'utilisateur par su en exécutant la commande suivante.

   ```
   sudo su
   ```

   Vous devez disposer du privilège de superutilisateur pour exécuter des commandes dans ce didacticiel.
2. Accédez au dossier `<your-tomcat-path>/extra/`.

   ```
   cd <your-tomcat-path>/extra/
   ```

3. Copiez l'une des dépendances ci-dessus selon les besoins dans ce dossier.
4. Arrêtez et démarrez le tomcat.service en exécutant les commandes suivantes.

```
systemctl stop tomcat.service
```

```
systemctl start tomcat.service
```

5. Vérifiez l'état du service pour vous assurer qu'il fonctionne.

```
systemctl status tomcat.service
```

6. Vérifiez les journaux.

Configuration des dépendances pour les applications Web JAC et BAC

1. Si votre base de données JICS ou Blusam est hébergée sur Oracle, vous devez fournir le pilote de base de données Oracle. `<your-tomcat-path>/extra`
2. Créez le dossier s'il n'est pas déjà présent.
3. Arrêtez et redémarrez votre serveur Apache Tomcat.
4. Vérifiez les journaux.

Modifiez le code source avec Blu Age Developer IDE

Si vous utilisez le moteur d'exécution AWS Blu Age AWS géré, vous pouvez utiliser Blu Age Developer pour modifier le code source généré. Vous pouvez le faire si vous devez mettre à jour le code modernisé pour une raison ou une autre, ou si une partie de l'ancien code source ne peut pas être modernisée. Vous accédez à Blu Age Developer via Amazon AppStream 2.0. Cette section explique comment configurer Blu Age Developer sur AppStream 2.0. Il explique également comment utiliser Blu Age Developer pour mettre à jour le code source, à l'aide de l'exemple d'application PlanetsDemo.

Rubriques

- Tutoriel : Configuration de la AppStream version 2.0 pour AWS Blu Age Developer IDE
- Tutoriel : Utiliser AWS Blu Age Developer sur AppStream 2.0

Tutoriel : Configuration de la AppStream version 2.0 pour AWS Blu Age Developer IDE

AWS La modernisation du mainframe fournit plusieurs outils via Amazon AppStream 2.0. AppStream 2.0 est un service de streaming d'applications sécurisé et entièrement géré qui vous permet de diffuser des applications de bureau aux utilisateurs sans avoir à réécrire les applications. AppStream La version 2.0 fournit aux utilisateurs un accès instantané aux applications dont ils ont besoin avec une expérience utilisateur réactive et fluide sur l'appareil de leur choix. L'utilisation de la AppStream version 2.0 pour héberger des outils spécifiques au moteur d'exécution permet aux équipes d'application des clients d'utiliser les outils directement depuis leur navigateur Web, en interagissant avec les fichiers d'application stockés dans des compartiments ou des référentiels Amazon S3. CodeCommit

Pour plus d'informations sur la prise en charge des navigateurs dans la AppStream version 2.0, consultez la section Configuration système requise et prise en charge des fonctionnalités (navigateur Web) dans le guide d'administration Amazon AppStream 2.0. Si vous rencontrez des problèmes lors de l'utilisation de la AppStream version 2.0, consultez la section Résolution AppStream des problèmes liés aux utilisateurs de la AppStream version 2.0 dans le guide d'administration Amazon 2.0.

Ce document décrit comment configurer l'IDE AWS Blu Age Developer sur un parc AppStream 2.0.

Rubriques

Prérequis

Téléchargez le fichier d'archive qui contient les artefacts dont vous avez besoin pour configurer AWS Blu Age Developer IDE sous AppStream 2.0.

> **Note**
>
> Il s'agit d'un fichier volumineux. Si vous rencontrez des problèmes avec le délai d'expiration de l'opération, nous vous recommandons d'utiliser une instance Amazon EC2 pour améliorer les performances de chargement et de téléchargement.

Étape 1 : Créer un compartiment Amazon S3

Créez un compartiment Amazon S3 Région AWS identique à la flotte AppStream 2.0 que vous allez créer. Ce compartiment contiendra les artefacts dont vous avez besoin pour terminer ce didacticiel.

Étape 2 : associer une politique au compartiment S3

Attachez la politique suivante au bucket que vous créez pour ce didacticiel. Assurez-vous de le `MYBUCKET` remplacer par le nom réel du bucket que vous créez.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowAppStream2.0ToRetrieveObjects",
        "Effect": "Allow",
        "Principal": {
            "Service": "appstream.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::MYBUCKET/*"
    }]
}
```

Étape 3 : télécharger des fichiers dans le compartiment Amazon S3

Décompressez les fichiers que vous avez téléchargés dans le prérequis et chargez le `appstream` dossier dans votre compartiment. Le téléchargement de ce dossier crée la structure appropriée dans votre compartiment. Pour plus d'informations, consultez la section Chargement d'objets dans le guide de l'utilisateur Amazon S3.

Étape 4 : Téléchargez les AWS CloudFormation modèles

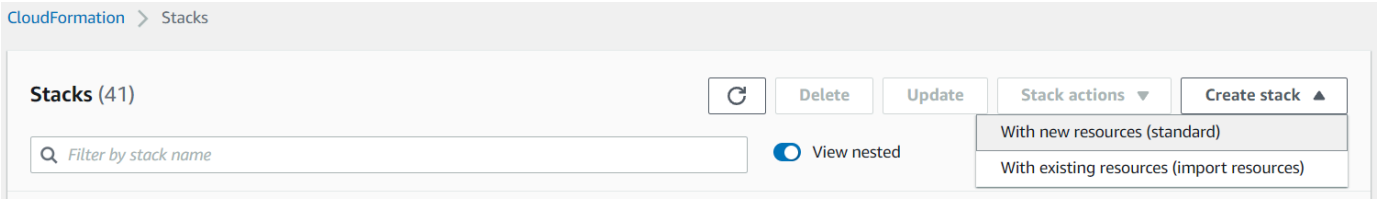
Téléchargez les AWS CloudFormation modèles suivants. Vous avez besoin de ces modèles pour créer et alimenter le parc AppStream 2.0.

- cfn-m2-.yaml appstream-elastic-fleet-linux
- cfn-m2- -linux.yaml appstream-bluage-dev-tools
- cfn-m2-.yaml appstream-bluage-shared-linux
- cfn-m2-.yaml appstream-chrome-linux
- cfn-m2-.yaml appstream-eclipse-jee-linux
- cfn-m2-.yaml appstream-pgadmin-linux

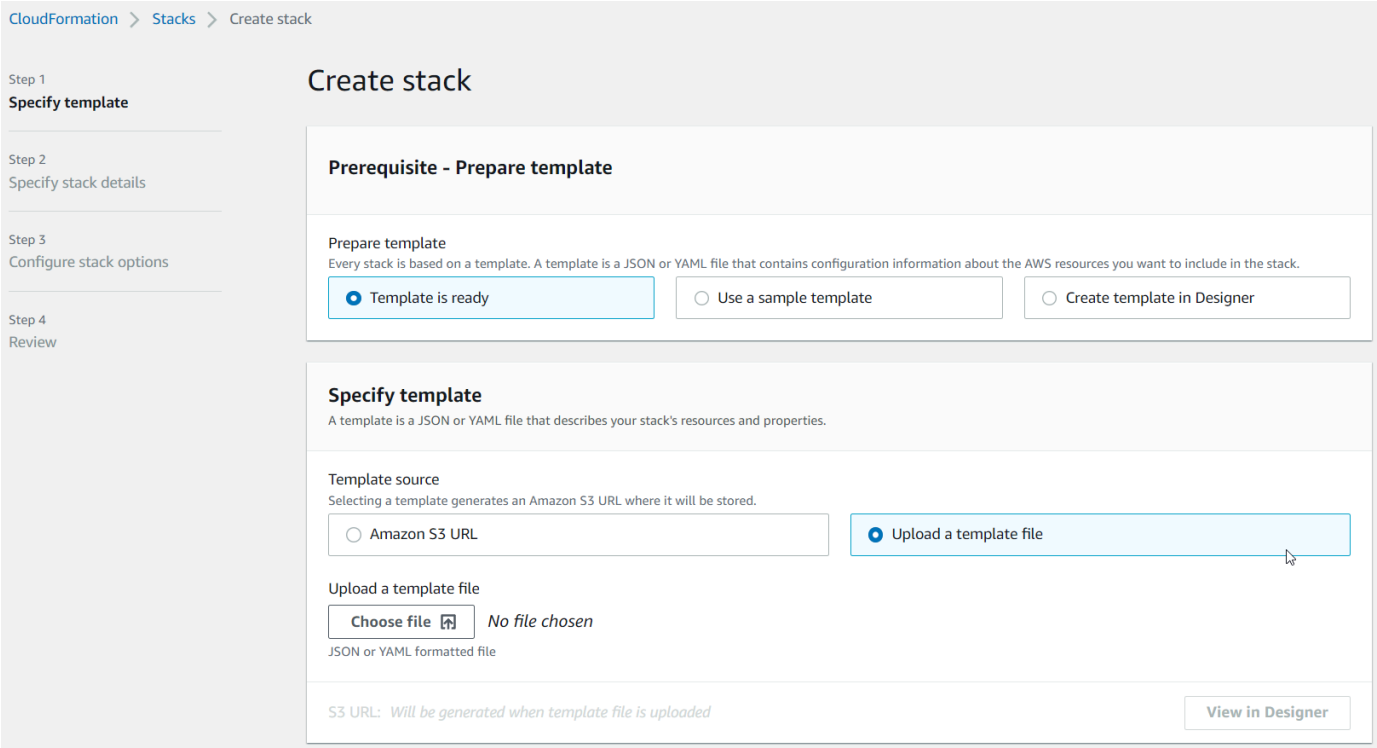
Étape 5 : Créez la flotte avec AWS CloudFormation

Au cours de cette étape, vous allez utiliser le `cfn-m2-appstream-elastic-fleet-linux.yaml` AWS CloudFormation modèle pour créer une flotte et une pile AppStream 2.0 afin d'héberger l'IDE AWS Blu Age Developer. Après avoir créé le parc et la pile, vous exécuterez les autres AWS CloudFormation modèles que vous avez téléchargés à l'étape précédente pour installer l'IDE Developer et les autres outils requis.

1. Accédez AWS CloudFormation à la console AWS de gestion, puis sélectionnez Stacks.
2. Dans Stacks, choisissez Create stack et With new Resources (standard) :



3. Dans Créer une pile, sélectionnez Le modèle est prêt et Chargez un fichier modèle :



4. Choisissez Choisir un fichier, puis naviguez jusqu'au fichier`cfn-m2-appstream-elastic-fleet-linux.yaml`. Choisissez Suivant.
5. Dans Spécifier les détails de la pile, fournissez les informations suivantes :

   - Nom de la pile.
   - Votre groupe de sécurité par défaut et les deux sous-réseaux de ce groupe de sécurité.

   > **Note**
   >
   > Les deux sous-réseaux du groupe de sécurité doivent se trouver dans des zones de disponibilité différentes.

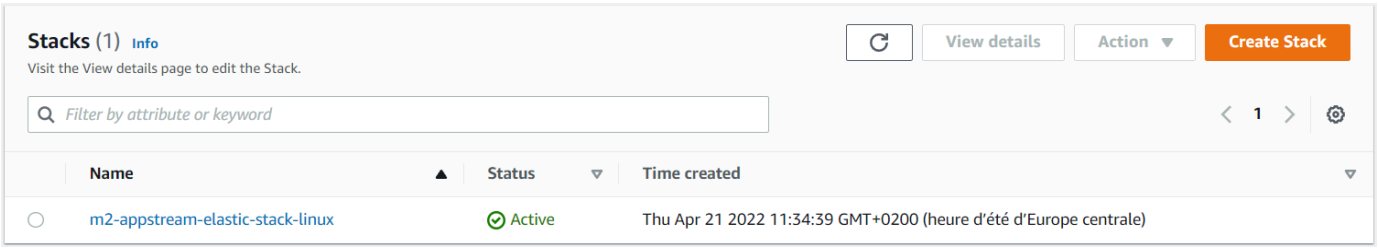
6. Choisissez Next, puis de nouveau Next.
7. Choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM avec des noms personnalisés. , puis choisissez Soumettre.
8. Après avoir créé le parc, créez des CloudFormation piles avec les autres modèles téléchargés pour terminer la configuration des applications. Veillez à effectuer la mise à jour à BucketNamechaque fois pour pointer vers le compartiment S3 approprié. Vous pouvez le modifier BucketNamedans la CloudFormation console. Vous pouvez également modifier directement les fichiers modèles et mettre à jour la `S3Bucket` propriété.

> **Note**
>
> Les modèles téléchargés s'attendent à trouver des ressources dans un compartiment S3 dont la structure de dossiers est appelée`appstream/bluage/developer-ide/`. Le compartiment doit se trouver dans le même emplacement Région AWS que le parc que vous avez créé.

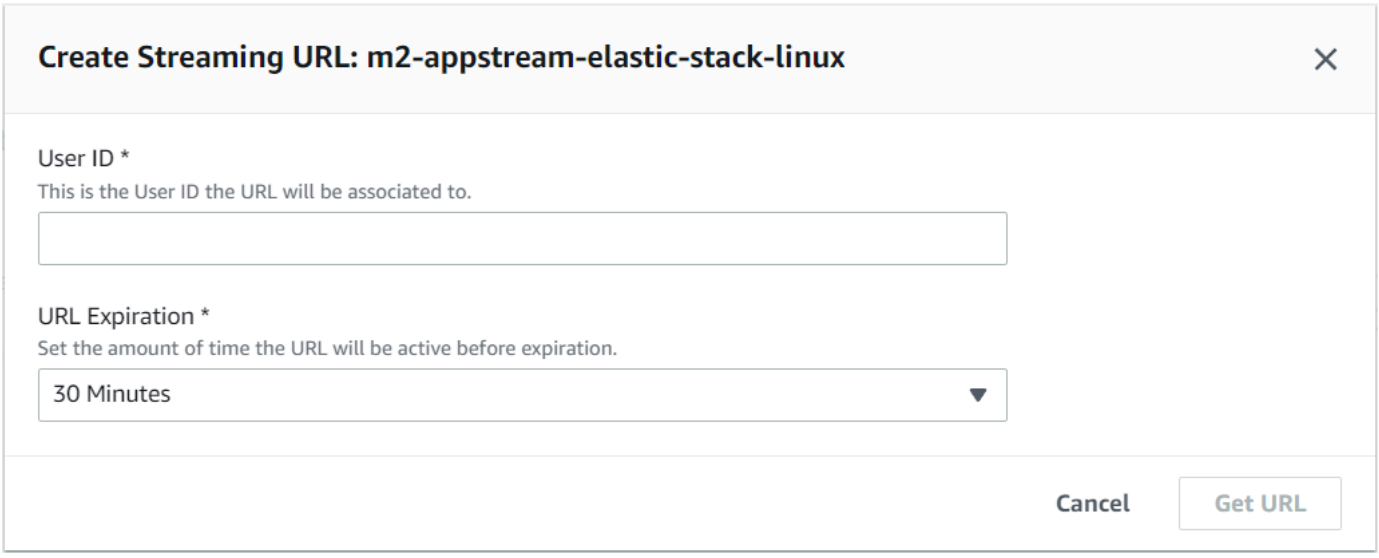
Étape 6 : accéder à une instance

Après avoir créé et démarré la flotte, vous pouvez créer un lien temporaire pour accéder à la flotte via le client natif.

1. Accédez à la AppStream version 2.0 dans le AWS Management Console et choisissez la pile créée précédemment :



2. Sur la page des détails de la pile, choisissez Action, puis Create Streaming URL :



3. Dans Create Streaming URL, entrez un ID utilisateur arbitraire et une date d'expiration de l'URL, puis choisissez Get URL. Vous obtenez une URL que vous pouvez utiliser pour diffuser du

contenu vers un navigateur ou vers le client natif. Nous vous recommandons de diffuser sur le client natif.

Nettoyage des ressources

Pour la procédure de nettoyage de la pile et des flottes créées, voir Créer une flotte et une pile AppStream 2.0.

Lorsque vous avez supprimé les objets AppStream 2.0, vous ou l'administrateur du compte pouvez également nettoyer les compartiments S3 pour les paramètres de l'application et les dossiers personnels.

> **Note**
>
> Le dossier d'accueil d'un utilisateur donné étant unique dans toutes les flottes, vous devrez peut-être le conserver si d'autres piles AppStream 2.0 sont actives sur le même compte.

Vous ne pouvez pas utiliser la console AppStream 2.0 pour supprimer des utilisateurs. Vous devez plutôt utiliser l'API du service avec le AWS CLI. Pour plus d'informations, consultez la section Administration du groupe d'utilisateurs dans le guide d'administration Amazon AppStream 2.0.

Tutoriel : Utiliser AWS Blu Age Developer sur AppStream 2.0

Ce didacticiel vous explique comment accéder à AWS Blu Age Developer sur AppStream 2.0 et comment l'utiliser avec un exemple d'application afin de tester les fonctionnalités. Lorsque vous aurez terminé ce didacticiel, vous pourrez suivre les mêmes étapes avec vos propres applications.

Rubriques

- Étape 1 : Créer une base de données
- Étape 2 : Accès à l'environnement
- Étape 3 : configurer le moteur d'exécution
- Étape 4 : démarrer l'IDE Eclipse
- Étape 5 : configurer un projet Maven
- Étape 6 : Configuration d'un serveur Tomcat
- Étape 7 : Déployer sur Tomcat

Étape 1 : Créer une base de données

Au cours de cette étape, vous utilisez Amazon RDS pour créer une base de données PostgreSQL gérée que l'application de démonstration utilise pour stocker les informations de configuration.

1. Ouvrez la console Amazon RDS.
2. Choisissez Bases de données > Créer une base de données.
3. Choisissez Standard create > PostgreSQL, conservez la version par défaut, puis choisissez Free tier.
4. Choisissez un identifiant d'instance de base de données.
5. Pour les paramètres d'identification, choisissez Gérer les informations d'identification principales dans AWS Secrets Manager. Pour plus informations, consultez Gestion des mots de passe avec Amazon RDS etAWS Secrets Manager (français non garanti) dans le Guide de l'utilisateur Amazon RDS (français non garanti).
6. Assurez-vous que le VPC est le même que celui que vous utilisez pour l'instance AppStream 2.0. Vous pouvez demander cette valeur à votre administrateur.
7. Pour le groupe de sécurité VPC, choisissez Create New.
8. Réglez l'accès public sur Oui.
9. Conservez toutes les autres valeurs par défaut. Passez en revue ces valeurs.
10. Choisissez Create database (Créer une base de données).

Pour rendre le serveur de base de données accessible depuis votre instance, sélectionnez le serveur de base de données dans Amazon RDS. Sous Connectivité et sécurité, choisissez le groupe de sécurité VPC pour le serveur de base de données. Ce groupe de sécurité a déjà été créé pour vous et doit avoir une description similaire à celle décrite dans la section Créé par la console de gestion RDS. Choisissez Action > Modifier les règles entrantes, choisissez Ajouter une règle et créez une règle de type PostgreSQL. Pour la source des règles, utilisez le groupe de sécurité par défaut. Vous pouvez commencer à taper le nom de la source dans le champ Source, puis accepter l'ID suggéré. Enfin, choisissez Enregistrer les règles.

Étape 2 : Accès à l'environnement

Au cours de cette étape, vous accédez à l'environnement de développement AWS Blu Age AppStream 2.0.

1. Contactez votre administrateur pour savoir comment accéder correctement à votre instance AppStream 2.0. Pour obtenir des informations générales sur les clients et les configurations possibles, consultez la section Méthodes d'accès et clients AppStream 2.0 du guide d'administration Amazon AppStream 2.0. Envisagez d'utiliser le client natif pour une expérience optimale.
2. Dans la AppStream version 2.0, choisissez Desktop.

Étape 3 : configurer le moteur d'exécution

Au cours de cette étape, vous allez configurer le moteur d'exécution AWS Blu Age. Vous devez configurer le moteur d'exécution au premier lancement, puis à nouveau si vous êtes informé d'une mise à niveau du moteur d'exécution. Cette étape permet de remplir votre `.m2` dossier.

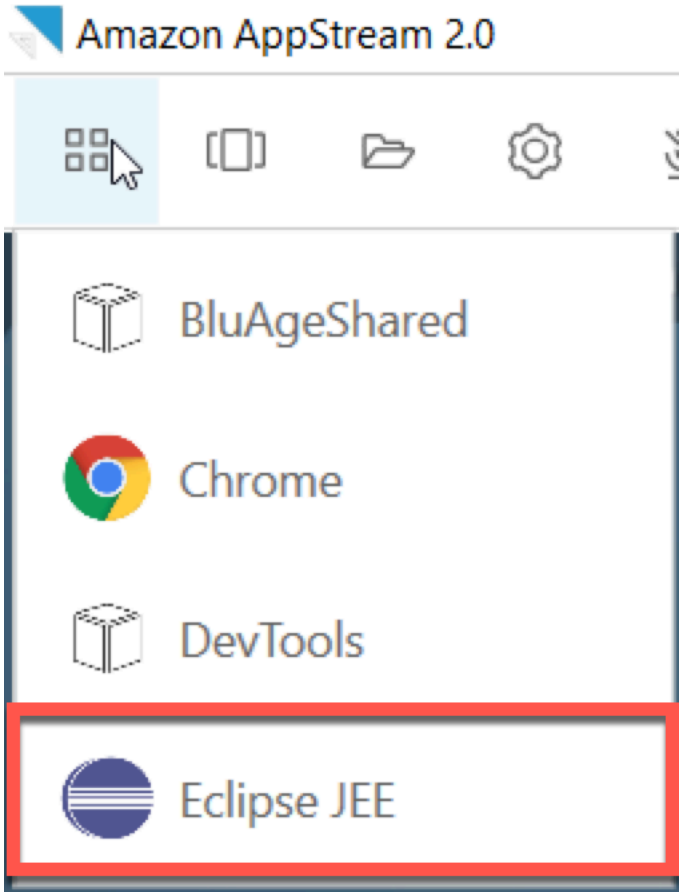
1. Choisissez Applications dans la barre de menu, puis Terminal.
2. Entrez la commande suivante :

```
~/_install-velocity-runtime.sh
```

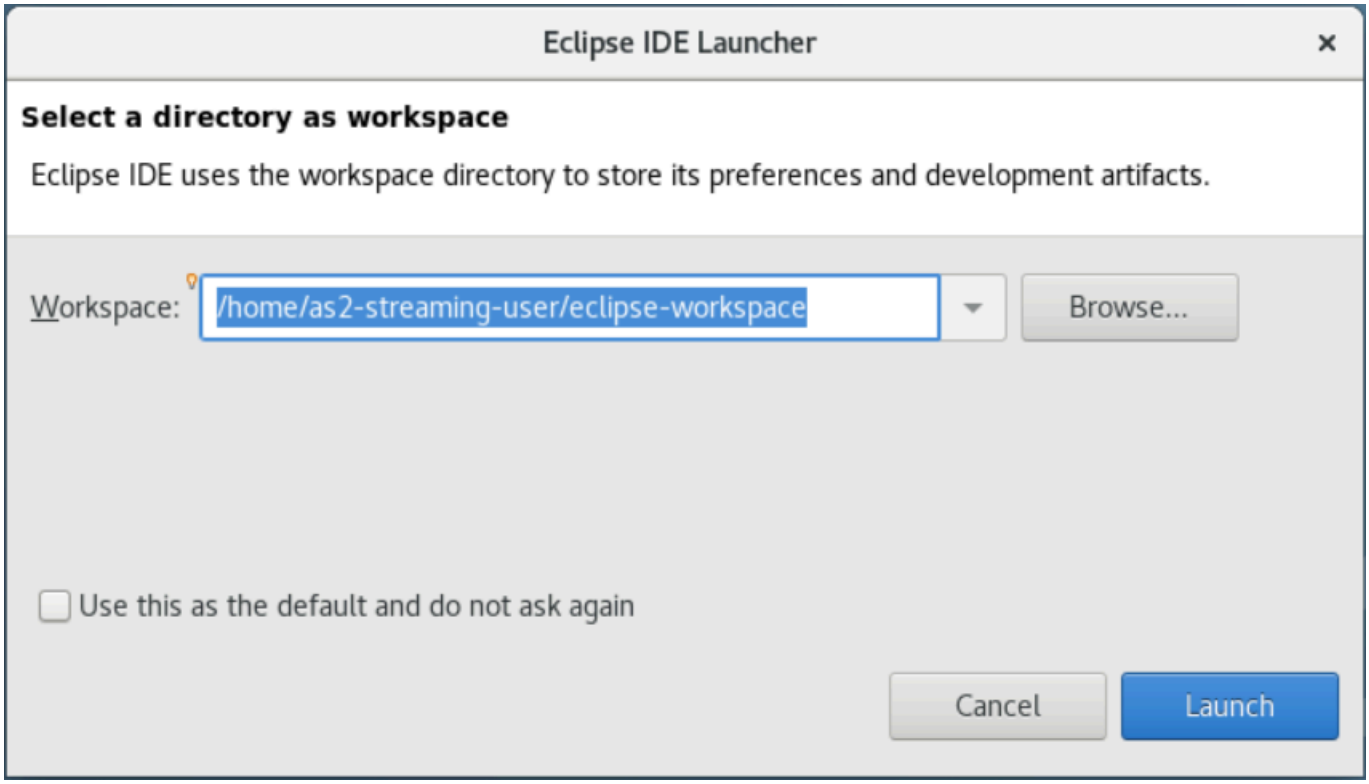
Étape 4 : démarrer l'IDE Eclipse

Au cours de cette étape, vous démarrez l'IDE Eclipse et choisissez un emplacement dans lequel vous souhaitez créer un espace de travail.

1. Dans la AppStream version 2.0, choisissez l'icône Lancer l'application dans la barre d'outils, puis choisissez Eclipse JEE.



2. Lorsque le lanceur s'ouvre, entrez l'emplacement où vous souhaitez créer votre espace de travail, puis choisissez Launch.



Vous pouvez éventuellement lancer Eclipse depuis la ligne de commande, comme suit :

```
~/eclipse &
```

Étape 5 : configurer un projet Maven

Au cours de cette étape, vous importez un projet Maven pour l'application de démonstration Planets.

1. Téléchargez le PlanetsDemofichier -pom.zip dans votre dossier d'accueil. Pour ce faire, vous pouvez utiliser la fonctionnalité « Mes fichiers » du client natif.
2. Utilisez l'outil de ligne de `unzip` commande pour extraire les fichiers.
3. Naviguez dans le dossier décompressé et ouvrez la racine `pom.xml` de votre projet dans un éditeur de texte.
4. Modifiez la `gapwalk.version` propriété afin qu'elle corresponde au moteur d'exécution AWS Blu Age installé.

   Si vous n'êtes pas sûr de la version installée, lancez la commande suivante dans un terminal :

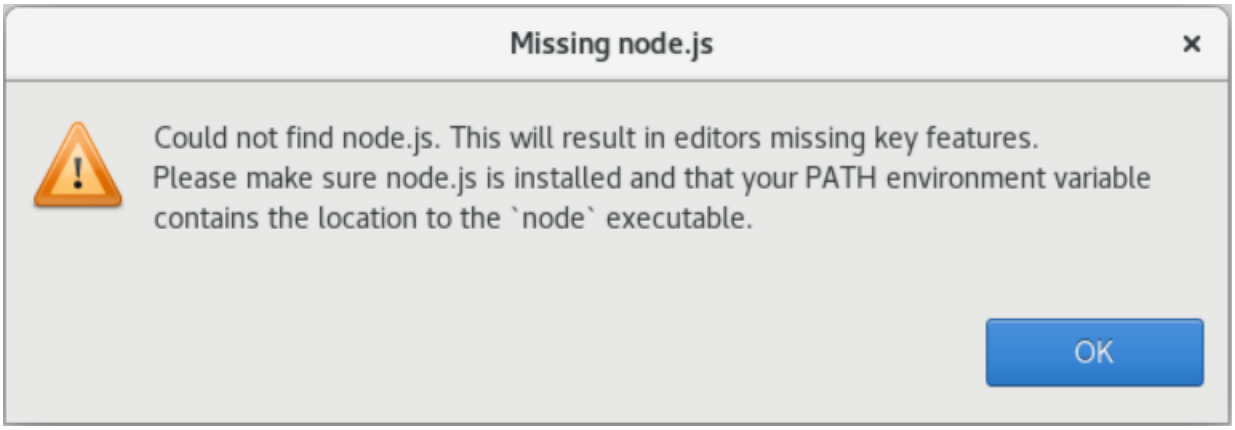
   ```
   cat ~/runtime-version.txt
   ```

Cette commande imprime la version d'exécution actuellement disponible, par exemple`3.1.0-b3257-dev`.

> **Note**
>
> N'incluez pas le `-dev` suffixe dans. `gapwalk.version` Par exemple, une valeur valide serait`<gapwalk.version>3.1.0-b3257</gapwalk.version>`.

5. Dans Eclipse, choisissez Fichier, puis Importer. Dans la fenêtre de dialogue Importer, développez Maven et choisissez Existing Maven Projects. Choisissez Suivant.
6. Dans Import Maven Projects, indiquez l'emplacement des fichiers extraits et choisissez Terminer.

   Vous pouvez ignorer la fenêtre contextuelle suivante en toute sécurité. Maven télécharge une copie locale de `node.js` pour créer la partie Angular (*-web) du projet :

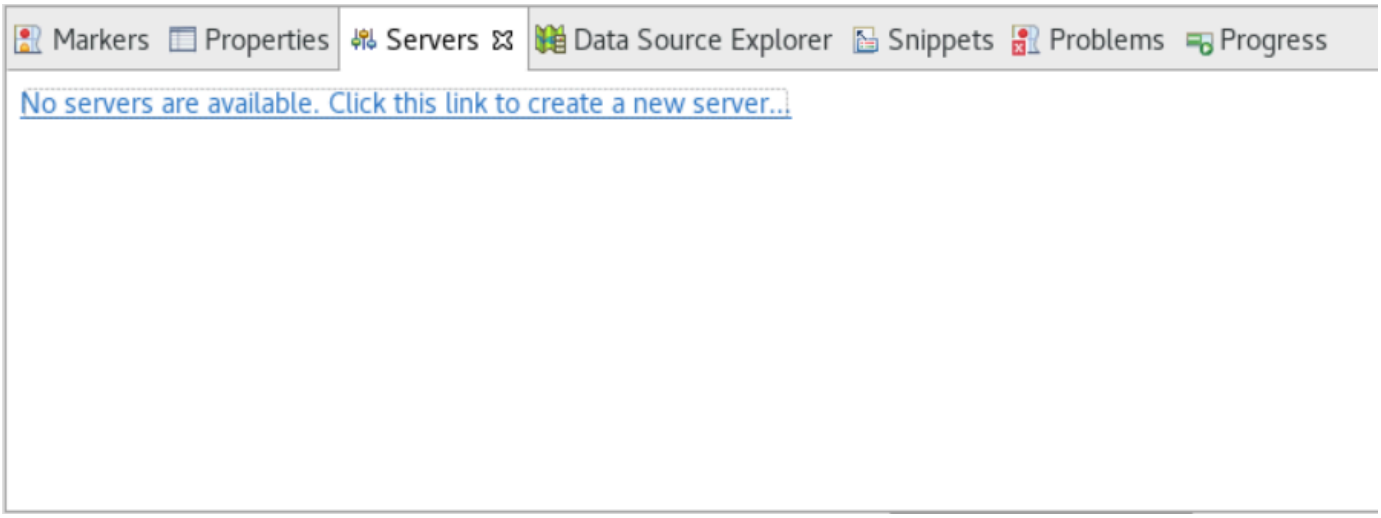
   

   Attendez la fin de la construction. Vous pouvez suivre la construction dans la vue Progression.
7. Dans Eclipse, sélectionnez le projet et choisissez Exécuter en tant que. Choisissez ensuite l'installation de Maven. Une fois l'installation de Maven réussie, le `war` fichier est créé sous. `PlanetsDemoPom/PlanetsDemo-web/target/PlanetsDemo-web-1.0.0.war`

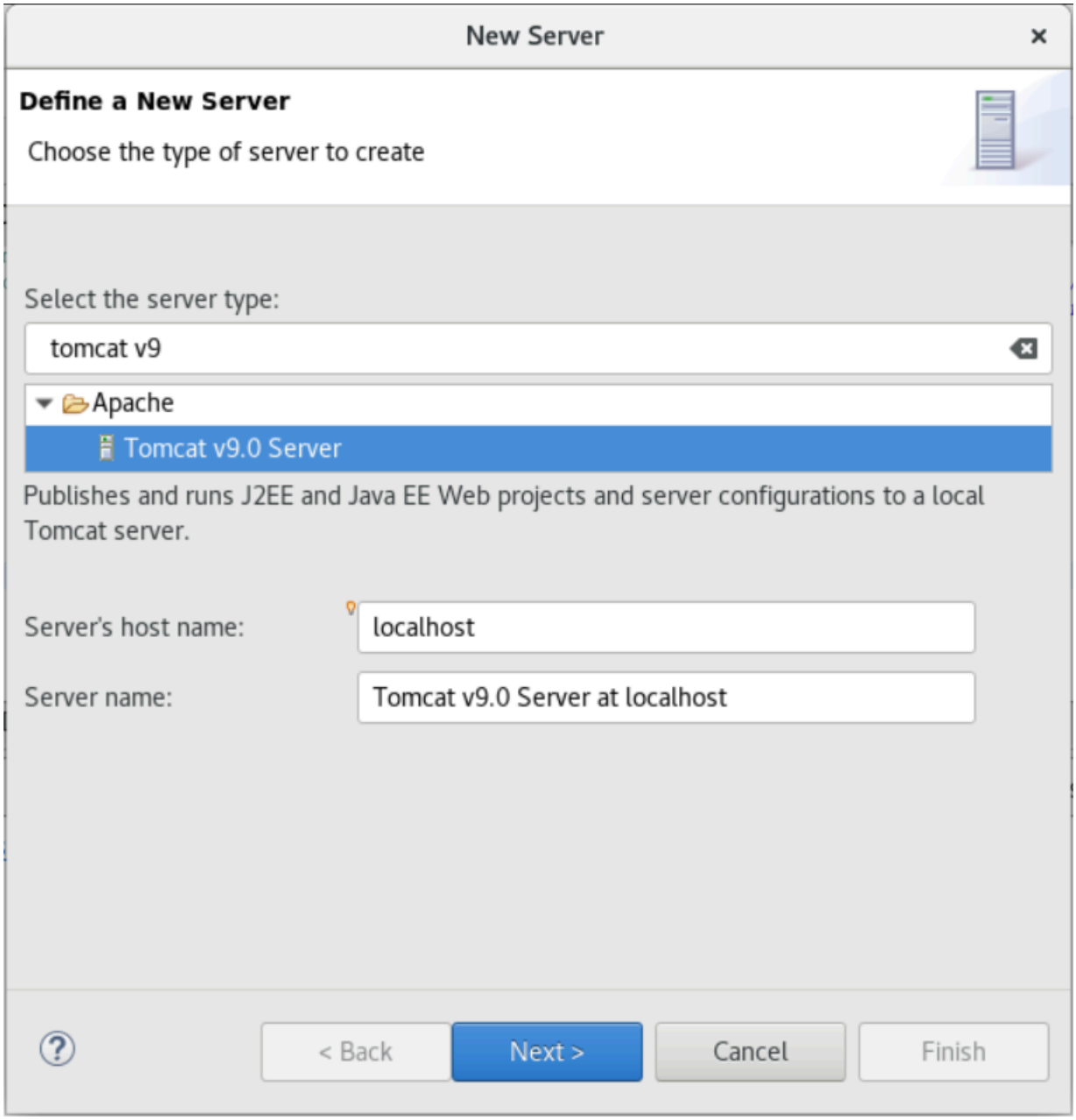
Étape 6 : Configuration d'un serveur Tomcat

Au cours de cette étape, vous configurez un serveur Tomcat sur lequel vous déployez et démarrez votre application compilée.

1. Dans Eclipse, choisissez Fenêtre > Afficher la vue > Serveurs pour afficher la vue Serveurs :

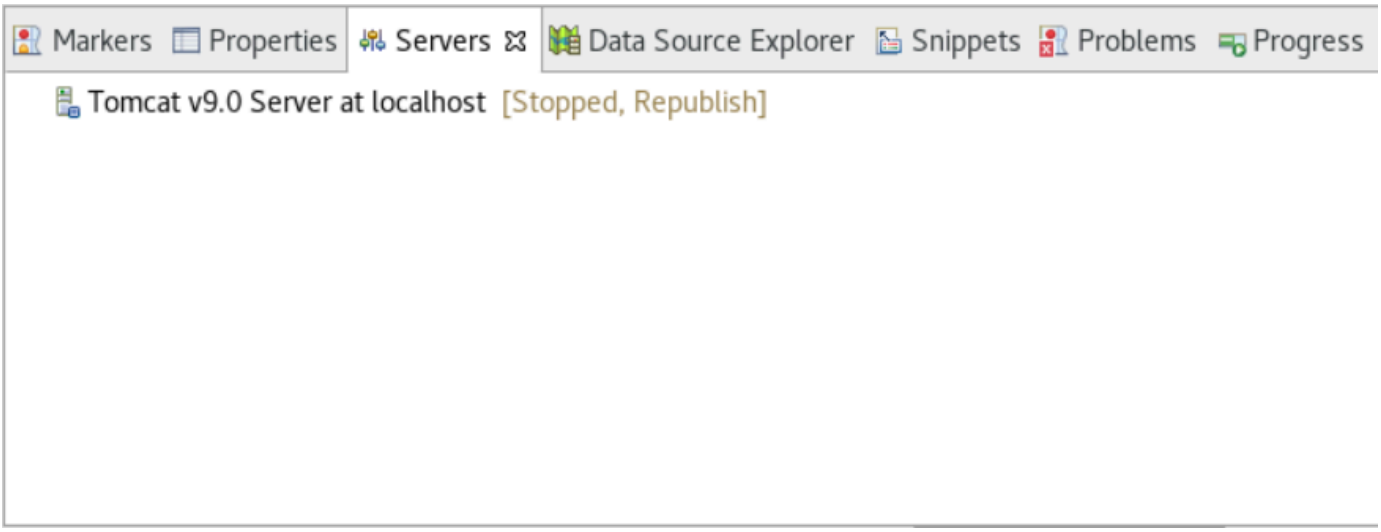


2. Choisissez Aucun serveur disponible. Cliquez sur ce lien pour créer un nouveau serveur... . L'assistant Nouveau serveur apparaît. Dans le champ Sélectionnez le type de serveur de l'assistant, entrez tomcat v9, puis choisissez Tomcat v9.0 Server. Ensuite, sélectionnez Suivant.



3. Choisissez Browse, puis choisissez le dossier Tomcat à la racine du dossier Home. Laissez le JRE à sa valeur par défaut et choisissez Terminer.

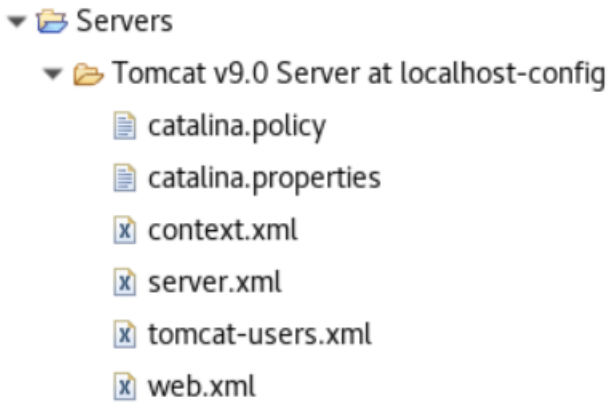
   Un projet Servers est créé dans l'espace de travail, et un serveur Tomcat v9.0 est désormais disponible dans la vue Serveurs. C'est ici que l'application compilée sera déployée et démarrée :



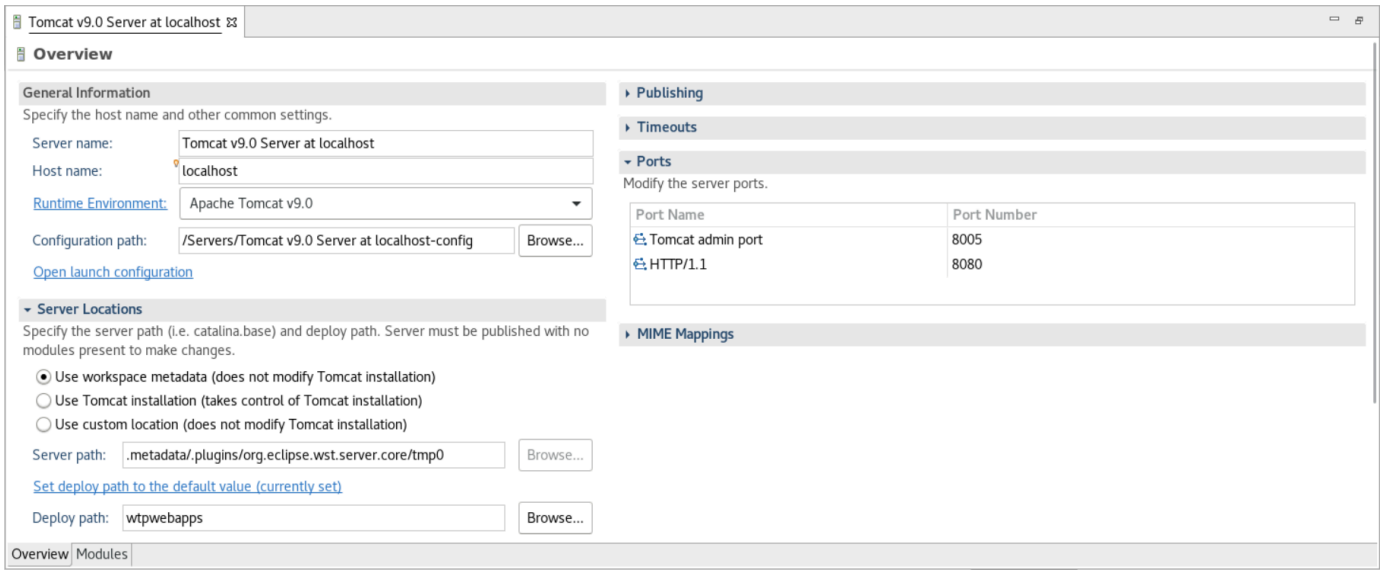
Étape 7 : Déployer sur Tomcat

Au cours de cette étape, vous déployez l'application de démonstration Planets sur le serveur Tomcat afin de pouvoir exécuter l'application.

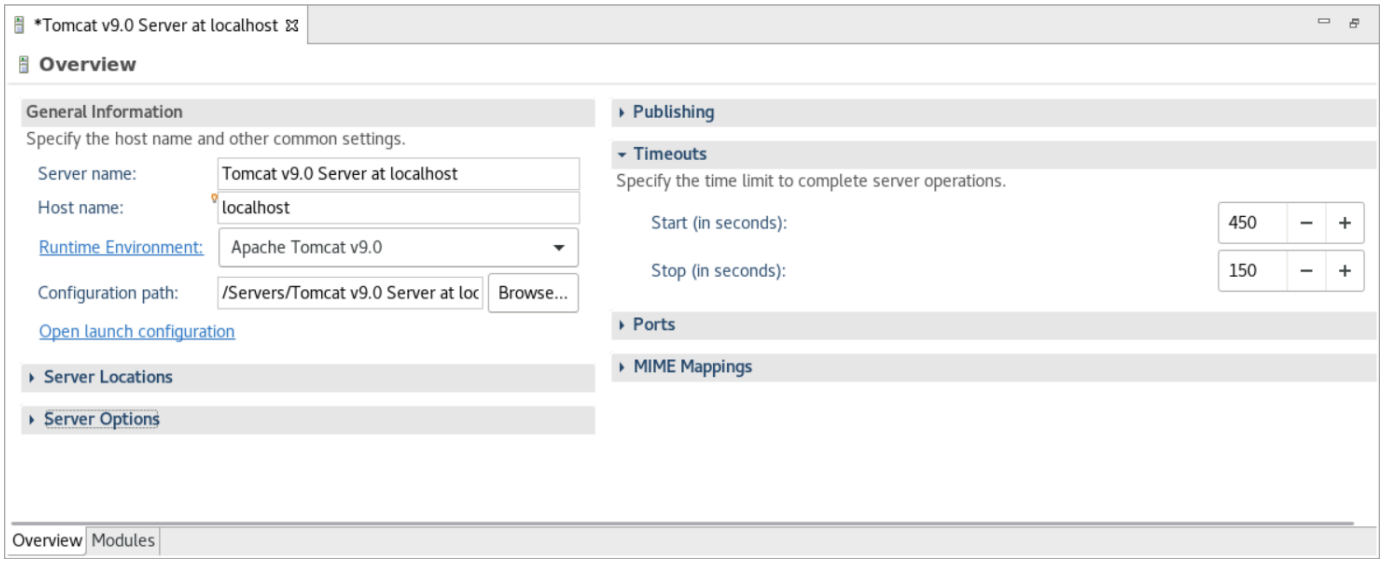
1. Sélectionnez le `PlanetsDemo-web` fichier et choisissez Exécuter sous > Installation de Maven. Sélectionnez `PlanetsDemo-web` à nouveau et choisissez Refresh pour vous assurer que le frontend compilé par npm est correctement compilé dans un fichier .war et remarqué par Eclipse.
2. Téléchargez le PlanetsDemofichier -runtime.zip sur l'instance, puis décompressez le fichier à un emplacement accessible. Cela garantit que l'application de démonstration peut accéder aux dossiers et fichiers de configuration dont elle a besoin.
3. Copiez le contenu de `PlanetsDemo-runtime/tomcat-config` dans le `Servers/Tomcat v9.0...` sous-dossier que vous avez créé pour votre serveur Tomcat :



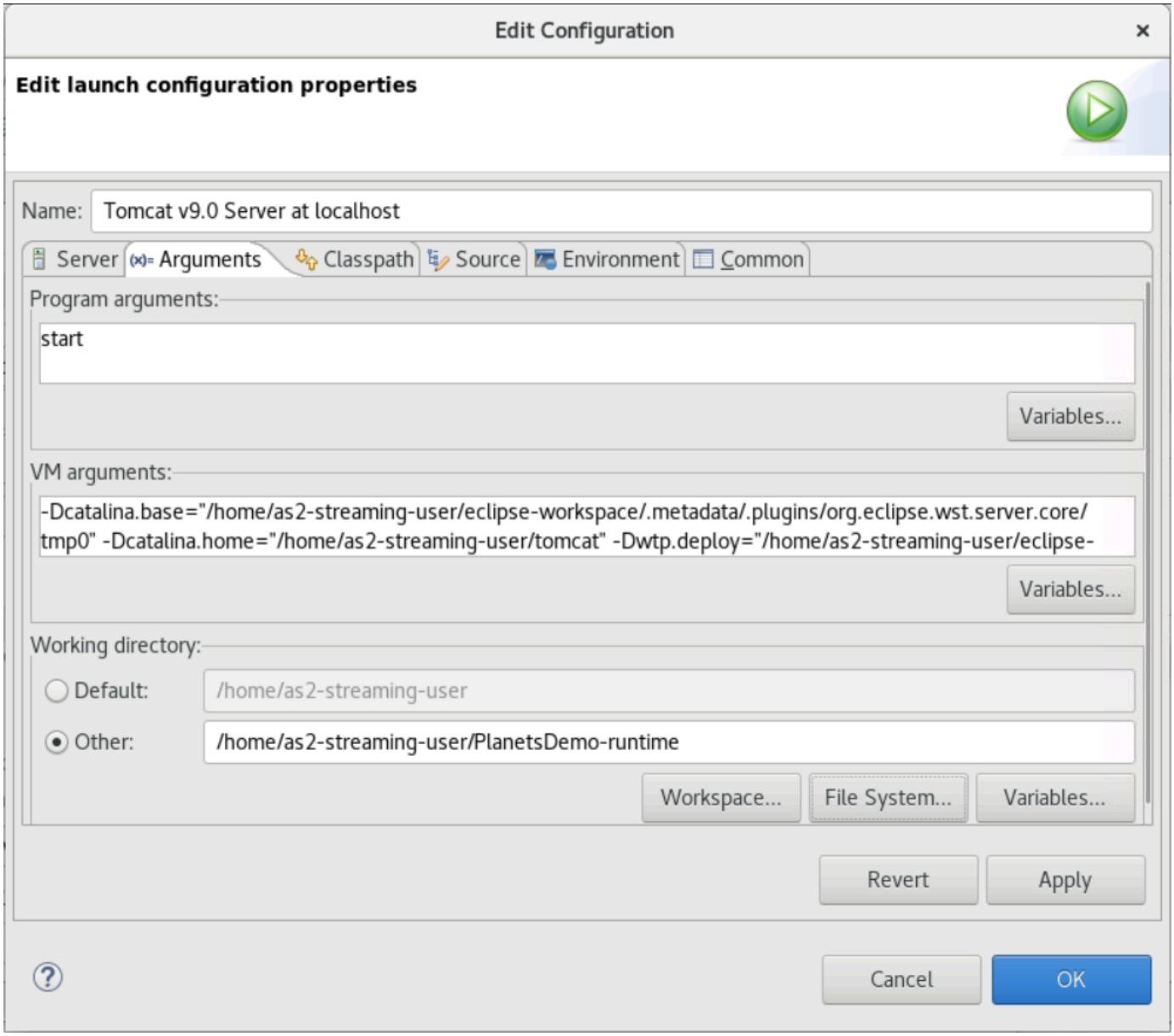
4. Ouvrez l'entrée `tomcat v9.0` du serveur dans la vue Serveurs. L'éditeur de propriétés du serveur apparaît :



5. Dans l'onglet Aperçu, augmentez les valeurs des délais d'attente à 450 secondes pour le démarrage et à 150 secondes pour l'arrêt, comme indiqué ici :



6. Choisissez Ouvrir la configuration de lancement. Un magicien apparaît. Dans l'assistant, accédez au dossier Arguments et, dans le répertoire de travail, sélectionnez Autre. Choisissez Système de fichiers, puis accédez au `PlanetsDemo-runtime` dossier décompressé précédemment. Ce dossier doit contenir un sous-dossier direct appelé config.

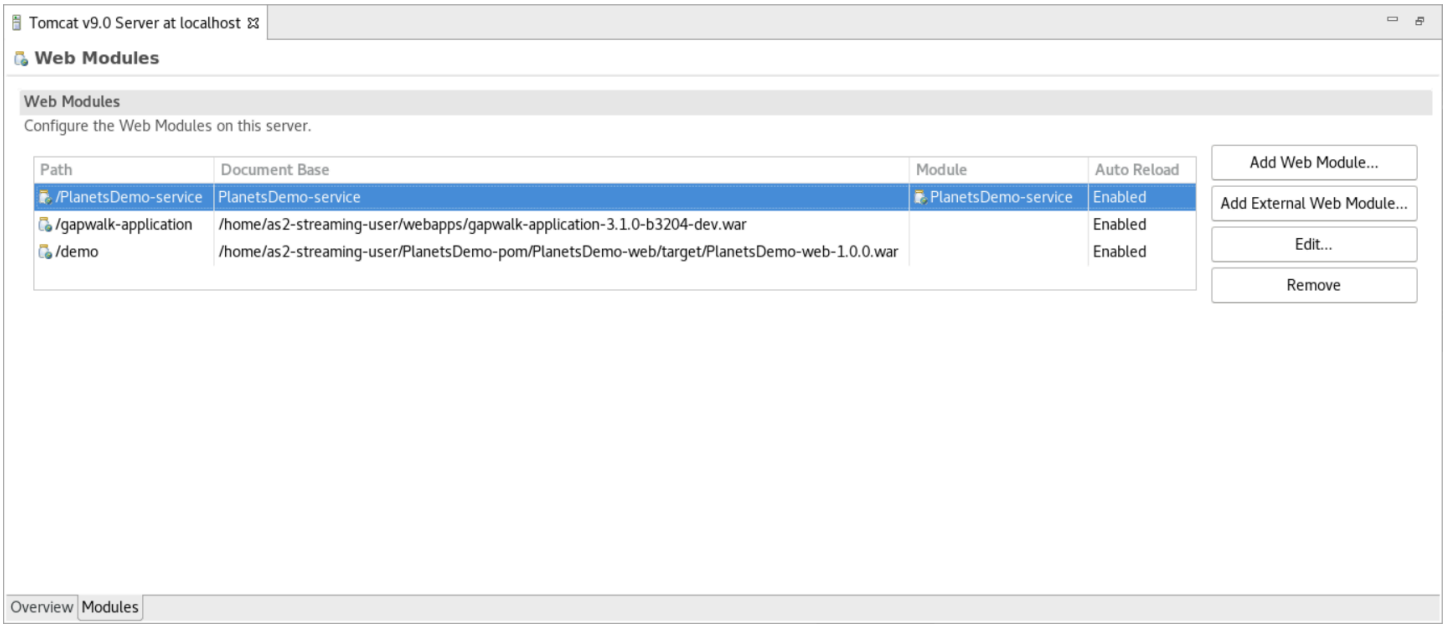


7. Choisissez l'onglet Modules de l'éditeur de propriétés du serveur et apportez les modifications suivantes :

   - Choisissez Ajouter un module Web et ajoutez`PlanetsDemo-service`.
   - Choisissez Ajouter un module Web externe. La fenêtre de dialogue Ajouter un module Web apparaît. Effectuez les modifications suivantes :
     - Dans la base de documents, choisissez Parcourir et naviguez vers `~/webapps/gapwalk-application...war`
     - Dans Path, entrez`/gapwalk-application`.
   - Choisissez OK.

- Choisissez à nouveau Ajouter un module Web externe et apportez les modifications suivantes :
  - Pour Document base, entrez le chemin d'accès au frontend .war (in) `PlanetsDemo-web/target`
  - Pour Path, entrez `/demo`
- Sélectionnez OK
- Enregistrez les modifications de l'éditeur (Ctrl + S).

Le contenu de l'éditeur doit maintenant être similaire à ce qui suit.



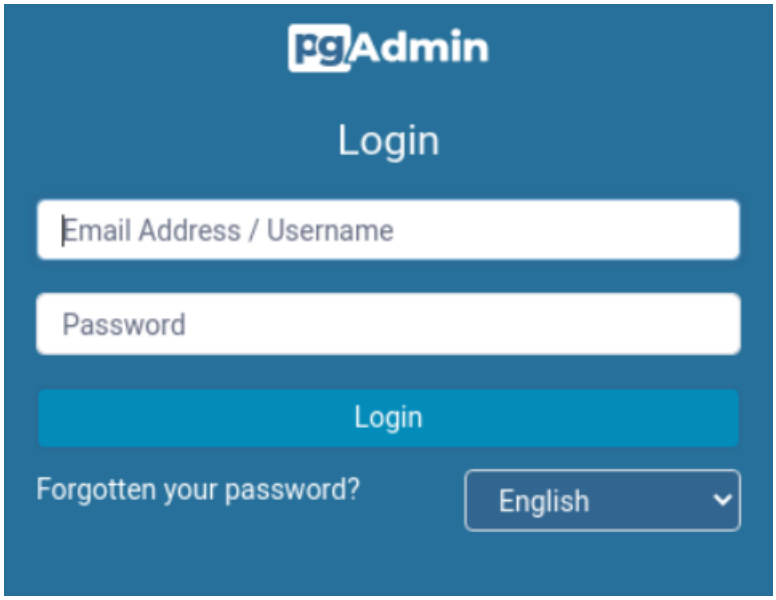
Étape 8 : Création de la base de données JICS

Au cours de cette étape, vous vous connectez à la base de données que vous avez créée dansÉtape 1 : Créer une base de données.

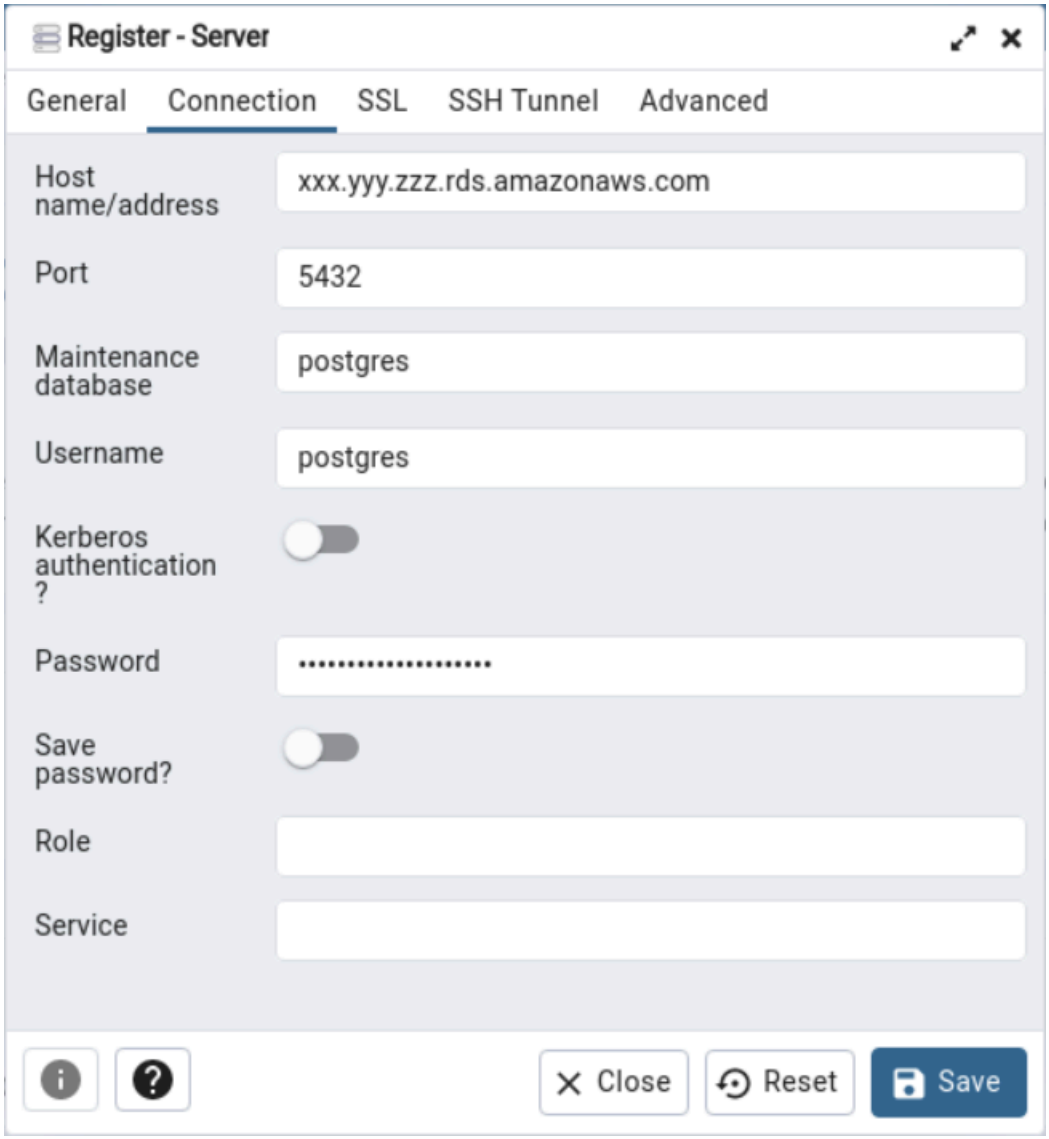
1. À partir de l'instance AppStream 2.0, exécutez la commande suivante dans un terminal pour le lancer `pgAdmin` :

   ```
   ./pgadmin-start.sh
   ```

2. Choisissez une adresse e-mail et un mot de passe comme identifiants de connexion. Prenez note de l'URL fournie (généralement http://127.0.0.1:5050). Lancez Google Chrome dans l'instance, copiez-collez l'URL dans le navigateur et connectez-vous à l'aide de vos identifiants.



3. Une fois connecté, choisissez Ajouter un nouveau serveur et entrez les informations de connexion à la base de données créée précédemment comme suit.

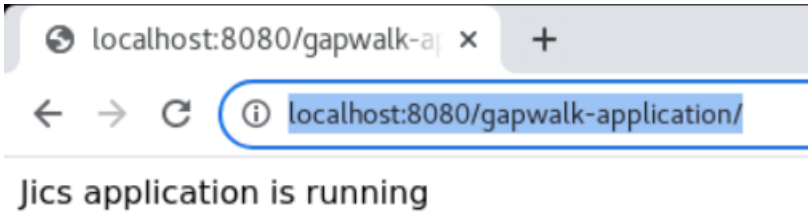


4. Lorsque vous vous connectez au serveur de base de données, utilisez Objet > Créer > Base de données et créez une nouvelle base de données nommée jics.
5. Modifiez les informations de connexion à la base de données utilisées par l'application de démonstration. Ces informations sont définies dans`PlanetsDemo-runtime/config/application-main.yml`. Recherchez l'`jicsDs`entrée. Pour récupérer les valeurs pour `username` et`password`, dans la console Amazon RDS, accédez à la base de données. Dans l'onglet Configuration, sous Master Credentials ARN, choisissez Gérer dans Secrets Manager. Ensuite, dans la console Secrets Manager, dans le secret, choisissez Retrieve secret value.

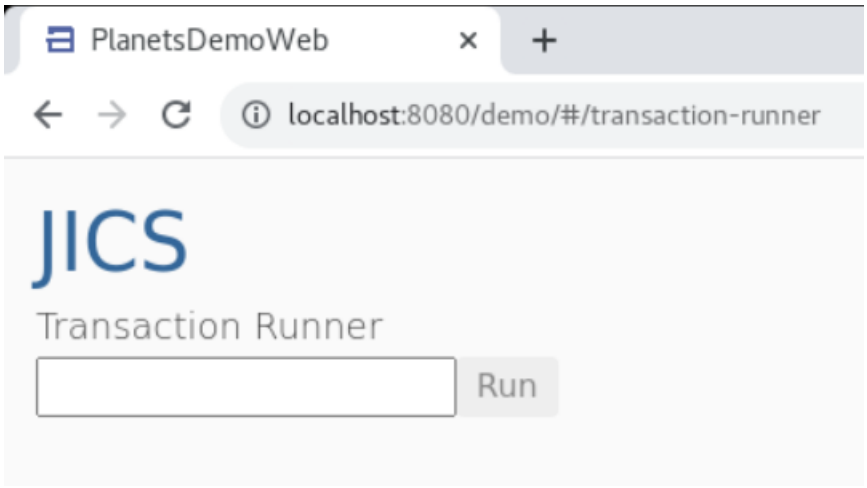
Étape 9 : démarrer et tester l'application

Au cours de cette étape, vous lancez le serveur Tomcat et l'application de démonstration afin de pouvoir le tester.

1. Pour démarrer le serveur Tomcat et les applications précédemment déployées, sélectionnez l'entrée du serveur dans la vue Serveurs et choisissez Démarrer. Une console affiche les journaux de démarrage.
2. Vérifiez l'état du serveur dans la vue Serveurs ou attendez le message de démarrage du serveur en [xxx] millisecondes dans la console. Après le démarrage du serveur, vérifiez que l'application gapwalk est correctement déployée. Pour ce faire, accédez à l'URL http://localhost:8080/gapwalk-application dans un navigateur Google Chrome. Vous devriez voir ce qui suit.

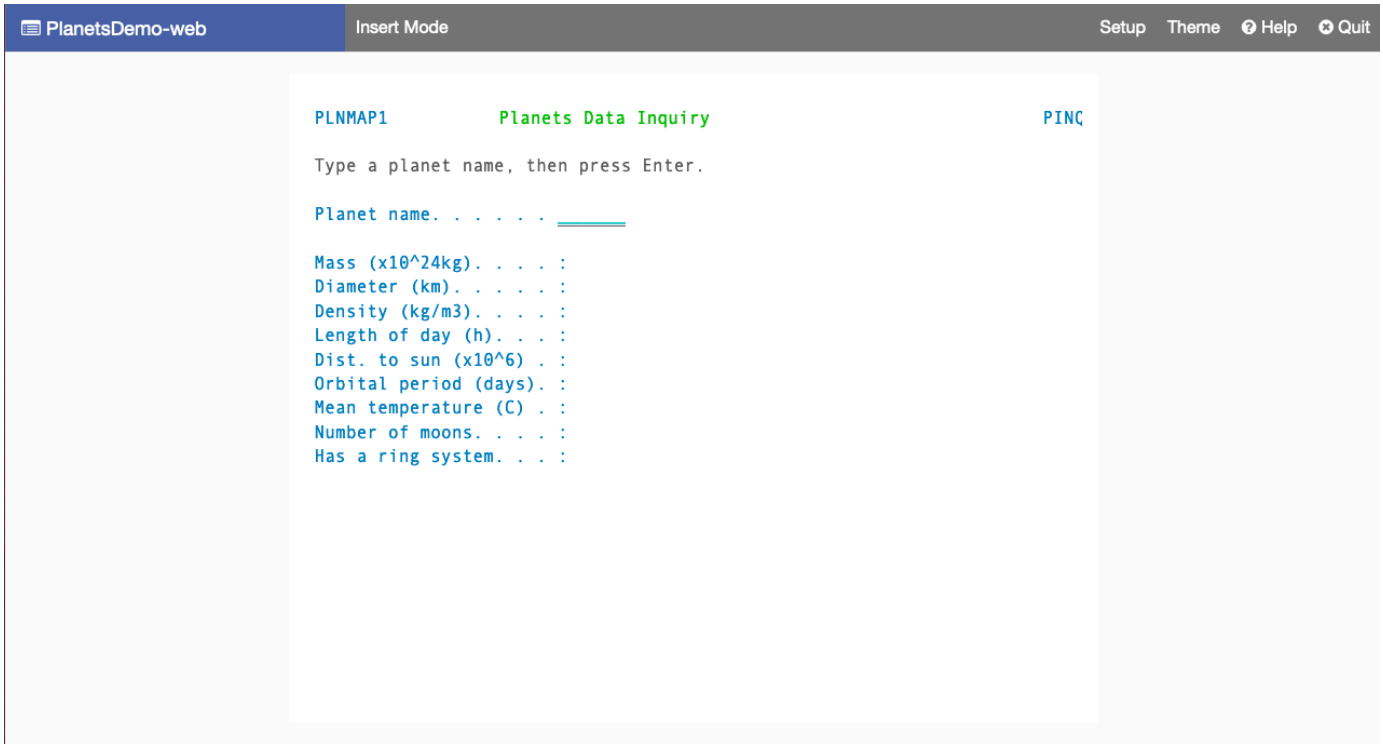


3. Accédez à l'interface de l'application déployée depuis Google Chrome à l'adresse http://localhost:8080/demo. La page Transaction Launcher suivante devrait apparaître.

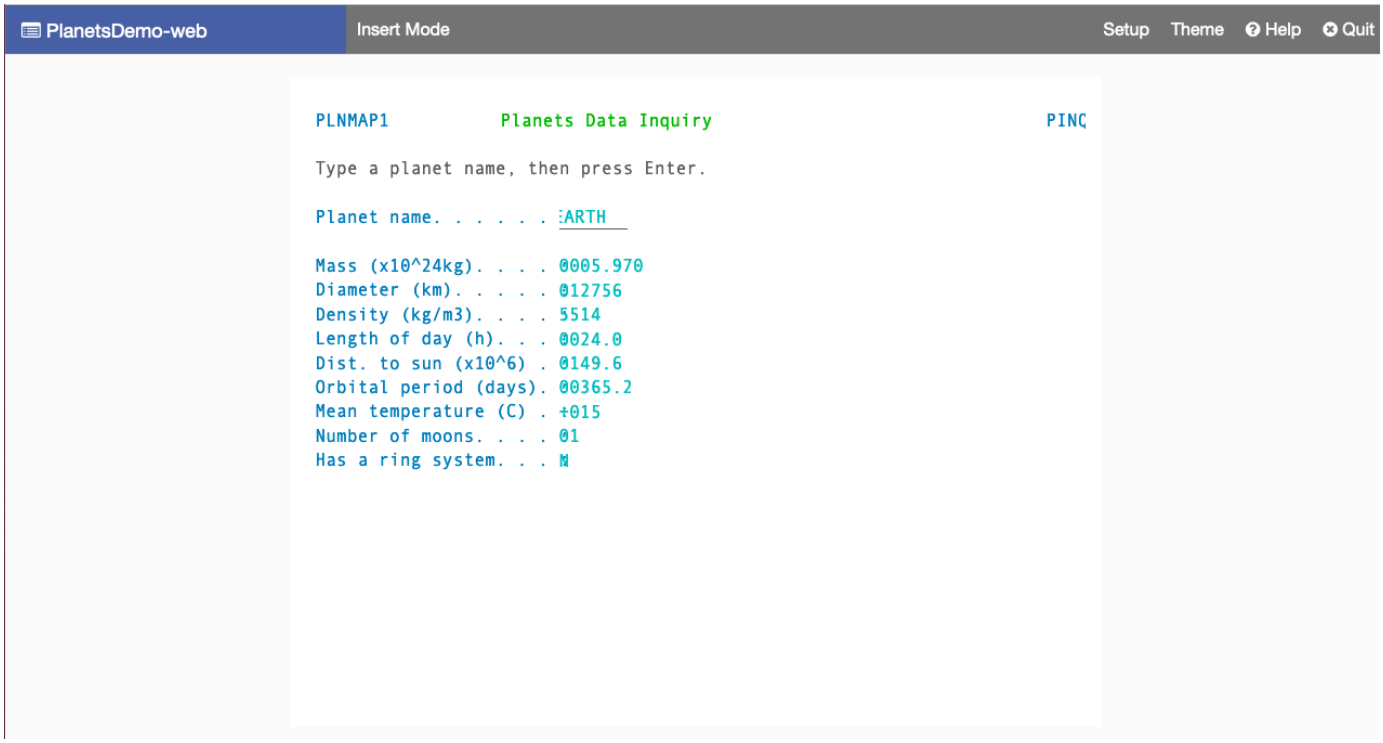


4. Pour démarrer la transaction de l'application, entrez `PINQ` dans le champ de saisie et choisissez Exécuter (ou appuyez sur Entrée).

   L'écran de l'application de démonstration devrait apparaître.



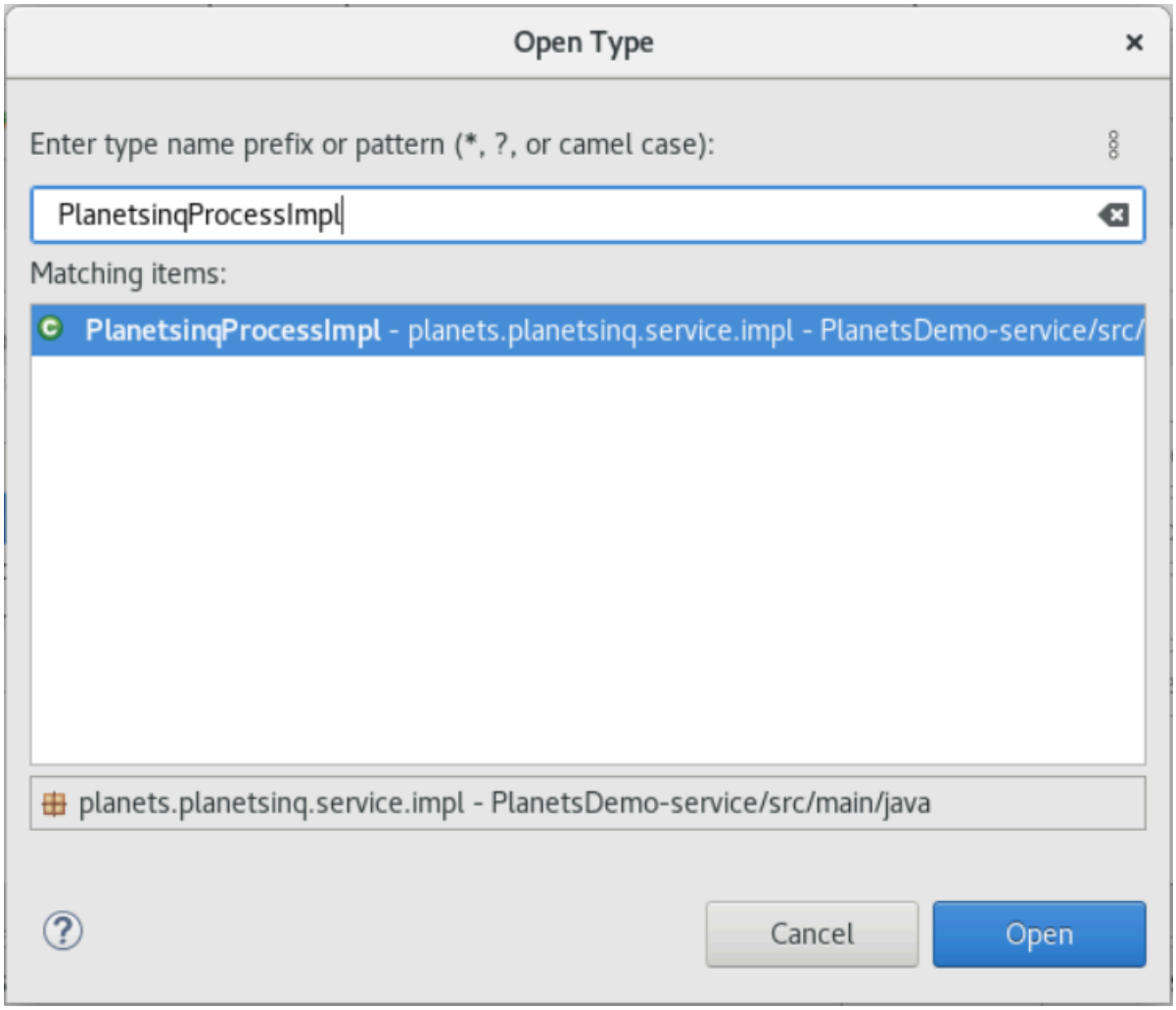
5. Tapez le nom d'une planète dans le champ correspondant et appuyez sur Entrée.



Étape 10 : Déboguer l'application

Au cours de cette étape, vous testez à l'aide des fonctionnalités de débogage standard d'Eclipse. Ces fonctionnalités sont disponibles lorsque vous travaillez sur une application modernisée.

1. Pour ouvrir la classe de service principale, appuyez sur Ctrl + Shift + T. Puis entrez. `PlanetsinqProcessImpl`



2. Accédez à la `searchPlanet` méthode et placez-y un point d'arrêt.
3. Sélectionnez le nom du serveur, puis sélectionnez Redémarrer dans le débogage.
4. Répétez les étapes précédentes. C'est-à-dire, accédez à l'application, entrez le nom d'une planète et appuyez sur Entrée.

   Eclipse arrêtera l'application dans la `searchPlanet` méthode. Vous pouvez maintenant l'examiner.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin de ne pas avoir à payer de frais supplémentaires. Procédez comme suit :

- Si l'application Planets est toujours en cours d'exécution, arrêtez-la.
- Supprimez la base de données que vous avez créée dansÉtape 1 : Créer une base de données. Pour plus d'informations, consultez Suppression d'une instance de base de données.

Replateforme des applications avec Micro Focus

Cette section décrit chaque étape du processus de replateforme. Il décrit toutes les tâches et inclut des informations sur la configuration et le fonctionnement du moteur d'exécution AWS Mainframe Modernization sur Amazon EC2.

Rubriques

Configuration de Micro Focus Runtime (sur Amazon EC2)

AWS Mainframe Modernization fournit plusieurs Amazon Machine Images (AMI) qui incluent des produits sous licence Micro Focus. Ces AMI vous permettent de provisionner rapidement des instances Amazon Elastic Compute Cloud (Amazon EC2) afin de prendre en charge les environnements Micro Focus que vous contrôlez et gérez. Cette rubrique décrit les étapes nécessaires pour accéder à ces AMI et les lancer. L'utilisation de ces AMI est entièrement facultative et elles ne sont pas obligatoires pour suivre les didacticiels de ce guide de l'utilisateur.

Rubriques

- Création du AWS Identity and Access Management rôle
- Accordez à License Manager les autorisations requises
- Abonnez-vous aux Amazon Machine Images
- Lancer une AWS instance Micro Focus de modernisation du mainframe
- Sous-réseau ou VPC sans accès Internet
- Résolution des problèmes de licence

Prérequis

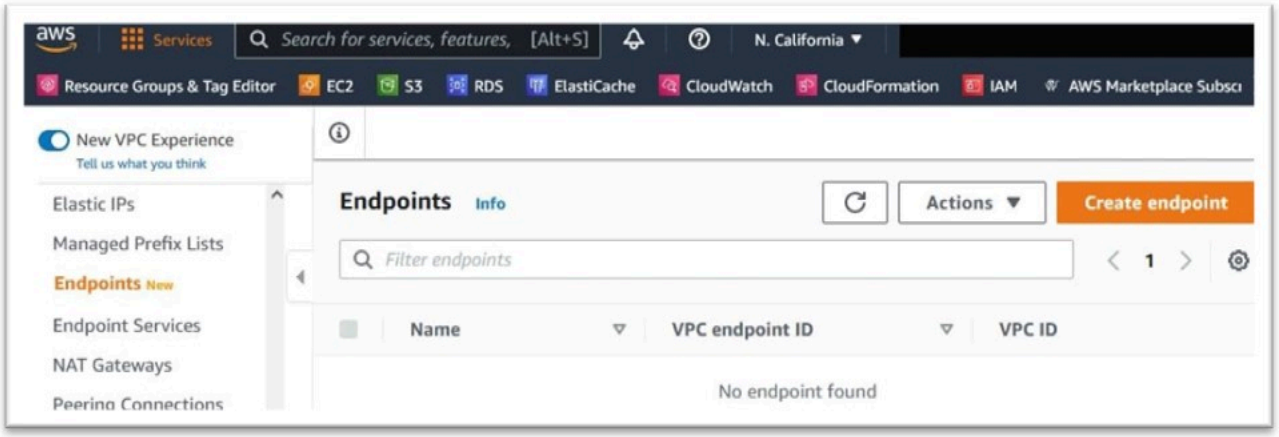
Assurez-vous de remplir les conditions préalables suivantes.

- Accès administrateur au compte sur lequel les instances Amazon EC2 seront créées.
- Identifiez l' Région AWS endroit où les instances Amazon EC2 seront créées et vérifiez que le service de modernisation du AWS mainframe est disponible. Consultez la section AWS Services par région. Assurez-vous de choisir une région dans laquelle le service est disponible.
- Identifiez l'Amazon Virtual Private Cloud (Amazon VPC) dans lequel les instances Amazon EC2 seront créées.

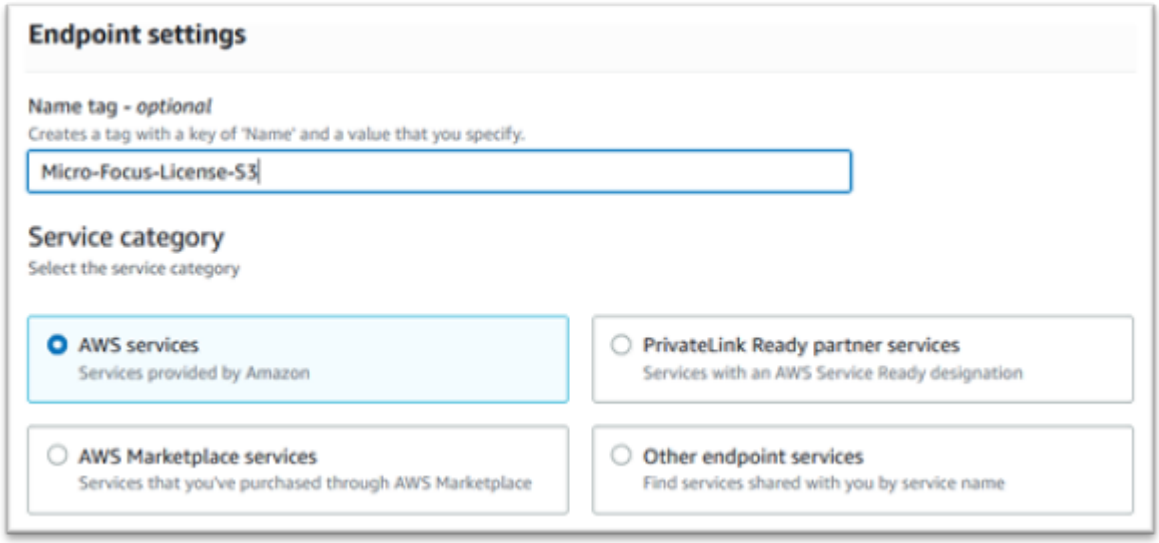
Création du point de terminaison Amazon VPC pour Amazon S3

Dans cette section, vous allez créer un point de terminaison Amazon VPC à utiliser par Amazon S3.

1. Accédez à Amazon VPC dans le. AWS Management Console
2. Dans le panneau de navigation, choisissez Points de terminaison.
3. Choisissez Créer un point de terminaison.

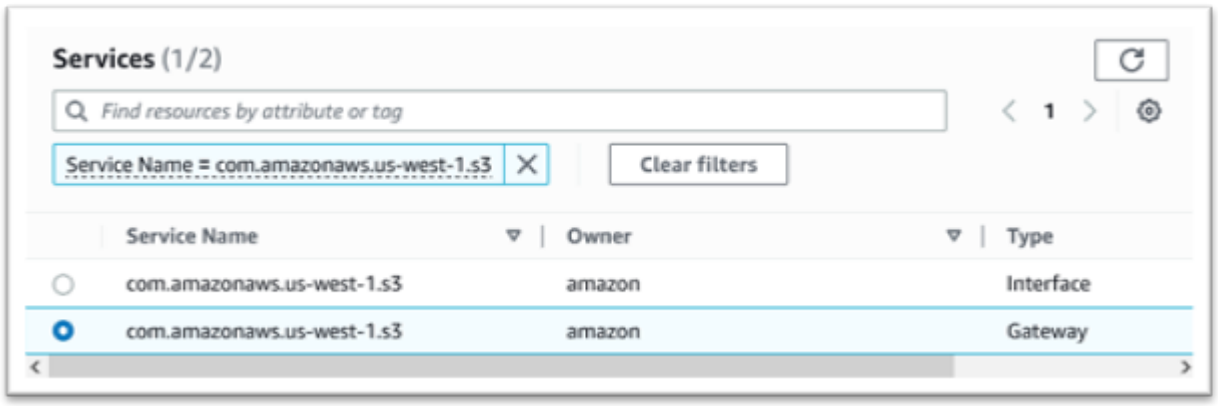


4. Entrez un tag nominatif significatif, par exemple : « Micro-Focus-License-S3 ».
5. Choisissez AWS Services comme catégorie de service.

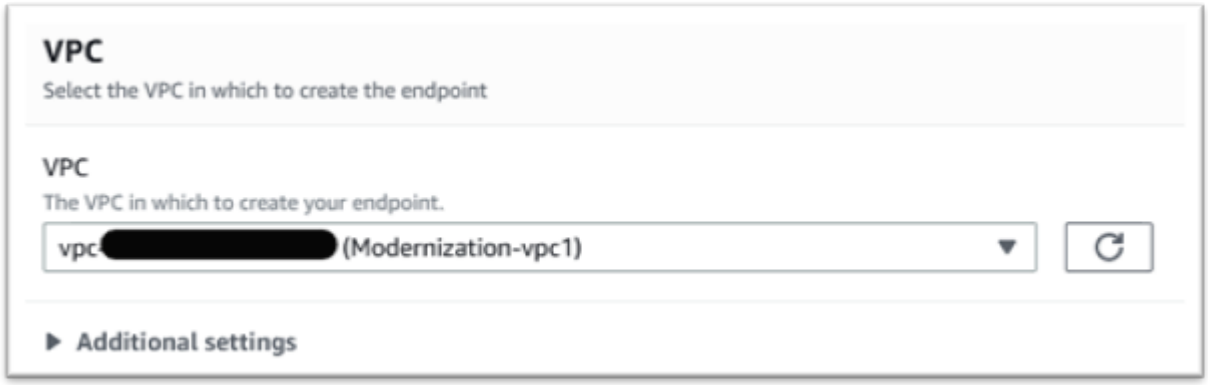


6. Sous Services, recherchez le service Amazon S3 Gateway : com.amazonaws. [région] .s3.

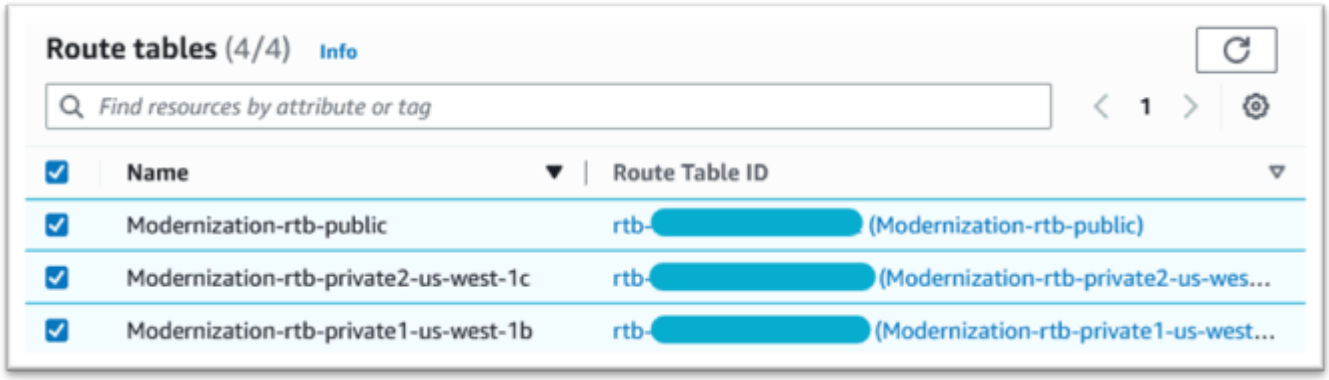
   Car `us-west-1` ce serait :`com.amazonaws.us-west-1.s3`.
7. Choisissez le service Gateway.



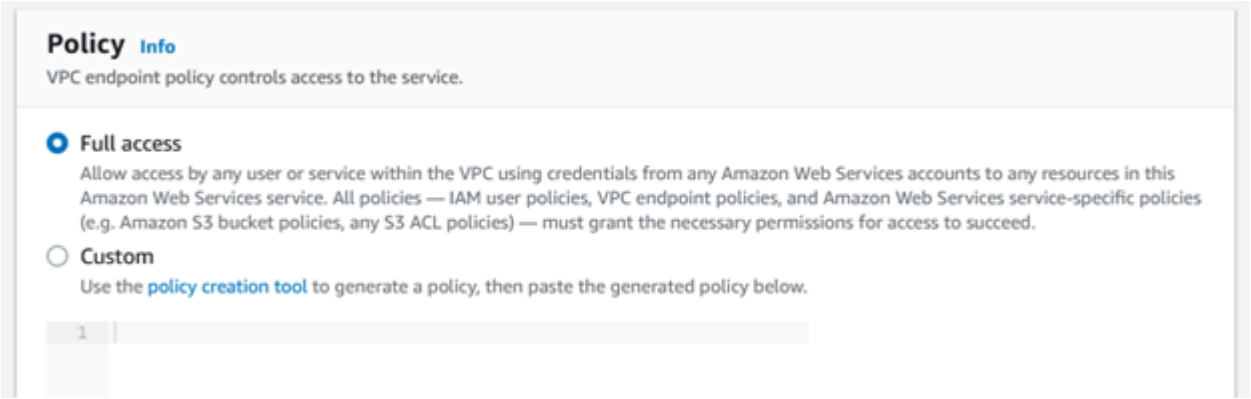
8. Pour VPC, choisissez le VPC que vous utiliserez.



9. Choisissez toutes les tables de routage pour le VPC.



10. Sous Politique, sélectionnez Accès complet.



11. Choisissez Créer un point de terminaison.

Demander la mise à jour de la liste d'autorisation pour le compte

Contactez votre AWS représentant pour que votre compte soit autorisé à figurer sur la liste des AMI de modernisation du AWS mainframe. Veuillez fournir les informations suivantes :

- L' Compte AWS identifiant.

- L' Région AWS endroit où le point de terminaison Amazon VPC a été créé.
- L'identifiant du point de terminaison Amazon VPC Amazon S3 créé dans. Création du point de terminaison Amazon VPC pour Amazon S3 Il s'agit de l'`vpce-xxxxxxxxxxxxxxxxx`identifiant du fichier com.amazonaws. Point de terminaison de la passerelle [region] .s3.
- Le nombre de licences requises pour toutes les instances AMI Amazon EC2 de Micro Focus Enterprise Suite.

  Une licence est requise par cœur de processeur (pour 2 vCPU pour la plupart des instances Amazon EC2).

  Pour plus d'informations, consultez la section Optimisation des options du processeur.

  Le numéro demandé peut être ajusté à l'avenir par AWS.

> **Note**
>
> Le AWS représentant doit ouvrir le ticket d'assistance pour la demande de liste d'autorisation. Il ne peut pas être demandé directement et le traitement de la demande peut prendre plusieurs jours.

Création du AWS Identity and Access Management rôle

Créez une AWS Identity and Access Management politique et un rôle à utiliser par les instances Amazon EC2 de modernisation du AWS mainframe. La création du rôle via la console IAM créera un profil d'instance associé du même nom. L'attribution de ce profil d'instance aux instances Amazon EC2 permet d'attribuer des licences Micro Focus. Pour plus d'informations sur les profils d'instance, consultez Utilisation d'un rôle IAM pour accorder des autorisations aux applications exécutées sur des instances Amazon EC2.

Création d'une stratégie IAM

Une politique IAM est d'abord créée, puis attachée au rôle.

1. Naviguez vers AWS Identity and Access Management dans le AWS Management Console.
2. Choisissez Politiques, puis Créer une politique.

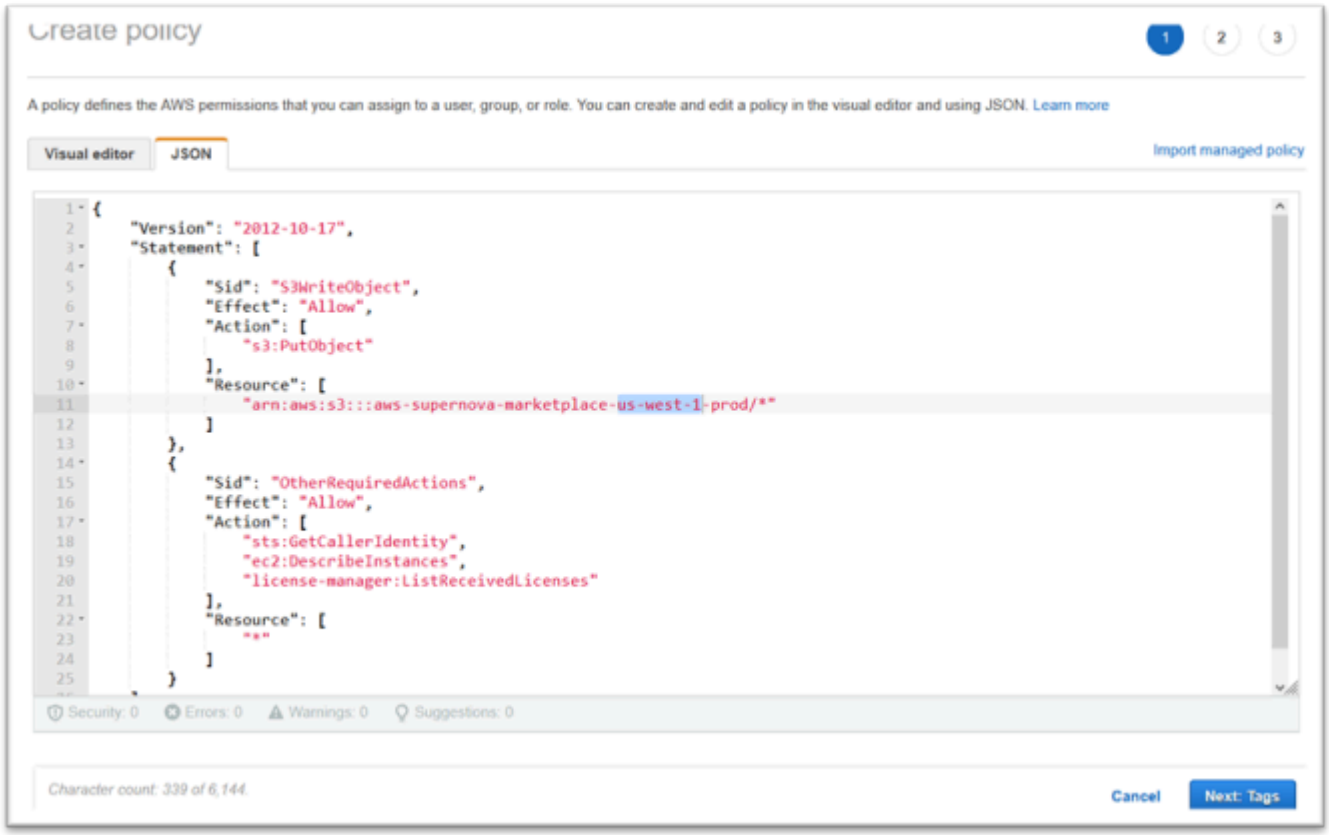
3. Sélectionnez l'onglet JSON.

4. Remplacez `us-west-1` le JSON suivant par celui Région AWS où le point de terminaison Amazon S3 a été défini, puis copiez-collez le JSON dans l'éditeur de politiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3WriteObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-supernova-marketplace-us-west-1-prod/*"
            ]
        },
        {
            "Sid": "OtherRequiredActions",
```

