Guide de l’utilisateur

AWS Amplify Hébergement

AWS Amplify Hébergement: Guide de l’utilisateur

Table of Contents

Bienvenue chez AWS Amplify Hosting

AWS Amplifyest un ensemble d'outils et de fonctionnalités spécialement conçus qui permettent aux développeurs Web et mobiles frontaux de créer rapidement et facilement des applications complètes sur. AWS Amplify propose deux services : Amplify Hosting et Amplify Studio. Amplify Hosting fournit un flux de travail basé sur git pour héberger des piles complètes d'applications Web sans serveur avec déploiement continu. Ce guide de l'utilisateur fournit les informations dont vous avez besoin pour démarrer avec Amplify Hosting.

Fonctionnalités d'Amplify Hosting

- Amplify Hosting prend en charge les frameworks SPA courants, par exemple React, Angular, Vue.js, Ionic et Ember, ainsi que les générateurs de sites statiques tels que Gatsby, Eleventy, Hugo et Jekyll. VuePress
- Gérez les environnements de production et de préparation pour votre frontend et votre backend en connectant de nouvelles succursales. Voir, proposer des déploiements dans des succursales.
- Connectez votre application à un domaine personnalisé. Voir, configurer des domaines personnalisés.
- Déployez et hébergez des applications Web SSR. Amplify Hosting détecte automatiquement les applications créées à l'aide du framework Next.js.

  Amplify prend également en charge tout framework SSR basé sur Javascript avec un adaptateur de build open source qui transforme la sortie de compilation d'une application en la structure de répertoire attendue par Amplify Hosting. Un adaptateur est disponible pour déployer une application Nuxt sur Amplify.
- Prévisualisez les modifications lors des révisions de code en configurant des aperçus par pull request.
- Améliorez la qualité de votre application grâce à des tests de bout en bout. Tu vois, end-to-end tester.
- Protégez votre application web par un mot de passe pour pouvoir utiliser de nouvelles fonctions sans les rendre accessibles publiquement. Vous voyez, restreindre l'accès.
- Configurez des réécritures et des redirections pour maintenir les classements SEO et acheminer le trafic en fonction des exigences de votre application cliente. Vous voyez, utiliser les redirections.

- Les déploiements atomiques éliminent les fenêtres de maintenance en garantissant que l'application Web n'est mise à jour qu'une fois le déploiement complet terminé. Cela permet d'éliminer les scénarios dans lesquels les fichiers ne sont pas correctement chargés.

Commencer à utiliser Amplify Hosting

Pour commencer à utiliser les fonctionnalités d'hébergement d'Amplify, consultez le Commencer avec le code existant didacticiel. Après avoir terminé le didacticiel, vous serez en mesure de connecter votre dépôt git (GitHub, BitBucket Cloud GitLab, etAWS CodeCommit) pour configurer un déploiement continu. Vous pouvez également commencer avec l'un des exemples de déploiement continu Fullstack.

Amplify Studio

Vous pouvez accéder à Amplify Studio depuis la AWS Amplify console du. AWS Management Console Amplify Studio est un environnement de développement visuel qui simplifie la création de piles complètes d'applications Web et mobiles évolutives. Utilisez Studio pour créer votre interface utilisateur frontale à l'aide d'un ensemble de composants d' ready-to-use interface utilisateur, créer un backend d'application, puis connecter les deux ensemble. Consultez le guide de l'utilisateur d'Amplify Studio dans la documentation d'Amplify.

Fonctionnalités d'Amplify Studio

- La modélisation visuelle des données vous permet de vous concentrer sur les objets spécifiques à votre domaine plutôt que sur l'infrastructure cloud.
- Configurez l'authentification pour votre application.
- Autorisation puissante et facile à comprendre.
- Il nfrastructure-as-code configure toutes les fonctionnalités du backend avec. AWS CloudFormation
- Fonctionne avec l'interface de ligne de commande (CLI) Amplify. Toutes les mises à jour que vous effectuez dans Studio peuvent être intégrées dans la CLI.
- Invitez les utilisateurs par e-mail à configurer et à gérer le backend. Ces utilisateurs pourront également se connecter à la CLI Amplify avec leur adresse e-mail.
- Gestion de contenu avec support Markdown.
- Gérez les utilisateurs et les groupes de votre application.

- Utilisez le concepteur visuel de Studio pour créer des composants d'interface utilisateur frontale. Choisissez parmi des dizaines de modèles dans la bibliothèque de composants d'interface utilisateur prédéfinie.
- Importez des prototypes Figma créés par des designers dans Studio sous forme de code React.
- Personnalisez l'interface utilisateur de votre interface utilisateur avec des thèmes pour appliquer des styles globaux aux composants de votre application.
- Configurez et testez les composants de votre interface utilisateur directement dans Studio pour voir comment ils mettent à jour et affichent les données.
- Liez votre backend connecté au cloud à l'interface utilisateur de votre frontend en quelques étapes simples.

Commencer à utiliser Amplify Studio

Vous n'avez pas besoin de AWS compte pour commencer à utiliser Studio afin de créer un backend. Sans AWS compte, vous pouvez commencer à modéliser les données pour votre backend localement.

Avec un AWS compte, vous avez accès à un ensemble étendu de fonctionnalités de Studio pour gérer votre environnement principal, ainsi qu'au concepteur visuel pour créer des composants d'interface utilisateur que vous pouvez connecter au backend de votre application. Pour plus d'informations, consultez la section Mise en route dans la documentation Amplify.

Applications Web SPA modernes

Ce guide de l'utilisateur est destiné aux clients qui ont une connaissance de base des applications Web modernes d'une seule page (SPA). Les applications Web modernes sont conçues comme des SPA qui regroupent tous les composants de l'application dans des fichiers statiques. Les architectures Web client-serveur traditionnelles entraînaient des expériences médiocres ; chaque clic sur un bouton ou chaque recherche nécessitait un aller-retour vers le serveur, ce qui entraînait un nouveau rendu de l'application dans son intégralité. Les applications Web modernes offrent une expérience utilisateur native similaire à celle d'une application en fournissant efficacement le frontend de l'application, ou interface utilisateur, aux navigateurs sous forme de JavaScript fichiers HTML/ prédéfinis qui peuvent ensuite invoquer les fonctionnalités du backend sans recharger la page.