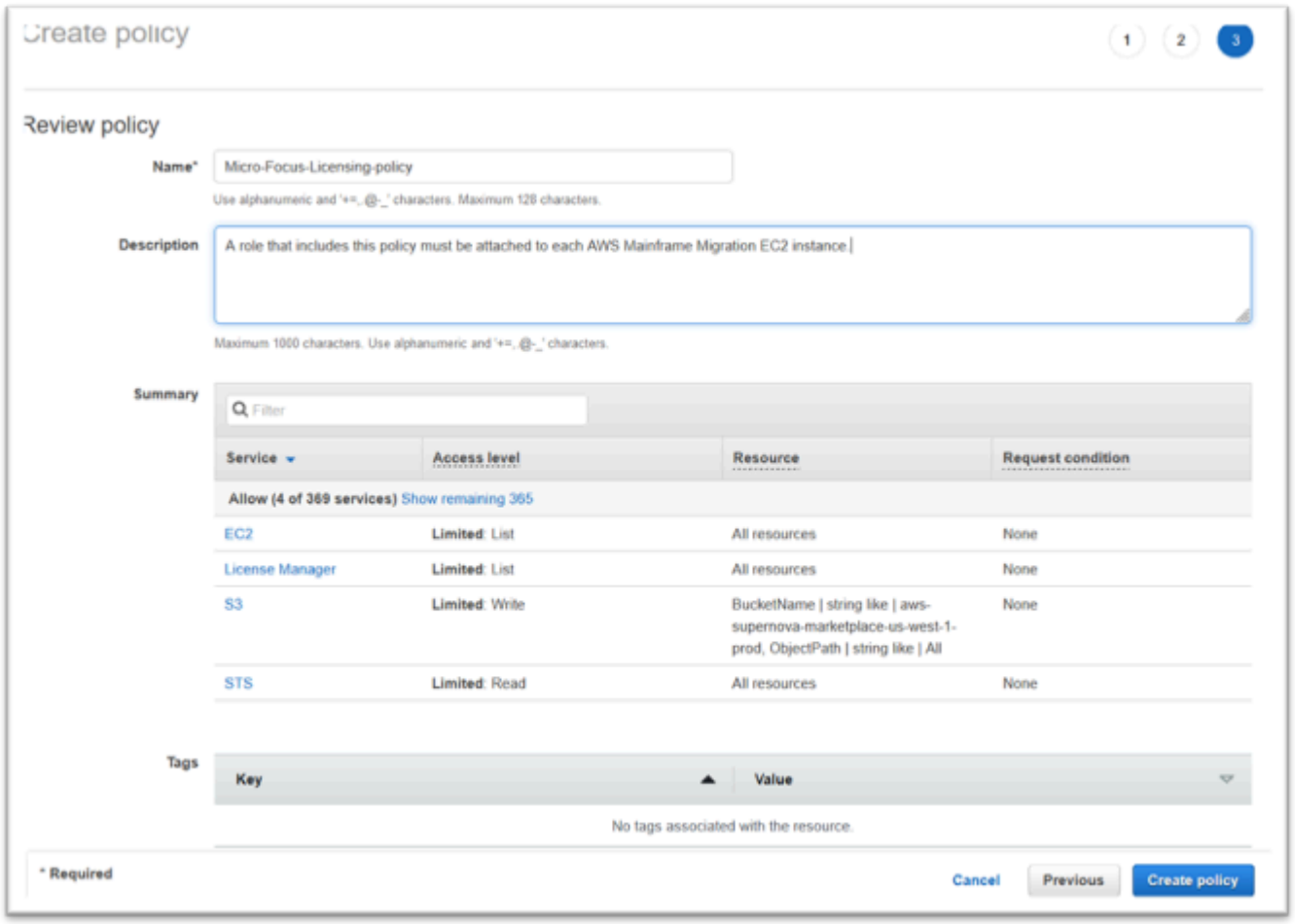
```
            "Effect": "Allow",
            "Action": [
                "sts:GetCallerIdentity",
                "ec2:DescribeInstances",
                "license-manager:ListReceivedLicenses"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

> **Note**
>
> Les actions situées sous le Sid `OtherRequiredActions` ne prennent pas en charge les autorisations au niveau des ressources et doivent être spécifiées `*` dans l'élément ressource.



5. Choisissez Suivant : Balises.

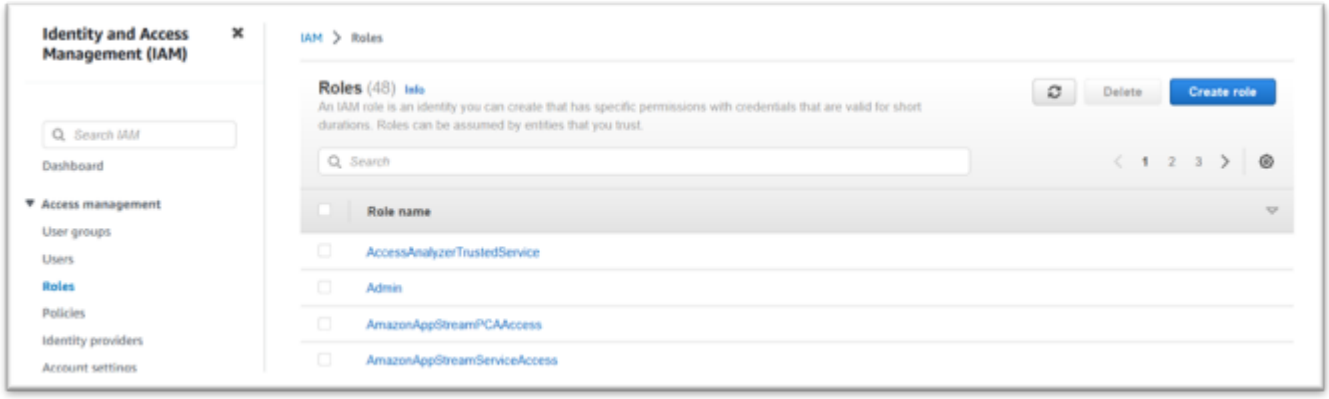
6. Entrez éventuellement des balises, puis choisissez Next : Review.
7. Entrez un nom pour la politique, par exemple « Micro-Focus-Licensing-Policy ». Entrez éventuellement une description, par exemple « Un rôle incluant cette politique doit être attaché à chaque instance Amazon EC2 de modernisation du AWS mainframe ».



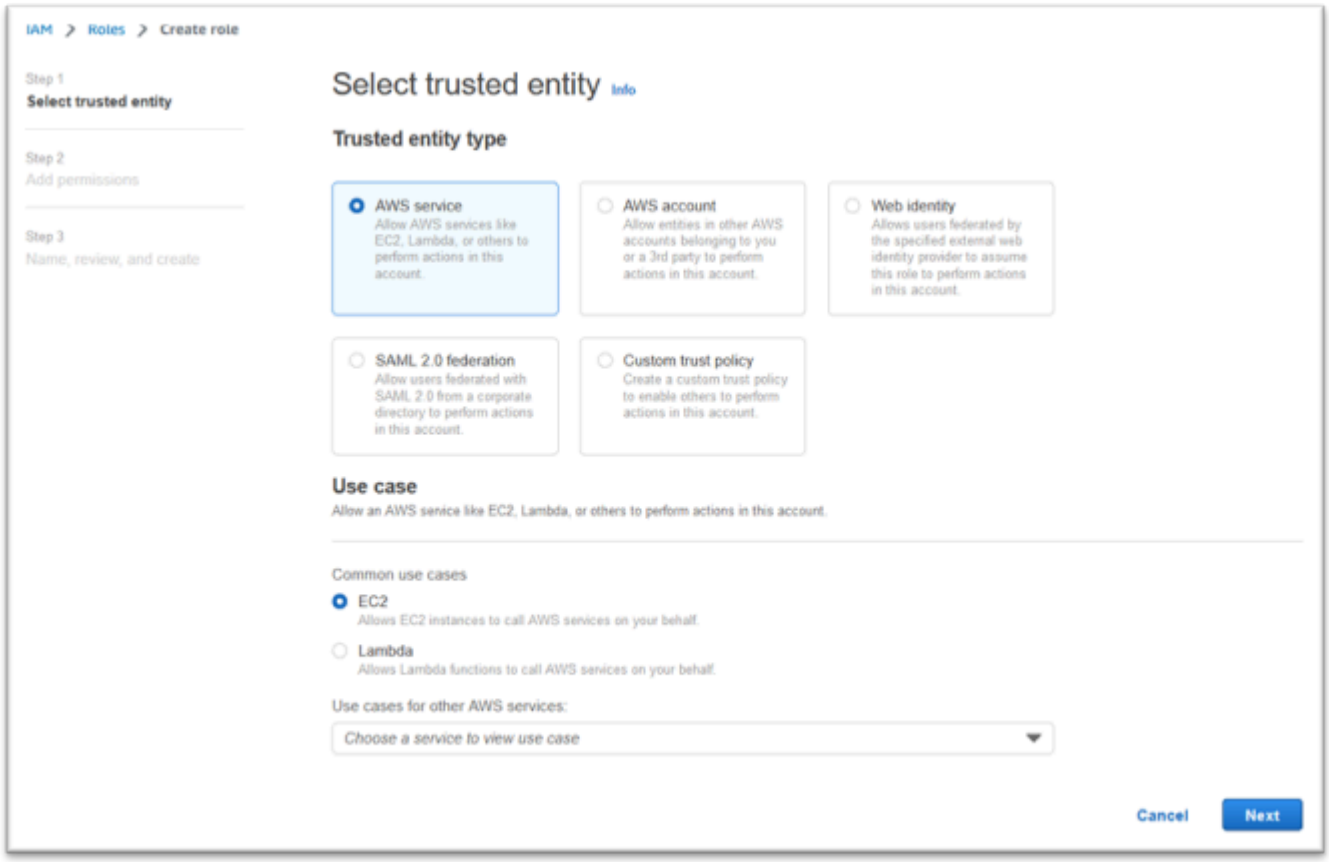
8. Choisissez Create Policy (Créer une politique).

Création du rôle IAM

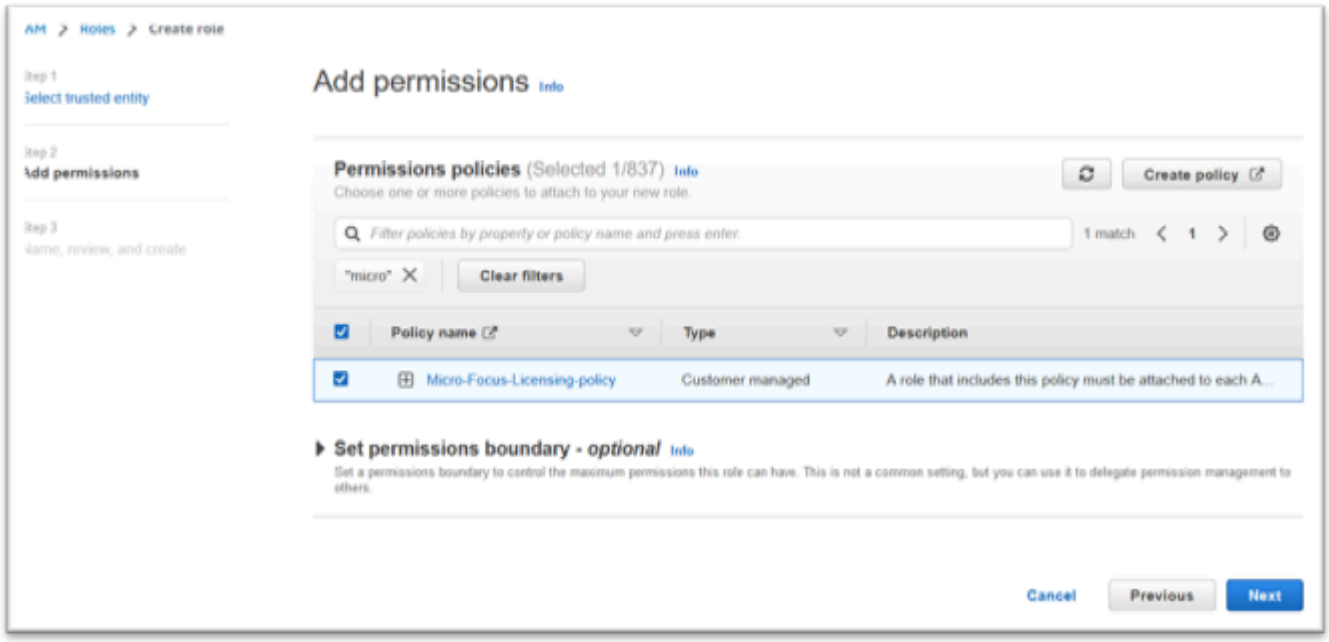
1. Accédez à IAM dans le. AWS Management Console
2. Choisissez Rôles, puis Créer un rôle.



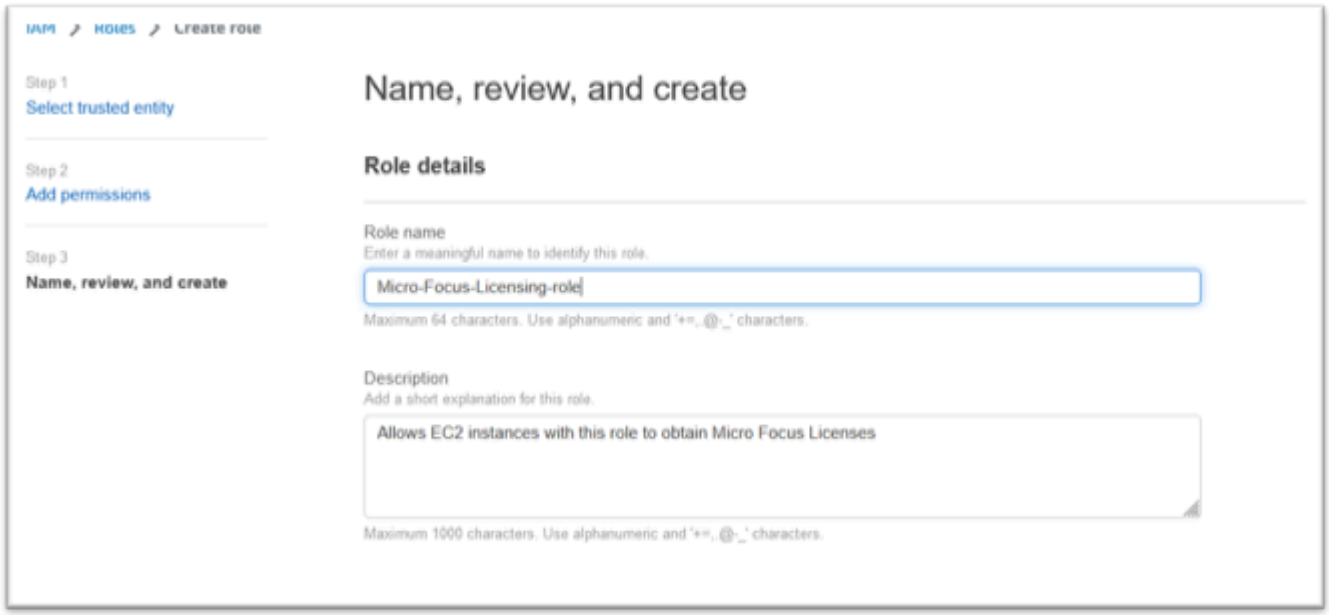
3. Laissez le type d'entité de confiance comme AWS service et choisissez le cas d'utilisation courant EC2.



4. Choisissez Suivant.
5. Entrez « Micro » dans le filtre et appuyez sur Entrée pour appliquer le filtre.
6. Choisissez la politique qui vient d'être créée, par exemple la « Micro-Focus-Licensing-Policy ».
7. Choisissez Suivant.



8. Entrez le nom du rôle, par exemple « Micro-Focus-Licensing-Role ».
9. Remplacez la description par la vôtre, par exemple « Autorise les instances Amazon EC2 dotées de ce rôle à obtenir des licences Micro Focus ».

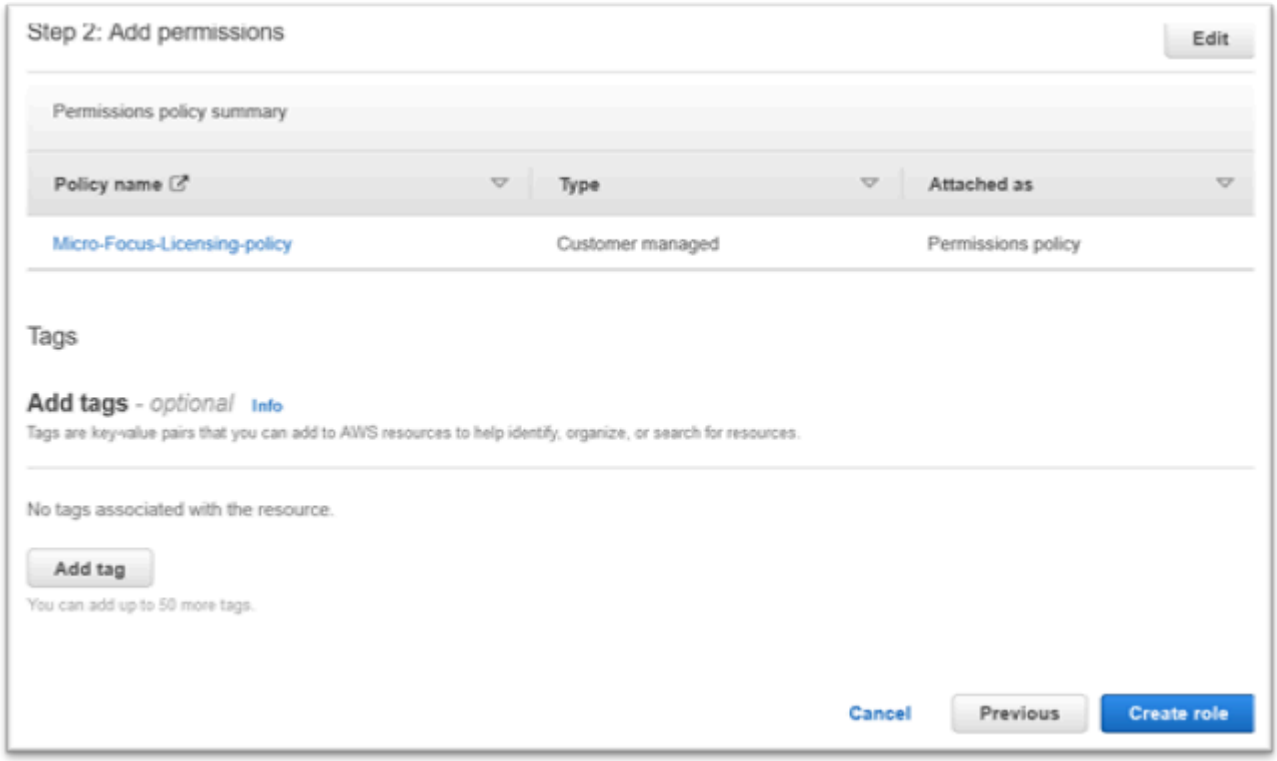


10. À l'étape 1 : Sélectionnez les entités de confiance, examinez le JSON et confirmez qu'il contient les valeurs suivantes :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

> **Note**
>
> L'ordre de l'effet, de l'action et du principal n'est pas significatif.

11. Vérifiez que l'étape 2 : Ajouter des autorisations indique votre politique de licence.

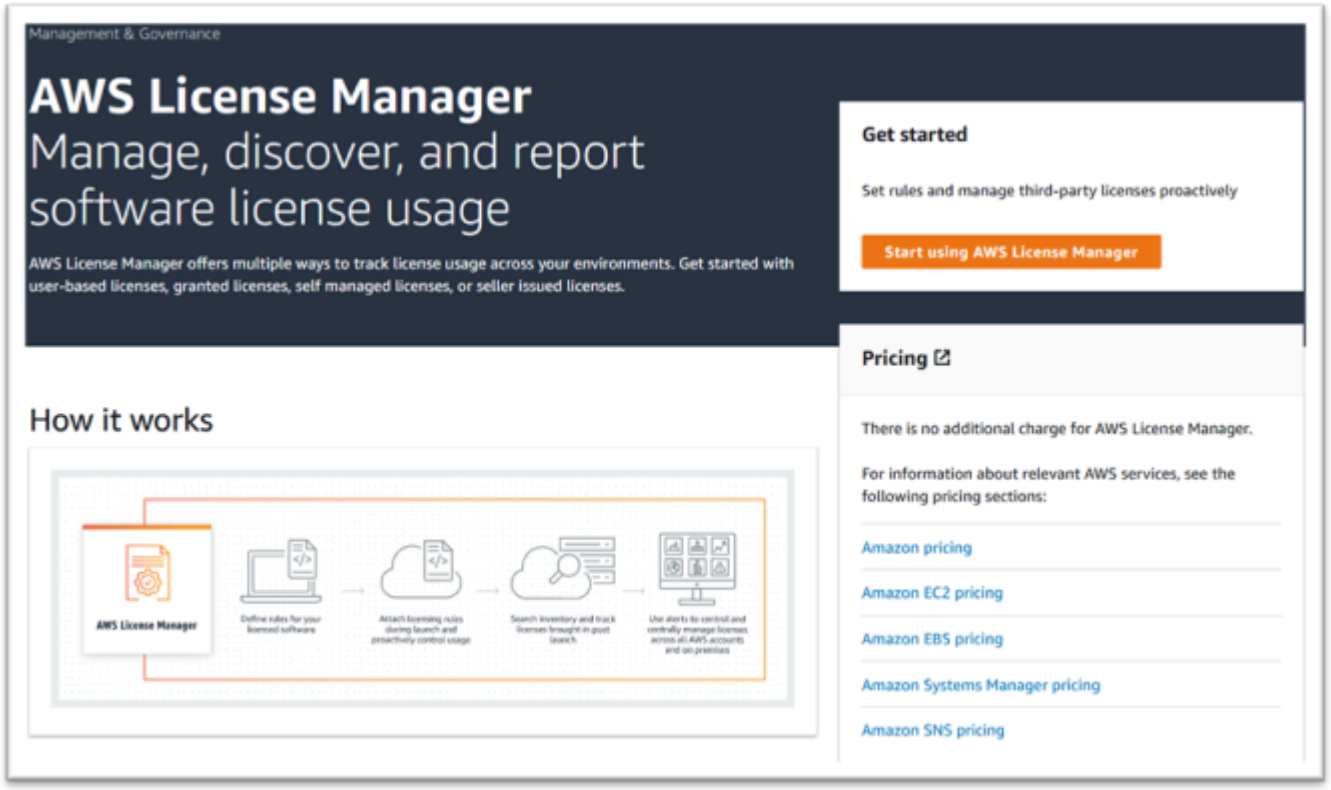


12. Sélectionnez Créer un rôle.

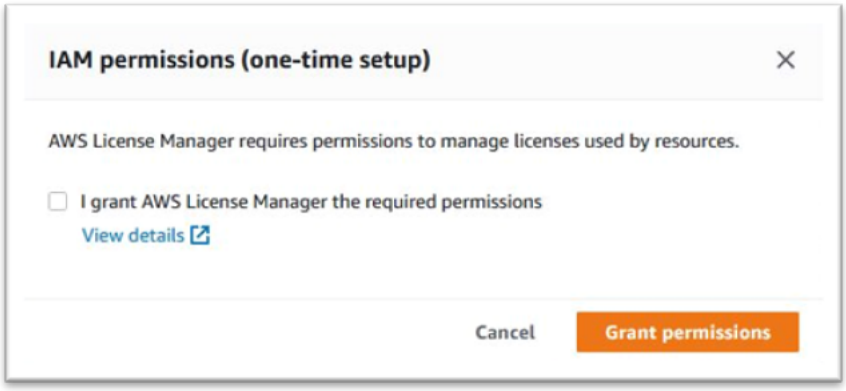
Une fois la demande de liste d'autorisation terminée, passez aux étapes suivantes.

Accordez à License Manager les autorisations requises

1. Naviguez vers AWS License Manager dans le AWS Management Console.



2. Choisissez Start using AWS License Manager.
3. Si la fenêtre contextuelle suivante s'affiche, affichez les détails, cochez la case et appuyez sur Accorder les autorisations.

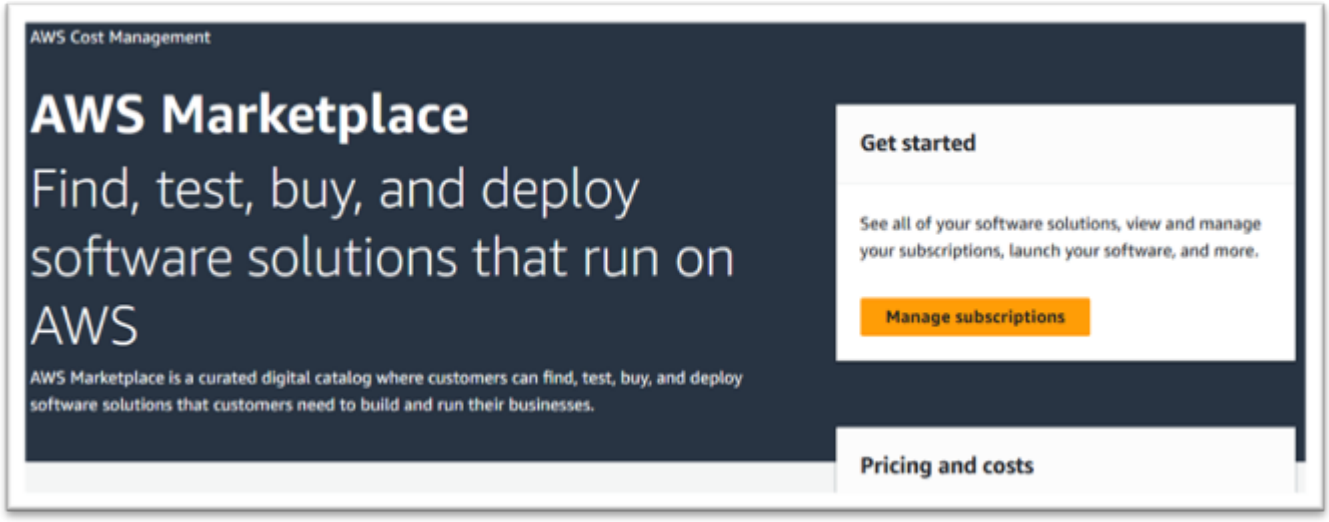


Abonnez-vous aux Amazon Machine Images

Une fois que vous êtes abonné à un AWS Marketplace produit, vous pouvez lancer une instance depuis l'AMI du produit.

1. Accédez à AWS Marketplace Abonnements dans le AWS Management Console.

2. Sélectionnez Manage subscriptions (Gérer les abonnements).



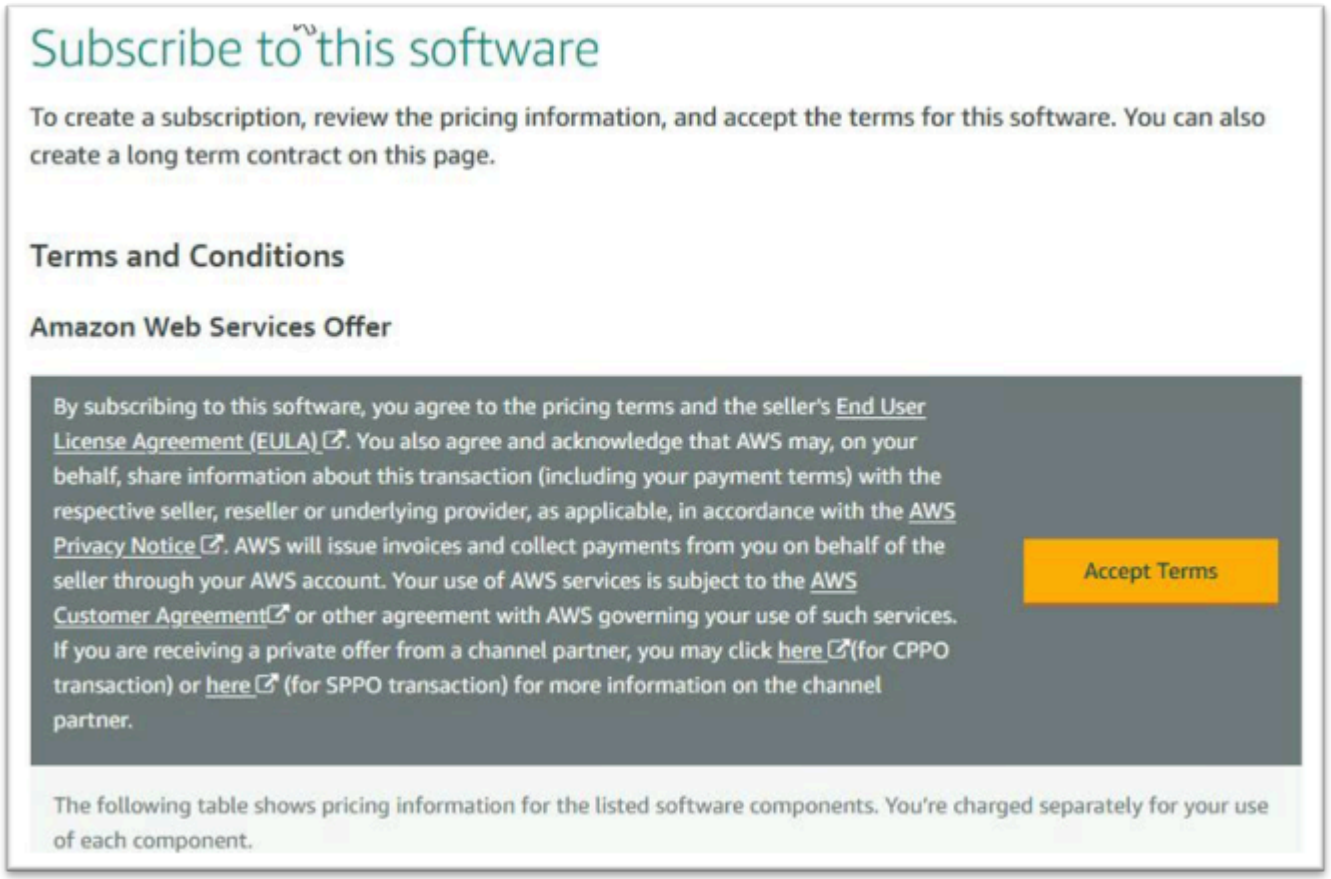
3. Copiez et collez l'un des liens suivants dans la barre d'adresse du navigateur.

> **Note**
>
> Choisissez un lien uniquement pour l'un des produits que vous êtes autorisé à utiliser.

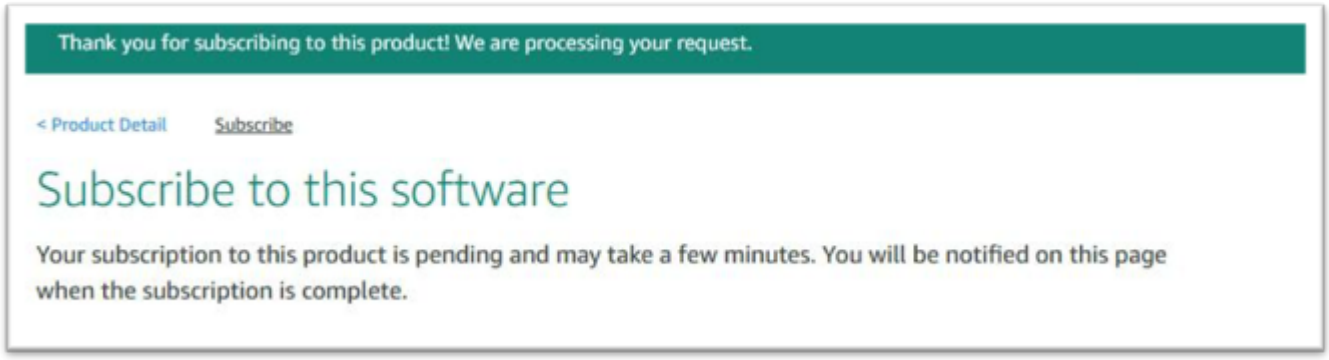
- Serveur d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-g5emev63l7blc
- Serveur d'entreprise pour Windows : https://aws.amazon.com/marketplace/pp/prodview-lwybsiyikbhc2
- Développeur d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-77qmpr42yzxwk
- Développeur d'entreprise avec Visual Studio 2022 : https://aws.amazon.com/marketplace/pp/prodview-m4l3lqiszo6cm
- Analyseur d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-tttheylcmcihm
- Outils de création d'entreprise pour Windows : https://aws.amazon.com/marketplace/pp/prodview-2rw35bbt6uozi
- Procédures stockées d'entreprise : https://aws.amazon.com/marketplace/pp/prodview-zoeyqnsdsj6ha
- Procédures stockées d'entreprise avec SQL Server 2019 : https://aws.amazon.com/marketplace/pp/prodview-ynfklquwubnz4

4. Choisissez Continue to Subscribe (Continuer pour s'abonner).

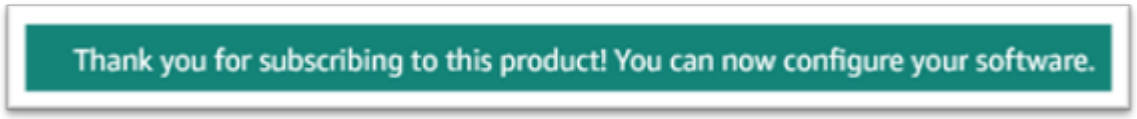
5. Si les conditions générales sont acceptables, choisissez Accepter les conditions.



6. Le traitement de l'abonnement peut prendre quelques minutes.

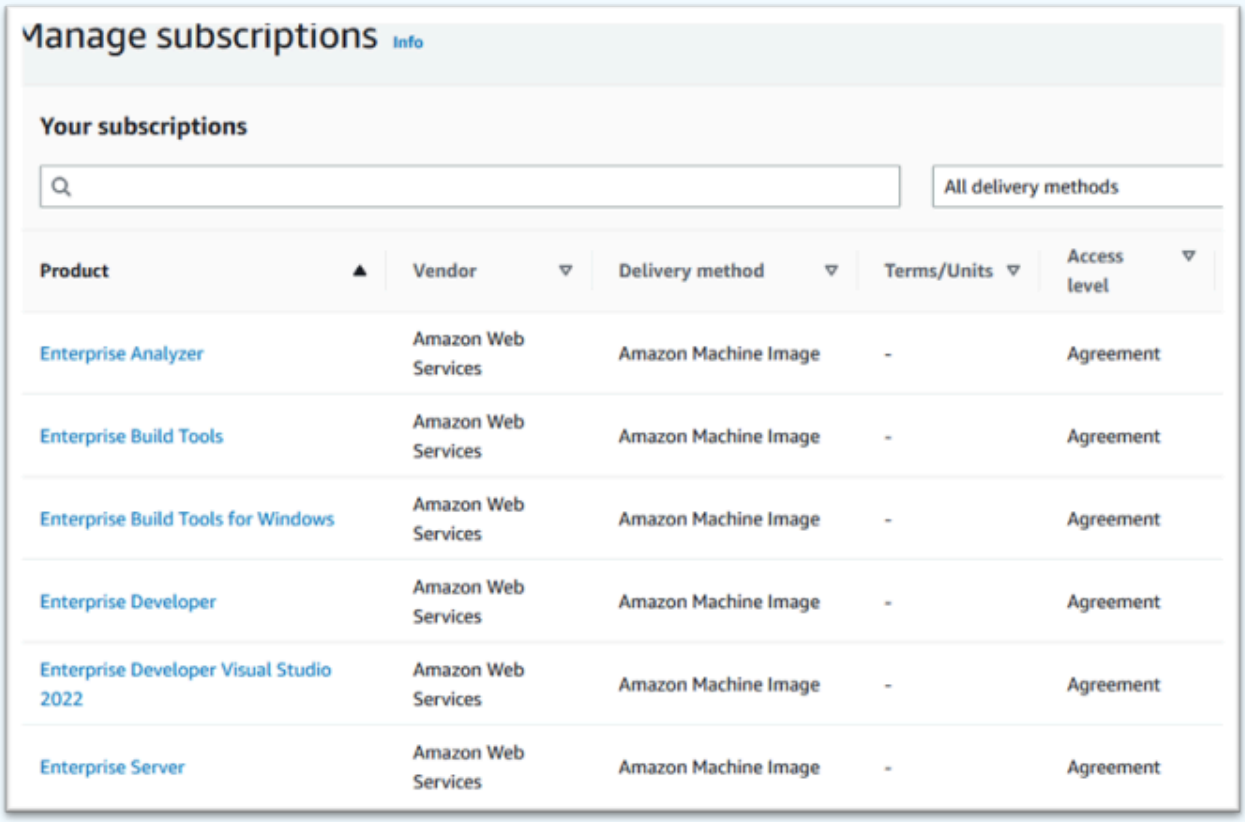


7. Une fois le message de remerciement affiché, copiez et collez le lien suivant de l'étape 3 pour continuer à ajouter des abonnements.



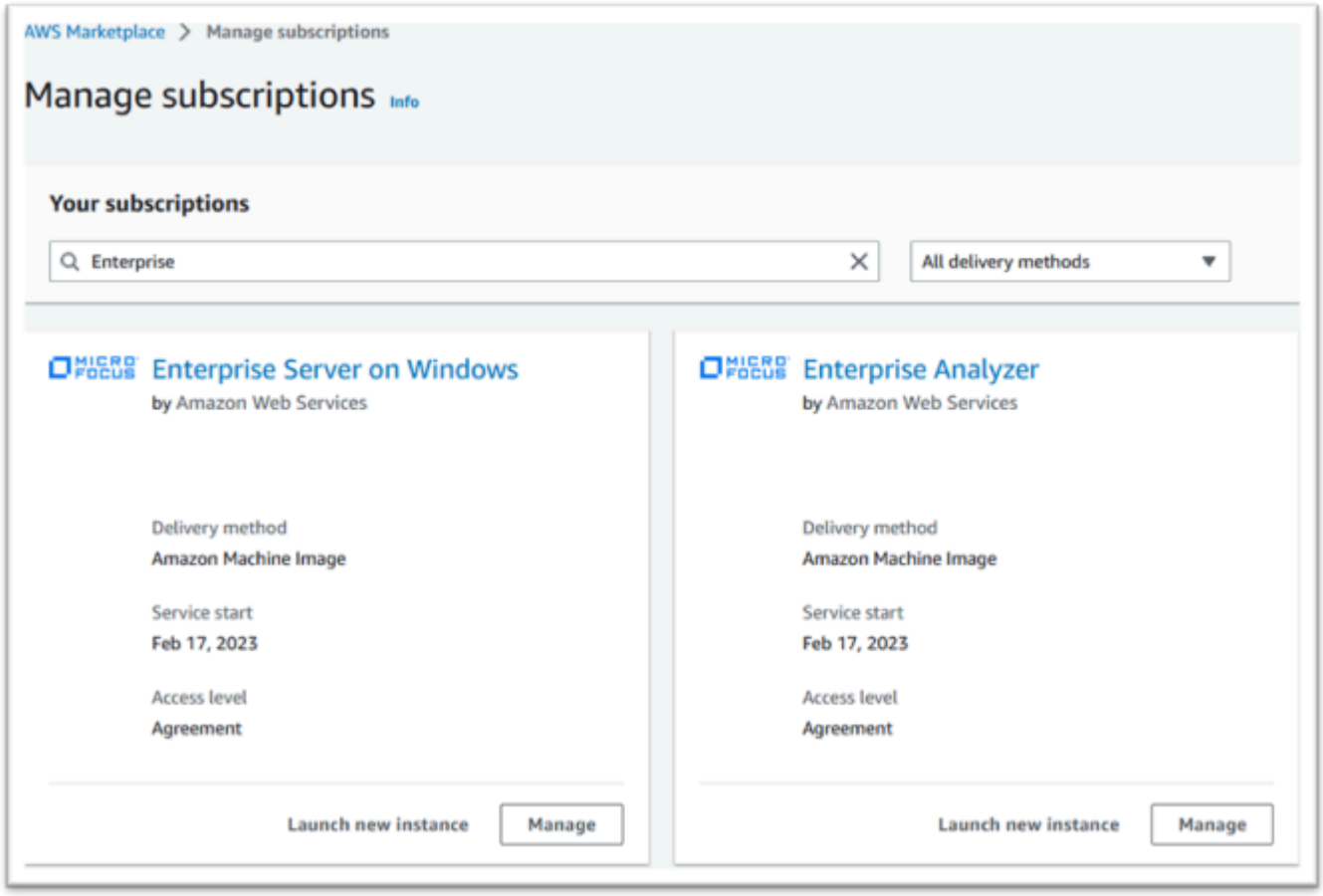
8. Arrêtez lorsque la section Gérer les abonnements affiche toutes les AMI auxquelles vous êtes abonné.

> **Note**
>
> Les préférences du panneau (icône en forme de roue dentée) sont définies pour afficher la vue sous forme de tableau.



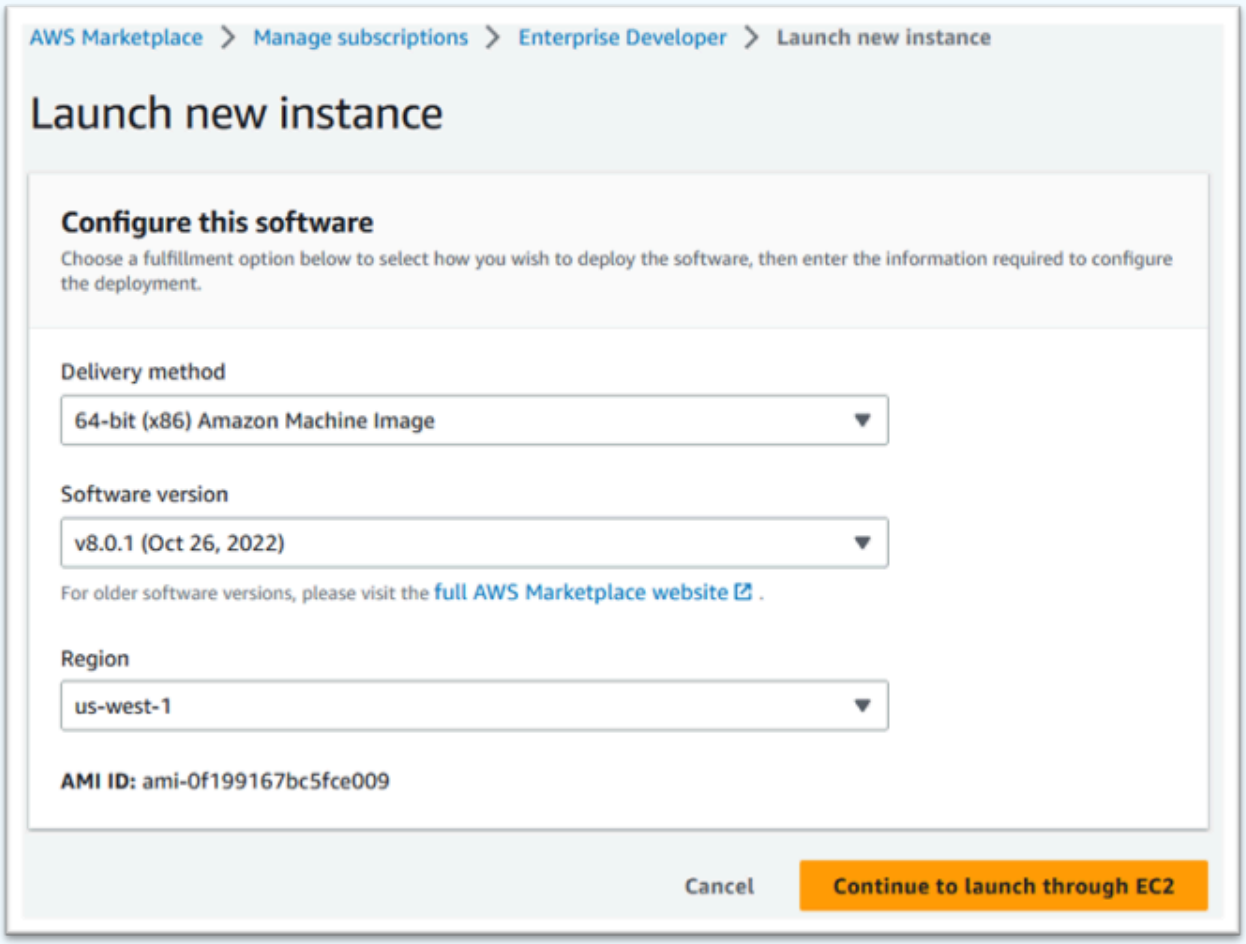
Lancer une AWS instance Micro Focus de modernisation du mainframe

1. Accédez à AWS Marketplace Abonnements dans le AWS Management Console.
2. Localisez l'AMI à lancer et choisissez Launch New Instance.



3. Dans la boîte de dialogue de lancement d'une nouvelle instance, assurez-vous que la région autorisée est sélectionnée.
4. Appuyez sur Continuer pour lancer via EC2.

> **Note**
>
> L'exemple suivant montre le lancement d'une AMI Enterprise Developer, mais le processus est le même pour toutes les AMI de modernisation du AWS mainframe.

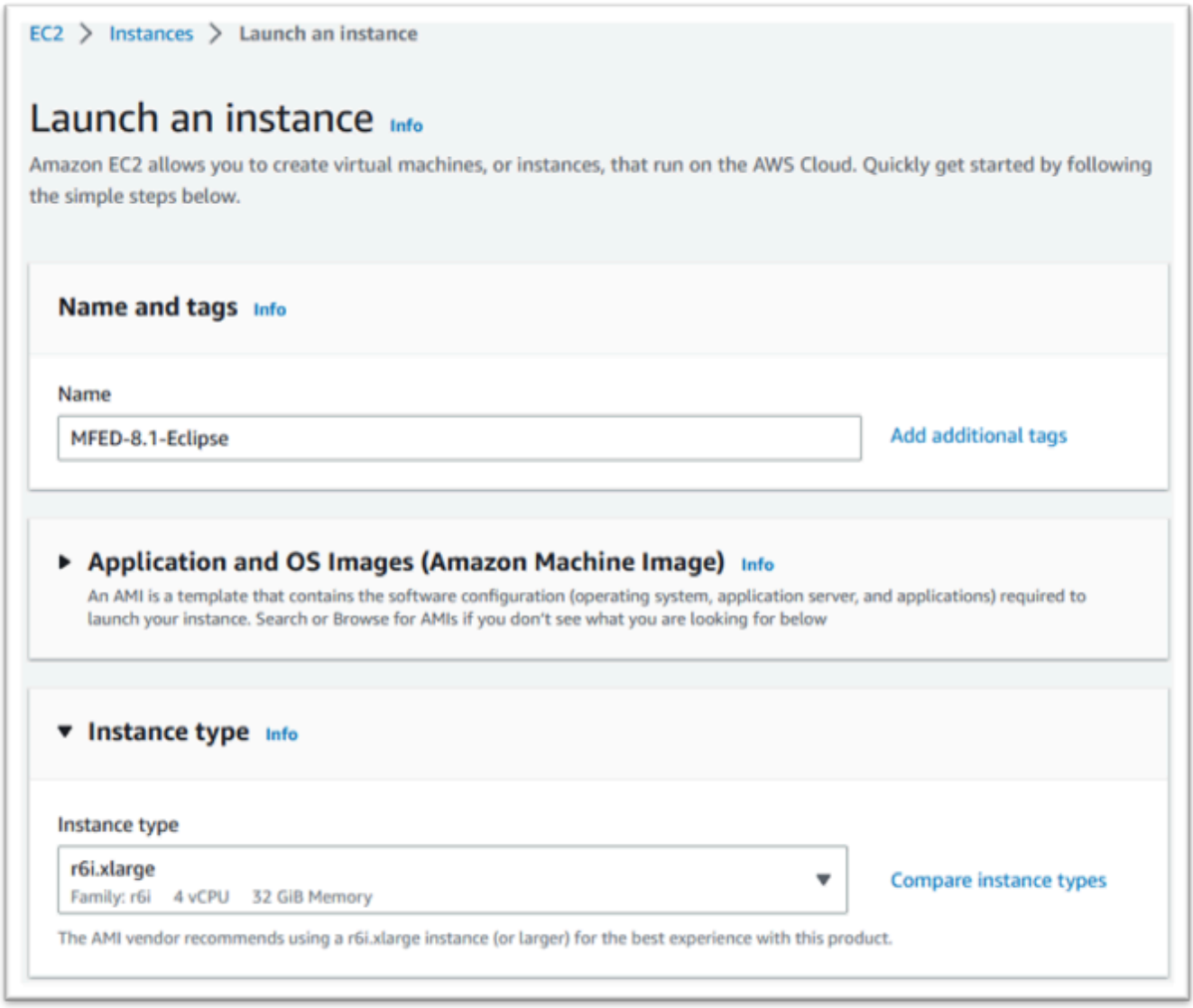


5. Entrez un nom pour le serveur.
6. Choisissez un type d'instance.

   Le type d'instance sélectionné doit être déterminé par les exigences de performance et de coût du projet. Les points de départ suggérés sont les suivants :

   - Pour Enterprise Analyzer, un r6i.xlarge
   - Pour les développeurs d'entreprise, un r6i.large
   - Pour une instance autonome d'Enterprise Server, un r6i.xlarge
   - Pour le cluster de disponibilité des performances (PAC) Micro Focus avec évolutivité, un r6i.large

> **Note**
>
> La section Images de l'application et du système d'exploitation a été réduite pour la capture d'écran.



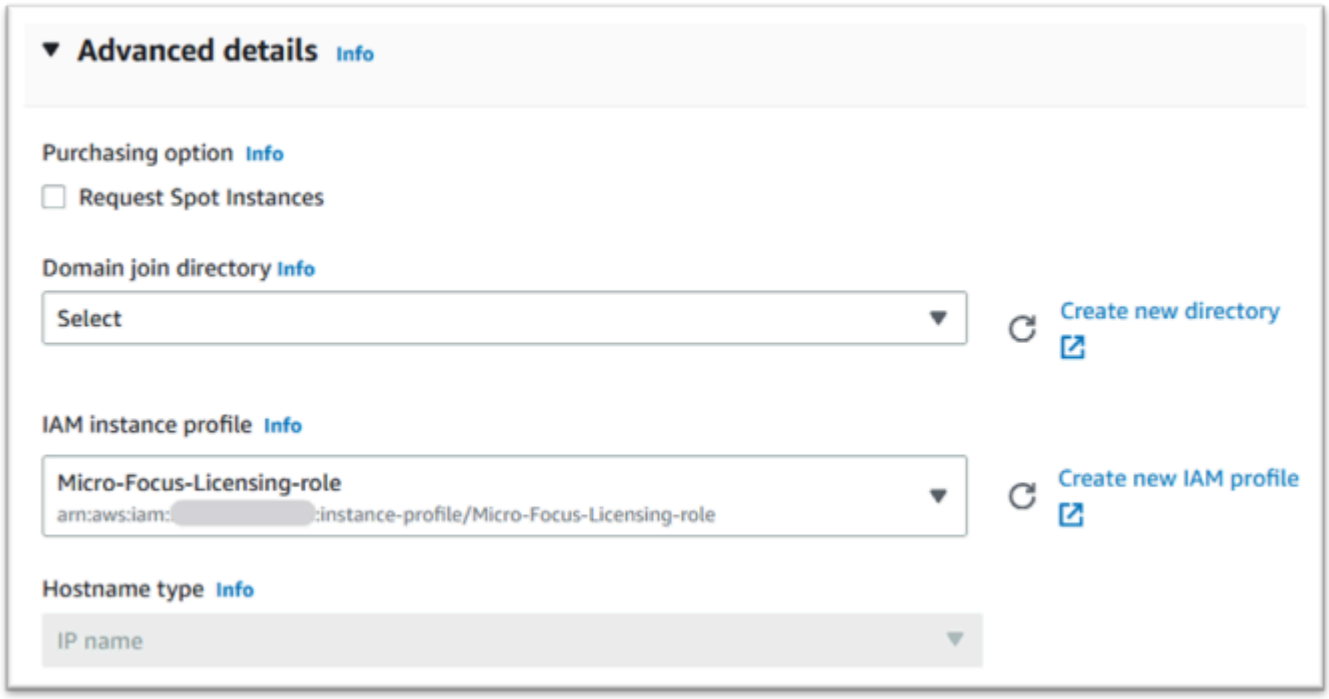
7. Choisissez ou créez (et enregistrez) une paire de clés (non illustrée).

   Pour plus d'informations sur les paires de clés pour les instances Linux, consultez les paires de clés Amazon EC2 et les instances Linux.

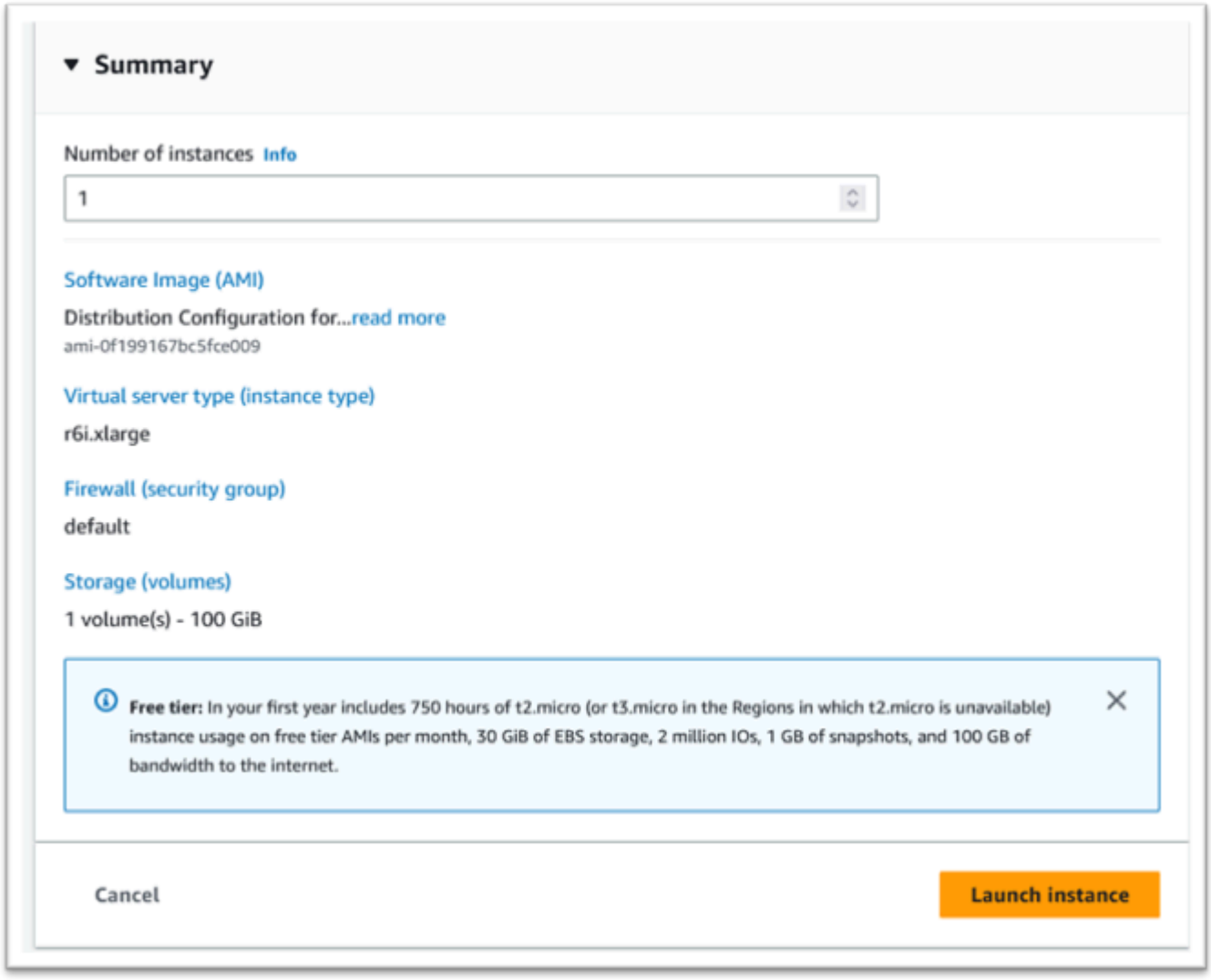
Pour plus d'informations sur les paires de clés pour les instances Windows, consultez la section Paires de clés Amazon EC2 et instances Windows.

8. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.
9. Choisissez ou créez un groupe de sécurité. S'il s'agit d'une instance Enterprise Server EC2, il est courant d'autoriser le trafic TCP vers les ports 86 et 10086 pour administrer la configuration Micro Focus.
10. Configurez éventuellement le stockage pour l'instance Amazon EC2.
11. Important - Développez les détails avancés et sous Profil d'instance IAM, choisissez le rôle de licence créé précédemment, par exemple « Micro-Focus-Licensing-Role ».

> **Note**
>
> Si cette étape est manquée, une fois l'instance créée, vous pouvez modifier le rôle IAM à partir de l'option Sécurité du menu Action pour l'instance EC2.

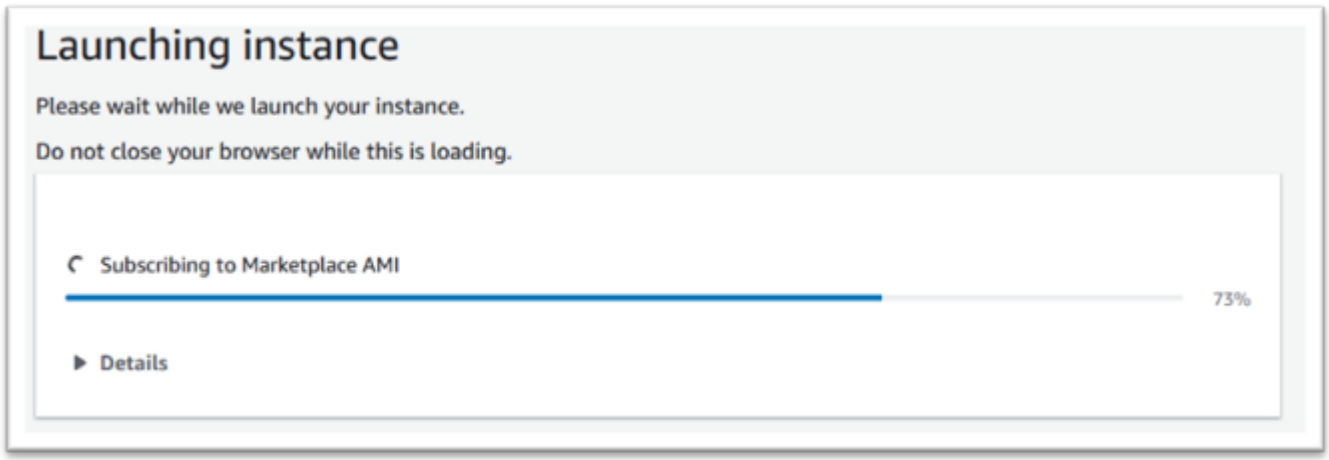


12. Consultez le résumé et appuyez sur Launch Instance.

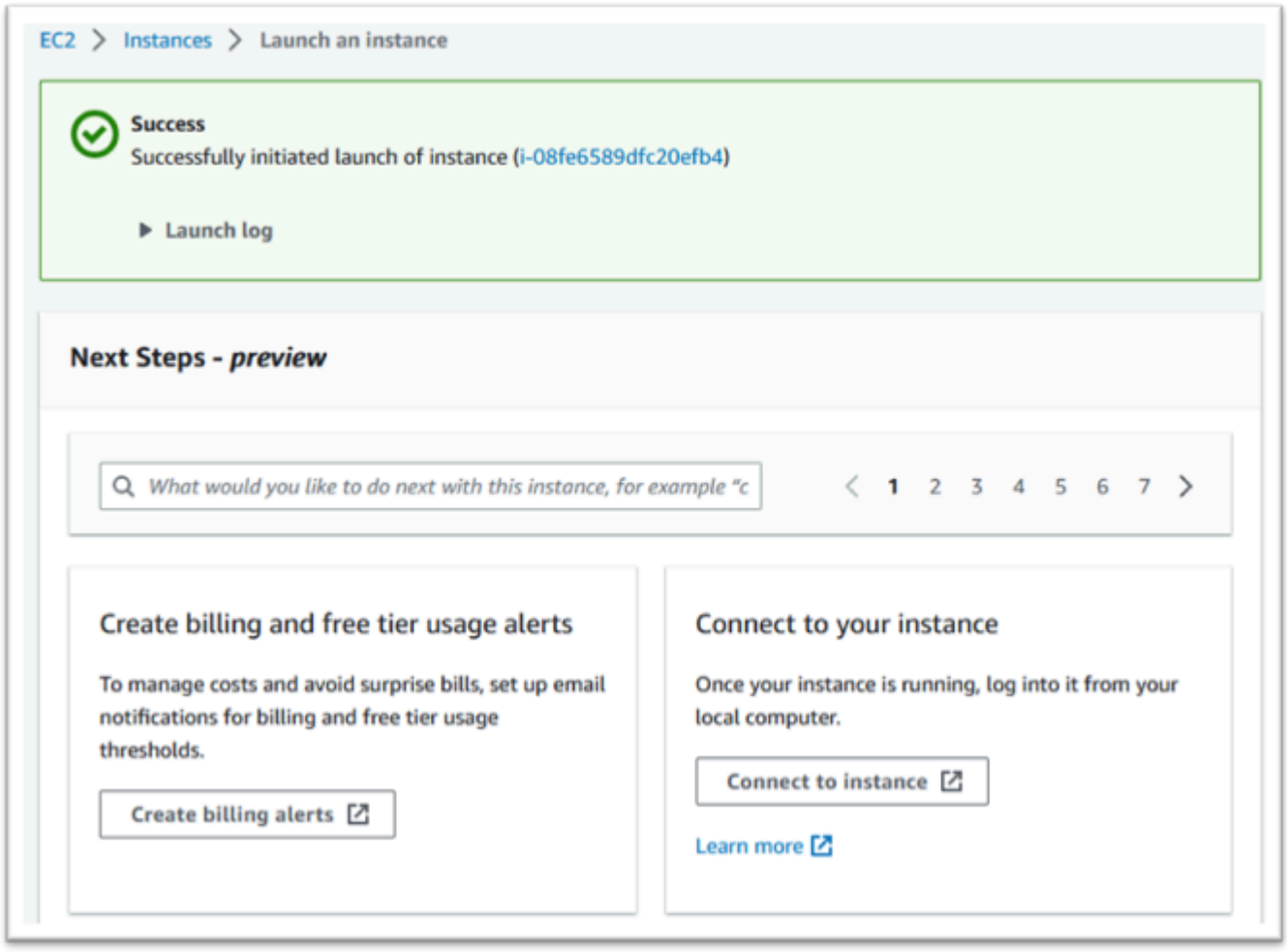


13. Le lancement de l'instance échouera si un type de serveur virtuel non valide est sélectionné.

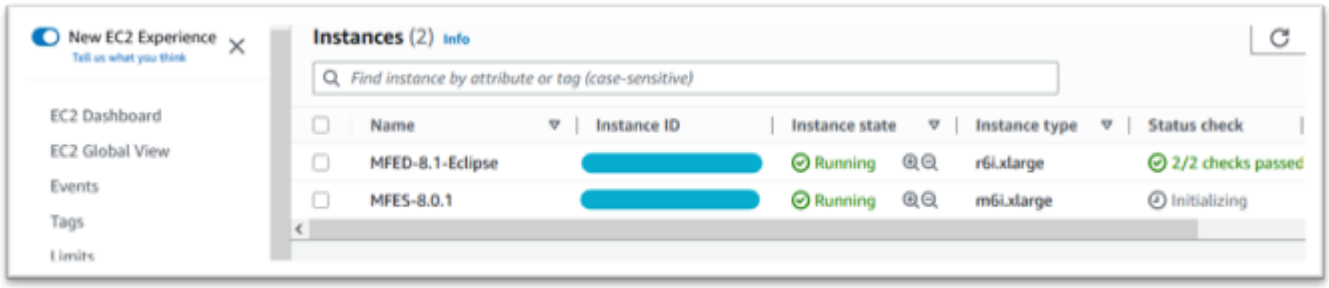
    Dans ce cas, choisissez Modifier la configuration de l'instance et modifiez le type d'instance.



14. Une fois le message « Success » affiché, choisissez Connect to instance pour obtenir les détails de connexion.



15. Vous pouvez également accéder à EC2 dans le AWS Management Console.
16. Choisissez Instances pour voir le statut de la nouvelle instance.



Sous-réseau ou VPC sans accès Internet

Apportez ces modifications supplémentaires si le sous-réseau ou le VPC ne dispose pas d'un accès Internet sortant.

Le gestionnaire de licence doit avoir accès aux services AWS suivants :

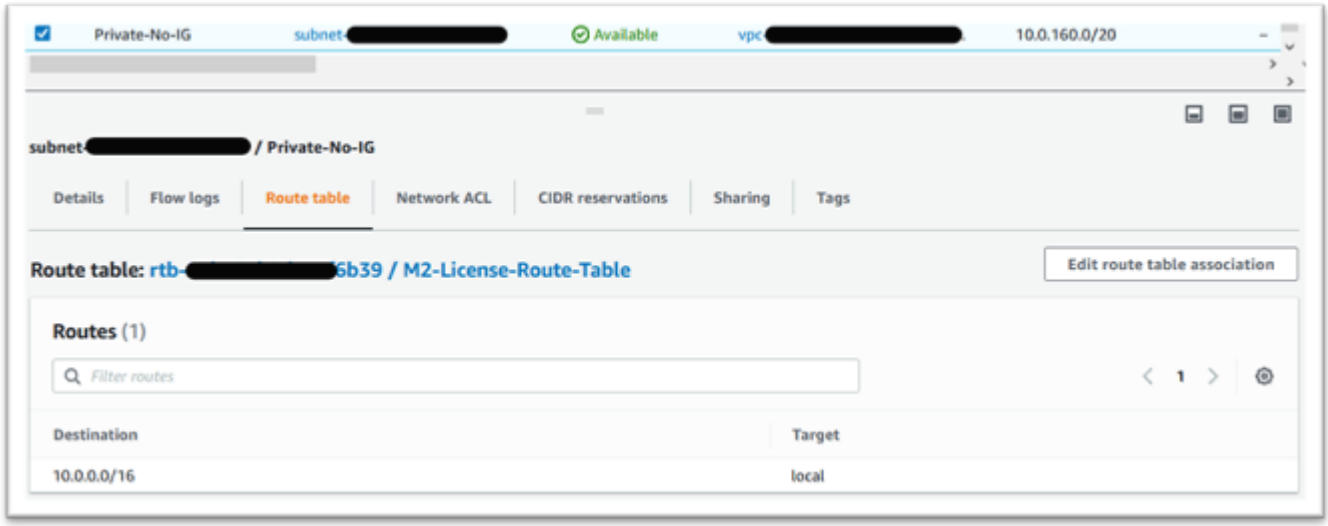
- com.amazonaws.*region*.s3
- com.amazonaws.*region*.ec2
- com.amazonaws.*region*.license-manager
- com.amazonaws.*région*.sts

Les étapes précédentes ont défini le com.amazonaws. service *region* .s3 en tant que point de terminaison de passerelle. Ce point de terminaison a besoin d'une entrée de table de routage pour tous les sous-réseaux sans accès à Internet.

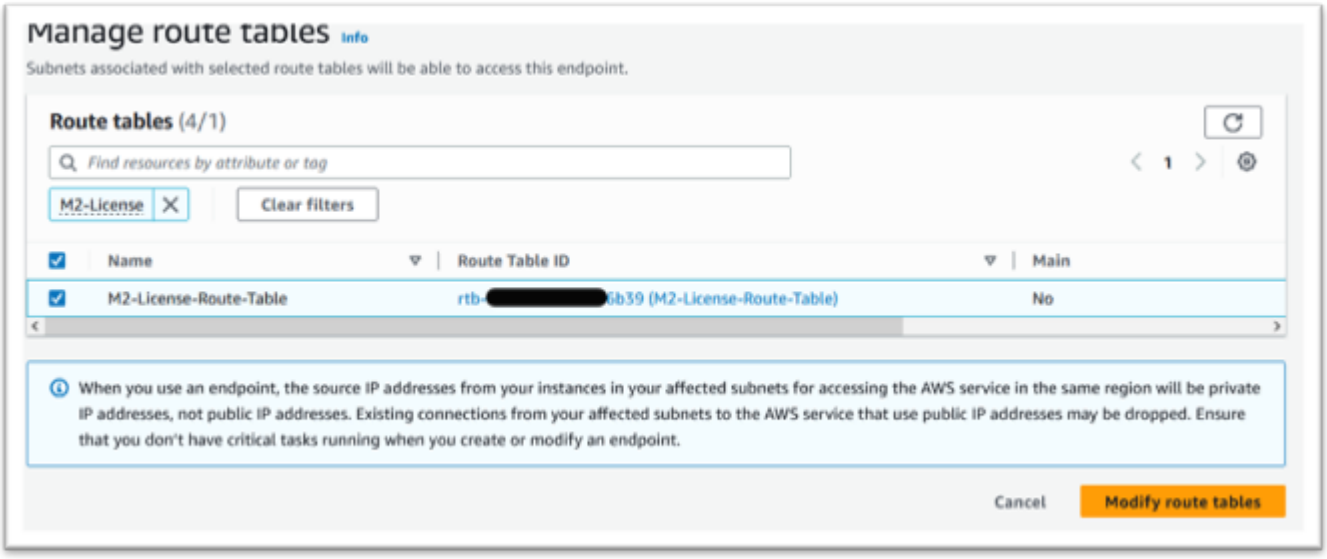
Les trois services supplémentaires seront définis comme des points de terminaison d'interface.

Ajoutez l'entrée de la table de routage pour le point de terminaison Amazon S3

1. Accédez à VPC dans le AWS Management Console et choisissez Subnets.
2. Choisissez le sous-réseau dans lequel les instances Amazon EC2 seront créées et cliquez sur l'onglet Route Table.
3. Notez les quelques derniers chiffres de l'identifiant de la table de routage. Par exemple, le 6b39 dans l'image ci-dessous.



4. Choisissez Endpoints dans le volet de navigation.
5. Choisissez le point de terminaison créé précédemment, puis gérez les tables de routage, soit dans l'onglet Tables de routage du point de terminaison, soit dans le menu déroulant Actions.
6. Choisissez la table de routage à l'aide des chiffres identifiés précédemment et appuyez sur Modifier les tables de routage.

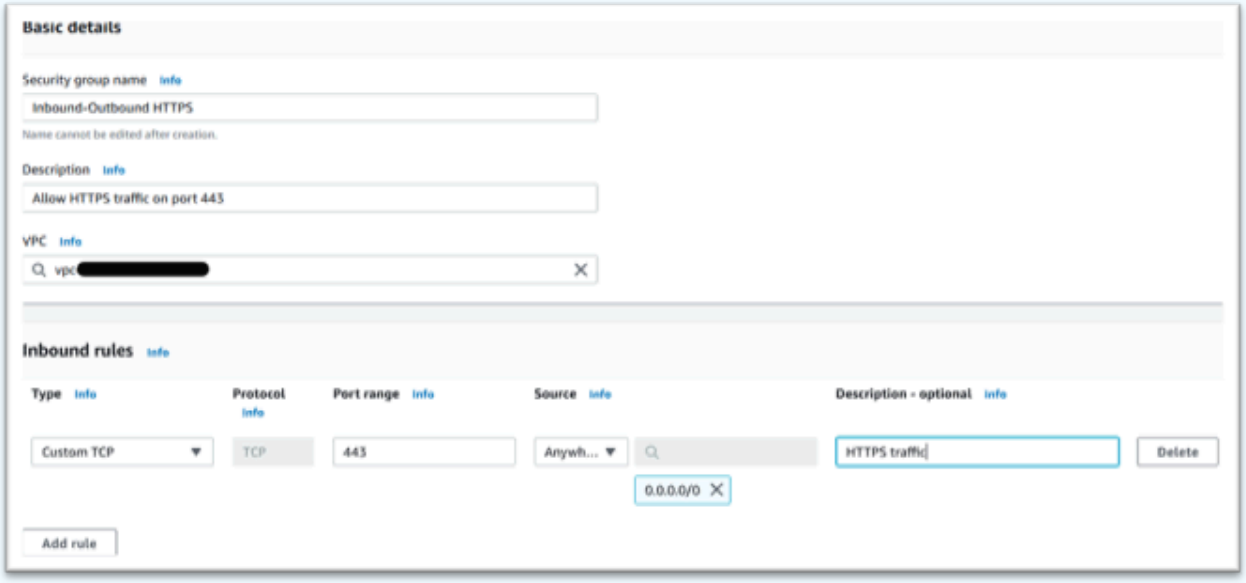


Définissez le groupe de sécurité requis

Les services Amazon EC2 et License Manager communiquent via HTTPS via le port 443. AWS STS Cette communication est bidirectionnelle et nécessite des règles entrantes et sortantes pour permettre à l'instance de communiquer avec les services.

1. Accédez à Amazon VPC dans le. AWS Management Console
2. Localisez les groupes de sécurité dans la barre de navigation et choisissez Créer un groupe de sécurité.
3. Entrez le nom et la description du groupe de sécurité, par exemple « Inbound-Outbound HTTPS ».
4. Appuyez sur le X dans la zone de sélection du VPC pour supprimer le VPC par défaut, puis choisissez le VPC qui contient le point de terminaison S3.
5. Ajoutez une règle entrante qui autorise le trafic TCP sur le port 443 depuis n'importe où.

> **Note**
>
> Les règles entrantes (et sortantes) peuvent être restreintes davantage en limitant la source. Pour plus d'informations, consultez la section Contrôlez le trafic vers vos AWS ressources à l'aide de groupes de sécurité dans le guide de l'utilisateur Amazon VPC.

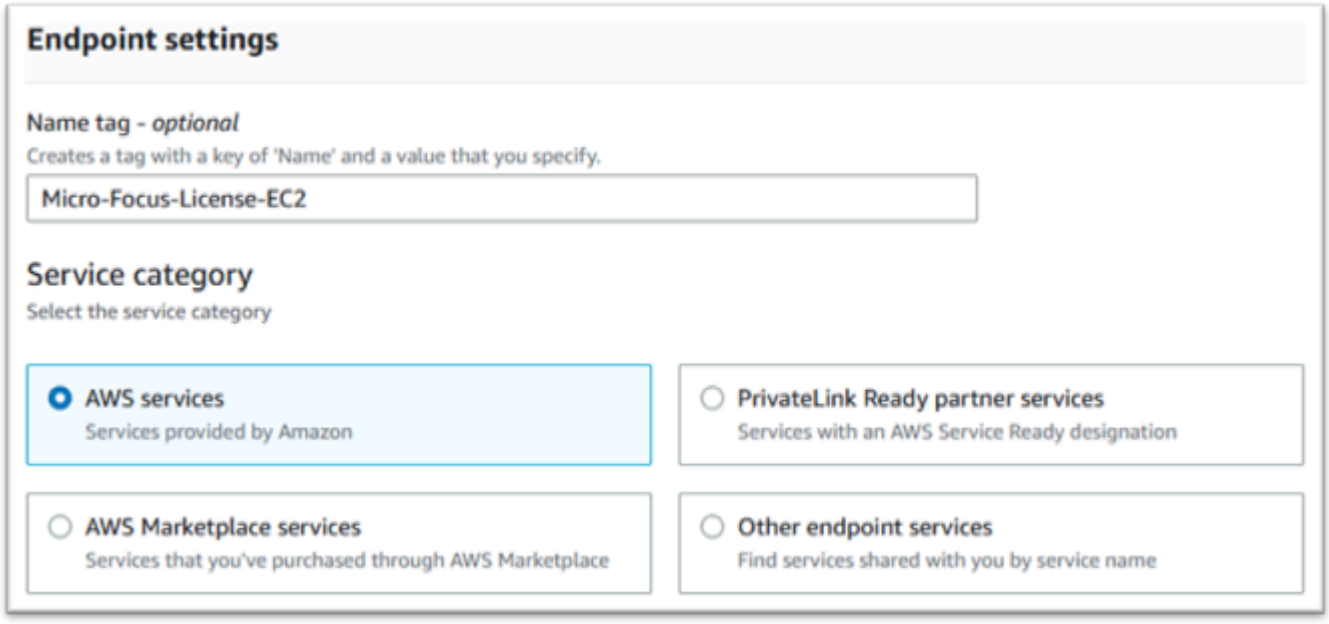


6. Appuyez sur Créer un groupe de sécurité.

Création des points de terminaison du service

Répétez ce processus trois fois, une fois pour chaque service.

1. Accédez à Amazon VPC dans le AWS Management Console et choisissez Endpoints.
2. Appuyez sur Créer un point de terminaison.
3. Entrez un nom, par exemple « Micro-Focus-License-EC2 », « Micro-Focus-License-STS » ou « Micro-Focus-License-Manager ».
4. Choisissez la catégorie de service AWS Services.



5. Sous Services, recherchez le service d'interface correspondant, qui est l'un des suivants :

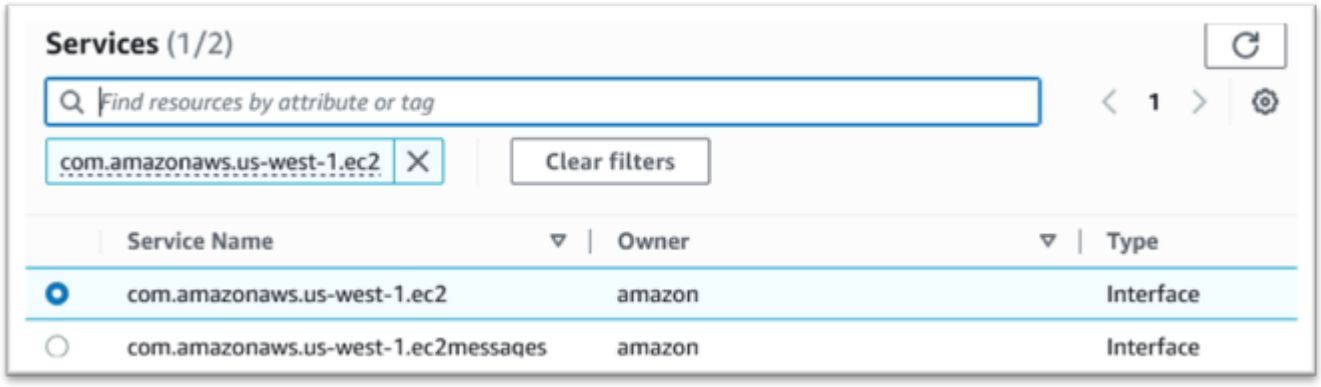
   - « com.amazonaws. *région* .ec2 »
   - « com.amazonaws. *région* « .sts »
   - « com.amazonaws. *region* .license-manager »

   Par exemple :

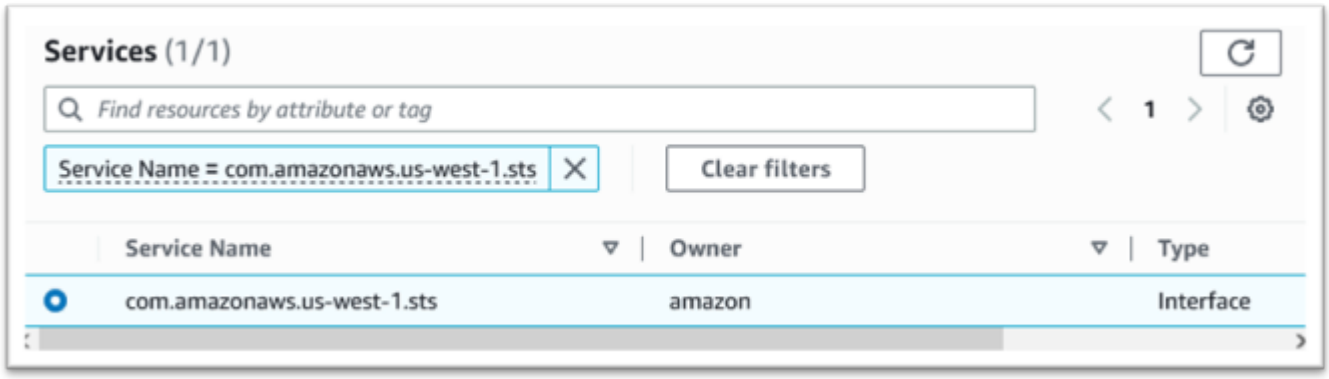
   - « com.amazonaws.us-west-1.ec2 »
   - « com.amazonaws.us-west-1.sts »
   - « com.amazonaws.us-west-1.license-manager »

6. Choisissez le service d'interface correspondant.

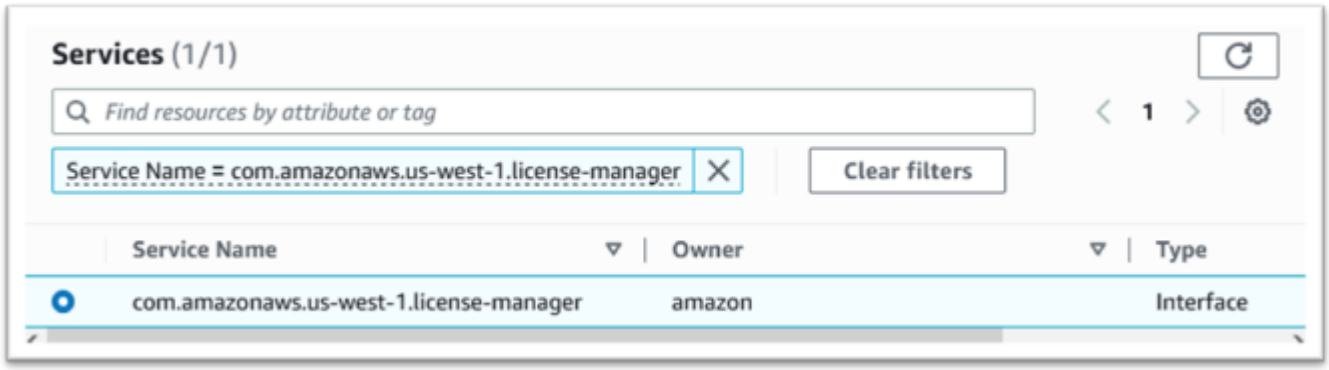
   com.amazonaws. ***région*** .ec2 :



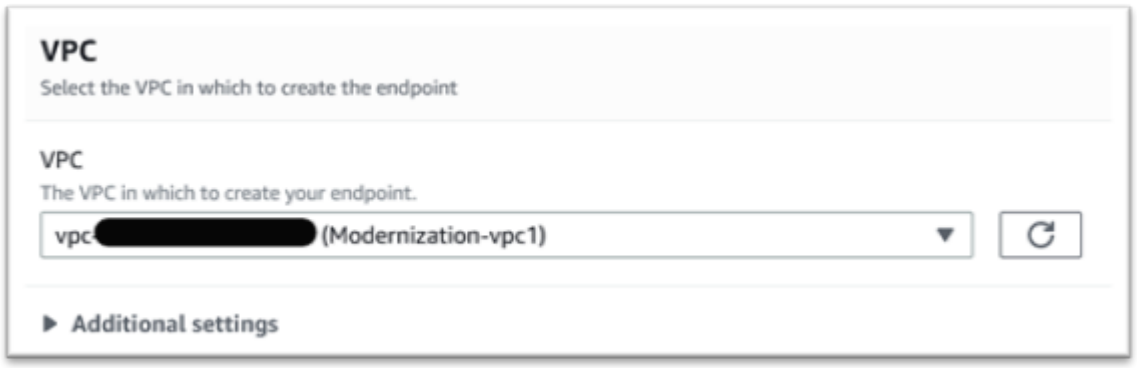
com.amazonaws. ***région*** .sts :



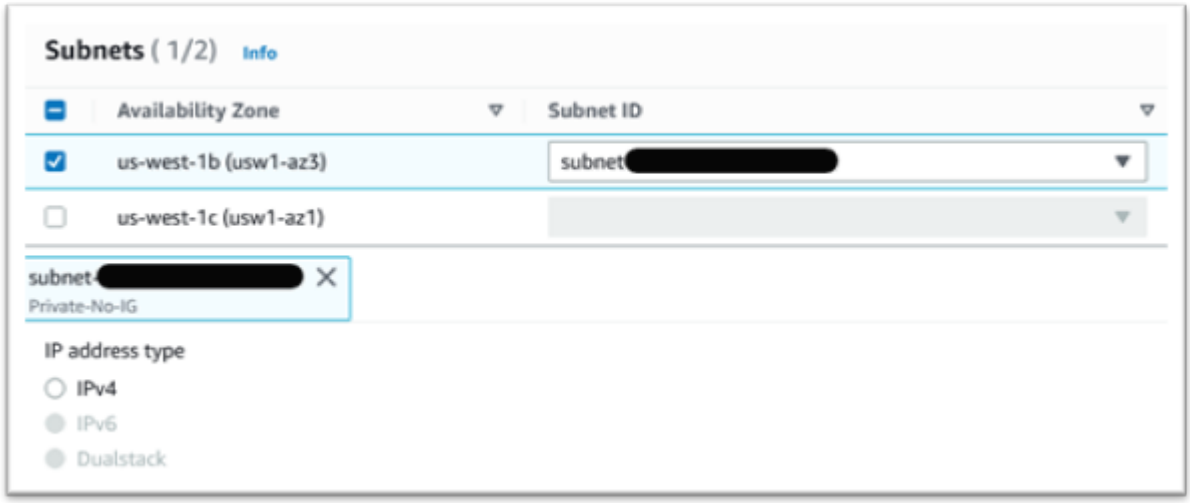
com.amazonaws. ***region*** .license-manager :



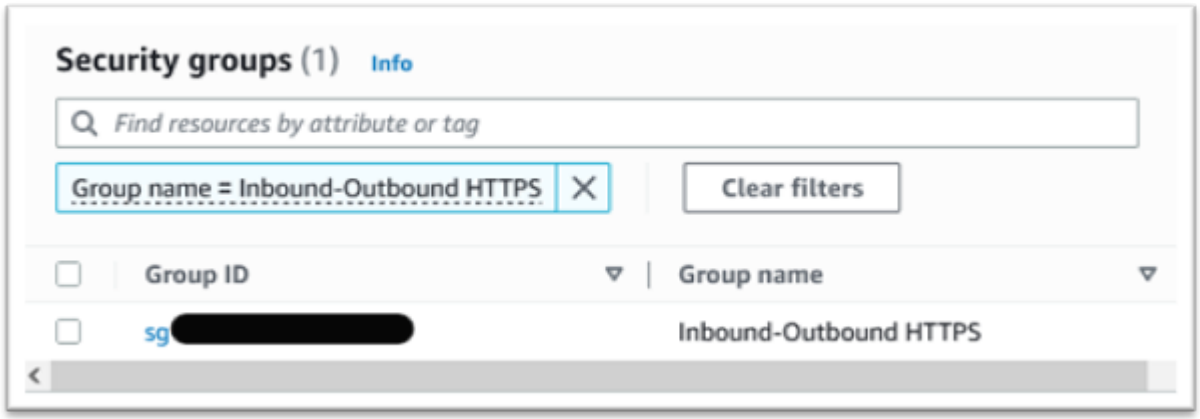
7. Pour VPC, choisissez le VPC pour l'instance.



8. Choisissez la zone de disponibilité et les sous-réseaux pour le VPC.



9. Choisissez le groupe de sécurité créé précédemment.



10. Sous Politique, sélectionnez Accès complet.

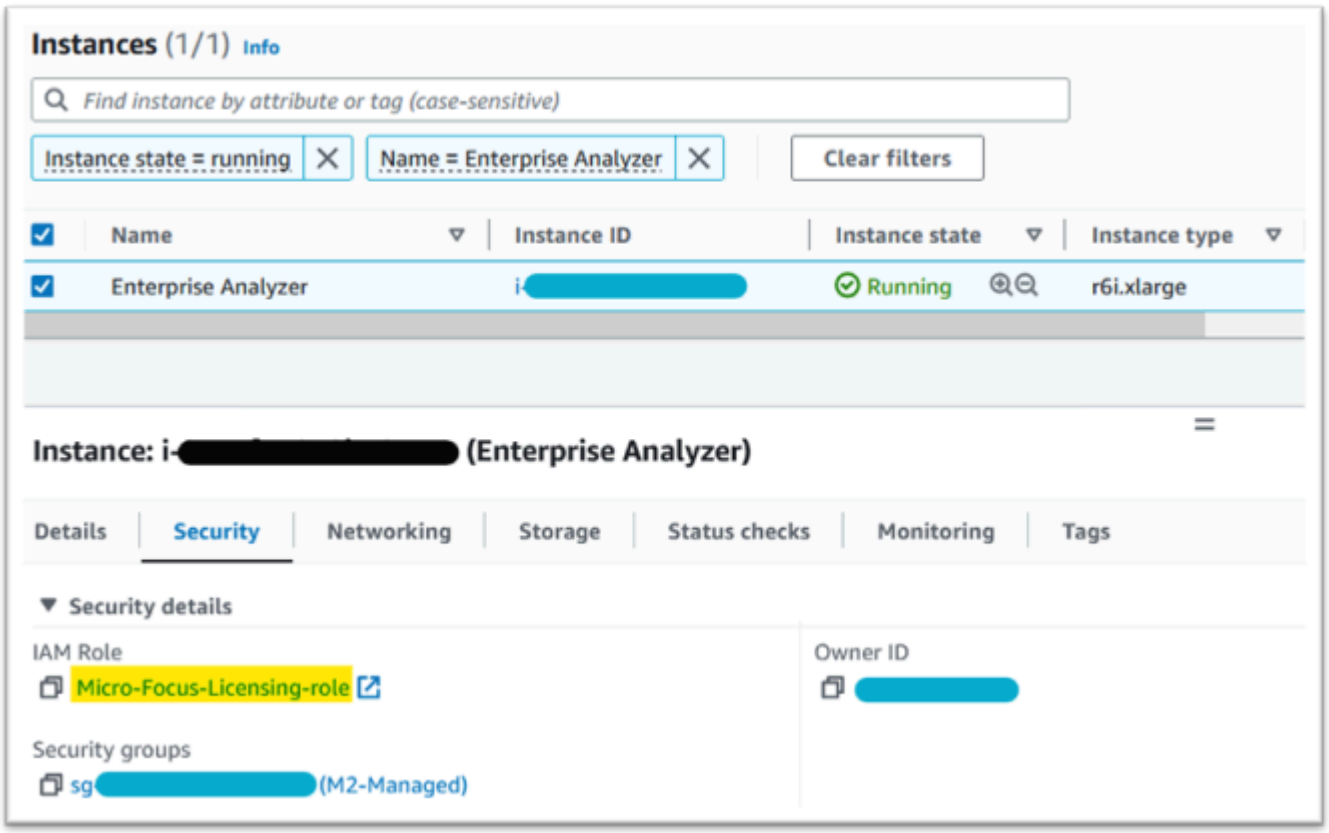
11. Choisissez Créer un point de terminaison.
12. Répétez cette procédure pour les autres interfaces.

Résolution des problèmes de licence

Si vous ne parvenez pas à accéder aux AMI ou à les utiliser, les informations suivantes peuvent vous être utiles.

Vérifiez que l'instance Amazon EC2 possède le rôle de licence IAM

Cela peut être vérifié dans l'onglet Sécurité des détails de l'instance Amazon EC2. Cela peut être modifié à l'aide de l'option de sécurité du menu déroulant Actions.



Utiliser l'Analyzer de Reachability

Trouvez l'Analyseur de Reachability sur la page de la console. AWS Network Manager

Créez et analysez un chemin entre l'instance Amazon EC2 créée à partir de l'AMI et le point de terminaison Amazon S3 VPC.

Si l'instance Amazon EC2 n'a pas accès à Internet, répétez l'analyse du chemin vers les 4 points de terminaison.

Pour plus d'informations sur Reachability Analyzer, consultez Getting started with Reachability Analyzer dans le guide Reachability Analyzer.

Exécutez le daemon de licence

Sur Windows Enterprise Developer, utilisez la commande suivante à partir d'une invite de commande :

```
“C:\Program Files (x86)\Micro Focus\Enterprise Developer\AdoptOpenJDK\bin\java” -jar
 "C:\Program Files (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

et examinez le résultat. Ignorez les messages SLF4J et recherchez la première exception.

Sur Enterprise Analyzer, utilisez la commande suivante à partir d'une invite de commande :

```
"C:\Program Files (x86)\Micro Focus\AdoptOpenJDK\bin\java" -jar "C:\Program Files
 (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

et examinez le résultat. Ignorez les messages SLF4J et recherchez la première exception.

Sur Linux, exécutez :

```
java -jar /var/microfocuslicensing/bin/aws-license-daemon.jar
```

Ignorez les messages SLF4J et recherchez la première exception.

Par exemple, si la ressource Amazon S3 n'est pas disponible, l'exception est la suivante :

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Exception in thread "main" software.amazon.awssdk.services.s3.model.S3Exception: Access
 Denied (Service: S3, Status Code: 403, Request ID: P6
```

Le message d'exception indique quelle ressource n'est pas disponible. Comparez les valeurs de configuration à celles présentées dans cette rubrique.

Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer

AWSLa modernisation des mainframes fournit plusieurs outils via Amazon AppStream 2.0. AppStream2.0 est un service de streaming d'applications sécurisé et entièrement géré qui vous permet de diffuser des applications de bureau aux utilisateurs sans réécrire les applications. AppStreamLa version 2.0 fournit aux utilisateurs un accès instantané aux applications dont ils ont besoin avec une expérience utilisateur réactive et fluide sur l'appareil de leur choix. L'utilisation de

la AppStream version 2.0 pour héberger des outils spécifiques au moteur d'exécution permet aux équipes chargées des applications des clients d'utiliser les outils directement depuis leur navigateur Web, en interagissant avec les fichiers d'application stockés dans des compartiments ou des référentiels Amazon S3. CodeCommit

Pour plus d'informations sur la prise en charge des navigateurs dans la AppStream version 2.0, consultez la section Configuration système requise et prise en charge des fonctionnalités (navigateur Web) dans le Guide d'administration Amazon AppStream 2.0. Si vous rencontrez des problèmes lors de l'utilisation de la AppStream version 2.0, consultez la section Résolution des AppStream problèmes liés à l'utilisation de la AppStreamversion 2.0 dans le Guide d'administration d'Amazon 2.0.

Ce document est destiné aux membres de l'équipe des opérations clients. Il explique comment configurer les flottes et les piles Amazon AppStream 2.0 pour héberger les outils Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer utilisés dans le cadre de la modernisation des AWS mainframes. Micro Focus Enterprise Analyzer est généralement utilisé pendant la phase d'évaluation et Micro Focus Enterprise Developer est généralement utilisé pendant la phase de migration et de modernisation de l'approche de modernisation du AWS mainframe. Si vous prévoyez d'utiliser à la fois Enterprise Analyzer et Enterprise Developer, vous devez créer des flottes et des piles distinctes pour chaque outil. Chaque outil nécessite son propre parc et sa propre pile car leurs conditions de licence sont différentes.

> **Important**
>
> Les étapes de ce didacticiel sont basées sur le AWS CloudFormation modèle téléchargeable cfn-m2 - .yml. appstream-fleet-ea-ed

Rubriques

- Nettoyage des ressources

Prérequis

- Téléchargez le modèle : cfn-m2- appstream-fleet-ea-ed .yml.
- Obtenez l'ID de votre VPC et de votre groupe de sécurité par défaut. Pour plus d'informations sur le VPC par défaut, consultez les VPC par défaut dans le Guide de l'utilisateur Amazon VPC. Pour plus d'informations sur le groupe de sécurité par défaut, consultez la section Groupes de sécurité par défaut et personnalisés dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- Assurez-vous de disposer des autorisations suivantes :
  - créez des piles, des flottes et des utilisateurs dans AppStream la version 2.0.
  - créez des piles à AWS CloudFormation l'aide d'un modèle.
  - créez des compartiments et chargez des fichiers vers des compartiments dans Amazon S3.
  - téléchargez les informations d'identification (`access_key_idetsecret_access_key`) depuis IAM.

Étape 1 : Obtenir les images AppStream 2.0

Au cours de cette étape, vous allez partager les images AppStream 2.0 pour Enterprise Analyzer et Enterprise Developer avec votre AWS compte.

1. Ouvrez la console AWS Mainframe Modernization à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le volet de navigation de gauche, choisissez Outils.
3. Dans Analyser, développer et créer des actifs, choisissez Partager des actifs avec mon AWS compte.

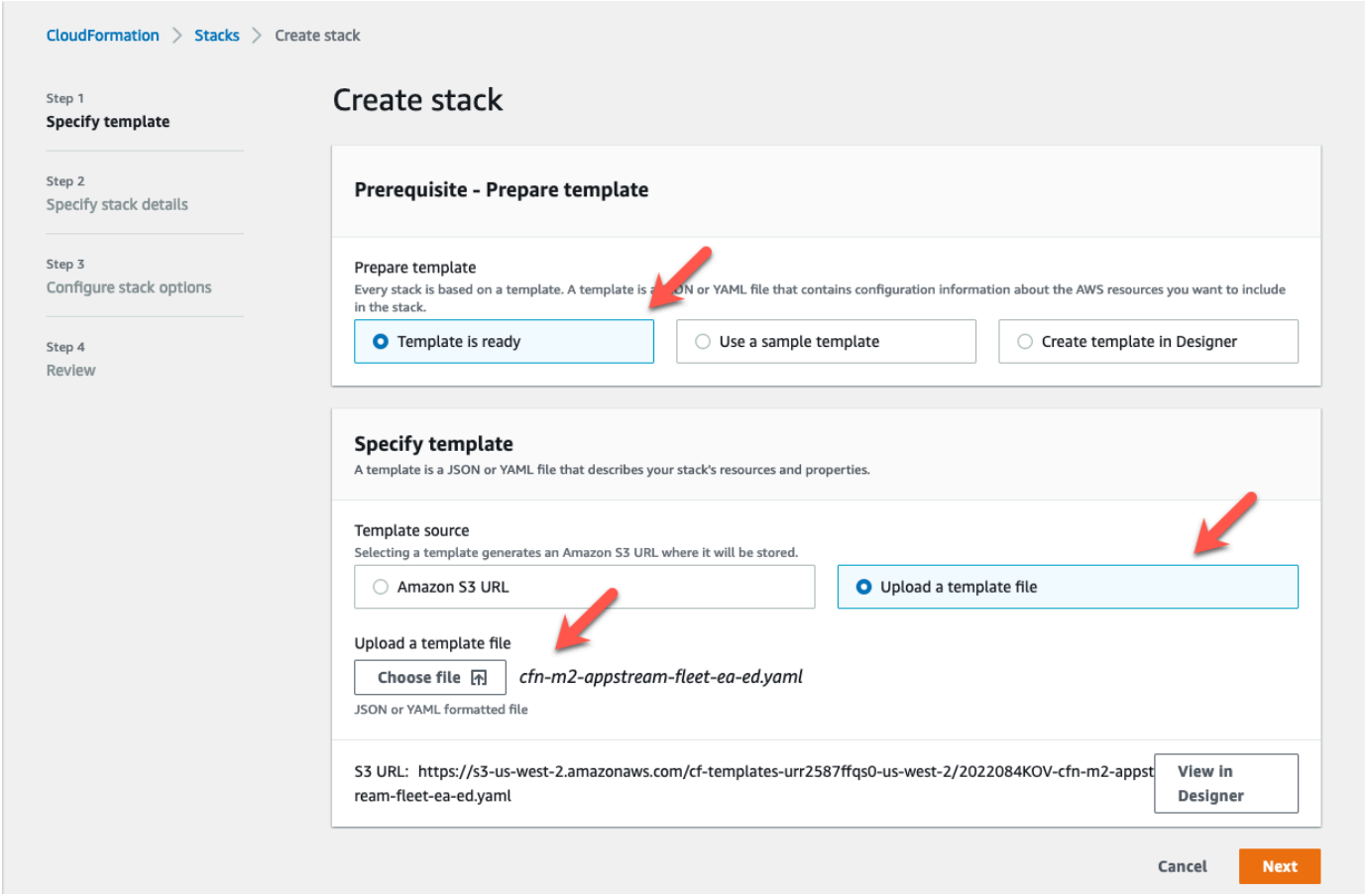
Étape 2 : créer la pile à l'aide du AWS CloudFormation modèle

Au cours de cette étape, vous allez utiliser le AWS CloudFormation modèle téléchargé pour créer une pile et un parc AppStream 2.0 pour exécuter Micro Focus Enterprise Analyzer. Vous pouvez répéter cette étape ultérieurement pour créer une autre pile et un autre parc AppStream 2.0 pour exécuter Micro Focus Enterprise Developer, étant donné que chaque outil nécessite son propre parc et sa

propre pile dans la AppStream version 2.0. Pour plus d'informations sur les AWS CloudFormation piles, consultez la section Utilisation des piles dans le Guide de l'AWS CloudFormationutilisateur.

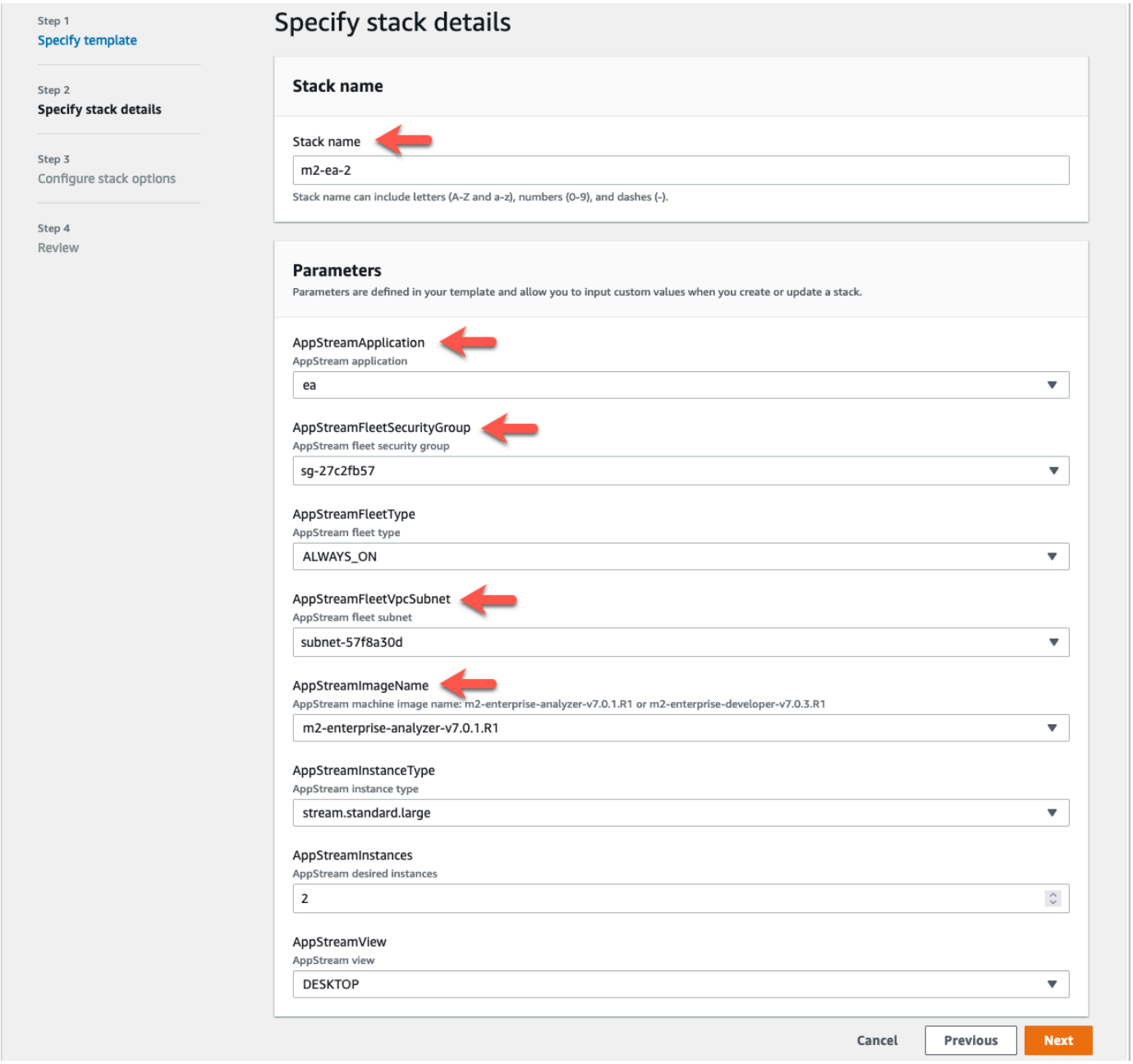
> **Note**
>
> AWSLa modernisation du mainframe ajoute des frais supplémentaires à la tarification standard AppStream 2.0 pour l'utilisation d'Enterprise Analyzer et d'Enterprise Developer. Pour plus d'informations, consultez la section Tarification de la modernisation des AWS mainframes.

1. Téléchargez le modèle cfn-m2- appstream-fleet-ea-ed .yml, si nécessaire.
2. Ouvrez la AWS CloudFormation console et choisissez Create Stack et avec de nouvelles ressources (standard).
3. Dans Prérequis - Préparer le modèle, choisissez Le modèle est prêt.
4. Dans Spécifier le modèle, choisissez Charger un fichier modèle.
5. Dans Charger un fichier modèle, choisissez Choisir un fichier et chargez le modèle cfn-m2-appstream-fleet-ea-ed .yml.
6. Choisissez Suivant.



7. Dans Spécifier les détails de la pile, entrez les informations suivantes :

   - Dans Nom de la pile, entrez le nom de votre choix. Par exemple, **`m2-ea`**.
   - Dans AppStreamApplication, choisissez le thé.
   - Dans AppStreamFleetSecurityGroup, choisissez le groupe de sécurité par défaut de votre VPC.
   - Dans AppStreamFleetVpcSubnet, choisissez un sous-réseau au sein de votre VPC par défaut.
   - Dans AppStreamImageName, choisissez l'image commençant par`m2-enterprise-analyzer`. Cette image contient la version actuellement prise en charge de l'outil Micro Focus Enterprise Analyzer.
   - Acceptez les valeurs par défaut pour les autres champs, puis choisissez Suivant.



8. Acceptez tous les paramètres par défaut, puis choisissez à nouveau Suivant.
9. Lors de la révision, assurez-vous que tous les paramètres correspondent à vos attentes.
10. Faites défiler la page vers le bas, choisissez Je reconnais qu'AWS CloudFormation peut créer des ressources IAM avec des noms personnalisés, puis choisissez Create Stack.

La création de la pile et de la flotte prend entre 20 et 30 minutes. Vous pouvez sélectionner Actualiser pour voir les AWS CloudFormation événements au fur et à mesure qu'ils se produisent.

Étape 3 : Création d'un utilisateur dans la AppStream version 2.0

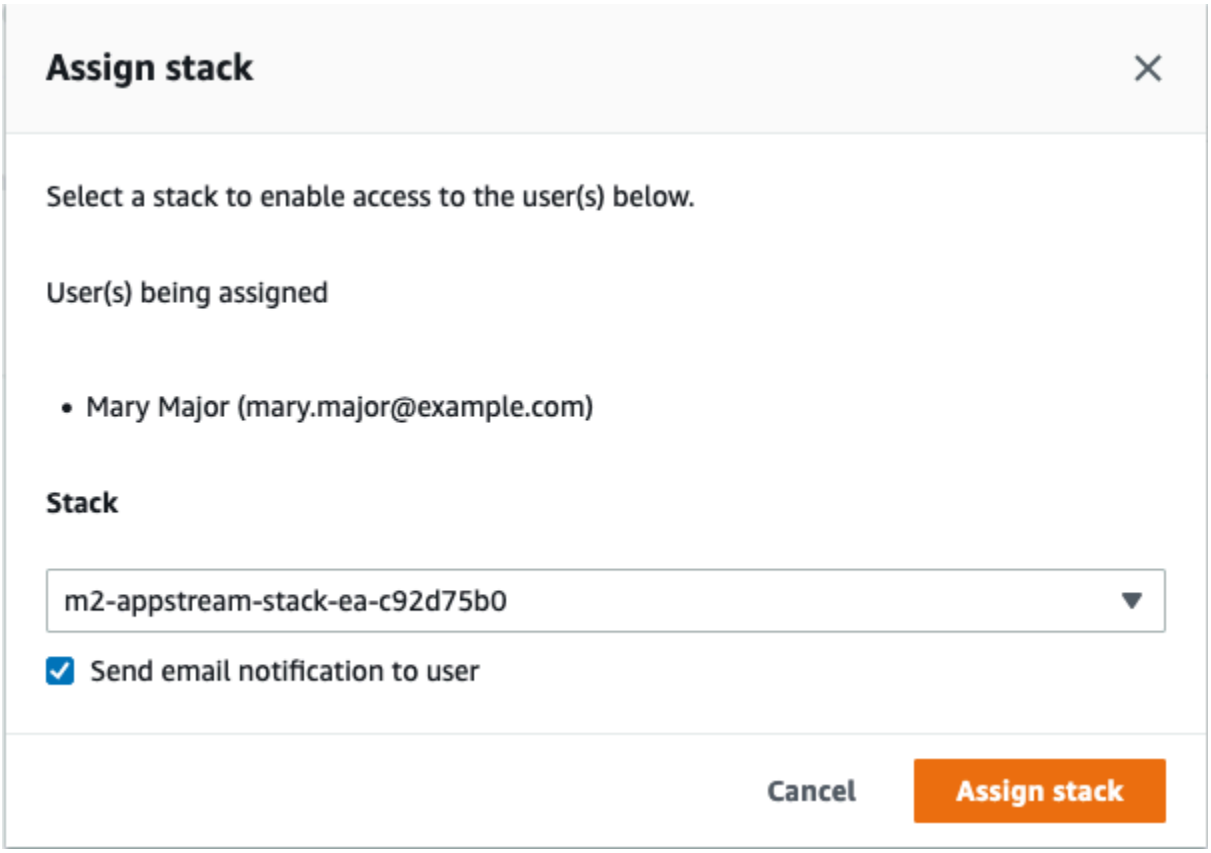
Pendant que vous attendez la fin AWS CloudFormation de la création de la pile, vous pouvez créer un ou plusieurs utilisateurs dans la AppStream version 2.0. Ces utilisateurs sont ceux qui utiliseront Enterprise Analyzer dans la AppStream version 2.0. Vous devez spécifier une adresse e-mail pour chaque utilisateur et vous assurer que chaque utilisateur dispose des autorisations suffisantes pour créer des compartiments dans Amazon S3, charger des fichiers dans un compartiment et créer un lien vers un compartiment pour cartographier son contenu.

1. Ouvrez la console AppStream 2.0.
2. Dans le volet de navigation de gauche, sélectionnez Groupe d'utilisateurs.
3. Choisissez Create user (Créer un utilisateur).
4. Indiquez une adresse e-mail à laquelle l'utilisateur peut recevoir une invitation par e-mail à utiliser la AppStream version 2.0, un prénom et un nom de famille, puis choisissez Créer un utilisateur.
5. Répétez l'opération si nécessaire pour créer d'autres utilisateurs. L'adresse e-mail de chaque utilisateur doit être unique.

Pour plus d'informations sur la création d'utilisateurs AppStream 2.0, consultez AppStream la section Groupes d'utilisateurs 2.0 dans le Guide d'administration Amazon AppStream 2.0.

Lorsque vous avez AWS CloudFormation terminé de créer la pile, vous pouvez attribuer l'utilisateur que vous avez créé à la pile, comme suit :

1. Ouvrez la console AppStream 2.0.
2. Choisissez le nom d'utilisateur.
3. Choisissez Action, puis Attribuer une pile.
4. Dans Attribuer une pile, choisissez la pile qui commence par `m2-appstream-stack-ea`.
5. Choisissez Assign stack.



L'affectation d'un utilisateur à une pile amène AppStream 2.0 à envoyer un e-mail à l'utilisateur à l'adresse que vous avez fournie. Cet e-mail contient un lien vers la page de connexion AppStream 2.0.

Étape 4 : Connectez-vous à la AppStream version 2.0

Au cours de cette étape, vous vous connectez à la AppStream version 2.0 à l'aide du lien figurant dans l'e-mail envoyé par la AppStream version 2.0 à l'utilisateur que vous avez créé[Étape 3 : Création d'un utilisateur dans la AppStream version 2.0](#).

1. Connectez-vous à la AppStream version 2.0 en utilisant le lien fourni dans l'e-mail envoyé par la AppStream version 2.0.
2. Modifiez votre mot de passe si vous y êtes invité. L'écran AppStream 2.0 qui s'affiche est similaire au suivant :

3. Choisissez Desktop.
4. Dans la barre des tâches, choisissez Rechercher et entrez **`D:`** pour accéder au dossier d'accueil.

> **Note**
>
> Si vous ignorez cette étape, vous risquez de recevoir un `Device not ready` message d'erreur lorsque vous essayez d'accéder au dossier de base.

À tout moment, si vous ne parvenez pas à vous connecter à la AppStream version AppStream 2.0, vous pouvez redémarrer votre flotte 2.0 et essayer de vous reconnecter en procédant comme suit.

1. Ouvrez la console AppStream 2.0.
2. Dans le menu de navigation de gauche, sélectionnez Flottes.
3. Choisissez la flotte que vous souhaitez utiliser.
4. Choisissez Action, puis sélectionnez Arrêter.
5. Attendez que la flotte s'arrête.
6. Choisissez Action, puis sélectionnez Démarrer.

Ce processus peut prendre environ 10 minutes.

Étape 5 : vérifier les compartiments dans Amazon S3 (facultatif)

L'une des tâches effectuées par le AWS CloudFormation modèle que vous avez utilisé pour créer la pile consistait à créer deux compartiments dans Amazon S3, nécessaires pour enregistrer et restaurer les données utilisateur et les paramètres de l'application au cours des sessions de travail. Ces godets sont les suivants :

- Le nom commence par`appstream2-`. Ce bucket mappe les données vers votre dossier personnel dans la AppStream version 2.0 (`D:\PhotonUser\My Files\Home Folder`).

  > **Note**
  >
  > Le dossier d'accueil est unique pour une adresse e-mail donnée et est partagé entre toutes les flottes et les piles d'un compte donnéAWS. Le nom du dossier de base est un hachage SHA256 de l'adresse e-mail de l'utilisateur et est stocké sur un chemin basé sur ce hachage.

- Le nom commence par`appstream-app-settings-`. Ce bucket contient les informations de session utilisateur pour la AppStream version 2.0 et inclut des paramètres tels que les favoris du navigateur, les profils de connexion à l'IDE et aux applications, ainsi que les personnalisations de l'interface utilisateur. Pour plus d'informations, consultez Comment fonctionne la persistance des paramètres d'application dans le Guide d'administration d'Amazon AppStream 2.0.

Pour vérifier que les compartiments ont été créés, procédez comme suit :

1. Ouvrez la console Amazon S3.
2. Dans le menu de navigation de gauche, choisissez Buckets.
3. Dans Rechercher des compartiments par nom, entrez **`appstream`** pour filtrer la liste.

Si vous voyez les compartiments, aucune autre action n'est nécessaire. Sachez simplement que les compartiments existent. Si vous ne voyez pas les compartiments, cela signifie que le AWS CloudFormation modèle n'a pas fini de s'exécuter ou qu'une erreur s'est produite. Accédez à la AWS CloudFormation console et consultez les messages de création de pile.

Étapes suivantes

Maintenant que l'infrastructure AppStream 2.0 est configurée, vous pouvez configurer Enterprise Analyzer et commencer à l'utiliser. Pour plus d'informations, veuillez consulter Tutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream . Vous pouvez également configurer Enterprise Developer. Pour plus d'informations, veuillez consulter Tutoriel : Configuration de Micro Focus Enterprise Developer sur AppStream 2.0.

Nettoyage des ressources

La procédure de nettoyage de la pile et des flottes créées est décrite dans Créer une flotte et une pile AppStream 2.0.

Lorsque les objets AppStream 2.0 ont été supprimés, l'administrateur du compte peut également, le cas échéant, nettoyer les compartiments Amazon S3 pour les paramètres de l'application et les dossiers personnels.

> **Note**
>
> Le dossier d'accueil d'un utilisateur donné étant unique pour toutes les flottes, vous devrez peut-être le conserver si d'autres piles AppStream 2.0 sont actives sur le même compte.

Enfin, la AppStream version 2.0 ne permet pas actuellement de supprimer des utilisateurs à l'aide de la console. Vous devez plutôt utiliser l'API du service avec l'interface de ligne de commande. Pour plus d'informations, consultez la section Administration du pool d'utilisateurs dans le Guide d'administration Amazon AppStream 2.0.

Tutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream

Ce didacticiel explique comment configurer Micro Focus Enterprise Analyzer pour analyser une ou plusieurs applications mainframe. L'outil Enterprise Analyzer fournit plusieurs rapports basés sur son analyse du code source de l'application et des définitions du système.

Cette configuration est conçue pour favoriser la collaboration au sein des équipes. L'installation utilise un compartiment Amazon S3 pour partager le code source avec des disques virtuels. Pour ce faire,

vous devez utiliser Rclone) sur l'ordinateur Windows. Avec une instance Amazon RDS commune exécutant PostgreSQL, tous les membres de l'équipe peuvent accéder à tous les rapports demandés.

Les membres de l'équipe peuvent également monter le disque virtuel sauvegardé par Amazon S3 sur leurs machines personnelles et mettre à jour le compartiment source depuis leur poste de travail. Ils peuvent potentiellement utiliser des scripts ou toute autre forme d'automatisation sur leurs machines s'ils sont connectés à d'autres systèmes internes sur site.

La configuration est basée sur les images Windows AppStream 2.0 que AWS Mainframe Modernization partage avec le client. La configuration repose également sur la création de flottes et de piles AppStream 2.0, comme décrit dans. Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer

> **Important**
>
> Les étapes de ce didacticiel supposent que vous avez configuré la AppStream version 2.0 avec le AWS CloudFormation modèle téléchargeable cfn-m2- appstream-fleet-ea-ed .yml. Pour plus d'informations, consultez Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer. Pour effectuer les étapes de ce didacticiel, vous devez avoir configuré votre parc et votre stack Enterprise Analyzer et ils doivent être en cours d'exécution.

Pour une description complète des fonctionnalités et des livrables d'Enterprise Analyzer, consultez la documentation d'Enterprise Analyzer sur le site Web de Micro Focus.

Contenu de l'image

Outre l'application Enterprise Analyzer elle-même, l'image contient les outils et bibliothèques suivants.

Outils tiers

- Python
- Recloner
- pgAdmin
- git-scm
- pilote ODBC pour PostgreSQL

Bibliothèques de `C:\Users\Public`

- BankDemo code source et définition du projet pour Enterprise Developer :`m2-bankdemo-template.zip`.
- Package d'installation MFA pour le mainframe :. `mfa.zip` Pour plus d'informations, consultez la section Présentation de l'accès au mainframe dans la documentation Micro Focus Enterprise Developer.
- Fichiers de commande et de configuration pour Rclone (instructions pour leur utilisation dans les didacticiels) : `m2-rclone.cmd` et`m2-rclone.conf`.

Rubriques

Prérequis

- Téléchargez le code source et les définitions du système pour l'application client que vous souhaitez analyser dans un compartiment S3. Les définitions du système incluent CICS CSD, les définitions d'objets DB2, etc. Vous pouvez créer une structure de dossiers dans le compartiment adaptée à la manière dont vous souhaitez organiser les artefacts de l'application. Par exemple, lorsque vous décompressez l' BankDemo échantillon, sa structure est la suivante :

```
demo
     |--> jcl
     |--> RDEF
     |--> transaction
     |--> xa
```

- Créez et démarrez une instance Amazon RDS exécutant PostgreSQL. Cette instance stockera les données et les résultats produits par Enterprise Analyzer. Vous pouvez partager cette instance

avec tous les membres de l'équipe chargée de l'application. Créez également un schéma vide appelé `m2_ea` (ou tout autre nom approprié) dans la base de données. Définissez les informations d'identification pour les utilisateurs autorisés qui leur permettent de créer, d'insérer, de mettre à jour et de supprimer des éléments dans ce schéma. Vous pouvez obtenir le nom de la base de données, l'URL du point de terminaison du serveur et le port TCP depuis la console Amazon RDS ou auprès de l'administrateur du compte.

- Assurez-vous d'avoir configuré l'accès programmatique à votreCompte AWS. Pour plus d'informations, consultez la section Accès par programmation dans leRéférence générale d'Amazon Web Services.

Étape 1 : configuration

1. Démarrez une session avec la AppStream version 2.0 avec l'URL que vous avez reçue dans le message électronique de bienvenue de la AppStream version 2.0.
2. Utilisez votre adresse e-mail comme nom d'utilisateur et définissez votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Analyzer.
4. Sur la page du menu AppStream 2.0, choisissez Bureau pour accéder au bureau Windows que le parc diffuse.

Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows

> **Note**
>
> Si vous avez déjà utilisé Rclone lors de la version préliminaire de AWS Mainframe Modernization, vous devez effectuer la mise à jour `m2-rclone.cmd` vers la version la plus récente située dans. `C:\Users\Public`

1. Copiez les `m2-rclone.cmd` fichiers `m2-rclone.conf` et fournis dans votre dossier `C:\Users\Public` de base à `C:\Users\PhotonUser\My Files\Home Folder` l'aide de l'explorateur de fichiers.
2. Mettez à jour les paramètres de `m2-rclone.conf` configuration avec votre clé AWS d'accès et le secret correspondant, ainsi que votreRégion AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Dans `m2-rclone.cmd`, effectuez les modifications suivantes :

   - `your-s3-bucket`Changez le nom de votre compartiment Amazon S3. Par exemple, `m2-s3-mybucket`.
   - Passez `your-s3-folder-key` à la clé de votre compartiment Amazon S3. Par exemple, `myProject`.
   - Accédez `your-local-folder-path` au chemin du répertoire dans lequel vous souhaitez que les fichiers d'application soient synchronisés à partir du compartiment Amazon S3 qui les contient. Par exemple, `D:\PhotonUser\My Files\Home Folder\m2-new`. Ce répertoire synchronisé doit être un sous-répertoire du dossier d'accueil pour que la AppStream version 2.0 puisse le sauvegarder et le restaurer correctement au début et à la fin de la session.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:your-s3-bucket/your-s3-folder-
key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Ouvrez une invite de commande Windows, envoyez-la `C:\Users\PhotonUser\My Files\Home Folder` si nécessaire et lancez-la`m2-rclone.cmd`. Ce script de commande exécute une boucle continue, synchronisant votre compartiment et votre clé Amazon S3 avec le dossier local toutes les 10 secondes. Vous pouvez ajuster le délai d'attente selon vos besoins. Vous devriez voir le code source de l'application situé dans le compartiment Amazon S3 de l'Explorateur de fichiers Windows.

Pour ajouter de nouveaux fichiers à l'ensemble sur lequel vous travaillez ou pour mettre à jour les fichiers existants, chargez les fichiers dans le compartiment Amazon S3 et ils seront synchronisés

avec votre répertoire lors de la prochaine itération définie dans. `m2-rclone.cmd` De même, si vous souhaitez supprimer certains fichiers, supprimez-les du compartiment Amazon S3. La prochaine opération de synchronisation les supprimera de votre répertoire local.

Étape 3 : créer une source ODBC pour l'instance Amazon RDS

1. Pour démarrer l'outil EA_Admin, accédez au menu de sélection d'applications dans le coin supérieur gauche de la fenêtre du navigateur et choisissez MF EA_Admin.
2. Dans le menu Administrer, choisissez Sources de données ODBC, puis choisissez Ajouter dans l'onglet DSN utilisateur.
3. Dans la boîte de dialogue Créer une nouvelle source de données, choisissez le pilote Unicode PostgreSQL, puis cliquez sur Terminer.
4. Dans la boîte de dialogue de configuration du pilote ODBC Unicode PostgreSQL (PSQLodBC), définissez et notez le nom de la source de données que vous souhaitez. Complétez les paramètres suivants avec les valeurs de l'instance RDS que vous avez créée précédemment :

   Description

   Description facultative pour vous aider à identifier rapidement cette connexion à la base de données.

   Base de données

   La base de données Amazon RDS que vous avez créée précédemment.

   Serveur

   Le point de terminaison Amazon RDS.

   Port

   Le port Amazon RDS.

   Nom utilisateur

   Tel que défini dans l'instance Amazon RDS.

   Mot de passe

   Tel que défini dans l'instance Amazon RDS.

5. Choisissez Test pour vérifier que la connexion à Amazon RDS est réussie, puis sélectionnez Enregistrer pour enregistrer votre nouveau DSN utilisateur.

6. Attendez de voir le message confirmant la création de l'espace de travail approprié, puis cliquez sur OK pour terminer avec les sources de données ODBC et fermer l'outil EA_Admin.
7. Accédez à nouveau au menu de sélection d'applications, puis choisissez Enterprise Analyzer pour démarrer l'outil. Choisissez Créer un nouveau.
8. Dans la fenêtre de configuration de l'espace de travail, entrez le nom de votre espace de travail et définissez son emplacement. L'espace de travail peut être le disque basé sur Amazon S3 si vous travaillez sous cette configuration, ou votre dossier personnel si vous préférez.
9. Choisissez Choose Other Database pour vous connecter à votre instance Amazon RDS.
10. Choisissez l'icône Postgre parmi les options, puis cliquez sur OK.
11. Pour les paramètres Windows, sous Options — Définir les paramètres de connexion, entrez le nom de la source de données que vous avez créée. Entrez également le nom de la base de données, le nom du schéma, le nom d'utilisateur et le mot de passe. Choisissez OK.
12. Attendez qu'Enterprise Analyzer crée toutes les tables, tous les index, etc. dont il a besoin pour stocker les résultats. Ce processus peut prendre quelques minutes. Enterprise Analyzer confirme le moment où la base de données et l'espace de travail sont prêts à être utilisés.
13. Accédez à nouveau au menu de sélection d'applications et choisissez Enterprise Analyzer pour démarrer l'outil.
14. La fenêtre de démarrage d'Enterprise Analyzer apparaît dans le nouvel emplacement d'espace de travail sélectionné. Choisissez OK.
15. Accédez à votre référentiel dans le volet de gauche, sélectionnez le nom du référentiel, puis choisissez Ajouter des fichiers/dossiers à votre espace de travail. Sélectionnez le dossier dans lequel le code de votre application est stocké pour l'ajouter à l'espace de travail. Vous pouvez utiliser l' BankDemoexemple de code précédent si vous le souhaitez. Lorsque Enterprise Analyzer vous invite à vérifier ces fichiers, choisissez Verify pour démarrer le rapport de vérification initial d'Enterprise Analyzer. Le traitement peut prendre quelques minutes, en fonction de la taille de votre demande.
16. Développez votre espace de travail pour afficher les fichiers et dossiers que vous y avez ajoutés. Les types d'objets et les rapports de complexité cyclomatique sont également visibles dans le quadrant supérieur du volet Chart Viewer.

Vous pouvez désormais utiliser Enterprise Analyzer pour toutes les tâches nécessaires.

Sessions suivantes

1. Démarrez une session avec la AppStream version 2.0 avec l'URL que vous avez reçue dans le message électronique de bienvenue de la AppStream version 2.0.
2. Connectez-vous à l'aide de votre e-mail et de votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Analyzer.
4. Lancez `Rclone` pour vous connecter au disque sauvegardé par Amazon S3 si vous utilisez cette option pour partager les fichiers de l'espace de travail.
5. Lancez Enterprise Analyzer pour effectuer vos tâches.

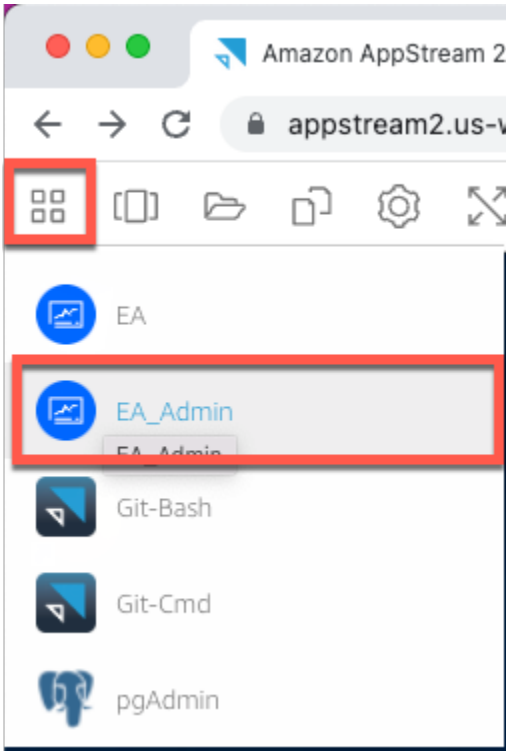
Résolution des problèmes de connexion aux espaces

Lorsque vous essayez de vous reconnecter à votre espace de travail Enterprise Analyzer, le message d'erreur suivant peut s'afficher :

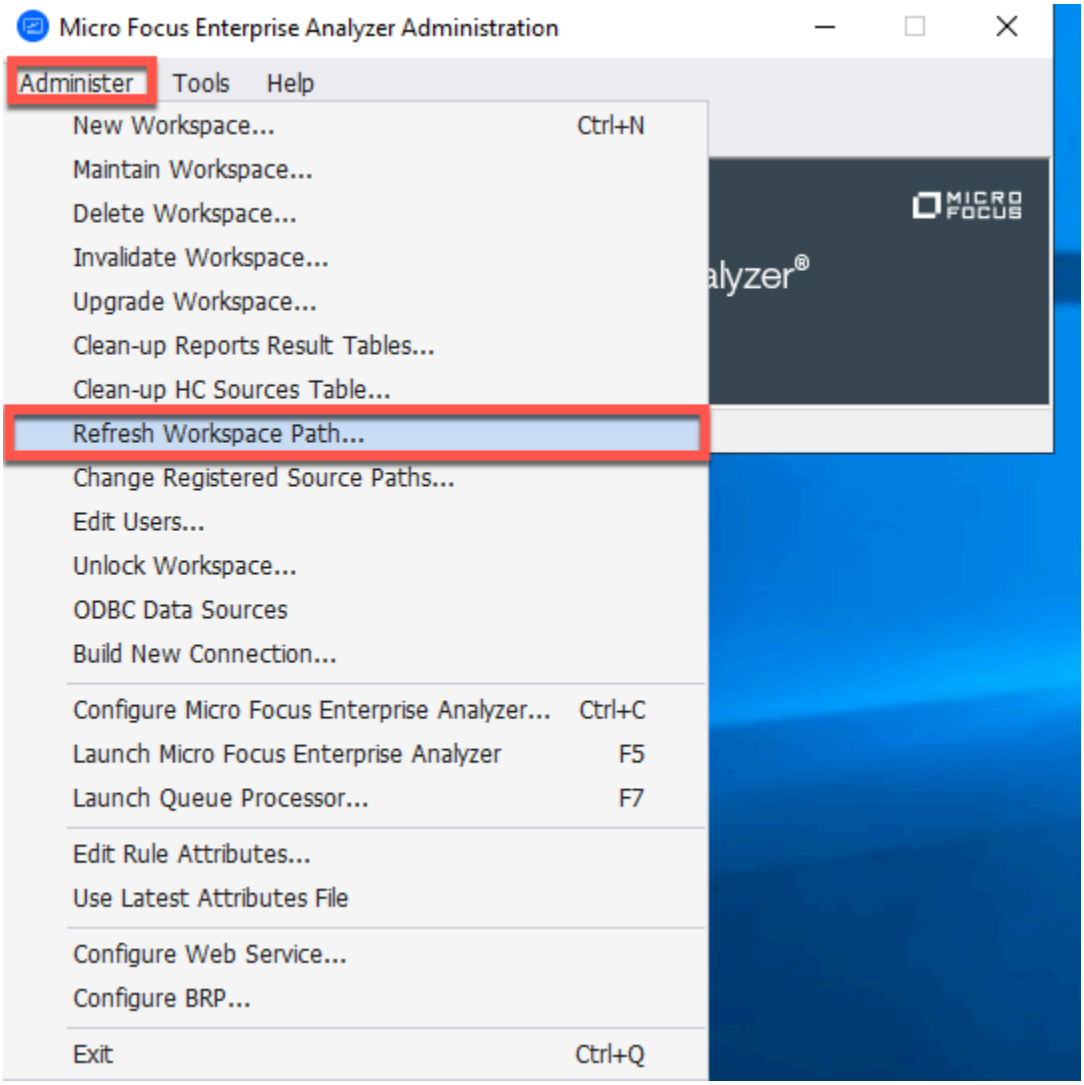
```
Cannot access the workspace directory D:\PhotonUser\My Files\Home Folder\EA_BankDemo.
 The workspace has been created on a non-shared disk of the EC2AMAZ-E6LC33H computer.
 Would you like to correct the workspace directory location?
```

Pour résoudre ce problème, cliquez sur OK pour effacer le message, puis effectuez les étapes suivantes.

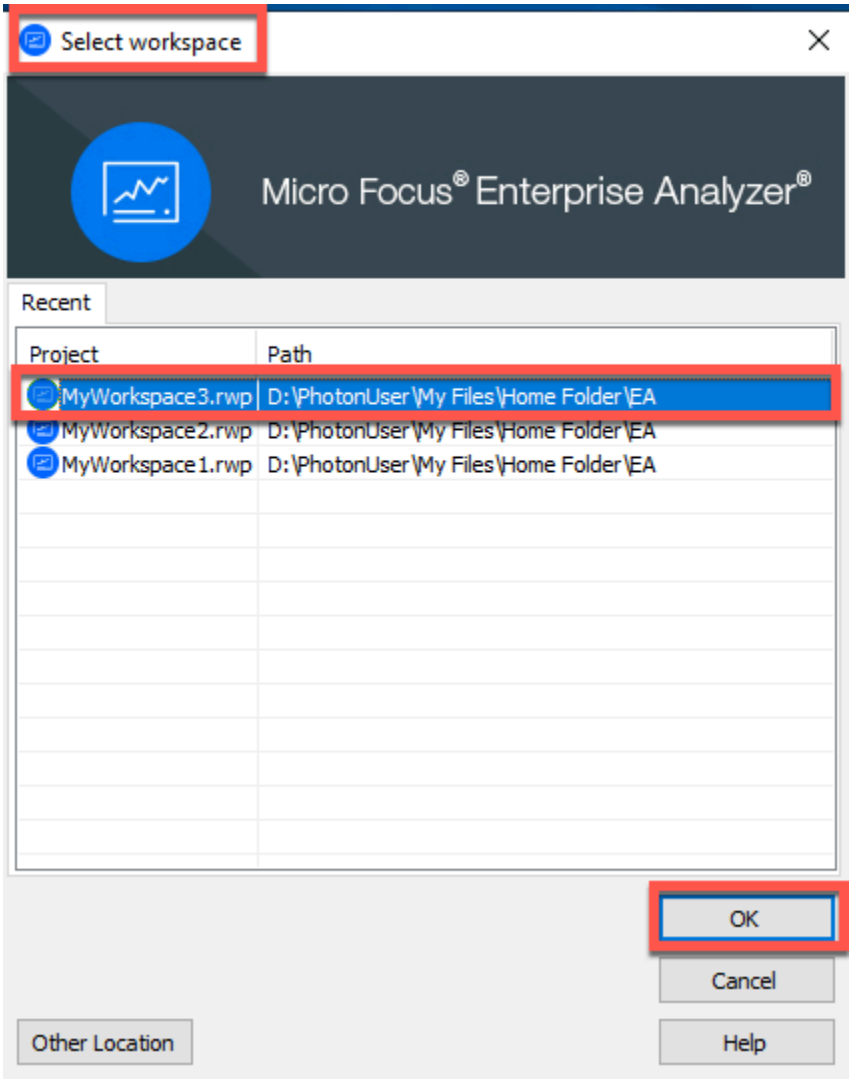
1. Dans la AppStream version 2.0, cliquez sur l'icône Lancer l'application dans la barre d'outils, puis choisissez EA_Admin pour démarrer l'outil d'administration de Micro Focus Enterprise Analyzer.



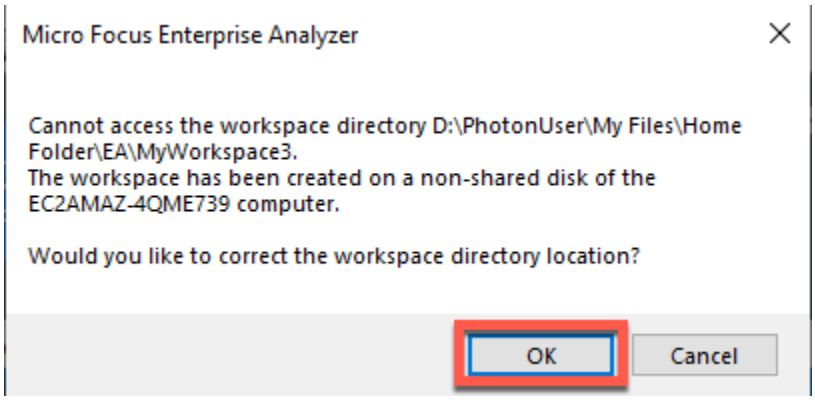
2. Dans le menu Administrer, choisissez Actualiser le chemin de l'espace de travail... .



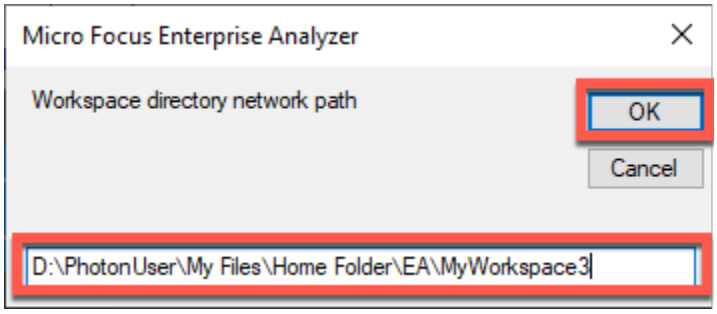
3. Sous Sélectionner un espace de travail, choisissez l'espace de travail de votre choix, puis cliquez sur OK.



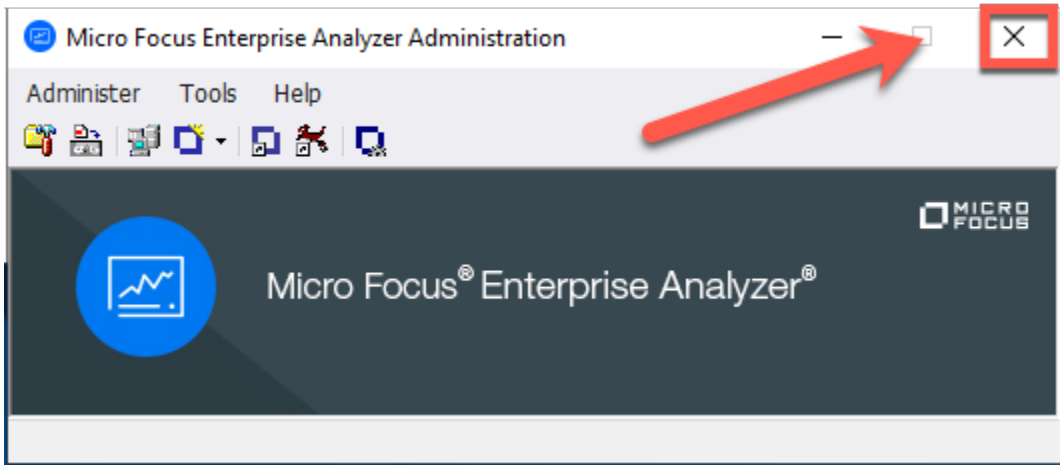
4. Cliquez sur OK pour confirmer le message d'erreur.



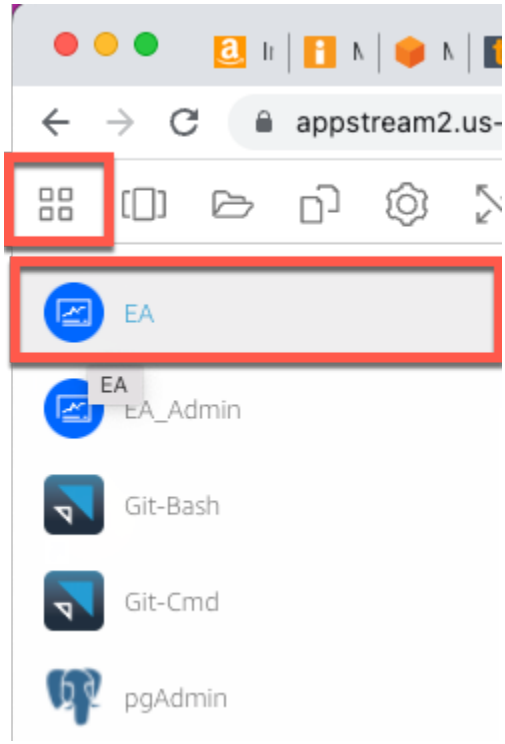
5. Sous Chemin réseau du répertoire de l'espace de travail, entrez le chemin d'accès correct à votre espace de travail, par exemple,`D:\PhotonUser\My Files\Home Folder\EA\MyWorkspace3`.



6. Fermez l'outil d'administration de Micro Focus Enterprise Analyzer.



7. Dans la AppStream version 2.0, cliquez sur l'icône Lancer l'application dans la barre d'outils, puis choisissez EA pour démarrer Micro Focus Enterprise Analyzer.



8. Répétez les étapes 3 à 5.

Micro Focus Enterprise Analyzer devrait maintenant s'ouvrir avec l'espace de travail existant.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin de ne pas avoir à payer de frais supplémentaires. Procédez comme suit :

- Utilisez l'outil EA_Admin pour supprimer l'espace de travail.
- Supprimez les compartiments S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur Amazon S3.
- Supprimez la base de données que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Suppression d'une instance de base de données.

Tutoriel : Configuration de Micro Focus Enterprise Developer sur AppStream 2.0

Ce didacticiel explique comment configurer Micro Focus Enterprise Developer pour une ou plusieurs applications mainframe afin de les maintenir, de les compiler et de les tester à l'aide des fonctionnalités d'Enterprise Developer. La configuration est basée sur les images Windows AppStream 2.0 que AWS Mainframe Modernization partage avec le client et sur la création de flottes et de piles AppStream 2.0, comme décrit dans. Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer

> **Important**
>
> Les étapes de ce didacticiel supposent que vous configurez la AppStream version 2.0 à l'aide du AWS CloudFormation modèle téléchargeable cfn-m2- appstream-fleet-ea-ed .yaml. Pour plus d'informations, consultez Tutoriel : Configuration de la AppStream version 2.0 pour une utilisation avec Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer.
> Vous devez effectuer les étapes de cette configuration lorsque le parc et le stack d'Enterprise Developer sont opérationnels.

Pour une description complète des fonctionnalités et des livrables d'Enterprise Developer v7, consultez sa documentation up-to-date en ligne (v7.0) sur le site Micro Focus.

Contenu de l'image

Outre Enterprise Developer lui-même, l'image contient l'image contenant Rumba (un émulateur TN3270). Il contient également les outils et bibliothèques suivants.

Outils tiers

- Python
- Recloner
- pgAdmin
- git-scm
- pilote ODBC pour PostgreSQL

Bibliothèques de `C:\Users\Public`

- BankDemo code source et définition du projet pour Enterprise Developer :`m2-bankdemo-template.zip`.
- Package d'installation MFA pour le mainframe :. `mfa.zip` Pour plus d'informations, consultez la section Présentation de l'accès au mainframe dans la documentation Micro Focus Enterprise Developer.
- Fichiers de commande et de configuration pour Rclone (instructions pour leur utilisation dans les didacticiels) : `m2-rclone.cmd` et`m2-rclone.conf`.

Si vous devez accéder à du code source qui n'est pas encore chargé dans CodeCommit les référentiels, mais qui est disponible dans un compartiment Amazon S3, par exemple pour effectuer le chargement initial du code source dans git, suivez la procédure de création d'un disque Windows virtuel comme décrit dansTutoriel : Configuration d'Enterprise Analyzer sur la version 2.0 AppStream .

Rubriques

- Prérequis
- Étape 1 : Configuration par les utilisateurs individuels d'Enterprise Developer
- Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows (facultatif)

Prérequis

- Un ou plusieurs CodeCommit référentiels chargés avec le code source de l'application à maintenir. La configuration du référentiel doit correspondre aux exigences du pipeline CI/CD ci-dessus afin de créer des synergies en combinant les deux outils.
- Chaque utilisateur doit disposer d'informations d'identification pour le CodeCommit ou les référentiels définis par l'administrateur du compte conformément aux informations de la section Authentification et contrôle d'accès pour AWS CodeCommit. La structure de ces informations d'identification est examinée dans Authentification et contrôle d'accès pour AWS CodeCommit et la référence complète pour les autorisations IAM CodeCommit se trouve dans la référence des CodeCommit autorisations : l'administrateur peut définir des politiques IAM distinctes pour des rôles distincts en ayant des informations d'identification spécifiques au rôle de chaque référentiel et en limitant les autorisations de l'utilisateur à l'ensemble de tâches spécifiques qu'il doit accomplir sur un référentiel donné. Ainsi, pour chaque responsable du CodeCommit référentiel, l'administrateur du compte créera un utilisateur principal et accordera à cet utilisateur l'autorisation d'accéder au ou aux référentiels requis en sélectionnant la ou les politiques IAM appropriées pour l'accès. CodeCommit

Étape 1 : Configuration par les utilisateurs individuels d'Enterprise Developer

1. Obtenez vos informations d'identification IAM :

    1. Connectez-vous à la AWS console à l'adresse https://console.aws.amazon.com/iam/.
    2. Suivez la procédure décrite à l'étape 3 de Configuration pour les utilisateurs HTTPS à l'aide des informations d'identification Git dans le guide de AWS CodeCommit l'utilisateur.
    3. Copiez les informations de CodeCommit connexion spécifiques qu'IAM a générées pour vous, soit en affichant, copiant puis collant ces informations dans un fichier sécurisé sur votre ordinateur local, soit en choisissant Télécharger les informations d'identification pour télécharger ces informations sous forme de fichier .CSV. Vous avez besoin de ces informations pour vous connecter à CodeCommit.

2. Démarrez une session avec AppStream 2.0 en fonction de l'URL reçue dans l'e-mail de bienvenue. Utilisez votre adresse e-mail comme nom d'utilisateur et créez votre mot de passe.
3. Sélectionnez votre stack de développeurs d'entreprise.
4. Sur la page de menu, choisissez Desktop pour accéder au bureau Windows diffusé par le parc.

Étape 2 : créer le dossier virtuel basé sur Amazon S3 sous Windows (facultatif)

Si Rclone est nécessaire (voir ci-dessus), créez le dossier virtuel basé sur Amazon S3 sous Windows : (facultatif si tous les artefacts de l'application proviennent CodeCommit exclusivement de l'accès).

> **Note**
>
> Si vous avez déjà utilisé Rclone lors de la version préliminaire de AWS Mainframe Modernization, vous devez passer `m2-rclone.cmd` à la version la plus récente située dans. `C:\Users\Public`

1. Copiez les `m2-rclone.cmd` fichiers `m2-rclone.conf` et fournis dans votre dossier `C:\Users\Public` de base à `C:\Users\PhotonUser\My Files\Home Folder` l'aide de l'explorateur de fichiers.
2. Mettez à jour les paramètres de `m2-rclone.conf` configuration avec votre clé AWS d'accès et le secret correspondant, ainsi que votreRégion AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Dans `m2-rclone.cmd`, effectuez les modifications suivantes :

   - `your-s3-bucket`Changez le nom de votre compartiment Amazon S3. Par exemple, `m2-s3-mybucket`.

- Passez `your-s3-folder-key` à la clé de votre compartiment Amazon S3. Par exemple, `myProject`.
- Accédez `your-local-folder-path` au chemin du répertoire dans lequel vous souhaitez que les fichiers d'application soient synchronisés à partir du compartiment Amazon S3 qui les contient. Par exemple, `D:\PhotonUser\My Files\Home Folder\m2-new`. Ce répertoire synchronisé doit être un sous-répertoire du dossier d'accueil pour que la AppStream version 2.0 puisse le sauvegarder et le restaurer correctement au début et à la fin de la session.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:your-s3-bucket/your-s3-folder-
key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Ouvrez une invite de commande Windows, envoyez-la `C:\Users\PhotonUser\My Files \Home Folder` si nécessaire et lancez-la`m2-rclone.cmd`. Ce script de commande exécute une boucle continue, synchronisant votre compartiment et votre clé Amazon S3 avec le dossier local toutes les 10 secondes. Vous pouvez ajuster le délai d'attente selon vos besoins. Vous devriez voir le code source de l'application situé dans le compartiment Amazon S3 de l'Explorateur de fichiers Windows.

Pour ajouter de nouveaux fichiers à l'ensemble sur lequel vous travaillez ou pour mettre à jour les fichiers existants, téléchargez les fichiers dans le compartiment Amazon S3 et ils seront synchronisés avec votre répertoire lors de la prochaine itération définie dans. `m2-rclone.cmd` De même, si vous souhaitez supprimer certains fichiers, supprimez-les du compartiment Amazon S3. La prochaine opération de synchronisation les supprimera de votre répertoire local.

Étape 3 : cloner le dépôt

1. Accédez au menu de sélection d'applications dans le coin supérieur gauche de la fenêtre du navigateur et sélectionnez Enterprise Developer.
2. Terminez la création de l'espace de travail requise par Enterprise Developer dans votre dossier principal en choisissant `C:\Users\PhotonUser\My Files\Home Folder` (alias`D: \PhotonUser\My Files\Home Folder`) comme emplacement pour l'espace de travail.

3. Dans Enterprise Developer, clonez votre CodeCommit dépôt en accédant à l'explorateur de projets, en cliquant avec le bouton droit de la souris et en choisissant Importer, Importer..., Git, Projects from Git Clone URI. Entrez ensuite vos informations de CodeCommit connexion spécifiques et complétez la boîte de dialogue Eclipse pour importer le code.

Le dépôt CodeCommit git est désormais cloné dans votre espace de travail local.

Votre espace de travail Enterprise Developer est maintenant prêt à démarrer les travaux de maintenance de votre application. Vous pouvez notamment utiliser l'instance locale de Microfocus Enterprise Server (ES) intégrée à Enterprise Developer pour déboguer et exécuter votre application de manière interactive afin de valider vos modifications localement.

> **Note**
>
> L'environnement de développement d'entreprise local, y compris l'instance locale de serveur d'entreprise, s'exécute sous Windows tandis que AWS Mainframe Modernization s'exécute sous Linux. Nous vous recommandons d'exécuter des tests complémentaires dans l'environnement Linux fourni par AWS Mainframe Modernization après avoir validé CodeCommit et reconstruit la nouvelle application pour cette cible et avant de déployer la nouvelle application en production.

Sessions suivantes

Lorsque vous sélectionnez un dossier géré par la AppStream version 2.0, tel que le dossier de base pour le clonage de votre CodeCommit dépôt, il sera enregistré et restauré de manière transparente d'une session à l'autre. Procédez comme suit la prochaine fois que vous aurez besoin de travailler avec l'application :

1. Démarrez une session avec AppStream 2.0 en fonction de l'URL reçue dans l'e-mail de bienvenue.
2. Connectez-vous avec votre e-mail et votre mot de passe permanent.
3. Sélectionnez la pile Enterprise Developer.
4. Lancez `Rclone` pour vous connecter (voir ci-dessus) au disque sauvegardé par Amazon S3 lorsque cette option est utilisée pour partager les fichiers de l'espace de travail.
5. Lancez Enterprise Developer pour faire votre travail.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin qu'elles ne continuent pas à vous être facturées. Procédez comme suit :

- Supprimez le CodeCommit référentiel que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un CodeCommit référentiel dans le Guide de AWS CodeCommit l'utilisateur.
- Supprimez la base de données que vous avez créée pour ce didacticiel. Pour plus d'informations, consultez Suppression d'une instance de base de données.

Configuration de l'automatisation pour les sessions de streaming Micro Focus Enterprise Analyzer et Micro Focus Enterprise Developer

Vous pouvez exécuter automatiquement un script au début et à la fin de la session pour permettre une automatisation spécifique au contexte de votre client. Pour plus d'informations sur cette fonctionnalité AppStream 2.0, consultez la section Utiliser des scripts de session pour gérer l'expérience de streaming de vos utilisateurs AppStream 2.0 dans le Guide d'administration d'Amazon AppStream 2.0.

Cette fonctionnalité nécessite que vous disposiez au moins des versions suivantes des images Enterprise Analyzer et Enterprise Developer :

- `m2-enterprise-analyzer-v8.0.4.R1`
- `m2-enterprise-developer-v8.0.4.R1`

Rubriques

- Configurer l'automatisation au démarrage de la session
- Configurer l'automatisation à la fin de la session

Configurer l'automatisation au démarrage de la session

Si vous souhaitez exécuter un script d'automatisation lorsque les utilisateurs se connectent à la AppStream version 2.0, créez votre script et nommez-lem2`-user-setup.cmd`. Stockez le script

dans le dossier d'accueil AppStream 2.0 de l'utilisateur. Les images AppStream 2.0 fournies parAWS Mainframe Modernization recherchent un script portant ce nom à cet emplacement et l'exécutent s'il existe.

> **Note**
>
> La durée du script ne peut pas dépasser la limite définie par la AppStream version 2.0, qui est actuellement de 60 secondes. Pour plus d'informations, consultez Exécuter des scripts avant le début des sessions de streaming dans le Guide d'administration d'Amazon AppStream 2.0.

Configurer l'automatisation à la fin de la session

Si vous souhaitez exécuter un script d'automatisation lorsque les utilisateurs se déconnectent de la AppStream version 2.0, créez votre script et nommez-le`m2-user-teardown.cmd`. Stockez le script dans le dossier d'accueil AppStream 2.0 de l'utilisateur. Les images AppStream 2.0 fournies parAWS Mainframe Modernization recherchent un script portant ce nom à cet emplacement et l'exécutent s'il existe.

> **Note**
>
> La durée du script ne peut pas dépasser la limite définie par la AppStream version 2.0, qui est actuellement de 60 secondes. Pour plus d'informations, consultez Exécuter des scripts après la fin des sessions de streaming dans le Guide d'administration d'Amazon AppStream 2.0.

Afficher les ensembles de données sous forme de tables et de colonnes dans Enterprise Developer

Vous pouvez accéder aux ensembles de données du mainframe déployés dans AWS Mainframe Modernization à l'aide du moteur d'exécution Micro Focus. Vous pouvez afficher les ensembles de données migrés sous forme de tables et de colonnes à partir d'une instance de Micro Focus Enterprise Developer. L'affichage des ensembles de données de cette façon vous permet de :

- Effectuez `SQL SELECT` des opérations sur les fichiers de données migrés.
- Exposez les données en dehors de l'application mainframe migrée sans modifier l'application.
- Filtrez facilement les données et enregistrez-les au format CSV ou dans un autre format de fichier.

> **Note**
>
> Les étapes 1 et 2 sont des activités ponctuelles. Répétez les étapes 3 et 4 pour chaque ensemble de données afin de créer les vues de base de données.

Rubriques

Prérequis

- Vous devez avoir accès à Micro Focus Enterprise Developer Desktop via la AppStream version 2.0.
- Une application doit être déployée et exécutée dans le cadre de la modernisation du AWS mainframe à l'aide du moteur d'exécution Micro Focus.
- Vous stockez les données de votre application dans Aurora PostgreSQL Compatible Edition.

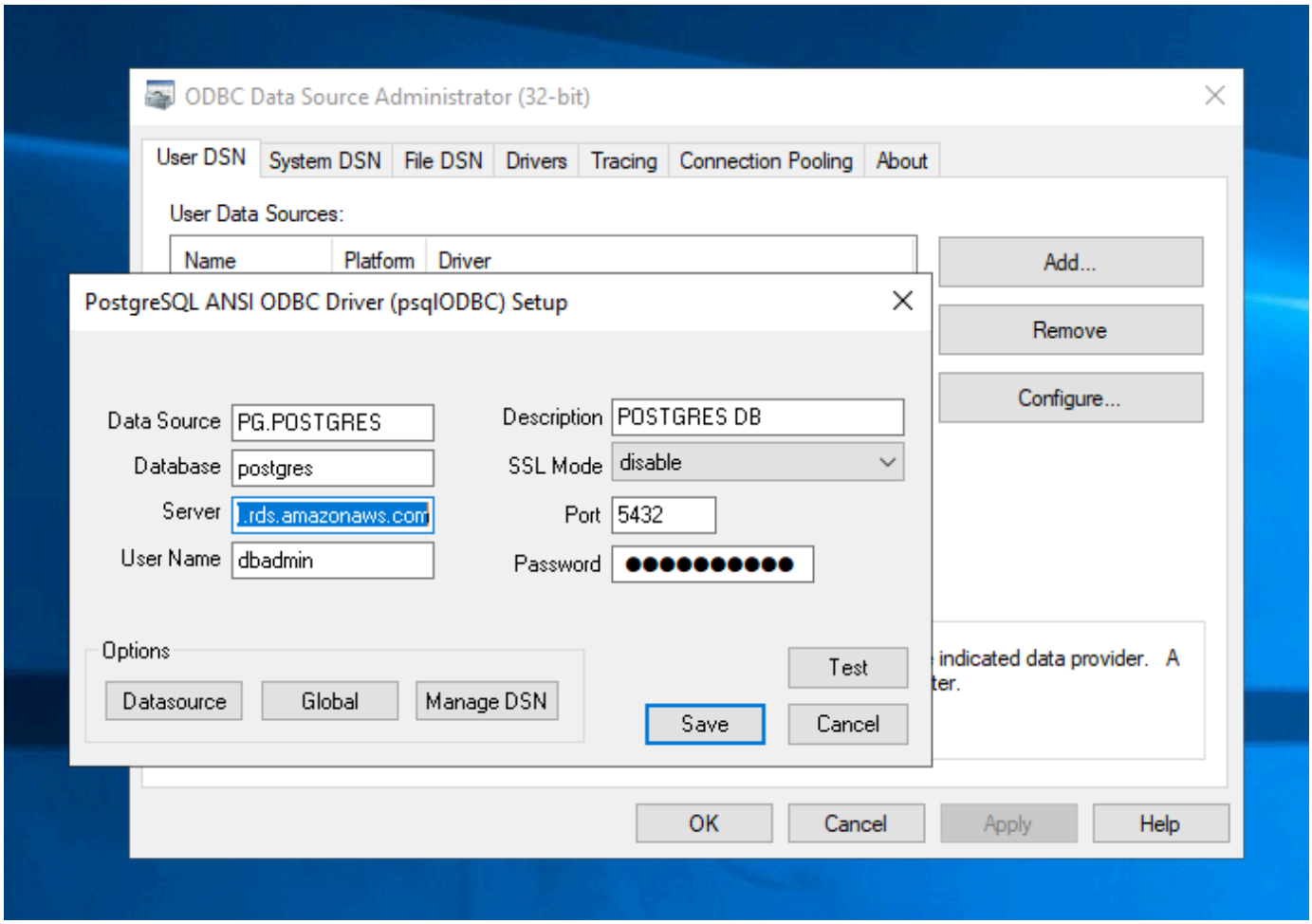
Étape 1 : configurer la connexion ODBC à la banque de données Micro Focus (base de données Amazon RDS)

Au cours de cette étape, vous configurez une connexion ODBC à la base de données qui contient les données que vous souhaitez afficher sous forme de tables et de colonnes. Il s'agit d'une étape unique.

1. Connectez-vous à Micro Focus Enterprise Developer Desktop à l'aide de l'URL de streaming AppStream 2.0.
2. Ouvrez l'administrateur de sources de données ODBC, choisissez User DSN, puis choisissez Ajouter.

3. Dans Create New Data Source, sélectionnez PostgreSQL ANSI, puis Finish.
4. Créez une source de données pour `PG.POSTGRES` en fournissant les informations de base de données nécessaires, comme suit :

```
Data Source : PG.POSTGRES
Database    : postgres
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
Password    : user_password
```



5. Choisissez Test pour vous assurer que la connexion fonctionne. Le message devrait s'afficher en `Connection successful` cas de réussite du test.

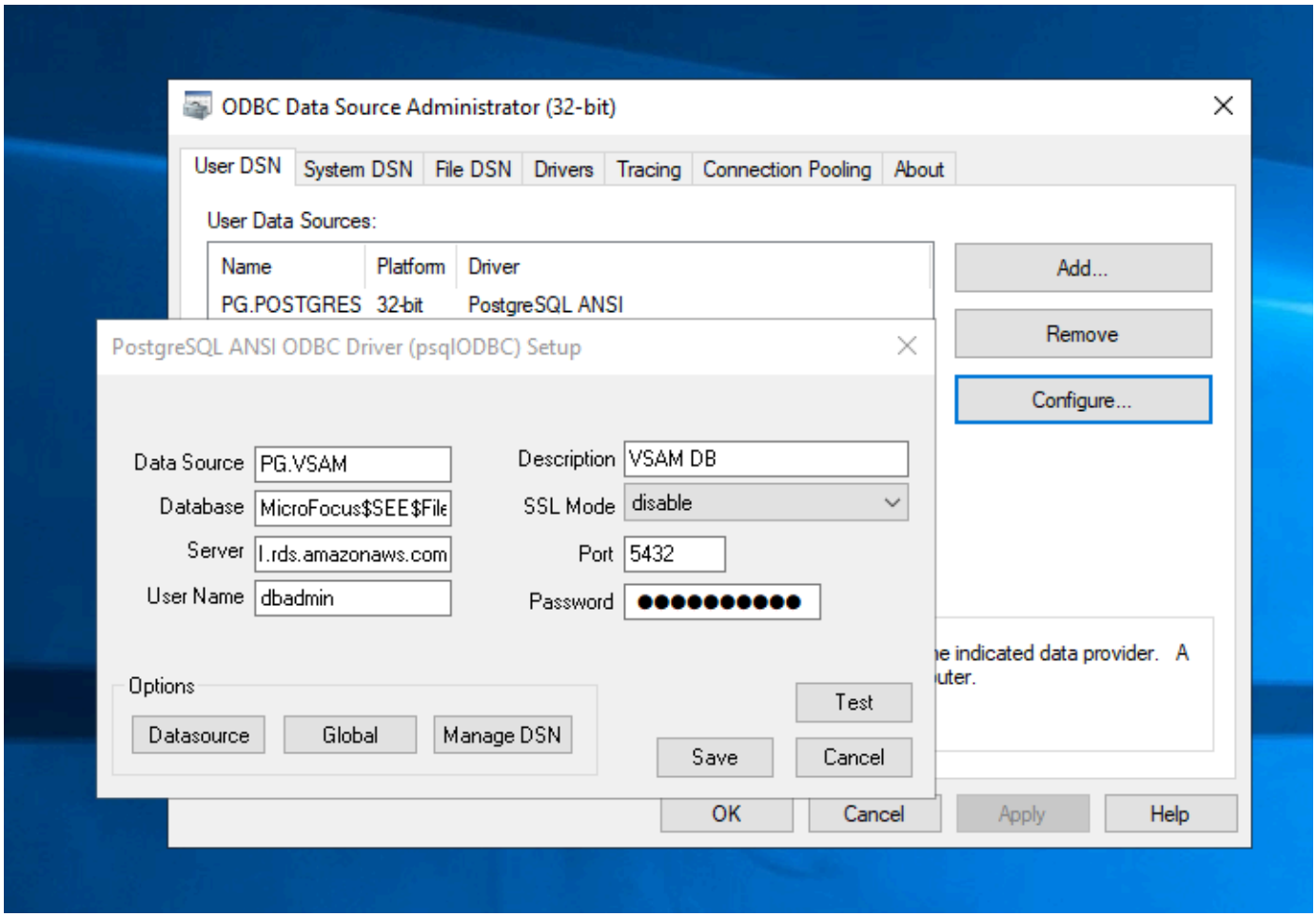
   Si le test échoue, consultez les informations suivantes.

   - Résolution des problèmes pour Amazon RDS

- Comment résoudre les problèmes lors de la connexion à mon instance de base de données Amazon RDS ?

6. Enregistrez la source de données.
7. Créez une source de données pour`PG.VSAM`, testez la connexion et enregistrez-la. Fournissez les informations de base de données suivantes :

```
Data Source : PG.VSAM
Database    : MicroFocus$SEE$Files$VSAM
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
Password    : user_password
```



Étape 2 : Création du fichier MFDBFH.cfg

Au cours de cette étape, vous allez créer un fichier de configuration qui décrit le magasin de données Micro Focus. Il s'agit d'une étape de configuration unique.

1. Dans votre dossier principal, par exemple, dans`D:\PhotonUser\My Files\Home Folder \MFED\cfg\MFDBFH.cfg`, créez le fichier MFDBFH.cfg avec le contenu suivant.

```
<datastores>
       <server name="ESPACDatabase" type="postgresql" access="odbc">
        <dsn name="PG.POSTGRES" type="database" dbname="postgres"/>
        <dsn name="PG.VSAM" type="datastore" dsname="VSAM"/>
       </server>
      </datastores>
```

2. Vérifiez la configuration du MFDBFH en exécutant les commandes suivantes pour interroger la banque de données Micro Focus :

```
*##*
*## Test the connection by running the following commands*
*##*

set MFDBFH_CONFIG="D:\PhotonUser\My Files\Home Folder\MFED\cfg\MFDBFH.cfg"

dbfhdeploy list sql://ESPACDatabase/VSAM?folder=/DATA
```

Étape 3 : Création d'un fichier de structure (STR) pour la mise en page de votre cahier

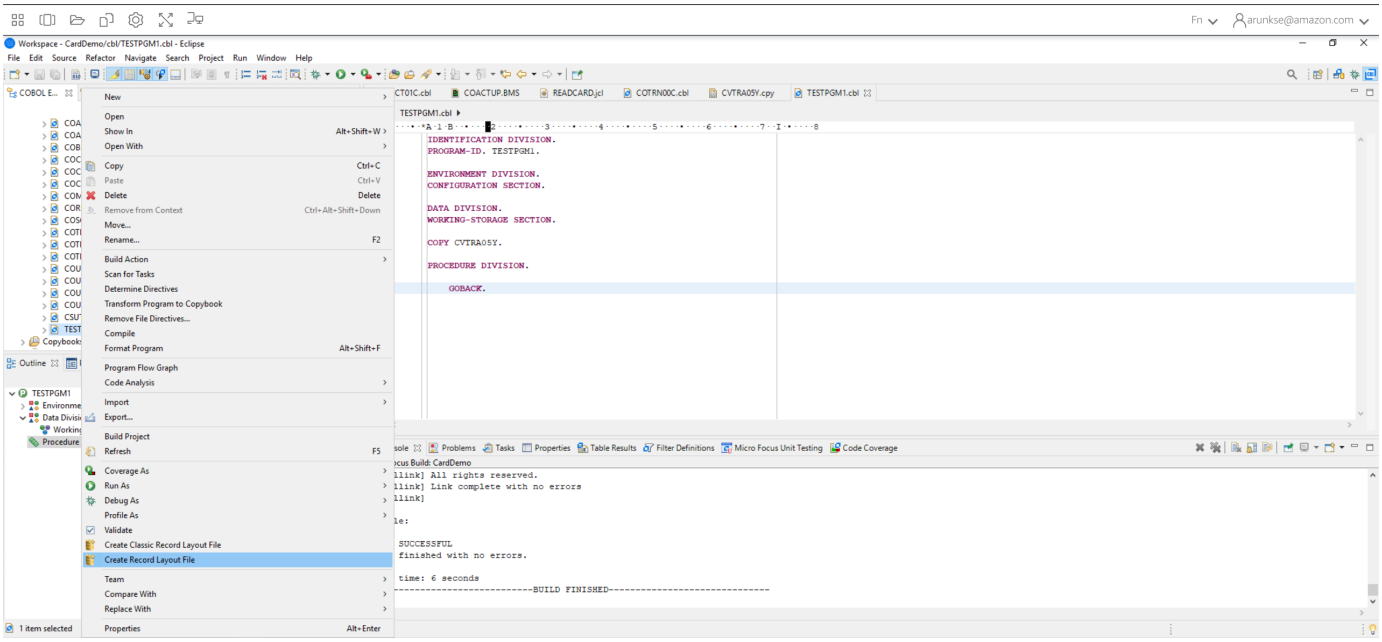
Au cours de cette étape, vous allez créer un fichier de structure pour la mise en page de votre cahier afin de pouvoir l'utiliser ultérieurement pour créer des vues de base de données à partir des ensembles de données.

1. Compilez le programme associé à votre cahier. Si aucun programme n'utilise le cahier, créez et compilez un programme simple comme celui-ci avec une instruction COPY pour votre cahier.

```
IDENTIFICATION DIVISION.
      PROGRAM-ID. TESTPGM1.
```

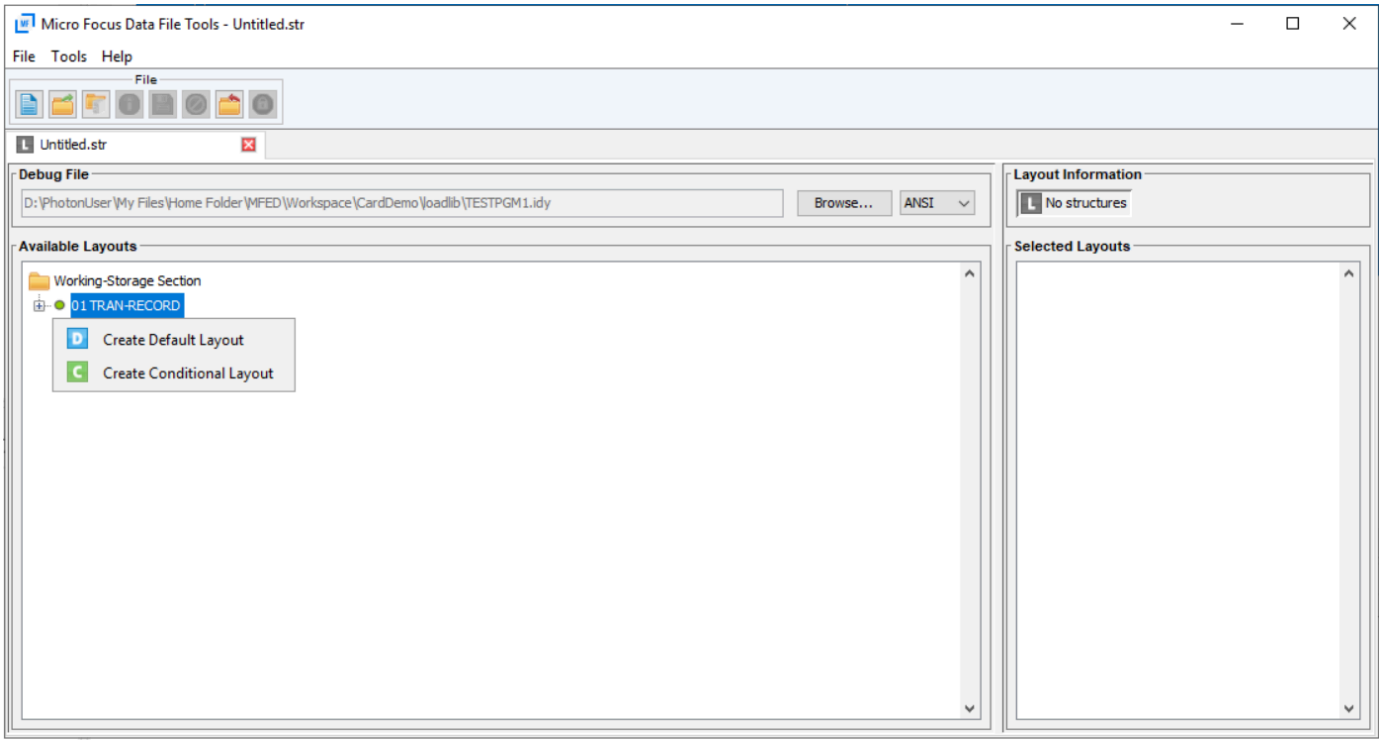
```
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.

       DATA DIVISION.
       WORKING-STORAGE SECTION.

       COPY CVTRA05Y.

       PROCEDURE DIVISION.

       GOBACK.
```

2. Une fois la compilation réussie, cliquez avec le bouton droit sur le programme et choisissez Créer un fichier de mise en page d'enregistrement. Cela ouvrira les outils de fichiers de données Micro Focus à l'aide du fichier .idy généré lors de la compilation.

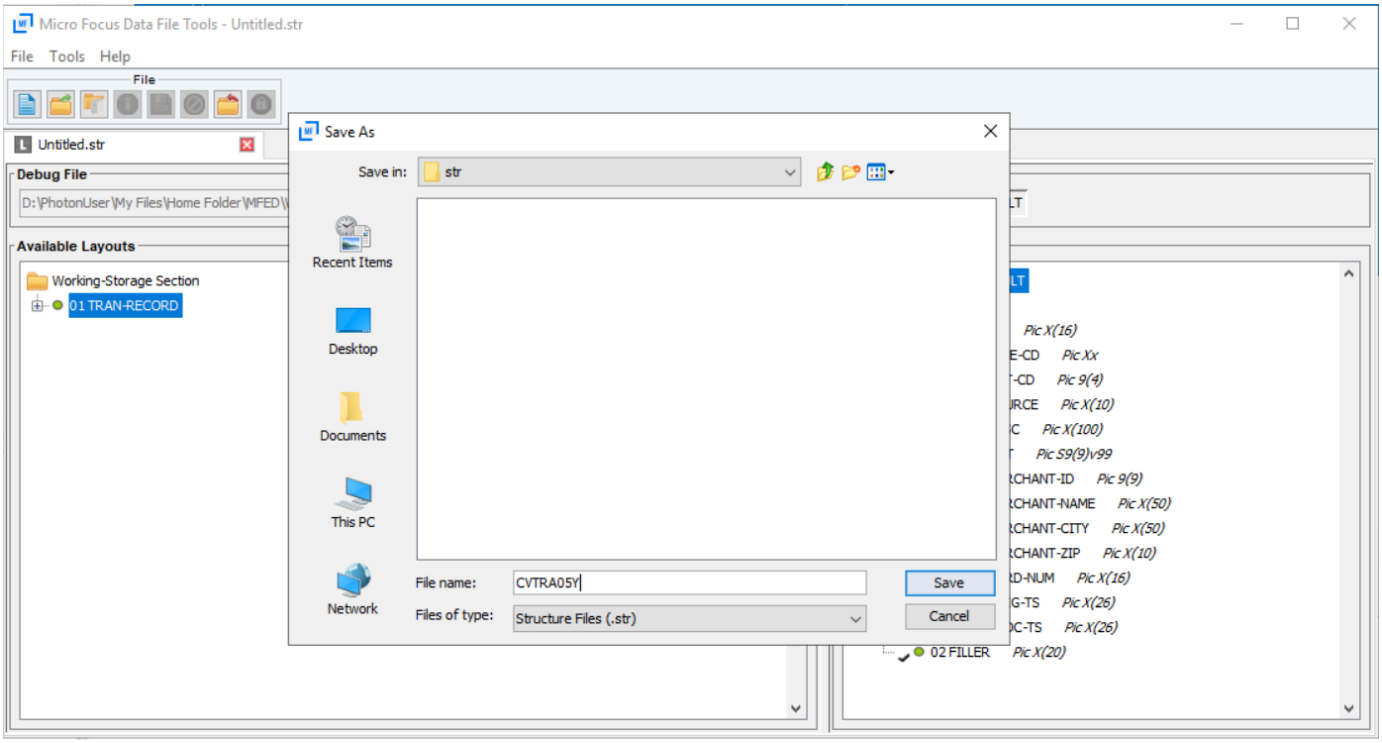


3. Cliquez avec le bouton droit sur la structure d'enregistrement et choisissez Créer une mise en page par défaut (structure unique) ou Créer une mise en page conditionnelle (structure multiple) en fonction de la mise en page.