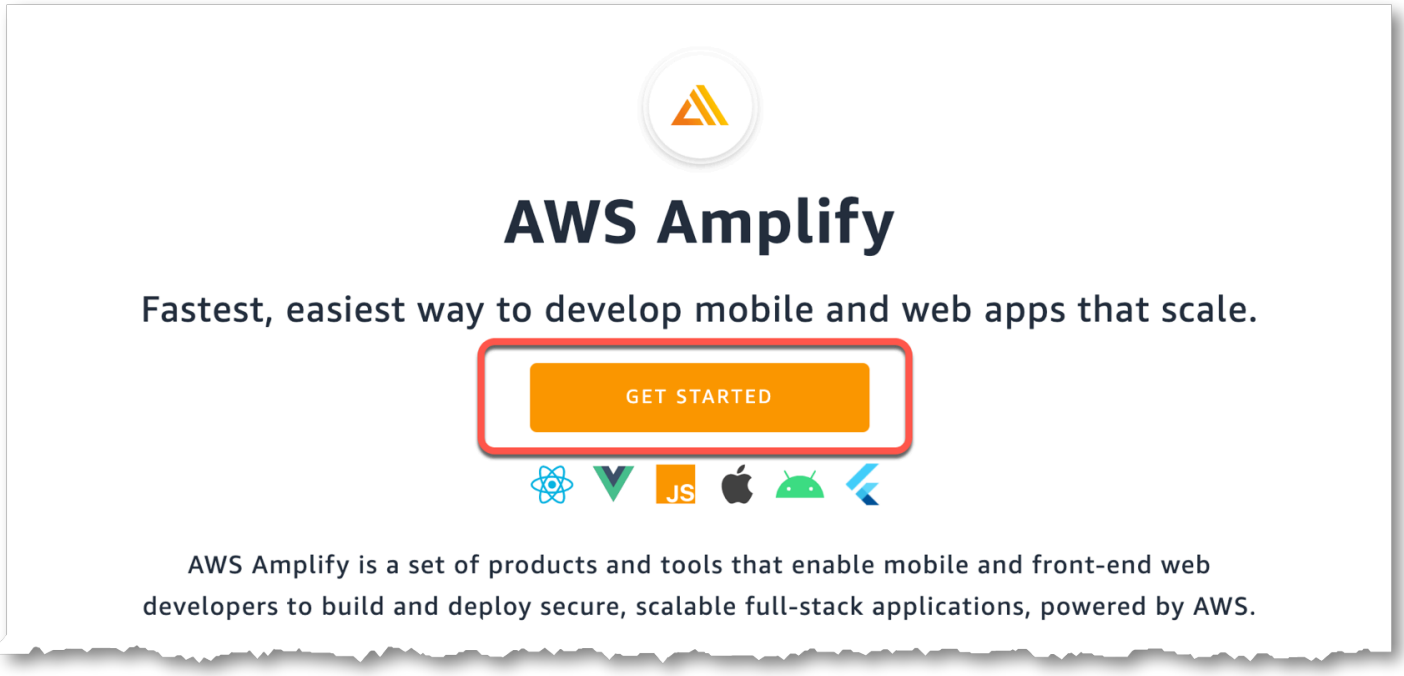
Les fonctionnalités d'une application Web moderne sont souvent réparties sur plusieurs sites, tels que les bases de données, les services d'authentification, le code frontal exécuté dans le navigateur

et la logique ou les AWS Lambda fonctions métier du backend exécutées dans le cloud. Cela rend les déploiements d'applications complexes et fastidieux, car les développeurs doivent soigneusement coordonner les déploiements sur le frontend et le backend pour éviter des déploiements partiels ou des échecs. Amplify simplifie le déploiement du frontend et du backend dans un seul flux de travail.

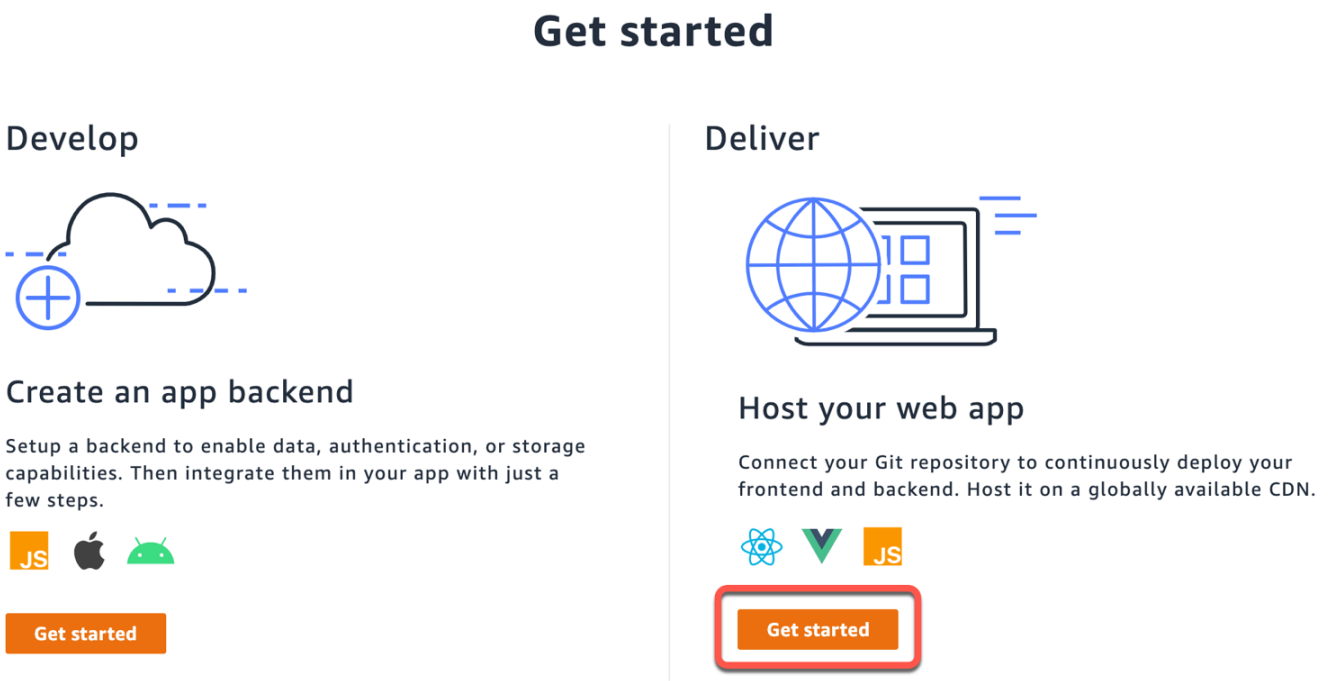
Commencer avec le code existant

Dans cette procédure, vous découvrez comment créer, déployer et héberger en continu une application web moderne. Les applications Web modernes incluent des frameworks d'applications à page unique (SPA) (par exemple, React, Angular ou Vue) et des générateurs de sites statiques (SSG) (par exemple, Hugo, Jekyll ou Gatsby). Amplify Hosting prend également en charge les applications Web qui utilisent le rendu côté serveur (SSR) et sont créées à l'aide de Next.js.

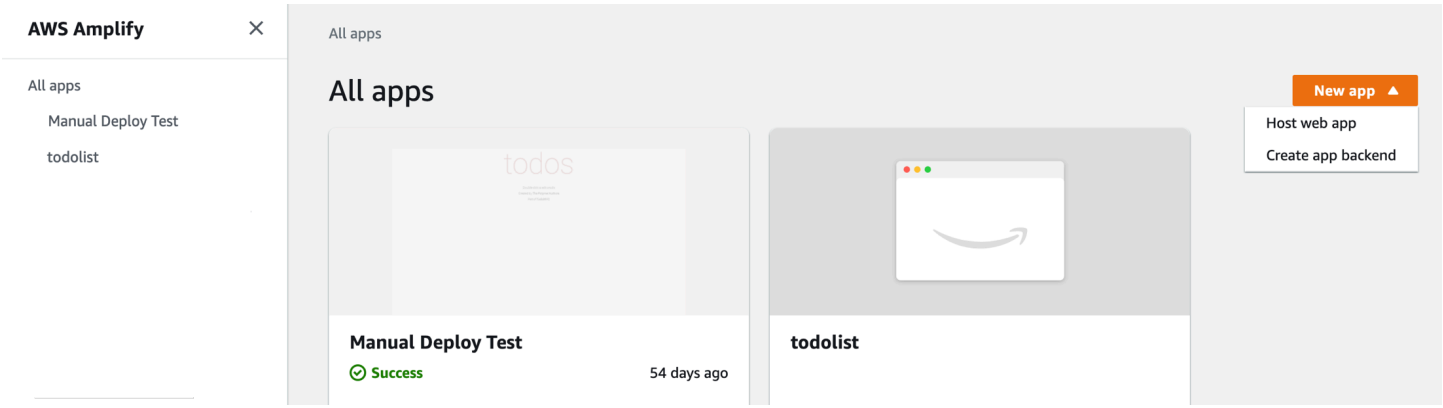
Pour commencer, connectez-vous à la console Amplify. Si vous partez de la page d'AWS Amplifyaccueil, choisissez Get Started en haut de la page.



Choisissez ensuite Commencer sous Deliver.

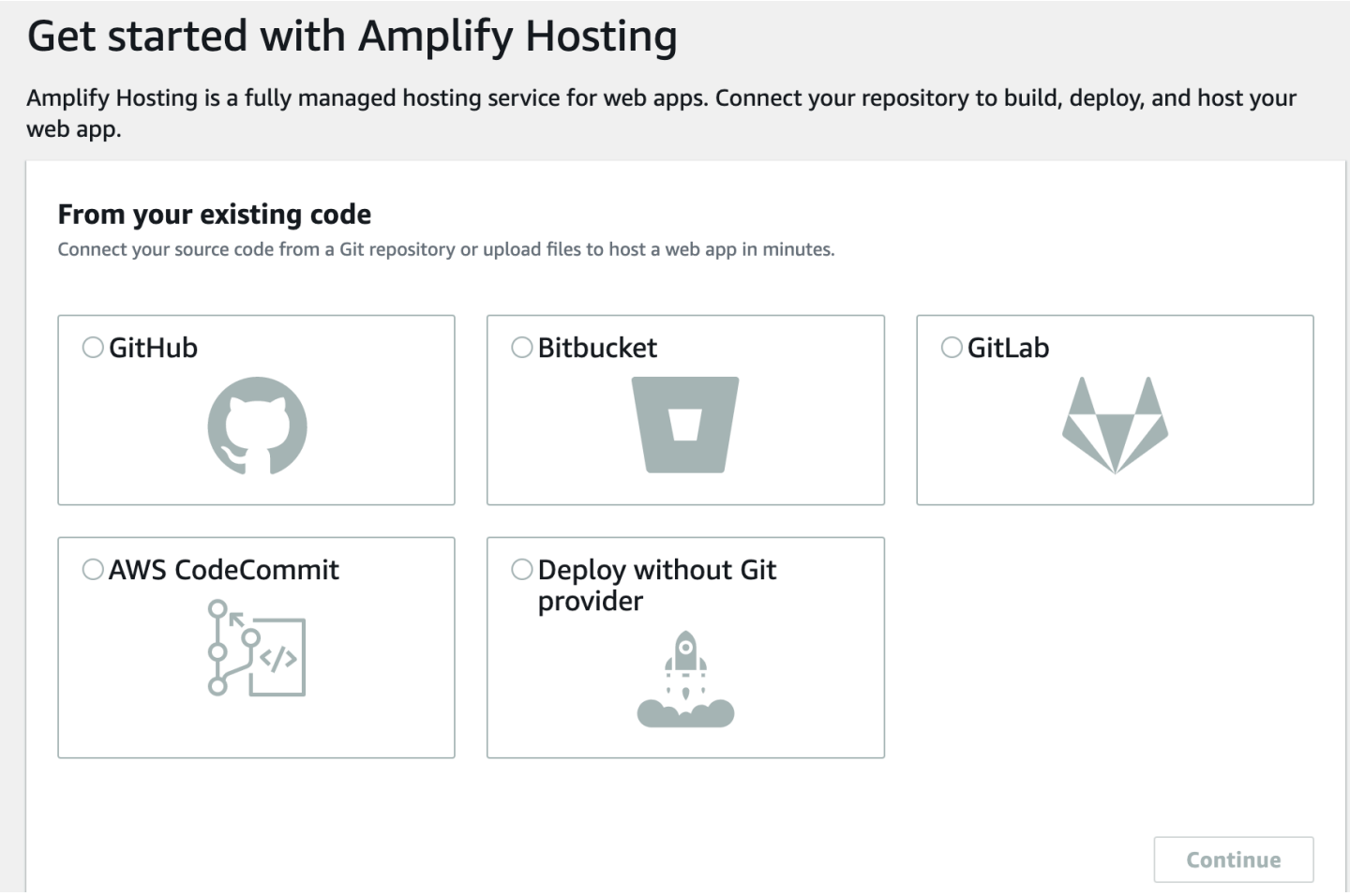


Si vous partez de la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web dans le coin supérieur droit.