   Pour plus d'informations, consultez la section Création de fichiers de structure et de mises en page dans la documentation Micro Focus.



4. Après avoir créé la mise en page, choisissez Fichier dans le menu, puis cliquez sur Enregistrer sous. Parcourez et enregistrez le fichier dans votre dossier personnel avec le même nom de fichier que votre bloc-notes. Vous pouvez choisir de créer un dossier appelé `str` et d'y enregistrer tous vos fichiers de structure.



Étape 4 : Création d'une vue de base de données à l'aide du fichier de structure (STR)

Au cours de cette étape, vous utilisez le fichier de structure créé précédemment pour créer une vue de base de données pour un ensemble de données.

- Utilisez la `dbfhview` commande pour créer une vue de base de données pour un ensemble de données qui se trouve déjà dans la banque de données Micro Focus, comme illustré dans l'exemple suivant.

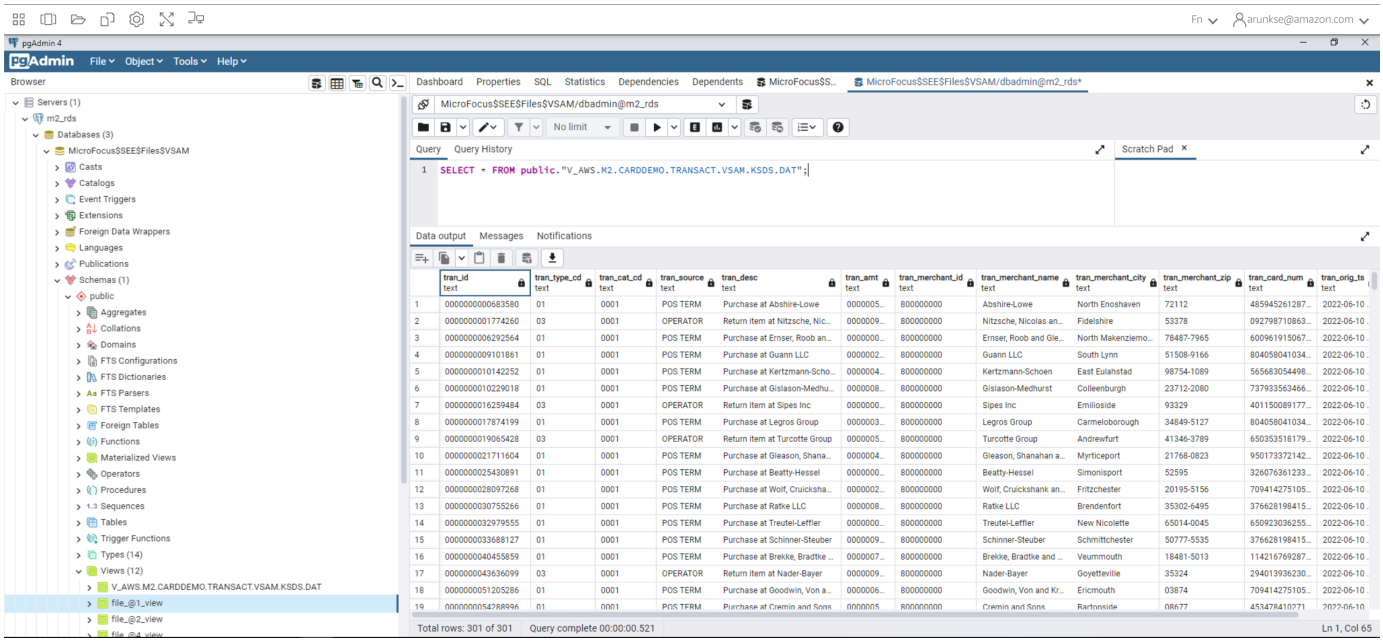
```
##
      ## The below command creates database view for VSAM file
 AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS
      ## using the STR file CVTRA05Y.str
      ##

      dbfhview -create -struct:"D:\PhotonUser\My Files\Home Folder\MFED\str
\CVTRA05Y.str" -name:V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT -file:sql://
ESPACDatabase/VSAM/AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT?folder=/DATA

      ##
      ## Output:
      ##

      Micro Focus Database File Handler - View Generation Tool Version 8.0.00
      Copyright (C) 1984-2022 Micro Focus. All rights reserved.

      VGN0017I Using structure definition 'TRAN-RECORD-DEFAULT'
      VGN0022I View 'V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT' installed in
 datastore 'sql://espacdatabase/VSAM'
      VGN0002I The operation completed successfully
```

Étape 5 : Afficher les ensembles de données Micro Focus sous forme de tableaux et de colonnes

Au cours de cette étape, connectez-vous à la base de données `pgAdmin` afin de pouvoir exécuter des requêtes pour afficher les ensembles de données tels que les tables et les colonnes.

- Connectez-vous à la base de données à `MicroFocus$SEE$Files$VSAM` l'aide de pgAdmin et interrogez la vue de base de données que vous avez créée à l'étape 4.

```
SELECT * FROM public."V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT";
```



Didacticiel : Utiliser des modèles avec Micro Focus Enterprise Developer

Ce didacticiel décrit comment utiliser des modèles et des projets prédéfinis avec Micro Focus Enterprise Developer. Il couvre les trois cas d'utilisation. Tous les cas d'utilisation utilisent l'exemple de code fourni dans le BankDemo Exemple Pour télécharger l'exemple, choisissez l'exemple`bankdemo.zip`.

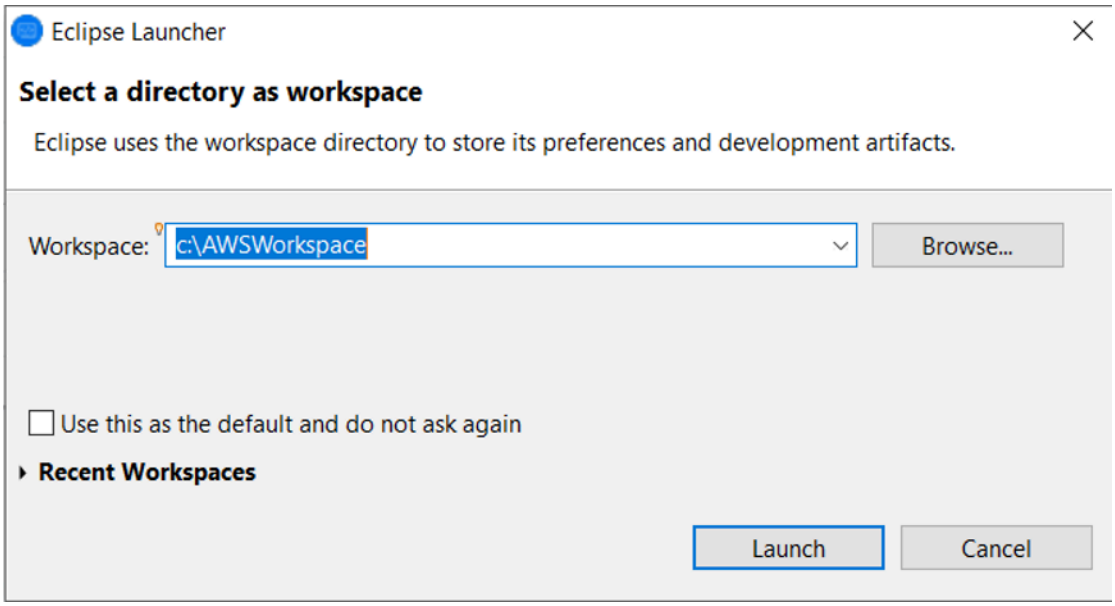
> **Important**
>
> Si vous utilisez la version d'Enterprise Developer pour Windows, les binaires générés par le compilateur ne peuvent s'exécuter que sur Enterprise Server fourni avec Enterprise Developer. Vous ne pouvez pas les exécuter sous la rubriqueAWSRuntime Mainframe Modernization, basé sur Linux.

Rubriques

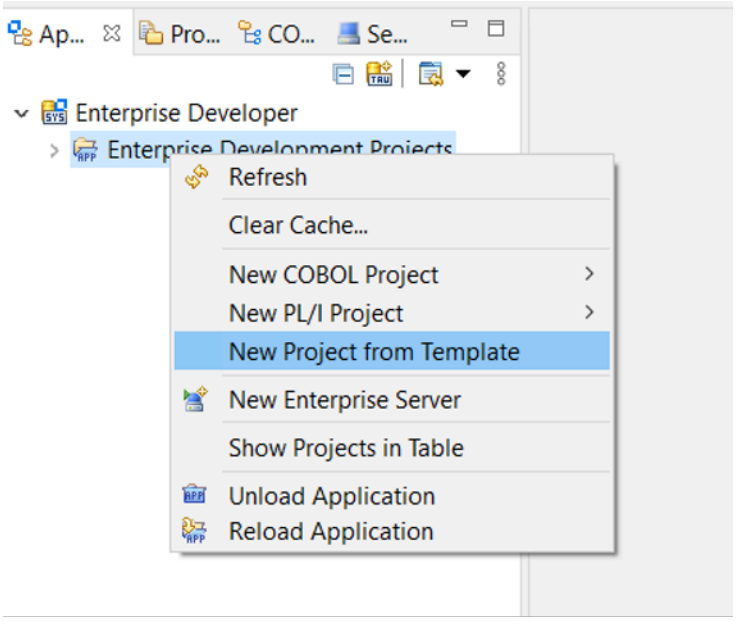
Cas d'utilisation 1 : utilisation du modèle de projet COBOL contenant les composants source

Ce cas d'utilisation nécessite que vous copiez les composants sources dans la structure de répertoires Template dans le cadre des étapes de pré-configuration de la démo. Dans`bankdemo.zip`cela a été modifié par rapport à l'original`AWSTemplates.zip`livraison pour éviter d'avoir deux copies de la source.

1. Démarrez Enterprise Developer et définissez l'espace de travail choisi.

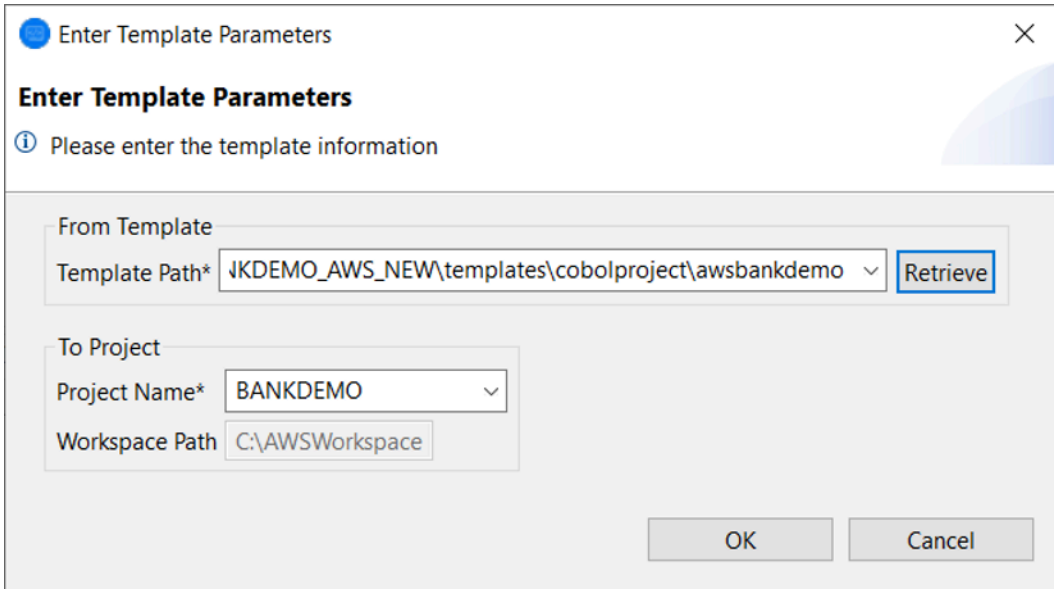


2. Au sein duExplorer d'applicationsvue, depuis leProjet de développement d'entrepriseélément de l'arborescence, choisissezNouveau projet depuis le modèledepuis le menu contextuel.

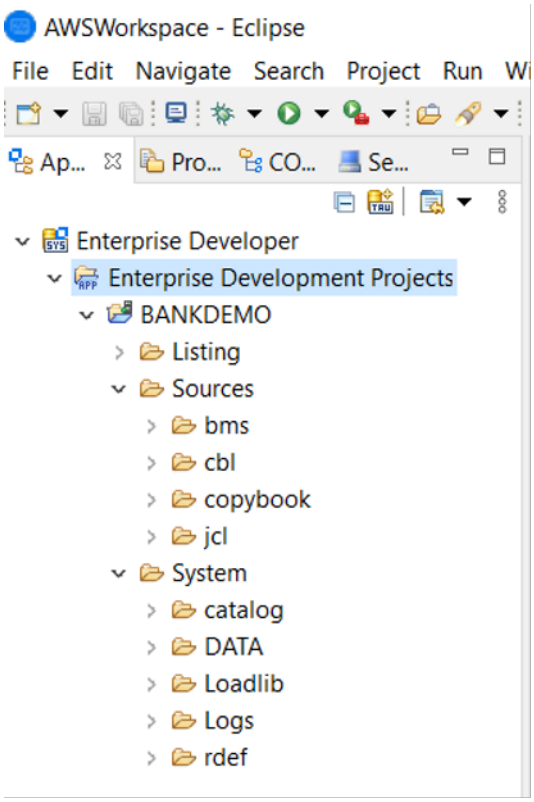


3. Entrez les paramètres du modèle comme indiqué.

> **Note**
>
> Le chemin du modèle fera référence à l'endroit où le ZIP a été extrait.



4. Choisir OK créera un projet Eclipse de développement local basé sur le modèle fourni, avec une structure complète d'environnement source et d'exécution.



Le`System`contient un fichier de définition de ressource complet avec les entrées requises pour BANKDEMO, le catalogue requis avec les entrées ajoutées et les fichiers de données ASCII correspondants.

Comme la structure du modèle source contient tous les éléments sources, ces fichiers sont copiés dans le projet local et sont donc automatiquement créés dans Enterprise Developer.

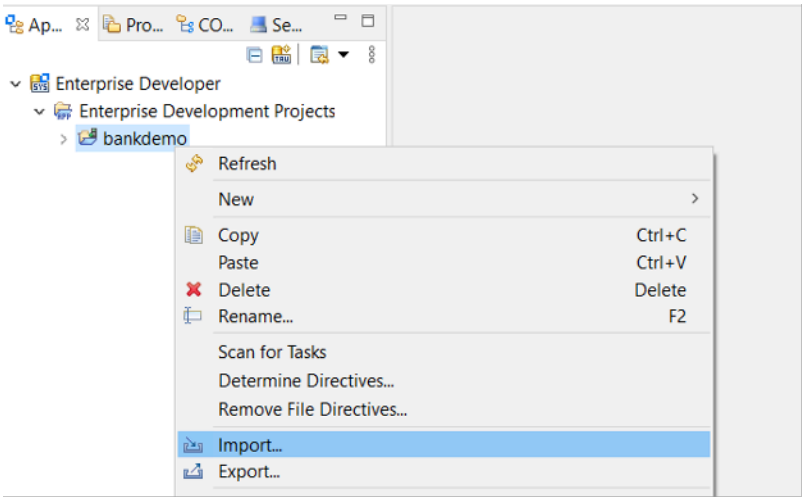
Cas d'utilisation 2 : utilisation du modèle de projet COBOL sans composants source

Les étapes 1 à 3 sont identiques àCas d'utilisation 1 : utilisation du modèle de projet COBOL contenant les composants source.

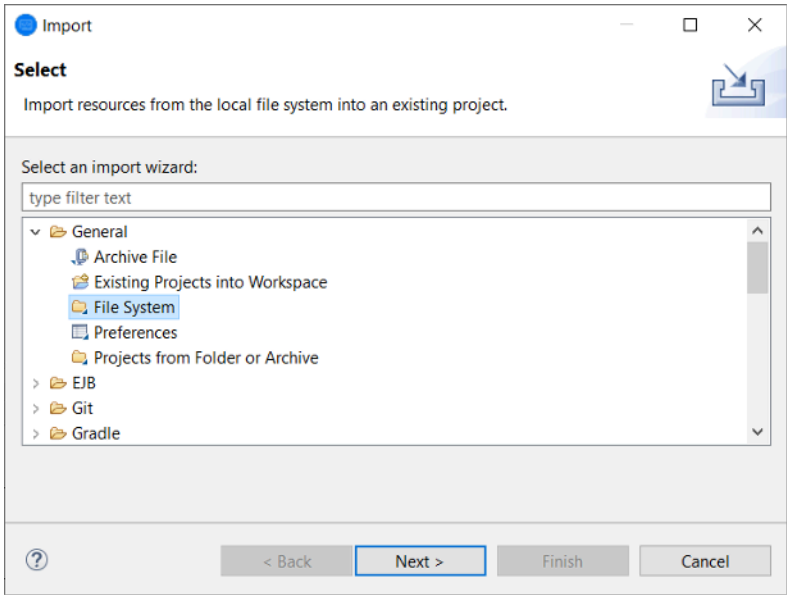
Le`System`dans cet exemple d'utilisation contient également un fichier de définition de ressource complet avec les entrées requises pour BankDemo, le catalogue requis avec des entrées ajoutées et les fichiers de données ASCII correspondants.

Toutefois, la structure source du modèle ne contient aucun composant. Vous devez les importer dans le projet à partir du référentiel source que vous utilisez.

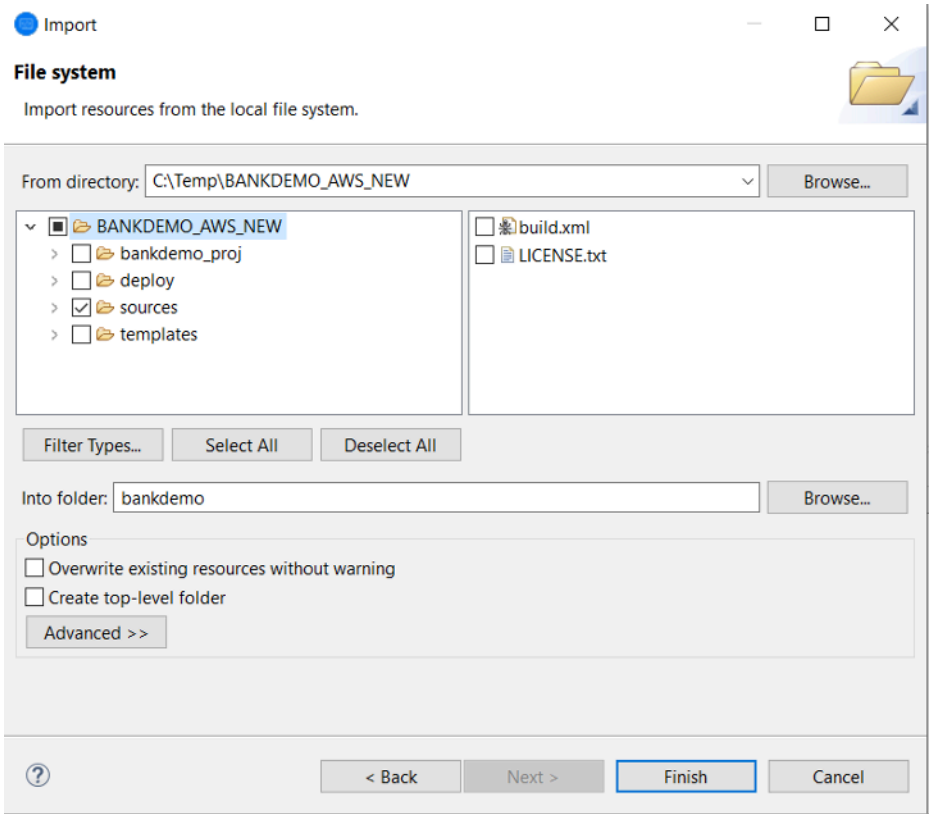
1. Choisissez le nom du projet. Dans le menu contextuel associé, choisissez l'optionImporter.



2. Dans la boîte de dialogue qui s'affiche, sous leGénéralSection, choisissez choisissezSystème de fichiers, puis choisissez Suivant.



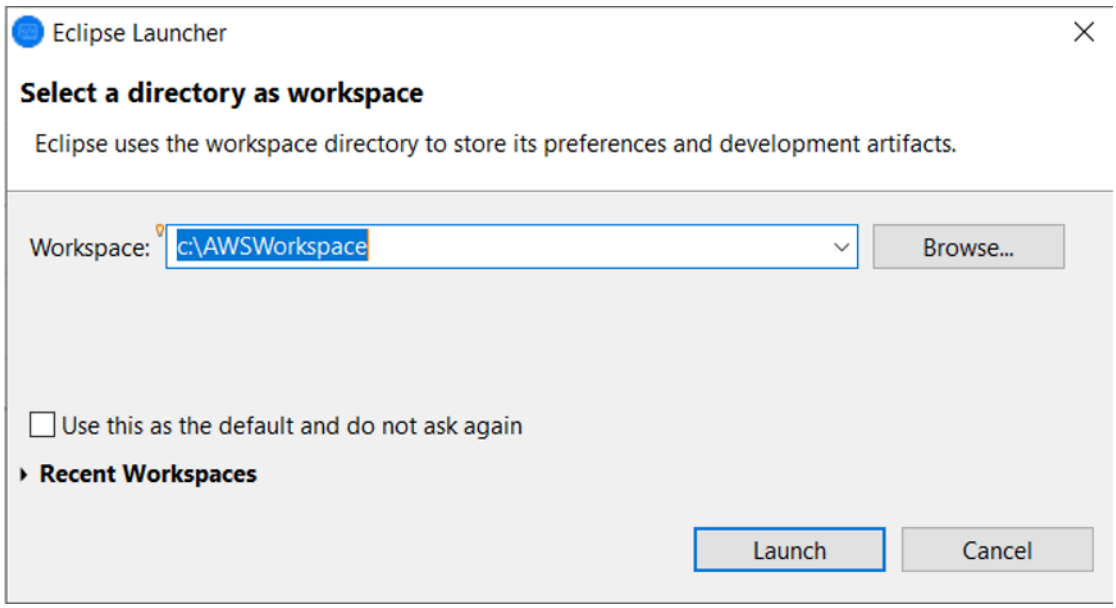
3. Remplissage duÀ partir du répertoireen parcourant le système de fichiers pour pointer vers le dossier du référentiel. Choisissez tous les dossiers que vous souhaitez importer, tels que`sources`. Le`Into folder`sera prérempli. Choisissez Finish (Terminer).



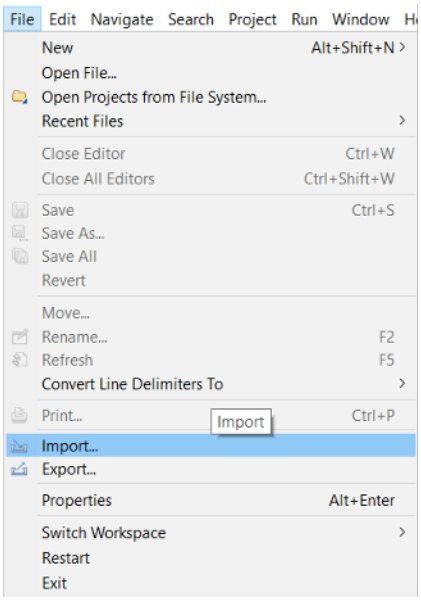
Une fois que la structure du modèle source contient tous les éléments sources, ceux-ci sont créés automatiquement dans Enterprise Developer.

Cas d'utilisation 3 - Utilisation du projet COBOL prédéfini lié aux dossiers sources

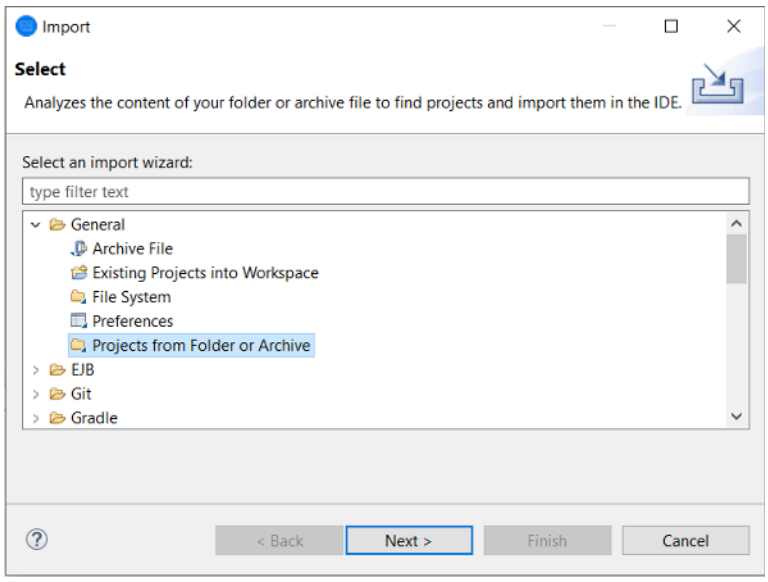
1. Démarrez Enterprise Developer et définissez l'espace de travail choisi.



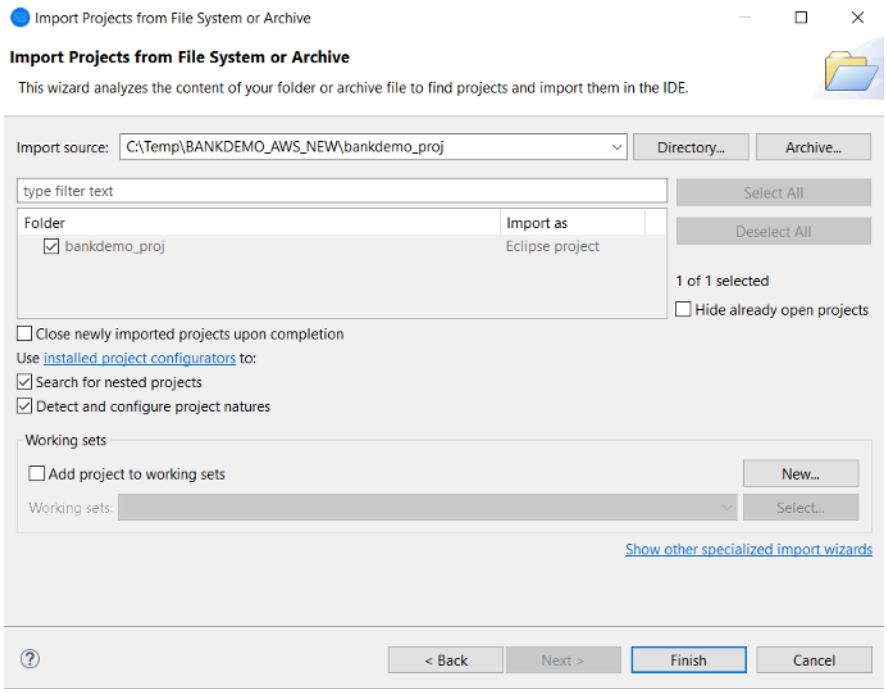
2. Dans le menu File (Fichier), choisissez Import (Importer).



3. Dans la boîte de dialogue qui s'affiche, sousGénéral, choisissezProjets d'un dossier ou d'une archiveet choisissezSuivant.



4. RemplissageSource d'importationChoisissez ChoisirDirectoryet parcourez le système de fichiers pour sélectionner le dossier de projet prédéfini. Le projet qu'il contient contient des liens vers les dossiers sources du même référentiel.

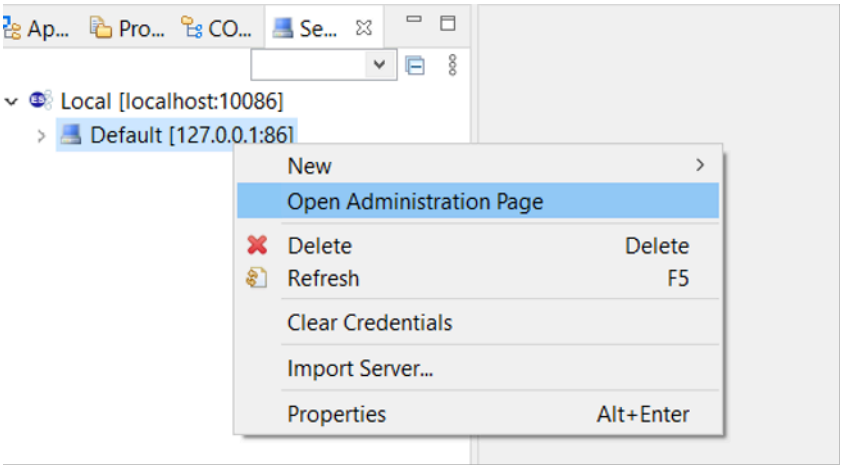


Choisissez Finish (Terminer).

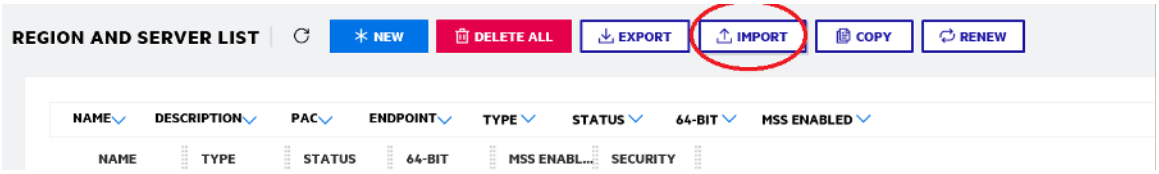
Comme le projet est renseigné par les liens vers le dossier source, le code est automatiquement créé.

Utilisation du modèle JSON de définition de région

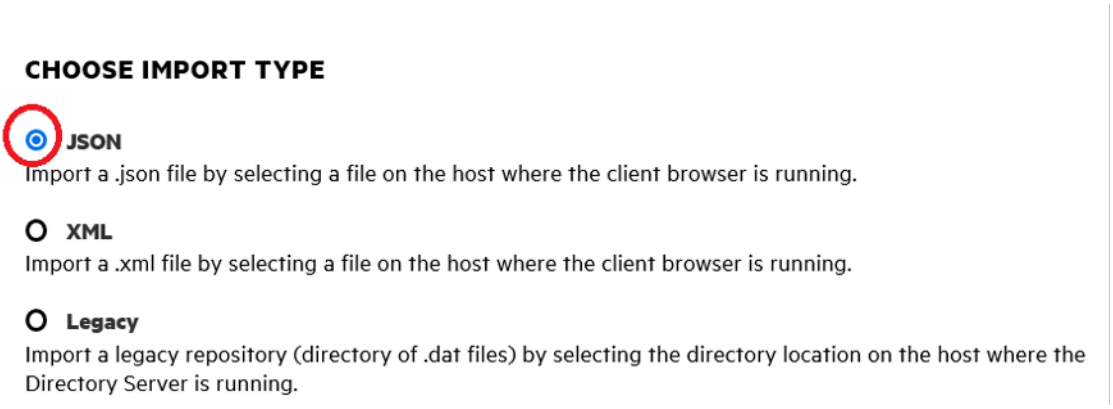
1. Basculer vers la vue Explorer de serveurs. Dans le menu contextuel associé, choisissez l'optionOuvrir la page Administration, qui démarre le navigateur par défaut.



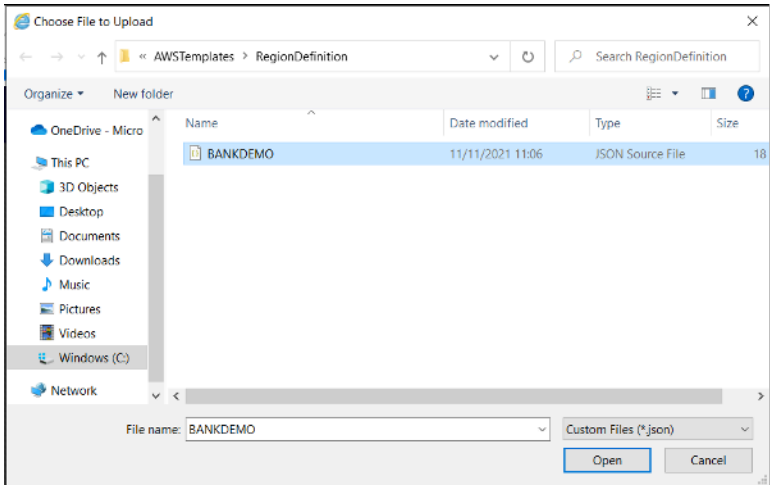
2. Dans l'écran Enterprise Server Common Web Administration (ESCWA) qui s'affiche, choisissezImporter.



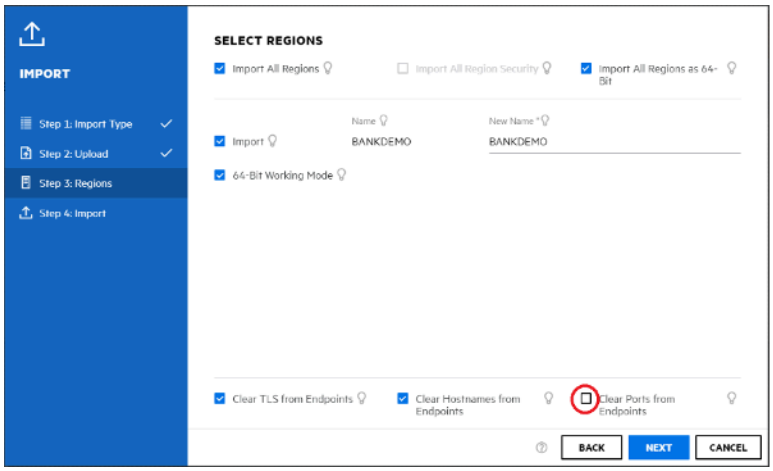
3. Cliquez sur l'ongletJASONImportez le type et choisissezSuivant.



4. Upload le fourniBANKDEMO.JSONdans le fichier.

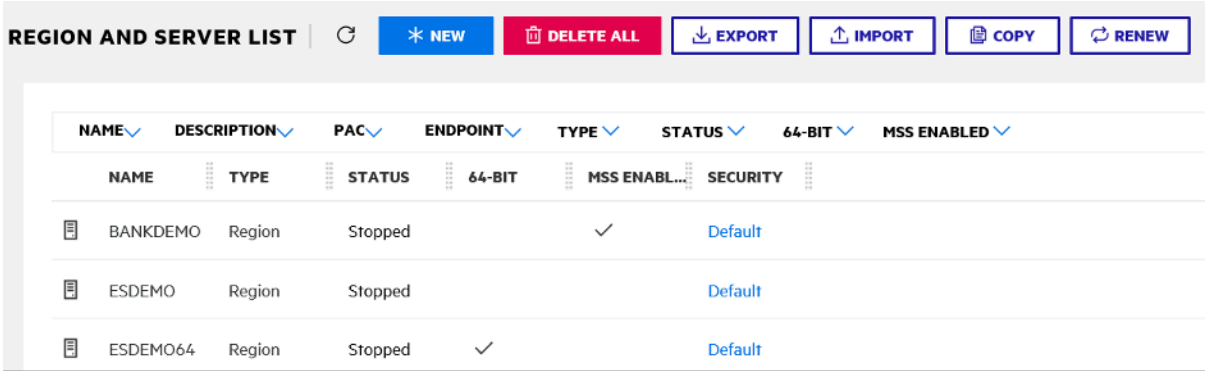


Une fois sélectionné, choisissezSuivant.

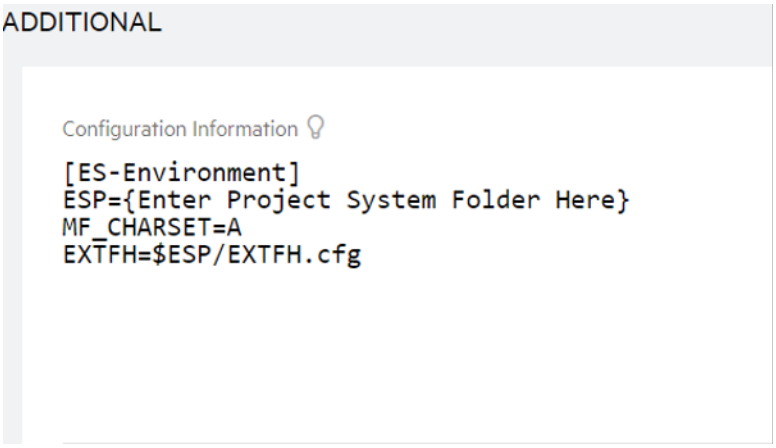


Dans la pageSélectionner des régions, assurez-vous que le panneauEffacer les ports des points de terminaisonn'est pas sélectionnée, puis continuez à choisir ChoisirSuivantà travers les panneaux jusqu'auEffectuer l'importationle panneau est affiché. Choisissez ensuite Import (Importer).

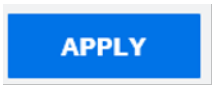
Enfin, cliquez surTerminer. La région BANKDEMO sera ensuite ajoutée à la liste des serveurs.



5. Accédez à .Propriétés généralespour la région BANKDEMO.
6. Scroll to theConfigurationSection.
7. La variable d'environnement ESP doit être définie sur la valeur`System`dossier correspondant au projet Eclipse créé lors des étapes précédentes. doit être`workspacefolder/projectname/System`.

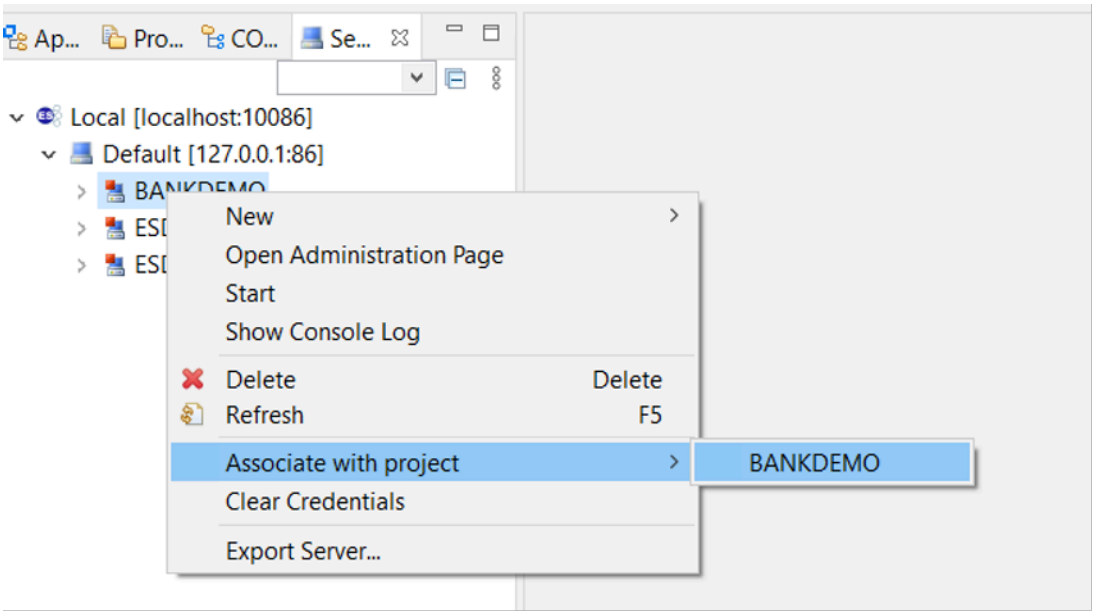


8. Cliquez sur Apply.



La région est désormais entièrement configurée pour fonctionner en conjonction avec le projet Eclipse COBOL.

9. Enfin, de retour dans Enterprise Developer, associez la région importée au projet.



L'environnement Enterprise Developer est maintenant prêt à être utilisé, avec une version fonctionnelle complète de BankDemo. Vous pouvez modifier, compiler et déboguer du code par rapport à la région.

> **Important**
>
> Si vous utilisez la version d'Enterprise Developer pour Windows, les binaires générés par le compilateur ne peuvent s'exécuter que sur Enterprise Server fourni avec Enterprise Developer. Vous ne pouvez pas les exécuter sous le moteur d'exécution Mainframe Modernization, qui est basé sur Linux.

Tutoriel : Configuration de la version Micro Focus pour l' BankDemo exemple d'application

AWSLa modernisation du mainframe vous permet de configurer des builds et des pipelines d'intégration continue/de livraison continue (CI/CD) pour vos applications migrées. Ces builds et pipelines utilisent AWS CodeBuildAWS CodeCommit, et AWS CodePipeline pour fournir ces fonctionnalités. CodeBuild est un service de génération entièrement géré qui compile votre code source, exécute des tests unitaires et produit des artefacts prêts à être déployés. CodeCommit est un service de contrôle de version qui vous permet de stocker et de gérer de manière privée des

référentiels Git dans le cloud. AWS CodePipeline est un service de livraison continue qui vous permet de modéliser, de visualiser et d'automatiser les étapes nécessaires à la publication de votre logiciel.

Ce didacticiel explique AWS CodeBuild comment compiler l' BankDemo exemple de code source de l'application à partir d'Amazon S3, puis exporter le code compilé vers Amazon S3.

AWS CodeBuildest un service d'intégration continue entièrement géré qui compile le code source, exécute des tests et produit des progiciels prêts à être déployés. Vous pouvez utiliser des CodeBuild environnements de génération préemballés ou créer des environnements de génération personnalisés utilisant vos propres outils de génération. Ce scénario de démonstration utilise la deuxième option. Il s'agit d'un environnement de CodeBuild construction qui utilise une image Docker préemballée.

> **Important**
>
> Avant de démarrer votre projet de modernisation du mainframe, nous vous recommandons de vous renseigner sur le AWSMigration Acceleration Program (MAP) pour mainframe ou de contacter des spécialistes du AWS mainframe pour connaître les étapes nécessaires à la modernisation d'une application mainframe.

Rubriques

Prérequis

Avant de commencer ce didacticiel, remplissez les conditions préalables suivantes.

- Téléchargez l'BankDemo exemple d'application et décompressez-le dans un dossier. Le dossier source contient les programmes COBOL et les copybooks, ainsi que les définitions CICS BMS. Il contient également un dossier JCL à titre de référence, bien que vous n'ayez pas besoin de compiler JCL. Le dossier contient également les méta-fichiers nécessaires à la compilation.
- Dans la console AWS Mainframe Modernization, sélectionnez Tools. Dans Analyse, développement et création d'actifs, choisissez Partager des actifs avec mon compte AWS.

Étape 1 : créer des compartiments Amazon S3

Au cours de cette étape, vous allez créer deux compartiments Amazon S3. Le premier est un compartiment d'entrée pour contenir le code source, et l'autre est un compartiment de sortie pour contenir la sortie de compilation. Pour plus d'informations, consultez la section Création, configuration et utilisation des compartiments Amazon S3 dans le guide de l'utilisateur Amazon S3.

1. Pour créer le compartiment d'entrée, connectez-vous à la console Amazon S3 et choisissez Create bucket.
2. Dans Configuration générale, donnez un nom au compartiment et spécifiez l'Région AWSendroit où vous souhaitez le créer. Par exemple`codebuild-regionId-accountId-input-bucket`, le nom est « où se `regionId` trouve le Région AWS compartiment » et « où `accountId` est votre Compte AWS identifiant ».

   > **Note**
   >
   > Si vous créez le bucket dans un pays différent Région AWS de celui de l'est des États-Unis (Virginie du Nord), spécifiez le `LocationConstraint` paramètre. Pour plus d'informations, consultez Create Bucket dans le manuel Amazon Simple Storage Service API Reference.

3. Conservez tous les autres paramètres et choisissez Create bucket.
4. Répétez les étapes 1 à 3 pour créer le compartiment de sortie. Par exemple`codebuild-regionId-accountId-output-bucket`, le nom est « où se `regionId` trouve le Région AWS compartiment » et « où `accountId` est votre Compte AWS identifiant ».

Quels que soient les noms que vous choisissez pour ces compartiments, veillez à les utiliser tout au long de ce didacticiel.

Étape 2 : Création du fichier de spécifications de construction

Au cours de cette étape, vous créez un fichier de spécifications de construction,. Ce fichier fournit les commandes de construction et les paramètres associés, au format YAML, CodeBuild pour exécuter la génération. Pour plus d'informations, reportez-vous à la section Référence des spécifications de construction CodeBuild dans le Guide de AWS CodeBuild l'utilisateur.

1. Créez un fichier nommé `buildspec.yml` dans le répertoire que vous avez décompressé comme condition préalable.
2. Ajoutez le contenu suivant au fichier et enregistrez-le. Aucune modification n'est requise pour ce fichier.

```
version: 0.2
env:
  exported-variables:
    - CODEBUILD_BUILD_ID
    - CODEBUILD_BUILD_ARN
phases:
  install:
    runtime-versions:
      python: 3.7
  pre_build:
    commands:
      - echo Installing source dependencies...
      - ls -lR $CODEBUILD_SRC_DIR/source
  build:
    commands:
      - echo Build started on `date`
      - /start-build.sh -Dbasedir=$CODEBUILD_SRC_DIR/source -Dloaddir=
$CODEBUILD_SRC_DIR/target
  post_build:
    commands:
      - ls -lR $CODEBUILD_SRC_DIR/target
      - echo Build completed on `date`
artifacts:
  files:
```

```
    - $CODEBUILD_SRC_DIR/target/**
```

Voici`CODEBUILD_BUILD_ID`, `CODEBUILD_BUILD_ARN``$CODEBUILD_SRC_DIR/source`, et `$CODEBUILD_SRC_DIR/target` les variables d'environnement sont-elles disponibles dans CodeBuild. Pour plus d'informations, consultez la section Variables d'environnement dans les environnements de génération.

À ce stade, votre répertoire devrait ressembler à ceci.

```
(root directory name)
    |-- build.xml
    |-- buildspec.yml
    |-- LICENSE.txt
    |-- source
         |... etc.
```

3. Compressez le contenu du dossier dans un fichier nomméBankDemo.zip.. Pour ce didacticiel, vous ne pouvez pas compresser le dossier. Compressez plutôt le contenu du dossier dans le fichierBankDemo.zip.

Étape 3 : télécharger les fichiers sources

Au cours de cette étape, vous chargez le code source de l' BankDemo exemple d'application dans votre compartiment d'entrée Amazon S3.

1. Connectez-vous à la console Amazon S3 et choisissez Buckets dans le volet de navigation de gauche. Choisissez ensuite le compartiment d'entrée que vous avez créé précédemment.
2. Sous Objets, choisissez Charger.
3. Dans la section Fichiers et dossiers, choisissez Ajouter des fichiers.
4. Accédez à votre BankDemo.zip fichier et choisissez-le.
5. Sélectionnez Charger.

Étape 4 : créer des politiques IAM

Au cours de cette étape, vous allez créer deux politiques IAM. Une politique autorise AWS Mainframe Modernization à accéder à l'image Docker qui contient les outils de génération Micro Focus et à l'utiliser. Cette politique n'est pas personnalisée pour les clients. L'autre politique autorise AWS

Mainframe Modernization à interagir avec les compartiments d'entrée et de sortie, ainsi qu'avec les CloudWatch journaux Amazon qui CodeBuild en sont générés.

Pour en savoir plus sur la création d'une stratégie IAM, consultez la section Modification des politiques IAM dans le Guide de l'utilisateur IAM.

Pour créer une politique d'accès aux images Docker

1. Dans la console IAM, copiez le document de stratégie suivant et collez-le dans l'éditeur de stratégie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::aws-m2-repo-*/*"
        }
    ]
}
```

2. Donnez un nom à la politique, par exemple,`m2CodeBuildPolicy`.

Pour créer une politique permettant à la modernisation du AWS mainframe d'interagir avec les buckets et les journaux

1. Dans la console IAM, copiez le document de stratégie suivant et collez-le dans l'éditeur de stratégie. Assurez-vous de mettre `regionId` à jour leRégion AWS, et `accountId` votreCompte AWS.

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Action": [
                  "logs:CreateLogGroup",
                  "logs:CreateLogStream",
                  "logs:PutLogEvents"
              ],
              "Resource": [
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project",
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project:*"
              ],
              "Effect": "Allow"
          },
          {
              "Action": [
                  "s3:PutObject",
                  "s3:GetObject",
                  "s3:GetObjectVersion",
                  "s3:GetBucketAcl",
                  "s3:GetBucketLocation",
                  "s3:List*"
              ],
              "Resource": [
                  "arn:aws:s3:::codebuild-regionId-accountId-input-bucket",
                  "arn:aws:s3:::codebuild-regionId-accountId-input-bucket/*",
                  "arn:aws:s3:::codebuild-regionId-accountId-output-bucket",
                  "arn:aws:s3:::codebuild-regionId-accountId-output-bucket/*"
              ],
              "Effect": "Allow"
          }
      ]
  }
```

2. Donnez un nom à la politique, par exemple,`BankdemoCodeBuildRolePolicy`.

Étape 5 : Création d'un rôle IAM

Au cours de cette étape, vous créez un nouveau rôle IAM qui permet d'interagir avec les AWS ressources CodeBuild à votre place, après avoir associé les politiques IAM que vous avez créées précédemment à ce nouveau rôle IAM.

Pour plus d'informations sur la création d'un rôle de service, consultez la section Création d'un rôle pour déléguer des autorisations à un AWS service dans le guide de l'utilisateur IAM,.

1. Connectez-vous à la console IAM et choisissez Rôles dans le volet de navigation de gauche.
2. Sélectionnez Créer un rôle.
3. Sous Type d'entité fiable, choisissez le service AWS.
4. Sous Cas d'utilisation pour d'autres services AWS CodeBuild, choisissez, puis choisissez CodeBuildà nouveau.
5. Choisissez Suivant.
6. Sur la page Add permissions (Ajouter des autorisations), sélectionnez Next (Suivant). Vous attribuez une politique au rôle ultérieurement.
7. Sous Détails du rôle, saisissez un nom pour le rôle, par exemple,`BankdemoCodeBuildServiceRole`.
8. Sous Sélectionner les entités de confiance, vérifiez que le document de politique ressemble à ce qui suit :

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        }
```

9. Sélectionnez Créer un rôle.

Étape 6 : associer les politiques IAM au rôle IAM

Au cours de cette étape, vous associez les deux politiques IAM que vous avez créées précédemment au rôle `BankdemoCodeBuildServiceRole` IAM.

1. Connectez-vous à la console IAM et choisissez Rôles dans le volet de navigation de gauche.
2. Dans Rôles, choisissez le rôle que vous avez créé précédemment, par exemple`BankdemoCodeBuildServiceRole`.
3. Dans Politiques d'autorisations, choisissez Ajouter des autorisations, puis Joindre des politiques.
4. Dans Autres politiques d'autorisation, choisissez les politiques que vous avez créées précédemment, par exemple, `m2CodeBuildPolicy` et`BankdemoCodeBuildRolePolicy`.
5. Sélectionnez Attach Policies (Attacher des politiques).

Étape 7 : Création du CodeBuild projet

Au cours de cette étape, vous créez le CodeBuild projet.

1. Connectez-vous à la CodeBuild console et choisissez Create build project.
2. Dans la section Configuration du projet, saisissez un nom pour le projet, par exemple,`codebuild-bankdemo-project`.
3. Dans la section Source, pour Source provider, choisissez Amazon S3, puis choisissez le bucket d'entrée que vous avez créé précédemment, par exemple,`codebuild-regionId-accountId-input-bucket`.
4. Dans le champ Clé d'objet S3 ou dossier S3, entrez le nom du fichier zip que vous avez chargé dans le compartiment S3. Dans ce cas, le nom du fichier est`bankdemo.zip`.
5. Dans la section Environnement, choisissez Image personnalisée.
6. Dans le champ Type d'environnement, sélectionnez Linux.
7. Sous Registre d'images, choisissez Autre registre.
8. Dans le champ URL du registre externe, entrez `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:latest`

9. Sous Rôle de service, choisissez Rôle de service existant, puis dans le champ ARN du rôle, choisissez le rôle de service que vous avez créé précédemment, par exemple,`BankdemoCodeBuildServiceRole`.
10. Dans la section Buildspec, choisissez Utiliser un fichier buildspec.
11. Dans la section Artifacts, sous Type, choisissez Amazon S3, puis choisissez votre compartiment de sortie, par exemple,`codebuild-regionId-accountId-output-bucket`.
12. Dans le champ Nom, entrez le nom d'un dossier dans le compartiment dans lequel vous souhaitez contenir les artefacts de sortie de génération, par exemple`bankdemo-output.zip`.
13. Sous Emballage des artefacts, sélectionnez Zip.
14. Choisissez Créer un projet de génération.

Étape 8 : démarrer la construction

Au cours de cette étape, vous lancez la construction.

1. Connectez-vous à la CodeBuild console.
2. Dans le volet de navigation de gauche, choisissez Créer des projets.
3. Choisissez le projet de construction que vous avez créé précédemment, par exemple`codebuild-bankdemo-project`.
4. Choisissez Démarrer la génération.

Cette commande lance le build. La compilation s'exécute de manière asynchrone. La sortie de la commande est un fichier JSON qui inclut l'identifiant de l'attribut. Cet attribut id est une référence à l'identifiant de CodeBuild construction de la construction que vous venez de démarrer. Vous pouvez consulter l'état du build dans la CodeBuild console. Vous pouvez également consulter les journaux détaillés de l'exécution de la compilation dans la console. Pour plus d'informations, voir Afficher les informations de construction détaillées dans le guide de AWS CodeBuild l'utilisateur.

Lorsque la phase en cours est TERMINÉE, cela signifie que votre compilation s'est terminée avec succès et que vos artefacts compilés sont prêts sur Amazon S3.

Étape 9 : Télécharger les artefacts de sortie

Au cours de cette étape, vous devez télécharger les artefacts de sortie depuis Amazon S3. L'outil de génération Micro Focus peut créer différents types d'exécutables. Dans ce didacticiel, il génère des objets partagés.

1. Connectez-vous à la console Amazon S3.
2. Dans la section Buckets, choisissez le nom de votre bucket de sortie, par exemple,`codebuild-regionId-accountId-output-bucket`.
3. Choisissez Téléchargement.
4. Décompressez le fichier téléchargé. Accédez au dossier cible pour voir les artefacts de construction. Il s'agit notamment des objets partagés `.so` Linux.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les pour éviter des frais supplémentaires. Pour ce faire, exécutez les étapes suivantes :

- Supprimez les compartiments S3 que vous avez créés pour ce didacticiel. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.
- Supprimez les politiques IAM que vous avez créées pour ce didacticiel. Pour plus d'informations, consultez la section Suppression des politiques IAM dans le guide de l'utilisateur IAM.
- Supprimez le rôle IAM que vous avez créé pour ce didacticiel. Pour plus d'informations, consultez Suppression de rôles ou de profils d'instance dans le guide de l'utilisateur IAM.
- Supprimez le CodeBuild projet que vous avez créé pour ce didacticiel. Pour plus d'informations, voir Supprimer un projet de construction CodeBuild dans le guide de AWS CodeBuild l'utilisateur.

Tutoriel : Configuration d'un pipeline CI/CD à utiliser avec Micro Focus Enterprise Developer

Ce didacticiel explique comment importer, modifier, compiler et exécuter l' BankDemo exemple d'application dans Micro Focus Enterprise Developer, puis comment valider vos modifications pour déclencher un pipeline CI/CD.

Table des matières

Prérequis

Téléchargez les fichiers suivants.

- `basic-infra.yaml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `pipeline.yaml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `m2-code-sync-function.zip`

  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `config_git.yml`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `BANKDEMO-source.zip`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).
- `BANKDEMO-exercise.zip`
  - Télécharger depuis la région Europe (Francfort).
  - Télécharger depuis la région USA Est (Virginie du Nord).

L'objectif de chaque fichier est le suivant :

`basic-infra.yaml`

Ce AWS CloudFormation modèle crée l'infrastructure de base nécessaire au pipeline CI/CD : VPC, compartiments Amazon S3, etc.

`pipeline.yaml`

Ce AWS CloudFormation modèle est utilisé par une fonction Lambda pour lancer la pile de pipelines. Assurez-vous que ce modèle se trouve dans un compartiment Amazon S3 accessible au public. Ajoutez le lien vers ce compartiment comme valeur par défaut pour le `PipelineTemplateURL` paramètre dans le `basic-infra.yaml` modèle.

`m2-code-sync-function.zip`

Cette fonction Lambda crée le CodeCommit référentiel, la structure de répertoire basée sur`config_git.yaml`, et lance la pile de pipelines à l'aide de. `pipeline.yaml` Assurez-vous que ce fichier zip est disponible dans un compartiment Amazon S3 accessible au public dans tous les pays Régions AWS où la modernisation des AWS mainframes est prise en charge. Nous vous recommandons de stocker le fichier dans un compartiment Région AWS et de le répliquer dans tous les compartiments. Régions AWS Utilisez une convention de dénomination pour le compartiment avec un suffixe identifiant le compartiment spécifique Région AWS (par exemple,`m2-cicd-deployment-source-eu-west-1`) et ajoutez le préfixe `m2-cicd-deployment-source` comme valeur par défaut pour le paramètre

`DeploymentSourceBucket` et formez le compartiment complet en utilisant la fonction de AWS CloudFormation substitution `!Sub {DeploymentSourceBucket}-${AWS::Region}` tout en faisant référence à ce compartiment dans le `basic-infra.yaml` modèle de ressource. `SourceSyncLambdaFunction`

`config_git.yml`

CodeCommit définition de la structure du répertoire. Pour plus d'informations, consultez Exemple de fichier déclencheur YAML config_git.yml.

`BANKDEMO-source.zip`.

BankDemo code source et fichier de configuration créés à partir du CodeCommit référentiel.

`BANKDEMO-exercise.zip`.

BankDemo source pour les exercices didacticiels créés à partir du CodeCommit référentiel.

Création d'une infrastructure de base pour le pipeline CI/CD

Utilisez le AWS CloudFormation modèle `basic-infra.yaml` pour créer la pile d'infrastructure de base du pipeline CI/CD via la AWS CloudFormation console. Cette pile crée des compartiments Amazon S3 dans lesquels vous pouvez télécharger le code et les données de votre application, ainsi qu'une AWS Lambda fonction de support pour créer d'autres ressources nécessaires, telles qu'un AWS CodeCommit référentiel et un AWS CodePipeline pipeline.

> **Note**
>
> Pour lancer cette pile, vous avez besoin d'autorisations pour administrer IAM, Amazon S3, Lambda AWS CloudFormation et d'autorisations d'utilisation. AWS KMS

1. Connectez-vous à la AWS Management Console et ouvrez la console AWS CloudFormation à l'adresse https://console.aws.amazon.com/cloudformation.
2. Créez une nouvelle pile avec l'une des options suivantes :
   - Sélectionnez Créer une pile. C'est la seule option disponible si une pile est en cours d'exécution.
   - Sur la page Stacks, choisissez Create Stack. Cette option n'est visible que si aucune pile n'est en cours d'exécution.

3. Sur la page Spécifier le modèle :

   - Dans Préparer le modèle, sélectionnez Le modèle est prêt.
   - Dans Spécifier le modèle, choisissez l'URL Amazon S3 comme source du modèle et entrez l'une des URL suivantes en fonction de votreRégion AWS.
     - `https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/basic-infra.yaml`
     - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/basic-infra.yaml`
   - Pour accepter vos paramètres, choisissez Next.

   La page Créer une pile s'ouvre.

Specify stack details

Stack name

Stack name

m2-cicd-pipeline-foundation

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Networking Configuration

Do you want to use an existing VPC in your account?

If you select 'Yes', then you must provide the VPC ID and the Subnet IDs.

No (Create one)

Which VPC ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID in a different AZ should be used for HA?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Enter the CIDR block that should be used for the new VPC

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

11.11.0.0/16

CIDR bits for creating subnets. Choose 5 for /27, 6 for /26, 7 for /25, 8 for /24 range

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

5

Deployment Configuration

Name of the S3 bucket which contains the source files for this stack deployment

Don't change unless you know what you are doing.

m2

Name of the source package file for the infrastructure Lambda function

Don't change unless you know what you are doing.

cicd/mf/m2-code-sync-function.zip

Full URL of the pipeline CloudFormation template file

Don't change unless you know what you are doing.

https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/pipeline.yaml

What name prefix to use for the new S3 buckets?

A name prefix for the S3 buckets that will be created by this stack.

mf

Cancel Previous Next

Effectuez les modifications suivantes :

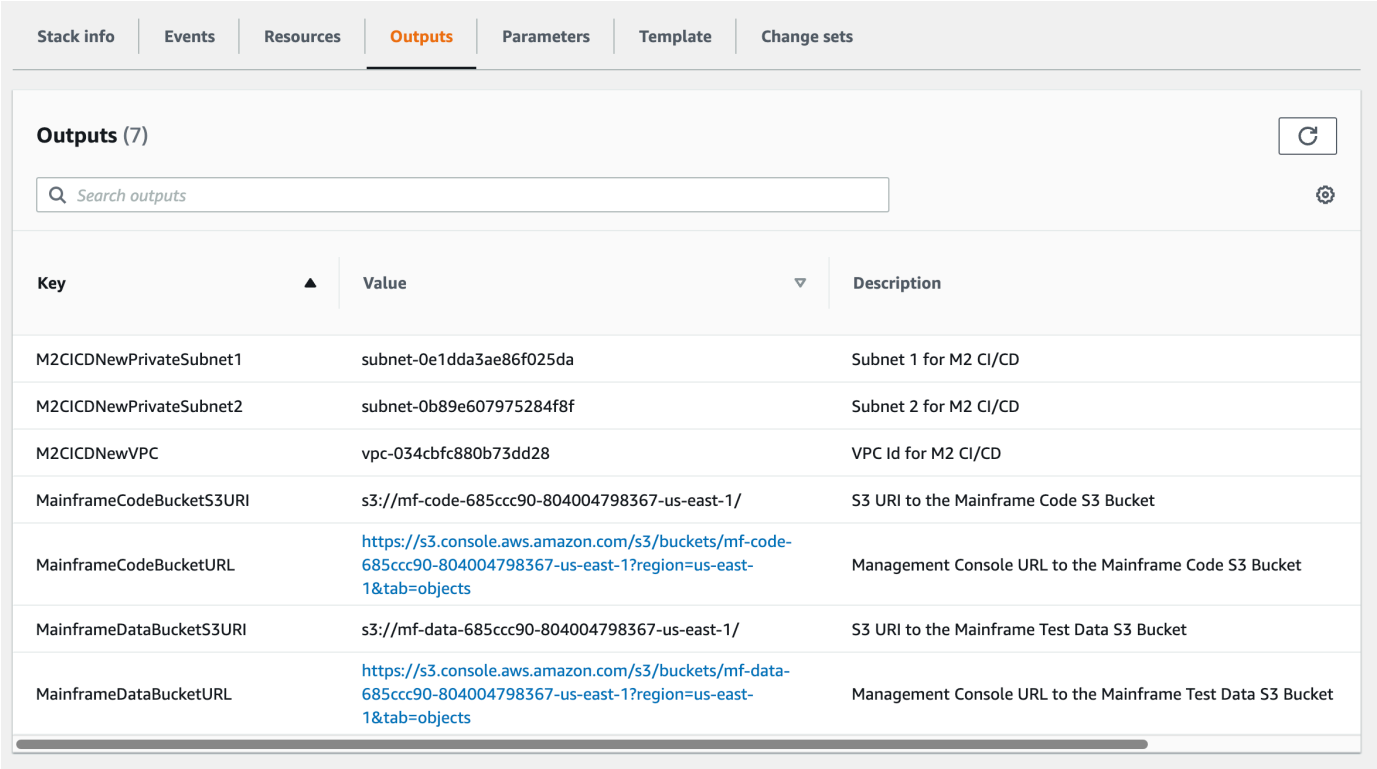
- Fournissez les valeurs appropriées pour le nom de la pile et les paramètres de configuration réseau.
- La plupart des paramètres des configurations de déploiement sont préremplis de manière appropriée, vous n'avez donc pas besoin de les modifier. Selon votre modèleRégion AWS, remplacez le AWS CloudFormation modèle de pipeline par l'une des URL Amazon S3 suivantes.
  - `https://m2-us-east-1.s3.amazonaws.com/cicd/mf/pipeline.yaml`
  - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/pipeline.yaml`
- Choisissez Suivant.

> **Note**
>
> Ne modifiez pas les valeurs des paramètres par défaut, sauf si vous avez vous-même modifié le AWS CloudFormation modèle.

4. Dans Configurer les options de pile, choisissez Next.
5. Dans Fonctionnalités, choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM pour autoriser la création AWS CloudFormation d'un rôle IAM en votre nom. Sélectionnez Créer la pile.

> **Note**
>
> Le provisionnement de cette pile peut prendre de 3 à 5 minutes.

6. Une fois la pile créée avec succès, accédez à la section Sorties de la pile nouvellement provisionnée. Vous y trouverez le compartiment Amazon S3 dans lequel vous devez télécharger le code de votre mainframe et les fichiers dépendants.



Création d'un AWS CodeCommit référentiel et d'un pipeline CI/CD

Au cours de cette étape, vous créez un CodeCommit référentiel et approvisionnez une pile de pipeline CI/CD en appelant une fonction Lambda qui appelle AWS CloudFormation pour créer la pile de pipelines.

1. Téléchargez l'BankDemo exemple d'application sur votre ordinateur local.
2. Chargez `bankdemo.zip` depuis votre machine locale vers le compartiment Amazon S3 créé dansCréation d'une infrastructure de base pour le pipeline CI/CD.
3. Téléchargement `config_git.yml`.
4. Modifiez le `config_git.yml` si nécessaire, comme suit :

   - Ajoutez votre propre nom de référentiel cible, votre branche cible et votre message de validation.

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
```

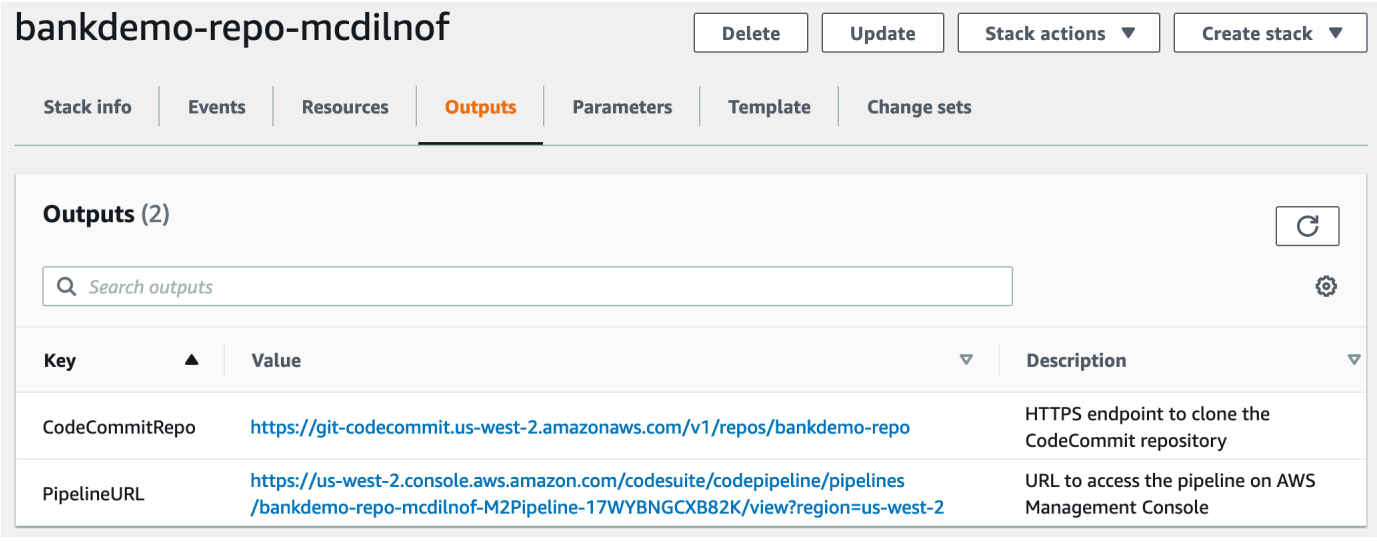
- Ajoutez l'adresse e-mail à laquelle vous souhaitez recevoir des notifications.

```
pipeline-config:
  # Send pipeline failure notifications to these email addresses
  alert-notifications:
    - myname@mycompany.com
  # Send notifications for manual approval before production deployment to these
 email addresses
  approval-notifications:
    - myname@mycompany.com
```

5. Téléchargez le `config_git.yml` fichier contenant la définition de la structure des dossiers du CodeCommit référentiel dans le compartiment Amazon S3 créé dansCréation d'une infrastructure de base pour le pipeline CI/CD. Cela invoquera la fonction Lambda qui approvisionnera automatiquement le référentiel et le pipeline.

   Cela créera un CodeCommit référentiel avec le nom fourni dans le `config_git.yml` fichier `target-repository` défini ; par exemple,`bankdemo-repo`.

   La fonction Lambda créera également la pile du pipeline CI/CD. AWS CloudFormation La AWS CloudFormation pile aura le même préfixe que le `target-repository` nom fourni, suivi d'une chaîne aléatoire (par exemple`bankdemo-repo-`*`01234567`*. Vous pouvez trouver l'URL du CodeCommit référentiel et l'URL permettant d'accéder au pipeline créé dans la console AWS de gestion.



6. Si la création du CodeCommit référentiel est terminée, le pipeline CI/CD sera immédiatement déclenché pour exécuter un CI/CD complet.

7. Une fois que le fichier a été transféré, il déclenche automatiquement le pipeline qui sera créé, déployé en phase de préparation, exécutera des tests et attendra l'approbation manuelle avant de le déployer dans l'environnement de production.

Exemple de fichier déclencheur YAML config_git.yml

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
  directory-structure:
    - '/':
        files:
        - build.xml
        - '*.yaml'
        - '*.yml'
        - '*.xml'
        - 'LICENSE.txt'
        readme: |
          # Root Folder
          - 'build.xml' : Build configuration for the application
    - tests:
        files:
        - '*.py'
        readme: |
          # Test Folder
          - '*.py' : Test scripts
    - config:
        files:
        - 'BANKDEMO.csd'
        - 'BANKDEMO.json'
        - 'BANKDEMO_ED.json'
        - 'dfhdrdat'
        - 'ESPGSQLXA.dll'
        - 'ESPGSQLXA64.so'
        - 'ESPGSQLXA64_S.so'
        - 'EXTFH.cfg'
        - 'm2-2021-04-28.normal.json'
        - 'MFDBFH.cfg'
        - 'application-definition-template-config.json'
        readme: |
          # Config Folder
```

```
          This folder contains the application configuration files.
          - 'BANKDEMO.csd'     : CICS Resource definitions export file
          - 'BANKDEMO.json'    : Enterprise Server configuration
          - 'BANKDEMO_ED.json' : Enterprise Server configuration for ED
          - 'dfhdrdat'         : CICS resource definition file
          - 'ESPGSQLXA.dll'    : XA switch module Windows
          - 'ESPGSQLXA64.so'   : XA switch module Linux
          - 'ESPGSQLXA64_S.so' : XA switch module Linux
          - 'EXTFH.cfg'        : Micro Focus File Handler configuration
          - 'm2-2021-04-28.normal.json' : M2 request document
          - 'MFDBFH.cfg'       : Micro Focus Database File Handler
          - 'application-definition-template-config.json' : Application definition for
 M2
    - source:
        subdirs:
        - .settings:
            files:
              - '.bms.mfdirset'
              - '.cbl.mfdirset'
        - copybook:
            files:
              - '*.cpy'
              - '*.inc'
            readme: |
              # Copy folder
              This folder contains the source for COBOL copy books, PLI includes, ...
              - .cpy COBOL copybooks
              - .inc PLI includes
#        - ctlcards:
#            files:
#              - '*.ctl'
#              - 'KBNKSRT1.txt'
#            readme: |
#              # Control Card folder
#              This folder contains the source for Batch Control Cards
#              - .ctl Control Cards
        - ims:
            files:
              - '*.dbd'
              - '*.psb'
            readme: |
              # ims folder
              This folder contains the IMS DB source files with the extensions
              - .dbd for IMS DBD source
```

```
              - .psb for IMS PSB source
        - jcl:
            files:
              - '*.jcl'
              - '*.ctl'
              - 'KBNKSRT1.txt'
              - '*.prc'
            readme: |
              # jcl folder
              This folder contains the JCL source files with the extensions
              - .jcl
#        - proclib:
#            files:
#              - '*.prc'
#            readme: |
#              # proclib folder
#              This folder contains the JCL procedures referenced via PROCLIB
 statements in the JCL with extensions
#              - .prc
        - rdbms:
            files:
              - '*.sql'
            readme: |
              # rdbms folder
              This folder contains any DB2 related source files with extensions
              - .sql for any kind of SQL source
        - screens:
            files:
            - '*.bms'
            - '*.mfs'
            readme: |
              # screens folder
              This folder contains the screens source files with the extensions
              - .bms for CICS BMS screens
              - .mfs for IMS MFS screens
            subdirs:
            - .settings:
                files:
                    - '*.bms.mfdirset'
        - cobol:
            files:
              - '*.cbl'
              - '*.pli'
            readme: |
```

```
              # source folder
              This folder contains the program source files with the extensions
              - .cbl for COBOL source
              - .pli for PLI source
            subdirs:
            - .settings:
                files:
                   - '*.cbl.mfdirset'
    - tests:
        files:
        - 'test_script.py'
        readme: |
          # tests Folder
          This folder contains the application test scripts
pipeline-config:
  alert-notifications:
    - myname@mycompany.com
  approval-notifications:
    - myname@mycompany.com
```

Création d'Enterprise Developer AppStream 2.0

Pour configurer Micro Focus Enterprise Developer sur AppStream 2.0, voirTutoriel : Configuration de Micro Focus Enterprise Developer sur AppStream 2.0.

Pour connecter le CodeCommit référentiel à Enterprise Developer, utilisez le nom indiqué `target-repository` dansExemple de fichier déclencheur YAML config_git.yml.

Configuration et test pour les développeurs d'entreprise

Rubriques

- Cloner le BankDemo CodeCommit référentiel dans Enterprise Developer
- Création d'un projet COBOL sur BankDemo mainframe et création d'une application
- Création d'un environnement BankDemo CICS et batch local pour les tests
- Démarrez le serveur BANKDEMO depuis Enterprise Developer
- Démarrez le terminal Rumba 3270
- Exécuter une BankDemo transaction
- Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Connectez-vous à l'instance Enterprise Developer AppStream 2.0 que vous avez créée dansCréation d'Enterprise Developer AppStream 2.0.

1. Démarrez Enterprise Developer à partir du démarrage de Windows. Choisissez Micro Focus Enterprise Developer, puis choisissez Enterprise Developer pour Eclipse. Si vous commencez pour la première fois, cela peut prendre un certain temps.
2. Dans le lanceur Eclipse, dans Workspace : entrez `C:\Users\<username>\workspace` puis choisissez Launch.

   > **Note**
   >
   > Assurez-vous de choisir le même emplacement après vous être reconnecté à l'instance AppStream 2.0. La sélection de l'espace de travail n'est pas permanente.

3. Dans Bienvenue, choisissez Open COBOL Perspective. Cela ne sera affiché que la première fois pour un nouvel espace de travail.

Cloner le BankDemo CodeCommit référentiel dans Enterprise Developer

1. Choisissez Fenêtre/Perspective /Perspective ouverte/Autre... /Git.
2. Choisissez Cloner un dépôt Git.
3. Dans Clone Git Repository, entrez les informations suivantes :

   - Dans Location URI, entrez l'URL HTTPS du CodeCommit référentiel.

     > **Note**
     >
     > Copiez l'URL de clonage HTTPS du CodeCommit référentiel dans la console AWS de gestion et collez-la ici. L'URI sera divisée en chemins d'hôte et de référentiel.

   - CodeCommit Référencez les informations d'identification de l'utilisateur dans Authentication User and Password et choisissez Stocker dans Secure Store.
4. Dans Sélection de branche, choisissez Branche principale, puis Suivant.
5. Dans Destination locale, dans Répertoire, entrez `C:\Users\<username>\workspace` et choisissez Terminer.

Le processus de clonage est terminé lorsqu'il `BANKDEMO [main]` est affiché dans la vue Git Repositories.

Création d'un projet COBOL sur BankDemo mainframe et création d'une application

1. Passez à la perspective COBOL.
2. Dans Project, désactivez la création automatique.
3. Dans Fichier, choisissez Nouveau, puis Mainframe COBOL Project.
4. Dans New Mainframe COBOL Project, entrez les informations suivantes :
   - Dans Nom du projet, entrez`BankDemo`.
   - Choisissez le modèle Micro Focus [64 bits].
   - Choisissez Finish (Terminer).
5. Dans COBOL Explorer, développez le nouveau BankDemo projet.

   > **Note**
   >
   > `[BANKDEMO main]`entre crochets indique que le projet est connecté au BankDemo CodeCommit dépôt local.

6. Si l'arborescence n'affiche pas les entrées relatives aux programmes COBOL, aux copybooks, aux sources BMS et aux fichiers JCL, choisissez Actualiser dans le menu contextuel du BankDemo projet.
7. BankDemo Dans le menu contextuel, choisissez Propriétés/Micro Focus/Paramètres du projet/COBOL :
   - Choisissez Jeu de caractères - ASCII.
   - Choisissez Appliquer, puis Fermer.
8. Si la génération de la source BMS et COBOL ne démarre pas immédiatement, vérifiez dans le menu Projet que l'option Créer automatiquement est activée.

   La sortie de compilation sera affichée dans la vue de la console et devrait être terminée au bout de quelques minutes avec des messages `BUILD SUCCESSFUL` et`Build finished with no errors`.

   L' BankDemo application doit maintenant être compilée et prête pour une exécution locale.

Création d'un environnement BankDemo CICS et batch local pour les tests

1. Dans COBOL Explorer, développezBANKDEMO / config.
2. Dans l'éditeur, ouvrezBANKDEMO_ED.json.
3. Recherchez la chaîne ED_Home= et modifiez le chemin pour pointer vers le projet Enterprise Developer, comme suit :D:\\<username>\\workspace\\BANKDEMO. Notez l'utilisation de barres obliques doubles (\\) dans la définition du chemin.
4. Enregistrez et fermez le fichier .
5. Choisissez Server Explorer.
6. Dans le menu contextuel par défaut, choisissez Ouvrir la page d'administration. La page d'administration de Micro Focus Enterprise Server s'ouvre dans le navigateur par défaut.
7. Pour les sessions AppStream 2.0 uniquement, apportez les modifications suivantes afin de préserver votre région Enterprise Server locale pour les tests locaux :

   - Dans Directory Server/Default, choisissez PROPERTIES/Configuration.
   - Remplacez l'emplacement du référentiel parD:\<username>\My Files\Home Folder \MFDS.

   > **Note**
   >
   > Vous devez effectuer les étapes 5 à 8 après chaque nouvelle connexion à une instance AppStream 2.0.

8. Dans Directory Server/Default, choisissez Importer, puis effectuez les étapes suivantes :

   - À l'étape 1 : Type d'importation, choisissez JSON, puis Next.
   - À l'étape 2 : Télécharger, cliquez pour télécharger le fichier dans un carré bleu.
   - Dans Choisir le fichier à télécharger, entrez :
     - Nom du fichier :D:\<username>\workspace\BANKDEMO\config\BANKDEMO_ED.json.
     - Choisissez Ouvrir.
   - Choisissez Suivant.
   - À l'étape 3 : les régions effacent les ports des points de terminaison.
   - Choisissez Suivant.
   - À l'étape 4 : Importer, choisissez Importer.

- Choisissez Finish (Terminer).

La liste affiche désormais un nouveau nom de serveur`BANKDEMO`.

Démarrez le serveur BANKDEMO depuis Enterprise Developer

1. Choisissez Enterprise Developer.
2. Dans l'Explorateur de serveurs, choisissez Par défaut, puis sélectionnez Actualiser dans le menu contextuel.

   La liste des serveurs devrait désormais également afficher BANKDEMO.
3. Choisissez BANKDEMO.
4. Dans le menu contextuel, choisissez Associer au projet, puis BANKDEMO.
5. Dans le menu contextuel, choisissez Démarrer.

   La vue de la console doit afficher le journal du démarrage du serveur.

   Si le message `BANKDEMO CASSI5030I PLTPI Phase 2 List(PI) Processing Completed` s'affiche, le serveur est prêt à tester l'application CICS BANKDEMO.

Démarrez le terminal Rumba 3270

1. Depuis le démarrage de Windows, lancez Micro Focus Rumba+ Desktop/Rumba+ Desktop.
2. Dans Bienvenue, choisissez CREATE NEW SESSION/Mainframe Display.
3. Dans Mainframe Display, choisissez Connection/ Configure.
4. Dans Configuration de session, choisissez Connection/TN3270.
5. Dans Nom d'hôte/adresse, choisissez Insérer et entrez l'adresse IP`127.0.0.1`.
6. Dans Port Telnet, entrez port`6000`.
7. Choisissez Appliquer.
8. Choisissez Se connecter.

   L'écran de bienvenue du CICS affiche un écran avec le message de ligne 1 :`This is the Micro Focus MFE CICS region BANKDEMO`.
9. Appuyez sur Ctrl+Shift+Z pour effacer l'écran.

Exécuter une BankDemo transaction

1. Dans un écran vide, entrez`BANK`.
2. Dans l'écran BANK10, dans le champ de saisie du nom d'utilisateur... :, entrez `guest` et appuyez sur Entrée.
3. Dans l'écran BANK20, dans le champ de saisie situé avant Calculer le coût d'un prêt, entrez `/` (barre oblique) et appuyez sur Entrée.
4. Dans l'écran BANK70 :

   - Dans Le montant que vous souhaitez emprunter... :, entrez`10000`.
   - Dans À un taux d'intérêt de... :, entrez`5.0`.
   - Dans Depuis combien de mois... :, entrez`10`.
   - Appuyez sur Entrée.

   Le résultat suivant doit être affiché :

   ```
   Resulting monthly payment.............:   $1023.06
   ```

   Ceci termine la configuration de l'application BANKDEMO dans Enterprise Developer.

Arrêtez le serveur BANKDEMO depuis Enterprise Developer

1. Dans l'Explorateur de serveurs, choisissez Par défaut, puis sélectionnez Actualiser dans le menu contextuel.
2. Choisissez BANKDEMO.
3. Dans le menu contextuel, choisissez Stop.

   La vue de la console doit afficher le journal de l'arrêt du serveur.

   Si le message `Server: BANKDEMO stopped successfully` s'affiche, cela signifie que le serveur s'est correctement arrêté.

Exercice 1 : Améliorer le calcul du prêt dans l'application BANKDEMO

Rubriques

Dans ce scénario, vous allez suivre le processus consistant à apporter un exemple de modification au code, à le déployer et à le tester.

Le service des prêts souhaite un nouveau champ sur l'écran de calcul du prêt BANK70 pour afficher le montant total du prêt. Cela nécessite une modification de l'écran BMS MBANK70.CBL, l'ajout d'un nouveau champ et le programme de gestion d'écran correspondant SBANK70P.CBL avec les copybooks associés. En outre, la routine de calcul des prêts dans BBANK70P.CBL doit être étendue avec la formule supplémentaire.

Pour terminer cet exercice, assurez-vous de remplir les conditions préalables suivantes.

- Téléchargez BANKDEMO-exercise.zip pour`D:\PhotonUser\My Files\Home Folder`.
- Extrayez le fichier zip dans`D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise`.
- Créez un dossier`D:\PhotonUser\My Files\Home Folder\AnalysisRules`.
- Copiez le fichier `Loan+Calculation+Update.General-1.xml` de règles du `BANKDEMO-exercise` dossier vers`D:\PhotonUser\My Files\Home Folder\AnalysisRules`.

> **Note**
>
> Les modifications de code dans *.CBL et *.CPY sont signalées par EXER01 dans les colonnes 1 à 6 pour cet exercice.

Ajouter une règle d'analyse des prêts à Enterprise Developer Code Analysis

Les règles d'analyse définies dans Micro Focus Enterprise Analyzer peuvent être exportées depuis Enterprise Analyzer et importées dans Enterprise Developer pour exécuter les mêmes règles d'analyse dans toutes les sources du projet Enterprise Developer.

1. Ouvrir `Window/Preferences/Micro Focus/COBOL/Code Analysis/Rules`.

2. Choisissez Modifier... et entrez le nom du dossier `D:\PhotonUser\My Files \Home Folder\AnalysisRules` contenant le fichier de règles`Loan+Calculation +Update.General-1.xml`.
3. Choisissez Finish (Terminer).
4. Choisissez Appliquer, puis Fermer.
5. Dans le menu contextuel du projet BANKDEMO, choisissez Code Analysis.

   Vous devriez voir une entrée pour la mise à jour du calcul du prêt.

Étape 1 : Effectuer une analyse de code pour le calcul du prêt

Avec la nouvelle règle d'analyse, nous voulons identifier les programmes COBOL et les lignes de code qui correspondent aux modèles de recherche, `*LOAN*` ainsi que `*RATE*` dans les expressions`*PAYMENT*`, les instructions et les variables. Cela vous aidera à naviguer dans le code et à identifier les modifications de code requises.

1. Dans le menu contextuel du projet BANKDEMO, choisissez Analyse du code/Mise à jour du calcul du prêt.

   Cela exécutera la règle de recherche et listera les résultats dans un nouvel onglet appelé Analyse du code. L'analyse est terminée lorsque la barre de progression verte en bas à droite disparaît.

   L'onglet Analyse du code doit afficher une liste détaillée des instructions`BBANK20P.CBL`, expressions `BBANK70P.CBL` et `SBANK70P.CBL` variables correspondant aux modèles de recherche, et chacune répertoriant chacune d'entre elles.

   Si vous regardez le résultat`BBANK20P.CBL`, seuls les littéraux déplacés correspondent au modèle de recherche. Ce programme peut donc être ignoré.
2. Dans la barre de menu de l'onglet, choisissez - Icône pour tout réduire.
3. Développez `SBANK70P.CBL` et sélectionnez n'importe quelle ligne dans n'importe quel ordre en double-cliquant pour voir comment cela ouvrira le code source et surlignera la ligne sélectionnée dans le code source. Vous reconnaîtrez également que toutes les lignes de source identifiées sont marquées.

Étape 2 : Modifier la carte CICS BMS et le programme COBOL et tester

Nous allons d'abord modifier la carte `MBANK70.BMS` BMS, le programme de gestion d'écran `SBANK70P.CBL` et le cahier `CBANKDAT.CPY` pour afficher le nouveau champ. Pour éviter tout codage inutile dans cet exercice, les modules source modifiés sont disponibles dans le `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01` dossier. Normalement, un développeur utilise les résultats de l'analyse du code pour naviguer et modifier les sources. Si vous avez le temps et souhaitez effectuer les modifications manuelles, utilisez les informations fournies dans *Modification manuelle dans MBANK70.BMS et SBANK70P.CBL (facultatif) *.

Pour des modifications rapides, copiez les fichiers suivants :

1. `..\BANKDEMO-exercise\Exercis01\screens\MBANK70.BMS` à `D:\PhotonUser\workspace\bankdemo\source\screens`.
2. `.\BANKDEMO-exercise\Exercis01\cobol\SBANK70P.CBL` à `D:\PhotonUser\workspace\bankdemo\source\cobol`.
3. `..\BANKDEMO-exercise\Exercis01\copybook\CBANKDAT.CPY` à `D:\PhotonUser\workspace\bankdemo\source\copybook`.
4. Pour vous assurer que tous les programmes concernés par les modifications sont compilés, choisissez Project/Clean... /Nettoyez tous les projets.

Pour les modifications manuelles apportées à `MBANK70.BMS` et`SBANK70P.CBL`, procédez comme suit :

- Pour une modification manuelle de la `MBANK70.BMS` source BMS, ajoutez après le `PAYMENT` champ :
  - TXT09 avec les mêmes attributs que TXT08 et valeur INITIALE « Montant total du prêt »
  - TOTAL avec les mêmes attributs que PAYMENT

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes :

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270

3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................:`.

4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Étape 3 : Ajouter le calcul du montant total dans le programme COBOL

Dans un deuxième temps, nous modifierons `BBANK70P.CBL` et ajouterons le calcul du montant total du prêt. La source préparée avec les modifications requises est disponible dans `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01` le dossier. Si vous avez le temps et souhaitez effectuer les modifications manuelles, utilisez les informations fournies dans *Modification manuelle dans BBANK70P.CBL (facultatif) *.

Pour un changement rapide, copiez le fichier suivant :

- `..\BANKDEMO-exercise\Exercis01\source\cobol\BBANK70P.CBL` à `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Pour apporter une modification manuelle à`BBANK70P.CBL`, procédez comme suit :

- Utilisez le résultat de l'analyse du code pour identifier les modifications requises.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes :

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.

4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

Étape 4 : valider les modifications et exécuter le pipeline CI/CD

Validez les modifications dans le CodeCommit référentiel central et déclenchez le pipeline CI/CD pour créer, tester et déployer les modifications.

1. Dans le projet BANKDEMO, dans le menu contextuel, choisissez Team/Commit.
2. Dans l'onglet Git Staging, entrez le message de validation suivant : `Added Total Amount Calculation`
3. Choisissez Commit et Push... .
4. Ouvrez la CodePipeline console et vérifiez l'état de l'exécution du pipeline.

> **Note**
>
> Si vous rencontrez un problème avec la fonction Commit ou Push de Teams ou Enterprise Developer, utilisez l'interface de ligne de commande Git Bash.

Exercice 2 : Extraire le calcul du prêt dans BankDemo l'application

Rubriques

Dans le prochain exercice, vous allez travailler sur un autre exemple de demande de modification. Dans ce scénario, le service des prêts souhaite réutiliser la routine de calcul des prêts de manière autonome. WebService La routine doit rester en COBOL et doit également être appelable à partir du programme CICS COBOL existant. `BBANK70P.CBL`

Étape 1 : Refactoriser la routine de calcul des prêts dans une section COBOL

Dans un premier temps, nous extrayons la routine de calcul du prêt dans une section COBOL. Cette étape est nécessaire pour extraire le code dans un programme COBOL autonome à l'étape suivante.

1. Ouvrez `BBANK70P.CBL` dans l'éditeur COBOL.

2. Dans l'éditeur, choisissez dans le menu contextuel Analyse du code/Mise à jour du calcul du prêt. Cela analysera uniquement la source actuelle à la recherche de modèles définis dans la règle d'analyse.
3. Dans le résultat de l'onglet Analyse du code, recherchez la première instruction `DIVIDE WS-LOAN-INTEREST BY 12` arithmétique.
4. Double-cliquez sur l'instruction pour accéder à la ligne source dans l'éditeur. Il s'agit du premier énoncé de la routine de calcul des prêts.
5. Marquez le bloc de code suivant pour que la routine de calcul du prêt soit extraite dans une section.