Étape 1 : Connecter un dépôt

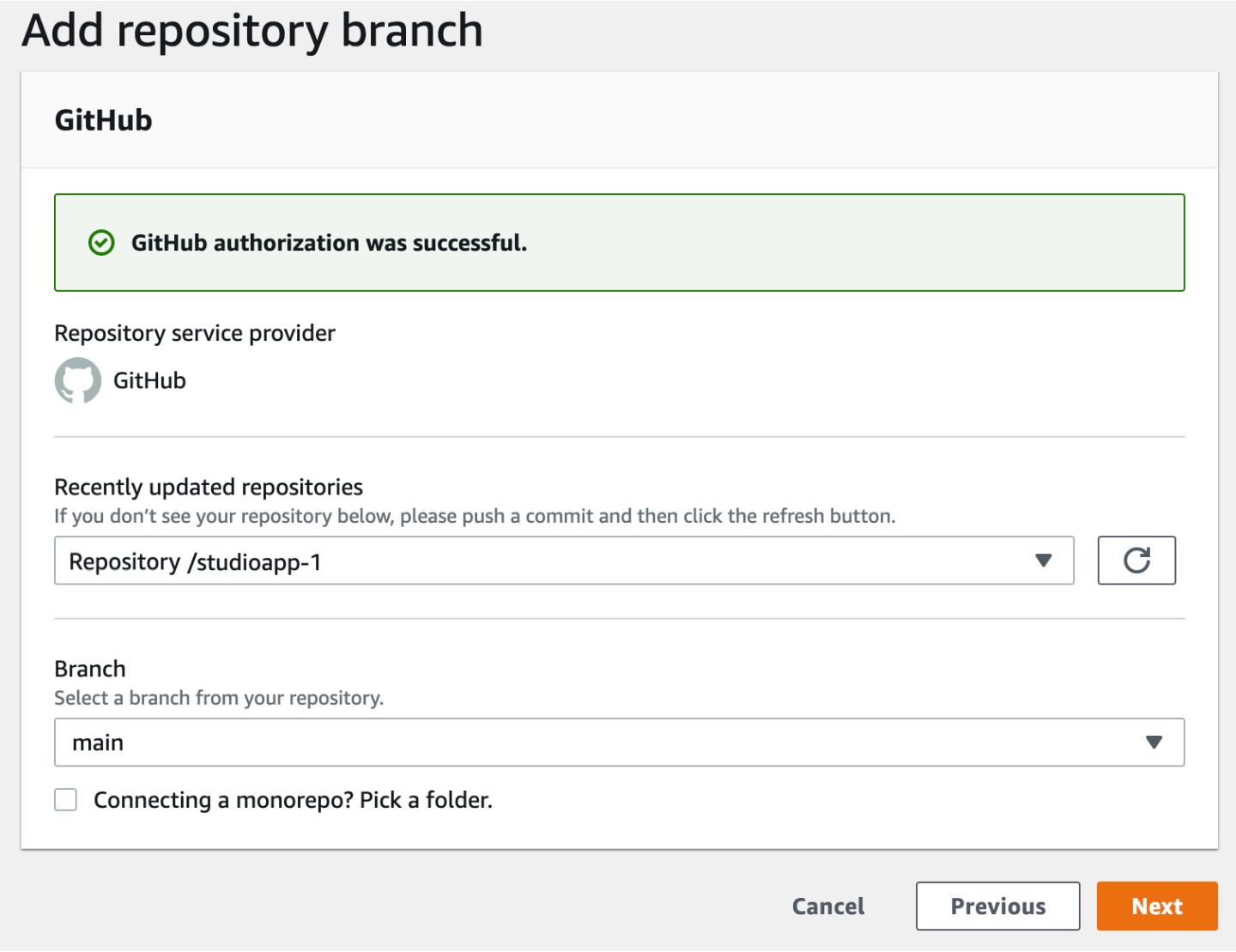
Connectez votre GitHub, Bitbucket ou votre AWS CodeCommit dépôt. GitLab Vous avez également la possibilité de télécharger manuellement vos artefacts de build sans connecter un dépôt Git. Pour plus d'informations, consultez la section Déploiements manuels.



Après avoir autorisé la console Amplify avec Bitbucket, or GitLab, AWS CodeCommit Amplify récupère un jeton d'accès auprès du fournisseur de référentiel, mais ne le stocke pas sur les serveurs. AWS Amplify accède à votre référentiel à l'aide de clés de déploiement installées dans un référentiel spécifique.

Pour les GitHub référentiels, Amplify utilise désormais la fonctionnalité Apps pour autoriser GitHub l'accès à Amplify. Avec l' GitHub application Amplify, les autorisations sont mieux affinées, ce qui vous permet d'accorder à Amplify l'accès uniquement aux référentiels que vous spécifiez. Pour plus d'informations sur l'installation et l'autorisation de l' GitHub application, consultezConfiguration de l'accès Amplify aux GitHub référentiels.

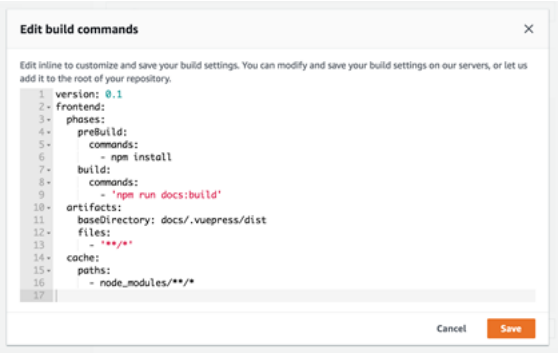
Une fois que vous avez connecté le fournisseur de services de référentiel, choisissez un référentiel, puis choisissez une branche correspondante à créer et déployer.



Étape 2a : Confirmer les paramètres de construction pour le front-end

Pour la branche sélectionnée, Amplify inspecte votre référentiel afin de détecter automatiquement la séquence de commandes de compilation à exécuter.

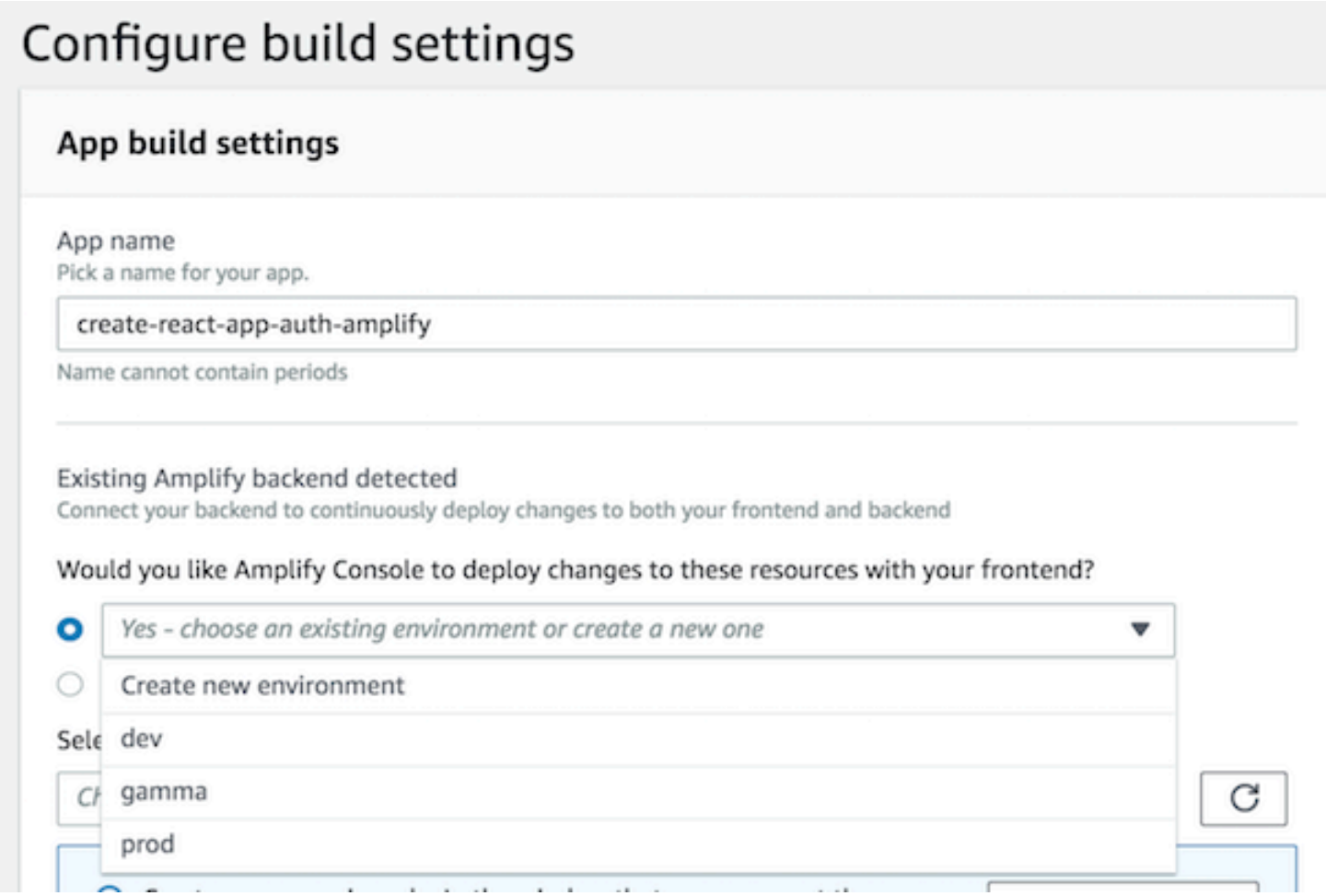
Important : vérifiez que les commandes de création de build et le répertoire de sortie de génération (à savoir, artifacts > baseDirectory) sont exacts. Si vous devez modifier ces informations, choisissez Modifier pour ouvrir l'éditeur YML. Vous pouvez enregistrer vos paramètres de compilation sur nos serveurs ou télécharger le code YML et l'ajouter à la racine de votre dépôt (pour monorepos, stockez le code YML dans le répertoire racine de l'application).



Pour plus d'informations, consultez la section Syntaxe YAML des spécifications de construction.

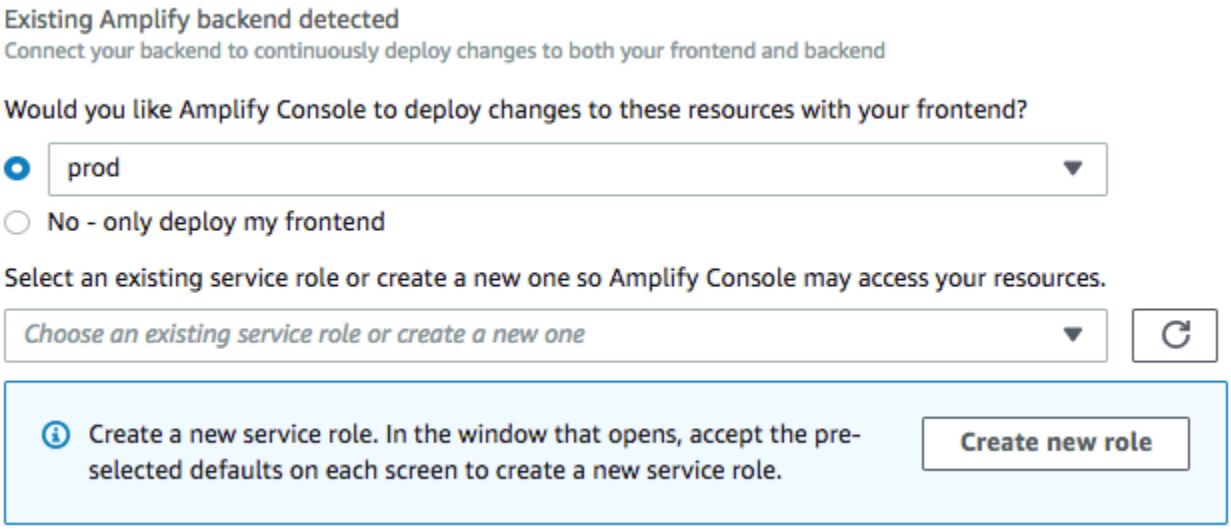
Étape 2b : Confirmer les paramètres de construction pour le backend

Si vous avez connecté un référentiel provisionné par la CLI Amplify v1.0+ (exécutez amplify -v pour trouver la version de la CLI), Amplify Hosting déploiera ou mettra automatiquement à jour les ressources du backend (toute ressource fournie par l'Amplify CLI) dans un seul flux de travail avec la version du frontend. Vous pouvez choisir de renvoyer un environnement backend existant vers une branche ou de créer un autre environnement. Pour un step-by-step didacticiel, consultez Getting started with fullstack déploiements.



Pour déployer des fonctionnalités de backend à l'aide de la CLI Amplify lors de votre génération, créez ou AWS Identity and Access Management réutilisez un rôle de service (IAM). Les rôles IAM constituent un moyen sécurisé d'accorder à Amplify les autorisations nécessaires pour agir sur les ressources de votre compte. Pour obtenir des instructions complètes, veuillez consulter Ajouter un rôle de service.

Remarque : La CLI Amplify ne s'exécute pas si un rôle de service IAM n'est pas activé.



Étape 2c : Ajouter des variables d'environnement (facultatif)

Toutes les applications ou presque ont besoin d'informations de configuration au moment de l'exécution. Ces configurations peuvent être des détails de connexion de base de données, des clés d'API ou différents paramètres. Les variables d'environnement permettent d'exposer ces configurations au moment de la création.