```
DIVIDE WS-LOAN-INTEREST BY 12
             GIVING WS-LOAN-INTEREST ROUNDED.
           COMPUTE WS-LOAN-MONTHLY-PAYMENT ROUNDED =
             ((WS-LOAN-INTEREST * ((1 + WS-LOAN-INTEREST)
                 ** WS-LOAN-TERM)) /
             (((1 + WS-LOAN-INTEREST) * WS-LOAN-TERM) - 1 ))
                 * WS-LOAN-PRINCIPAL.
EXER01     COMPUTE WS-LOAN-TOTAL-PAYMENT =
EXER01               (WS-LOAN-MONTHLY-PAYMENT * WS-LOAN-TERM).
```

6. Dans le menu contextuel de l'éditeur, choisissez Refactor/Extract to Section... .
7. Entrez le nom de la nouvelle section : LOAN-CALCULATION.
8. Choisissez OK.

   Le bloc de code marqué a maintenant été extrait dans la nouvelle `LOAN-CALCULATION` section et le bloc de code a été remplacé par l'`PERFROM LOAN-CALCULATIO`Ninstruction.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes.

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.
4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

> **Note**
>
> Si vous souhaitez éviter les étapes ci-dessus pour extraire le bloc de code dans une section, vous pouvez copier la source modifiée pour l'étape 1 de `..\BANKDEMO-exercise\Exercis02\Step1\cobol\BBANK70P.CBL` vers`D:\PhotonUser\workspace\bankdemo\source\cobol`.

Étape 2 : Extraire la routine de calcul des prêts dans un programme COBOL autonome

À l'étape 2, le bloc de code de la `LOAN-CALCULATION` section sera extrait vers un programme autonome et le code d'origine sera remplacé par un code permettant d'appeler le nouveau sous-programme.

1. Ouvrez `BBANK70P.CBL` dans l'éditeur et recherchez la nouvelle `PERFORM LOAN-CALCULATION` déclaration créée à l'étape 1.
2. Placez le curseur dans le nom de la section. Il sera marqué en gris.
3. Dans le menu contextuel, sélectionnez Refactor->Extraire la section/le paragraphe vers le programme... .
4. Dans Extraire la section/le paragraphe à programmer, entrez le nouveau nom de fichier : LOANCALC.CBL.
5. Choisissez OK.

   Le nouveau `LOANCALC.CBL` programme s'ouvre dans l'éditeur.
6. Faites défiler la page vers le bas et passez en revue le code extrait et généré pour l'interface d'appel.
7. Sélectionnez l'éditeur avec `BBANK70P.CBL` et accédez à`LOAN-CALCULATION SECTION`. Vérifiez le code généré pour appeler le nouveau sous-programme`LOANCALC.CBL`.

> **Note**
>
> L'`CALL`instruction utilise `DFHEIBLK` et appelle `LOANCALC` avec `DFHCOMMAREA` des blocs de contrôle CICS. Comme nous voulons appeler le nouveau `LOANCALC.CBL` sous-programme un programme non-CICS, nous devons supprimer `DFHEIBLK` et `DFHCOMMAREA` annuler l'appel en commentant ou en supprimant.

Changements de test

Pour tester les modifications, répétez les étapes décrites dans les sections suivantes.

1. Démarrez le serveur BANKDEMO depuis Enterprise Developer
2. Démarrez le terminal Rumba 3270
3. Exécuter une BankDemo transaction

   De plus, vous devriez maintenant également voir le texte`Total Loan Amount.....................: $10230.60`.
4. Arrêtez le serveur BANKDEMO depuis Enterprise Developer

> **Note**
>
> Si vous souhaitez éviter les étapes ci-dessus pour extraire le bloc de code dans une section, vous pouvez copier la source modifiée pour l'étape 1 depuis `..\BANKDEMO-exercise\Exercis02\Step2\cobol\BBANK70P.CBL` et `LOANCALC.CBL` vers`D:\PhotonUser\workspace\bankdemo\source\cobol`.

Étape 3 : valider les modifications et exécuter le pipeline CI/CD

Validez les modifications dans le CodeCommit référentiel central et déclenchez le pipeline CI/CD pour créer, tester et déployer les modifications.

1. Dans le projet BANKDEMO, dans le menu contextuel, choisissez Team/Commit.
2. Dans l'onglet Git Staging
   - Ajoutez les étapes non mises en scène LOANCALC.CBL et LoanCalc.CBL.MFDirSet.
   - Entrez un message de validation :`Added Total Amount Calculation`.
3. Choisissez Commit et Push... .
4. Ouvrez la CodePipeline console et vérifiez l'état de l'exécution du pipeline.

> **Note**
>
> Si vous rencontrez un problème avec la fonction Commit ou Push de Teams ou Enterprise Developer, utilisez l'interface de ligne de commande Git Bash.

Nettoyage des ressources

Si vous n'avez plus besoin des ressources que vous avez créées pour ce didacticiel, supprimez-les afin qu'elles ne continuent pas à vous être facturées. Procédez comme suit :

- Supprimez le CodePipeline pipeline. Pour plus d'informations, voir Supprimer un pipeline CodePipeline dans le Guide de AWS CodePipeline l'utilisateur.
- Supprimez le CodeCommit référentiel. Pour plus d'informations, voir Supprimer un CodeCommit référentiel dans le Guide de AWS CodeCommit l'utilisateur.
- Supprimez le compartiment S3 ;. Pour plus d'informations, consultez Supprimer un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.
- Supprimez la AWS CloudFormation pile. Pour plus d'informations, consultez Suppression d'une pile dans la console AWS CloudFormation dans le Guide de l'utilisateur AWS CloudFormation.

Utilitaires par lots dans le cadre de la modernisation des AWS mainframes

Les applications mainframe utilisent souvent des programmes utilitaires par lots pour exécuter des fonctions spécifiques telles que le tri des données, le transfert de fichiers par FTP, le chargement de données dans des bases de données telles que DB2, le déchargement de données depuis des bases de données, etc.

Lorsque vous migrez vos applications vers la modernisation du AWS mainframe, vous avez besoin d'utilitaires de remplacement fonctionnellement équivalents, capables d'effectuer les mêmes tâches que ceux que vous utilisiez sur le mainframe. Certains de ces utilitaires sont peut-être déjà disponibles dans le cadre des moteurs d'exécution AWS Mainframe Modernization, mais nous proposons les utilitaires de remplacement suivants :

- M2SFTP - permet un transfert de fichiers sécurisé à l'aide du protocole SFTP.

- M2WAIT - attend pendant un certain temps avant de passer à l'étape suivante d'un traitement par lots.
- TXT2PDF - convertit les fichiers texte au format PDF.
- M2DFUTIL : fournit des fonctions de sauvegarde, de restauration, de suppression et de copie sur des ensembles de données similaires au support fourni par l'utilitaire ADRDSSU du mainframe.
- M2RUNCMD : permet d'exécuter des commandes, des scripts et des appels système Micro Focus directement depuis JCL.

Nous avons développé ces utilitaires par lots en fonction des commentaires des clients et les avons conçus pour fournir les mêmes fonctionnalités que les utilitaires du mainframe. L'objectif est de faciliter au maximum votre transition du mainframe à la modernisation du AWS mainframe.

Rubriques

Emplacement binaire

Ces utilitaires sont préinstallés sur les produits Micro Focus Enterprise Developer (ED) et Micro Focus Enterprise Server (ES). Vous pouvez les trouver à l'emplacement suivant pour toutes les variantes de ED et ES :

- Linux : `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Utilitaire M2SFTP Batch

M2SFTP est un utilitaire JCL conçu pour effectuer des transferts de fichiers sécurisés entre systèmes à l'aide du protocole SFTP (Secure File Transfer Protocol). Le programme utilise le client Putty SFTP

pour effectuer `psftp` les transferts de fichiers réels. Le programme fonctionne de la même manière qu'un utilitaire FTP sur ordinateur central et utilise l'authentification par utilisateur et mot de passe.

> **Note**
>
> L'authentification par clé publique n'est pas prise en charge.

Pour convertir les JCL FTP de votre mainframe en SFTP, remplacez par. `PGM=FTP` `PGM=M2SFTP`

Rubriques

Plateformes prises en charge

Vous pouvez utiliser M2SFTP sur l'une des plateformes suivantes :

- AWS Modernisation du mainframe | Micro Focus Managed
- Micro Focus Runtime (sur Amazon EC2)
- Toutes les variantes des produits Micro Focus Enterprise Developer (ED) et Micro Focus Enterprise Server (ES).

Installation des dépendances

Pour installer le client SFTP Putty sous Windows

- Téléchargez le client SFTP PuTTY et installez-le.

Pour installer le client SFTP Putty sous Linux :

- Exécutez la commande suivante pour installer le client SFTP Putty :

```
sudo yum -y install putty
```

Configuration du M2SFTP pour la gestion de la modernisation du AWS mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, vous devez configurer M2SFTP comme suit.

- Définissez les variables d'environnement Micro Focus Enterprise Server appropriées pour le MFFTP. Voici quelques exemples :
  - `MFFTP_TEMP_DIR`
  - `MFFTP_SENDEOL`
  - `MFFTP_TIME`
  - `MFFTP_ABEND`

  Vous pouvez définir aussi peu ou autant de variables que vous le souhaitez. Vous pouvez les définir dans votre JCL à l'aide de l'`ENVAR DD`instruction. Pour plus d'informations sur ces variables, consultez la section Variables de contrôle MFFTP dans la documentation de Micro Focus.

Pour tester votre configuration, consultezExemples de JCL.

Configuration du M2SFTP pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2SFTP comme suit.

1. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.

- Linux : `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

2. Définissez les variables d'environnement Micro Focus Enterprise Server appropriées pour le MFFTP. Voici quelques exemples :

   - `MFFTP_TEMP_DIR`
   - `MFFTP_SENDEOL`
   - `MFFTP_TIME`
   - MFFTP_ABEND

   Vous pouvez définir aussi peu ou autant de variables que vous le souhaitez. Vous pouvez les définir dans votre JCL à l'aide de l'`ENVAR DD`instruction. Pour plus d'informations sur ces variables, consultez la section Variables de contrôle MFFTP dans la documentation de Micro Focus.

Pour tester votre configuration, consultezExemples de JCL.

Exemples de JCL

Pour tester l'installation, vous pouvez utiliser l'un des exemples de fichiers JCL suivants.

M2SFTP1.jcl

Cette JCL montre comment appeler M2SFTP pour envoyer un fichier à un serveur SFTP distant. Notez les variables d'environnement définies dans l'`ENVVAR DD`instruction.

```
//M2SFTP1 JOB 'M2SFTP1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Sample SFTP JCL step to send a file to SFTP server*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP,
//            PARM='127.0.0.1 (EXIT=99 TIMEOUT 300'
```

```
//*
//SYSFTPD  DD  *
RECFM FB
LRECL 80
SBSENDEOL CRLF
MBSENDEOL CRLF
TRAILINGBLANKS FALSE
/*
//NETRC    DD  *
machine 127.0.0.1 login sftpuser password sftppass
/*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
type a
locsite notrailingblanks
cd files
put 'AWS.M2.TXT2PDF1.PDF' AWS.M2.TXT2PDF1.pdf
put 'AWS.M2.CARDDEMO.CARDDATA.PS' AWS.M2.CARDDEMO.CARDDATA.PS1.txt
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

M2SFTP2.jcl

Cette JCL montre comment appeler M2SFTP pour recevoir un fichier d'un serveur SFTP distant. Notez les variables d'environnement définies dans l'`ENVVAR  DD`instruction.

```
//M2SFTP2 JOB 'M2SFTP2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Sample SFTP JCL step to receive a file from SFTP server*
//*-------------------------------------------------------------------**
//*
```

```
//STEP01 EXEC PGM=M2SFTP
//*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
open 127.0.0.1
sftpuser
sftppass
cd files
locsite recfm=fb lrecl=150
get AWS.M2.CARDDEMO.CARDDATA.PS.txt +
'AWS.M2.CARDDEMO.CARDDATA.PS2' (replace
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

> **Note**
>
> Nous recommandons vivement de stocker les informations d'identification FTP dans un fichier NETRC et de restreindre l'accès aux seuls utilisateurs autorisés.

Référence de commande client Putty SFTP (PSFTP)

Le client PSFTP ne prend pas en charge toutes les commandes FTP. La liste suivante répertorie toutes les commandes prises en charge par PSFTP.

Command	Description
!	Exécuter une commande locale
au revoir	Terminez votre session SFTP
cd	Modifier votre répertoire de travail à distance

Command	Description
chmod	Modifier les autorisations et les modes des fichiers
close	Terminez votre session SFTP mais ne quittez pas PSFTP
del	Supprimer des fichiers sur le serveur distant
dir	Lister les fichiers distants
exit	Terminez votre session SFTP
get	Téléchargez un fichier depuis le serveur vers votre machine locale
aide	Donnez de l'aide
lcd	Modifier le répertoire de travail local
lpwd	Imprimer le répertoire de travail local
ls	Lister les fichiers distants
mget	Téléchargez plusieurs fichiers à la fois
mkdir	Création de répertoires sur le serveur distant
mput	Téléchargez plusieurs fichiers à la fois
mv	Déplacer ou renommer un ou plusieurs fichiers sur le serveur distant
ouvrir	Se connecter à un hôte
put	Téléchargez un fichier depuis votre machine locale vers le serveur
mobylette	Imprimez votre répertoire de télétravail

Command	Description
quit	Terminez votre session SFTP
reget	Poursuivre le téléchargement des fichiers
ren	Déplacer ou renommer un ou plusieurs fichiers sur le serveur distant
réputation	Poursuivre le téléchargement des fichiers
rm	Supprimer des fichiers sur le serveur distant
rmdir	Supprimer des répertoires sur le serveur distant

Étapes suivantes

Pour charger et télécharger des fichiers dans Amazon Simple Storage Service à l'aide du protocole SFTP, vous pouvez utiliser le protocole M2SFTP conjointement avec le protocole AWS Transfer Family, comme décrit dans les articles de blog suivants.

- Utilisation de répertoires logiques AWS SFTP pour créer un service de distribution de données simple
- Activer l'authentification par mot de passe pour AWS Transfer for SFTP l'utilisation AWS Secrets Manager

Utilitaire M2WAIT Batch

M2WAIT est un utilitaire pour mainframe qui vous permet d'introduire une période d'attente dans vos scripts JCL en spécifiant une durée en secondes, minutes ou heures. Vous pouvez appeler M2WAIT directement depuis JCL en indiquant le temps d'attente comme paramètre d'entrée. En interne, le programme M2WAIT appelle le module fourni par Micro Focus `C$SLEEP` pour attendre un temps spécifié.

> **Note**
>
> Vous pouvez utiliser des alias Micro Focus pour remplacer le contenu de vos scripts JCL. Pour plus d'informations, consultez la section JES Alias dans la documentation de Micro Focus.

Rubriques

Plateformes prises en charge

Vous pouvez utiliser M2WAIT sur l'une des plateformes suivantes :

- AWS Modernisation du mainframe | Micro Focus Managed
- Micro Focus Runtime (sur Amazon EC2)
- Toutes les variantes des produits Micro Focus Enterprise Developer (ED) et Micro Focus Enterprise Server (ES).

Configuration de M2WAIT pour la gestion de la AWS modernisation du mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, vous devez configurer M2WAIT comme suit.

- Utilisez le programme M2WAIT dans votre JCL en passant le paramètre d'entrée comme indiqué dans. Exemple de JCL

Configuration de M2WAIT pour le runtime de modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2WAIT comme suit.

1. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.
   - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Utilisez le programme M2WAIT dans votre JCL en transmettant le paramètre d'entrée comme indiqué dans. Exemple de JCL

Exemple de JCL

Pour tester l'installation, vous pouvez utiliser le `M2WAIT1.jcl` programme.

Cet exemple de JCL montre comment appeler M2WAIT et lui transmettre plusieurs durées différentes.

```
//M2WAIT1 JOB 'M2WAIT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Wait for 12 Seconds*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2WAIT,PARM='S012'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 0 Seconds (defaulted to 10 Seconds)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=M2WAIT,PARM='S000'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 1 Minute*
//*-------------------------------------------------------------------**
//*
//STEP03 EXEC PGM=M2WAIT,PARM='M001'
```

```
//SYSOUT DD SYSOUT=*
//*
//
```

Utilitaire TXT2PDF Batch

TXT2PDF est un utilitaire mainframe couramment utilisé pour convertir un fichier texte en fichier PDF. Cet utilitaire utilise le même code source pour TXT2PDF (logiciel gratuit z/OS). Nous l'avons modifié pour qu'il fonctionne dans l'environnement d'exécution AWS Mainframe Modernization Micro Focus.

Rubriques

Plateformes prises en charge

Vous pouvez utiliser TXT2PDF sur l'une des plateformes suivantes :

- AWS Modernisation du mainframe | Micro Focus Managed
- Micro Focus Runtime (sur Amazon EC2)
- Toutes les variantes des produits Micro Focus Enterprise Developer (ED) et Micro Focus Enterprise Server (ES).

Configuration de TXT2PDF pour AWS la gestion de la modernisation du mainframe

Si vos applications migrées s'exécutent sur AWS Mainframe Modernization Managed, configurez TXT2PDF comme suit.

- Créez une bibliothèque REXX EXEC appelée. `AWS.M2.REXX.EXEC` Téléchargez ces modules REXX et copiez-les dans la bibliothèque.
  - `TXT2PDF.rex`- Logiciel gratuit TXT2PDF z/OS (modifié)

- `TXT2PDFD.rex`- Logiciel gratuit TXT2PDF z/OS (non modifié)
- `TXT2PDFX.rex`- Logiciel gratuit TXT2PDF z/OS (modifié)
- `M2GETOS.rex`- Pour vérifier le type de système d'exploitation (Windows ou Linux)

Pour tester votre configuration, consultezExemple de JCL.

Configurer TXT2PDF pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez TXT2PDF comme suit.

1. Définissez la variable `MFREXX_CHARSET` d'environnement Micro Focus sur la valeur appropriée, telle que « `A` » pour les données ASCII.

   > **Important**
   >
   > La saisie d'une valeur incorrecte peut entraîner des problèmes de conversion des données (de l'EBCDIC au format ASCII), rendant ainsi le PDF illisible ou inutilisable. Nous vous recommandons de `MFREXX_CHARSET` le régler en conséquence`MF_CHARSET`.

2. Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez des deux-points (`:`) pour séparer les chemins sous Linux des points-virgules (`;`) sous Windows.

   - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

3. Créez une bibliothèque REXX EXEC appelée. `AWS.M2.REXX.EXEC` ` Téléchargez ces modules REXX et copiez-les dans la bibliothèque.

   - `TXT2PDF.rex`- Logiciel gratuit TXT2PDF z/OS (modifié)
   - `TXT2PDFD.rex`- Logiciel gratuit TXT2PDF z/OS (non modifié)
   - `TXT2PDFX.rex`- Logiciel gratuit TXT2PDF z/OS (modifié)
   - `M2GETOS.rex`- Pour vérifier le type de système d'exploitation (Windows ou Linux)

Pour tester votre configuration, consultezExemple de JCL.

Exemple de JCL

Pour tester l'installation, vous pouvez utiliser l'un des exemples de fichiers JCL suivants.

TXT2PDF1.jcl

Cet exemple de fichier JCL utilise un nom DD pour la conversion TXT2PDF.

```
//TXT2PDF1 JOB 'TXT2PDF1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
```

```
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//OUTDD    DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=VB,BLKSIZE=0)
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT DD:OUTDD +
CC YES
/*
//*
//*-------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF1.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

TXT2PDF2.jcl

Cet exemple de JCL utilise un nom DSN pour la conversion TXT2PDF.

```
//TXT2PDF2 JOB 'TXT2PDF2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* PRE DELETE*
```

```
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF2.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT 'AWS.M2.TXT2PDF2.PDF.VB' +
CC YES
/*
//*
//*--------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*--------------------------------------------------------------------**
//*
```

```
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF2.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

Modifications

Pour exécuter le programme TXT2PDF dans l'environnement d'exécution AWS Mainframe Modernization Micro Focus, nous avons apporté les modifications suivantes :

- Modifications apportées au code source pour garantir la compatibilité avec le moteur d'exécution Micro Focus REXX
- Modifications visant à garantir que le programme peut s'exécuter à la fois sur les systèmes d'exploitation Windows et Linux
- Modifications pour prendre en charge les environnements d'exécution EBCDIC et ASCII

Références

Références et code source TXT2PDF :

- Convertisseur de texte en PDF
- Outils TCP/IP et de messagerie z/OS Freeware
- Guide de référence de l'utilisateur TXT2PDF

Utilitaire M2DFUTIL Batch

M2DFUTIL est un utilitaire JCL qui fournit des fonctions de sauvegarde, de restauration, de suppression et de copie sur des ensembles de données, de la même manière que le support fourni par l'utilitaire ADRDSSU du mainframe. Ce programme conserve de nombreux paramètres SYSIN d'ADRDSSU, ce qui simplifie le processus de migration vers ce nouvel utilitaire.

Rubriques

Plateformes prises en charge

Vous pouvez utiliser M2DFUTIL sur l'une des plateformes suivantes :

- Micro Focus ES sous Windows (64 bits et 32 bits)
- Micro Focus ES sous Linux (64 bits)

Exigences relatives à la plateforme

M2DFUTIL dépend de l'appel d'un script pour effectuer un test d'expression régulière. Sous Windows, vous devez installer Windows Services pour Linux (WSL) pour que ce script s'exécute.

Assistance future prévue

Les fonctionnalités qui ne sont pas actuellement disponibles dans l'utilitaire ADRDSSU du mainframe, mais qui seront étendues à l'avenir sont les suivantes :

- M2 géré
- VSAM
- Support COPY pour le changement de nom de fichier
- Support de renommage pour RESTORE
- Plusieurs options INCLUDE et EXCLUDE
- Clause BY pour la sous-sélection par DSORG, CREDT, EXPDT
- Clause MWAIT permettant de réessayer les échecs de la file d'attente
- Support de stockage S3 pour DUMP/RESTORE

Localisation des actifs

Le module de chargement de cet utilitaire est `M2DFUTIL.so` appelé sous `M2DFUTIL.dll` Linux et Windows. Ce module de chargement se trouve aux emplacements suivants :

- Linux : `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Le script utilisé pour tester les expressions régulières est appelé`compare.sh`. Ce script se trouve aux emplacements suivants :

- Linux : `/opt/aws/m2/microfocus/utilities/scripts`
- Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\scripts`

Configurer le runtime M2DFUTIL ou AWS Mainframe Modernization sur Amazon EC2 (y compris 2.0) AppStream

Configurez la région de votre serveur d'entreprise avec les éléments suivants :

- Ajoutez les variables suivantes dans [ES-Environment]
  - `M2DFUTILS_BASE_LOC`- L'emplacement par défaut pour la sortie DUMP
  - `M2DFUTILS_SCRIPTPATH`- L'emplacement du `compare.sh` script documenté dans Asset Locations
  - `M2DFUTILS_VERBOSE`- [VERBEUX ou NORMAL]. Cela contrôle le niveau de détail de la `SYSPRINT` sortie
- Vérifiez que le chemin du module de charge est ajouté au `JES\Configuration\JES Program Path` paramètre
- Vérifiez que les scripts du répertoire des utilitaires disposent des autorisations d'exécution. Vous pouvez ajouter une autorisation d'exécution à l'aide de la `chmod + x <script name>` commande, dans l'environnement Linux

Syntaxe générale

DUMP

Permet de copier des fichiers de l'emplacement catalogué actuel vers un emplacement de sauvegarde. Cet emplacement doit actuellement être un système de fichiers.

Processus

DUMP effectuera les opérations suivantes :

1. Créez le répertoire de localisation cible.
2. Cataloguez le répertoire de localisation cible en tant que membre du PDS.
3. Déterminez les fichiers à inclure en traitant le paramètre INCLUDE.
4. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
5. Déterminez si les fichiers en cours de vidage doivent être SUPPRIMÉS.
6. Mettez en file d'attente les fichiers à traiter.
7. Copiez les fichiers.
8. Exportez les informations DCB cataloguées des fichiers copiés vers un fichier secondaire situé à l'emplacement cible afin de faciliter les futures opérations de restauration.

Syntaxe

```
DUMP
TARGET ( TARGET LOCATION  )    -
INCLUDE ( DSN. )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Paramètres requis

Les paramètres requis pour DUMP sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `TARGET`- Emplacement cible. Il peut s'agir de :
  - Chemin complet de l'emplacement du dépotoir

  - Nom du sous-répertoire créé à l'emplacement défini dans la variable M2DFUTILS_BASE_LOC
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe
- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Les fichiers traités seront conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin
- SUPPRIMER - Si aucune erreur ENQ ne se produit, le fichier est supprimé et n'est pas catalogué

DELETE

Permet de supprimer et de décataloguer des fichiers en masse. Les fichiers ne sont pas sauvegardés.

Processus

DELETE effectuera les opérations suivantes :

1. Déterminez les fichiers à inclure en traitant le paramètre INCLUDE.
2. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
3. Mettez en file d'attente les fichiers à traiter. Régler la disposition sur OLD, DELETE, KEEP.

Syntaxe

```
DELETE
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Paramètres requis

Les paramètres requis pour DELETE sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe
- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Les fichiers traités seront conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin

RESTORE

Permet de restaurer des fichiers précédemment sauvegardés à l'aide de DUMP. Les fichiers sont restaurés à leur emplacement catalogué d'origine, sauf si RENAME est utilisé pour modifier le DSNAME restauré.

Processus

RESTORE effectuera les opérations suivantes :

1. Validez le répertoire de localisation source.
2. Déterminez les fichiers à inclure en traitant le fichier d'exportation du catalogue.
3. Désélectionnez les fichiers inclus en traitant le paramètre EXCLUDE.
4. Mettez en file d'attente les fichiers à traiter.
5. Cataloguez les fichiers qui ne sont pas catalogués en fonction de leurs informations d'exportation.
6. Si un fichier est déjà catalogué et que les informations du catalogue d'exportation sont les mêmes, RESTORE remplacera le jeu de données catalogué si l'option REPLACE est définie.

Syntaxe

```
RESTORE
SOURCE ( TARGET LOCATION )
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
```

```
[ REPLACE]
```

Paramètres requis

Les paramètres requis pour RESTORE sont les suivants :

- `SYSPRINT DD NAME`- Pour contenir des informations de journalisation supplémentaires
- `SOURCE`- Emplacement de la source. Il peut s'agir de :
  - Chemin complet de l'emplacement du dépotoir
  - Nom du sous-répertoire créé à l'emplacement défini dans la variable M2DFUTILS_BASE_LOC
- `INCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe
- `EXCLUDE`- Soit une chaîne de recherche DSNAME unique nommée, soit une chaîne de recherche DSN valide pour le mainframe

Paramètres facultatifs

- ANNULER - Annulez en cas d'erreur. Fichiers traités conservés
- (Par défaut) IGNORER - Ignore toute erreur et tout processus jusqu'à la fin
- REMPLACER - Si le fichier en cours de restauration est déjà catalogué et que les notices du catalogue sont les mêmes, remplacez le fichier catalogué

Exemples de JCL

tâche DUMP

Cette tâche créera un sous-répertoire appelé`TESTDUMP`. Il s'agit de l'emplacement de sauvegarde par défaut spécifié par la variable M2DFUTILS_BASE_LOC. Il créera une bibliothèque PDS pour cette sauvegarde appelée`M2DFUTILS.TESTDUMP`. Les données du catalogue exportées sont stockées dans un fichier séquentiel de lignes situé dans le répertoire de sauvegarde appelé`CATDUMP.DAT`. Tous les fichiers sélectionnés seront copiés dans ce répertoire de sauvegarde.

```
//M2DFDMP JOB 'M2DFDMP',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDUMP.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
```

```
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSIN    DD *
DUMP TARGET(TESTDUMP)               -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
/*
//
```

SUPPRIMER une tâche

Cette tâche supprimera tous les fichiers du catalogue qui correspondent au paramètre INCLUDE.

```
/M2DFDEL JOB 'M2DFDEL',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDEL.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE                            -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
/*
 //
```

tâche RESTORE

Cette tâche restaurera les fichiers correspondant au paramètre INCLUDE à partir de l'emplacement de TESTDUMP sauvegarde. Les fichiers catalogués seront remplacés si le fichier catalogué est le même que celui de l'export CATDUMP et si l'option REPLACE est spécifiée.

```
//M2DFREST JOB 'M2DFREST',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
////SYSPRINT DD DSN=TESTREST.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
RESTORE SOURCE(TESTDUMP)            -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 IGNORE
 REPLACE
```

```
/*
//
```

Utilitaire M2RUNCMD Batch

Vous pouvez utiliser M2RUNCMD, un utilitaire de traitement par lots, pour exécuter des commandes, des scripts et des appels système Micro Focus directement depuis JCL au lieu de les exécuter depuis un terminal ou une invite de commande. Le résultat des commandes est enregistré dans le journal des spoules du traitement par lots.

Rubriques

- Plateformes prises en charge
- Configurer M2RUNCMD pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream
- Exemples de JCL

Plateformes prises en charge

Vous pouvez utiliser M2RUNCMD sur les plateformes suivantes :

- Micro Focus Runtime (sur Amazon EC2)
- Toutes les variantes des produits Micro Focus Enterprise Developer (ED) et Micro Focus Enterprise Server (ES).

Configurer M2RUNCMD pour l'exécution de la modernisation AWS du mainframe sur Amazon EC2 (y compris 2.0) AppStream

Si vos applications migrées s'exécutent sur le moteur d'exécution AWS Mainframe Modernization sur Amazon EC2, configurez M2RUNCMD comme suit.

- Modifiez le chemin du programme Micro Focus JES pour inclure l'emplacement binaire des utilitaires de traitement par lots. Si vous devez spécifier plusieurs chemins, utilisez deux points (:) pour séparer les chemins sous Linux des points-virgules (;) sous Windows.
  - Linux : `/opt/aws/m2/microfocus/utilities/64bit`
  - Windows (32 bits) : `C:\AWS\M2\MicroFocus\Utilities\32bit`
  - Windows (64 bits) : `C:\AWS\M2\MicroFocus\Utilities\64bit`

Exemples de JCL

Pour tester l'installation, vous pouvez utiliser l'un des exemples de JCL suivants.

Exécutez SCRL1.jcl

Cet exemple de JCL crée un script et l'exécute. La première étape consiste à créer un script appelé `/tmp/TEST_SCRIPT.sh` et dont le contenu provient de données `SYSUT1` in-stream. La deuxième étape définit l'autorisation d'exécution et exécute le script créé lors de la première étape. Vous pouvez également choisir de n'exécuter que la deuxième étape pour exécuter Micro Focus et les commandes système déjà existantes.

```
//RUNSCRL1 JOB 'RUN SCRIPT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*  CREATE SCRIPT (LINUX)
//*--------------------------------------------------------------------*
//*
//STEP0010 EXEC PGM=IEBGENER
//*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//SYSUT1   DD *
#!/bin/bash

set -x

## ECHO PATH ENVIRONMNET VARIABLE
echo $PATH

## CLOSE/DISABLE VSAM FILE
casfile -r$ES_SERVER -oc  -ed -dACCTFIL

## OPEN/ENABLE VSAM FILE
casfile -r$ES_SERVER -ooi -ee -dACCTFIL

exit $?
/*
//SYSUT2   DD DSN=&&TEMP,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=LSEQ,LRECL=300,DSORG=PS,BLKSIZE=0)
```

```
//*MFE: %PCDSN='/tmp/TEST_SCRIPT.sh'
//*
//*--------------------------------------------------------------------*
//*   RUN SCRIPT (LINUX)                                               *
//*--------------------------------------------------------------------*
//*
//STEP0020 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*RUN SCRIPT
 sh /tmp/TEST_SCRIPT.sh
/*
//
```

SYSOUT

Le résultat de la commande ou du script exécuté est écrit dans le SYS0UT journal. Pour chaque commande exécutée, il affiche la commande, le code de sortie et le code de retour.

```
************ CMD Start ************

CMD_STR: sh /tmp/TEST_SCRIPT.sh

CMD_OUT:


+ echo /opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
/opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
+ casfile -rMYDEV -oc -ed -dACCTFIL

-Return Code:   0

Highest return code:    0

+ casfile -rMYDEV -ooi -ee -dACCTFIL

-Return Code:   8

Highest return code:    8
```

```
+ exit 8


CMD_RC=8


************  CMD End  ************
```

Exécutez CMDL1.jcl

Cet exemple de JCL utilise RUNCMD pour exécuter plusieurs commandes.

```
//RUNCMDL1 JOB 'RUN CMD',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*   RUN SYSTEM COMMANDS                                              *
//*--------------------------------------------------------------------*
//*
//STEP0001 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*LIST DIRECTORY
 ls
*ECHO PATH ENVIRONMNET VARIABLE
 echo $PATH
/*
//
```

AWS Modernisation du mainframe et réplication des données avec Precisely

AWS Mainframe Modernization propose une variété d'Amazon Machine Images (AMI). Ces AMI facilitent le provisionnement rapide des instances Amazon EC2, créant ainsi un environnement personnalisé pour la réplication des données depuis les systèmes AWS mainframe vers l'utilisation de Precisely. Ce guide décrit les étapes nécessaires pour accéder à ces AMI et les utiliser.

Prérequis

- Assurez-vous de disposer d'un accès administrateur à un AWS compte sur lequel vous pouvez créer des instances Amazon EC2.
- Vérifiez que le service de modernisation du AWS mainframe est disponible dans la région où vous prévoyez de créer les instances Amazon EC2. Consultez la liste des services AWS disponibles par région.
- Identifiez l'Amazon Virtual Private Cloud (Amazon VPC) dans lequel les instances Amazon EC2 seront créées.
- Lorsque vous créez des instances Amazon EC2 dans un Amazon VPC, assurez-vous que la table de routage associée possède une passerelle Internet ou une passerelle NAT.

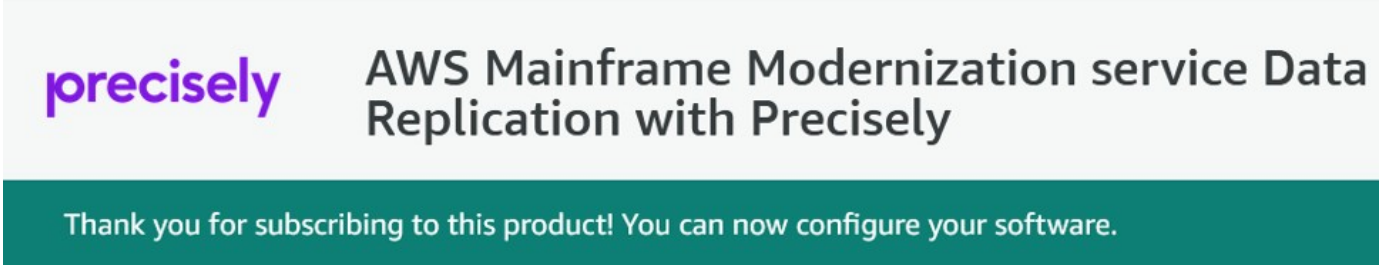
> **Note**
>
> Une réplication de données réussie nécessite que l'instance AWS EC2 dispose d'un accès de communication avec AWS Marketplace. En cas de problème de connectivité avec AWS Marketplace, le processus de réplication échouera.

Abonnez-vous à l'Amazon Machine Image

Lorsque vous vous abonnez à un produit AWS Marketplace, vous pouvez lancer une instance depuis l'AMI du produit.

1. Connectez-vous à la AWS Marketplace console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/marketplace.

2. Sélectionnez Manage subscriptions (Gérer les abonnements).
3. Copiez et collez le lien suivant dans la barre d'adresse du navigateur : https://aws.amazon.com/marketplace/pp/prodview-en3xrbgzbs3dk
4. Choisissez Continue to Subscribe (Continuer pour s'abonner).
5. Si les termes et conditions sont acceptables, choisissez Accepter les termes. Le traitement de l'abonnement peut prendre quelques minutes.
6. Attendez que le message de remerciement apparaisse, comme indiqué ci-dessous. Ce message confirme que vous êtes bien abonné au produit.



7. Dans le volet de navigation de gauche, choisissez Gérer les abonnements. Cette vue affiche tous les abonnements auxquels vous êtes abonné.

Lancez la réplication des données de modernisation du AWS mainframe avec Precisely

1. Ouvrez la AWS Marketplace console à l'adresse https://console.aws.amazon.com/marketplace.
2. Dans le volet de navigation de gauche, choisissez Gérer les abonnements.
3. Recherchez l'AMI que vous souhaitez lancer, puis choisissez Launch new instance.
4. Sous Région, sélectionnez la région autorisée.
5. Choisissez Continuer pour lancer via EC2. Cette action vous amène à la console Amazon EC2.
6. Entrez un nom pour le serveur.
7. Sélectionnez un type d'instance qui correspond aux exigences de performance et de coût de votre projet. Le point de départ suggéré pour la taille de l'instance est `c5.2xLarge`.
8. Choisissez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Pour plus d'informations sur les paires de clés, consultez les paires de clés Amazon EC2 et les instances Linux dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.
9. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.

10. Choisissez un groupe de sécurité existant ou créez-en un nouveau. En plus d'autoriser l'accès SSH (par défaut sur le port 22), pour la réplication des données avec une instance EC2 du serveur Precisely, il est courant d'autoriser le trafic TCP vers son port par défaut 2626.
11. Configurez le stockage pour l'instance Amazon EC2.
12. Consultez le résumé et choisissez Launch instance. Pour que le lancement réussisse, le type d'instance doit être valide. Si le lancement échoue, choisissez Modifier la configuration de l'instance et choisissez un autre type d'instance.
13. Après avoir vu le message de réussite, choisissez Connect to instance.
14. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
15. Dans le volet de navigation de gauche, sous le menu Instances, sélectionnez Instances.
16. Dans le volet principal, vérifiez l'état de votre instance.

Créer une politique IAM

Pour faire fonctionner correctement les instances EC2 de modernisation du AWS mainframe déployées via notre AWS Marketplace liste, vous devez configurer un rôle et une politique IAM. Cette configuration IAM spécialement adaptée n'est pas facultative ; elle autorise vos instances Amazon EC2 à interagir avec le service. AWS Marketplace Le rôle et la politique IAM permettent à AWS Mainframe Modernisation d'enregistrer avec précision les données d'utilisation, ce qui est essentiel pour une facturation précise. L'échec de la mise en œuvre de cette configuration peut entraîner l'échec des tentatives de réplication des données et des interruptions du fonctionnement.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).
3. Si c'est la première fois que vous choisissez Politiques, la page Bienvenue dans Managed Policies apparaît. Sélectionnez Mise en route.
4. En haut de la page, sélectionnez Créer une politique.
5. Dans la section Éditeur de politiques, choisissez l'option JSON.
6. Entrez la politique JSON suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": ["aws-marketplace:MeterUsage"],
```

```
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Créer un rôle IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, sélectionnez Rôles, puis Créer un rôle.
3. Sous la section Type d'entité de confiance, sélectionnez Service AWS .
4. Dans la section Cas d'utilisation, sous Service ou cas d'utilisation, sélectionnez Amazon EC2.
5. Choisissez Suivant.
6. Dans la liste des politiques, sélectionnez Client géré dans le menu déroulant Filtrer par type et entrez le nom de la politique que vous avez créée. Cochez la case à côté du nom de la politique.
7. Choisissez Suivant.
8. Entrez un nom et, éventuellement, une description pour le rôle.
9. Passez en revue la politique de confiance et les autorisations, puis choisissez Create role.

Attachez le rôle IAM à l'instance Amazon EC2

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans le panneau de navigation, sélectionnez Instances.
3. Sélectionnez votre instance Amazon EC2.
4. Dans le menu Actions, choisissez Sécurité, puis Modifier le rôle IAM.
5. Sélectionnez le rôle à associer à votre instance, puis choisissez Mettre à jour le rôle IAM.

Intégration de Charon

Présentation de Charon-SSP

En 1987, Sun Microsystems a lancé le processeur SPARC V7, un processeur RISC 32 bits. Le SPARC V8 a suivi en 1990, une révision du SPARC V7 original, avec notamment l'inclusion d'instructions matérielles de division et de multiplication. Les processeurs SPARC V8 ont constitué la base d'un certain nombre de serveurs et de stations de travail tels que les SPARCstation 5, 10 et 20. En 1993, le SPARC V8 a été suivi par le processeur SPARC V9 64 bits. Cela est également devenu la base d'un certain nombre de serveurs et de stations de travail, tels que les Enterprise 250 et 450.

En raison de l'obsolescence du matériel et du manque de pièces de rechange ou remises à neuf, les logiciels et systèmes développés pour ces anciens postes de travail et serveurs basés sur SPARC sont devenus plus difficiles à entretenir. Pour répondre au besoin permanent de certains systèmes end-of-life basés sur SPARC, Stromasys S.A. a développé la gamme Charon-SSP d'émulateurs SPARC. Les produits suivants remplacent des machines virtuelles basées sur des logiciels pour les systèmes SPARC matériels natifs spécifiés. Voici un aperçu général des familles de matériel émulé.

Charon-SSP/4M émule le matériel SPARC suivant :

- Famille Sun-4m (représentée par la Sun SparcStation 20) : à l'origine, une variante multiprocesseur Sun-4, basée sur le bus du module processeur MBus introduit dans la série SparcServer 600MP. Plus tard, l'architecture Sun-4m a également inclus des systèmes uniprocesseurs non MBUS tels que le SPAR Cstation 5, utilisant des processeurs d'architecture SPARC V8. Supporté à partir de SunOS 4.1.2 et de Solaris 2.1 à Solaris 9. Le support du SparcServer 600MP a été abandonné après Solaris 2.5.1.

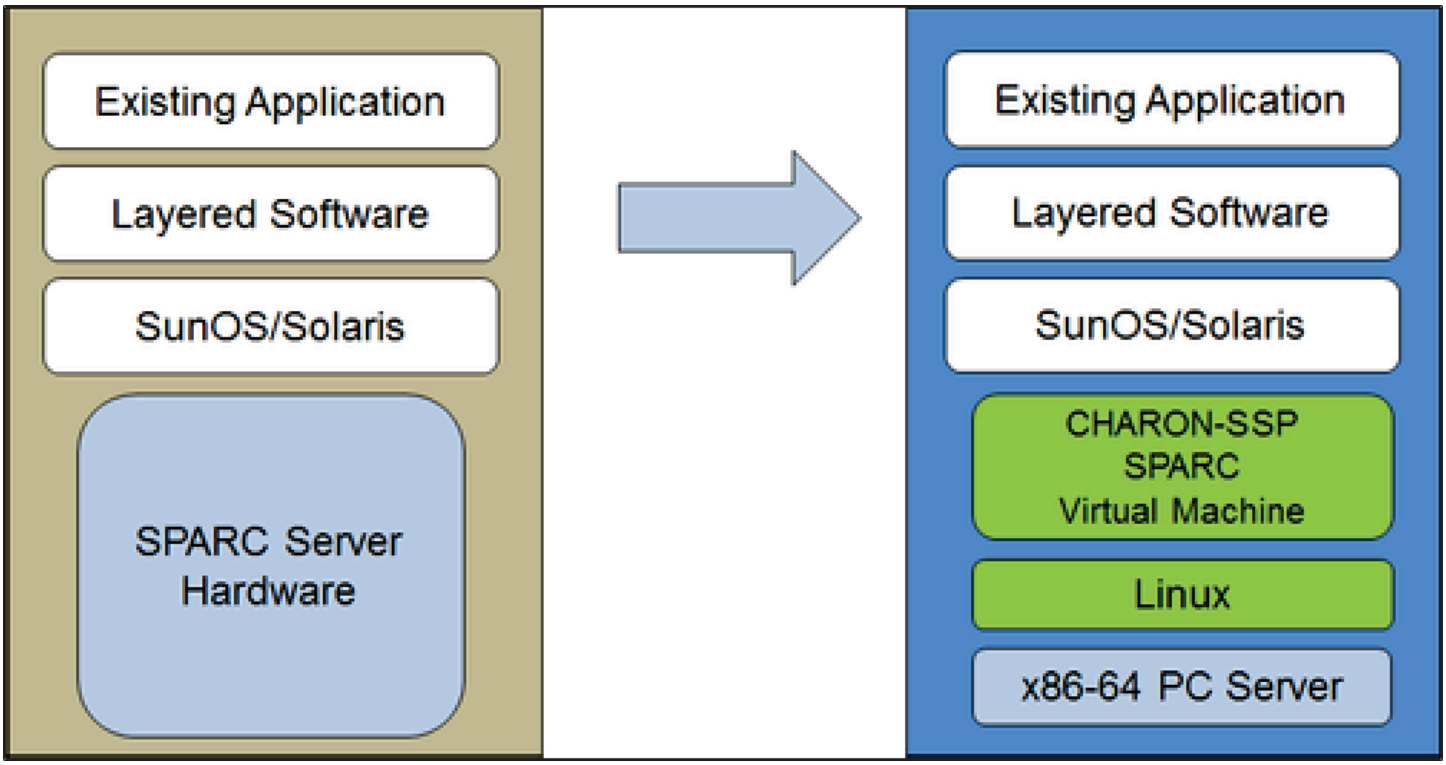
Charon-SSP/4U (+) émule le matériel SPARC suivant :

- Famille Sun-4u (représentée par le Sun Enterprise 450) : (U pour UltraSPARC) : cette variante a introduit l'architecture du processeur SPARC V9 64 bits et l'interconnexion des processeurs UPA utilisés pour la première fois dans la série Sun Ultra. Pris en charge par les versions 32 bits de Solaris à partir de la version 2.5.1. La première version 64 bits de Solaris pour Sun-4u était Solaris 7. Le support d'UltraSPARC I a été abandonné après Solaris 9. Solaris 10 prend en charge les implémentations Sun-4u d'UltraSP ARC II à UltraSPARC IV.

Charon-SSP/4V (+) émule le matériel SPARC suivant :

- Famille Sun-4v (représentée par les SPARC T2 et T4) : cette variante a ajouté la virtualisation des processeurs d'hyperviseur au Sun-4u, introduite dans le processeur multicœur Ultra SPARC T1. Le matériel sélectionné était pris en charge par Solaris version 10 à partir de la version 3/05 HW2 (la plupart des modèles, y compris le matériel émulé par Charon-SSP, nécessitent des versions plus récentes de Solaris 10). Plusieurs versions de Solaris 11 sont également prises en charge.

L'image suivante montre le concept de base de la migration du matériel physique vers un émulateur.



Les machines virtuelles Charon-SSP permettent aux utilisateurs d'ordinateurs basés sur Sun et Oracle SPARC de remplacer leur matériel natif d'une manière qui ne nécessite que peu ou pas de modification de la configuration système d'origine. Cela signifie que vous pouvez continuer à exécuter vos applications et vos données sans avoir à changer de plateforme ou à effectuer un portage sur une autre plateforme. Le logiciel Charon-SSP fonctionne sur des systèmes Intel 64 bits standard, garantissant ainsi la protection continue de votre investissement.

Charon-SSP/4U+ prend en charge les mêmes plateformes virtuelles SPARC que Charon-SSP/4U, et Charon-SSP/4v+ est identique à Charon-SSP/4V. Cependant, les versions 4U+ et 4V+ tirent parti de la technologie de virtualisation assistée par matériel VTx/EPT d'Intel et AMD-V/NPT d'AMD dans les processeurs modernes afin d'offrir de meilleures performances des processeurs virtuels. Charon-

SSP/4U+ et Charon-SSP/4v+ nécessitent des processeurs compatibles VT-x/EPT ou AMD-V/NPT et doivent être installés sur un système hôte dédié. L'exécution de ces variantes de produit sur une machine virtuelle (par exemple, sur VMware) n'est pas prise en charge.

> **Note**
>
> Si vous envisagez d'exécuter Charon-SSP/4U+ ou 4V+ dans un environnement cloud, contactez Stromasys ou un VAR de Stromasys pour discuter de vos besoins.

Systèmes d'exploitation clients pris en charge

Les machines virtuelles Charon-SSP/4M prennent en charge les versions de système d'exploitation client suivantes :

- SunOS 4.1.3 - 4.1.4
- Solaris 2.3 à Solaris 9

Les machines virtuelles Charon-SSP/4U (+) prennent en charge les versions suivantes du système d'exploitation client :

- Solaris 2.5.1 à Solaris 10

Les machines virtuelles Charon-SSP/4V (+) prennent en charge les versions suivantes du système d'exploitation client :

- Solaris 10 (à partir de la mise à jour 4, 08/07) et Solaris 11.1 vers Solaris 11.4

Pour Charon-SSP/4V (+), notez ce qui suit :

- Pour le SPARC T4 émulé, les versions de Solaris 10 prises en charge sont les suivantes : Oracle Solaris 10 1/13, Oracle Solaris 10 8/11 et Solaris 10 9/10, ou Solaris 10 10/09 avec le jeu de correctifs Oracle Solaris 10 8/11.
- Le modèle SPARC T4 émulé est une condition préalable à l'exécution de Solaris 11.4 dans l'émulateur.
- Les zones du noyau Solaris ne sont pas prises en charge.

Conditions préalables à l'instance cloud Charon-SSP

En sélectionnant un type ou une forme d'instance, vous sélectionnez le matériel virtuel qui sera utilisé pour l'instance hôte Charon-SSP dans le cloud. Par conséquent, la sélection d'un type ou d'une forme d'instance détermine les caractéristiques matérielles du matériel hôte virtuel Charon-SSP (par exemple, le nombre de cœurs de processeur et la quantité de mémoire dont disposera votre système hôte virtuel Charon).

> **Note**
>
> Si vous utilisez une image du marché Charon-SSP pour lancer votre instance, toutes les exigences du système d'exploitation hôte Linux sont satisfaites.

La configuration matérielle minimale requise est décrite ci-dessous.

Points importants concernant les directives relatives aux tailles :

- Les directives de dimensionnement ci-dessous, en particulier en ce qui concerne le nombre de cœurs du processeur hôte et la mémoire hôte, indiquent les exigences minimales. Chaque situation de déploiement doit être revue et le dimensionnement réel de l'hôte doit être adapté si nécessaire. Par exemple, le nombre de cœurs de processeur disponibles pour les E/S doit être augmenté si les applications clientes génèrent une charge d'E/S élevée. En outre, un système doté de nombreux processeurs émulés est généralement capable de créer une charge d'E/S plus élevée et il peut donc être nécessaire d'augmenter le nombre de cœurs de processeur disponibles pour les E/S. Dans un environnement d'hyperthreading, pour de meilleures performances, le nombre de cœurs de processeur (c'est-à-dire des processeurs réels/physiques) doit être suffisant pour répondre aux exigences de processeur des émulateurs actifs, évitant ainsi que des threads à charge de travail élevée partagent un cœur de processeur physique.
- L'allocation de cœurs de processeur pour les processeurs émulés et de cœurs de processeur pour le traitement des E/S est déterminée par la configuration. Voir Configuration du processeur dans le guide général de l'utilisateur de Charon-SSP pour plus d'informations à ce sujet et sur l'allocation par défaut des cœurs de processeur pour le traitement des E/S.

⚠ Informations générales importantes

- Pour faciliter le transfert rapide des données d'émulateur d'une instance cloud à une autre, il est vivement recommandé de stocker toutes les données pertinentes de l'émulateur sur un volume de disque distinct qui peut être facilement détaché de l'ancienne instance et attaché à une nouvelle instance.
- Assurez-vous de dimensionner correctement votre instance dès le début (vérifiez les exigences minimales ci-dessous). La licence Charon-SSP pour Charon-SSP AL est créée lors du premier lancement de l'instance. Le fait de changer ultérieurement de taille/type d'instance et donc de modifier le nombre de cœurs de processeur invalidera la licence et empêchera ainsi le démarrage des instances Charon (une nouvelle instance est requise). Si vous prévoyez d'utiliser l'instance Charon-SSP AL en mode AutoVE, veillez à inclure les informations du serveur AutoVE avant le premier lancement, sinon les serveurs de licences publics seront utilisés. La licence pour Charon-SSP VE est créée sur la base de l'empreinte digitale prise sur le serveur de licences. Si le serveur de licences est exécuté directement sur l'hôte de l'émulateur et que celui-ci nécessite ultérieurement, par exemple, une modification du nombre de cœurs de processeur, la licence sera invalidée (nouvelle licence et éventuellement nouvelle instance requises).

Conditions préalables à l'instance

Exigences générales en matière de processeur : Charon-SSP prend en charge les processeurs d'architecture x86-64 modernes basés sur des instances Amazon EC2.

Exigences minimales pour Charon-SSP :

- Nombre minimal de cœurs de processeur du système hôte :
  - Au moins un cœur de processeur pour le système d'exploitation hôte, plus :
  - Pour chaque système SPARC émulé :
    - Un cœur de processeur pour chaque processeur émulé de l'instance, plus :
    - Au moins un cœur de processeur supplémentaire pour le traitement des E/S (au moins deux, si l'optimisation JIT du serveur est utilisée). Consultez la section Configuration du processeur mentionnée ci-dessus pour les options de configuration. Par défaut, Charon affecte 1/3 (min. 1 ; arrondi au chiffre inférieur) du nombre de processeurs visibles par l'hôte Charon au traitement des E/S.

- Mémoire minimale requise :
  - 4 Go ou plus de RAM pour le système d'exploitation hôte Linux. Les exigences réelles peuvent être plus élevées et dépendront des exigences des services non émulateurs exécutés sur l'hôte Linux. La recommandation précédente d'au moins 2 Go de RAM pour l'hôte Linux sera toujours valable pour de nombreux systèmes, mais les exigences croissantes du système d'exploitation et des applications Linux ont conduit à la mise à jour de la recommandation pour les nouvelles installations. De plus :
  - Pour chaque système SPARC émulé :
    - La mémoire configurée de l'instance émulée, plus :
    - 2 Go de RAM (6 Go de RAM si le serveur JIT est utilisé) pour permettre l'optimisation du DIT, les exigences de l'émulateur, les tampons d'exécution, le SMP et l'émulation graphique.
- Si l'hyperthreading est activé sur les processeurs x86-64 modernes, deux threads peuvent s'exécuter sur un cœur de processeur physique fournissant deux processeurs logiques au système d'exploitation hôte. Si possible, désactivez l'hyperthreading sur l'hôte Charon-SSP. Cependant, cela n'est souvent pas possible dans VMware et dans les environnements cloud, ou il n'est pas clair si l'hyperthreading est utilisé ou non. L'option d'hyperthreading Charon-SSP permet à Charon-SSP de s'adapter à de tels environnements. Consultez la section Configuration du processeur dans le guide d'utilisation général de Charon-SSP mentionné ci-dessus pour obtenir des informations de configuration détaillées. Remarque : pour de meilleures performances, les threads Charon-SSP ne doivent pas partager un cœur de processeur physique ; suffisamment de cœurs physiques doivent être disponibles sur le système hôte pour répondre aux exigences des émulateurs configurés.
- Une ou plusieurs interfaces réseau, selon les besoins du client.
- Charon-SSP/4U+ et Charon-SSP/4v+ doivent fonctionner sur du matériel physique compatible Intel VT-x/EPT ou AMD-V/NPT (instances baremetal) et ne peuvent donc pas fonctionner dans tous les environnements cloud. Consultez la documentation de votre fournisseur de cloud pour connaître la disponibilité de ce type de matériel. Notez également les points suivants :
  - Charon-SSP/4U+ et Charon-SSP/4v+ ne sont disponibles que si vous utilisez un noyau Linux pris en charge par Stromasys.
  - Si vous avez besoin de ce type de matériel SPARC émulé, contactez Stromasys ou votre VAR Stromasys pour discuter de vos besoins en détail.

Création et configuration d'une instance AWS cloud pour Charon (nouvelle interface graphique)

Cette section reflète la situation AWS Management Console au printemps 2022. Si vous utilisez toujours l'ancienne console, reportez-vous à l'annexe du guide de démarrage de Charon-SSP AWS .

Prérequis généraux

Cette description montre la configuration de base d'une instance Linux dans AWS. Il ne répertorie pas les prérequis spécifiques. Toutefois, en fonction de votre cas d'utilisation, tenez compte des conditions préalables suivantes :

- Compte Amazon et AWS Marketplace abonnements
  - Pour configurer une instance Linux dans AWS, vous avez besoin d'un AWS compte avec accès administrateur.
  - Identifiez la AWS région dans laquelle vous prévoyez de lancer votre instance. Assurez-vous que AWS les services que vous prévoyez d'utiliser sont disponibles dans cette région. Voir AWS Services par région.
  - Identifiez le VPC et le sous-réseau dans lesquels vous prévoyez de lancer votre instance.
  - Si votre instance nécessite un accès à Internet, assurez-vous que la table de routage associée à votre VPC dispose d'une passerelle Internet. Si votre instance nécessite un accès VPN à votre réseau local, assurez-vous qu'une passerelle VPN est disponible. La configuration exacte de votre VPC et de ses sous-réseaux dépend de la conception de votre réseau et des exigences de l'application.
  - Pour vous abonner à un AWS Marketplace service spécifique, sélectionnez AWS Marketplace Subscriptions dans le, AWS Management Console puis sélectionnez Gérer les abonnements.
  - Recherchez le service que vous comptez utiliser et abonnez-vous à celui-ci. Après un abonnement réussi, vous trouverez l'abonnement dans la section Gérer les abonnements. De là, vous pouvez directement lancer une nouvelle instance.
- Les prérequis matériels et logiciels de l'instance seront différents en fonction de l'utilisation prévue de l'instance :
  - Option 1 : l'instance doit être utilisée comme système hôte de l'émulateur Charon :
    - Reportez-vous aux sections relatives aux prérequis matériels et logiciels du guide de l'utilisateur et/ou du guide de démarrage de votre produit Charon pour déterminer les prérequis matériels et logiciels exacts qui doivent être remplis par l'instance Linux. L'image que vous

utilisez pour lancer votre instance et le type d'instance que vous avez choisi déterminent le logiciel et le matériel de votre instance cloud.

- Une licence de produit Charon est requise pour exécuter les anciens systèmes émulés. Reportez-vous aux informations de licence figurant dans la documentation de votre produit Charon, ou contactez votre représentant Stromasys ou le VAR de Stromasys pour plus d'informations.

- Option 2 : l'instance doit être utilisée comme serveur de licences VE dédié :
  - Consultez le guide du serveur de licences VE pour connaître les prérequis détaillés.

- Certains systèmes d'exploitation existants qui peuvent fonctionner dans les systèmes émulés fournis par les produits d'émulation Charon nécessitent une licence du fournisseur d'origine du système d'exploitation. L'utilisateur est responsable de toutes les obligations de licence liées à l'ancien système d'exploitation et doit fournir les licences appropriées.

Utilisation du AWS Management Console pour lancer une nouvelle instance

Pour créer une nouvelle instance

1. Connectez-vous à la console Amazon EC2 AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Launch instance (Lancer une instance).
3. Entrez un nom pour l'instance.
4. Sélectionnez une AMI. Une AMI est une image préemballée utilisée pour lancer des instances cloud. Il inclut le système d'exploitation et le logiciel d'application applicable. Le choix de l'AMI dépend de la manière dont vous prévoyez d'utiliser l'instance :

   - Si l'instance doit être utilisée comme système hôte de l'émulateur Charon, plusieurs choix d'AMI sont possibles :
     - Installation du système hôte Charon à partir d'une image de marché Charon prépackagée : ils contiennent le système d'exploitation sous-jacent et le logiciel Charon préinstallé.
       - Vérifiez auprès de votre représentant Stromasys quelles options sont actuellement disponibles sur le marché de vos fournisseurs de cloud.
       - Selon le fournisseur de cloud et les plans de lancement des produits Stromasys, il peut y avoir deux variantes :
         - Licences automatiques (AL) à utiliser avec un serveur de licences public géré par Stromasys ou avec un serveur de licences AutoVE privé géré par le client

    - Environnement virtuel (VE) à utiliser avec un serveur de licences VE privé géré par le client
  - Installation du système hôte Charon à l'aide d'une installation d'émulateur Charon classique avec les packages RPM d'installation de l'émulateur Charon pour Linux :
    - Choisissez une AMI Linux d'une distribution prise en charge par le produit et la version de Charon que vous avez sélectionnés. Consultez le guide d'utilisation de votre produit sur le site de documentation de Stromasys.
  - Si l'instance doit être utilisée comme serveur de licences VE dédié, consultez le guide du serveur de licences VE dans la documentation des licences pour connaître les exigences de l'instance Linux.

  Après avoir déterminé quelle AMI est requise, sélectionnez une AMI de produit Linux ou Charon correspondante. Si vous ne trouvez pas l'AMI dont vous avez besoin, choisissez Parcourir d'autres AMI. Choisissez l'AMI Linux qui correspond à la manière dont vous prévoyez d'utiliser l'instance. Il peut s'agir de l'un des périphériques suivants :

  - Une image du marché Charon VE préemballée. Le nom de l'AMI inclura la chaîne « ve ».
  - Une image du marché Charon AL préemballée pour Automatic Licensing ou AutoVE.
  - Version Linux prise en charge pour l'installation d'un produit RPM.
  - Version Linux prise en charge pour le serveur de licences VE.
5. Sélectionnez un type d'instance. Amazon EC2 propose des types d'instances avec différentes combinaisons de capacité de processeur, de mémoire, de stockage et de réseau. Sélectionnez un type d'instance qui répond aux exigences du produit Charon que vous souhaitez utiliser. Certaines images du site de vente comportent une sélection limitée de types d'instances.
6. Sélectionnez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Si vous sélectionnez une paire de clés existante, assurez-vous de disposer de la clé privée correspondante. Dans le cas contraire, vous ne pourrez pas vous connecter à votre instance.

   > **Note**
   >
   > Si votre système de gestion le prend en charge, pour RHEL 9.x, Rocky Linux 9.x et Oracle Linux 9.x, utilisez une clé SSH de type ECDSA ou ED25519. Ces types vous permettent de vous connecter à ces systèmes Linux hôtes Charon à l'aide d'un tunnel SSH sans avoir à modifier les paramètres de politique cryptographique par

défaut sur l'hôte Charon pour des paramètres moins sécurisés. Par exemple, cela est important pour le gestionnaire Charon-SSP. Consultez la section Utilisation de politiques cryptographiques à l'échelle du système dans la documentation Red Hat.

7. Dans la section Paramètres réseau, choisissez Modifier. Choisissez les paramètres qui correspondent à votre environnement.

   - Spécifiez un VPC.
   - Spécifiez un sous-réseau existant ou créez-en un nouveau.
   - Activez ou désactivez l'attribution automatique d'une adresse IP publique à l'interface principale. L'attribution automatique n'est possible que si l'instance possède une seule interface réseau.
   - Attribuez un groupe de sécurité personnalisé existant ou nouveau. Le groupe de sécurité doit au moins autoriser SSH à accéder à l'instance. Tous les ports requis par les applications que vous prévoyez d'exécuter sur l'instance doivent également être autorisés. Vous pouvez modifier le groupe de sécurité à tout moment après avoir créé l'instance.

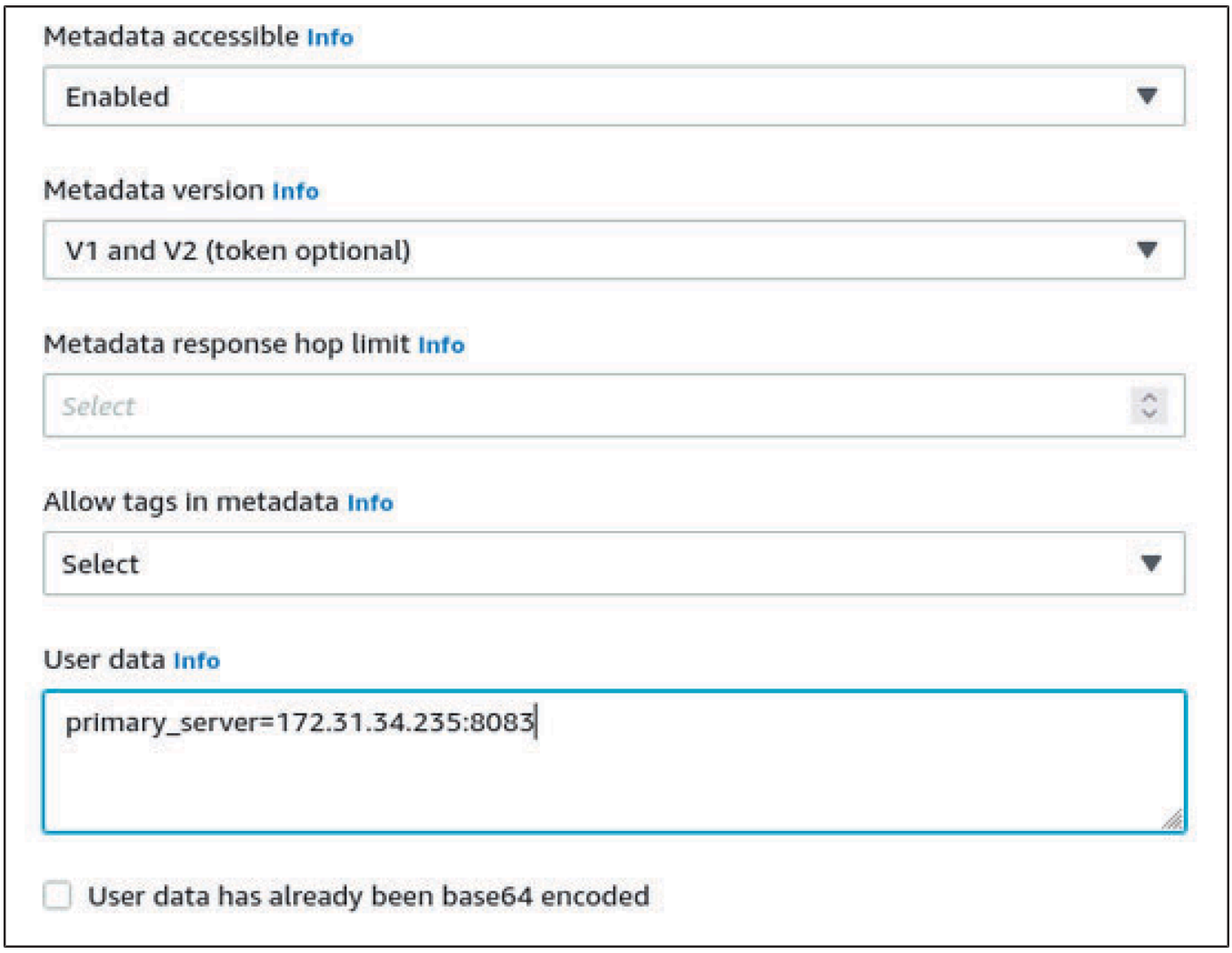
8. Dans la section Stockage, pour le volume racine (le disque système), choisissez une taille adaptée à votre environnement. La taille minimale de disque système recommandée pour le système Linux est de 30 GiB. Pour fournir de l'espace pour les conteneurs de disques virtuels et pour d'autres besoins de stockage, vous pouvez ajouter de l'espace de stockage maintenant ou après le lancement de l'instance. Toutefois, la taille du disque système doit couvrir les exigences du système Linux, y compris les applications et les utilitaires que vous prévoyez d'installer.

   > **Note**
   >
   > Nous vous recommandons de créer des volumes de stockage distincts pour les données de l'application Charon (par exemple, les images de disque). Si nécessaire, vous pouvez ultérieurement migrer ces volumes vers une autre instance.

9. Développez la section Détails avancés, faites défiler l'écran vers le bas et sélectionnez Spécifier les options du processeur. Trois des exemples les plus susceptibles d'être utiles à un environnement d'émulateur Charon sont présentés dans l'image suivante.

10. Pour un système de serveur de licences VE dont la version est antérieure à 1.1.23, vous devez attribuer le rôle IAM requis à l'instance. Ce doit être un rôle qui permet l'`ListUsers`action. Pour attribuer un rôle, dans la section détaillée des détails avancés, sélectionnez un rôle dans le profil d'instance IAM ou choisissez Créer un nouveau profil IAM. Pour plus d'informations, consultez Rôles IAM pour Amazon EC2.
11. Si votre instance est basée sur une AWS Marketplace image Charon AL et que vous prévoyez d'utiliser les serveurs de licences publics gérés par Stromasys, vous devez ajouter les informations correspondantes à la configuration de l'instance avant de lancer l'instance.

    Entrez les informations relatives au serveur de licences AutoVE comme indiqué dans l'image suivante.



Les options de configuration des données utilisateur valides sont les suivantes :

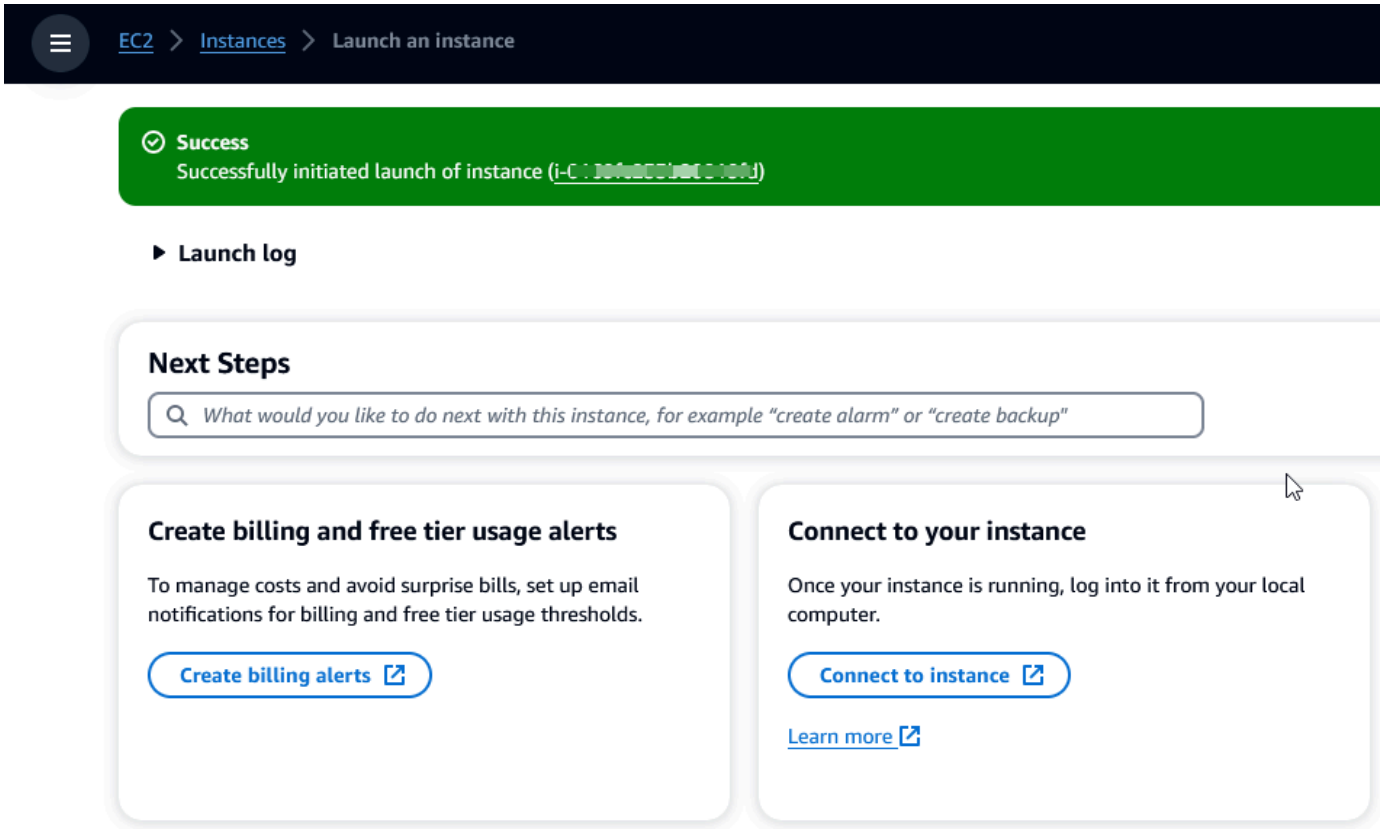
- **`primary_server=`***`<ip-address>`*`[:`*`<port>`*`]`
- **`backup_server=`***`<ip-address>`*`[:`*`<port>`*`]`

Où

- <ip-address>représente l'adresse IP du serveur principal et du serveur de sauvegarde, le cas échéant.
- <port>représente un port TCP autre que celui par défaut utilisé pour communiquer avec le serveur de licences (par défaut : TCP/8083).

> **Note**
>
> Au moins un serveur de licences doit être configuré lors du lancement initial pour activer le mode AutoVE. Dans le cas contraire, l'instance sera liée à l'un des serveurs de licences publics exploités par Stromasys.

12. Dans la section Résumé, choisissez Launch instance. Au bout d'un moment, le message de réussite suivant s'affichera :



13. Dans le coin inférieur droit de l'écran, choisissez Afficher toutes les instances.
14. Pour voir les détails de votre instance, cochez la case située à gauche de la ligne qui représente l'instance dans le tableau Instances. Les détails de votre instance apparaîtront dans la moitié inférieure de l'écran. Pour plus d'informations sur la façon de se connecter à votre instance, consultez Connect dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.

AWS Modernisation du mainframe et replateforme avec NTT DATA

AWS Mainframe Modernization propose une variété d'Amazon Machine Images (AMI). Ces AMI facilitent le provisionnement rapide des instances Amazon EC2, en créant un environnement sur mesure pour le réhébergement et la replateforme des applications mainframe à l'aide de NTT Data. AWS Ce guide décrit les étapes nécessaires pour accéder à ces AMI et les utiliser.

Prérequis

- Assurez-vous de disposer d'un accès administrateur à un AWS compte sur lequel vous pouvez créer des instances Amazon EC2.
- Vérifiez que le service de modernisation du AWS mainframe est disponible dans la région où vous prévoyez de créer les instances Amazon EC2. Consultez la liste des services AWS disponibles par région.
- Identifiez le VPC Amazon sur lequel vous souhaitez créer les instances Amazon EC2.

Abonnez-vous à l'Amazon Machine Image

Lorsque vous vous abonnez à un produit AWS Marketplace, vous pouvez lancer une instance depuis l'AMI du produit.

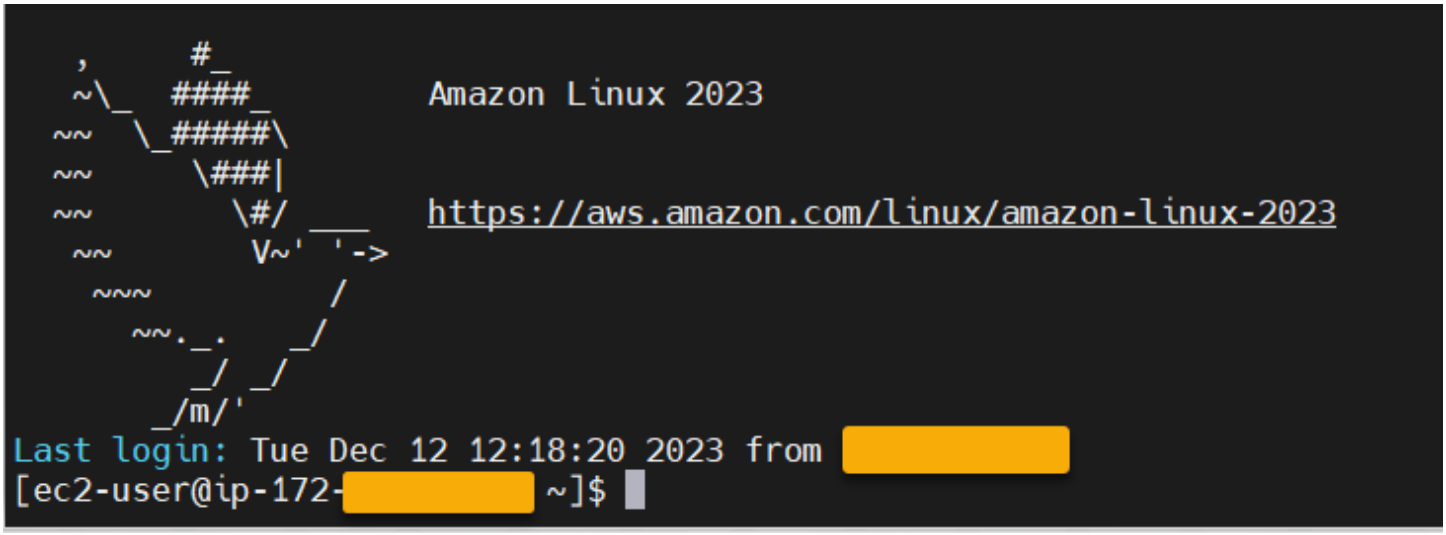
1. Connectez-vous à la AWS Marketplace console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/marketplace.
2. Sélectionnez Manage subscriptions (Gérer les abonnements).
3. Copiez et collez le lien suivant dans la barre d'adresse du navigateur : https://aws.amazon.com/marketplace/pp/prodview-eg227ymldsnx2
4. Choisissez Continue to Subscribe (Continuer pour s'abonner).
5. Si les termes et conditions sont acceptables, choisissez Accepter les termes. Le traitement de l'abonnement peut prendre quelques minutes.
6. Attendez qu'un message de remerciement apparaisse. Ce message confirme que vous êtes bien abonné au produit.
7. Dans le volet de navigation de gauche, choisissez Gérer les abonnements. Cette vue affiche tous vos abonnements.

Lancez la replateforme de modernisation du AWS mainframe avec l'instance NTT DATA

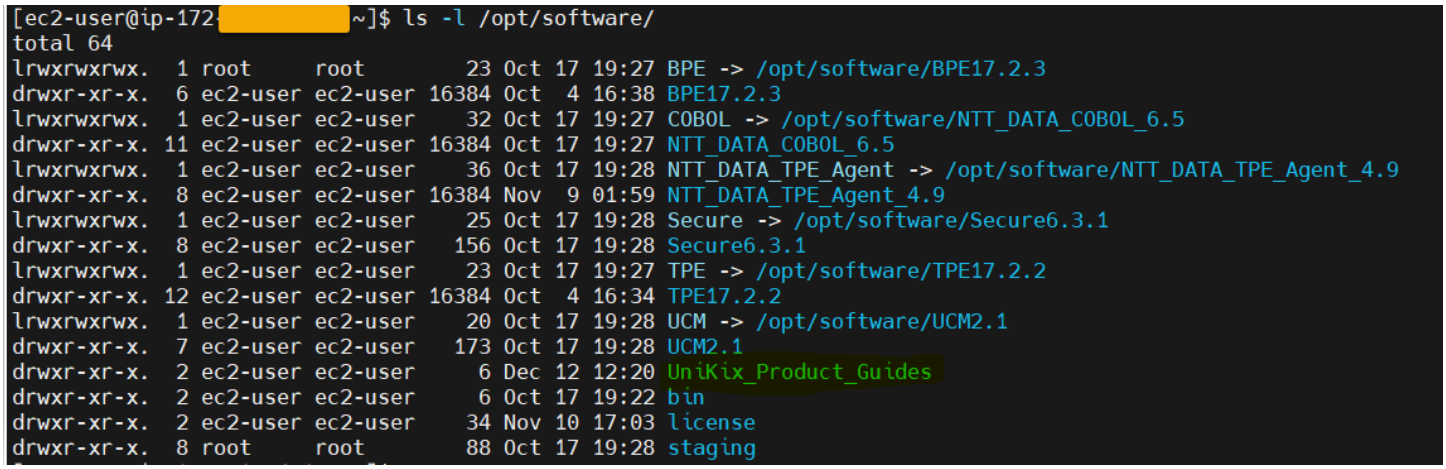
1. Ouvrez la AWS Marketplace console à l'adresse https://console.aws.amazon.com/marketplace.
2. Dans le volet de navigation de gauche, choisissez Gérer les abonnements.
3. Recherchez l'AMI que vous souhaitez lancer, puis choisissez Launch new instance.
4. Sous Région, sélectionnez la région autorisée.
5. Choisissez Continuer pour lancer via EC2. Cette action vous amène à la console Amazon EC2.
6. Entrez un nom pour le serveur.
7. Sélectionnez un type d'instance qui correspond aux exigences de performance et de coût de votre projet. Le point de départ suggéré pour la taille de l'instance est `c5.2xLarge`.
8. Choisissez une paire de clés existante ou créez-en une nouvelle et enregistrez-en une nouvelle. Pour plus d'informations sur les paires de clés, consultez les paires de clés Amazon EC2 et les instances Linux dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.
9. Modifiez les paramètres réseau et choisissez le VPC autorisé et le sous-réseau approprié.
10. Choisissez un groupe de sécurité existant ou créez-en un nouveau. S'il s'agit d'une instance Amazon EC2 de serveur d'entreprise, il est courant d'autoriser le trafic TCP vers les ports 86 et 10086 pour administrer la configuration Micro Focus.
11. Configurez le stockage pour l'instance Amazon EC2.
12. Consultez le résumé et choisissez Launch instance. Pour que le lancement réussisse, le type d'instance doit être valide. Si le lancement échoue, choisissez Modifier la configuration de l'instance et choisissez un autre type d'instance.
13. Une fois le message de réussite affiché, choisissez Connect to instance.
14. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
15. Dans le volet de navigation de gauche, sous le menu Instances, sélectionnez Instances.
16. Dans le volet principal, vérifiez l'état de votre instance.

Commencer à utiliser NTT Data

Après avoir approvisionné l'instance Amazon EC2, connectez-vous à celle-ci par SSH avec le nom d'utilisateur. `ec2-user` L'écran ressemblera à l'image suivante.



Sous le `/opt/software/` dossier se trouve un dossier nomméUniKix_Product_Guides, comme illustré dans l'image suivante.



Le `UniKix_Product_Guides` dossier inclut la documentation des composants suivants installés sur cette instance Amazon EC2 :

- TYPE DE DONNÉES NTT
- BPE DE DONNÉES NTT
- NTT DATA Enterprise COBOL
- NTT DATA Sécurisé UniKix
- Directeur UniKix central de NTT DATA

Le `software` dossier qui apparaît dans l'image précédente contient les fichiers binaires des composants répertoriés ci-dessus.

Après avoir validé avec succès l'instance Amazon EC2, commencez à utiliser AWS Mainframe Modernization Replatform avec NTT DATA en suivant la documentation de NTT Data.

Applications dans le cadre de la AWS modernisation des mainframes

Si vous débutez dans le domaine de la modernisation des AWS mainframes, consultez les rubriques suivantes pour commencer :

- Qu'est-ce que la modernisation AWS du mainframe ?
- Configuration de la modernisation AWS du mainframe
- Tutoriel : Managed Runtime pour AWS Blu Age
- Tutoriel : environnement d'exécution géré pour Micro Focus

Dans AWS Mainframe Modernization, une application contient une charge de travail de mainframe migrée. L'application est analogue à une charge de travail sur le mainframe et est associée à un environnement d'exécution. Vous pouvez ajouter des fichiers batch et des ensembles de données aux applications et surveiller les applications lorsqu'elles s'exécutent. Vous créez des applications de modernisation AWS du mainframe pour chaque charge de travail que vous migrez. Lorsque vous créez une application de modernisation AWS du mainframe, vous spécifiez le moteur sur lequel l'application s'exécute lorsque vous la créez. Choisissez AWS Blu Age si vous utilisez le modèle de refactorisation automatique, et choisissez Micro Focus si vous utilisez le modèle de replateforme.

Rubriques

- Création d'une application de modernisation AWS du mainframe
- Déployer une application de modernisation AWS du mainframe
- Mettre à jour une application de modernisation AWS du mainframe
- Supprimer une application de modernisation AWS du mainframe d'un environnement
- Supprimer une application de modernisation AWS du mainframe
- Soumettre des tâches par lots pour les AWS applications de modernisation des mainframes
- Importer des ensembles de données pour les AWS applications de modernisation des ordinateurs centraux
- Gérez les transactions pour les AWS applications de modernisation des mainframes
- Création de AWS ressources pour une application migrée
- Configuration de l'application gérée

- AWS Référence de définition des applications de modernisation des mainframes
- AWSRéférence de définition des ensembles de données sur la modernisation du mainframe

Création d'une application de modernisation AWS du mainframe

Utilisez la console de modernisation du AWS mainframe pour créer une application de modernisation AWS du mainframe.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Création d'une application

Pour créer une application

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Dans la page Applications, choisissez Create application (Créer une application).
4. Sur la page Spécifier les informations de base, dans la section Nom et description, entrez le nom de l'application.
5. (Facultatif) Dans le champ Description de l'application, entrez une description de l'application. Cette description peut vous aider, ainsi que les autres utilisateurs, à identifier l'objectif de l'application.
6. Dans la section Type de moteur, choisissez Blu Age pour le refactoring automatique ou Micro Focus pour le replatforming.
7. Dans la section Clé KMS, choisissez Personnaliser les paramètres de chiffrement si vous souhaitez utiliser une AWS KMS clé gérée par le client. Pour plus d'informations, consultez Chiffrement des données interrompu pour le service de modernisation des AWS mainframes.

   > **Note**
   >
   > Par défaut, AWS Mainframe Modernization chiffre vos données à l'aide d'une AWS KMS clé que AWS Mainframe Modernization possède et gère pour vous. Toutefois, vous pouvez choisir d'utiliser une AWS KMS clé gérée par le client.

8. (Facultatif) Choisissez une AWS KMS clé par son nom ou Amazon Resource Name (ARN), ou choisissez Create an AWS KMS key pour accéder à la AWS KMS console et créer une nouvelle AWS KMS clé.
9. (Facultatif) Dans la section Balises, choisissez Ajouter une nouvelle balise pour ajouter une ou plusieurs balises d'application à votre application. Une balise d'application est une étiquette d'attribut personnalisée qui vous aide à organiser et à gérer vos AWS ressources).
10. Choisissez Suivant.
11. Dans la section Ressources et configurations, utilisez l'éditeur intégré pour saisir la définition de l'application. Vous pouvez également choisir Utiliser un fichier JSON de définition d'application dans un compartiment Amazon S3 et indiquer l'emplacement de la définition d'application que vous souhaitez utiliser. Pour plus d'informations, consultez AWS Exemple de définition d'application Blu Age ou Définition de l'application Micro Focus.
12. Choisissez Suivant.
13. Sur la page Réviser et créer, passez en revue les informations que vous avez saisies, puis choisissez Créer une application.

Déployer une application de modernisation AWS du mainframe

Utilisez la console de modernisation du AWS mainframe pour déployer une application de modernisation AWS du mainframe.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Déployer une application

Pour exécuter une application de modernisation AWS du mainframe, vous devez d'abord la déployer dans un environnement d'exécution. Une application peut avoir plusieurs versions. Chaque version d'une application possède sa propre définition d'application. Pour déployer une application, vous devez spécifier la version que vous souhaitez déployer.

Vous ne pouvez déployer qu'une seule version d'une application donnée à la fois. Si vous déployez une version d'une application, puis décidez de déployer une version différente, vous devez d'abord arrêter l'application si elle est en cours d'exécution.

Pour déployer une application

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Sur la page Applications, choisissez l'application que vous souhaitez déployer.
4. Choisissez Déployer l'application.
5. Dans la section Versions disponibles, choisissez la version que vous souhaitez déployer.
6. Dans la section Environnements, choisissez un environnement d'exécution dans lequel vous souhaitez exécuter votre application.
7. Choisissez Deploy (Déployer).

Pour déployer une version différente d'une application déployée

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'application.
3. Sur la page Applications, choisissez l'application que vous souhaitez déployer.
4. Dans le menu Actions, choisissez Arrêter l'application.
5. Une fois l'application arrêtée, choisissez Déployer l'application.
6. Dans la section Versions disponibles, choisissez la version que vous souhaitez déployer. Dans la section Environnements, l'environnement dans lequel l'application est déjà déployée est présélectionné.
7. Choisissez Deploy (Déployer).

Mettre à jour une application de modernisation AWS du mainframe

Utilisez la console de modernisation du AWS mainframe pour mettre à jour une application de modernisation AWS du mainframe.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Mise à jour d'une application

Une application de modernisation AWS du mainframe peut avoir plusieurs versions, chacune ayant sa propre définition d'application. Pour mettre à jour une application, fournissez une nouvelle définition d'application. Cela crée une nouvelle version de l'application.

Pour mettre à jour une application

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez mettre à jour a été créée.
3. Sur la page Applications, choisissez l'application que vous souhaitez mettre à jour.
4. Sur la page des détails de l'application, dans la section Définition actuelle, choisissez Modifier pour mettre à jour la définition actuelle de l'application.
5. Sur la page Mettre à jour l'application, utilisez l'éditeur intégré pour mettre à jour la définition actuelle de l'application.

   Vous pouvez également choisir Utiliser un fichier JSON de définition d'application dans un compartiment Amazon S3 et indiquer l'emplacement de la définition d'application que vous souhaitez utiliser. Pour plus d'informations, consultez AWS Exemple de définition d'application Blu Age ou Définition de l'application Micro Focus.
6. Lorsque vous avez terminé de mettre à jour la définition de l'application, choisissez Mettre à jour.

> **Note**
>
> Après avoir mis à jour l'application, vous devez la déployer à nouveau. Pour plus d'informations, consultez Déployer une application de modernisation AWS du mainframe.

Supprimer une application de modernisation AWS du mainframe d'un environnement

Vous pouvez supprimer une application de modernisation du AWS mainframe d'un environnement à l'aide de la console AWS Mainframe Modernization.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Supprimer une application d'un environnement

Si vous devez supprimer une application de modernisation AWS du mainframe alors qu'elle est en cours d'exécution, assurez-vous de l'arrêter au préalable. Vous pouvez voir le statut de la demande sur la page des candidatures.

Pour supprimer une application d'un environnement

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez supprimer de l'environnement a été créée.
3. Sur la page Applications, choisissez l'application que vous souhaitez supprimer de l'environnement, puis sélectionnez Actions.
4. (Facultatif) Si le statut de l'application est`Running`, choisissez Arrêter l'application.
5. Choisissez Supprimer de l'environnement.

   Le processus de suppression démarre immédiatement.

Supprimer une application de modernisation AWS du mainframe

Utilisez la console de modernisation du AWS mainframe pour supprimer une application de modernisation AWS du mainframe.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Supprimer une application

Si vous devez supprimer une application de modernisation AWS du mainframe alors qu'elle est en cours d'exécution, assurez-vous de l'arrêter au préalable. Vous pouvez voir le statut de la demande sur la page des candidatures.

Pour supprimer une application

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application que vous souhaitez supprimer a été créée.
3. Sur la page Applications, choisissez l'application que vous souhaitez supprimer, puis sélectionnez Actions.
4. (Facultatif) Si le statut de l'application est`Running`, choisissez Arrêter l'application.
5. Choisissez Supprimer l'application.
6. Dans la fenêtre Supprimer l'application, entrez `delete` pour confirmer que vous souhaitez supprimer l'application, puis choisissez Supprimer.

Soumettre des tâches par lots pour les AWS applications de modernisation des mainframes

Dans AWS Mainframe Modernization, vous pouvez soumettre des tâches par lots pour vos applications. Vous pouvez soumettre ou annuler des tâches par lots et consulter les informations relatives à l'exécution des tâches par lots. Chaque fois que vous soumettez une tâche par lots, AWS Mainframe Modernization crée une exécution de tâche par lots distincte. Vous pouvez surveiller l'exécution de cette tâche. Vous pouvez rechercher des tâches par lots par nom et fournir des fichiers JCL ou des fichiers de script aux tâches par lots.

> **Important**
>
> Si vous annulez un travail par lots, cela ne le supprime pas. Cela annule une exécution particulière de la tâche par lots. Les enregistrements des tâches par lots restent disponibles pour que vous puissiez les consulter dans les détails de l'exécution des tâches par lots.

Si votre traitement par lots nécessite l'accès à un ou plusieurs ensembles de données, utilisez la console AWS Mainframe Modernization ou le AWS Command Line Interface (AWS CLI) pour importer les ensembles de données. Pour plus d'informations, consultez Importer des ensembles de données pour les AWS applications de modernisation des ordinateurs centraux.

Ces instructions supposent que vous avez effectué les étapes dans Configuration de la modernisation AWS du mainframe et dansCréation d'une application de modernisation AWS du mainframe.

Soumettre une tâche par lots

Pour soumettre une tâche par lots

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application pour laquelle vous souhaitez soumettre un traitement par lots a été créée.
3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez soumettre un traitement par lots.

   > **Note**
   >
   > Avant de pouvoir soumettre un traitement par lots à une application, vous devez déployer l'application avec succès.

4. Sur la page des détails de l'application, sélectionnez Batch jobs.
5. Choisissez Soumettre une tâche.
6. Dans la section Sélectionnez un script, choisissez un script. Vous pouvez rechercher le script que vous souhaitez par son nom.
7. Choisissez Soumettre une tâche.

Importer des ensembles de données pour les AWS applications de modernisation des ordinateurs centraux

Grâce à la modernisation AWS du mainframe, vous pouvez importer des ensembles de données à utiliser avec vos applications. Vous pouvez spécifier les ensembles de données dans un fichier JSON stocké dans un compartiment Amazon S3, ou vous pouvez spécifier les valeurs de configuration des ensembles de données séparément. Après avoir importé les ensembles de données, vous pouvez consulter les détails de la tâche d'importation pour confirmer que les ensembles de données souhaités ont été importés. Tous les ensembles de données catalogués pour une application sont répertoriés ensemble dans la console.

Utilisez la console de modernisation du AWS mainframe pour importer des ensembles de données pour une application de modernisation AWS du mainframe.

Ces instructions supposent que vous avez effectué les étapes dans Configuration de la modernisation AWS du mainframe et dansCréation d'une application de modernisation AWS du mainframe.

Importer un ensemble de données

Pour importer un ensemble de données

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'application pour laquelle vous souhaitez importer des ensembles de données a été créée.
3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez importer des ensembles de données.
4. Sur la page des détails de l'application, sélectionnez Ensembles de données.
5. Choisissez Import (Importer).
6. Effectuez l'une des actions suivantes :
   - Choisissez Utiliser le fichier JSON de configuration d'ensemble de données dans un compartiment Amazon S3 et indiquez l'emplacement de la configuration de l'ensemble de données.
   - Choisissez Spécifier les valeurs de configuration de l'ensemble de données séparément avec une configuration guidée. Reportez-vous the section called "Référence de définition de l'ensemble de données" aux détails de définition spécifiques.

     Entrez le nom, l'organisation de l'ensemble de données (VSAM, GDG, PO, PS), l'emplacement et l'emplacement externe d'Amazon S3, ainsi que les paramètres pour chaque valeur de configuration de l'ensemble de données. Dans la configuration guidée, vous pouvez également choisir Generate JSON pour revoir la configuration JSON à partir de vos entrées.
7. Sélectionnez Envoyer.

Gérez les transactions pour les AWS applications de modernisation des mainframes

Avec la modernisation AWS du mainframe, vous pouvez exécuter une application, sur demande, en même temps que de nombreux autres utilisateurs qui soumettent des demandes pour exécuter la même application en utilisant les mêmes fichiers et programmes. Une transaction unique consiste en un ou plusieurs programmes d'application qui effectuent le traitement nécessaire.

Ces instructions supposent que vous avez effectué les étapes dans Configuration de la modernisation AWS du mainframe et dansCréation d'une application de modernisation AWS du mainframe.

Gérez les transactions pour les applications

Vous pouvez afficher et modifier les transactions pour les applications.

Pour gérer les transactions pour les applications

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez l'Région AWSendroit où l'application que vous souhaitez exécuter a été créée.
3. Sur la page Applications, choisissez l'application pour laquelle vous souhaitez gérer les transactions.
4. Dans l'onglet Transactions, sous Ressources relatives aux transactions, choisissez le mode d'affichage de vos ressources dans la liste déroulante. Vous pouvez afficher les ressources en fonction des ressources de transaction, des groupes, des listes ou des SIT.

    - Les ressources de transaction vous permettent de choisir le type de ressource en fonction des définitions de fichiers, des définitions de transactions, des définitions de programmes ou des définitions de files d'attente de données transitoires.

    > **Note**
    >
    > La transaction AWS Mainframe Modernization Managed prend en charge un plus grand nombre de types de ressources que ceux-ci, mais vous ne pouvez pas les

modifier directement ici. Les autres ressources doivent être modifiées en externe et vous devez recréer l'application avec des entrées de ressources mises à jour.

- Les groupes sont un ensemble de ressources transactionnelles. Vous pouvez également choisir les groupes que vous souhaitez associer à votre ressource de transaction.
- Les listes sont des ensembles ordonnés de groupes. Vous pouvez voir toutes vos ressources et tous vos groupes de transactions dans une vue de liste. La liste de démarrage détermine les ressources qui sont chargées lors de l'initialisation du serveur.
  - Avec le moteur de refactorisation Blu Age, vous spécifiez les listes à inclure au démarrage et le nombre de listes n'est pas limité.
  - Avec le moteur de replateforme Micro Focus, vous pouvez définir jusqu'à quatre listes dans un SIT.
- Le SIT (System Initialization Table) affiche toutes les configurations de transaction disponibles. Vous pouvez trouver des SIT en fonction de leurs propriétés (nom, description et listes de démarrage). Vous pouvez également choisir des listes à associer au SIT de votre choix.

> **Note**
>
> Les SIT ne sont applicables qu'au moteur de replateforme Micro Focus.

5. Choisissez une ressource de transaction pour afficher toutes les informations relatives à la ressource. Vous pouvez également consulter tous les attributs associés à votre ressource de transaction et consulter ou modifier tout attribut supplémentaire.
6. Si vous souhaitez modifier les informations relatives à votre ressource de transaction actuelle, choisissez Modifier.

   a. Sur la page Modifier, vous pouvez ajouter ou modifier la description de la ressource.

      1. Pour les ressources de transaction affichées sous forme de listes, vous pouvez également ajouter, supprimer ou réorganiser les listes comme vous le souhaitez.
      2. Pour des types de ressources spécifiques (définitions de fichiers, définitions de transactions, définitions de programmes et définitions de files d'attente de données transitoires), vous pouvez modifier les propriétés des ressources individuelles. Ces propriétés varient en fonction du moteur et du type de ressource.

   b. Après avoir apporté les modifications souhaitées, choisissez Enregistrer les modifications.

      Un message s'affiche lorsque la ressource est correctement mise à jour.

Création de AWS ressources pour une application migrée

Pour exécuter votre application migrée dansAWS, vous devez créer des AWS ressources avec d'autresServices AWS. Les ressources que vous devez créer sont les suivantes :

- Un compartiment S3 contenant le code de l'application, la configuration, les fichiers de données et les autres artefacts requis.
- Une base de données Amazon RDS ou Amazon Aurora contenant les données requises par l'application.
- EtAWS KMS key, qui est requis AWS Secrets Manager pour créer et stocker des secrets.
- Un secret du Gestionnaire de Secrets pour contenir les informations d'identification de la base de données.

> **Note**
>
> Chaque application migrée nécessite son propre ensemble de ressources. Il s'agit d'un ensemble minimum. Votre application peut également nécessiter des ressources supplémentaires, telles que les secrets Amazon Cognito ou les files d'attente MQ.

Autorisations nécessaires

Vérifiez que vous disposez des autorisations suivantes :

- `s3:CreateBucket`, `s3:PutObject`
- `rds:CreateDBInstance`
- `kms:CreateKey`
- `secretsmanager:CreateSecret`

Compartiment Amazon S3

Les applications refactorisées et replateformes nécessitent un compartiment S3 que vous configurez comme suit :

```
bucket-name/root-folder-name/application-name
```

bucket-name

N'importe quel nom respectant les contraintes de dénomination d'Amazon S3. Nous vous recommandons d'inclure le Région AWS nom dans le nom de votre bucket. Assurez-vous de créer le compartiment dans la même région que celle dans laquelle vous prévoyez de déployer l'application migrée.

root-folder-name

Nom requis pour satisfaire aux contraintes de la définition de l'application, que vous créez dans le cadre de l'application AWS Mainframe Modernization. Vous pouvez utiliser le `root-folder-name` pour distinguer les différentes versions d'une application, par exemple V1 et V2.

nom-application

Le nom de votre application migrée, par exemple, PlanetsDemo ou BankDemo.

Base de données

Les applications refactorisées et replateformes peuvent nécessiter une base de données. Vous devez créer, configurer et gérer la base de données conformément aux exigences spécifiques de chaque moteur d'exécution. AWS La modernisation du mainframe prend en charge le chiffrement en transit sur cette base de données. Si vous activez le protocole SSL sur votre base de données, assurez-vous de le spécifier `sslMode` dans le secret de la base de données ainsi que les détails de connexion de la base de données. Pour plus d'informations, consultez AWS Secrets Managersecret.

Si vous utilisez le modèle de refactorisation AWS Blu Age et que vous avez besoin d'une BluSam base de données, le moteur d'exécution AWS Blu Age nécessite une base de données Amazon Aurora PostgreSQL, que vous devez créer, configurer et gérer. La BluSam base de données est facultative. Créez cette base de données uniquement si votre application l'exige. Pour créer la base de données, suivez les étapes décrites dans la section Création d'un cluster de base de données Amazon Aurora dans le guide de l'utilisateur Amazon Aurora.

Si vous utilisez le modèle de replateforme Micro Focus, vous pouvez créer une base de données Amazon RDS ou Amazon Aurora PostgreSQL. Pour créer la base de données, suivez les étapes décrites dans Création d'une instance de base de données Amazon RDS dans le guide de l'utilisateur Amazon RDS ou dans Création d'un cluster de base de données Amazon Aurora dans le guide de l'utilisateur Amazon Aurora.

Pour les deux moteurs d'exécution, vous devez stocker les informations d'identification de la base de données à AWS Secrets Manager l'aide d'un AWS KMS key pour les chiffrer.

Clé AWS Key Management Service

Vous devez stocker les informations d'identification de la base de données de l'application de manière sécurisée dansAWS Secrets Manager. Pour créer un secret dans Secrets Manager, vous devez créer unAWS KMS key. Pour créer une clé KMS, suivez les étapes décrites dans la section Création de clés du Guide du AWS Key Management Service développeur.

Après avoir créé la clé, vous devez mettre à jour la politique de clé pour accorder des autorisations de déchiffrement à AWS Mainframe Modernization. Ajoutez les déclarations de politique suivantes :

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "kms:Decrypt",
   "Resource" : "*"
   }
```

AWS Secrets Managersecret

Vous devez stocker les informations d'identification de la base de données de l'application de manière sécurisée dansAWS Secrets Manager. Pour créer un secret, suivez les étapes décrites dans Créer un secret de base de données dans le Guide de AWS Secrets Manager l'utilisateur.

AWSLa modernisation du mainframe prend en charge le chiffrement en transit sur cette base de données. Si vous activez le protocole SSL sur votre base de données, assurez-vous de le spécifier `sslMode` dans le secret de la base de données ainsi que les détails de connexion de la base de données. Vous pouvez spécifier l'une des valeurs suivantes pour `sslMode` : `verify-full`verify-ca, ou`disable`.

Au cours du processus de création de la clé, choisissez Autorisations relatives aux ressources - facultatif, puis sélectionnez Modifier les autorisations. Dans l'éditeur de stratégie, ajoutez une stratégie basée sur les ressources, telle que la suivante, pour récupérer le contenu des champs chiffrés.

```
{
   "Effect" : "Allow",
   "Principal" : {
```

```
    "Service" : "m2.amazonaws.com"
    },
    "Action" : "secretsmanager:GetSecretValue",
    "Resource" : "*"
    }
```

Configuration de l'application gérée

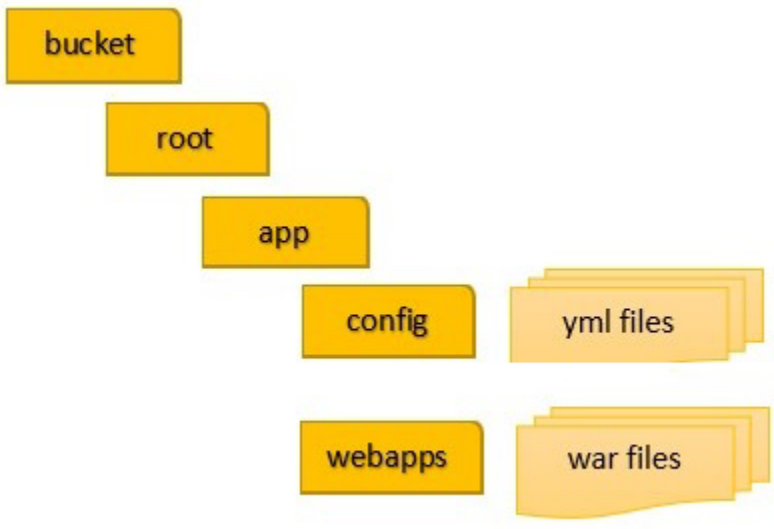
Vous pouvez configurer votre application pour inclure l'accès aux anciens utilitaires. Vous pouvez également personnaliser des propriétés supplémentaires. Afin de comprendre ce que vous pouvez configurer et où, il est utile de comprendre la structure globale d'une application AWS Blu Age modernisée.

Rubriques

- Structure des applications gérées par AWS Blu Age
- Configuration de l'accès aux utilitaires pour les applications gérées
- Ajouter des propriétés de configuration pour le moteur AWS Blu Age

Structure des applications gérées par AWS Blu Age

Si vous utilisez le modèle de refactorisation AWS Blu Age, le moteur d'exécution AWS Blu Age attend la structure suivante dans le `application-name` dossier de votre compartiment S3 :



config

Contient les fichiers YAML de votre projet. Il s'agit des fichiers YAML spécifiques à votre application, généralement appelés quelque chose comme ça, `application-planetsdemo.yaml` et non le `application-main.yaml` fichier que AWS Mainframe Modernization fournit et configure automatiquement pour vous.

applications Web

Contient les `war` fichiers de votre candidature. Ces fichiers sont le résultat du processus de modernisation.

Une application peut également comporter les dossiers facultatifs suivants :

jics/sql

Contient le `initJics.sql` script qui initialise la base de données JICS pour votre application.

scripts

Contient des scripts d'application, que vous pouvez également fournir directement dans les `war` fichiers.

sql

Contient des fichiers SQL d'application, que vous pouvez également fournir directement dans les `war` fichiers.

lnk

Contient les fichiers LNK de l'application, que vous pouvez également fournir directement dans les `war` fichiers.

Gérer la consommation de mémoire Java d'une application

Pour gérer la consommation de mémoire Java de l'application, ajoutez un fichier de propriétés nommé `tomcat.properties application-name` dans le dossier. Ce fichier peut avoir deux propriétés : `xms` qui spécifie la consommation de mémoire Java minimale et `xmx` qui spécifie la consommation de mémoire Java maximale. Voici un exemple du contenu d'un `tomcat.properties` fichier valide.

```
xms=512M
```

```
xmx=1G
```

Les valeurs que vous spécifiez pour ces deux propriétés peuvent être exprimées dans l'une des unités suivantes :

- Octets : ne spécifiez pas d'unité.
- Kilo-octets : ajoutez un K à la valeur.
- Mégaoctets : ajoutez un M à la valeur.
- Gigaoctets : ajoutez un G à la valeur.

Configuration de l'accès aux utilitaires pour les applications gérées

Lorsque vous refactorisez une application mainframe avec AWS Blu Age, vous devrez peut-être fournir un support pour divers programmes utilitaires de plate-forme existants, tels que IDCAMS, INFUTILB, SORT, etc., si votre application en dépend. AWS Le refactoring de Blu Age fournit cet accès grâce à une application Web dédiée qui est déployée parallèlement à des applications modernisées. Cette application Web nécessite un fichier de `application-utility-pgm.yml` configuration que vous devez fournir. Si vous ne fournissez pas ce fichier de configuration, l'application Web ne pourra pas être déployée en même temps que votre application et ne sera pas disponible.

Rubriques

- Propriétés de configuration

Cette rubrique décrit toutes les propriétés possibles que vous pouvez spécifier dans le fichier de `application-utility-pgm.yml` configuration, ainsi que leurs valeurs par défaut. La rubrique décrit les propriétés obligatoires et facultatives. L'exemple suivant est un fichier de configuration complet. Il répertorie les propriétés dans l'ordre que nous recommandons. Vous pouvez utiliser cet exemple comme point de départ pour votre propre fichier de configuration.

```
# If the datasource support mode is not static-xa, spring JTA transactions
 autoconfiguration must be disabled
 spring.jta.enabled: false
 logging.config: 'classpath:logback-utility.xml'

 # Encoding
 encoding: cp1047
```

```
 # Encoding to be used by INFUTILB and DSNUTILB to generate and read SYSPUNCH files
 sysPunchEncoding: cp1047

 # Utility database access
 spring.aws.client.datasources.primary.secret: `arn:aws:secretsmanager:us-
west-2:111122223333:secret:business-FfmXLG`

 treatLargeNumberAsInteger: false

 # Zoned mode : valid values = EBCDIC_STRICT, EBCDIC_MODIFIED, AS400
 zonedMode: EBCDIC_STRICT

 jcl.type: mvs

 # Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
  sqlCodePointShift: 384
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
  chunkSize:500
  fetchSize: 500
  varCharIsNull: false
  columnFiller: space

 # Load properties
 # Batch size for DSNUTILB Load Task
 load:
  sqlCodePointShift: 384
  batchSize: 500
  format:
    localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
    dbDate: yyyy-MM-dd
    localTime: 'HH:mm:ss|HH.mm.ss'
    dbTime: 'HH:mm:ss'
```

```
table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

Propriétés de configuration

Vous pouvez spécifier les propriétés suivantes dans votre fichier de configuration.

spring.jta.enabled

Facultatif. Contrôle si le support JTA est activé. Pour les utilitaires, nous vous recommandons de définir cette valeur sur`false`.

```
spring.jta.enabled : false
```

logging.config

Obligatoire. Spécifie le chemin d'accès au fichier de configuration de l'enregistreur dédié. Nous vous recommandons d'utiliser le nom `logback-utility.xml` et de fournir ce fichier dans le cadre de l'application modernisée. La méthode courante pour organiser ces fichiers consiste à placer tous les fichiers de configuration de l'enregistreur au même endroit, généralement dans le sous-dossier contenant le dossier `/config/logback` contenant les fichiers de configuration yaml. `/config` Pour plus d'informations, consultez le chapitre 3 : Configuration du logback dans la documentation Logback.

```
logging.config : classpath:logback-utility.xml
```

encoding

Obligatoire. Spécifie le jeu de caractères utilisé par le programme utilitaire. Dans la plupart des cas, lorsque vous migrez depuis des plateformes z/OS, ce jeu de caractères est une variante EBCDIC et doit correspondre au jeu de caractères configuré pour les applications modernisées. La valeur par défaut si elle n'est pas définie est`ASCII`.

```
encoding : cp1047
```

sysPunchEncoding

Facultatif. Spécifie le jeu de caractères utilisé par INFUTILB et DSNUTILB pour générer et lire les fichiers SYSPUNCH. Si vous utilisez les fichiers SYSPUNCH de l'ancienne plateforme tels

quels, cette valeur doit être une variante EBCDIC. La valeur par défaut si elle n'est pas définie est`ASCII`.

```
sysPunchEncoding : cp1047
```

Configuration de la source de données principale

Certains utilitaires liés aux bases de données, tels que LOAD et UNLOAD, nécessitent l'accès à une base de données cible via une source de données. Comme les autres définitions de sources de données dans AWS Mainframe Modernization, cet accès nécessite que vous AWS Secrets Manager utilisiez. Les propriétés qui pointent vers les secrets appropriés dans Secrets Manager sont les suivantes :

spring.aws.client.datasources.primary.secret

Facultatif. Spécifie le secret dans Secrets Manager qui contient les propriétés de la source de données.

```
spring.aws.client.datasources.primary.secret: datasource-secret-ARN
```

spring.aws.client.datasources.primary.dbname

Facultatif. Spécifie le nom de la base de données cible s'il n'est pas fourni directement dans le secret de base de données, avec la `dbname` propriété.

```
spring.aws.client.datasources.primary.dbname: target-database-name
```

treatLargeNumberAsInteger

Facultatif. Lié aux spécificités du moteur de base de données Oracle et à l'utilisation des utilitaires DSNTEP2/DSNTEP4. Si vous définissez cet indicateur sur true, les grands nombres provenant de la base de données Oracle (NUMBER (38,0)) sont traités comme des entiers. Par défaut : `false`

```
treatLargeNumberAsInteger : false
```

Mode zoné

Facultatif. Définit le mode zoné pour coder ou décoder les types de données zonés. Ce paramètre influence la façon dont les chiffres des signes sont représentés. Les valeurs suivantes sont valides :

- EBCDIC_STRICT : Par défaut. Utilisez une définition stricte pour la gestion des panneaux. Selon que le jeu de caractères est EBCDIC ou ASCII, la représentation numérique des signes utilise les caractères suivants :
  - Caractères EBCDIC correspondant à des octets (Cn+Dn) pour représenter des plages de chiffres positifs et négatifs (+0à+9, -0 à-9). Les caractères sont affichés sous { la forme I}, A J à, R
  - Caractères ASCII correspondant à des octets (3n+7n) pour représenter des plages de chiffres positifs et négatifs (+0à+9, -0 à-9). Les caractères sont affichés comme 0 9 suit p : y
- EBCDIC_MODIFIED : utilisez une définition modifiée pour la gestion des signes. Pour l'EBDIC et l'ASCII, la même liste de caractères représente les chiffres du signe, c'est-à-dire +9 mappés +0 vers { + vers I et mappés A vers + -0 vers-9. } J R \
- AS400 : à utiliser pour les actifs existants modernisés provenant des plateformes iSeries (AS400).

```
zonedMode:EBCDIC_STRICT
```

jcl type

Facultatif. Indique le type existant de scripts JCL modernisés. L'utilitaire IDCAMS utilise ce paramètre pour personnaliser le code de retour si le JCL invoquant est de type. vse Les valeurs valides sont les suivantes :

- mvs (par défaut)
- vse

```
jcl.type : mvs
```

Propriétés associées aux utilitaires de déchargement de base de données

Utilisez ces propriétés pour configurer les utilitaires qui déchargent les tables de base de données vers des ensembles de données. Toutes les propriétés suivantes sont facultatives.

Cet exemple montre toutes les propriétés de déchargement possibles.

```
# Unload properties
```

```
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntaxe to specify the byte value
unload:
sqlCodePointShift: 0
nbi:
whenNull: "6F"
whenNotNull: "00"
useDatabaseConfiguration: false
format:
date: MM/dd/yyyy
time: HH.mm.ss
timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
chunkSize: 0
fetchSize: 0
varCharIsNull: false
columnFiller: space
```

sqlCodePointShift

Facultatif. Spécifie une valeur entière qui représente le décalage de points de code SQL utilisé sur les données. La valeur par défaut est 0. Cela signifie qu'aucun changement de point de code n'est effectué. Alignez ce paramètre avec le paramètre de décalage des points de code SQL utilisé pour les applications modernisées. Lorsque le décalage des points de code est utilisé, la valeur la plus courante pour ce paramètre est 384.

```
unload.sqlCodePointShift: 0
```

nbi

Facultatif. Spécifie un octet indicateur nul. Il s'agit d'une valeur hexadécimale (sous forme de chaîne) ajoutée à droite de la valeur des données. Les deux valeurs possibles sont les suivantes :

- WhenNull : ajoutez la valeur hexadécimale lorsque la valeur des données est nulle. La valeur par défaut est 6`. Parfois, la valeur la plus élevée FF est utilisée à la place.

  ```
  unload.nbi.whenNull: "6F"
  ```

- whenNotNull: Ajoutez la valeur hexadécimale lorsque la valeur des données n'est pas nulle, mais que la colonne est nullable. La valeur par défaut est `00` (faible valeur).

  ```
  unload.nbi.whenNotNull: "00"
  ```

useDatabaseConfiguration

Facultatif. Spécifie les propriétés de formatage de la date et de l'heure. Ceci est utilisé pour traiter les objets date/heure dans les requêtes UNLOAD. La valeur par défaut est `false`.

- S'il est défini sur `true`pgmDateFormat, utilise les pgmTimestampFormat propriétéspgmTimeFormat, et du fichier de configuration principal (`application-main.yml`).
- S'il est défini sur`false`, utilise les propriétés de mise en forme de date et d'heure suivantes :
  - `unload.format.date`: Spécifie un modèle de mise en forme de date. La valeur par défaut est `MM/dd/yyyy`.
  - `unload.format.time`: Spécifie un modèle de formatage de l'heure. La valeur par défaut est `HH.mm.ss`.
  - `unload.format.timestamp`: Spécifie un modèle de formatage d'horodatage. La valeur par défaut est `yyyy-MM-dd-HH.mm.ss.SSSSSS`.

Taille du morceau

Facultatif. Spécifie la taille des segments de données utilisés pour créer des ensembles de données SYSREC. Ces ensembles de données sont la cible de l'opération de déchargement des ensembles de données, avec des opérations parallèles. La valeur par défaut est `0` (pas de morceaux).

```
unload.chunkSize:0
```

Taille de récupération

Facultatif. Spécifie la taille d'extraction des données. La valeur est le nombre d'enregistrements à récupérer simultanément lorsqu'une stratégie de segmentation de données est utilisée. Par défaut: `0`.

```
unload.fetchSize:0
```

varCharIsNull

Facultatif. Spécifie comment gérer une colonne varchar non nullable dont le contenu est vide. La valeur par défaut est `false`.

Si vous définissez cette valeur sur`true`, le contenu de la colonne est traité comme une chaîne vide à des fins de déchargement, au lieu d'une seule chaîne d'espace. Définissez cet indicateur sur `true` pour le cas du moteur de base de données Oracle uniquement.

```
unload.varCharIsNull: false
```

Remplisseur de colonnes

Facultatif. Spécifie la valeur à utiliser pour le remplissage des colonnes déchargées dans les colonnes varchar. Les valeurs possibles sont l'espace ou les valeurs faibles. La valeur par défaut est l'espace.

```
unload.columnFiller: space
```

Propriétés liées au chargement de la base de données

Utilisez ces propriétés pour configurer des utilitaires qui chargent des enregistrements d'ensembles de données dans une base de données cible, par exemple DSNUTILB. Toutes les propriétés suivantes sont facultatives.

Cet exemple montre toutes les propriétés de charge possibles.

```
# Load properties
 # Batch size for DSNUTILB Load Task
 load:
 sqlCodePointShift: 384
 batchSize: 500
 format:
 localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
 dbDate: yyyy-MM-dd
 localTime: HH:mm:ss|HH.mm.ss
 dbTime: HH:mm:ss

 table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

sqlCodePointShift

Facultatif. Spécifie une valeur entière qui représente le décalage de points de code SQL utilisé sur les données. La valeur par défaut est 0, ce qui signifie que les applications ne modifient aucun point de code. Alignez ce paramètre avec le paramètre de décalage des points de code SQL utilisé pour les applications modernisées. Lorsque vous utilisez des décalages de points de code, la valeur la plus courante pour ce paramètre est 384.

```
load.sqlCodePointShift : 384
```

batchSize

Facultatif. Spécifie une valeur entière qui représente le nombre d'enregistrements à traiter avant d'envoyer une instruction de lot réelle à la base de données. La valeur par défaut est 0.

```
load.batchSize: 500
```

format

Facultatif. Spécifie les modèles de mise en forme de date et d'heure à utiliser pour les conversions de date et d'heure lors des opérations de chargement de la base de données.

- `load.format.localDate`: modèle de formatage de date local. La valeur par défaut est `dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd`.
- `load.format.dbDate`: modèle de formatage des dates de base de données. La valeur par défaut est `yyyy-MM-dd`.
- `load.format.localTime`: modèle de formatage de l'heure locale. La valeur par défaut est `HH:mm:ss|HH.mm.ss`.
- `load.format.dbTime`: modèle de formatage horaire de la base de données. La valeur par défaut est `HH:mm:ss`.

mappages de tables

Facultatif. Spécifie un ensemble de mappages fournis par le client entre les noms de table existants et modernes. Le programme utilitaire DSNUTILB utilise ces mappages.

Spécifiez les valeurs au format suivant : MODERN_TABLE_NAME : LEGACY_TABLE_NAME

Voici un exemple :

```
table-mappings:
```

```
TABLE_1_NAME : LEGACY_TABLE_1_NAME
TABLE_2_NAME : LEGACY_TABLE_2_NAME
...
TABLE_*N*_NAME : LEGACY_TABLE_*N*_NAME
```

> **Note**
>
> Lorsque l'application utilitaire démarre, elle enregistre explicitement tous les mappages fournis.

Ajouter des propriétés de configuration pour le moteur AWS Blu Age

Vous pouvez ajouter un fichier dans le `config` dossier de votre application refactorisée qui vous donnera accès aux nouvelles fonctionnalités du moteur d'exécution AWS Blu Age. Vous devez nommer ce fichier`user-properties.yml`. Ce fichier ne remplace pas la définition de l'application mais l'étend. Cette rubrique décrit les propriétés que vous pouvez inclure dans le `user-properties.yml` fichier.

> **Note**
>
> Vous ne pouvez pas modifier certains paramètres car ils sont contrôlés soit par la modernisation du AWS mainframe, soit par la définition de l'application. Tous les paramètres définis dans la définition de l'application pour votre application ont priorité sur les paramètres que vous spécifiez dans`user-properties.yml`.

Pour plus d'informations sur la structure des applications refactorisées, consultez. Structure des applications gérées par AWS Blu Age

Le schéma suivant indique où placer le `user-properties.yml` fichier dans la structure de l'exemple d'application AWS Blu Age, PlanetsDemo.

```
PlanetsDemo-v1/
   ## config/
   #  ## application-PlanetsDemo.yml
   #  ## user-properties.yml
   ## jics/
   ## webapps/
```

Référence sur les propriétés de configuration

Voici la liste des propriétés disponibles. Tous les paramètres sont facultatifs.

Rubriques

Propriétés de l'application Gapwalk

Bluesam.fileloading.commit Intervalle

Facultatif. L'intervalle de validation du bluesam.

Type : nombre

Par défaut : 100000

encodage de la carte

Facultatif. Encodage de la carte : à utiliser avec`useControlMVariable`.

Type : chaîne

Par défaut : CP1145

vérifier la taille du fichier d'entrée

Facultatif. Spécifie s'il faut lancer une vérification si la taille du fichier est un multiple de la taille de l'enregistrement.

Type : valeur booléenne

Valeur par défaut : false

database.cursor.overflow.allowed

Facultatif. Spécifie s'il faut autoriser le dépassement du curseur. Réglez sur `true` pour effectuer un appel suivant sur le curseur, quelle que soit sa position. Réglez `false` sur pour vérifier si le curseur est à la dernière position avant d'effectuer un prochain appel sur le curseur. Activé uniquement si le curseur est SCROLLABLE (SENSITIVE ou INSENSITIVE)

Type : valeur booléenne

Valeur par défaut : true

Simplificateur de données. onInvalidNumericDonnées

Facultatif. Comment réagir lors du décodage de données numériques non valides Les valeurs autorisées sont `rejecttoleratespaces`,`toleratespaceslowvalues`,`toleratemost`.

Type : chaîne

Par défaut : rejeter

defaultKeepExistingDossiers

Facultatif. Spécifie s'il faut définir la valeur précédente par défaut de l'ensemble de données.

Type : valeur booléenne

Valeur par défaut : false

disposition.checkexistence

Facultatif. Spécifie s'il convient de vérifier l'existence du fichier pour Dataset avec DISP SHR ou OLD.

Type : valeur booléenne

Valeur par défaut : false

ExternalSort.Threshold

Facultatif. Le seuil de tri : quand passer au tri externe (fusion).

Type : chaîne

Par défaut:null

```
externalSort.threshold: 12MB
```

Force HR

Facultatif. Spécifie s'il faut utiliser SYSPRINT lisible par l'homme, soit en sortie de console, soit en sortie de fichier.

Type : valeur booléenne

Valeur par défaut : false

Date forcée

Facultatif. Force la saisie d'une date et d'une heure spécifiques dans la base de données. À utiliser uniquement pendant le développement et les tests.

Par défaut:null

```
forcedDate: 2022-08-26T12:59:58.123456+01:57
```

Date congelée

Facultatif. Gèle la date et l'heure dans la base de données. À utiliser uniquement pendant le développement et les tests.

Valeur par défaut : false

```
frozenDate: false
```

IMS.Messages.Taille étendue

Facultatif. Spécifie s'il faut définir la valeur ExtendedSize pour les messages ims.

Type : valeur booléenne

Valeur par défaut : false

Délai de verrouillage

Facultatif. Délai d'expiration en millisecondes d'une transaction en cas d'impossibilité d'obtenir un verrou dans un délai spécifié.

Type : nombre

Par défaut : 500

MapTransfo. Prefixes

Facultatif. Liste des préfixes à utiliser lors de la transformation des variables ControlM. Chacune est séparée par une virgule.

Type : chaîne

Par défaut : &, @, % %

requête. useConcatCondition

Facultatif. Spécifie si la condition clé est créée par concaténation de clés ou non.

Type : valeur booléenne

Valeur par défaut : false

Retour sur RTE

Facultatif. Spécifie s'il faut annuler la transaction d'unité d'exécution implicite sur les exceptions d'exécution.

Type : valeur booléenne

Valeur par défaut : false

sctThreadLimit

Facultatif. La limite de threads pour le déclenchement de scripts.

Type : nombre

Par défaut: 5

sqlCodePointShift

Facultatif. Le changement de point de code SQL. Déplace le point de code pour les caractères de contrôle que nous pouvons rencontrer lors de la migration des données d'un SGBDR existant vers un SGBDR moderne. Par exemple, vous pouvez spécifier de `384` faire correspondre le caractère Unicode`\u0180`.

Type : nombre

Par défaut : 0

sqlIntegerOverflowAutorisé

Facultatif. Spécifie s'il faut autoriser le dépassement des nombres entiers SQL, c'est-à-dire s'il est permis de placer des valeurs plus importantes dans la variable hôte.

Type : valeur booléenne

Valeur par défaut : false

stepFailWhenAbend

Facultatif. Spécifie s'il faut déclencher un abend en cas d'échec ou de fin d'exécution d'une étape.

Type : valeur booléenne

Valeur par défaut : true

stopExecutionWhenProgNotFound

Facultatif. Spécifie s'il faut arrêter l'exécution si aucun programme n'est trouvé. S'il est défini sur`true`, interrompt l'exécution si aucun programme n'est trouvé.

Type : valeur booléenne

Valeur par défaut : true

uppercaseUserInput

Facultatif. Spécifie si les données saisies par l'utilisateur doivent être en majuscules.

Type : valeur booléenne

Valeur par défaut : true

Utiliser la variable ControlM

Facultatif. Spécifie s'il faut utiliser la spécification Control-M pour le remplacement des variables.

Type : valeur booléenne

Valeur par défaut : false

Propriétés du batchscript Gapwalk

encoding

Facultatif. L'encodage utilisé dans les projets de script par lots (pas avec groovy). Exige un encodage valide `CP1047IBM930`,,`ASCII`,`UTF-8`...

Type : chaîne

Par défaut : ASCII

Propriétés de Gapwalk Blugen

managers.trancode

Facultatif. Le mappage des trancodes du gestionnaire de dialogues. Permet de mapper un code de transaction JICS à un gestionnaire de dialogue. Le format attendu est`trancode1:dialogManager1;trancode2:dialogManager2;`.

Type : chaîne

Par défaut:null

```
managers.trancode: OR12:MYDIALOG1
```

Propriétés de la commande Gapwalk CL

commandes désactivées

Facultatif. Liste des commandes à désactiver, séparées par des virgules. Les valeurs autorisées sont `PGM_BASICRCVMSG`,`SNDRCVF`,`CHGVAR`,`QCLRDTAQ`,,`RTVJOBA`,`ADDLFM`,`ADDPFM`,`RCVF`,`OVRDBF`,`DLTOVR`,`CP`. Utile lorsque vous souhaitez désactiver ou remplacer un programme existant. `PGM_BASIC`est un programme de vélocité spécifique conçu à des fins de débogage.

Type : chaîne

Par défaut:null

spring.datasource.primary.jndi-name

Facultatif. La principale source de données Java Naming And Directory Interface (jndi).

Type : chaîne

Par défaut : jdbc/primary

Mode zoné

Facultatif. Mode de codage ou de décodage des types de données zonés. Les valeurs autorisées sont`EBCDIC_STRICT`/`EBCDIC_MODIFIED`/`AS400`.

Type : chaîne

Par défaut : EBCDIC_STRICT

Propriétés du coureur Gapwalk CL

cl.configuration.context.encoding

Facultatif. L'encodage des fichiers CL. Exige un encodage valide `CP1047IBM930`,,`ASCII`,`UTF-8`...

Type : chaîne

Par défaut : CP297

CL. Mode zoné

Facultatif. Mode d'encodage ou de décodage des commandes du langage de contrôle (CL). Les valeurs autorisées sont`EBCDIC_STRICT`/`EBCDIC_MODIFIED`/`AS400`.

Type : chaîne

Par défaut : EBCDIC_STRICT

Propriétés de Gapwalk JHDB

ims.programs

Facultatif. Liste des programmes IMS à utiliser. Séparez chaque paramètre par un point-virgule (`;`) et chaque transaction par une virgule (). `,` Par exemple : `ims.programs: PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`

Type : chaîne

Par défaut:null

JHDB.Checkpoint Path

Facultatif. Si ce n'`jhdb.checkpointPersistence`est pas le cas`none`, ce paramètre vous permet de configurer le chemin de persistance du point de contrôle (emplacement de stockage du fichier checkpoint.dat). Toutes les données des points de contrôle contenues dans le registre sont sérialisées et sauvegardées dans un fichier (checkpoint.dat) situé dans le dossier fourni. Notez que seules les données du point de contrôle (ScriptID, StepID, position de la base de données et zone du point de contrôle) sont concernées par cette sauvegarde.

Type : chaîne

Par défaut : file :. /configuration/

JHDB.Checkpoint Persistance

Facultatif. Le mode de persistance du point de contrôle. Les valeurs autorisées sont`none`/`add`/`end`. `add`À utiliser pour conserver les points de contrôle lorsqu'un nouveau point est créé et ajouté au registre. `end`À utiliser pour conserver le point de contrôle lors de l'arrêt du serveur. Toute autre valeur désactive la persistance. Notez que chaque fois qu'un nouveau point de contrôle est ajouté au registre, tous les points de contrôle existants sont sérialisés et le fichier est effacé. Il ne s'agit pas d'un ajout aux données existantes du fichier. Ainsi, en fonction du nombre de points de contrôle, cela peut avoir un effet sur les performances.

Type : chaîne

Par défaut: Aucun

jhdb.configuration.context.encoding

Facultatif. Le codage JHDB (base de données hiérarchique Java). Exige une chaîne de codage valide`CP1047`,`IBM930`,`ASCII`,`UTF-8`...

Type : chaîne

Par défaut : CP297

jhdb. identificationCardData

Facultatif. Utilisé pour coder en dur certaines « données de la carte d'identification de l'opérateur » dans le champ MID désigné par le paramètre CARD.

Type : chaîne

Par défaut: ""

jhdb.lterm

Facultatif. Permet de forcer un identifiant de terminal logique commun dans le cas d'une émulation IMS. S'il n'est pas défini, SessionId est utilisé.

Type : chaîne

Par défaut:null

jhdb.metadata.extrapath

Paramètre de configuration qui spécifie un dossier racine supplémentaire spécifique à l'exécution pour les dossiers psbs et dbds.

Type : chaîne

Par défaut : file :. /configuration/

> **Note**
>
> Actuellement, pour des raisons de déploiement, vous devez copier vos répertoires dbds et psbs dans le répertoire de configuration de votre application ou dans un sous-répertoire du répertoire de configuration : par exemple, config/setup
>
> ```
> config
> |- setup
>    |- dbds
>    |- psbs
> ```
>
> et défini dans application-jhdb.yml
> `jhdb.metadata.extrapath: file: ./config/setup/`

jhdb.navigation.cachenexts

Facultatif. Durée du cache (en millisecondes) utilisée dans la navigation hiérarchique pour un SGBDR.

Type : nombre

Par défaut : 5000

jhdb.query. limitJoinUsage

Facultatif. Spécifie s'il faut utiliser le paramètre d'utilisation limite des jointures sur les graphes RDBMS.

Type : valeur booléenne

Valeur par défaut : true

jhdb. use-db-prefix

Facultatif. Spécifie s'il faut activer un préfixe de base de données dans la navigation hiérarchique pour un SGBDR.

Type : valeur booléenne

Valeur par défaut : true

Propriétés de Gapwalk JICS

jics.data. dataJsonInitEmplacement

Facultatif. Emplacement du fichier json préparé par l'analyseur à partir de l'analyse CSD et utilisé pour initialiser la base de données jics,

Type : chaîne

Par défaut: ""

jics.db. dataScriptLocation

Facultatif. Emplacement du script initJics.sql, préparé par Analyzer à partir de l'analyse des exportations CSD depuis le mainframe.

Type : chaîne

Par défaut: ""

jics.db. dataTestQueryEmplacement

Facultatif. Emplacement d'un script SQL contenant une seule requête SQL censée renvoyer un nombre d'objets (par exemple : compter le nombre d'enregistrements dans la table du

programme jics). Si le nombre est égal à 0, la base de données sera chargée à l'aide du `jics.db.dataScriptLocation` script, sinon le chargement de la base de données sera ignoré.

Type : chaîne

Par défaut: ""

jics.db. ddlScriptLocation

Facultatif. L'emplacement du script Jics DDL. Permet de lancer le schéma de base de données jics à l'aide d'un script .sql.

Type : chaîne

Par défaut: ""

```
jics.db.ddlScriptLocation: ./jics/sql/jics.sql
```

jics.db. schemaTestQueryEmplacement

Facultatif. Emplacement du fichier SQL qui doit contenir une requête unique renvoyant le nombre d'objets du schéma jics (le cas échéant).

Type : chaîne

Par défaut: ""

jics. runUnitLauncherPool.Enable

Facultatif. Spécifie s'il faut activer le pool de lanceurs d'unités exécutées dans JICS.

Type : valeur booléenne

Valeur par défaut : false

jics. runUnitLauncherTaille de la piscine

Facultatif. Taille du pool de lanceurs d'unités exécutées dans JICS.

Type : nombre

Valeur par défaut : 20

jics. runUnitLauncherPool. Intervalle de validation

Facultatif : intervalle de validation du pool de lanceurs d'unités exécutées dans JICS, exprimé en millisecondes.

Type : nombre

Par défaut: 1000

jics.queues.sqs.region

Facultatif. Le Région AWS pour Amazon SQS, utilisé dans JICS. Il est conseillé de définir la même région de l'application déployée pour des raisons de performances, mais ce n'est pas obligatoire.

Type : chaîne

Par défaut : eu-west-1

jics.xa.agent.timeout

Facultatif. Définit la durée maximale pendant laquelle l'agent xa chargé de gérer les transactions distribuées doit effectuer ses opérations.

Type : nombre

Par défaut:null

mq.queues.sqs.region

Facultatif. Le Région AWS pour le service Amazon SQS MQ.

Type : chaîne

Par défaut : eu-west-3

Exécuteur de tâches. allowCoreThreadTimeOut

Facultatif. Spécifie s'il faut autoriser les threads principaux à expirer dans JCIS. Cela permet une croissance et une réduction dynamiques, même en combinaison avec une file d'attente différente de zéro (étant donné que la taille maximale du pool n'augmentera que lorsque la file d'attente sera pleine).

Type : valeur booléenne

Valeur par défaut : false

Exécuteur de tâches. corePoolSize

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille du pool principal.

Type : nombre

Par défaut: 5

Exécuteur de tâches. maxPoolSize

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille maximale du pool (nombre maximal de threads parallèles).

Type : nombre

Par défaut: 10

TaskExecutor.QueueCapacity

Facultatif. Lorsqu'une transaction dans un terminal est initiée via un script groovy, un nouveau thread est créé. Utilisez ce paramètre pour configurer la taille de la file d'attente. (= nombre maximum de transactions en attente lorsqu'il `taskExecutor.maxPoolSize` est atteint)

Type : nombre

Par défaut: 50

Propriétés d'exécution de Gapwalk

Méta-données du cache

Facultatif. Spécifie s'il faut mettre en cache les métadonnées de base de données.

Type : valeur booléenne

Valeur par défaut : true

check-groovy-file

Facultatif. Spécifie s'il faut vérifier le contenu des fichiers groovy avant de les enregistrer.

Type : valeur booléenne

Valeur par défaut : true

Statistiques de base de données

Facultatif. Spécifie s'il faut autoriser les générateurs SQL à collecter et à afficher des informations statistiques.

Type : valeur booléenne

Valeur par défaut : false

dateTimeFormat

Facultatif. dateTimeFormat décrit comment répartir la date, l'heure et le type d'horodatage de la base de données dans des entités simplificatrices de données. Les valeurs autorisées sont`ISO/EUR/USA/LOCAL`

Type : chaîne

Par défaut : ISO

dbDateFormat

Facultatif. Format de date cible de la base de données.

Type : chaîne

Par défaut : yyyy-mm-dd

dbTimeFormat

Facultatif. Format horaire cible de la base de données.

Type : chaîne

Par défaut : HH:MM:SS

dbTimestampFormat

Facultatif. Format d'horodatage cible de la base de données.

Type : chaîne

Par défaut : YYYY-MM-DD HH:MM:SS.SSSSSS

Taille de récupération

Facultatif. La valeur FetchSize pour les curseurs. À utiliser lors de la récupération de données à l'aide de fragments par des utilitaires de chargement/déchargement.

Type : nombre

Par défaut: 10

Forcer la désactivation de SQL TrimStringType

Facultatif. Spécifie s'il faut désactiver le découpage de tous les paramètres de chaîne SQL.

Type : valeur booléenne

Valeur par défaut : false

localDateFormat

Facultatif. Liste des formats de date locaux. Séparez chaque format par |.

Type : chaîne

localTimeFormat

Facultatif. Liste des formats d'heure locale. Séparez chaque format par |.

Type : chaîne

localTimestampFormat

Facultatif. Liste des formats d'horodatage locaux. Séparez chaque format par |.

Type : chaîne

Par défaut :

pgmDateFormat

Facultatif. Format de date et d'heure utilisé dans les programmes.

Type : chaîne

Par défaut : yyyy-mm-dd

pgmTimeFormat

Facultatif. Le format d'heure utilisé pour l'exécution de pgm (programmes).

Type : chaîne

Par défaut : HH.mm.ss

pgmTimestampFormat

Facultatif. Le format d'horodatage.

Type : chaîne

Par défaut : yyyy-MM-DD-hh.mm.ss.ssssss

Propriétés du programme utilitaire Gapwalk

jcl type

Facultatif `.jcl`type de fichier. Les valeurs autorisées sont`jcl`/`vse`. Les commandes PRINT/REPRO de l'utilitaire IDCAMS renvoient 4 si le fichier est vide pour un jcl non vse.

Type : chaîne

Par défaut : mvs

listcat.préprocesseur à longueur variable .enabled

Facultatif. Spécifie s'il faut activer le préprocesseur de longueur variable pour la commande LISTCAT.

Type : valeur booléenne

Valeur par défaut : false

listcat.préprocesseur.type à longueur variable

Facultatif. Type d'objets contenus dans le fichier listcat, si vous l'activez`listcat.variablelengthpreprocessor.enabled`. Les valeurs autorisées sont`rdw`/`bdw`.

Type : chaîne

Par défaut : rdw

Charger. Taille du lot

Facultatif. Taille du lot utilitaire de chargement.

Type : nombre

Par défaut : 0

Charger .format.DBDate

Facultatif. Format de base de données de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : yyyy-mm-dd

Charger .format.DBTime

Facultatif. Durée d'utilisation de la base de données de l'utilitaire de chargement.

Type : chaîne

Par défaut : HH:MM:SS

Charger .format.localDate

Facultatif. Le format de date local de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : DD.MM.YYYY|DD/MM/YYYYY|YYYY-MM-DD

Load.format.LocalTime

Facultatif. Format d'heure locale de l'utilitaire de chargement à utiliser.

Type : chaîne

Par défaut : HH:MM:SS|HH.mm.ss

charge. sqlCodePointShift

Facultatif. L'utilitaire SQL Pointshift for Load. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible de DB2 est Postgresql.

Type : nombre

Par défaut : 0

sysPunchEncoding

Facultatif. Le jeu de caractères de codage Syspunch. Les valeurs prises en charge sont`Cp1047`/`ASCII`.

Type : chaîne

Par défaut : ASCII

treatLargeNumberAsInteger

Facultatif. Spécifie s'il faut traiter les grands nombres comme`Integer`. Ils sont traités comme `BigDecimal` par défaut.

Type : valeur booléenne

Valeur par défaut : false

Déchargez .chunksize

Facultatif. Taille du morceau utilisée pour l'utilitaire de déchargement.

Type : nombre

Par défaut : 0

Décharger .columnFiller

Facultatif. Le remplisseur de colonnes utilitaire de déchargement.

Type : chaîne

Par défaut : espace

Décharger .fetchSize

Facultatif. Vous permet de régler la taille de lecture lorsque vous manipulez des curseurs dans l'utilitaire de déchargement.

Type : nombre

Par défaut : 0

décharger.format.date

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format de date à utiliser dans l'utilitaire de déchargement. Pour plus d'informations, consultez unload.format.date.

Type : chaîne

Par défaut : MM/DD/YYYY

unload.format.time

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format d'heure à utiliser dans l'utilitaire de déchargement.

Type : chaîne

Par défaut : HH.mm.ss

décharger.format.timestamp

Facultatif. Si cette option `unload.useDatabaseConfiguration` est activée, le format d'horodatage à utiliser dans l'utilitaire de déchargement.

Type : chaîne

Par défaut : yyyy-MM-DD-hh.mm.ss.ssssss

déchargez .nbi. whenNotNull

Facultatif. La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données n'est pas nulle.

Type : hexadécimal

Par défaut : 00

Décharger .nbi.whennull

Facultatif. La valeur de l'indicateur d'octet nul (nbi) à ajouter lorsque la valeur de la base de données est nulle.

Type : hexadécimal

Par défaut : 6F

déchargez .nbi. writeNullIndicator

Facultatif. Spécifie s'il faut écrire l'indicateur nul dans le fichier de sortie de déchargement.

Type : valeur booléenne

Valeur par défaut : false

décharger. sqlCodePointShift

Facultatif. L'utilitaire SQL Pointshift for Unload. Exécute le processus de changement de personnage. Obligatoire lorsque votre base de données cible de DB2 est Postgresql.

Type : nombre

Par défaut : 0

décharger. useDatabaseConfiguration

Facultatif. Spécifie s'il faut utiliser la configuration de date ou d'heure de application-main.yml dans l'utilitaire de déchargement.

Type : valeur booléenne

Valeur par défaut : false

décharger. varCharIsNull

Facultatif. Utilisez ce paramètre dans le programme INFTILB. S'il est défini sur, tous les champs non nullables contenant des valeurs vides (espaces) renvoient une chaîne vide. `true`

Type : valeur booléenne

Valeur par défaut : false

Autres propriétés

qtemp.cleanup.threshold. hours

Facultatif. Pour spécifier quand `qtemp.dblog` est activé. Durée de vie de la partition de base de données (en heures).

Type : nombre

Par défaut : 0

qtemp.dblog

Facultatif. S'il faut activer la journalisation de la base de données QTEMP.

Type : valeur booléenne

Valeur par défaut : false

qtemp.uuid.length

Facultatif. La longueur de l'identifiant unique QTEMP.

Type : nombre

Par défaut : 9

quartz.scheduler. stand-by-if-error

Facultatif. Spécifie s'il faut déclencher l'exécution des tâches si le planificateur de tâches est en mode veille. Si vrai, lorsque cette option est activée, l'exécution de la tâche n'est pas déclenchée.

Type : valeur booléenne

Valeur par défaut : false

warmUpCache

Facultatif. Spécifie s'il faut charger toutes les données de la table Datacom dans un cache de préchauffage au démarrage du serveur.

Type : valeur booléenne

Valeur par défaut : false

AWS Référence de définition des applications de modernisation des mainframes

Dans AWS Mainframe Modernization, vous configurez les applications mainframe migrées dans un fichier JSON de définition d'application, spécifique au moteur d'exécution que vous choisissez. Une définition d'application contient à la fois des informations générales et des informations spécifiques au moteur. Cette rubrique décrit les définitions des applications AWS Blu Age et Micro Focus et identifie tous les éléments obligatoires et facultatifs.

Rubriques

Section d'en-tête générale

Chaque définition d'application commence par des informations générales sur la version du modèle et les emplacements des sources. La version actuelle de la définition de l'application est 2.0. Bien que la version 1 fonctionne toujours, elle est sur le point de devenir obsolète. Nous vous recommandons d'utiliser la version 2 lorsque vous créez ou mettez à jour des applications.

Utilisez la structure suivante pour spécifier la version du modèle et les emplacements des sources.

```
"template-version": "2.0",
    "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ]
```

> **Note**
>
> Vous pouvez utiliser la syntaxe suivante si vous souhaitez saisir l'ARN S3 en tant que s3-bucket :
>
> ```
> "template-version": "2.0",
>   "source-locations": [
>       {
>           "source-id": "s3-source",
> ```

```
            "source-type": "s3",
            "properties": {
                "s3-bucket": "arn:aws:s3:::mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ]
```

version du modèle

Obligatoire. Spécifie la version du fichier de définition de l'application. Ne modifiez pas cette valeur. Actuellement, la seule valeur autorisée est 2,0. Spécifiez `template-version` avec une chaîne.

localisations des sources

Spécifie l'emplacement des fichiers et des autres ressources dont l'application a besoin pendant l'exécution.

propriétés

Fournit les détails de l'emplacement de la source. Chaque propriété est spécifiée avec une chaîne.

- `s3-bucket`- Obligatoire. Spécifie le nom du compartiment Amazon S3 dans lequel les fichiers sont stockés.
- `s3-key-prefix`- Obligatoire. Spécifie le nom du dossier dans le compartiment Amazon S3 dans lequel les fichiers sont stockés.

> **Note**
>
> Assurez-vous de spécifier le nom du compartiment Amazon S3, et non l'ARN du compartiment. Ne spécifiez pas de chemin absolu vers les ressources du compartiment.

Présentation de la section consacrée aux définitions

Spécifie les définitions des ressources des services, des paramètres, des données et des autres ressources typiques dont l'application a besoin pour s'exécuter. Lorsque vous mettez à jour une définition d'application, AWS Mainframe Modernization détecte les modifications en comparant les

`definition` listes `source-locations` et des versions précédente et actuelle du fichier JSON de définition d'application.

La section de définition est spécifique au moteur et peut être modifiée. Les sections suivantes présentent des exemples de définitions d'applications spécifiques au moteur pour les deux moteurs.

AWS Exemple de définition d'application Blu Age

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 8194,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/murachs-v6/"
        },
        "blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
```

```
        }
    }
}
```

AWS Détails de la définition de Blu Age

Écouteur (s) - obligatoire

Spécifiez le port que vous utiliserez pour accéder à l'application via l'Elastic Load Balancing créé par AWS Mainframe Modernization. Utilisez la structure suivante :

```
"listeners": [{
            "port": 8194,
            "type": "http"
        }],
```

port

Obligatoire. Vous pouvez utiliser n'importe quel port disponible, à l'exception des ports connus compris entre 0 et 1023. Nous recommandons d'utiliser la plage de 8192 à 8199. Assurez-vous qu'aucun autre écouteur ou application ne fonctionne sur ce port.

type

Obligatoire. Actuellement, seul `http` est pris en charge.

AWS Application Blu Age - obligatoire

Spécifiez l'emplacement où le moteur récupère le fichier image de l'application à l'aide de la structure suivante.

```
"ba-application": {
            "app-location": "${s3-source}/murachs-v6/",
            "files-directory": "/m2/mount/myfolder",
            "enable-jics": <true|false>,
            "shared-app-location": "${s3-source}/shared/"
        },
```

emplacement de l'application

Emplacement spécifique dans Amazon S3 où le fichier image de l'application est stocké.

répertoire-fichiers

Facultatif. Emplacement des fichiers d'entrée/sortie pour les lots. Il doit s'agir d'un sous-dossier de la configuration du point de montage Amazon EFS ou Amazon FSx au niveau de l'environnement.

activer-jics

Facultatif. Spécifie s'il faut activer JICS. La valeur par défaut est true (vrai). La définition de ce paramètre sur false empêche la création de la base de données JICS.

shared-app-location

Facultatif. Emplacement supplémentaire dans Amazon S3 où les éléments d'application partagés sont stockés. Il peut contenir le même type de structure d'application que app-location.

BluSam - facultatif

Spécifiez la base de données BluSam et le cache Redis à l'aide de la structure suivante.

```
"blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
        }
```

db

Spécifie les propriétés de la base de données utilisée avec l'application. La base de données doit être une base de données Aurora PostgreSQL. Vous pouvez définir les propriétés suivantes :

- `nb-threads`- Facultatif. Spécifie le nombre de threads dédiés utilisés pour le mécanisme d'écriture secondaire sur lequel repose le moteur blusam. La valeur par défaut est de 8.

- `batch-size`- Facultatif. Spécifie le seuil utilisé par le mécanisme d'écriture différée pour démarrer les opérations de stockage par lots. Le seuil représente le nombre d'enregistrements modifiés qui démarreront une opération de stockage par lots afin de garantir la persistance des enregistrements modifiés. Le déclencheur lui-même est basé sur la combinaison de la taille du lot et d'un temps écoulé d'une seconde, selon la première valeur atteinte. La valeur par défaut est 10 000.
- `name`- Facultatif. Spécifie le nom de la base de données.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient les informations d'identification de la base de données. Pour de plus amples informations, veuillez consulter Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données.

redis

Spécifie les propriétés du cache Redis que l'application utilise pour stocker les données temporaires dont elle a besoin dans un emplacement central afin d'améliorer les performances. Nous vous recommandons de chiffrer et de protéger le cache Redis par mot de passe.

- `hostname`- Spécifie l'emplacement du cache Redis.
- `port`- Spécifie le port, généralement 6379, où le cache Redis envoie et reçoit les communications.
- `useSsl`- Spécifie si le cache Redis est crypté. Si le cache n'est pas chiffré, `useSsl` définissez-le sur false.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient le mot de passe du cache Redis. Si le cache Redis n'est pas protégé par mot de passe, ne le spécifiez pas. `secret-manager-arn` Pour de plus amples informations, veuillez consulter Étape 4 : Création et configuration d'un secret AWS Secrets Manager de base de données.

AWS files d'attente de messages Blu Age - facultatif

Spécifiez les détails de connexion JMS-MQ pour l'application AWS Blu Age.

```
"message-queues": [
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr1",
      "channel": "mqChannel1",
      "hostname": "mqserver-host1",
      "port": 1414,
```

```
      "user-id": "app-user1",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  },
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr2",
      "channel": "mqChannel2",
      "hostname": "mqserver-host2",
      "port": 1412,
      "user-id": "app-user2",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  }
]
```

type de produit

Obligatoire. Spécifie le type de produit. Actuellement, il ne peut s'agir que de « JMS-MQ » pour les applications AWS Blu Age.

gestionnaire de files d'attente

Obligatoire. Spécifie le nom du gestionnaire de files d'attente.

channel

Obligatoire. Spécifie le nom du canal de connexion au serveur.

hostname

Obligatoire. Spécifie le nom d'hôte du serveur de file d'attente de messages.

port

Obligatoire. Spécifie le numéro de port du récepteur sur lequel le serveur écoute.

identifiant d'utilisateur

Facultatif. Spécifie l'ID du compte utilisateur autorisé à effectuer des opérations de file de messages sur le canal spécifié.

secret-manager-arn

Facultatif. Spécifie le nom de ressource Amazon (ARN) de Secrets Manager qui fournit le mot de passe de l'utilisateur spécifié.

Définition de l'application Micro Focus

L'exemple de section de définition suivant concerne le moteur d'exécution Micro Focus et contient des éléments obligatoires et facultatifs.

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 5101,
            "type": "tn3270"
        }],
        "dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }]
        },
        "cognito-auth-handler": {
            "user-pool-id": "cognito-idp.us-west-2.amazonaws.com/us-west-2_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "us-west-2:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
        },
        "ldap-ad-auth-handler": {
            "ldap-ad-connection-secrets": [LIST OF AD-SECRETS]
        },
        "batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl"
        },
```

```
        "cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
        },
        "xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
        }]
    }
}
```

Détails de la définition de Micro Focus

Le contenu de la section de définition du fichier de définition de l'application Micro Focus varie en fonction des ressources dont votre application mainframe migrée a besoin au moment de l'exécution.

Écouteur (s) - obligatoire

Spécifiez un écouteur à l'aide de la structure suivante :

```
"listeners": [{
    "port": 5101,
    "type": "tn3270"
}],
```

port

Pour tn3270, la valeur par défaut est 5101. Pour les autres types d'auditeurs de service, le port varie. Vous pouvez utiliser n'importe quel port disponible, à l'exception des ports connus compris entre 0 et 1023. Chaque écouteur doit avoir un port distinct. Les écouteurs ne doivent pas partager de ports. Pour plus d'informations, voir Listener Control dans la documentation de Micro Focus Enterprise Server.

type

Spécifie le type d'écouteur de service. Pour plus d'informations, consultez la section Listeners dans la documentation de Micro Focus Enterprise Server.

Emplacement des ensembles de données : obligatoire

Spécifiez l'emplacement de l'ensemble de données à l'aide de la structure suivante.

```
"dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }],
 }
```

emplacements de base de données

Spécifie l'emplacement des ensembles de données créés par l'application migrée. Actuellement, la modernisation AWS du mainframe ne prend en charge que les ensembles de données provenant d'une seule base de données VSAM.

- `name`- Spécifie le nom de l'instance de base de données qui contient les ensembles de données créés par l'application migrée.
- `secret-manager-arn`- Spécifie le nom de ressource Amazon (ARN) du secret qui contient les informations d'identification de la base de données.

Gestionnaire d'authentification et d'autorisation Amazon Cognito (facultatif)

AWS La modernisation du mainframe utilise Amazon Cognito pour l'authentification et l'autorisation des applications migrées. Spécifiez le gestionnaire d'authentification Amazon Cognito à l'aide de la structure suivante.

```
"cognito-auth-handler": {
            "user-pool-id": "cognito-idp.Region.amazonaws.com/Region_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "Region:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
}
```

user-pool-id

Spécifie le groupe d'utilisateurs Amazon Cognito que AWS Mainframe Modernization utilise pour authentifier les utilisateurs de l'application migrée. La valeur Région AWS destinée au groupe

d'utilisateurs doit correspondre à celle de Région AWS l'application de modernisation du AWS mainframe.

identificateur du client

Spécifie l'application migrée à laquelle l'utilisateur authentifié peut accéder.

identity-pool-id

Spécifie le pool d'identités Amazon Cognito dans lequel l'utilisateur authentifié échange un jeton de groupe d'utilisateurs contre des informations d'identification lui permettant d'accéder AWS à Mainframe Modernization. Le nom Région AWS du pool d'identités doit correspondre à celui de Région AWS l'application de modernisation du AWS mainframe.

Gestionnaire LDAP et Active Directory (facultatif)

Vous pouvez intégrer votre application à Active Directory (AD) ou à n'importe quel type de serveur LDAP pour permettre aux utilisateurs de l'application d'utiliser leurs informations d'identification LDAP/AD pour l'autorisation et l'authentification.

Pour intégrer votre application à AD

1. Suivez les étapes décrites dans Configuration d'Active Directory pour la sécurité des serveurs d'entreprise dans la documentation de Micro Focus Enterprise Server.
2. Créez un AWS Secrets Manager secret contenant vos informations AD/LDAP pour chaque serveur AD/LDAP que vous souhaitez utiliser avec votre application. Pour plus d'informations sur la création d'un secret, consultez la section Créer un secret AWS Secrets Manager dans le Guide de AWS Secrets Manager l'utilisateur. Pour le type de secret, choisissez Autre type de secret et incluez les paires clé-valeur suivantes.

```
{
    "connectionPath"       : "<HOST-ADDRESS>:<PORT>",
    "authorizedId"         : "<USER-FULL-DN>",
    "password"             : "<PASSWORD>",
    "baseDn"               : "<BASE-FULL-DN>",
    "userClassDn"          : "<USER-TYPE>",
    "userContainerDn"      : "<USER-CONTAINER-DN>",
    "groupContainerDn"     : "<GROUP-CONTAINER-DN>",
    "resourceContainerDn"  : "<RESOURCE-CONTAINER-DN>"
}
```

⚠ Recommandations en matière de sécurité

- En effet`connectionPath`, la modernisation AWS du mainframe prend en charge les protocoles LDAP et LDAP over SSL (LDAPS). Nous vous recommandons d'utiliser le protocole LDAPS car il est plus sécurisé et empêche les informations d'identification d'apparaître dans les transmissions réseau.
- Pour `authorizedId` et`password`, nous vous recommandons de spécifier les informations d'identification d'un utilisateur ne disposant que des autorisations de lecture seule et de vérification les plus restrictives requises pour l'exécution de votre application.
- Nous vous recommandons de changer régulièrement les informations d'identification AD/LDAP.
- Ne créez pas d'utilisateurs AD avec le nom d'utilisateur `awsuser` ou`mfuser`. Ces deux noms d'utilisateur sont réservés à l' AWS usage.

Voici un exemple.

```
{
    "connectionPath" : "ldaps://msad4.m2.example.people.aws.dev:636",
    "authorizedId" :
 "CN=LDAPUser,OU=Users,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "password" : "ADPassword",
    "userContainerDn" : "CN=Enterprise Server Users,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "groupContainerDn" : "CN=Enterprise Server Groups,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "resourceContainerDn" : "CN=Enterprise Server Resources,CN=Micro
 Focus,CN=Program Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev"
}
```

Créez le secret à l'aide d'une clé KMS gérée par le client. Vous devez accorder à AWS Mainframe Modernization les `DescribeSecret` autorisations `GetSecretValue` et sur le secret `Decrypt` et `DescribeKey` les autorisations sur la clé KMS. Pour plus d'informations, consultez la section Autorisations relatives à la clé KMS dans le guide de AWS Secrets Manager l'utilisateur.

3. Ajoutez ce qui suit à la définition de votre application.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": [LIST OF AD/LDAP SECRETS]
}
```

Voici un exemple.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": ["arn:aws:secrets:1234:us-east-1:secret:123456"]
}
```

Le gestionnaire d'authentification LDAP/AD est disponible pour Micro Focus 8.0.11 et versions ultérieures.

Réglages par lots : obligatoire

Spécifiez les détails requis par les tâches par lots exécutées dans le cadre de l'application à l'aide de la structure suivante.

```
"batch-settings": {
            "initiators": [{
                "classes": ["A","B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcls"
 }
```

initiateurs

Spécifie un initiateur par lots qui démarre lorsque l'application migrée démarre avec succès et continue de fonctionner jusqu'à ce que l'application s'arrête. Vous pouvez définir une ou plusieurs classes par initiateur. Vous pouvez également définir plusieurs initiateurs. Par exemple :

```
"batch-settings": {
            "initiators": [
                {
                  "classes": ["A", "B"],
                  "description": "initiator...."
                },
```

```
                {
                  "classes": ["C", "D"],
                  "description": "initiator...."
                }
            ],
            "jcl-file-location": "${s3-source}/batch/jcls"
}
```

Pour plus d'informations, voir Pour définir un initiateur par lots ou un SEP d'imprimante dans la documentation de Micro Focus Enterprise Server.

- `classes`- Spécifie les classes de travail que l'initiateur peut exécuter. Vous pouvez utiliser jusqu'à 36 caractères. Vous pouvez utiliser les caractères suivants : A-Z ou 0-9.
- `description`- Décrit à quoi sert l'initiateur.
- `jcl-file-location`- Spécifie l'emplacement des fichiers JCL requis par les tâches par lots exécutées par l'application migrée.

Réglages CICS - obligatoires

Spécifiez les détails requis pour les transactions CICS exécutées dans le cadre de l'application à l'aide de la structure suivante.

```
"cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
        }
```

binary-file-location

Spécifie l'emplacement des fichiers du programme de transaction CICS.

csd-file-location

Spécifie l'emplacement du fichier de définition de ressource CICS (CSD) pour cette application. Pour plus d'informations, consultez les définitions des ressources CICS dans la documentation de Micro Focus Enterprise Server.

system-initialization-table

Spécifie la table d'initialisation du système (SIT) utilisée par l'application migrée. Le nom de la table SIT peut comporter jusqu'à 8 caractères. Vous pouvez utiliser A-Z, 0-9, $, @ et #. Pour plus

d'informations, consultez les définitions des ressources CICS dans la documentation de Micro Focus Enterprise Server.

Ressources XA - obligatoires

Spécifiez les détails requis pour les ressources XA dont l'application a besoin à l'aide de la structure suivante.

```
"xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
        }]
```

name

Obligatoire. Spécifie le nom de la ressource XA.

secret-manager-arn

Spécifie le nom de ressource Amazon (ARN) pour le secret qui contient les informations d'identification pour la connexion à la base de données.

module

Spécifie l'emplacement du fichier exécutable du module de commutation RM. Pour plus d'informations, consultez la section Planification et conception de XAR dans la documentation de Micro Focus Enterprise Server.

AWSRéférence de définition des ensembles de données sur la modernisation du mainframe

Si le traitement de votre application nécessite plusieurs ensembles de données, il est inefficace de les saisir un par un dans la console de modernisation du AWS mainframe. Nous vous recommandons plutôt de créer un fichier JSON pour spécifier chaque ensemble de données. Les différents types d'ensembles de données sont spécifiés différemment dans le JSON, bien que de nombreux paramètres soient communs. Ce document décrit les détails du JSON requis pour importer différents types d'ensembles de données.

> **Note**
>
> Avant d'importer des ensembles de données, vous devez les transférer du mainframe versAWS. Vous devez ensuite vous assurer que les ensembles de données sont convertis du format mainframe vers un format AWS utilisable. Si nécessaire, transformez les données selon vos besoins et stockez les ensembles de données transformés dans Amazon S3. Spécifiez le nom du compartiment et du dossier dans le fichier JSON de définition de l'ensemble de données.
>
> Si vous utilisez le moteur d'exécution Micro Focus, vous pouvez utiliser l'`DFCONV`utilitaire pour convertir les ensembles de données. Nous incluons cet utilitaire dans nos images Micro Focus Enterprise Developer et Enterprise Server. Pour plus d'informations, consultez la section Conversion de fichiers par lots DFCONV dans la documentation Micro Focus Enterprise Developer.

Rubriques

Propriétés communes

Plusieurs paramètres sont communs à tous les ensembles de données. Ces paramètres couvrent les domaines suivants :

- Informations sur l'ensemble de données (`datasetName`,`datasetOrg`,`recordLength`,`encoding`)
- Informations sur l'emplacement d'où vous effectuez l'importation, c'est-à-dire l'emplacement source de l'ensemble de données. Il ne s'agit pas de l'emplacement sur le mainframe. Il s'agit du chemin d'accès à l'emplacement Amazon S3 où vous avez chargé le jeu de données (`externalLocation`).

- Informations sur l'emplacement vers lequel vous effectuez l'importation, c'est-à-dire l'emplacement cible de l'ensemble de données. Cet emplacement est soit une base de données, soit un système de fichiers, selon votre moteur d'exécution. (`storageTypeetrelativePath`).
- Informations sur le type de jeu de données (type d'ensemble de données spécifique, format, encodage, etc.).

Chaque définition d'ensemble de données possède la même structure JSON. L'exemple JSON suivant montre tous ces paramètres courants.

```
{
    "dataSet": {
        "storageType": "Database",
        "datasetName": "MFI01V.MFIDEMO.BNKACC",
        "relativePath": "DATA",
        "datasetOrg": {
            "type": {
                type-specific properties
                    ...
            },
        },
    },
}
```

Les propriétés suivantes sont communes à tous les ensembles de données.

storageType

Obligatoire. S'applique à l'emplacement cible. Spécifie si l'ensemble de données est stocké dans une base de données ou un système de fichiers. Les valeurs possibles sont `Database` ou `FileSystem`.

- AWSMoteur d'exécution Blu Age : les systèmes de fichiers ne sont pas pris en charge. Vous devez utiliser une base de données.
- Moteur d'exécution Micro Focus : les bases de données et les systèmes de fichiers sont tous deux pris en charge. Vous pouvez utiliser Amazon Relational Database Service ou Amazon Aurora pour les bases de données, et Amazon Elastic File System ou Amazon FSx for Lustre pour les systèmes de fichiers.

datasetName

Obligatoire. Spécifie le nom complet de l'ensemble de données tel qu'il apparaît sur le mainframe.

Trajectoire relative

Obligatoire. S'applique à l'emplacement cible. Spécifie l'emplacement relatif de l'ensemble de données dans la base de données ou le système de fichiers.

Ensemble de données Org

Obligatoire. Spécifie le type de jeu de données. Les valeurs possibles sont `vsam`, `gdg`, `ps`, `po` ou `unknown`.

- AWSMoteur d'exécution Blu Age : seuls les ensembles de données de type VSAM sont pris en charge.
- Moteur d'exécution Micro Focus : les ensembles de données de type VSAM, GDG, PS, PO ou Unknown sont pris en charge.

> **Note**
>
> Si votre application nécessite des fichiers qui ne sont pas des fichiers de données COBOL mais des fichiers PDF ou d'autres fichiers binaires, vous pouvez les spécifier comme suit :
>
> ```
> "datasetOrg": {
>             "type": PS {
>                 "format": U
>             },
> ```

Exemple de format de demande d'ensemble de données pour VSAM

- AWSMoteur d'exécution Blu Age : pris en charge.
- Moteur d'exécution Micro Focus : pris en charge.

Si vous importez des ensembles de données VSAM, spécifiez `vsam` comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.VSAM.KSDS",
```

```
    "relativePath": "DATA",
    "datasetOrg": {
        "vsam": {

            "encoding": "A",
            "format": "KS",
            "primaryKey": {
                "length": 11,
                "offset": 0
            }
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.VSAM.KSDS.DAT"
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données VSAM.

encoding

Obligatoire. Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (`A`), EBCDIC () et `E` Unknown (). ?

format

Obligatoire. Spécifie le type de jeu de données VSAM et le format d'enregistrement.

- AWSMoteur d'exécution Blu Age : les valeurs possibles sont ESDS (`ES`), KSDS (`KS`) et RRDS (). `RR` Le format d'enregistrement peut être fixe ou variable.
- Moteur d'exécution Micro Focus : les valeurs possibles sont ESDS (`ES`), KSDS (`KS`) et RRDS (). `RR` La définition VSAM inclut le format d'enregistrement, il n'est donc pas nécessaire de le spécifier séparément.

Clé primaire

S'applique uniquement aux ensembles de données VSAM KSDS. Spécifie la clé primaire. Comprend le nom de la clé primaire, le décalage de la clé et la longueur de la clé. Elles `name` `length` sont facultatives `offset` et obligatoires.

Durée de l'enregistrement

Obligatoire. Spécifie la longueur d'un enregistrement. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

- AWSMoteur d'exécution Blu Age : pour VSAM ESDS, KSDS et RRDS, `min` il est facultatif et obligatoire. `max`
- Moteur d'exécution Micro Focus : `min` et `max` sont requis.

Emplacement externe

Obligatoire. Spécifie l'emplacement de la source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé le jeu de données.

Propriétés spécifiques au moteur Blu Age

Le moteur d'exécution AWS Blu Age prend en charge la compression des ensembles de données VSAM. L'exemple suivant montre comment vous pouvez spécifier cette propriété au format JSON.

```
{
    common properties
        ...
        "datasetOrg": {
            "vsam": {
                common properties
                ...
                "compressed": boolean,
                common properties
                ...
            }
        }
}
```

Spécifiez la propriété de compression comme suit :

compression

Facultatif. Spécifie si les index de cet ensemble de données sont stockés sous forme de valeurs compressées. Si votre ensemble de données est volumineux (généralement > 100 Mo), pensez à attribuer à cet indicateur la valeur`true`.

Exemple de format de demande d'ensemble de données pour GDG Base

- AWSMoteur d'exécution Blu Age : non pris en charge.
- Moteur d'exécution Micro Focus : pris en charge.

Si vous importez des ensembles de données de base GDG, spécifiez gdg comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.GDG",
    "relativePath": "DATA",
    "datasetOrg": {
        "gdg": {
            "limit": "3",
            "rollDisposition": "Scratch and No Empty"
        }
    }
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données de base GDG.

limite

Obligatoire. Spécifie le nombre de générations actives, ou de biais. Pour un cluster de base GDG, le maximum est de 255.

Disposition des rouleaux

Facultatif. Spécifie comment gérer les ensembles de données de génération lorsque le maximum est atteint ou dépassé. Les valeurs possibles sont `No Scratch and No Empty`, `Scratch and No Empty`, `Scratch and Empty` ou `No Scratch and Empty`. L'argument par défaut est `Scratch and No Empty`.

Exemple de format de demande d'ensemble de données pour les générations PS ou GDG

- AWSMoteur d'exécution Blu Age : non pris en charge.
- Moteur d'exécution Micro Focus : pris en charge.

Si vous importez des ensembles de données de générations PS ou GDG, spécifiez `ps` comme. `datasetOrg` Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PS.FB",
    "relativePath": "DATA",
    "datasetOrg": {
        "ps": {
            "format": "FB",
            "encoding": "A"
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.PS.LSEQ"
}
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données des générations PS ou GDG.

format

Obligatoire. Spécifie le format des enregistrements de l'ensemble de données. Les valeurs possibles sont `FFA`,`FB`,`FBA`,`FBM`,`FBS`,`FM`,`FS`,`LSEQ`,`U`,`V`,`VA`,`VB`,`VBA`,`VBM`,`VBS`,`VM`, et`VS`.

encoding

Obligatoire. Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (`A`), EBCDIC () et `E` Unknown () `?`

Durée de l'enregistrement

Obligatoire. Spécifie la longueur d'un enregistrement. Vous devez spécifier à la fois la longueur minimale (`min`) et maximale (`max`) de l'enregistrement. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

Emplacement externe

Obligatoire. Spécifie l'emplacement de la source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé le jeu de données.