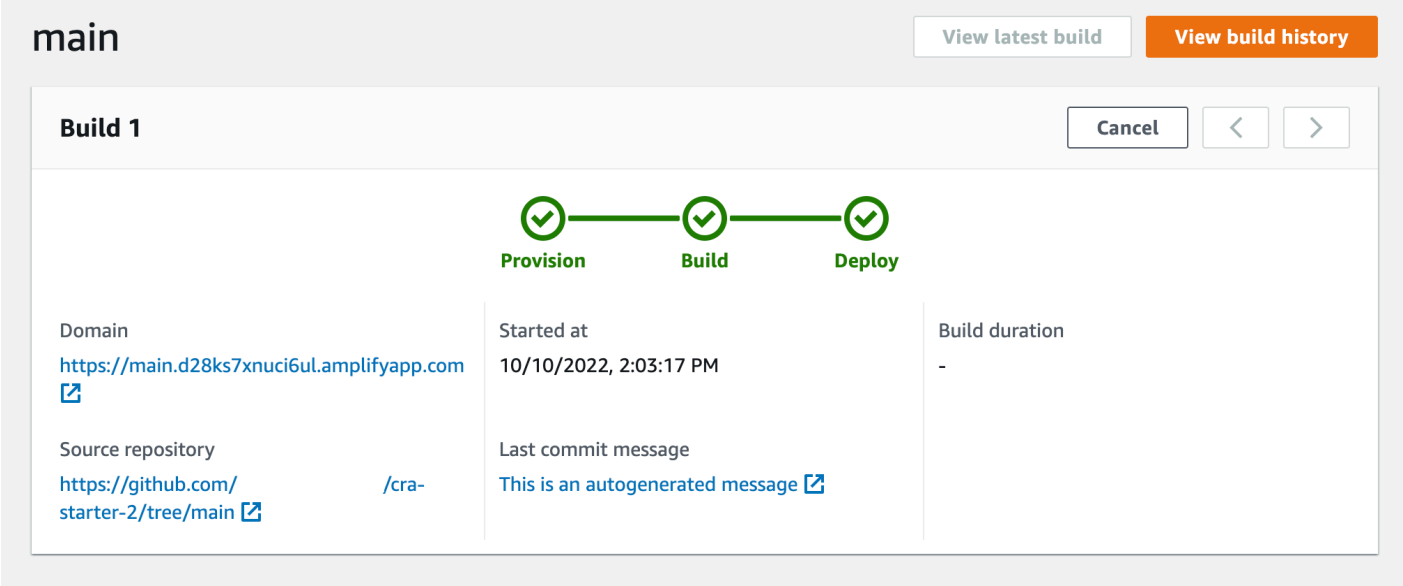
Étape 3 : enregistrer et déployer

Passez en revue tous vos paramètres pour vous assurer que tout est configuré correctement. Choisissez Enregistrer et déployer pour déployer votre application Web sur le réseau AWS mondial de diffusion de contenu (CDN). La création de votre interface prend généralement 1 à 2 minutes, mais cela peut varier en fonction de la taille de l'application.

Accédez à l'écran des journaux de construction en choisissant un indicateur de progression dans la section branche. Une génération comporte les étapes suivantes :

1. Mise en service : votre environnement de génération est configuré à l'aide d'une image Docker sur un hôte avec 4 vCPU, 7 Go de mémoire. Chaque génération dispose d'une instance hôte propre, garantissant ainsi que toutes les ressources sont bien isolées. Le contenu du fichier Docker est affiché pour s'assurer que l'image par défaut prend en charge vos exigences.
2. Création : la phase de génération se compose de trois étapes : configuration (clone le référentiel dans le conteneur), déploiement du backend (exécute l'interface de ligne de commande Amplify pour déployer des ressources backend) et génération du frontend (crée vos artefacts frontend).

3. Déploiement - Lorsque la construction est terminée, tous les artefacts sont déployés dans un environnement d'hébergement géré par Amplify Hosting. Vous pouvez consulter votre application sur le `amplifyapp.com` domaine. Chaque déploiement est atomique : le déploiement atomique permet d'éliminer la fenêtre de maintenance en garantissant que la mise à jour de l'application web ne s'effectue qu'une fois le déploiement terminé.



> **Note**
>
> Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Étapes suivantes

- Ajouter un domaine personnalisé à votre application
- Gérer plusieurs environnements
- Prévisualisez les pull requests avant de les fusionner

Démarrez avec les déploiements continus Fullstack

Amplify Hosting permet aux développeurs qui créent des applications avec le framework Amplify de déployer en permanence des mises à jour de leur backend et de leur frontend à chaque validation de code. Avec Amplify Hosting, vous pouvez déployer des backends sans serveur avec les API GraphQL/REST, l'authentification, l'analyse et le stockage, créés à l'aide d'Amplify Studio, sur le même commit que votre code d'interface.

Dans ce didacticiel, vous allez configurer un flux de travail CI/CD complet avec Amplify. Vous allez déployer une application frontale sur Amplify Hosting. Ensuite, vous allez créer un backend à l'aide d'Amplify Studio. Enfin, vous allez connecter le backend cloud à l'application frontale.

Rubriques

Prérequis

Avant de commencer ce didacticiel, vous devez effectuer les opérations suivantes :

- Inscrivez-vous pour un Compte AWS. Ouvrez https://portal.aws.amazon.com/billing/signup#/start/email pour commencer.
- Créez un compte auprès d'un fournisseur de dépôt git GitHub, tel que Bitbucket ou AWS CodeCommit. GitLab
- Installez l'interface de ligne de commande (CLI) Amplify. Pour obtenir des instructions, voir Installer la CLI Amplify dans la documentation Amplify Framework.

Étape 1 : Déployer un frontend

Si vous avez une application frontale existante dans un dépôt git que vous souhaitez utiliser pour cet exemple, vous pouvez suivre les instructions de déploiement d'une application frontale.

Si vous devez créer une nouvelle application frontale à utiliser pour cet exemple, vous pouvez suivre les instructions de création d'application React dans la documentation de création d'application React.

Pour déployer une application frontale

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web dans le coin supérieur droit.
3. Sélectionnez votre fournisseur GitHub, Bitbucket ou de AWS CodeCommit dépôt GitLab, puis choisissez Continuer.
4. Amplify autorise l'accès à votre dépôt git. Pour les GitHub référentiels, Amplify utilise désormais la fonctionnalité Apps pour autoriser GitHub l'accès à Amplify.

   Pour plus d'informations sur l'installation et l'autorisation de l' GitHub application, consultezConfiguration de l'accès Amplify aux GitHub référentiels.
5. Sur la page Ajouter une branche de référentiel, procédez comme suit :

   a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.

   b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.

   c. Choisissez Suivant.
6. Sur la page Configurer les paramètres de build, choisissez Next.
7. Sur la page Révision, choisissez Enregistrer et déployer. Lorsque le déploiement est terminé, vous pouvez afficher votre application sur le domaine `amplifyapp.com` par défaut.

> **Note**
>
> Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Étape 2 : créer un backend

Maintenant que vous avez déployé une application frontale sur Amplify Hosting, vous pouvez créer un backend. Utilisez les instructions suivantes pour créer un backend avec une base de données simple et un point de terminaison d'API GraphQL.

Pour créer un backend

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, sélectionnez l'application que vous avez créée à l'étape 1.
3. Sur la page d'accueil de l'application, choisissez l'onglet Environnements principaux, puis choisissez Commencer. Cela lance le processus de configuration d'un environnement intermédiaire par défaut.
4. Une fois la configuration terminée, choisissez Launch Studio pour accéder à l'environnement principal de mise en scène dans Amplify Studio.