Exemple de format de demande d'ensemble de données pour PO

Si vous importez des ensembles de données de bons de commande, spécifiez po commedatasetOrg. Votre JSON doit ressembler à l'exemple suivant :

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PO.PROC",
    "relativePath": "DATA",
    "datasetOrg": {
        "po": {
            "format": "LSEQ",
            "encoding": "A",
            "memberFileExtensions": ["PRC"]
        }
    },
    "recordLength": {
        "min": 80,
        "max": 80
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/source/proc/"
}
}
```

Les propriétés suivantes sont prises en charge pour les ensembles de données PO.

format

Obligatoire. Spécifie le format des enregistrements de l'ensemble de données. Les valeurs possibles sont FFA,FB,FBA,FBM,FBS,FM,FS,LSEQ,U,V,VA,VB,VBA,VBM,VBS,VM, etVS.

encoding

Obligatoire. Spécifie le codage du jeu de caractères du jeu de données. Les valeurs possibles sont ASCII (A), EBCDIC () et E Unknown (). ?

memberFileExtensions

Obligatoire. Spécifie un tableau contenant une ou plusieurs extensions de nom de fichier, ce qui vous permet de spécifier les fichiers à inclure en tant que membre PDS.

Durée de l'enregistrement

Facultatif. Spécifie la longueur d'un enregistrement. La longueur minimale (`min`) et maximale (`max`) de l'enregistrement est facultative. Pour les formats d'enregistrement de longueur fixe, ces valeurs doivent correspondre.

Emplacement externe

Obligatoire. Spécifie l'emplacement de la source : c'est-à-dire le compartiment Amazon S3 dans lequel vous avez chargé le jeu de données.

> **Note**
>
> L'implémentation actuelle du moteur d'exécution Micro Focus ajoute des entrées PDS sous forme d'ensembles de données dynamiques.

Environnements d'exécution gérés dans le cadre de la modernisation des AWS mainframes

Si vous débutez dans le domaine de la modernisation des AWS mainframes, consultez les rubriques suivantes pour commencer :

- Qu'est-ce que la modernisation AWS du mainframe ?
- Configuration de la modernisation AWS du mainframe
- Commencer à moderniser le AWS mainframe
- Tutoriel : Managed Runtime pour AWS Blu Age
- Tutoriel : environnement d'exécution géré pour Micro Focus

Dans AWS Mainframe Modernization, un environnement d'exécution est une combinaison nommée de ressources de AWS calcul, d'un moteur d'exécution et des détails de configuration que vous spécifiez. L'environnement d'exécution héberge une ou plusieurs applications. Dans le cadre de la modernisation AWS du mainframe, les applications contiennent des charges de travail du mainframe migrées. Vous pouvez choisir le moteur d'exécution pour les environnements que vous créez. Choisissez AWS Blu Age si vous utilisez le modèle de refactorisation automatique et Micro Focus si vous utilisez le modèle de replateforme. Vous pouvez également choisir la quantité de ressources de calcul adaptée à votre application et éventuellement associer du stockage aux environnements d'exécution. AWS La modernisation du mainframe permet à Amazon d'utiliser les CloudWatch métriques et la journalisation pour vous permettre de surveiller votre environnement d'exécution.

Rubriques

- Création d'un environnement d'exécution pour la modernisation du AWS mainframe
- Mettre à jour un environnement d'exécution de modernisation du AWS mainframe
- Arrêter un environnement d'exécution de modernisation du AWS mainframe
- Redémarrer un environnement d'exécution de modernisation du AWS mainframe
- Supprimer un environnement d'exécution de modernisation du AWS mainframe

Création d'un environnement d'exécution pour la modernisation du AWS mainframe

Utilisez la console de modernisation du AWS mainframe pour créer un environnement de modernisation AWS du mainframe.

Ces instructions supposent que vous avez effectué les étapes décrites dansConfiguration de la modernisation AWS du mainframe.

Création d'un environnement d'exécution

Pour créer un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez créer l'environnement.
3. Sur la page Environnements, choisissez Create environment.
4. Sur la page Spécifier les informations de base, fournissez les informations suivantes :
    a. Dans la section Nom et description, entrez le nom de l'environnement.
    b. (Facultatif) Dans le champ Description de l'environnement, entrez une description de l'environnement. Cette description peut vous aider, ainsi que les autres utilisateurs, à identifier l'objectif de l'environnement d'exécution.
    c. Dans la section des options du moteur, choisissez Blu Age pour le refactoring automatique ou Micro Focus pour le replatforming.
    d. Choisissez une version pour le moteur que vous avez sélectionné.
    e. (Facultatif) Dans la section Balises, choisissez Ajouter une nouvelle balise pour ajouter une ou plusieurs balises d'environnement à votre environnement. Une balise d'environnement est une étiquette d'attribut personnalisée qui vous aide à organiser et à gérer vos AWS ressources.
    f. Choisissez Suivant.
5. Sur la page Spécifier les configurations, fournissez les informations suivantes :
    a. Dans la section Disponibilité, choisissez Environnement d'exécution autonome ou Cluster à haute disponibilité.

Le modèle de disponibilité détermine le niveau de disponibilité de votre application lors de son exécution. Le mode autonome convient parfaitement à des fins de développement. La haute disponibilité concerne les applications qui doivent être disponibles à tout moment.

b. Dans Ressources, choisissez un type d'instance et la capacité souhaitée.

Ces ressources sont les instances Amazon EC2 gérées par AWS Mainframe Modernization qui hébergeront votre environnement d'exécution. Les environnements d'exécution autonomes offrent deux choix de type d'instance et n'autorisent qu'une seule instance. Les environnements d'exécution à haute disponibilité offrent deux choix de type d'instance et autorisent jusqu'à deux instances.

Pour plus d'informations, consultez la section Types d'instances Amazon EC2 et contactez un spécialiste du AWS mainframe pour obtenir des conseils.

6. Dans la section Sécurité et réseau, procédez comme suit :

a. Si vous souhaitez que les applications soient accessibles au public, choisissez Autoriser les applications déployées dans cet environnement à être accessibles au public.

b. Choisissez un Virtual Private Cloud (VPC).

c. Si vous utilisez le modèle de haute disponibilité, choisissez deux sous-réseaux ou plus. Si vous utilisez le modèle autonome avec le moteur AWS Blu Age, choisissez deux sous-réseaux ou plus. Si vous utilisez le modèle autonome avec le moteur Micro Focus, vous pouvez spécifier un sous-réseau.

d. Choisissez un groupe de sécurité pour le VPC que vous avez sélectionné.

> **Note**
>
> AWSLa modernisation du mainframe crée un Network Load Balancer qui vous permet de distribuer les connexions à votre environnement d'exécution. Assurez-vous que les règles entrantes de votre groupe de sécurité autorisent l'accès depuis une adresse IP au port que vous avez spécifié dans la `listener` propriété de la définition de l'application. Pour plus d'informations, consultez la section Enregistrer des cibles dans le guide de l'utilisateur pour les équilibreurs de charge réseau.

e. Dans le champ Clé KMS, choisissez Personnaliser les paramètres de chiffrement si vous souhaitez utiliser un système géré par le clientAWS KMS key. Pour plus d'informations,

consultez Chiffrement des données interrompu pour le service de modernisation des AWS mainframes.

> **Note**
>
> Par défaut, AWS Mainframe Modernization chiffre vos données avec des données AWS KMS key que AWS Mainframe Modernization possède et gère pour vous. Cependant, vous pouvez choisir d'utiliser un service géré par le clientAWS KMS key.

f. (Facultatif) Choisissez un nom AWS KMS key d'utilisateur ou un Amazon Resource Name (ARN). Vous pouvez également choisir Create an AWS KMS key pour accéder à la AWS KMS console et en créer un nouveauAWS KMS key.

g. Choisissez Suivant.

7. (Facultatif) Sur la page Joindre un stockage, choisissez un ou plusieurs systèmes de fichiers Amazon EFS ou Amazon FSx, puis choisissez Next.

8. Dans la section Fenêtre de maintenance, choisissez le moment où vous souhaitez appliquer les modifications en attente à l'environnement.

   - Si vous choisissez Aucune préférence, AWS Mainframe Modernization choisit une fenêtre de maintenance optimisée pour vous.
   - Si vous souhaitez spécifier une fenêtre de maintenance particulière, choisissez Sélectionner une nouvelle fenêtre de maintenance. Choisissez ensuite un jour de la semaine, une heure de début et une durée pour la fenêtre de maintenance.

   Pour plus d'informations sur la fenêtre de maintenance, consultezAWSFenêtre de maintenance de la modernisation du mainframe.

   Choisissez Suivant.

9. Sur la page Réviser et créer, passez en revue les informations que vous avez saisies, puis choisissez Créer un environnement.

Mettre à jour un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour mettre à jour un environnement d'exécution AWS Mainframe Modernization. Vous pouvez mettre à jour la version secondaire du moteur d'exécution ou le type d'instance qui héberge l'environnement d'exécution. Vous pouvez choisir d'appliquer les mises à jour immédiatement ou pendant la période de maintenance préférée.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Mettre à jour un environnement d'exécution

Pour mettre à jour un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez mettre à jour a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez mettre à jour.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.
5. Effectuez une ou plusieurs des modifications suivantes :

   - Dans la section Options du moteur, choisissez la version du moteur que vous souhaitez.
   - Dans la section Ressources, choisissez le type d'instance que vous souhaitez.
   - Dans la section Fenêtre de maintenance, choisissez le jour, l'heure et la durée souhaités.

   > **Note**
   >
   > Les seules modifications que vous pouvez choisir d'appliquer pendant la période de maintenance concernent la version du moteur. Vous devez appliquer immédiatement toutes les autres modifications.

6. Choisissez Suivant.
7. Dans Quand appliquer ces modifications, choisissez Immédiatement ou Pendant la fenêtre de maintenance suivante. Choisissez ensuite l'environnement de mise à jour.

Si vous choisissez Immédiatement, un message s'affiche lorsque la mise à jour de l'environnement est terminée.

AWSFenêtre de maintenance de la modernisation du mainframe

Chaque environnement d'exécution dispose d'une fenêtre de maintenance hebdomadaire d'une heure. Toutes les modifications du système sont appliquées pendant cette période. La fenêtre de maintenance vous permet de contrôler le moment où les modifications et les correctifs logiciels sont appliqués. Si un événement de maintenance est prévu pour une semaine donnée, il commence pendant cette fenêtre de maintenance d'une heure. La plupart des événements de maintenance se terminent également pendant la fenêtre de maintenance d'une heure, bien que les événements de maintenance plus importants puissent prendre plus d'une heure.

La fenêtre de maintenance d'une heure est sélectionnée au hasard parmi une tranche de 8 heures par région. Si vous ne spécifiez pas de fenêtre de maintenance lorsque vous créez un environnement d'exécution, AWS Mainframe Modernization attribue une fenêtre de maintenance d'une heure un jour de la semaine sélectionné au hasard.

AWSLa modernisation du mainframe consomme certaines des ressources de votre instance d'environnement pendant que la maintenance est en cours. Il est possible que vous observiez un effet minime sur les performances ou des interruptions dans les applications pendant la maintenance.

Le tableau suivant indique les plages horaires par défaut lorsque des fenêtres de maintenance sont attribuées à chaque région.

Nom de la région	Région	Bloc chronologique
US East (Virginie du Nord)	us-east-1	03:00–11:00 UTC
USA Ouest (Oregon)	us-west-2	06:00–14:00 UTC
Asia Pacific (Mumbai)	ap-south-1	06:00–14:00 UTC
Asie-Pacifique (Singapour)	ap-southeast-1	14:00–22:00 UTC
Asia Pacific (Sydney)	ap-southeast-2	12:00–20:00 UTC
Asia Pacific (Tokyo)	ap-northeast-1	13:00–21:00 UTC

Nom de la région	Région	Bloc chronologique
Canada (Central)	ca-central-1	03:00–11:00 UTC
Europe (Francfort)	eu-central-1	21:00–05:00 UTC
Europe (Ireland)	eu-west-1	22:00–06:00 UTC
Europe (London)	eu-west-2	22:00–06:00 UTC
Europe (Paris)	eu-west-3	23:59–07:29 UTC
Amérique du Sud (São Paulo)	sa-east-1	00:00–08:00 UTC

Arrêter un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour arrêter un environnement d'exécution AWS Mainframe Modernization. Lorsque vous arrêtez un environnement, les déploiements d'applications en cours sont conservés et vous ne serez pas facturé pour l'environnement tant que celui-ci n'est pas redémarré.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Arrêter un environnement d'exécution

Si vous devez arrêter un environnement d'exécution AWS Mainframe Modernization, vous devez suivre des étapes similaires à celles décrites dans la section relative à l'environnement de mise à jour.

Utilisez la console AWS Mainframe Modernization pour arrêter un environnement d'exécution AWS Mainframe Modernization. Lorsque vous arrêtez un environnement, les déploiements d'applications en cours sont conservés et vous ne serez pas facturé pour l'environnement tant que celui-ci n'est pas redémarré.

Arrêter un environnement d'exécution

Pour arrêter un environnement d'exécution de modernisation AWS du mainframe, vous devez suivre des étapes similaires à celles décrites dans la section relative à l'environnement de mise à jour.

> **Note**
>
> Vous devez arrêter toutes les applications avant d'arrêter l'environnement.

Pour arrêter un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez arrêter a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez arrêter.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.
5. Sur la page Modifier l'environnement, recherchez la section Ressources et mettez à jour la capacité souhaitée à zéro.

   > **Note**
   >
   > Pour arrêter un environnement, vous pouvez uniquement choisir de l'arrêter immédiatement.

6. Choisissez Suivant.
7. Dans Quand appliquer ces modifications, sélectionnez Immédiatement. Choisissez ensuite l'environnement de mise à jour.

Un message s'affiche lorsque la capacité de l'environnement est mise à jour.

Redémarrer un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour redémarrer un environnement d'exécution AWS Mainframe Modernization. Lorsque vous redémarrez un environnement d'exécution, la facturation de l'environnement reprend.

Redémarrer un environnement d'exécution

Pour redémarrer un environnement d'exécution AWS Mainframe Modernization, vous devez suivre des étapes similaires à celles décrites dans la section relative à l'environnement d'arrêt.

Pour redémarrer un environnement d'exécution

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez redémarrer a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez redémarrer.
4. Sur la page de détails de l'environnement, choisissez Actions, puis Modifier l'environnement.

   > **Note**
   >
   > La capacité souhaitée pour un environnement autonome ne peut être mise à jour qu'à 1. Pour redémarrer un environnement d'exécution, vous pouvez uniquement choisir de le redémarrer immédiatement.

5. Sur la page Modifier l'environnement, recherchez la section Ressources et mettez à jour la capacité souhaitée de zéro à la capacité requise.
6. Choisissez Suivant.
7. Dans Quand appliquer ces modifications, sélectionnez Immédiatement. Choisissez ensuite l'environnement de mise à jour.

Un message s'affiche lorsque la capacité de l'environnement est mise à jour et que l'environnement est redémarré.

Supprimer un environnement d'exécution de modernisation du AWS mainframe

Utilisez la console AWS Mainframe Modernization pour supprimer un environnement d'exécution AWS Mainframe Modernization.

Ces instructions supposent que vous avez déjà réalisé les étapes de Configuration de la modernisation AWS du mainframe.

Supprimer un environnement d'exécution

Si vous devez supprimer un environnement d'exécution AWS Mainframe Modernization, assurez-vous d'abord de supprimer toutes les applications déployées de l'environnement. Vous ne pouvez pas supprimer un environnement d'exécution dans lequel des applications sont déployées.

Pour supprimer un d'environnement

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle l'environnement que vous souhaitez supprimer a été créé.
3. Sur la page Environnements, choisissez l'environnement que vous souhaitez supprimer, puis sélectionnez Actions et Supprimer l'environnement.
4. Dans la fenêtre Supprimer l'environnement, entrez `delete` pour confirmer que vous souhaitez supprimer l'environnement d'exécution, puis choisissez Supprimer.

Tests d'applications dans le cadre de la modernisation des AWS mainframes

AWS Les tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

AWS Les tests d'applications de modernisation du mainframe fournissent des tests d'équivalence fonctionnelle automatisés pour vos projets de migration.

Rubriques

Qu'est-ce que le test des applications de modernisation des AWS mainframes ?

AWSLes tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

Les tests ont un impact significatif sur les projets de migration. Cela peut prendre jusqu'à 70 % du temps et des efforts de votre projet de migration, de modernisation ou d'augmentation. AWS Les

tests d'applications, une fonctionnalité de la modernisation du AWS mainframe, fournissent des tests d'équivalence fonctionnelle automatisés pour vos applications migrées. Les tests d'équivalence fonctionnelle vous aident à valider que vos applications sur le AWS Cloud sont équivalentes à celles de votre mainframe. AWS Les tests d'applications comparent automatiquement les modifications apportées aux ensembles de données, aux enregistrements de base de données et aux écrans 3270 en ligne entre votre ordinateur central et. AWS En outre, les tests d'applications permettent des tests reproductibles, ce qui vous permet d'exécuter vos scénarios de test de nombreuses fois au fur et à mesure que vous mettez à jour l'architecture cible, que vous résolvez des problèmes et que vous progressez vers une application entièrement migrée. Après la migration, vous pouvez continuer à utiliser les tests d'applications pour les tests de régression, afin de vous assurer que les mises à jour des moteurs d'exécution ou d'autres composants n'entraînent pas de régressions. Les tests d'applications sont rentables : les environnements de test cibles sont créés à l'aide des CloudFormation modèles fournis par l'utilisateur, en s'appuyant sur les concepts d'infrastructure en tant que code (IaC). Les tests d'applications accélèrent les projets de migration en utilisant l'élasticité du cloud. Vous pouvez exécuter des scénarios de test indépendants sur autant d'environnements parallèles que nécessaire, réduisant ainsi les délais de test.

Rubriques

Utilisez-vous les tests d'applications pour la première fois ?

Si vous utilisez les tests d'applications pour la première fois, nous vous recommandons de commencer par lire les sections suivantes :

- Concepts de test d'applications
- Tutoriel : Configuration CardDemo

Avantages des tests d'applications

Les tests d'applications offrent plusieurs avantages pour vous aider dans votre processus de migration :

- Tester l'accélération, l'agilité et la flexibilité
- Concepts de test « Enregistrer une fois sur mainframe, rejouer plusieurs fois dans AWS »
- Création iAc d'environnements cibles à l'aide de modèles fournis par l'utilisateur CloudFormation
- Hauts degrés de répétabilité des tests
- Conçu pour le cloud, dans un souci d'évolutivité et d'élasticité
- Tests à grande échelle avec haut degré d'automatisation
- Cost efficiency (Rentabilité)

Intégration à AWS CloudFormation

Les tests d'applications utilisent l'infrastructure sous forme de code avecAWS CloudFormation. Ce choix de conception simplifie et améliore votre expérience de test. AWS CloudFormationvous donne l'autonomie et l'indépendance nécessaires pour définir l'infrastructure la mieux adaptée à vos besoins. Vous pouvez sélectionner ou définir de nombreux paramètres (taille de l'instance, instance RDS, groupe de sécurité optimal) indépendamment. Vous pouvez ajouter des ressources, telles qu'une file d'attente Amazon SQS, dont vous avez besoin pour que votre application fonctionne correctement dans des conditions de test.

Dans les AWS CloudFormation modèles fournis au téléchargement, vous remarquerez certaines caractéristiques communes :

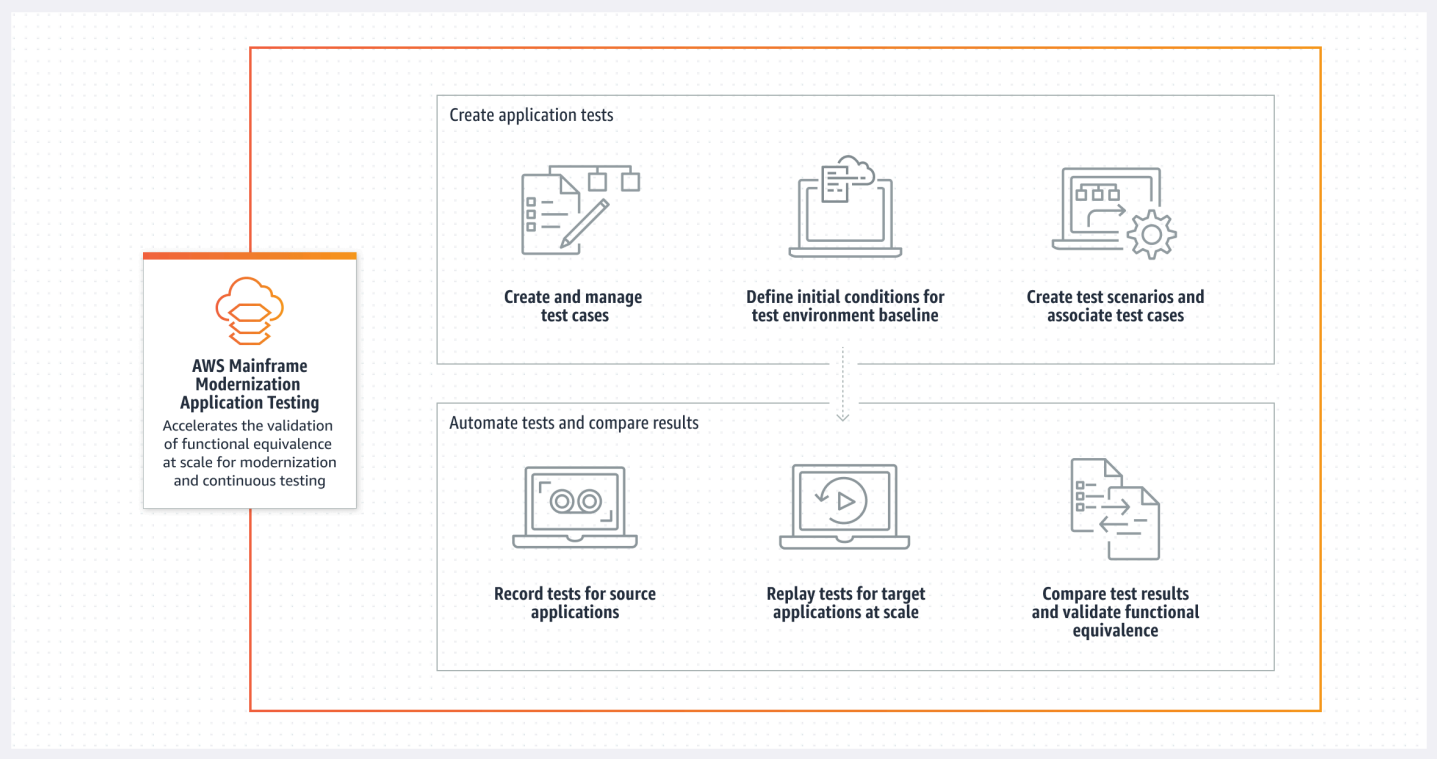
- Les tests d'applications créent une pile totalement isolée, comprenant un environnement d'exécution et une application de modernisation du AWS mainframe, dotés de leurs propres définitions de réseau et de sécurité. Cette pile isolée apporte de la résilience, car les autres acteurs de la même pile Compte AWS ne peuvent pas interférer avec les activités de test. Cela permet également d'éviter les situations dans lesquelles les opérateurs système modifient le VPC ou le groupe de sécurité par défaut, ce qui peut entraîner l'échec des activités de test.
- Le groupe de sécurité vous permet également de contrôler l'accès externe aux ressources utilisées lors des tests. Par exemple, une base de données peut contenir des données confidentielles.
- L'isolation complète empêche les autres acteurs partageant le VPC d'espionner le trafic.

- Elle améliore les performances. Par exemple, la communication entre l'application AWS Mainframe Modernization créée par le modèle et sa base de données Amazon RDS s'effectue sur un réseau distinct (un VPC privé), ce qui évite aux autres acteurs de ralentir le trafic.

Nous vous recommandons d'implémenter également ces fonctionnalités dans les AWS CloudFormation modèles que vous créez.

Comment fonctionnent les tests d'applications

La figure suivante donne un aperçu du fonctionnement des tests d'équivalence fonctionnelle dans le cadre des tests d'applications.



- - Vous pouvez transférer les données d'entrée de la source à l'AWSaide du transfert de fichiers de modernisation du AWS mainframe ou de vos outils préférés pour le transfert de données du mainframe.
- Vous exécutez la même logique métier à la fois sur la source et sur la cible.
- Les tests d'applications comparent automatiquement les données de sortie (ensembles de données, modifications de base de données relationnelle, écrans 3270 et interactions utilisateur) à la fois de la source et de la cible. Après avoir exécuté votre scénario de test sur le mainframe, vous capturez les données de sortie, vous les transférez versAWS, puis vous rejouez le scénario de

test sur la cible. Application Testing compare automatiquement les données de sortie du test AWS avec les données de sortie de la source. Vous pouvez voir en un coup d'œil quels enregistrements sont identiques, équivalents, différents ou manquants. En outre, vous pouvez définir des règles d'équivalence afin que les enregistrements qui ne sont pas identiques mais qui ont la même signification commerciale soient considérés comme équivalents.

Le flux de travail que vous suivez dans le cadre des tests d'applications comprend les étapes suivantes :

1. Créez des scénarios de test. Les cas de test constituent la plus petite unité d'actions de test. Lorsque vous créez un scénario de test, vous identifiez également les types de données à comparer qui représentent le mieux l'équivalence fonctionnelle entre la source et la cible.
2. Créez des scénarios de test. Les scénarios de test regroupent les cas de test connexes dans une séquence spécifique pour l'exécution.
3. Créez les conditions initiales. Les conditions initiales expliquent comment enregistrer un test sur le mainframe et comment l'utiliser AWS CloudFormation pour créer automatiquement le même état surAWS.
4. Enregistrez sur la source et rejouez sur la cible. Capturez les ensembles de données d'entrée et de sortie sur le mainframe et téléchargez-les sur. AWS Rejouez ensuite le scénario de test surAWS.
5. Comparez les ensembles de données source et cible. Les tests d'applications comparent automatiquement les ensembles de données de sortie provenant de la source et de la cible, afin que vous puissiez voir en un coup d'œil ce qui est correct et ce qui ne l'est pas.

L'action finale d'un scénario de test et l'objectif de l'ensemble du processus sont d'identifier les écarts entre les essais source et cible. Les tests d'applications comparent la version source et la version cible pour les données capturées sur tous les canaux d'interaction pendant le test. Il compare également les états finaux des données pertinentes (tels que définis dans les scénarios de test).

Services connexes

Les tests d'applications sont une fonctionnalité de la modernisation des AWS mainframes. Il utilise également l'infrastructure comme code AWS CloudFormation pour garantir la répétabilité, l'automatisation et la rentabilité des tests. Pour plus d'informations, consultez :

- Modernisation du mainframe AWS

- AWS CloudFormation

Accès aux tests d'applications

Vous pouvez accéder aux tests d'applications depuis la console de modernisation du AWS mainframe en choisissant Application Testing dans le menu de navigation de gauche.

Tarification des tests d'applications

Les tarifs des tests d'applications sont disponibles sur le site AWS Mainframe Modernization Pricing.

AWS Concepts de test des applications de modernisation du mainframe

> AWS Les tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

AWS Les tests d'applications utilisent des termes que d'autres services de test ou progiciels peuvent utiliser avec une signification légèrement différente. Les sections suivantes expliquent comment les tests d'applications de modernisation des AWS mainframes utilisent cette terminologie.

Rubriques

- Cas de test
- Scénario de test
- Projet de test
- État initial
- Enregistrer (capturer)
- Relire
- Compare
- Comparaison de bases de données
- Comparaisons de jeux de
- État de la comparaison

- Règles d'équivalence
- Comparaison des ensembles de données sur l'état final
- Comparaisons de bases de données sur l'état
- Equivalence fonctionnelle (FE)
- Comparaisons d'écrans 3270 en ligne
- Enregistrement
- Rejouer les données
- Données de référence
- Enregistrez, rejouez et comparez
- Différences
- Équivalences
- Application source
- Application cible

Cas de test

Un scénario de test est l'unité d'action la plus atomique de votre flux de travail de test. Généralement, un scénario de test est utilisé pour représenter une unité indépendante de logique métier qui modifie les données. Des comparaisons seront effectuées pour chaque cas de test. Les scénarios de test sont ajoutés à un scénario de test. Les scénarios de test contiennent des métadonnées sur les artefacts de données (ensembles de données, bases de données) que le scénario de test modifie et sur les fonctions métier déclenchées lors de l'exécution du scénario de test : tâches par lots, boîtes de dialogue interactives 3270, etc. Par exemple, les noms et les pages de code des ensembles de données.

Données d'entrée → Scénario de test → Données de sortie

Les cas de test peuvent être en ligne ou par lots :

- Les cas de test en ligne sont des cas de test dans lesquels l'utilisateur exécute des boîtes de dialogue d'écran interactives (3270) pour lire, modifier ou produire de nouvelles données commerciales (bases de données et/ou enregistrements d'ensembles de données).
- Les cas de test par lots sont des cas de test nécessitant la soumission d'un lot pour lire, traiter et modifier ou produire de nouvelles données commerciales (ensembles de données et/ou enregistrements de base de données).

Scénario de test

Les scénarios de test sont une série de cas de test exécutés dans un ordre séquentiel, un par un. La rediffusion se fait au niveau d'un scénario de test. Tous les scénarios de test du scénario de test sont exécutés dans l'environnement de test cible lorsqu'un scénario de test est rejoué. S'il existe des différences après avoir comparé les artefacts des tests de référence et de rediffusion, les différences seront affichées au niveau du scénario de test.

Par exemple, le scénario de test A :

Cas de test 1, cas de test 2, cas de test 3, etc.

Projet de test

Les projets de test représentent un ensemble de scénarios de test permettant d'atteindre le jalon de test souhaité. Par exemple, la migration d'une application spécifique peut être considérée comme un projet de test unique. Le regroupement des scénarios de test en projets de test permet aux responsables des tests de suivre l'état des projets de test, y compris les tests réussis ou échoués.

État initial

Une condition initiale contient un ensemble de ressources (calcul, banque de données, etc.) que vous devez créer, ainsi que des données d'application que vous devez restaurer sur ces ressources créées avant de pouvoir exécuter des scénarios de test. Cela crée la base de référence de l'environnement de test cible. Il vous permet de fournir un AWS CloudFormation modèle. Vous utilisez le modèle pour créer l'environnement de test cible et, éventuellement, l'extrait DDL de la base de données source si votre scénario de test modifie les enregistrements de base de données. Chaque scénario de test sera associé à une condition initiale. Une condition initiale peut être associée à plusieurs scénarios de test. Pour garantir la répétabilité et la cohérence des résultats et pour éviter les faux positifs dus à des données déjà modifiées, vous devez rétablir les conditions initiales avant l'exécution de chaque scénario de test.

Pour les scénarios de test contenant des cas de test modifiant les enregistrements de base de données, la condition initiale fait également référence à une exportation DDL des schémas et des tables de la base de données source.

Enregistrer (capturer)

Les enregistrements sont effectués au niveau d'un scénario de test. Pendant l'enregistrement, vous devez fournir un emplacement Amazon S3 contenant les artefacts, les ensembles de données et

les journaux CDC de la base de données relationnelle provenant du mainframe source à comparer. Elles seront considérées comme des données de référence provenant de l'ordinateur central source. Pendant la rediffusion, les données de rediffusion générées seront comparées aux données de référence enregistrées pour garantir l'équivalence des applications.

Relire

Les rediffusions sont effectuées au niveau d'un scénario de test. Pendant la rediffusion, les tests d'applications de modernisation du AWS mainframe utilisent le CloudFormation script référencé dans la condition initiale associée pour créer l'environnement de test cible et exécuter l'application. Les ensembles de données et les enregistrements de base de données modifiés pendant la réexécution sont capturés et comparés aux données de référence du mainframe. Généralement, vous enregistrez une fois sur le mainframe, puis vous le rejouez plusieurs fois, jusqu'à ce que l'équivalence fonctionnelle soit atteinte.

Compare

Les comparaisons sont effectuées automatiquement une fois la rediffusion terminée avec succès. Lors des comparaisons, les données référencées que vous avez téléchargées et capturées pendant la phase d'enregistrement sont comparées aux données de rediffusion générées pendant la phase de rediffusion. Les comparaisons sont effectuées séparément au niveau d'un scénario de test individuel pour les ensembles de données, les enregistrements de base de données et les écrans en ligne.

Comparaison de bases de données

Les tests d'applications utilisent une fonctionnalité de mise en correspondance de l'état de progression lors de la comparaison des modifications des enregistrements de base de données entre les applications source et cible. La correspondance par état et progression compare les différences entre chaque instruction INSERT, UPDATE et DELETE exécutée, contrairement à la comparaison des lignes du tableau à la fin du processus. La mise en correspondance de l'état d'avancement est plus efficace que les alternatives, car elle permet des comparaisons plus rapides et plus précises en comparant uniquement les données modifiées et en détectant les erreurs auto-corrigées dans le flux de transactions. En utilisant la technologie CDC (Changed Data Capture), les tests d'applications peuvent détecter les modifications de base de données de relations individuelles et les comparer entre la source et la cible.

Les modifications de la base de données relationnelle sont générées sur la source et la cible par le code de l'application testée à l'aide d'instructions DML (Data Modification Language) telles que

SQL INSERT, UPDATE ou DELETE, mais également indirectement lorsque l'application utilise des procédures stockées, ou lorsque des déclencheurs de base de données sont définis sur certaines tables, ou lorsque CASCADE DELETE est utilisé pour garantir l'intégrité référentielle, déclenchant automatiquement des suppressions supplémentaires.

Comparaisons de jeux de

Les tests d'applications comparent automatiquement les ensembles de données de référence et de réexécution produits sur les systèmes source (enregistrement) et cible (rediffusion).

Pour comparer des ensembles de données :

1. Commencez avec les mêmes données d'entrée (ensembles de données, base de données) à la fois sur la source et sur la cible.
2. Exécutez vos scénarios de test sur le système source (mainframe).
3. Capturez les ensembles de données produits et chargez-les dans un compartiment Amazon S3. Vous pouvez transférer des ensembles de données d'entrée de la source vers des AWS journaux, des écrans et des ensembles de données du CDC.
4. Spécifiez l'emplacement du compartiment Amazon S3 dans lequel les ensembles de données du mainframe ont été téléchargés lorsque vous avez enregistré le scénario de test.

Une fois la réexécution terminée, Application Testing compare automatiquement les ensembles de données de référence en sortie et les ensembles de données cibles, en indiquant si les enregistrements sont identiques, équivalents, différents ou manquants. Par exemple, les champs de date relatifs au moment de l'exécution de la charge de travail (jour +1, fin du mois en cours, etc.) sont automatiquement considérés comme équivalents. En outre, vous pouvez éventuellement définir des règles d'équivalence afin que les enregistrements qui ne sont pas identiques aient toujours la même signification commerciale et soient marqués comme équivalents.

État de la comparaison

Les tests d'applications utilisent les états de comparaison suivants : IDENTIQUE, ÉQUIVALENT et DIFFÉRENT.

IDENTIQUE

Les données source et cible sont exactement les mêmes.

ÉQUIVALENT

Les données source et cible contiennent de fausses différences considérées comme des équivalences, telles que des dates ou des horodatages qui n'affectent pas l'équivalence fonctionnelle lorsqu'elles sont relatives au moment de l'exécution de la charge de travail. Vous pouvez définir des règles d'équivalence pour identifier ces différences. Lorsque tous les scénarios de test rejoués par rapport à leurs scénarios de test de référence affichent le statut IDENTIQUE ou ÉQUIVALENT, votre scénario de test prouve l'équivalence fonctionnelle.

DIFFÉRENT

Les données source et cible présentent des différences, telles qu'un nombre différent d'enregistrements dans un ensemble de données ou des valeurs différentes dans le même enregistrement.

Règles d'équivalence

Ensemble de règles permettant d'identifier les fausses différences qui peuvent être considérées comme des résultats équivalents. Les tests d'équivalence fonctionnelle hors ligne (OFET) entraînent inévitablement des différences pour certains résultats entre les systèmes source et cible. Par exemple, les horodatages des mises à jour sont conçus différemment. Les règles d'équivalence expliquent comment ajuster ces différences et éviter les faux positifs au moment de la comparaison. Par exemple, si une date correspond à une durée d'exécution de plus de 2 jours dans une colonne de données donnée, la règle d'équivalence la décrit et accepte une durée sur le système cible correspondant à une durée d'exécution de plus de 2 jours au lieu d'une valeur strictement égale à la même colonne dans l'enregistrement de référence.

Comparaison des ensembles de données sur l'état final

État final des ensembles de données créés ou modifiés, y compris toutes les modifications ou mises à jour apportées aux ensembles de données par rapport à leur état initial. Pour les ensembles de données, Application Testing examine les enregistrements contenus dans ces ensembles de données à la fin de l'exécution d'un scénario de test et compare les résultats.

Comparaisons de bases de données sur l'état

Comparaisons des modifications apportées aux enregistrements de base de données sous la forme d'une séquence d'instructions DML (Delete, Update, Insert) individuelles. Les tests d'applications

comparent les modifications individuelles (insertion, mise à jour ou suppression d'une ligne de table) entre la base de données source et la base de données cible, et identifient les différences pour chaque modification individuelle. Par exemple, une instruction INSERT individuelle peut être utilisée pour insérer dans une table une ligne avec des valeurs différentes dans la base de données source par rapport à la base de données cible.

Equivalence fonctionnelle (FE)

Deux systèmes sont considérés comme fonctionnellement équivalents s'ils produisent les mêmes résultats sur toutes les opérations observables, avec les mêmes données d'entrée. Par exemple, deux applications sont considérées comme fonctionnellement équivalentes si les mêmes données d'entrée produisent des données de sortie identiques (par le biais d'écrans, de modifications d'ensembles de données ou de modifications de base de données).

Comparaisons d'écrans 3270 en ligne

Compare la sortie des écrans 3270 du mainframe avec la sortie des écrans Web des applications modernisées lorsque le système cible s'exécute sous le runtime AWS Blu Age dans le. AWS Cloud Il compare également la sortie des écrans 3270 du mainframe à celle des 3270 écrans de l'application réhébergée lorsque le système cible s'exécute sous le moteur d'exécution Micro Focus dans le. AWS Cloud

Enregistrement

Action consistant à restaurer un état connu des données, puis à capturer ou à enregistrer les données de référence d'un scénario de test de référence (pour un ou plusieurs cas de test de manière séquentielle) sur un système source.

Rejouer les données

Les données de rediffusion sont utilisées pour décrire les données générées par la réexécution d'un scénario de test dans l'environnement de test cible. Par exemple, les données de rediffusion sont générées lorsqu'un scénario de test est exécuté sur une application de service de modernisation AWS du mainframe. Les données de rediffusion sont ensuite comparées aux données de référence capturées à partir de la source. Chaque fois que vous rejouez la charge de travail dans l'environnement cible, une nouvelle génération de données de rediffusion est générée.

Données de référence

Les données de référence sont utilisées pour décrire les données capturées sur le mainframe source. Il s'agit de la référence à laquelle les données générées par le replay (cible) seront comparées. En général, pour chaque enregistrement du mainframe qui crée des données de référence, de nombreuses rediffusions sont effectuées. Cela est dû au fait que les utilisateurs capturent généralement l'état correct de l'application sur le mainframe et rejouent les scénarios de test sur l'application modernisée cible pour valider l'équivalence. Si des bogues sont détectés, ils sont corrigés et les scénarios de test sont rejoués. Souvent, plusieurs cycles de rediffusion, de correction de bogues et de rediffusion pour valider l'occurrence. C'est ce qu'on appelle le paradigme des tests « capture une fois, rejouer plusieurs fois ».

Enregistrez, rejouez et comparez

Les tests d'applications se déroulent en trois étapes :

- Enregistrement : capture les données référencées créées sur le mainframe pour chaque scénario de test d'un scénario de test. Il peut s'agir de 3 270 écrans en ligne, d'ensembles de données et d'enregistrements de base de données.
  - Pour les écrans 3270 en ligne, vous devez utiliser l'émulateur de terminal Blu Insights pour capturer votre charge de travail source. Pour plus d'informations, consultez la documentation Blu Insights.
  - Pour les ensembles de données, vous devrez capturer les ensembles de données produits par chaque scénario de test sur le mainframe à l'aide d'outils courants, tels que le FTP ou le service de transfert de jeux de données intégré à la modernisation du AWS mainframe.
  - Pour les modifications de base de données, vous utilisez la documentation AWS Mainframe Modernization Data Replication with Precisely pour capturer et générer des journaux CDC contenant les modifications.
- Rediffusion : le scénario de test est rejoué dans l'environnement cible. Tous les cas de test spécifiés dans le scénario de test exécuté. Les types de données spécifiques créés par les scénarios de test individuels, tels que les ensembles de données, les modifications de bases de données relationnelles ou les écrans 3270, seront capturés automatiquement. Ces données sont connues sous le nom de données de rediffusion et seront comparées aux données de référence capturées pendant la phase d'enregistrement.

> **Note**
>
> Les modifications apportées à la base de données relationnelle nécessiteront des options de configuration spécifiques au DMS dans votre modèle de condition initial. CloudFormation

- Comparaison : la source des tests, les données de référence et les données de rediffusion cibles sont comparées, et les résultats vous seront présentés sous forme de données identiques, différentes, équivalentes ou manquantes.

Différences

Indique que des différences ont été détectées entre les ensembles de données de référence et de réexécution par comparaison des données. Par exemple, un champ d'un écran 3270 en ligne qui affiche des valeurs différentes du point de vue de la logique métier entre le mainframe source et l'application modernisée cible sera considéré comme une différence. Un autre exemple est un enregistrement dans un ensemble de données qui n'est pas identique entre les applications source et cible.

Équivalences

Les enregistrements équivalents sont des enregistrements différents entre les ensembles de données de référence et de réexécution, mais ne doivent pas être traités comme différents du point de vue de la logique métier. Par exemple, un enregistrement contenant l'horodatage de la date de production de l'ensemble de données (heure d'exécution de la charge de travail). À l'aide de règles d'équivalence personnalisables, vous pouvez demander à Application Testing de traiter une telle différence faussement positive comme une équivalence, même si elle indique des valeurs différentes entre les données de référence et les données de rediffusion.

Application source

L'application mainframe source à laquelle la comparaison doit être effectuée.

Application cible

Application nouvelle ou modifiée sur laquelle les tests sont effectués et qui sera comparée à l'application source afin de détecter tout défaut et d'obtenir une équivalence fonctionnelle entre les applications source et cible. L'application cible s'exécute généralement dans le AWS cloud.

Tutoriel : Configuration de l' CardDemo exemple d'application

AWSLes tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

Dans le cadre de ce didacticiel, vous allez créer une AWS CloudFormation pile qui vous aide à configurer l'CardDemo exemple d'application pour le replatforming avec le service géré Micro Focus on AWS Mainframe Modernization et des fonctionnalités telles que les tests d'applications de modernisation des AWS mainframes. Le didacticiel décrit un exemple de AWS CloudFormation modèle que vous pouvez utiliser pour créer la pile. Nous fournissons également un fichier compressé contenant les artefacts d'application nécessaires. L'exemple de modèle fournit une base de données, un environnement d'exécution, une application et un environnement réseau totalement isolé.

Ce modèle crée plusieurs AWS ressources. Ils vous seront facturés si vous créez une pile à partir de ce modèle.

Prérequis

- Téléchargez et décompressez le fichier `IC3-card-demo-zip` et. `datasets_Mainframe_ebcdic.zip` Ces fichiers contiennent les CardDemo exemples et les ensembles de données à utiliser pour les tests d'AWSapplications.
- Créez un compartiment Amazon S3 pour contenir les CardDemo fichiers et autres artefacts. Par exemple, `my-carddemo-bucket`.

Étape 1 : Préparation de la configuration CardDemo

Téléchargez les fichiers CardDemo d'exemple et modifiez le AWS CloudFormation modèle qui créera l' CardDemoapplication.

1. Téléchargez les `IC3-card-demo` dossiers `datasets_Mainframe_ebcdic` et que vous avez décompressés précédemment dans votre bucket.
2. Téléchargez le `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modèle depuis votre bucket. Il se trouve dans le `IC3-card-demo` dossier.
3. Modifiez le `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modèle comme suit :

- Remplacez le `BucketName` paramètre par le nom du compartiment que vous avez défini précédemment, par exemple`my-carddemo-bucket`.
- Remplacez `ImportJsonPath` le fichier par l'emplacement du `mf-carddemo-datasets-import.json` fichier dans votre compartiment. Par exemple, la `s3://my-carddemo-bucket/IC3-card-demo/mf-carddemo-datasets-import.json` mise à jour de cette valeur garantit que la sortie `M2ImportJson` contient la bonne valeur.
- (Facultatif) Adaptez les `InstanceType` paramètres `EngineVersion` et en fonction de vos normes.

> **Note**
>
> Ne modifiez pas les `M2ApplicationId` sorties `M2EnvironmentId` et. Les tests d'applications utilisent ces valeurs pour localiser les ressources avec lesquelles elles interagiront.

Étape 2 : créer toutes les ressources nécessaires

Exécutez votre AWS CloudFormation modèle personnalisé pour créer toutes les ressources dont vous avez besoin pour terminer ce didacticiel avec succès. Ce modèle configure l' CardDemo application afin que vous puissiez l'utiliser lors des tests.

1. Connectez-vous à la AWS CloudFormation console et choisissez Create stack, puis choisissez With new resources (standard).
2. Dans Prérequis - Préparer le modèle, sélectionnez Le modèle est prêt.
3. Dans Spécifier le modèle, choisissez Charger un fichier modèle, puis choisissez Choisir un fichier.
4. Naviguez jusqu'à l'endroit où vous avez téléchargé `aws-m2-math-mf-carddemo.yaml` le fichier, puis choisissez Next.
5. Dans Spécifier les détails de la pile, donnez un nom à la pile afin que vous puissiez la retrouver facilement dans une liste, puis choisissez Suivant.
6. Dans Configurer les options de pile, conservez les valeurs par défaut et choisissez Next.
7. Dans Révision, vérifiez ce qui AWS CloudFormation est créé pour vous, puis choisissez Soumettre.

La création de la pile prend environ 10 AWS CloudFormation à 15 minutes.

> **Note**
>
> Le modèle est configuré pour ajouter un suffixe unique aux noms des ressources qu'il crée. Cela signifie que vous pouvez créer plusieurs instances de ce modèle de pile en parallèle, une fonctionnalité clé pour les tests d'applications qui vous permet d'exécuter plusieurs scénarios de test en même temps.

Étape 3 : Déployer et démarrer l'application

Déployez l' CardDemo application AWS CloudFormation créée pour vous et assurez-vous qu'elle est en cours d'exécution.

1. Ouvrez la console AWS Mainframe Modernization et sélectionnez Applications dans le menu de navigation de gauche.
2. Choisissez l' CardDemo application, qui porte un nom comme`aws-m2-math-mf-carddemo-abc1d2e3`.
3. Choisissez Actions, puis choisissez Déployer l'application.
4. Dans Environnements, choisissez l'environnement d'exécution correspondant à l'application. Le même identifiant unique sera ajouté à la fin du nom. Par exemple, `aws-m2-math-mf-carddemo-abc1d2e3`.
5. Choisissez Deploy (Déployer). Attendez que l'application soit déployée avec succès et qu'elle soit à l'état Prêt.
6. Choisissez l'application, puis sélectionnez Actions et Démarrer l'application. Attendez que l'application soit en cours d'exécution.
7. Sur la page des détails de l'application, copiez le port et le nom d'hôte DNS dont vous avez besoin pour vous connecter à l'application en cours d'exécution.

Étape 4 : Importer les données initiales

Pour utiliser l' CardDemo exemple d'application, vous devez importer un premier ensemble de données. Procédez comme suit.

1. Téléchargez le fichier `mf-carddemo-datasets-import.json`.

2. Modifiez le fichier dans votre éditeur de texte préféré.
3. Localisez le `s3Location` paramètre et mettez à jour la valeur pour qu'elle pointe vers le compartiment Amazon S3 que vous avez créé.
4. Effectuez la même modification pour toutes les occurrences des`s3Location`, puis enregistrez le fichier.
5. Connectez-vous à la console Amazon S3 et accédez au compartiment que vous avez créé précédemment.
6. Téléchargez le `mf-carddemo-datasets-import.json` fichier personnalisé.
7. Ouvrez la console AWS Mainframe Modernization et sélectionnez Applications dans le menu de navigation de gauche.
8. Choisissez l' CardDemo application.
9. Choisissez Ensembles de données, puis Importer.
10. Accédez à l'emplacement dans Amazon S3 où vous avez chargé le fichier JSON personnalisé et choisissez Soumettre.

Cette tâche importe 23 ensembles de données. Pour contrôler le résultat de la tâche d'importation, consultez la console. Lorsque tous les ensembles de données sont correctement importés, connectez-vous à l'application.

> **Note**
>
> Lorsque vous utilisez ce modèle dans le cadre de tests d'applications, la sortie gère `M2ImportJson` automatiquement le processus d'importation.

Étape 5 : Connectez-vous à l' CardDemoapplication

Connectez-vous à l' CardDemo exemple d'application à l'aide de l'émulateur 3270 de votre choix.

- Lorsque l'application est en cours d'exécution, utilisez votre émulateur 3270 pour vous connecter à l'application, en spécifiant le nom d'hôte DNS et le nom de port, si nécessaire.

  Par exemple, si vous utilisez l'émulateur open source c3270, votre commande ressemble à ceci :

```
c3270 -port port-number DNS-hostname
```

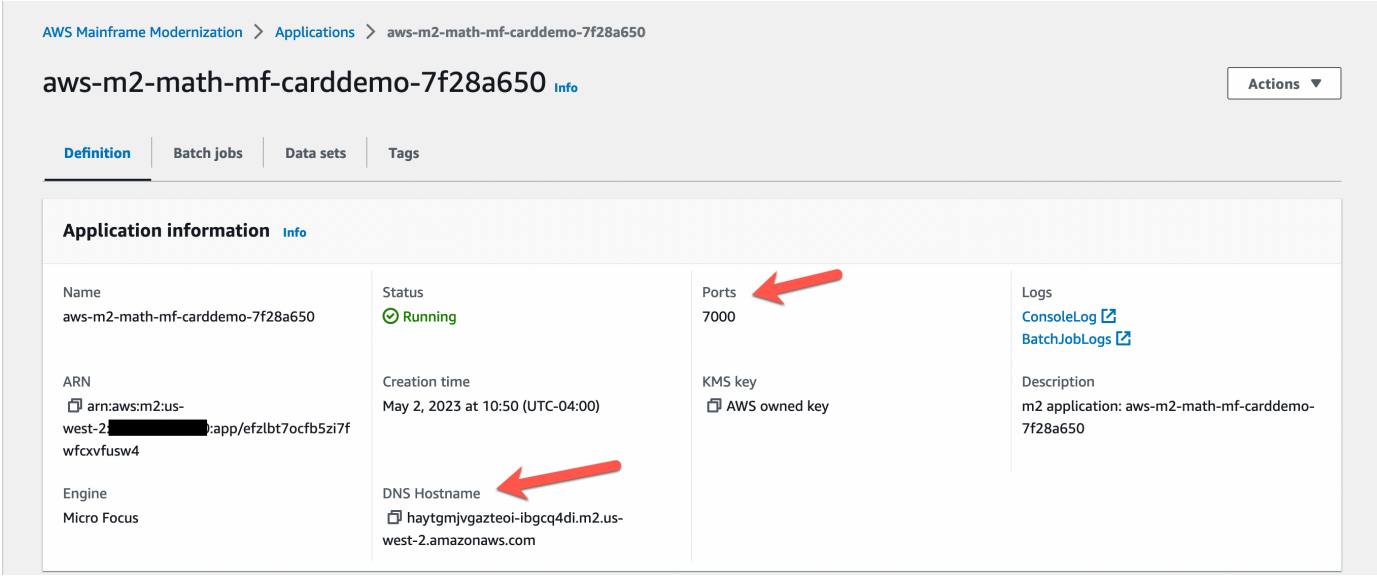
port

Port spécifié sur la page détaillée de l'application. Par exemple, 6000.

Hostname

Le nom d'hôte DNS spécifié sur la page détaillée de l'application.

La figure suivante indique où trouver le port et le nom d'hôte DSN.



Tutoriel : Modernisation AWS du mainframe : test des applications, rediffusion et comparaison de leur utilisation CardDemo pour AWS Blu Age déployé sur Amazon EC2

AWSLes tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

Dans ce didacticiel, vous allez suivre les étapes requises pour rejouer et comparer les charges de travail de test avec l' CardDemo application exécutée sur AWS Blu Age déployée sur Amazon EC2.

Étape 1 : obtenir une image machine Amazon EC2 (AMI) AWS Blu Age Amazon EC2

Suivez les instructions du didacticiel de configuration de AWSAWS Blu Age Runtime (sur Amazon EC2) pour connaître les étapes d'intégration requises pour accéder à l'AMI AWS Blu Age sur Amazon EC2.

Étape 2 : démarrer une instance Amazon EC2 à l'aide de l'AMI AWS Blu Age

1. Configurez vos AWS informations d'identification.
2. Identifiez l'emplacement du fichier binaire AMI Amazon EC2 3.5.0 (version AWS CLI seulement/ Blu Age) dans le compartiment Amazon S3 :

```
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/3.5.0/AMI/
```

> **Note**
>
> La fonctionnalité de test d'application n'est disponible que dans 4 régions de prod (us-east-1, sa-east-1, eu-central-1 et ap-southeast-2).

3. Restaurez l'AMI dans votre compte à l'aide de la commande suivante :

```
aws ec2 create-restore-image-task --object-key 3.5.0/AMI/ami-0182ffe3b9d63925b.bin
 --bucket aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1 --region eu-west-1 --name
 "AWS BLUAGE RUNTIME AMI"
```

> **Note**
>
> Remplacez le nom du fichier bin de l'AMI et la région dans laquelle vous souhaitez créer l'AMI.

4. Après avoir créé une instance Amazon EC2, vous pouvez trouver l'ID d'AMI correct qui a été restauré à partir du compartiment Amazon S3 dans le catalogue d'images Amazon EC2.

> **Note**
>
> Dans ce didacticiel, l'ID de l'AMI est ami-0d0fafcc636fd1e6d, et vous devez remplacer cet identifiant dans les différents fichiers de configuration par celui qui vous a été fourni.

1. Si aws ec2 create-restore-image-task échoue, vérifiez votre version de Python et de la CLI à l'aide de la commande suivante :

```
aws --version
```

> **Note**
>
> La version Python doit être >= 3 et la version CLI doit être >= 2.

2. Si ces versions sont obsolètes, la CLI doit être mise à jour. Pour mettre à jour la CLI :
   a. Suivez les instructions de la section Installer ou mettre à jour la dernière version de l'AWS CLI.
   b. Supprimez la CLI v1 à l'aide de la commande suivante :

```
sudo yum remove awscli
```

   c. Et installez CLI v2 avec les commandes suivantes :

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

   d. Enfin, vérifiez la version de Python et de la CLI avec la commande suivante :

```
aws --version
```

3. Vous pouvez ensuite refaire le aws create-restore-image-task ec2.

Étape 3 : télécharger les fichiers CardDemo dépendants sur S3

Copiez le contenu des dossiers, des bases de données, du système de fichiers et des données utilisateur. Téléchargez et décompressez les CardDemo applications. Ces trois dossiers doivent être copiés dans l'un de vos compartiments appelé your-s3-bucket dans cette documentation.

Étape 4 : Chargement des bases de données et initialisation de l'application CardDemo

Créez une instance Amazon EC2 temporaire que vous utiliserez comme ressource de calcul pour générer les instantanés de base de données requis pour l'application. CardDemo Cette instance EC2 n'exécutera pas l' CardDemo application elle-même, mais générera les instantanés de base de données qui seront utilisés ultérieurement.

Commencez par modifier le CloudFormation modèle fourni nommé « load-and-create-ba - snapshots.yml ». Il s'agit du CloudFormation modèle utilisé pour créer l'instance Amazon EC2 utilisée pour générer les instantanés de base de données.

1. Générez et fournissez la paire de clés EC2 qui sera utilisée pour l'instance EC2. Pour plus d'informations, consultez la section Création de paires de clés.

   Exemple :

   ```
   Ec2KeyPair:
       Description: 'ec2 key pair'
       Default: 'm2-tests-us-west-2'
       Type: String
   ```

2. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier de base de données à l'étape précédente :

   ```
   S3DBScriptsPath:
       Description: 'S3 DB scripts folder path'
       Type: String
       Default: 's3://your-s3-bucket/databases'
   ```

3. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier du système de fichiers de l'étape précédente :

   ```
   S3ApplicationFilesPath:
   ```

```
    Description: 'S3 application files folder path'
    Type: String
    Default: 's3://your-s3-bucket/file-system'
```

4. Spécifiez le chemin Amazon S3 de votre dossier dans lequel vous avez placé le dossier userdata de l'étape précédente :

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://your-s3-bucket/userdata'
```

5. Spécifiez également un chemin Amazon S3 dans lequel vous enregistrerez les fichiers de résultats à utiliser à l'étape suivante.

```
S3SaveProducedFilesPath:
    Description: 'S3 path folder to save produced files'
    Type: String
    Default: 's3://your-s3-bucket/post-produced-files'
```

6. Modifiez l'ID de l'AMI par le bon ID obtenu précédemment dans ce didacticiel à l'aide du modèle suivant :

```
  BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0bd41245734fd20d9'
    Type: String
```

- Vous pouvez éventuellement modifier le nom des trois instantanés qui seront créés lors de l'exécution des bases withCloudFormation de données de chargement. Ils seront visibles dans la CloudFormation pile lors de sa création et seront utilisés ultérieurement dans ce didacticiel. N'oubliez pas de noter les noms utilisés pour les instantanés de base de données.

```
SnapshotPrimary:
    Description: 'Snapshot Name DB BA Primary'
    Type: String
    Default: 'snapshot-primary'

  SnapshotBluesam:
```

```
    Description: 'Snapshot Name DB BA Bluesam'
    Type: String
    Default: 'snapshot-bluesam'

  SnapshotJics:
    Description: 'Snapshot Name DB BA Jics'
    Type: String
    Default: 'snapshot-jics'
```

> **Note**
>
> Dans ce document, nous partons du principe que le nom des instantanés reste cohérent.

7. Exécutez la CLI ou AWS la console CloudFormation avec le bouton Create Stack et l'assistant. À la fin du processus, vous devriez voir apparaître trois instantanés dans la console RDS portant le nom que vous avez choisi suivi d'un identifiant unique. Vous aurez besoin de ces noms à l'étape suivante.

> **Note**
>
> RDS ajoutera des suffixes aux noms des instantanés définis dans le modèle. AWS CloudFormation Assurez-vous d'obtenir le nom complet de l'instantané auprès de RDS avant de passer à l'étape suivante.

Exemple de commande CLI-

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots --
template-url https://your-apptest-bucket.s3.us-west-2.amazonaws.com/load-and-
create-ba-snapshots.yml --capabilities CAPABILITY_NAMED_IAM
```

Vous pouvez également vérifier dans le chemin Amazon S3 que vous avez fourni pour S3 SaveProducedFilesPath que les ensembles de données ont été correctement créés.

Étape 5 : Lancez le moteur d'exécution AWS Blu Age CloudFormation

CloudFormation À utiliser pour exécuter l'instance Amazon EC2 avec l'application CardDemo AWS Blu Age. Vous devez remplacer certaines variables du CloudFormation nom en `m2-with-ba-using-snapshots-https-authentication.yml` éditant le fichier YAML ou en modifiant les valeurs dans la console lors du lancement du CFN.

1. Modifiez le `AllowedVpcEndpointPrincipals` pour spécifier quel compte atteindra le point de terminaison VPC pour accéder au moteur d'exécution AWS Blu Age, à l'aide des commandes suivantes :

```
AllowedVpcEndpointPrincipals:
    Description: 'comma-separated list of IAM users, IAM roles, or AWS accounts'
    Default: 'apptest.amazonaws.com'
    Type: String
```

2. Modifiez la valeur SnapshotPrimaryDb des SnapshotBlusamDb variables et SnapshotJicsDb le nom des instantanés. Obtenez également les noms des instantanés auprès de RDS après leur création à l'étape précédente.

```
SnapshotPrimary:
    Description: 'Snapshot DB cluster for DB Primary'
    Type: String
    Default: 'snapshot-primary87d067b0'

  SnapshotBluesam:
    Description: 'Snapshot DB cluster for DB Bluesam'
    Type: String
    Default: 'snapshot-bluesam87d067b0'

  SnapshotJics:
    Description: 'Snapshot DB cluster for DB Jics'
    Type: String
    Default: 'snapshot-jics87d067b0'
```

> **Note**
>
> RDS ajoutera son propre suffixe aux noms des instantanés.

3. Fournissez votre paire de clés Amazon EC2 pour l'instance EC2 à l'aide de cette commande :

```
Ec2KeyPair:
    Description: 'ec2 key pair'
    Default: 'm2-tests-us-west-2'
    Type: String
```

4. Fournissez l'ID d'AMI que vous avez obtenu lors du processus d'enregistrement de l'AMI pour la variable BaaAmiId, en utilisant :

```
BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0d0fafcc636fd1e6d'
    Type: String
```

5. Indiquez le chemin du dossier Amazon S3 que vous avez utilisé à l'étape précédente pour enregistrer les fichiers produits, à l'aide de la commande suivante :

```
S3ApplicationFilesPath:
    Description: 'bucket name'
    Type: String
    Default: 's3://your-s3-bucket/post-produced-files'
```

6. Enfin, indiquez le chemin du dossier du s3- userdata-folder-path :

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://your-s3-bucket/userdata'
```

- (Facultatif) Vous pouvez activer le mode HTTPS et l'authentification HTTP de base pour Tomcat. Bien que les paramètres par défaut fonctionneraient également.

> **Note**
>
> Par défaut, le mode HTTPS est désactivé et défini sur le mode HTTP dans le paramètre BacHttpsMode:

Par exemple :

```
BacHttpsMode:
    Description: 'http or https for Blue Age Runtime connection mode '
    Default: 'http'
    Type: String
    AllowedValues: [http, https]
```

- (Facultatif) Pour activer le mode HTTPS, vous devez remplacer la valeur par HTTPS et fournir l'ARN de votre certificat ACM en modifiant la valeur de la variable CertArnACM :

```
ACMCertArn:
    Type: String
    Description: 'ACM certificate ARN'
    Default: 'your arn certificate'
```

- (Facultatif) L'authentification de base est désactivée par défaut avec le paramètre WithBacBasicAuthenticationdéfini sur false. Vous pouvez l'activer en définissant la valeur sur true.

```
WithBacBasicAuthentication:
    Description: 'false or true for Blue Age Runtime Basic Authentication '
    Default: false
    Type: String
    AllowedValues: [true, false]
```

7. Lorsque vous avez terminé la configuration, vous pouvez créer la pile à l'aide du CloudFormation modèle modifié.

Étape 6 : Test de l'instance Amazon EC2 de AWS Blu Age

Exécutez manuellement le CloudFormation modèle pour créer l'instance AWS Blu Age Amazon EC2 de l' CardDemo application afin de vous assurer qu'elle démarre sans erreur. Ceci est fait pour vérifier que le CloudFormation modèle et tous les prérequis sont valides, avant d'utiliser le CloudFormation modèle avec la fonctionnalité de test d'application. Vous pouvez ensuite utiliser les tests d'application pour créer automatiquement l'instance Amazon EC2 AWS Blu Age cible pendant la rediffusion et la comparer selon une condition initiale.

1. Exécutez la commande CloudFormation create stack pour créer l'instance Amazon EC2 AWS Blu Age, en fournissant le modèle m2- with-ba-using-snapshots -https-authentication.yml CloudFormation que vous avez modifié à l'étape précédente :

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots —-
template-url https://apptest-ba-demo.s3.us-west-2.amazonaws.com/m2-with-ba-using-
snapshots-https-authentication.yml --capabilities CAPABILITY_NAMED_IAM --region us-
west-2
```

> **Note**
>
> N'oubliez pas de spécifier la bonne région dans laquelle l'AMI AWS Blu Age a été restaurée.

2. Assurez-vous que tout fonctionne correctement en recherchant dans la console l'instance Amazon EC2 en cours d'exécution. Connectez-vous-y à l'aide du gestionnaire de session.
3. Une fois connecté à l'instance Amazon EC2, utilisez les commandes suivantes :

```
sudo su
cd /m2-anywhere/tomcat.gapwalk/velocity/logs
cat catalina.log
```

4. Assurez-vous qu'il n'y a aucune exception ou erreur dans le journal.
5. Vérifiez ensuite que l'application répond à l'aide de cette commande :

```
curl http://localhost:8080/gapwalk-application/
```

Vous verrez le message « L'application Jics est en cours d'exécution ».

Étape 7 : Valider que les étapes précédentes ont été effectuées correctement

Au cours des prochaines étapes, nous testerons les applications de modernisation AWS du mainframe pour rejouer et comparer les ensembles de données créés par l'application. CardDemo Ces étapes dépendent de la réussite de toutes les étapes précédentes de ce didacticiel. Validez les points suivants avant de continuer :

1. Vous avez créé avec succès l'instance AWS Blu Age on Amazon EC2 via le AWS CloudFormation modèle.
2. Le service Tomcat sur le AWS Blu Age sur Amazon EC2 est opérationnel, sans exception.

Lorsque l'instance EC2 est exécutée avec l' CardDemo application, effectuez les étapes suivantes sur la console de test des applications pour effectuer une réexécution et une comparaison des ensembles de données par lots.

Étape 8. Créer une condition initiale

Au cours de cette étape, vous créez une condition initiale en fournissant le CloudFormation modèle que vous avez utilisé pour déployer l' CardDemo application AWS Blu Age sur Amazon EC2.

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le volet de navigation de gauche, choisissez Application Testing.
3. Dans Application Testing, choisissez Create Initial Condition.
4. Utilisez votre copie locale avec le chemin Amazon S3 modifié et les identifiants de capture qui pointent vers vos ressources.

Étape 9 : Création du scénario de test

Au cours de cette étape, vous créez le scénario de test qui sera utilisé pour comparer les ensembles de données créés dans l'application Card Demo.

1. Créez un nouveau scénario de test. Donnez-lui un nom et une description.
2. Spécifiez `CREASTMT.JCL` comme nom JCL.
3. Ajoutez les ensembles de données suivants à la définition du cas de test :

Nom	CCSID	RecordFormat	RecordLength
AWS.M2.CARDDEMO.STATEMNT.PS	« 037 »	FB	80
AWS.M2.CARDDEMO.STATEMENT.HTML	« 037 »	FB	100

> **Note**
>
> Le nom de votre JCL et les détails de votre jeu de données doivent correspondre.

Étape 10 : Création d'un scénario de test

1. Créez un nouveau scénario de test et donnez-lui un nom et une description.
2. Ajoutez le scénario de test que vous avez créé à l'étape précédente à votre scénario de test.
3. Une fois le scénario de test créé, sélectionnez la condition initiale créée à l'étape 1 sur la page de présentation du scénario de test.

Étape 11 : Enregistrez votre scénario de test

Au cours de cette étape, vous devez exécuter des scénarios de test sur la source. Pour ce faire :

1. Téléchargez et exécutez les ensembles de données issus de l'exécution de l'application sur le CardDemo mainframe.
2. Téléchargez le dossier décompressé dans votre compartiment Amazon S3. Ce compartiment Amazon S3 doit se trouver dans la même région que vos autres ressources de test d'applications.

> **Note**
>
> Il doit y avoir deux fichiers dont les noms correspondent aux noms des ensembles de données passés dans le scénario de test précédent.

3. Sur la page de présentation du scénario de test, cliquez sur le bouton Enregistrer.
4. Sur la page d'enregistrement du scénario de test, spécifiez l'emplacement Amazon S3 où vous avez chargé les ensembles de données obtenus à partir du mainframe source.
5. Cliquez sur Soumettre pour démarrer le processus d'enregistrement.

> **Note**
>
> Attendez que l'enregistrement soit terminé avant de procéder à la rediffusion et à la comparaison.

Étape 12 : Rejouer et comparer

Exécutez le scénario de test et les scénarios de test dans l'environnement cible AWS AWS Blu Age on Amazon EC2. Les tests d'applications captureront les ensembles de données produits par rediffusion et les compareront aux ensembles de données de référence enregistrés sur le mainframe.

1. Choisissez Rejouer et comparez. La création de la CloudFormation pile, la comparaison et la suppression de la pile devraient prendre environ trois minutes.

Une fois que tout est terminé, vous devriez avoir des résultats de comparaison avec quelques différences créées intentionnellement dans le cadre de cette démonstration.

AWS Modernisation du mainframe, tests d'applications, ensembles de données pris en charge, pages de code

AWS Les tests d'applications sont en cours de version préliminaire pour la modernisation du AWS mainframe et sont susceptibles d'être modifiés. Nous vous recommandons d'utiliser cette fonctionn alité uniquement avec des données de test et des applications, et non dans des environnements de production.

Utilisez le tableau suivant pour déterminer si l'identifiant de jeu de caractères codé (CCSID) de vos données est pris en charge lors des tests AWS d'applications. Si vos données utilisent un CCSID non pris en charge, nous vous recommandons de le convertir en un CCSID compatible ou de nous contacter pour obtenir de l'aide.

CCSID	Jeu de caractères	Description
37	IBM037, IBM-037, Cp037	Pays hôte : États-Unis, Canada (ESA), Pays-Bas, Portugal, Brésil, Australie, Nouvelle-Zélande
273	IBM273, IBM-273, Cp273	Pays hôte : Autriche, Allemagne
277	IBM277, IBM-277, Cp277	Pays hôte : Danemark, Norvège
278	IBM278, IBM-278, Cp278	Pays hôte : Finlande, Suède
280	IBM280, IBM-280, Cp280	Pays hôte : Italie
284	IBM284, IBM-284, Cp284	Pays hôte : Espagne, Amérique latine (espagnol)
285	IBM285, IBM-285, Cp285	Hôte : Royaume-Uni
297	IBM297, IBM-297, Cp297	Hôte : France
300	IBM-300	JAPAN DB EBCDIC
301	IBM-301	Données du PC : Japan DB
437	IBM437, IBM-437, US-ASCII, Cp437, US-ASCII	Données PC : PC Base USA, de nombreux autres pays
500	IBM500, IBM-500, Cp500	Pays hôte : Belgique, Canada (AS/400), Suisse, International Latin-1
720	IBM-720	MSDOS ARABE
737	IBM-737, X-IBM737	MDOS GREC
775	IBM775, IBM-775	MSDOS BALTIQUE

CCSID	Jeu de caractères	Description
808	IBM-808	Données informatiques : cyrillique, Russie, avec euro
813	ISO-8859-7, ISO8859_7	ISO 8859-7 : Grèce
819	ISO-8859-1, ISO8859_1	ISO 8859-1 : Pays latin-1
833	IBM-833	EBDIC CORÉEN
834	IBM-834, X-IBM834	BASE DE DONNÉES CORÉENNE EBCDIC
835	IBM-835	T-CHINESE DB EBCD
836	IBM-836	EBDIC S-CHINOIS
837	IBM-837	EBDIC S-CHINOIS
850	IBM850, IBM-850, Cp850	Données informatiques : pays latin-1
855	IBM855, IBM-855, Cp855	Données du PC : cyrillique
856	IBM-856, x-IBM856, Cp856	Données du PC : Hébreu
858	IBM00858, IBM-858, Cp858	Données informatiques : pays latin-1, avec euro
859	IBM-859	Données du PC : LATIN-9
860	IBM860, IBM-860	Données du PC : portugais
861	IBM861, IBM-861	Données informatiques : Islande
862	IBM862, IBM-862, Cp862	Données du PC : hébreu (migration)

CCSID	Jeu de caractères	Description
863	IBM863, IBM-863	Données informatiques : Canada
865	IBM865, IBM-865, Cp865	Données informatiques : Den/ Norvège
866	IBM866, IBM-866, Cp866	Données informatiques : cyrillique, Russie
867	IBM-867	Données informatiques : hébreu avec euro
870	IBM870, IBM-870, Cp870	Hébergeur : Latin-2 multilingue
871	IBM871, IBM-871, Cp871	Pays hôte : Islande
874	X-IBM874	Données du PC : thaïlandais
875	IBM-875, X-IBM875, Cp875	Pays hôte : Grèce
897	IBM-897	Données du PC : Japan SB
912	ISO-8859-2, ISO8859_2	ISO 8859-2 : Latin-2 multiling ue
915	ISO-8859-5, ISO8859_5	ISO 8859-5 : cyrillique
916	ISO-8859-8, ISO8859_8	ISO 8859-8 : Hébreu
918	IBM918, IBM-918, Cp918	Hôte : Urdu
920	ISO-8859-9, ISO8859_9	ISO 8859-9 : Latin-5 (ECMA-128, Turquie TS-5881)
921	IBM-921, X-IBM921, Cp921	Données informatiques : Lettonie, Lituanie
922	IBM-922, X-IBM922, Cp922	Données informatiques : Estonie

CCSID	Jeu de caractères	Description
923	ISO-8859-15, Cp923, ISO8859_15_FDIS	ISO 8859-15 : Latin-9
924	IBM-924	ISO 8859-15 : Latin-9
927	IBM-927	Données du PC : T-Chinese
930	IBM-930, X-IBM930, Cp930	Katakana Host : SBCS étendu. Kanji Host : DBCS comprenant 4 370 caractères définis par l'utilisateur
932	IBM-932	Données du PC : Japan Mix
933	IBM-933, x-IBM933, Cp933	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisa teur et 11 172 caractères Hangul complets
935	IBM-935, x-IBM935, Cp935	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisa teur.
937	IBM-937, X-IBM937, Cp937	Hôte : Extended SBCS. Hôte : DBCS comprenant 6204 caractères définis par l'utilisa teur
939	IBM-939, X-IBM939, Cp939	Latin Host : SBCS étendu. Kanji Host : DBCS comprenan t 4 370 caractères définis par l'utilisateur.

CCSID	Jeu de caractères	Description
942	IBM-942, IBM-942C, X-IBM942, X-IBM942c, Cp942, Cp942C	Données du PC : SBCS étendu. Données PC : DBCS comprenant 1880 caractères définis par l'utilisateur
943	IBM-943, IBM-943C, Shift_JIS , Windows-31j, Windows-9 32, X-IBM943, X-IBM943c, Cp943, Cp943C, MS932	Données du PC : SBCS Données PC : DBCS pour environnement ouvert comprenant 1880 caractères IBM définis par l'utilisateur
947	IBM-947	T-CHINESE BIG-5
948	IBM-948, X-IBM948, Cp948	Données du PC : SBCS étendu. Données PC : DBCS comprenant 6204 caractères définis par l'utilisateur
949	IBM-949, IBM-949C, X-IBM949, X-IBM949c, Cp949, Cp949C	Code IBM KS - Données du PC : SBCS. Code IBM KS - Données PC : DBCS comprenant 1880 caractères définis par l'utilisateur
950	Grand 5, IBM-950, X-IBM950, Cp950	Données du PC : SBCS (IBM BIG5). Données PC : DBCS comprenant 13493 CNS, 566 caractères sélectionnés par IBM, 6204 caractères définis par l'utilisateur
951	IBM-951	Données du PC : IBM KS
954	EUC-JP, IBM-954, IBM-954C	G0 : JIS X201 Roman. G1 : EST X208-1990. G1 : JIS X201 Katakana. G1 : JIS X212

CCSID	Jeu de caractères	Description
964	EUC-TW, IBM-964, X-IBM964, Cp964	G0 : ASCII. G1 : plan CNS 11643 1. G1 : plan CNS 11643 2.
970	EUC-KR, X-iBM970, Cp970	G0 : ASCII. G1 : KSC X5601-1989 comprenant 1880 caractères définis par l'utilisa teur
971	IBM-971	EUC CORÉEN
1006	IBM-1006, X-IBM1006, Cp1006	ISO-8 : ourdou
1025	IBM-1025, X-IBM1025, Cp1025	Hébergeur : cyrillique multiling ue
1026	IBM1026, IBM-1026, Cp1026	Hôte : Latin-5 (Turquie)
1027	IBM-1027	JAPAN LATIN EBCD
1041	IBM-1041	Données PC : Japon
1043	IBM-1043	Données du PC : T-Chinese
1046	IBM-1046, IBM-1046S, X-IBM1046	ARABE - PC
1047	IBM1047, IBM-1047	Hôte : Latin-1
1051	hp-roman 8	ÉMULATION HP
1088	IBM-1088	Données PC : Korea KS
1089	ISO-8859-6, ISO8859_6	ISO 8859-6 : arabe
1097	IBM-1097, X-IBM1097, Cp1097	Hôte : Farsi

CCSID	Jeu de caractères	Description
1098	IBM-1098, x-IBM1098, Cp1098	Données du PC : Farsi
1112	IBM-1112, X-IBM1112, Cp1112	Pays hôte : Lettonie, Lituanie
1114	IBM-1114	Données du PC : T-CH SB
1115	IBM-1115	Données du PC : S-CH Go
1122	IBM-1122, X-IBM1122, Cp1122	Pays hôte : Estonie
1123	IBM-1123, X-IBM1123, Cp1123	Hôte : Cyrillic Ukraine
1124	IBM-1124, X-IBM1124, Cp1124	8 bits : cyrillique, biélorusse
1140	IBM01140, IBM-1140, Cp1140	Pays hôte : États-Unis, Canada (ESA), Pays-Bas, Portugal, Brésil, Australie, Nouvelle-Zélande, avec euro
1141	IBM01141, IBM-1141, Cp1141	Pays hôte : Autriche, Allemagne, avec l'euro
1142	IBM01142, IBM-1142, Cp1142	Pays hôte : Danemark, Norvège, avec l'euro
1143	IBM01143, IBM-1143, Cp1143	Pays hôte : Finlande, Suède, avec l'euro
1144	IBM01144, IBM-1144, Cp1144	Pays hôte : Italie, avec euro
1145	IBM01145, IBM-1145, Cp1145	Pays hôte : Espagne, Amérique latine (espagnol), avec l'euro

CCSID	Jeu de caractères	Description
1146	IBM01146, IBM-1146, Cp1146	Pays hôte : Royaume-Uni, avec euro
1147	IBM01147, IBM-1147, Cp1147	Pays hôte : France, avec euro
1148	IBM01148, IBM-1148, Cp1148	Pays hôte : Belgique, Canada (AS/400), Suisse, International Latin-1, avec euro
1149	IBM01149, IBM-1149, Cp1149	Pays hôte : Islande, avec euro
1200	UTF-16BE	Unicode avec jeu de caractères 65535. En l'absence d'une marque d'ordre des octets (BOM), elle est supposée être UTF-16 BE (big-endian).
1202	UTF-16LE	UTF-16 LE avec IBM PUA
1204	UTF-16	UTF-16 avec IBM PUA
1208	UTF-8, UTF-8J, UTF-8	Unicode avec jeu de caractères 65535. UTF-8.
1232	UTF-32BE	UTF-32 BE avec IBM PUA
1234	UTF-32LE	UTF-32 LE avec IBM PUA
1236	UTF-32	UTF-32 avec IBM PUA
1351	IBM-1351	OPEN DU JAPON
1362	IBM-1362	MS-WIN CORÉEN

CCSID	Jeu de caractères	Description
1363	IBM-1363, IBM-1363C, Windows-949, MS949	Données du PC : MS Windows Korean SBCS. Données du PC : MS Windows Koran DBCS, y compris 11172 Hangul complet
1364	IBM-1364	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisateur et 11 172 caractères Hangul complets
1370	IBM-1370	Données PC : SBCS étendu, avec euro. Données PC : DBCS comprenant 6204 caractères définis par l'utilisateur, avec euro
1371	IBM-1371	Hôte : SBCS étendu, avec euro. Hôte : DBCS comprenant 6204 caractères définis par l'utilisateur, avec euro
1375	Big 5-HKSC	Mixed Big-5 Ext pour HKSCS
1380	IBM-1380	Données du PC : S-CH Go
1381	IBM-1381, X-IBM1381, Cp1381	Données du PC : SBCS étendu (IBM Go). Données PC : DBCS (IBM GB) comprenant 31 caractères sélectionnés par IBM, 1880 caractères définis par l'utilisateur

CCSID	Jeu de caractères	Description
1382	IBM-1382	EUC CHINOIS
1383	EUC-CN, GB2312, IBM-1383, X-IBM1383, Cp1383	G0 : ASCII. G1 : ensemble GB 2312-80
1385	IBM-1385	Données du PC : S-CH GBK
1386	GBK, IBM-1386, Windows-936, MS936	Données PC : S-Chinese GBK et T-Chinese IBM BIG-5. Données du PC : S-Chinese GBK
1388	IBM-1388	Hôte : Extended SBCS. Hôte : DBCS comprenant 1880 caractères définis par l'utilisateur
1390	IBM-1390	Katakana Host : SBCS étendu, avec euro. Kanji Host : DBCS comprenant 6205 caractères définis par l'utilisateur
1399	IBM-1399	Latin Host : SBCS étendu, avec euro. Kanji Host : DBCS comprenant 4 370 caractères définis par l'utilisateur, avec euro
5050	JIS0201, JIS0208, JIS0212, JIS0201, JIS0208, JIS0212	G0 : JIS X201 Roman. G1 : EST X208-1990. G1 : JIS X201 Katakana. G1 : JIS X212
5054	ISO-322 JP	TCP JAPONAIS
5346	Windows-1250, Cp1250	MS Windows : Latin-2, version 2 avec euro

CCSID	Jeu de caractères	Description
5347	Windows-1251, Cp1251	MS Windows : cyrillique, version 2 avec euro
5348	Windows-1252, Cp1252	MS Windows : pays Latin-1, version 2 avec euro
5349	Windows-1253, Cp1253	MS Windows : Grèce, version 2 avec euro
5350	Windows-1254, Cp1254	MS Windows : Turquie, version 2 avec euro
5351	Windows-1255, Cp1255	MS Windows : Hébreu, version 2 avec euro
5352	Windows-1256, Windows-1256s, Cp1256	MS Windows : arabe, version 2 avec euro
5353	Windows-1257, Cp1257	MS Windows : Baltic Rim, version 2 avec euro
5354	Windows-1258, Cp1258	MS Windows : vietnamien, version 2 avec euro
5488	GB18030	GB18030, données à 1 octet GB18030, données à 2 octets GB18030, données à 4 octets
9030	IBM-838, Cp838	Hôte : Thai extended SBCS
9066	IBM-874, Cp874	Données du PC : SBCS étendu thaïlandais
9400	CESU-8	CESU-8 avec IBM PUA
25546	ISO-282 KR	TCP CORÉEN
33722	IBM-33722, IBM-33722C	IBM EUCJP

Transfert de fichiers dans le cadre de la AWS modernisation du mainframe

AWS Mainframe Modernization File Transfer vous permet de transférer et de convertir des ensembles de données du mainframe vers Amazon S3 pour des cas d'utilisation liés à la modernisation, à la migration et à l'augmentation du mainframe.

Rubriques

Qu'est-ce que le transfert de fichiers pour la modernisation du mainframe AWS ?

Avec AWS Mainframe Modernization File Transfer, vous pouvez transférer et convertir des ensembles de données et des fichiers à l'aide d'un service entièrement géré afin d'accélérer et de simplifier les cas d'utilisation de la modernisation, de la migration et de l'augmentation vers le service AWS Mainframe Modernization et Amazon S3.

Rubriques

Avantages du transfert de fichiers lié à la modernisation du mainframe AWS

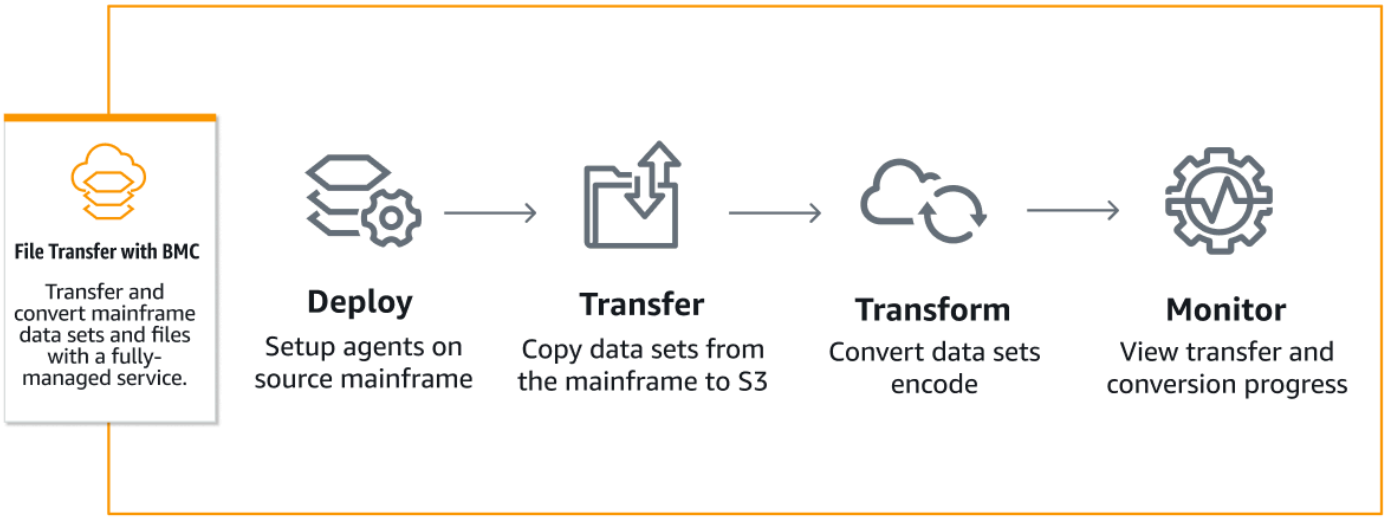
AWS Mainframe Modernization File Transfer vous aide à transférer des ensembles de données du mainframe vers Amazon S3. Certains avantages incluent :

- Découverte des ensembles de données et artefacts sources du mainframe
- Transferts automatisés et conversion des ensembles de données

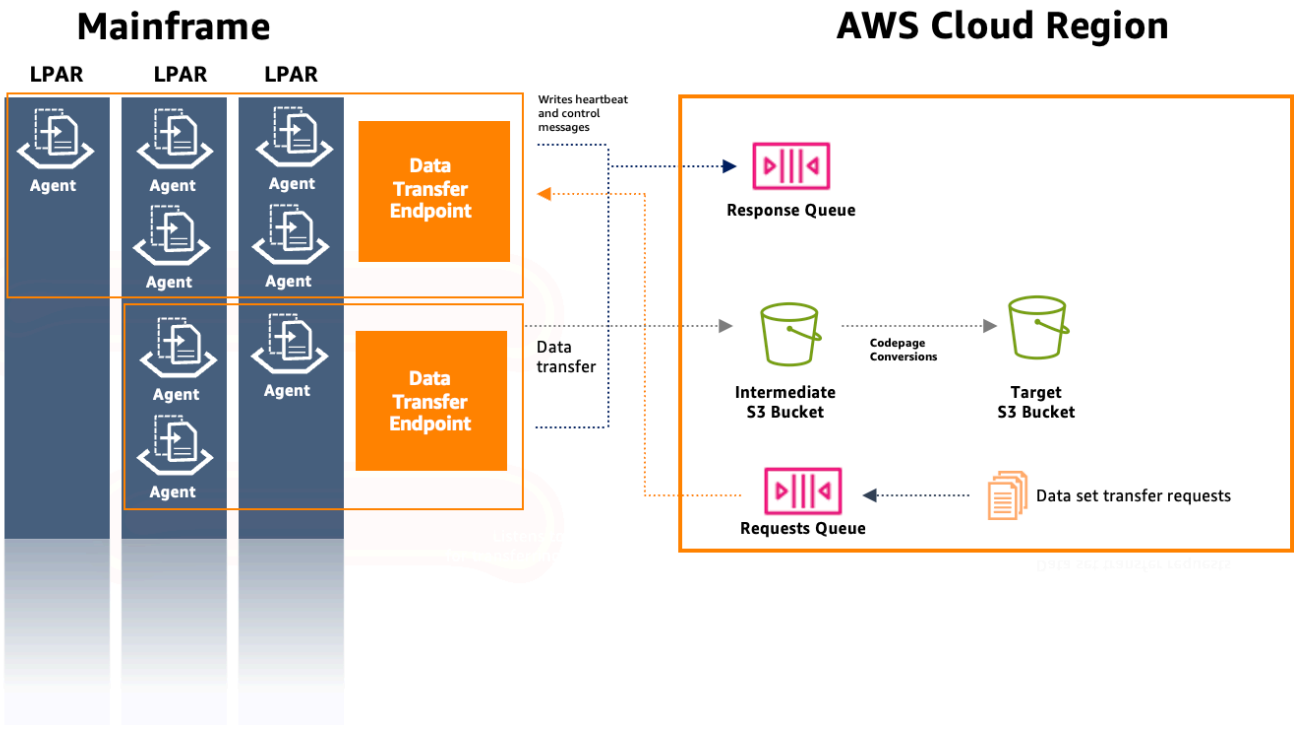
- Évolutivité, efficacité et rapidité pour accélérer les transferts de jeux de données vers AWS

Comment fonctionne le transfert de fichiers pour la modernisation du mainframe AWS

La figure suivante donne un aperçu du fonctionnement conceptuel du transfert de fichiers pour la modernisation du mainframe AWS.



La figure suivante présente une présentation architecturale de la fonctionnalité de transfert de fichiers de modernisation du mainframe AWS.



Installation d'un agent de transfert de fichiers

Suivez ce guide step-by-step pour vérifier les conditions préalables à l'installation d'un agent sur le mainframe source et pour configurer l'agent.

Rubriques

- Étape 1 : Connectez-vous à l'ISPF
- Étape 2 : Allouer un ensemble de données pour z/FS
- Étape 3 : Formater le jeu de données au format z/FS
- Étape 4 : définir le système de fichiers pour z/OS
- Étape 5 : monter le système de fichiers
- Étape 6 : vérifier le support
- Étape 7 : Entrez les OMV
- Étape 8 : définir la variable d'environnement du répertoire d'installation de l'agent
- Étape 9 : définir la variable d'environnement du répertoire de travail
- Étape 10 : Création du répertoire de travail

- Étape 11 : Copiez le package tar AWS Mainframe Modernization dans le répertoire de travail sous z/OS
- Étape 12 : prenez l'utilisateur root
- Configurer les autorisations et le STC
- Création d'un utilisateur IAM avec des informations d'accès à long terme
- Créez un rôle IAM que l'agent devra assumer
- Configuration de l'agent

Étape 1 : Connectez-vous à l'ISPF

Connectez-vous à votre session ISPF (Interactive System Productivity Facility). Cela se fait généralement via un émulateur de terminal 3270.

Étape 2 : Allouer un ensemble de données pour z/FS

À l'aide de l'utilitaire d'ensemble de données de l'ISPF, allouez un nouveau jeu de données pour z/FS. Cela se fait généralement dans l'utilitaire de liste d'ensembles de données à l'étape 1.

1. Passez à l'option 3.4 (Utilitaires -> Ensemble de données).
2. Appuyez sur la touche « C » pour créer un nouvel ensemble de données.
3. Entrez le nom de l'ensemble de données (par exemple, « YourHLQ.M2Agent.ZFS »).
4. Spécifiez le type de jeu de données « Grand format » avec une taille principale de 1 000 cylindres et une taille secondaire de 200.
5. Définissez l'organisation du jeu de données (DSORG) sur PS et le format d'enregistrement (RECFM) sur « U » (non défini).
6. Terminez le processus de création.

Étape 3 : Formater le jeu de données au format z/FS

Après avoir créé le jeu de données, formatez-le en tant que système de fichiers z/FS.

Pour ce faire, vous pouvez utiliser le langage JCL (Job Control Language) suivant :

```
//FORMAT  EXEC PGM=IOEAGFMT,PARM='AGGRNAME(yourhlq.M2AGENT.ZFS),FORMAT,AGGRSIZE(1200)'
```

```
//SYSPRINT DD  SYSOUT=A
```

Soumettez cette tâche et vérifiez si elle s'est terminée avec succès.

Étape 4 : définir le système de fichiers pour z/OS

Définissez z/FS à z/OS à l'aide d'une `DEF FILESYSTEM` commande ou d'un traitement par lots. Utilisez la commande suivante :

```
DEF FILESYSTEM('yourhlq.M2AGENT.ZFS') TYPE(ZFS) MODE(R/W) MOUNTPOINT('/usr/lpp/aws/m2-agent')
```

> **Note**
>
> Cette étape nécessite une autorité au niveau du système.

Étape 5 : monter le système de fichiers

Pour monter le système de fichiers, utilisez la `MOUNT` commande. Vous pouvez monter le système de fichiers en ligne de commande, dans ISPF ou par lots.

Par exemple :

```
MOUNT FILESYSTEM('yourhlq.M2AGENT.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/usr/lpp/aws/m2-agent')
```

Étape 6 : vérifier le support

Vérifiez que le système de fichiers est correctement monté à l'aide de `D OMVS,A` la commande ou en vérifiant dans Unix System Service (USS).

Étape 7 : Entrez les OMV

Utilisez la commande suivante pour saisir les OMV :

```
TSO OMVS
```

Étape 8 : définir la variable d'environnement du répertoire d'installation de l'agent

Utilisez la commande suivante pour définir l'environnement du répertoire d'installation de l'agent :

```
export AGENT_DIR=/usr/lpp/aws/m2-agent
```

Étape 9 : définir la variable d'environnement du répertoire de travail

Utilisez la commande suivante pour définir la variable d'environnement du répertoire de travail :

```
export WORK_DIR=$AGENT_DIR/tmp
```

Étape 10 : Création du répertoire de travail

Utilisez la commande suivante pour définir l'environnement du répertoire de travail :

```
mkdir -p $WORK_DIR
```

Étape 11 : Copiez le package tar AWS Mainframe Modernization dans le répertoire de travail sous z/OS

Lorsque vous utilisez le protocole FTP ou toute autre méthode de transfert, assurez-vous que le fichier tar est transféré en mode binaire.

Étape 12 : prenez l'utilisateur root

Utilisez la commande suivante pour prendre le nom d'utilisateur root :

```
su
```

Pour terminer l'installation de l'agent, procédez comme suit :

> **Note**
>
> Vous devez prendre le nom d'utilisateur root avant de poursuivre ces étapes.

1. Définissez la variable d'environnement m2-agent version sur la version actuellement installée à l'aide de la commande suivante :

```
export M2_AGENT_VERSION=1.0.0
```

2. Extrayez le package tar de l'agent à l'aide de la commande suivante :

```
tar -xpf m2-agent-package-$M2_AGENT_VERSION.tar -C $AGENT_DIR
```

3. Créez un lien `current-version` symbolique vers le répertoire d'installation actuel de l'agent à l'aide de la commande suivante :

```
ln -s $AGENT_DIR/m2-agent-v$M2_AGENT_VERSION $AGENT_DIR/current-version
```

4. Mettez à jour et soumettez `CPY#PDS` pour créer les ensembles de données de l'agent de transfert de fichiers.

> **Note**
>
> JCL utilise le HLQ SYS2.AWS.M2.

Pour créer l'agent de transfert de fichiers, définissez les lignes de paramètres 000006-000012. Mettez également à jour les trois variables symboliques `HLQVOLSER`, `AGNTPATH` qui seront utilisées ultérieurement dans la JCL :

```
oedit $AGENT_DIR/current-version/installation/CPY#PDS
submit $AGENT_DIR/current-version/installation/CPY#PDS
```

> **Note**
>
> Cette JCL est conçue pour configurer certains aspects de l'installation de l'agent sur le mainframe. Il alloue les ensembles de données nécessaires, puis copie des fichiers spécifiques du système de fichiers Unix vers ces ensembles de données.

Configurer les autorisations et le STC

1. Mettez à jour et soumettez l'un des `SYS2.AWS.M2.SAMPLIB(SEC#RACF)` (pour configurer les autorisations RACF) ou `SYS2.AWS.M2.SAMPLIB(SEC#TSS)` (pour configurer les autorisations TSS) conformément à leurs instructions. Ces membres ont été créés à l'`CPY#PDS`étape précédente.
2. Mettez à jour l'exportation PWD dans la JCL `SYS2.AWS.M2.SAMPLIB(M2AGENT)` STC, si le chemin de répertoire par défaut de l'agent de transfert de fichiers (`/usr/lpp/aws/m2-agent`) a été modifié.
3. Mettez à jour et copiez la `SYS2.AWS.M2.SAMPLIB(M2AGENT)` JCL dans. `SYS1.PROCLIB`
4. Ajoutez `SYS2.AWS.M2.LOADLIB` à la liste APF à l'aide de la commande suivante :

```
SETPROG APF ADD DSNAME(SYS2.AWS.M2.LOADLIB) SMS
```

5. Définissez le groupe et le propriétaire de l'agent `logs` et `diag` des dossiers sur l'utilisateur/groupe de l'agent (M2USER/M2GROUP). Utilisez la commande suivante :

```
chown -R M2USER:M2GROUP $AGENT_DIR/current-version/logs
chown -R M2USER:M2GROUP $AGENT_DIR/current-version/diag
```

Création d'un utilisateur IAM avec des informations d'accès à long terme

Vous devez créer l'agent mainframe requis par un utilisateur IAM pour utiliser les files d'attente de réponse et de demande et pour enregistrer des ensembles de données dans des compartiments Amazon S3.

Lors de la création de cet utilisateur :

1. Choisissez Joindre les politiques directement dans les options d'autorisations.
2. Une fois l'utilisateur créé, ouvrez l'onglet Informations d'identification de sécurité et créez une clé d'accès. Pour plus d'informations sur la création d'une clé d'accès IAM, consultez la section Gestion des clés d'accès pour les utilisateurs IAM.
3. Dans la section Clé d'accès, sélectionnez Autre lorsque vous êtes invité à saisir Cas d'utilisation.

> **Note**
>
> Enregistrez la clé d'accès et la clé d'accès secrète affichées sur la dernière page de l'assistant de création de clé d'accès, avant de choisir OK. Ces clés sont utilisées pour configurer l'agent mainframe.

> **Note**
>
> Enregistrez l'ARN de l'utilisateur IAM utilisé pour établir une relation de confiance avec un rôle IAM.

Créez un rôle IAM que l'agent devra assumer

Vous créez un nouveau rôle IAM avec une politique de confiance personnalisée pour le type d'entité de confiance. La politique utilisera le modèle suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DataTransferEndpointAgentSqsReceive",
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:ReceiveMessage"
            ],
            "Resource": "<data-transfer-endpoint-request-queue-arn>"
        },
        {
            "Sid": "DataTransferEndpointS3",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "<data-transfer-endpoint-intermediate-bucket-arn>/*"
        },
        {
            "Sid": "DataTransferEndpointAgentSqsSend",
            "Effect": "Allow",
            "Action": "sqs:SendMessage",
            "Resource": "<data-transfer-endpoint-response-queue-arn>"
```

```
        },
        {
            "Sid": "DataTransferEndpointAgentKmsDecrypt",
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "<kms-key-id>"
        }
    ]
}
```

Où :

- `request-queue-arn`et `response-queue-arn` sont l'ARN de la file d'attente Amazon SQS de requête créée lors de l'initialisation du point de terminaison de transfert de données.
- `transfer-bucket-arn`est l'ARN du bucket de transfert créé précédemment.

> **Note**
>
> Vous pouvez rechercher toutes ces valeurs à l'aide de la console AWS.

> **Note**
>
> Enregistrez le nom du rôle, que vous utiliserez ultérieurement pour configurer l'agent mainframe.

Configuration de l'agent

Pour configurer l'agent de transfert de fichiers :

1. Accédez à `$AGENT_DIR/current-version/config`.
2. Modifiez le fichier de configuration de l'agent `appication.properties` pour ajouter une configuration d'environnement à l'aide de la commande suivante :

   ```
   oedit $AGENT_DIR/current-version/config/application.properties
   ```

   Par exemple :

```
agent.environments[0].account-id=<AWS_ACCOUNT_ID>
agent.environments[0].agent-role-name=<AWS_IAM_ROLE_NAME>
agent.environments[0].access-key-id=<AWS_IAM_ROLE_ACCESS_KEY>
agent.environments[0].secret-access-id=<AWS_IAM_ROLE_SECRET_KEY>
agent.environments[0].bucket-name=<AWS_S3_BUCKET_NAME>
agent.environments[0].environment-name=<AWS_REGION>
agent.environments[0].region=<AWS_REGION>
```

Où :

- `AWS_ACCOUNT_ID`est l'identifiant du compte client.
- `AWS_IAM_ROLE_NAME`est le nom du rôle IAM créé dans lethe section called "Créez un rôle IAM que l'agent devra assumer".
- `AWS_IAM_ROLE_ACCESS_KEY`est la clé d'accès de l'utilisateur IAM créé dansthe section called "Création d'un utilisateur IAM avec des informations d'accès à long terme".
- `AWS_IAM_ROLE_SECRET_KEY`est la clé secrète d'accès de l'utilisateur IAM créé dansthe section called "Création d'un utilisateur IAM avec des informations d'accès à long terme".
- `AWS_S3_BUCKET_NAME`est le nom du compartiment de transfert créé avec le point de terminaison du transfert de données.
- `AWS_REGION`est la région dans laquelle vous configurez l'agent de transfert de fichiers.

> **Note**
>
> Il peut y avoir plusieurs sections de ce type, à condition que l'index entre crochets — `[0]` — soit incrémenté pour chacune d'entre elles.

Vous devez redémarrer l'agent pour que les modifications soient prises en compte.

Prérequis

1. Lorsqu'un paramètre est ajouté ou supprimé, l'agent doit être arrêté et démarré. Démarrez l'agent de transfert de fichiers à l'aide de la commande suivante dans la CLI :

```
/S M2AGENT
```

Pour arrêter l'agent M2, utilisez la commande suivante dans la CLI :

```
/P M2AGENT
```

2. Vous pouvez transférer l'agent de transfert de fichiers vers plusieurs régions et comptes en AWS définissant plusieurs environnements.

```
#Region 1
agent.environments[0].account-id=AWS_ACCOUNT_ID
agent.environments[0].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[0].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[0].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[0].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[0].environment-name=AWS_REGION
agent.environments[0].region=AWS_REGION

#Region 2
agent.environments[1].account-id=AWS_ACCOUNT_ID
agent.environments[1].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[1].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[1].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[1].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[1].environment-name=AWS_REGION
agent.environments[1].region=AWS_REGION
```

Points de terminaison de transfert de données

Les points de terminaison de transfert de données permettent une haute disponibilité, une évolutivité et une gestion rationalisée des agents sur le mainframe source. Des agents individuels sont installés sur les LPAR du mainframe et peuvent être regroupés dans un point de terminaison de transfert de données. Lorsqu'une demande est faite pour transférer un ensemble de données, un agent du point de terminaison du transfert de données gère le transfert. Pour initier des transferts de données, au moins un agent du point de terminaison de transfert de données doit être en ligne.

Cette procédure suppose que vous avez terminé les étapes de configuration de Configuration de la modernisation AWS du mainframe l'agent de transfert de fichiers sur le mainframe source.

Création d'un point de terminaison de transfert de données

Pour créer des points de terminaison de transfert de données pour le transfert de fichiers, vous devez suivre ces étapes dans la console de modernisation du AWS mainframe.

Pour créer un point de terminaison de transfert de données

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Points de terminaison de transfert de données, sous Transfert de fichiers, choisissez Créer un point de terminaison de transfert de données.
4. Sur la page Conditions requises pour le terminal de transfert de données, lisez toutes les instructions pour vous assurer que vous avez effectué ces étapes. Une fois confirmé, choisissez Next.
5. Sur la page Configurer le point de terminaison de transfert de données, ajoutez des informations de base pour votre point de terminaison de transfert de données.

    1. Dans la section des informations de base, entrez le nom, la description et la clé KMS de votre point de terminaison de transfert de données. Pour plus d'informations sur les clés KMS, reportez-vous à la section Créer des clés.

    > **Note**
    >
    > Le nom du point de terminaison du transfert de données doit correspondre au nom défini lors de la configuration de l'agent de transfert de fichiers sur le mainframe source.

    > **Note**
    >
    > Vous devez ajouter la politique basée sur les ressources suivante pour KMS afin que le service de modernisation du AWS mainframe puisse lire et utiliser ces clés pour le chiffrement/déchiffrement :

```
{
  "Sid" : "Enable AWS M2 Permissions",
```

```
  "Effect" : "Allow",
  "Principal" : {
      "Service" : "m2.amazonaws.com"
           },
   "Action" : [
       "kms:Encrypt",
       "kms:Decrypt"
              ],
   "Resource" : "*"
}
```

2. Spécifiez l'emplacement S3 pour les données intermédiaires, qui est l'emplacement S3 intermédiaire où sont stockés les ensembles de données transférés depuis le mainframe.

> **Note**
>
> Il est recommandé de créer un nouveau compartiment Amazon S3 pour vos tâches de transfert. Pour plus d'informations, consultez la section Création d'un bucket. Vous pouvez également parcourir vos compartiments Amazon S3 existants en choisissant l'option Parcourir S3.

3. Après avoir saisi les champs obligatoires, choisissez Next.

6. Sur la page Vérifier et créer un point de terminaison de transfert de données, vérifiez si vous avez rempli les conditions requises et consultez les informations de base. Une fois confirmé, choisissez Créer et connecter.
7. Sur la page Vérifier la connectivité, vous verrez tous les agents s'afficher avec leurs identifiants et leurs pulsations cardiaques une fois la connectivité établie. Une fois la connectivité établie, vous recevez un message « Le point de terminaison de transfert de données a été créé avec succès ».

> **Note**
>
> Si le message « La connectivité de l'agent n'a pas pu être établie » s'affiche, revenez à l'étape 4 pour vous assurer que vous avez rempli toutes les conditions requises.

8. Choisissez Finish (Terminer).

Vous serez redirigé vers la page de présentation des points de terminaison de transfert de données où vous pourrez voir la liste de tous les points de terminaison de transfert de données. Vous pourrez également voir les points de terminaison de transfert de données disponibles ou en panne.

Vous pouvez également rechercher des points de terminaison de transfert de données par nom et accéder à des informations supplémentaires pour chaque agent disponible.

Tâches de transfert

Les tâches de transfert sont utilisées pour définir le codage source et le codage cible des ensembles de données. Le codage source est le format des ensembles de données source, et le codage cible est le format dans lequel ces ensembles de données seront stockés dans le compartiment Amazon S3 cible. Ces compartiments cibles sont définis par des tâches de transfert.

Cette procédure part du principe que vous avez effectué les étapes de configuration Configuration de la modernisation AWS du mainframe et de configurationthe section called "Points de terminaison de transfert de données".

Rubriques

- Création de tâches de transfert
- Afficher les tâches de transfert

Création de tâches de transfert

Pour créer des tâches de transfert pour le transfert de fichiers, vous devez suivre ces étapes dans la console de modernisation du AWS mainframe.

Pour créer une tâche de transfert

> **Note**
>
> Vous devez disposer d'au moins un point de terminaison de transfert de données pour créer de nouvelles tâches de transfert.

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.

2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Transférer des tâches, sous Transfert de fichiers, sélectionnez un point de terminaison de transfert de données pour créer des tâches de transfert.
4. Si votre terminal de transfert de données ne comporte aucune tâche de transfert, vous pouvez créer de nouvelles tâches en choisissant Créer une tâche de transfert.
5. Sur la page Créer une tâche de transfert, configurez les propriétés de la tâche de transfert.

   - Sur cette page, entrez les informations de base de votre tâche de transfert, notamment le nom, la description, la clé secrète et les critères de recherche des ensembles de données.

> **Note**
>
> - Chiffrez le secret à l'aide de la clé KMS définie avec le point de terminaison du transfert de données. Le secret doit également contenir les informations d'identification du mainframe nécessaires pour accéder à l'ensemble de données sur le mainframe à l'aide des touches `userId` et`password`. Pour plus d'informations, consultez le code secret d'AWS Secrets Manager.
> - Vous devez configurer la clé secrète avec la politique basée sur les ressources suivante afin que le service AWS Mainframe Modernization puisse y accéder pour effectuer une tâche de transfert de données :
>
> ```
> {
>   "Version" : "2012-10-17",
>   "Statement" : [ {
>     "Effect" : "Allow",
>     "Principal" : {
>         "Service" : "m2.amazonaws.com"
>     },
>     "Action" : [ "secretsmanager:GetSecretValue",
>                  "secretsmanager:DescribeSecret" ],
>     "Resource" : "*"
>   } ]
> }
> ```

> **Note**
>
> La taille maximale du jeu de données actuellement prise en charge est de 90 GiB.

- Entrez ou parcourez l'emplacement de votre compartiment Amazon S3 cible pour vos fichiers.
- Le point de terminaison de transfert de données par défaut sera sélectionné. Vous pouvez également choisir de modifier le point de terminaison parmi les points de terminaison disponibles.

6. Choisissez Suivant.
7. Sur la page Sélectionner des ensembles de données, vous devez mettre à jour manuellement le codage source et le codage cible pour chacun des ensembles de données que vous avez choisis. Le codage source est le format du jeu de données source et le codage cible est le format du jeu de données cible. Il est utilisé pour convertir les ensembles de données.
8. Après avoir confirmé le codage source et cible, choisissez Next.
9. Sur la page Réviser et créer, vous pouvez consulter ou modifier les informations relatives à votre tâche de transfert.
10. Choisissez Créer une tâche de transfert.

Le message « La tâche de transfert a été créée avec succès » s'affiche.

Afficher les tâches de transfert

Pour afficher les tâches de transfert pour le transfert de fichiers, vous devez suivre ces étapes dans la console de modernisation du AWS mainframe.

Pour afficher les tâches de transfert

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le Région AWS sélecteur, choisissez la région dans laquelle vous souhaitez transférer les fichiers de votre mainframe vers un compartiment Amazon S3.
3. Sur la page Tâches de transfert, sous Transfert de fichiers, sélectionnez le point de terminaison de transfert de données pour afficher vos tâches de transfert.

4. Pour les terminaux dotés de tâches de transfert préexistantes, celles-ci seront renseignées dans la section Tâches de transfert. Vous pouvez choisir d'afficher les détails de n'importe quelle tâche de transfert dans cette liste.

Tutoriel : Démarrage avec le transfert de fichiers AWS Mainframe Modernization

AWS Mainframe Modernization File Transfer vous permet de transférer et de convertir des ensembles de données de mainframe pour des cas d'utilisation liés à la modernisation, à la migration et à l'augmentation du mainframe.

Suivez les étapes de ce didacticiel pour comprendre le fonctionnement du transfert de fichiers AWS Mainframe Modernization.

Présentation

Le transfert de fichiers comprend les éléments suivants :

1. Agent à installer sur le mainframe source.
2. Accédez aux fonctionnalités de découverte, de transfert et de conversion des ensembles de données directement depuis la console du service de gestion AWS Mainframe Modernization.

En tant qu'utilisateur, vous pouvez transférer des ensembles de données du mainframe vers votre compartiment Amazon S3.

Rubriques

Étape 1 : transférer le package tar des fichiers binaires de l'agent AWS vers la partition logique du mainframe

Téléchargez les fichiers tar à partir du lien tar de l'agent M2-Agent.

Étape 2 : Configuration de l'agent de transfert de fichiers sur le mainframe source

Au cours de cette étape, vous configurez et démarrez l'agent de transfert de fichiers AWS Mainframe Modernization sur le mainframe source. L'agent est nécessaire pour faciliter les communications entre le service de transfert de fichiers et le mainframe source. Au moins un agent est requis par ordinateur central. Plusieurs agents peuvent être démarrés pour une haute disponibilité et une évolutivité améliorée.

Suivez les instructions du the section called "Installation d'un agent de transfert de fichiers" guide pour terminer l'installation de l'agent de transfert de fichiers sur le mainframe.

Étape 3 : Création d'un point de terminaison de transfert de données

Suivez les étapes de la the section called "Points de terminaison de transfert de données" page pour créer un nouveau point de terminaison de transfert de données.

Étape 4 : Création d'une tâche de transfert

Suivez les étapes de the section called "Tâches de transfert" la page pour créer et gérer vos tâches de transfert.

Étape 5 : Afficher la progression de la tâche de transfert

Vous pouvez consulter la progression de votre tâche de transfert dans la console de modernisation du AWS mainframe. Pour plus de détails, reportez-vous à the section called "Afficher les tâches de transfert" la section.

La sécurité dans le cadre de la modernisation des AWS mainframes

Chez AWS, la sécurité dans le cloud est notre priorité numéro 1. En tant que client AWS, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité.

La sécurité est une responsabilité partagée entre AWS et vous. Le modèle de responsabilité partagéede décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est responsable de la protection de l'infrastructure qui exécute des services AWS dans le cloud AWS. AWS vous fournit également les services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des AWS programmes de conformité. Pour en savoir plus sur les programmes de conformité applicables à la modernisation des AWS mainframes, consultez la section Services AWS concernés par programme de conformité .
- Sécurité dans le cloud : votre responsabilité est déterminée par le service AWS que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris la sensibilité de vos données, les exigences de votre entreprise et la législation et la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de la modernisation des AWS mainframes. Il vous explique comment configurer la modernisation du AWS mainframe pour atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser les ressources de modernisation de votre AWS mainframe.

AWSLa modernisation du mainframe fournit ses propres ressources protégées par l'IAM (application, environnement, déploiement, etc.), qui sont les ressources administratives de modernisation du AWS mainframe, sur lesquelles toute action doit être autorisée par les politiques IAM.

AWSLa modernisation du mainframe pour le replatforming est également sécurisée par IAM. L'IAM accorde ou refuse à un mandant l'autorisation d'effectuer une action spécifique sur une ressource définie, dérivée de l'environnement mainframe d'origine, également par le biais de politiques IAM standard. Le moteur d'exécution de replateforme AWS Mainframe Modernization appelle le service d'autorisation IAM lorsqu'une application tente une telle action sur une ressource protégée. IAM

renverra l'autorisation ou le refus en fonction des mécanismes d'évaluation des politiques IAM standard.

Table des matières

Protection des données dans le cadre de la modernisation des AWS mainframes

Le modèle de responsabilité AWS partagée Le modèle de s'applique à la protection des données dans le AWS cadre de la modernisation des ordinateurs centraux. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour en savoir plus sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez le protocole SSL/TLS pour communiquer avec les ressources. AWS Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail.

- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-2 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour en savoir plus sur les points de terminaison FIPS (Federal Information Processing Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Name (Nom). Cela inclut lorsque vous utilisez la modernisation du AWS mainframe ou une autre méthode à Services AWS l'aide de la console, de l'API ou des AWS SDK. AWS CLI Toutes les données que vous saisissez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Données collectées par AWS Mainframe Modernization

AWS La modernisation du mainframe collecte plusieurs types de données auprès de vous :

- `Application configuration`: il s'agit d'un fichier JSON que vous créez pour configurer votre application. Il contient vos choix concernant les différentes options proposées par AWS Mainframe Modernization. Le fichier contient également des informations relatives aux AWS ressources dépendantes, telles que les chemins Amazon Simple Storage Service où sont stockés les artefacts de l'application ou le nom de ressource Amazon (ARN) indiquant AWS Secrets Manager où sont stockées les informations d'identification de votre base de données.
- `Application executable (binary)`: il s'agit d'un fichier binaire que vous compilez et que vous avez l'intention de déployer dans le cadre de la modernisation AWS du mainframe.
- `Application JCL or scripts`: Ce code source gère les tâches par lots ou d'autres traitements pour le compte de votre application.

- `User application data`: Lorsque vous importez des ensembles de données, AWS Mainframe Modernization les stocke dans la base de données relationnelle afin que votre application puisse y accéder.
- `Application source code`: via Amazon AppStream 2.0, AWS Mainframe Modernization fournit un environnement de développement dans lequel vous pouvez écrire et compiler du code.

AWS La modernisation du mainframe stocke ces données de manière native dans. AWS Les données que nous collectons auprès de vous sont stockées dans un compartiment Amazon S3 géré par AWS Mainframe Modernization. Lorsque vous déployez une application, AWS Mainframe Modernization télécharge les données sur une instance Amazon Elastic Compute Cloud basée sur Amazon Elastic Block Store. Lorsque le nettoyage est déclenché, les données sont supprimées du volume Amazon EBS et d'Amazon S3. Les volumes Amazon EBS sont réservés à un locataire unique, ce qui signifie qu'une instance est utilisée pour un client. Les instances ne sont jamais partagées. Lorsque vous supprimez un environnement d'exécution, le volume Amazon EBS est également supprimé. Lorsque vous supprimez une application, les artefacts et la configuration sont supprimés d'Amazon S3.

Les journaux des applications sont stockés sur Amazon CloudWatch. Les messages du journal des applications clientes sont également CloudWatch exportés vers. Les CloudWatch journaux peuvent contenir des données sensibles pour le client, telles que des données commerciales ou des informations de sécurité (dans les messages de débogage). Pour de plus amples informations, veuillez consulter Surveillance de la modernisation des AWS mainframes avec Amazon CloudWatch.

En outre, si vous choisissez d'associer un ou plusieurs systèmes de fichiers Amazon Elastic File System ou Amazon FSx à votre environnement d'exécution, les données de ces systèmes seront stockées dans. AWS Vous devrez nettoyer ces données si vous décidez de ne plus utiliser les systèmes de fichiers.

Vous pouvez utiliser toutes les options de chiffrement Amazon S3 disponibles pour sécuriser vos données lorsque vous les placez dans le compartiment Amazon S3 utilisé par AWS Mainframe Modernization pour le déploiement d'applications et les importations de jeux de données. En outre, vous pouvez utiliser les options de chiffrement Amazon EFS et Amazon FSx si vous associez un ou plusieurs de ces systèmes de fichiers à votre environnement d'exécution.

Chiffrement des données interrompu pour le service de modernisation des AWS mainframes

AWS La modernisation du mainframe s'intègre AWS Key Management Service pour fournir un chiffrement transparent côté serveur (SSE) sur toutes les ressources dépendantes qui stockent les données de façon permanente, à savoir Amazon Simple Storage Service, Amazon DynamoDB et Amazon Elastic Block Store. AWS La modernisation du mainframe crée et gère des AWS KMS clés de chiffrement symétriques pour vous. AWS KMS

Le chiffrement des données au repos par défaut permet de réduire les frais opérationnels et la complexité liés à la protection des données sensibles. Dans le même temps, il vous permet de migrer des applications qui nécessitent une conformité stricte en matière de chiffrement et des exigences réglementaires.

Vous ne pouvez pas désactiver cette couche de chiffrement ni sélectionner un autre type de chiffrement lorsque vous créez des environnements d'exécution et des applications.

Vous pouvez utiliser votre propre clé gérée par le client pour les applications de modernisation des AWS mainframes et les environnements d'exécution afin de chiffrer les ressources Amazon S3 et Amazon EBS.

Pour vos applications de modernisation du AWS mainframe, vous pouvez utiliser cette clé pour chiffrer la définition de votre application ainsi que d'autres ressources applicatives, telles que les fichiers JCL, qui sont enregistrées dans le compartiment Amazon S3 créé dans le compte du service. Pour de plus amples informations, veuillez consulter Création d'une application .

Pour vos environnements d'exécution AWS Mainframe Modernization, AWS Mainframe Modernization utilise votre clé gérée par le client pour chiffrer le volume Amazon EBS qu'il crée et attache à votre instance Amazon EC2 de AWS modernisation du mainframe, qui figure également dans le compte du service. Pour de plus amples informations, veuillez consulter Création d'un environnement d'exécution.

> **Note**
>
> Les ressources DynamoDB sont toujours chiffrées à l'aide d' Clé gérée par AWS un compte de service intégré au Mainframe AWS Modernization. Vous ne pouvez pas chiffrer les ressources DynamoDB à l'aide d'une clé gérée par le client.

AWS La modernisation du mainframe utilise votre clé gérée par le client pour les tâches suivantes :

- Redéploiement d'une application.
- Remplacement d'une AWS instance Amazon EC2 de modernisation du mainframe.

AWS Mainframe Modernization n'utilise pas votre clé gérée par le client pour chiffrer les bases de données Amazon Relational Database Service ou Amazon Aurora, les files d'attente Amazon Simple Queue Service et les caches ElastiCache Amazon créés pour prendre en charge AWS une application de modernisation du mainframe, car aucun d'entre eux ne contient de données client.

Pour plus d'informations, consultez Clés gérées par le client dans le Guide du développeur AWS Key Management Service (langue française non garantie).

Le tableau suivant résume la façon dont la modernisation AWS du mainframe chiffre vos données sensibles.

Type de données	Clé gérée par AWS chiffrement	Chiffrement des clés géré par le client
`Definition` Contient la définition d'une application particulière.	Activées	Activées
`EnvironmentSummary` Contient des informations sur l'environnement d'exécution.	Activées	Activées
`ApplicationSummary` Contient des informations sur l'application AWS Mainframe Modernization.	Activées	Activées
`DeploymentSummary` Contient des informations sur le déploiement d'une applicati	Activées	Activées

Type de données	Clé gérée par AWS chiffrement	Chiffrement des clés géré par le client
on de modernisation AWS du mainframe.		

> **Note**
>
> AWS La modernisation du mainframe active automatiquement le chiffrement au repos Clés gérées par AWS afin de protéger gratuitement vos données sensibles. Toutefois, AWS KMS des frais s'appliquent pour l'utilisation d'une clé gérée par le client. Pour plus d'informations sur la tarification, consultez Tarification d'AWS Key Management Service.

Pour plus d'informations sur AWS KMS, voir AWS Key Management Service.

Comment la modernisation AWS du mainframe utilise les subventions dans AWS KMS

AWS La modernisation du mainframe nécessite une subvention pour utiliser votre clé gérée par le client.

Lorsque vous créez une application ou un environnement d'exécution, ou que vous déployez une application dans AWS Mainframe Modernization chiffrée à l'aide d'une clé gérée par le client, AWS Mainframe Modernization crée une subvention en votre nom en envoyant une CreateGrantdemande à. AWS KMS Les subventions AWS KMS sont utilisées pour donner à AWS Mainframe Modernization l'accès à une clé KMS dans un compte client.

AWS La modernisation du mainframe nécessite l'autorisation d'utiliser votre clé gérée par le client pour les opérations internes suivantes :

- Envoyez DescribeKeydes demandes AWS KMS à pour vérifier que l'ID de clé symétrique géré par le client saisi lors de la création d'une application, d'un environnement d'exécution ou du déploiement d'une application est valide.
- Envoyez GenerateDataKeydes demandes AWS KMS à chiffrer le volume Amazon EBS attaché aux instances Amazon EC2 AWS hébergeant les environnements d'exécution Mainframe Modernization.

- Envoyez des demandes de déchiffrement AWS KMS à pour déchiffrer le contenu chiffré sur Amazon EBS.

AWS La modernisation du mainframe utilise des AWS KMS subventions pour déchiffrer vos secrets stockés dans Secrets Manager et lors de la création d'un environnement d'exécution, de la création ou du redéploiement d'une application et de la création d'un déploiement. Les subventions créées par la modernisation AWS du mainframe soutiennent les opérations suivantes :

- Créez ou mettez à jour une subvention d'environnement d'exécution :
  - Decrypt (Déchiffrer)
  - Encrypt (Chiffrer)
  - ReEncryptFrom
  - ReEncryptTo
  - GenerateDataKey
  - DescribeKey
  - CreateGrant
- Créez ou redéployez une subvention de candidature :
  - GenerateDataKey
- Créez une subvention de déploiement :
  - Decrypt

Vous pouvez révoquer l'accès à l'octroi ou supprimer l'accès du service à la clé gérée par le client à tout moment. Si vous le faites, AWS Mainframe Modernization ne pourra accéder à aucune des données chiffrées par la clé gérée par le client, ce qui affectera les opérations qui dépendent de ces données. Par exemple, si AWS Mainframe Modernization essayait d'accéder à une définition d'application chiffrée par une clé gérée par le client sans accorder cette clé, l'opération de création de l'application échouerait.

AWS Mainframe Modernization collecte les configurations des applications utilisateur (fichiers JSON) et les artefacts (fichiers binaires et exécutables). Il crée également des métadonnées qui suivent les différentes entités utilisées pour le fonctionnement de la modernisation du AWS mainframe, et crée des journaux et des métriques. Les journaux et statistiques visibles par le client incluent :

- CloudWatch journaux qui reflètent l'application et le moteur d'exécution (AWS Blu Age ou Micro Focus).

- CloudWatch métriques pour les tableaux de bord des opérations.

En outre, AWS Mainframe Modernization collecte des données d'utilisation et des mesures pour les mesures, les rapports d'activité, etc. concernant les services. Ces données ne sont pas visibles par le client.

AWS La modernisation du mainframe stocke ces données à différents endroits en fonction du type de données. Les données client que vous chargez sont stockées dans un compartiment Amazon S3. Les données de service sont stockées à la fois dans Amazon S3 et DynamoDB. Lorsque vous déployez une application, vos données et les données de service sont téléchargées sur les volumes Amazon EBS. Si vous choisissez d'associer le stockage Amazon EFS ou Amazon FSx à votre environnement d'exécution, les données stockées dans ces systèmes de fichiers sont également téléchargées sur le volume Amazon EBS.

Le chiffrement au repos est configuré par défaut. Vous ne pouvez ni le désactiver ni le modifier. Actuellement, vous ne pouvez pas non plus modifier sa configuration.

Création d'une clé gérée par le client

Vous pouvez créer une clé symétrique gérée par le client à l'aide des API AWS Management Console ou des AWS KMS API.

Pour créer une clé symétrique gérée par le client

Suivez les étapes de la rubrique Création d'une clé symétrique gérée par le client dans le Guide du développeur AWS Key Management Service .

Stratégie de clé

Les politiques de clés contrôlent l'accès à votre clé gérée par le client. Chaque clé gérée par le client doit avoir exactement une stratégie de clé, qui contient des instructions qui déterminent les personnes pouvant utiliser la clé et comment elles peuvent l'utiliser. Lorsque vous créez votre clé gérée par le client, vous pouvez spécifier une stratégie de clé. Pour plus d'informations, consultez Gestion de l'accès aux clés gérées par le client dans le Guide du développeur AWS Key Management Service .

Pour utiliser votre clé gérée par le client avec vos ressources de modernisation du AWS mainframe, les opérations d'API suivantes doivent être autorisées dans la politique clé :

- `kms:CreateGrant` : ajoute une attribution à une clé gérée par le client. Accorde un accès de contrôle à une clé KMS spécifiée, ce qui permet d'accéder aux opérations d'octroi requises par la modernisation AWS du mainframe. Pour plus d'informations sur l'utilisation des subventions, consultez le guide du AWS Key Management Service développeur.

  Cela permet à AWS Mainframe Modernisation d'effectuer les opérations suivantes :

  - Appelez `GenerateDataKey` pour générer une clé de données chiffrée et la stocker, car la clé de données n'est pas immédiatement utilisée pour chiffrer.
  - Appelez `Decrypt` pour utiliser la clé de données chiffrée stockée afin d'accéder aux données chiffrées.
  - Configurez un directeur partant à la retraite pour permettre au service de`RetireGrant`.
- `kms:DescribeKey`— Fournit les informations clés gérées par le client pour permettre à AWS Mainframe Modernization de valider la clé.

AWS La modernisation du mainframe nécessite `kms:CreateGrant` des `kms:DescribeKey` autorisations et des autorisations dans la politique clé du client. AWS La modernisation du mainframe utilise cette politique pour créer une subvention pour elle-même.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "Enable IAM User Permissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::AccountId:role/ExampleRole"
        },
        "Action": [
            "kms:CreateGrant",
            "kms:DescribeKey"
        ],
        "Resource": "*"
    }]
}
```

> **Note**
>
> Le rôle illustré `Principal` dans l'exemple précédent est celui que vous utilisez pour les opérations de modernisation AWS du mainframe telles que `CreateApplication` et`CreateEnvironment`.

Pour plus d'informations sur la spécification d'autorisations dans une politique, consultez le Guide du développeur AWS Key Management Service .

Pour plus d'informations sur le dépannage des clés d'accès, consultez le Guide du développeur AWS Key Management Service .

Spécification d'une clé gérée par le client pour la modernisation AWS du mainframe

Vous pouvez spécifier une clé gérée par le client pour les ressources suivantes :

- Application
- Environnement

Lorsque vous créez une ressource, vous pouvez spécifier la clé en saisissant un ID KMS, que AWS Mainframe Modernization utilise pour chiffrer les données sensibles stockées par la ressource.

- ID KMS : identifiant de clé pour une clé gérée par le client. Saisissez un ID de clé, un ARN de clé, un nom d'alias ou un ARN d'alias.

Vous pouvez spécifier une clé gérée par le client à l'aide du AWS Management Console ou du AWS CLI.

Pour spécifier votre clé gérée par le client lors de la création d'un environnement d'exécution dans le AWS Management Console, voirCréation d'un environnement d'exécution pour la modernisation du AWS mainframe. Pour spécifier votre clé gérée par le client lors de la création d'une application dans le AWS Management Console, voirCréation d'une application de modernisation AWS du mainframe.

Pour ajouter votre clé gérée par le client lorsque vous créez un environnement d'exécution avec le AWS CLI, spécifiez le `kms-key-id` paramètre comme suit :

```
aws m2 create-environment –engine-type microfocus –instance-type M2.m5.large
--publicly-accessible –engine-version 7.0.3 –name test
--high-availability-config desiredCapacity=2
--kms-key-id myEnvironmentKey
```

Pour ajouter votre clé gérée par le client lorsque vous créez une application avec le AWS CLI, spécifiez le `kms-key-id` paramètre comme suit :

```
aws m2 create-application –name test-application –description my description
--engine-type microfocus
--definition content="$(jq -c . raw-template.json | jq -R)"
--kms-key-id myApplicationKey
```

AWS Contexte de chiffrement de la modernisation du mainframe

Un contexte de chiffrement est un ensemble facultatif de paires clé-valeur qui contient des informations contextuelles supplémentaires sur les données.

AWS KMS utilise le contexte de chiffrement comme données authentifiées supplémentaires pour prendre en charge le chiffrement authentifié. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez inclure le même contexte de chiffrement dans la demande.

AWS Contexte de chiffrement de la modernisation du mainframe

AWS La modernisation du mainframe utilise le même contexte de chiffrement dans toutes les opérations AWS KMS cryptographiques liées à une application (création d'une application et création d'un déploiement), où la clé se trouve `aws:m2:app` et la valeur est l'identifiant unique de l'application.

Example

```
"encryptionContextSubset": {
        "aws:m2:app": "a1bc2defabc3defabc4defabcd"
}
```

Utilisation du contexte de chiffrement pour la surveillance

Lorsque vous utilisez une clé symétrique gérée par le client pour chiffrer vos applications ou vos environnements d'exécution, vous pouvez également utiliser le contexte de chiffrement dans les

enregistrements d'audit et les journaux pour identifier la manière dont la clé gérée par le client est utilisée.

Utilisation du contexte de chiffrement pour contrôler l'accès à votre clé gérée par le client

Vous pouvez utiliser le contexte de chiffrement dans les stratégies de clé et les politiques IAM comme `conditions` pour contrôler l'accès à votre clé symétrique gérée par le client. Vous pouvez également utiliser des contraintes de contexte de chiffrement dans un octroi.

AWS La modernisation du mainframe utilise une contrainte de contexte de chiffrement dans les autorisations afin de contrôler l'accès à la clé gérée par le client dans votre compte ou votre région. La contrainte d'octroi exige que les opérations autorisées par l'octroi utilisent le contexte de chiffrement spécifié. L'exemple suivant est une subvention que AWS Mainframe Modernization utilise pour chiffrer un artefact d'application lors de la création d'une application.

```
//This grant is retired immediately after create application finish
{
   "grantee-principal": m2.us-west-2.amazonaws.com,
   "retiring-principal": m2.us-west-2.amazonaws.com,
   "operations": [
       "GenerateDataKey"
   ]
   "condition": {
        "encryptionContextSubset": {
            “aws:m2:app”: “a1bc2defabc3defabc4defabcd”
   }
}
```

Surveillance de vos clés de chiffrement pour la modernisation AWS du mainframe

Lorsque vous utilisez une clé gérée par le AWS KMS client avec vos ressources de modernisation du AWS mainframe, vous pouvez utiliser AWS CloudTrailAmazon CloudWatch Logs pour suivre les demandes envoyées par AWS Mainframe Modernization. AWS KMS

Exemples d'environnements d'exécution

Les exemples suivants sont des AWS CloudTrail événements destinés à`DescribeKey`, `CreateGrantGenerateDataKey`, et `Decrypt` à surveiller les opérations KMS appelés par AWS Mainframe Modernization pour accéder aux données chiffrées par votre clé gérée par le client :

DescribeKey

AWS La modernisation du mainframe utilise cette `DescribeKey` opération pour vérifier si la clé gérée par le AWS KMS client associée à votre environnement d'exécution existe dans le compte et dans la région.

L'exemple d'événement suivant enregistre l'opération `DescribeKey` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T19:40:26Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.182",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
```

```
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer votre environnement d'exécution, AWS Mainframe Modernization envoie plusieurs `CreateGrant` demandes en votre nom pour effectuer les opérations KMS nécessaires. Certaines des subventions créées par AWS Mainframe Modernization sont supprimées immédiatement après leur utilisation. Les autres sont retirés lorsque vous supprimez l'environnement d'exécution.

L'exemple d'événement suivant enregistre l'`CreateGrant`opération pour le rôle d'exécution Lambda associé au flux de travail Create Environment.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:11:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
```

```
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

L'exemple d'événement suivant enregistre l'`CreateGrant`opération pour le rôle lié au service du groupe Auto Scaling. Le rôle d'exécution Lambda associé au flux de travail Create Environment appelle cette `CreateGrant` opération. Il autorise le rôle d'exécution à créer une sous-subvention pour le rôle lié au service du groupe Auto Scaling.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65MZFUPM4JO:EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "arn": "arn:aws:sts::111122223333:assumed-role/EnvironmentWorkflow-
alpha-CreateEnvironmentLambdaS-1AU4A8VNQEEKN/EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN",
                "accountId": "111122223333",
                "userName": "EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:22:28Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "54.148.236.160",
    "userAgent": "aws-sdk-java/2.18.21 Linux/4.14.255-276-224.499.amzn2.x86_64
 OpenJDK_64-Bit_Server_VM/11.0.14.1+10-LTS Java/11.0.14.1 vendor/Amazon.com_Inc. md/
internal exec-env/AWS_Lambda_java11 io/sync http/Apache cfg/retry-mode/legacy",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
}
```

GenerateDataKey

Lorsque vous activez une clé gérée par le AWS KMS client pour votre ressource d'environnement d'exécution, Auto Scaling crée une clé unique pour chiffrer le volume Amazon EBS associé à l'environnement d'exécution. Il envoie une `GenerateDataKey` demande AWS KMS qui spécifie la clé gérée par le AWS KMS client pour la ressource.

L'exemple d'événement suivant enregistre l'opération `GenerateDataKey` :

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65EEXVIEH7D:AutoScaling",
        "arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForAutoScaling/
AutoScaling",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAutoScaling"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:23:16Z",
```

```
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "autoscaling.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:18Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "autoscaling.amazonaws.com",
    "userAgent": "autoscaling.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "numberOfBytes": 64
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Lorsque vous accédez à un environnement d'exécution chiffré, Amazon EBS appelle l'`Decrypt`opération pour utiliser la clé de données cryptée stockée afin d'accéder aux données chiffrées.

L'exemple d'événement suivant enregistre l'opération `Decrypt` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ebs.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:22Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ebs.amazonaws.com",
    "userAgent": "ebs.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```

Exemples d'applications

Les exemples suivants sont des AWS CloudTrail événements pour `CreateGrant` et `GenerateDataKey` pour surveiller les opérations KMS appelés par AWS Mainframe Modernization afin d'accéder aux données chiffrées par votre clé gérée par le client :

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer les ressources de votre application, le rôle d'exécution Lambda envoie `CreateGrant` une demande en votre nom pour accéder à la clé KMS de votre compte. AWS La subvention permet au rôle d'exécution Lambda de télécharger les ressources de l'application client sur Amazon S3 à l'aide de votre clé gérée par le client. Cette subvention est retirée immédiatement après la création de la demande.

L'exemple d'événement suivant enregistre l'opération `CreateGrant` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T22:47:04Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "operations": [
            "GenerateDataKey"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


GenerateDataKey

Lorsque vous activez une clé gérée par le AWS KMS client pour votre ressource d'application, le rôle d'exécution Lambda crée une clé qu'il utilise pour chiffrer et télécharger les données

des clients sur Amazon Simple Storage Service. Le rôle d'exécution Lambda envoie une `GenerateDataKey` demande AWS KMS qui spécifie la clé gérée par le AWS KMS client pour la ressource.

L'exemple d'événement suivant enregistre l'opération `GenerateDataKey` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65CLCEKKC7Z:ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "arn": "arn:aws:sts::111122223333:assumed-role/ApplicationWorkflow-
alpha-CreateApplicationVersion-1IZRBZYDG20B/ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B",
                "accountId": "111122223333",
                "userName": "ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:28:32Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:29:08Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
```

```
            "aws:m2:app": "a1bc2defabc3defabc4defabcd",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcd/1/cics-transaction/ZBNKE35.so"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Exemples de déploiements

Les exemples suivants sont des AWS CloudTrail événements pour `CreateGrant` et `Decrypt` pour surveiller les opérations KMS appelés par AWS Mainframe Modernization afin d'accéder aux données chiffrées par votre clé gérée par le client :

CreateGrant

Lorsque vous utilisez une clé gérée par le AWS KMS client pour chiffrer vos ressources de déploiement, AWS Mainframe Modernization envoie deux `CreateGrant` demandes en votre nom. La première subvention est liée au rôle d'exécution Lambda actuel à appeler ListBatchJobScriptFiles et est retirée immédiatement après la fin du déploiement. La deuxième subvention va à l'encontre du rôle d'instance défini par Amazon EC2 afin qu'Amazon EC2 puisse télécharger les ressources des applications client depuis Amazon S3. Cette autorisation est retirée lorsque l'application est supprimée de l'environnement d'exécution.

L'exemple d'événement suivant enregistre l'opération `CreateGrant` :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:07Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Lorsque vous accédez à un déploiement, Amazon EC2 appelle l'`Decrypt`opération pour utiliser la clé de données cryptée stockée afin de déchiffrer et de télécharger les données client chiffrées depuis Amazon S3.

L'exemple d'événement suivant enregistre l'opération `Decrypt` :

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65BSPZ37E6G:m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "arn": "arn:aws:sts::111122223333:assumed-role/
SupernovaEnvironmentInstanceScopeDownRole/m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/
SupernovaEnvironmentInstanceScopeDownRole",
                "accountId": "111122223333",
                "userName": "SupernovaEnvironmentInstanceScopeDownRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:19:29Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcdm",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcdm/1/cics-transaction/BBANK40P.so"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
```

```
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

En savoir plus

Les ressources suivantes fournissent plus d'informations sur le chiffrement des données au repos.

- Pour plus d'informations sur les concepts de base AWS Key Management Service, consultez le Guide du développeur AWS Key Management Service .
- Pour plus d'informations sur les Bonnes pratiques de sécurité pour AWS Key Management Service, consultez le Guide du développeur AWS Key Management Service .

Chiffrement en transit

Pour les applications interactives qui font partie des charges de travail transactionnelles, les échanges de données entre l'émulateur de terminal et le point de terminaison du service AWS Mainframe Modernization pour le protocole TN3270 ne sont pas chiffrés en transit. Si l'application nécessite un chiffrement pendant le transit, vous souhaiterez peut-être implémenter des mécanismes de tunneling supplémentaires.

AWS La modernisation du mainframe utilise le protocole HTTPS pour chiffrer les API du service. Toutes les autres communications au sein de AWS Mainframe Modernization sont protégées par le VPC de service ou le groupe de sécurité, ainsi que par le protocole HTTPS. AWS La modernisation du mainframe transfère les artefacts, les configurations et les données des applications. Les artefacts d'application sont copiés depuis un compartiment Amazon S3 dont vous êtes le propriétaire, tout comme les données d'application. Vous pouvez fournir des configurations d'applications à l'aide d'un lien vers Amazon S3 ou en téléchargeant un fichier localement.

Le chiffrement de base en transit est configuré par défaut, mais ne s'applique pas au protocole TN3270. AWS La modernisation du mainframe utilise le protocole HTTPS pour les points de terminaison des API, qui sont également configurés par défaut.

Identity and Access Management pour la modernisation des AWS mainframes

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources de modernisation du AWS mainframe. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans le cadre de la modernisation du AWS mainframe.

Utilisateur du service : si vous utilisez le service AWS Mainframe Modernization pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Au fur et à mesure que vous utilisez de plus en plus de fonctionnalités de modernisation du AWS mainframe pour effectuer votre travail, vous aurez peut-être besoin d'autorisations supplémentaires. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur. Si vous ne parvenez pas à accéder à une fonctionnalité dans AWS Mainframe Modernization, consultezRésolution des problèmes liés à la modernisation AWS du mainframe (identité et accès).

Administrateur de service — Si vous êtes responsable des ressources de modernisation des AWS mainframes dans votre entreprise, vous avez probablement un accès complet à la modernisation des AWS mainframes. C'est à vous de déterminer les fonctionnalités et ressources de modernisation

du AWS mainframe auxquelles les utilisateurs du service doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour en savoir plus sur la manière dont votre entreprise peut utiliser l'IAM avec la modernisation du AWS mainframe, consultez. Comment fonctionne la modernisation AWS du mainframe avec IAM

Administrateur IAM : si vous êtes administrateur IAM, vous souhaiterez peut-être en savoir plus sur la manière dont vous pouvez rédiger des politiques pour gérer l'accès à la modernisation des AWS mainframes. Pour consulter des exemples de politiques basées sur l'identité pour la modernisation du AWS mainframe que vous pouvez utiliser dans IAM, consultez. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez la section Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer vous-même les demandes, consultez la section Signature des demandes AWS d'API dans le guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour en savoir plus, veuillez consulter Multi-factor authentication (Authentification multifactorielle) dans le Guide de l'utilisateur AWS IAM Identity Center et Utilisation de l'authentification multifactorielle (MFA) dans l'interface AWS dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur root, consultez Tâches nécessitant des informations d'identification d'utilisateur root dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons de faire pivoter les clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d'administrer des ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d'un rôle) dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Vous pouvez assumer temporairement un rôle IAM dans le en AWS Management Console changeant de rôle. Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Utilisation de rôles IAM dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré – Pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM

Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, veuillez consulter la rubrique Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, une fonction de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, l'action que vous effectuez est susceptible de lancer une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d'accès.
  - Fonction du service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.
  - Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un

administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage des clés d'accès au sein de l'instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, consultez Quand créer un rôle IAM (au lieu d'un utilisateur) dans le Guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Présentation des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui

autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux

politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, consultez Présentation des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonction avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations qui en résultent représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) — Les SCP sont des politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans. AWS Organizations AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctions d'une organisation, vous pouvez appliquer les politiques de contrôle de service (SCP) à l'un ou à l'ensemble de vos comptes. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Pour plus d'informations sur les organisations et les SCP, consultez Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations .
- politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de la séance obtenue sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques remplace l'autorisation. Pour plus d'informations, consultez Politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations obtenues sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Comment fonctionne la modernisation AWS du mainframe avec IAM

Avant d'utiliser l'IAM pour gérer l'accès à la modernisation du AWS mainframe, découvrez quelles fonctionnalités IAM peuvent être utilisées dans le cadre de la modernisation du mainframe. AWS

Fonctionnalités IAM que vous pouvez utiliser dans le cadre de la modernisation du AWS mainframe

Fonction IAM	AWS Assistance à la modernisation du mainframe
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition d'une politique	Oui
ACL	Non
ABAC (étiquettes dans les politiques)	Oui
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Fonctions de service	Oui
Rôles liés à un service	Oui

Pour obtenir une vue d'ensemble de la façon dont la modernisation du AWS mainframe et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez la section AWS Services compatibles avec IAM dans le Guide de l'utilisateur IAM.

Politiques basées sur l'identité pour AWS la modernisation du mainframe

Prend en charge les politiques basées sur l’identité	Oui

Les politiques basées sur l’identité sont des documents de politique d’autorisations JSON que vous pouvez attacher à une identité telle qu’un utilisateur, un groupe d’utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d’actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l’identité, veuillez consulter Création de politiques IAM dans le Guide de l’utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité car celle-ci s’applique à l’utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Politiques basées sur les ressources dans le cadre de la modernisation du mainframe AWS

Prend en charge les politiques basées sur les ressources	Non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de

confiance de rôle IAM et des politiques de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal entre comptes à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Différence entre les rôles IAM et les politiques basées sur une ressource dans le Guide de l'utilisateur IAM.

Actions stratégiques pour la modernisation des AWS ordinateurs centraux

Prend en charge les actions de politique	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une stratégie afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions de modernisation du AWS mainframe, voir Actions définies par la modernisation du AWS mainframe dans la référence d'autorisation de service.

Dans le cadre de la modernisation AWS du mainframe, les actions stratégiques utilisent le préfixe suivant avant l'action :

```
m2
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "m2:StartApplication",
      "m2:StopApplication"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `List`, incluez l'action suivante :

```
"Action": "m2:List*"
```

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Ressources relatives aux politiques relatives à la AWS modernisation des ordinateurs centraux

Prend en charge les ressources de politique	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets pour lesquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions

qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Vous pouvez restreindre l'accès à des ressources spécifiques de modernisation du AWS mainframe en utilisant leurs ARN pour identifier la ressource à laquelle s'applique la politique IAM. Pour de plus amples informations sur le format des ARN, veuillez consulter Amazon Ressource Names (ARN) dans la Références générales AWS.

Par exemple, un environnement de modernisation de AWS mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:m2:regionId:accountId:env/service-generated-unique-identifier"
```

Une application de modernisation AWS du mainframe possède l'ARN suivant.

```
"Resource": "arn:aws:m2:regionId:accountId:app/service-generated-unique-identifier"
```

Les actions de modernisation du AWS mainframe ne prennent pas toutes en charge les autorisations au niveau des ressources. Pour les actions qui ne prennent pas en charge les autorisations au niveau des ressources, vous devez utiliser le caractère générique (*).

Les actions de modernisation du AWS mainframe suivantes ne prennent pas en charge les autorisations au niveau des ressources.

```
ListApplications
            ListApplicationVersions
            ListBatchJobDefinitions
            ListBatchJobExecutions
            ListDataSetImportHistory
            ListDataSets
            ListDeployments
            ListEngineVersions
            ListEnvironments
            ListTagsForResource
```

Pour consulter la liste des types de ressources de modernisation du AWS mainframe et de leurs ARN, voir Ressources définies par la modernisation du AWS mainframe dans la référence d'autorisation de service. Pour savoir avec quelles actions vous pouvez spécifier l'ARN de chaque ressource, voir Actions définies par la modernisation du AWS mainframe.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

AWS Autorisations de l'API de modernisation du mainframe : référence aux actions, aux ressources et aux conditions

Lorsque vous rédigez des politiques d'autorisation que vous pouvez associer à une identité IAM (politiques basées sur l'identité), vous pouvez utiliser le tableau suivant comme référence. Le tableau inclut les éléments suivants :

- Chaque opération de l'API de modernisation du AWS mainframe
- Actions correspondantes pour lesquelles vous pouvez accorder des autorisations pour effectuer l'action
- La AWS ressource pour laquelle vous pouvez accorder les autorisations

Vous spécifiez les actions dans le champ `Action` de la politique, ainsi que la valeur des ressources dans le champ `Resource` de la politique.

Vous pouvez utiliser des clés de condition AWS globales dans vos politiques de modernisation AWS du mainframe pour exprimer des conditions. Pour obtenir la liste complète des AWS clés, consultez la section Clés de condition globale disponibles dans le guide de l'utilisateur IAM.

> **Note**
>
> Pour indiquer une action, utilisez le préfixe `m2:` suivi du nom de l'opération d'API (par exemple, `m2:CreateApplication`).

AWS API de modernisation du mainframe et autorisations requises pour les actions

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
CancelBatchJobExecution		Application
CreateApplication	s3:GetObject s3:ListBucket	Application
CreateDataSetImportTask	m2:CreateDataSetImportTask s3:GetObject	Application
CreateDeployment	elasticloadbalancing:AddTags elasticloadbalancing:CreateListener elasticloadbalancing:CreateTargetGroup elasticloadbalancing:RegisterTargets	Application
CreateEnvironment	ec2:CreateNetworkInterface ec2:CreateNetworkInterfacePermission ec2:DescribeNetworkInterfaces ec2:DescribeSecurityGroups ec2:DescribeSubnets ec2:DescribeVpcAttribute	Environnement

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources

	`ec2:DescribeVpcs`
	
	`ec2:ModifyNetworkInterfaceAttribute`
	
	`elasticfilesystem:DescribeMountTargets`
	
	`elasticloadbalancing:AddTags`
	
	`elasticloadbalancing:CreateLoadBalancer`
	
	`fsx:DescribeFileSystems`
	
	`iam:CreateServiceLinkedRole`	

DeleteApplication	`elasticloadbalancing:DeleteListener`
	
	`elasticloadbalancing:DeleteTargetGroup`
	
	`logs:DeleteLogDelivery`
		Application

DeleteApplicationFromEnvironment	`elasticloadbalancing:DeleteListener`
	
	`elasticloadbalancing:DeleteTargetGroup`
		Application
		
		Environnement

DeleteEnvironment	`elasticloadbalancing:DeleteLoadBalancer`	Environnement

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
GetApplication		Application
GetApplicationVersion		Application
GetBatchJobExecution		Application
GetDataSetDetails		Application
GetDataSetImportTask		Application
GetDeployment		Application
GetEnvironment		Environnement
ListApplications		*
ListApplicationVersions		*
ListBatchJobDefinitions		*
ListBatchJobExecutions		*
ListDataSetImportHistory		*
ListDataSets		*
ListDeployments		*
ListEngineVersions		*
ListEnvironments		*

AWS Opérations d'API de modernisation du mainframe	Autorisations requises (Action d'API)	Ressources
ListTagsForResource		*
StartApplication		Application
StartBatchJob		Application
StopApplication		Application
TagResource		*
UntagResource		*
UpdateApplication	`s3:GetObject` `s3:ListBucket`	Application
UpdateEnvironment		Environnement

Clés d'état des politiques pour la modernisation des AWS ordinateurs centraux

Prend en charge les clés de condition de politique spécifiques au service	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez

plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Les clés de condition suivantes sont spécifiques à la modernisation du AWS mainframe

```
m2:EngineType
            m2:InstanceType
```

Pour consulter la liste des clés de condition de modernisation du AWS mainframe, voir Clés de condition pour la modernisation du AWS mainframe dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, voir Actions définies par la modernisation du AWS mainframe.

Pour consulter des exemples de politiques basées sur l'identité en matière de modernisation du AWS mainframe, voir. Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Listes de contrôle d'accès (ACL) dans le cadre de la modernisation des AWS mainframes

Prend en charge les listes ACL	Non

Les listes de contrôle d'accès (ACL) vérifient quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Contrôle d'accès basé sur les attributs (ABAC) avec modernisation du mainframe AWS

Prend en charge ABAC (étiquettes dans les politiques)	Oui

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations de balise dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur l'ABAC, consultez Qu'est-ce que le contrôle d'accès basé sur les attributs (ABAC) ? dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès basé sur les attributs (ABAC) dans le Guide de l'utilisateur IAM.

Utilisation d'informations d'identification temporaires dans le cadre de la AWS modernisation du mainframe

Prend en charge les informations d'identification temporaires	Oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent

avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Changement de rôle (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Sessions d'accès direct pour la modernisation du AWS mainframe

Prend en charge les transmissions de sessions d'accès (FAS)	Oui

Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, l'action que vous effectuez est susceptible de lancer une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

> **Important**
>
> Ces jetons permettent à AWS Mainframe Modernization d'accéder aux données des clients sans votre accord explicite ; par exemple, AWS Mainframe Modernization déploie des artefacts d'application avec les données commerciales associées à partir d'un compartiment

Amazon S3 sans obtenir l'autorisation explicite du client. Il se peut que vous deviez mettre à jour toute documentation de conformité en conséquence.

Rôles de service pour la modernisation AWS du mainframe

Prend en charge les fonctions de service	Oui

Une fonction de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

AWS La modernisation du mainframe prend en charge les rôles de service liés aux accrocs d'activité (transaction ou tâche terminée, etc.).

Warning

La modification des autorisations associées à un rôle de service peut perturber la fonctionnalité de modernisation AWS du mainframe. Modifiez les rôles de service uniquement lorsque AWS Mainframe Modernization fournit des instructions à cet effet.

Choisir un rôle IAM dans le cadre de la modernisation du AWS mainframe

Si vous avez déjà créé un rôle IAM que vos applications exécutées sur Amazon EC2 peuvent assumer, vous pouvez choisir ce rôle lorsque vous créez un modèle de lancement ou une configuration de lancement. AWS La modernisation du mainframe vous fournit une liste de rôles parmi lesquels choisir. Lors de la création de ces rôles, il est important d'associer des politiques IAM de moindre privilège qui restreignent l'accès aux appels d'API spécifiques requis par l'application. Pour de plus amples informations, veuillez consulter Rôle IAM pour les applications qui s'exécutent sur des instances Amazon EC2 dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Rôles liés aux services pour AWS la modernisation du mainframe

Prend en charge les rôles liés à un service.	Oui

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus de détails sur la création ou la gestion des rôles liés au service AWS Mainframe Modernization, consultez. Utilisation de rôles liés à des services pour la modernisation du mainframe

Pour plus d'informations sur la création ou la gestion des rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.

Exemples de politiques basées sur l'identité pour AWS la modernisation du mainframe

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources de modernisation AWS du mainframe. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l'API AWS Management Console, AWS Command Line Interface (AWS CLI) ou de AWS l'API. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par AWS Mainframe Modernization, y compris le format des ARN pour chacun des types de ressources, voir Actions, ressources et clés de condition pour la modernisation du AWS mainframe dans la référence d'autorisation de service.

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console de modernisation AWS du mainframe
- Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer des ressources de modernisation du AWS mainframe dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez Politiques gérées AWS ou Politiques gérées AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accorder les autorisations de moindre privilège - Lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation de IAM pour appliquer des autorisations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utiliser des conditions dans les politiques IAM pour restreindre davantage l'accès - Vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles - IAM Access Analyzer valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politique IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue.

Compte AWS Pour exiger le MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console de modernisation AWS du mainframe

Pour accéder à la console AWS Mainframe Modernization, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et d'afficher des informations détaillées sur les ressources de modernisation du AWS mainframe de votre Compte AWS. Si vous créez une stratégie basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette stratégie.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API qu'ils tentent d'effectuer.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console de modernisation du AWS mainframe, associez également la politique de modernisation du AWS mainframe `ConsoleAccess` ou la politique `ReadOnly` AWS gérée aux entités. Pour plus d'informations, consultez Ajout d'autorisations à un utilisateur dans le Guide de l'utilisateur IAM.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Résolution des problèmes liés à la modernisation AWS du mainframe (identité et accès)

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation de AWS Mainframe Modernization et de l'IAM.

Rubriques

- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite permettre à des personnes extérieures à moi d'accéder Compte AWS à mes ressources de modernisation de l' AWS ordinateur central

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transférer un rôle à AWS Mainframe Modernization.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans AWS Mainframe Modernization. Toutefois, l'action nécessite que le service ait des autorisations accordées par une fonction de service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les stratégies de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite permettre à des personnes extérieures à moi d'accéder Compte AWS à mes ressources de modernisation de l' AWS ordinateur central

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, consultez les éléments suivants :

- Pour savoir si la modernisation AWS du mainframe prend en charge ces fonctionnalités, consultezComment fonctionne la modernisation AWS du mainframe avec IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.

- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour découvrir quelle est la différence entre l'utilisation des rôles et l'utilisation des politiques basées sur les ressources pour l'accès entre comptes, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

Utilisation de rôles liés à des services pour la modernisation du mainframe

AWS Mainframe Modernization utilise des rôles AWS Identity and Access Management (IAM) liés aux services. Un rôle lié à un service est un type unique de rôle IAM directement lié à la modernisation du mainframe. Les rôles liés aux services sont prédéfinis par Mainframe Modernization et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration de la modernisation du mainframe, car il n'est pas nécessaire d'ajouter manuellement les autorisations nécessaires. La modernisation du mainframe définit les autorisations associées à ses rôles liés aux services, et sauf indication contraire, seule la modernisation du mainframe peut assumer ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Cela protège les ressources de modernisation de votre mainframe, car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Pour plus d'informations sur les autres services prenant en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services présentant la mention Yes (Oui) dans la colonne Service-linked roles (Rôles liés à un service). Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations de rôle liées à un service pour la modernisation du mainframe

La modernisation du mainframe utilise le rôle lié à un service nommé AWSServiceRoleForAWSM2 : configurez le réseau pour qu'il se connecte à votre VPC et accède à des ressources telles que les systèmes de fichiers.

Le rôle lié à AWSServiceRoleForAWSM 2 services fait confiance aux services suivants pour assumer le rôle :

- `m2.amazonaws.com`

La politique d'autorisation des rôles nommée AWSM2ServicePolicy permet à Mainframe Modernization d'effectuer les actions suivantes sur les ressources spécifiées :

- Créez, supprimez, décrivez et attachez des autorisations aux interfaces réseau Amazon EC2 pour l'environnement de modernisation du mainframe afin d'établir une connectivité avec le VPC du client.
- Enregistrez ou désenregistrez des entrées dans Elastic Load Balancing, qui permet aux clients de se connecter à l'environnement de modernisation du mainframe.
- Décrivez le système de fichiers Amazon EFS ou Amazon FSx, le cas échéant.
- Envoyez des métriques au client CloudWatch depuis l'environnement d'exécution.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:CreateNetworkInterface",
    "ec2:DeleteNetworkInterface",
    "ec2:DescribeNetworkInterfaces",
    "ec2:CreateNetworkInterfacePermission",
    "ec2:ModifyNetworkInterfaceAttribute"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "elasticfilesystem:DescribeMountTargets"
   ],
   "Resource": "*"
  },
  {
```

```
    "Effect": "Allow",
    "Action": [
     "elasticloadbalancing:RegisterTargets",
     "elasticloadbalancing:DeregisterTargets"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "fsx:DescribeFileSystems"
    ],
    "Resource": "*"
   },
   {
    "Effect": "Allow",
    "Action": [
     "cloudwatch:PutMetricData"
    ],
    "Resource": "*",
    "Condition": {
     "StringEquals": {
      "cloudwatch:namespace": [
       "AWS/M2"
      ]
     }
    }
   }
  ]
}
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour la modernisation du mainframe

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement d'exécution dansAWS Management Console, l'API ou l'AWSAPIAWS CLI, Mainframe Modernization crée pour vous le rôle lié au service.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement d'exécution, Mainframe Modernization crée à nouveau le rôle lié au service pour vous.

Modification d'un rôle lié à un service pour la modernisation du mainframe

La modernisation du mainframe ne vous permet pas de modifier le rôle lié au service AWSServiceRoleForAWSM 2. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Supprimer un rôle lié à un service pour la modernisation du mainframe

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

> **Note**
>
> Si le service de modernisation du mainframe utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression risque d'échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources de modernisation du mainframe utilisées par les 2 AWSServiceRoleForAWSM

- Supprimez les environnements d'exécution dans Mainframe Modernization. Assurez-vous de supprimer les applications d'un environnement avant de supprimer l'environnement lui-même.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAMAWS CLI, le ou l'AWSAPI pour supprimer le rôle lié au service AWSServiceRoleForAWSM 2. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés au service de modernisation du mainframe

La modernisation du mainframe prend en charge l'utilisation de rôles liés au service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Régions et points de terminaison AWS.

AWSValidation de la

Des auditeurs la conformité de plusieurs la conformité de plusieurs plusieurs plusieurs programmes de conformité de plusieursAWS plusieurs plusieurs programmes de conformité de plusieurs plusieurs plusieurs programmes de conformité de plusieurs plusieurs plusieurs programmes deAWS conformité de Il s'agit notamment des certifications SOC, PCI, FedRAMP, HIPAA et d'autres.

Pour obtenir la liste des services AWS relevant de programmes de conformité spécifiques, consultez les Services AWS relevant de programmes de conformité. Pour obtenir des renseignements généraux, consultez Programmes de conformité AWS.

Vous pouvez télécharger les rapports de l'audit externe avec AWS Artifact. Pour de plus amples informations, veuillez consulter Téléchargement de rapports dans AWS Artifact.

Votre responsabilité de conformité lors de l'utilisation deAWS Mainframe est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise, ainsi que par la législation et la réglementation applicables. AWSfournit les ressources de conformité :

- Guides de Quick Start (démarrage rapide) de la sécurité et de la conformité. Ces guides de déploiement traitent des considérations architecturales et fournissent des étapes pour déployer des environnements de base axés sur la sécurité et la conformité sur AWS.
- Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA – Le livre blanc décrit comment les entreprises peuvent utiliser AWS pour créer des applications conformes à HIPAA.
- Ressources de conformité AWS – Cet ensemble de manuels et de guides peut s'appliquer à votre secteur et à votre emplacement.
- Évaluation des ressources à l'aide de règles dans le Guide du développeur AWS Config : AWS Config évalue dans quelle mesure vos configurations de ressources sont conformes aux pratiques internes, aux directives sectorielles et aux réglementations.
- AWS Security Hub : ce service AWS fournit une vue complète de votre état de sécurité au sein d'AWS qui vous permet de vérifier votre conformité aux normes du secteur et aux bonnes pratiques de sécurité.

Résilience dansAWSMainframe Modernization

L'infrastructure mondiale d'AWS repose sur les régions et les zones de disponibilité AWS. Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à débit élevé et à forte redondance. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour en savoir plus sur les régions AWS et zones de disponibilité , consultez Infrastructure mondiale AWS.

Sécurité de l'infrastructure dans AWS Mainframe Modernization

En tant que service géré, AWS Mainframe Modernization est protégé par les procédures de sécurité du réseau mondial AWS. Pour plus d'informations sur les services de sécurité AWS et la manière dont AWS protège l'infrastructure, consultez la section Sécurité du cloud AWS. Pour concevoir votre environnement AWS en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le Security Pillar AWS Well-Architected Framework (Pilier de sécurité de l'infrastructure Well-Architected Framework).

Vous utilisez des appels d'API AWS publiés pour accéder à la modernisation du mainframe via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et nous recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Accès AWS Mainframe Modernization via un point de terminaison d'interface (AWS PrivateLink)

Vous pouvez l'utiliser AWS PrivateLink pour créer une connexion privée entre votre VPC et. AWS Mainframe Modernization Vous pouvez accéder à la modernisation du mainframe comme si elle se trouvait dans votre VPC, sans utiliser de passerelle Internet, de périphérique NAT, de connexion VPN ou de connexion. AWS Direct Connect Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour accéder à la modernisation du mainframe.

Vous établissez cette connexion privée en créant un point de terminaison d'interface optimisé par AWS PrivateLink. Nous créons une interface réseau de point de terminaison dans chaque sous-réseau que vous activez pour le point de terminaison d'interface. Il s'agit d'interfaces réseau gérées par les demandeurs qui servent de point d'entrée au trafic destiné à la modernisation du mainframe.

Pour plus d'informations, consultez Accès aux Services AWS via AWS PrivateLink dans le Guide AWS PrivateLink.

Considérations relatives à la modernisation du mainframe

Avant de configurer un point de terminaison d'interface pour la modernisation du mainframe, consultez les considérations du AWS PrivateLinkguide.

La modernisation du mainframe permet d'appeler toutes ses actions d'API via le point de terminaison de l'interface.

Création d'un point de terminaison d'interface pour la modernisation du mainframe

Vous pouvez créer un point de terminaison d'interface pour la modernisation du mainframe à l'aide de la console Amazon VPC ou AWS Command Line Interface du AWS CLI (). Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide AWS PrivateLink.

Créez un point de terminaison d'interface pour la modernisation du mainframe en utilisant le nom de service suivant :

```
com.amazonaws.region.m2
```

Si vous activez le DNS privé pour le point de terminaison de l'interface, vous pouvez envoyer des demandes d'API à Mainframe Modernization en utilisant son nom DNS régional par défaut. Par exemple, `m2.us-east-1.amazonaws.com`.

Création d'une politique de point de terminaison pour votre point de terminaison d'interface

Une politique de point de terminaison est une ressource IAM que vous pouvez attacher à votre point de terminaison d'interface. La politique de point de terminaison par défaut permet un accès complet à la modernisation du mainframe via le point de terminaison de l'interface. Pour contrôler l'accès autorisé à la modernisation du mainframe depuis votre VPC, associez une politique de point de terminaison personnalisée au point de terminaison de l'interface.

Une politique de point de terminaison spécifie les informations suivantes :

- Les principaux qui peuvent effectuer des actions (Comptes AWSutilisateurs et rôles IAM).
- Les actions qui peuvent être effectuées.
- La ressource sur laquelle les actions peuvent être effectuées.

Pour plus d'informations, consultez Contrôle de l'accès aux services à l'aide de politiques de point de terminaison dans le Guide AWS PrivateLink.

Exemple : politique de point de terminaison VPC pour les actions de modernisation du mainframe

Voici un exemple de politique de point de terminaison personnalisée. Lorsque vous associez cette politique au point de terminaison de votre interface, elle donne accès aux actions de modernisation du mainframe répertoriées à tous les principaux sur toutes les ressources.

```
//Example of an endpoint policy where access is granted to the
//listed AWS Mainframe Modernization actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Allow",
         "Action": [
            "m2:ListApplications",
            "m2:ListEnvironments",
            "m2:ListDeployments"
         ],
         "Resource":"*"
```

```
      }
   ]
}

//Example of an endpoint policy where access is denied to all the
//AWS Mainframe Modernization CREATE actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Deny",
         "Action": [
            "m2:Create*"
         ],
         "Resource":"*"
      }
   ]
}
```

Surveillance de la AWS modernisation du mainframe

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de AWS Mainframe Modernization et de vos autres solutions AWS. AWS fournit les outils de surveillance suivants pour suivre la modernisation du AWS mainframe, signaler tout problème et prendre des mesures automatiques le cas échéant :

- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés, et définir des alarmes qui vous informent ou prennent des mesures lorsqu'une métrique spécifique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos instances Amazon EC2 et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d'instances Amazon EC2 et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.
- AWS CloudTrailcapture les appels d'API et les événements associés effectués par ou pour le compte de votre AWS compte et envoie les fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes appelés AWS, l'adresse IP source à partir de laquelle les appels ont été effectués et la date des appels. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur AWS CloudTrail.

Surveillance de la modernisation des AWS mainframes avec Amazon CloudWatch

Vous pouvez surveiller la modernisation AWS du mainframe à l'aide d' CloudWatchun outil qui collecte les données brutes et les traite en indicateurs lisibles en temps quasi réel. Ces statistiques sont enregistrées pour une durée de 15 mois ; par conséquent, vous pouvez accéder aux informations historiques et acquérir un meilleur point de vue de la façon dont votre service ou application web s'exécute. Vous pouvez également définir des alarmes qui surveillent certains seuils

et envoient des notifications ou prennent des mesures lorsque ces seuils sont atteints. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

Les tableaux suivants répertorient les mesures et les dimensions de la modernisation des AWS mainframes. L'espace de noms pour ces métriques est`AWS/M2`.

Métriques de l'environnement d'exécution

Métrique	Description
CPUUtilization	L'utilisation du processeur par les instances de l'environnement. Dimension : EnvironmentID Unités : pourcentage Statistiques valides : moyenne, minimum, maximum
InboundNetworkThroughput	Débit réseau entrant des instances de l'environnement. Dimension : EnvironmentID Unités : octets par seconde Statistiques valides : moyenne, minimum, maximum
MemoryUtilization	Utilisation de la mémoire par les instances de l'environnement. Dimension : EnvironmentID Unités : pourcentage Statistiques valides : moyenne, minimum, maximum

Métrique	Description
OutboundNetworkThroughput	Débit réseau sortant des instances de l'environnement. Dimension : EnvironmentID Unités : octets par seconde Statistiques valides : moyenne, minimum, maximum

Métriques d'application

Métrique	Description
BatchJobCompletedCount	Le nombre de tâches terminées pendant l'intervalle de temps. Cette métrique est disponible pour Micro Focus et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : somme
BatchJobFailedCount	Nombre de tâches ayant échoué pendant l'intervalle de temps. Cette métrique est disponible pour Micro Focus et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre

Métrique	Description
	Statistiques valides : somme
JvmMemoryFree	La quantité de mémoire disponible qui n'est pas actuellement utilisée par la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum
JvmMemoryMax	La quantité maximale de mémoire autorisée pour la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum

Métrique	Description
JvmMemoryUsed	La quantité de mémoire activement utilisée par la machine virtuelle Java. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : octets Statistiques valides : moyenne, minimum, maximum
ProcessesActiveCount	Nombre actif de processus d'exécution de services simultanés qui traitent des demandes. Cette métrique n'est disponible que pour le moteur d'exécution Micro Focus. Dimension : ApplicationID Unités : nombre Statistiques valides : somme

Métrique	Description
SessionCount	Nombre de sessions HTTP pour l'application. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : moyenne, minimum, maximum
SharedMemoryFree	Mémoire disponible pour que le serveur d'entreprise stocke toutes les informations dont il a besoin pour exécuter des transactions et des tâches. Cette métrique n'est disponible que pour le moteur d'exécution Micro Focus. Dimension : ApplicationID Unités : nombre Statistiques valides : moyenne, minimum, maximum

Métrique	Description
ThreadActiveCount	Nombre de threads du moteur qui traitent les demandes. Cette métrique n'est disponible que pour le moteur d'exécution AWS Blu Age. Il est disponible pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : moyenne, minimum, maximum
TransactionCompletedCount	Le nombre de transactions validées pendant l'intervalle de temps. Cette métrique est disponible pour Micro Focus et pour AWS Blu Age 3.7.0 et versions ultérieur es. Dimension : ApplicationID Unités : nombre Statistiques valides : somme

Métrique	Description
TransactionFailedCount	Le nombre de transactions ayant échoué pendant l'intervalle de temps. Cette métrique est disponible pour Micro Focus et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : nombre Statistiques valides : somme
TransactionResponseTime	Temps écoulé entre le moment où un utilisateur envoie une demande et le moment où l'application indique que la demande est terminée. Cette métrique est disponible pour Micro Focus et pour AWS Blu Age 3.7.0 et versions ultérieures. Dimension : ApplicationID Unités : millisecondes Statistiques valides : moyenne, minimum, maximum

Dimensions

Dimension	Description
applicationId	Cette dimension filtre la métrique en fonction de l'identifiant de l'application identifiée.
ID d'environnement	Cette dimension filtre la métrique en fonction de l'environnement identifié par ID.

Journalisation des appels d'API de modernisation AWS du mainframe à l'aide de AWS CloudTrail

AWSLa modernisation du mainframe est intégrée à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans le cadre de la modernisation du AWS mainframe. CloudTrail capture tous les appels d'API pour la modernisation AWS du mainframe sous forme d'événements. Les appels capturés incluent des appels provenant de la console de modernisation du AWS mainframe et des appels de code vers les opérations de l'API de modernisation du AWS mainframe. Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements liés à la modernisation du AWS mainframe. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à AWS Mainframe Modernization, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

AWSInformations sur la modernisation du mainframe dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans le cadre de la modernisation du AWS mainframe, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre compte AWS. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris les événements liés à la modernisation AWS du mainframe, créez une trace. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions AWS. Le journal d'activité consigne les événements de toutes les régions dans la partition AWS et livre les fichiers journaux dans le compartiment Amazon S3 de votre choix. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour en savoir plus, consultez les ressources suivantes :

- Présentation de la création d'un journal de suivi
- CloudTrail services et intégrations pris en charge

- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux provenant de plusieurs régions
- Réception de fichiers CloudTrail journaux provenant de plusieurs comptes

Toutes les actions de modernisation AWS du mainframe sont enregistrées CloudTrail et documentées dans le manuel AWSMainframe Modernization API Reference. Par exemple, les appels au`CreateApplication`, `CreateEnvironment` et les `CreateDeployment` actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou d'utilisateur root.
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été effectuée par un autre service AWS.

Pour de plus amples informations, veuillez consulter l'élément userIdentity CloudTrail .

Comprendre les AWS entrées des fichiers journaux de modernisation des mainframes

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'`CreateApplication`action.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
```

```
        "principalId": "AROAII6WZTHGYAEXAMPLE",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAII6WZTHGYAEXAMPLE",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-06-01T20:38:22Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-06-01T20:40:39Z",
    "eventSource": "m2.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.196.65",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:91.0) Gecko/20100101
 Firefox/91.0",
    "requestParameters": {
        "clientToken": "1abc23de-f45g-6789-h01i-jkl2m3456789",
        "name": "MyApp",
        "description": "",
        "engineType": "microfocus",
        "definition": {
            "content": "{}"
        },
        "tags": {}
    },
    "responseElements": {
        "applicationVersion": 1,
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "applicationArn": "arn:aws:m2:us-east-1:444455556666:app/
lsfhmwhw7fffrosff2lncwqcua",
        "applicationId": "lsfhmwhw7fffrosff2lncwqcua"
    },
```

```
    "requestID": "36982d38-fcde-4bfe-a89a-7bd78d43c926",
    "eventID": "d7f0fc36-46ae-4157-9a79-c79f385fda98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management"
}
```

Résolution des problèmes

Utilisez les informations de cette section pour vous aider à résoudre les erreurs courantes dans les applications de modernisation des AWS mainframes et les environnements d'exécution utilisant à la fois les moteurs AWS Blu Age et Micro Focus.

Rubriques

Erreur : expiration du délai d'attente pour que le nom du jeu de données soit déverrouillé

- Moteur : AWS Blu Age
- Composant : Blusam

Si vous voyez cette erreur dans les CloudWatch journaux Amazon d'une application de modernisation de AWS mainframe utilisant le moteur AWS Blu Age et exécutée dans un environnement utilisant le modèle de haute disponibilité, cela indique qu'une autre application bloque un ensemble de données partagé. Généralement, cette situation se produit si l'autre application se bloque ou échoue et ne libère pas le verrou.

Comment se produit cette erreur

L'application `example-app-1` tente de verrouiller un enregistrement `example-record-1` pour une opération d'écriture. Cette opération crée à la fois un verrou sur le jeu de données`example-dataset-1`, qui en est propriétaire`example-record-1`, et un verrou sur `example-record-1` lui-même. Maintenant`example-app-2`, une autre application essaie de verrouiller le même enregistrement`example-record-1`. L'ensemble de données et l'enregistrement étant déjà verrouillés, il `example-app-2` attend que le verrou soit relâché. En cas de `example-app-1` panne, le verrouillage bloqué sur l'ensemble de données existe `example-dataset-1` toujours, ce qui entraîne l'annulation de sa tentative `example-app-2` d'écriture et le déclenchement d'une exception de délai d'expiration. Cette situation de blocage empêche toutes les applications d'y accéder. `example-dataset-1`

Comment savez-vous si c'est votre situation ?

Recherchez une application défaillante et vérifiez si elle utilise le même ensemble de données mentionné dans le message d'erreur. Vérifiez si l'application s'exécute dans un environnement d'exécution avec le modèle de haute disponibilité. L'application qui a déclenché l'exception de délai d'expiration ne peut pas continuer et affiche le `Failed` statut.

Que peux-tu faire ?

Pour résoudre le problème immédiatement, vous pouvez forcer le déverrouillage. Pour éviter qu'une situation similaire ne se reproduise à l'avenir, vous pouvez configurer deux paramètres qui contrôlent le mécanisme de réparation auto Blusam.

Forcer le verrou à le relâcher

Le gestionnaire de verrous Blusam utilise Amazon ElastiCache pour Redis pour fournir des verrous partagés entre les applications. Pour débloquer les verrous ElastiCache, utilisez l'utilitaire Redis CLI. Vous ne pouvez pas supprimer un verrouillage d'enregistrement individuel. Vous devez supprimer tous les verrous du jeu de données propriétaire. Procédez comme suit :

1. Connectez-vous à votre à ElastiCache l'aide de la commande suivante :

   ```
   redis-cli -h hostname -p port
   ```

   Vous trouverez les informations vous concernant ElastiCache dans la ElastiCache console à l'adresse https://console.aws.amazon.com/elasticache/.
2. Saisissez votre mot de passe.
3. Entrez la commande que vous souhaitez exécuter, comme suit :

Command	Objectif
KEYS *	Récupérez toutes les clés existantes.
CLÉS * *YOUR_DATASET_NAME*	Obtenez une clé de verrouillage du jeu de données.
DEL *THE_RETURNED_KEY*	Supprimez un verrou de jeu de données.

Command	Objectif
FLUSHDB	Nettoyez tout le Redis. **Warning** Toutes les données du cache Redis seront perdues. Si le Redis est utilisé à d'autres fins, telles que la gestion de sessions HTTP, vous ne voudrez peut-être pas l'utiliserFLUSHDB.

Configurer le mécanisme de réparation auto Blusam

Le gestionnaire de verrous Blusam inclut un mécanisme de réparation automatique pour empêcher les blocages sur les ensembles de données ou les enregistrements. Vous pouvez ajuster les paramètres suivants dans la définition de l'application (`application-main.yml`) pour configurer le mécanisme de réparation automatique :

- `locksDeadTime`: fait référence à la durée maximale pendant laquelle une application peut maintenir un verrou. Lorsque ce délai est écoulé, le verrou est déclaré expiré et relâché immédiatement. La `locksDeadTime` valeur est exprimée en millisecondes et la valeur par défaut est 1000.
- `locksCheck`: définit la stratégie du gestionnaire de verrous Blusam pour vérifier les verrous. Tous les verrous Blusam ElastiCache sont horodatés et ont une date d'expiration. La valeur du `locksCheck` paramètre détermine si les verrous expirés sont supprimés.
  - `off`: aucune vérification n'est exécutée à aucun moment. Des blocages peuvent survenir. (Non recommandé)
  - `reboot`: les vérifications sont exécutées lorsqu'une instance d'application AWS Mainframe Modernization exécutée dans un environnement d'exécution AWS Mainframe Modernization est démarrée ou redémarrée. Tous les verrous expirés sont immédiatement libérés. (Par défaut)
  - `timeout`: les vérifications sont exécutées lorsqu'une instance de l'application AWS Mainframe Modernization exécutée dans un environnement d'exécution AWS Mainframe Modernization est démarrée ou redémarrée, ou lorsqu'un délai expire lors d'une tentative de verrouillage d'un ensemble de données. Les verrous expirés sont immédiatement libérés.

Pour plus d'informations sur la définition d'une application AWS Blu Age, consultezAWS Exemple de définition d'application Blu Age.

Gestionnaire de serrures Blusam

Dans le contexte d'un environnement d'exécution de modernisation des AWS mainframes utilisant le modèle de haute disponibilité, une application AWS Blu Age peut être déployée plusieurs fois. Pour les applications qui gèrent des ensembles de données Blusam, des problèmes d'accès simultanés peuvent survenir. Le gestionnaire de verrous Blusam garantit l'intégrité des données et gère l'accès en lecture et en écriture aux enregistrements et aux ensembles de données en fournissant des verrous partagés entre les applications qui les utilisent. ElastiCache Ce mécanisme permet à plusieurs applications de lire l'enregistrement simultanément et garantit qu'une seule application à la fois écrit l'enregistrement.

Serrures d'écriture

Pour mettre à jour ou supprimer un enregistrement spécifique, l'application doit d'abord verrouiller l'ensemble de données propriétaire de l'enregistrement, puis verrouiller l'enregistrement lui-même. Lorsque l'enregistrement est verrouillé, le verrouillage du jeu de données est libéré et les autres enregistrements du même ensemble de données peuvent être utilisés. Lorsque l'opération de mise à jour ou de suppression est terminée, le verrouillage des enregistrements conservés est libéré. Une seule application à la fois peut mettre à jour l'enregistrement, ce qui empêche les autres applications de lire ou d'écrire jusqu'à ce que le verrou soit relâché, si la politique d'application définie autorise l'attente de publication.

Lire les verrous

Tant qu'aucun verrou d'écriture n'est maintenu sur l'enregistrement ou le jeu de données, plusieurs applications peuvent lire les mêmes enregistrements en même temps. Pour verrouiller un enregistrement pour une opération d'écriture, tous les verrous de lecture doivent être libérés.

> **Note**
>
> Le gestionnaire de verrous Blusam gère l'accès depuis plusieurs threads dans une application donnée en utilisant le même mécanisme de verrouillage.

Impossible d'accéder à l'URL d'une application

- Moteur : AWS Blu Age et Micro Focus
- Composant : applications

Si vous ne pouvez pas accéder à l'URL d'une application de modernisation du AWS mainframe en cours d'exécution que vous avez créée et déployée dans un environnement d'exécution AWS Mainframe Modernization, vous devrez peut-être configurer les règles entrantes sur le groupe de sécurité que vous avez associé à l'environnement d'exécution.

Comment se produit cette erreur

Lorsque vous créez un environnement d'exécution, le groupe de sécurité que vous fournissez, y compris le groupe de sécurité par défaut, doit avoir des règles entrantes configurées pour autoriser le trafic vers les applications déployées depuis l'extérieur du VPC, si vous souhaitez autoriser ce type d'accès.

Comment savez-vous si c'est votre situation ?

L'application a démarré correctement et fonctionne normalement, mais vous ne pouvez pas vous y connecter à l'aide de son URL.

Que peux-tu faire ?

Vérifiez si le groupe de sécurité Amazon VPC associé à l'environnement d'exécution autorise le trafic vers l'environnement sur les ports d'application appropriés. Pour vérifier les règles du groupe de sécurité, procédez comme suit :

1. Ouvrez la console de modernisation du AWS mainframe à l'adresse https://console.aws.amazon.com/m2/.
2. Dans le menu de navigation de gauche, sélectionnez Environments.
3. Choisissez l'environnement d'exécution qui héberge l'application à laquelle vous souhaitez vous connecter.
4. Choisissez Configurations.
5. Dans Sécurité et réseau, choisissez le groupe de sécurité. Le lien ouvre les détails du groupe de sécurité dans la console Amazon VPC.

6. Si nécessaire, choisissez Modifier les règles entrantes et ajoutez la règle suivante si elle n'est pas déjà présente :

   Type

   TCP personnalisé

   Port

   8196 ou le port qui correspond aux propriétés de l'écouteur spécifiées dans la définition de l'application. Pour plus d'informations, consultez Étape 2 : Création de la définition de l'application.

   Source

   Adresse IP à partir de laquelle vous appelez l'application. Vous pouvez choisir MyIP dans le menu déroulant. Si le délai d'expiration persiste, essayez de choisir Anywhere IPV4 ou Anywhere IPV6. Assurez-vous d'arrêter l'application et de la redémarrer après avoir ajouté la règle entrante au groupe de sécurité.

   Pour plus d'informations, consultez la section Utiliser les règles des groupes de sécurité dans le guide de l'utilisateur Amazon VPC.

AWSBlu Insights ne s'ouvre pas depuis la console

- Moteur : AWS Blu Age
- Composant : Blu Insights

Lorsque vous essayez d'accéder à Blu Insights depuis la console AWS Mainframe Modernization, celle-ci ne s'ouvre pas et le nouvel onglet se ferme immédiatement.

Comment se produit cette erreur

Le rôle que vous utilisez pour accéder à Blu Insights ne dispose pas d'autorisations suffisantes.

Que peux-tu faire ?

Associez une politique IAM au rôle pour lui permettre d'accéder à Blu Insights. Assurez-vous que la politique inclut au moins les autorisations suivantes.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "m2:GetSignedBluinsightsUrl"
            ],
            "Resource": "arn:aws:m2:region:account:*"
            }
    ]
}
```

Assurez-vous de remplacer `region` et `account` d'utiliser le bon Région AWS etCompte AWS.

Historique des documents pour le guide de l'utilisateur sur la modernisation du AWS mainframe

Le tableau suivant décrit les versions de documentation relatives à la modernisation des AWS mainframes.

Modification	Description	Date
Tutoriel Managed Runtime mis à jour pour Micro Focus	Ce didacticiel explique comment déployer et exécuter l' CardDemo exemple d'application dans un environnement d'exécution géré AWS Mainframe Modernization avec le moteur d'exécution Micro Focus.	5 février 2024
Notes de publication pour la version 3.9.0 de AWS Blu Age Runtime and Modernization Tools.	Cette version de AWS Blu Age Runtime and Modernization Tools met l'accent sur de multiples améliorations transversales apportées au produit dans le but d'améliorer les performances des architectures à haute disponibilité, ainsi que sur de nouvelles fonctionnalités permettant d'améliorer l'exécution des tâches à un niveau supérieur.	18 décembre 2023
Transférez des fichiers entre le mainframe et AWS	Nouvelle fonctionnalité publiée pour transférer des fichiers du mainframe source versAWS.	27 novembre 2023
Gérez les transactions pour les applications	Nouvelle fonctionnalité publiée pour afficher et modifier les	16 octobre 2023

	transactions pour les applications de modernisation AWS du mainframe.	
Notes de publication pour la version 3.6.0 de AWS Blu Age Runtime and Modernization Tools.	Cette version de AWS Blu Age Runtime and Modernization Tools fournit de nouvelles fonctionnalités pour les migrations existantes vers zOS et AS400, principalement destinées à étendre les mécanismes de support CICS, à compléter les fonctionnalités JCL, à optimiser les performances des fonctionnalités simultanées et à volume élevé, et à ajouter des fonctionnalités. multi-data-source	4 août 2023
Vous pouvez désormais déployer une nouvelle version d'une application lorsque celle-ci est arrêtée.	Auparavant, pour déployer une nouvelle version d'une application, vous deviez supprimer la version déployée. Vous pouvez maintenant simplement arrêter la version déployée et déployer une nouvelle version.	26 juillet 2023
AWSPackage d'environnement d'exécution Blu Age pour faciliter le déploiement d'Amazon EC2	AWSLa modernisation du mainframe avec le runtime AWS Blu Age est désormais disponible avec plus de flexibilité pour configurer la pile complète et le déploiement sur les instances Amazon EC2 de votre ordinateur. Compte AWS	6 juillet 2023

Connexion unique à AWSAWS Blu Age Blu Insights.	AWSAWSBlu Age Blu Insights est disponible AWS Management Console via l'authentification unique.	31 mars 2023
Version GA	Publication générale du guide de l'utilisateur sur la modernisation du AWS mainframe.	8 juin 2022
Première version	Publication initiale (version préliminaire publique) du guide de l'utilisateur sur la modernisation du AWS mainframe.	30 novembre 2021

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.