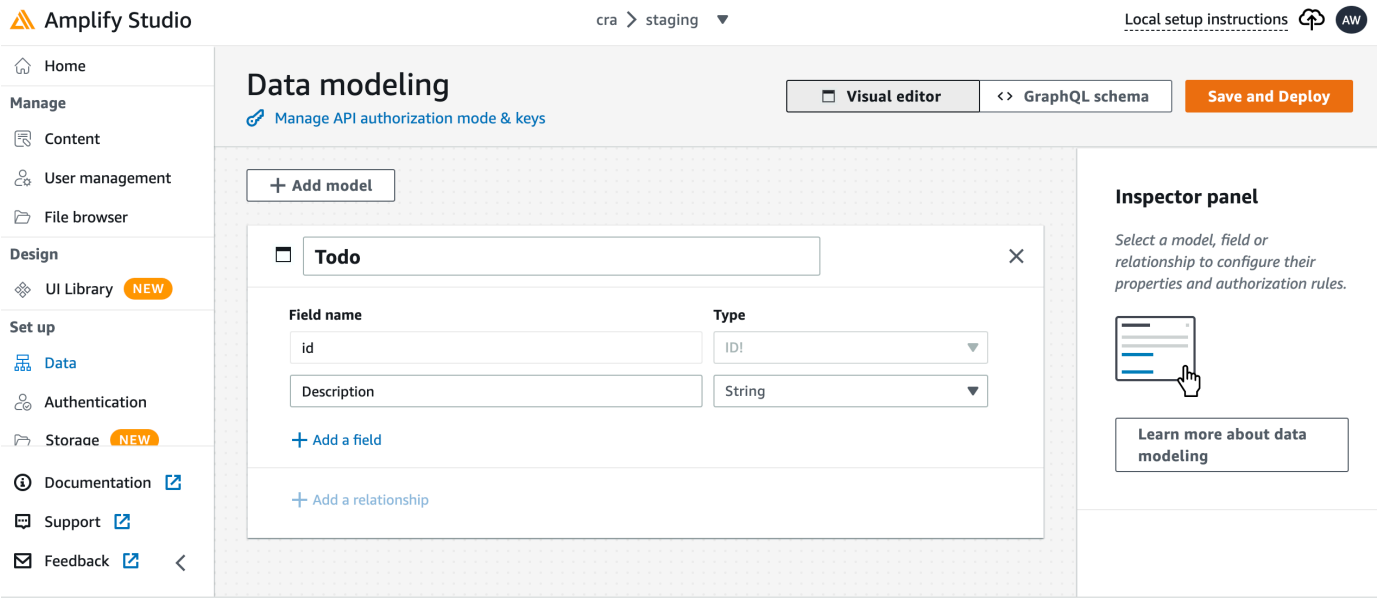
Amplify Studio est une interface visuelle permettant de créer et de gérer votre backend et d'accélérer le développement de votre interface utilisateur frontale. Pour plus d'informations sur Amplify Studio, consultez la documentation d'Amplify Studio.

Utilisez les instructions suivantes pour créer une base de données simple à l'aide de l'interface de création visuelle de backend d'Amplify Studio.

Création d'un modèle de données

1. Sur la page d'accueil de l'environnement intermédiaire de votre application, choisissez Create data model. Cela ouvre le concepteur de modèles de données.
2. Sur la page Modélisation des données, choisissez Ajouter un modèle.
3. Pour le titre, entrez**`Todo`**.
4. Choisissez Ajouter un champ.
5. Dans Nom du champ, entrez**`Description`**.

   La capture d'écran suivante est un exemple de l'apparence de votre modèle de données dans le concepteur.



6. Choisissez Enregistrer et déployer.
7. Retournez à la console Amplify Hosting et le déploiement de l'environnement de préparation sera en cours.

Pendant le déploiement, Amplify Studio crée toutes les AWS ressources requises dans le backend, notamment une API AWS AppSync GraphQL pour accéder aux données et une table Amazon DynamoDB pour héberger les éléments Todo. Amplify utilise AWS CloudFormation pour déployer votre backend, ce qui vous permet de stocker votre définition de backend sous forme de. infrastructure-as-code

Étape 3 : Connectez le backend au frontend

Maintenant que vous avez déployé un frontend et créé un backend cloud contenant un modèle de données, vous devez les connecter. Utilisez les instructions suivantes pour transférer votre définition de backend vers votre projet d'application local à l'aide de la CLI Amplify.

Pour connecter un backend cloud à un frontend local

1. Ouvrez une fenêtre de terminal et accédez au répertoire racine de votre projet local.
2. Exécutez la commande suivante dans la fenêtre du terminal, en remplaçant le texte rouge par l'identifiant unique de l'application et le nom de l'environnement principal de votre projet.

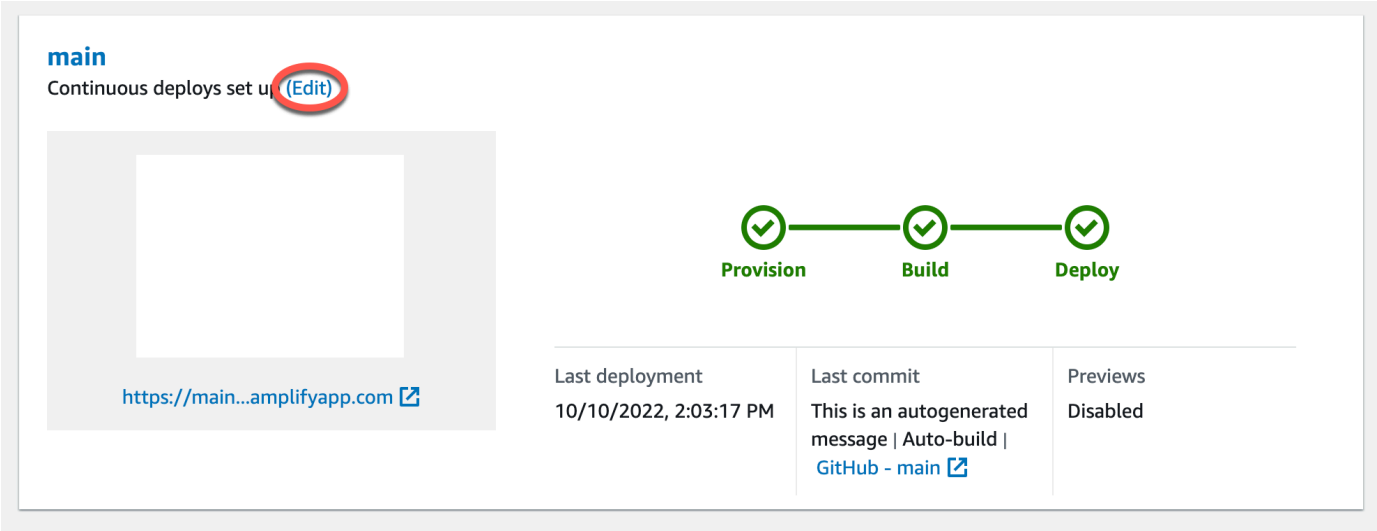
```
amplify pull --appId abcd1234 --envName staging
```

3. Suivez les instructions affichées dans la fenêtre du terminal pour terminer la configuration du projet.

Vous pouvez désormais configurer le processus de création pour ajouter le backend au flux de travail de déploiement continu. Suivez les instructions suivantes pour connecter une branche frontale à un backend dans la console Amplify Hosting.

Pour connecter une branche d'application frontale et un backend cloud

1. Sur la page d'accueil de l'application, choisissez l'onglet Environnements d'hébergement.
2. Localisez la branche principale et choisissez Modifier.



3. Dans la fenêtre Modifier le backend cible, pour Environnement, sélectionnez le nom du backend à connecter. Dans cet exemple, choisissez le backend de mise en scène que vous avez créé à l'étape 2.

   Par défaut, le CI/CD full stack est activé. Décochez cette option pour désactiver le CI/CD full stack pour ce backend. La désactivation du CI/CD complet entraîne l'exécution de l'application en mode pull uniquement. Au moment de la création, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.
4. Ensuite, vous devez configurer un rôle de service pour accorder à Amplify les autorisations nécessaires pour apporter des modifications au backend de votre application. Vous pouvez utiliser un rôle de service existant ou en créer un nouveau. Pour obtenir des instructions, veuillez consulter Ajouter un rôle de service.
5. Après avoir ajouté un rôle de service, revenez à la fenêtre Modifier le backend cible et choisissez Enregistrer.

6. Pour terminer la connexion du backend intermédiaire à la branche principale de l'application frontale, effectuez une nouvelle version de votre projet.

   Effectuez l'une des actions suivantes :

   - Depuis votre dépôt git, envoyez du code pour lancer une compilation dans la console Amplify.
   - Dans la console Amplify, accédez à la page des détails de compilation de l'application et choisissez Redeploy this version.

Étapes suivantes

Configurer les déploiements de succursales de fonctionnalités

Suivez notre flux de travail recommandé pour configurer des déploiements de branches fonctionnelles avec plusieurs environnements principaux.

Création d'une interface utilisateur frontale dans Amplify Studio

Utilisez Studio pour créer votre interface utilisateur frontale à l'aide d'un ensemble de composants d' ready-to-use interface utilisateur, puis connectez-la au backend de votre application. Pour plus d'informations et des didacticiels, consultez le guide de l'utilisateur d'Amplify Studio dans la documentation d'Amplify Framework.

Déployez des applications rendues côté serveur avec Amplify Hosting

Vous pouvez l'utiliser AWS Amplify pour déployer et héberger des applications Web qui utilisent le rendu côté serveur (SSR). Amplify Hosting détecte automatiquement les applications créées à l'aide du framework Next.js et vous n'avez pas à effectuer de configuration manuelle dans le. AWS Management ConsoleAmplify prend également en charge tout framework SSR basé sur Javascript avec un adaptateur de build open source qui transforme la sortie de compilation d'une application en la structure de répertoire attendue par Amplify Hosting.

Pour savoir comment Amplify prend en charge le SSR, consultez les rubriques suivantes.

Rubriques

Qu'est-ce que le rendu côté serveur

Amplify prend en charge le déploiement et l'hébergement d'applications Web statiques créées avec des frameworks d'applications monopages (SPA) tels que React, et d'applications créées avec un générateur de site statique (SSG) tel que Gatsby. Les applications Web statiques consistent en une combinaison de fichiers, tels que du HTML, du CSS et de JavaScript fichiers, qui sont stockés sur un réseau de diffusion de contenu (CDN). Lorsqu'un navigateur client adresse une demande au site Web, le serveur renvoie une page au client avec une réponse HTTP et le navigateur client interprète le contenu et l'affiche à l'utilisateur.

Amplify prend également en charge les applications Web avec rendu côté serveur (SSR). Lorsqu'un client envoie une demande à une page SSR, le code HTML de la page est créé sur le serveur à chaque demande. Le SSR permet à un développeur de personnaliser un site Web par demande et

par utilisateur. En outre, le SSR peut améliorer les performances et l'optimisation pour les moteurs de recherche (SEO) d'un site Web.

Amplify la prise en charge des frameworks SSR

Amplify Hosting prend en charge n'importe quel framework SSR JavaScript basé sur un bundle de déploiement conforme à la sortie de compilation attendue par Amplify. Amplify Hosting fournit une spécification de déploiement qui normalise les fichiers et la structure de répertoires pour la sortie de la version d'une application pour tout framework SSR.

Les auteurs de frameworks peuvent utiliser la spécification de déploiement basée sur le système de fichiers pour développer des adaptateurs de build open source personnalisés pour leurs frameworks spécifiques. Ces adaptateurs transformeront la sortie de compilation d'une application en un bundle de déploiement conforme à la structure de répertoire attendue d'Amplify Hosting. Ce bundle de déploiement inclura tous les fichiers et actifs nécessaires pour héberger une application, y compris la configuration d'exécution, telle que les règles de routage.

Si vous n'utilisez pas de framework ou d'adaptateur de framework, vous pouvez développer votre propre solution pour générer un bundle de déploiement conforme à la structure de répertoire attendue d'Amplify Hosting.

Amplify Hosting prend en charge les primitives suivantes : actifs statiques, calcul, optimisation des images et règles de routage. Vous pouvez tirer parti de ces primitives pour déployer des applications dotées de fonctionnalités plus riches. Pour des informations détaillées sur chaque primitive, consultezAmplify SSR : support primitif.

Vous pouvez choisir parmi les scénarios suivants pour commencer à déployer une application SSR sur Amplify.

Déployer une application Next.js

> Amplify prend en charge les applications créées à l'aide de Next.js sans avoir besoin d'un adaptateur ou d'une configuration manuelle dans la console. Pour de plus amples informations, veuillez consulter Amplify le support pour Next.js SSR.

Déployer une application qui utilise un adaptateur de framework

> Vous pouvez référencer n'importe quel adaptateur de framework open source disponible pour déployer votre application SSR sur Amplify Hosting. Pour de plus amples informations, veuillez consulter Utilisation d'un adaptateur de framework.

Un adaptateur est disponible pour le framework Nuxt. Pour plus d'informations sur l'utilisation de cet adaptateur, consultez la documentation Nuxt.

Création d'un adaptateur de framework

Les auteurs de frameworks qui souhaitent intégrer les fonctionnalités fournies par un framework peuvent utiliser la spécification de déploiement d'Amplify Hosting pour configurer la sortie de votre build afin qu'elle soit conforme à la structure attendue par Amplify. Pour de plus amples informations, veuillez consulter Déploiement d'un serveur Express à l'aide du manifeste de déploiement.

Configuration d'un script de post-construction

Vous pouvez utiliser la spécification de déploiement d'Amplify Hosting pour manipuler la sortie de votre build selon les besoins de scénarios spécifiques. Pour de plus amples informations, veuillez consulter Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de compilation. Pour obtenir un exemple, consultez Déploiement d'un serveur Express à l'aide du manifeste de déploiement.

Déployer une application SSR sur Amplify

Vous pouvez utiliser les instructions de cette rubrique pour déployer une application créée avec n'importe quel framework avec un bundle de déploiement conforme à la sortie de compilation attendue par Amplify. Si vous déployez une application Next.js, aucun adaptateur n'est nécessaire.

Si vous déployez une application SSR qui utilise un adaptateur framework, vous devez d'abord installer et configurer l'adaptateur. Pour obtenir des instructions, veuillez consulter Utilisation d'un adaptateur de framework.

Pour déployer une application SSR sur Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web.
3. Sélectionnez votre fournisseur GitHub, Bitbucket ou de AWS CodeCommit dépôt GitLab, puis choisissez Continuer.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :

    a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.

b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.

c. Choisissez Suivant.

5. Sur la page des paramètres de compilation, Amplify détecte automatiquement les applications Next.js SSR. Si vous déployez une application SSR qui utilise un adaptateur pour un autre framework, vous devez activer Amazon CloudWatch Logs de manière explicite. Dans la section Déploiement du rendu côté serveur (SSR), choisissez Activer les journaux des applications SSR.
6. L'application nécessite un rôle de service IAM qu'Amplify suppose de vous fournir des journaux. Compte AWSVous pouvez soit autoriser Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.
   - Pour permettre à Amplify de créer automatiquement un rôle et de l'associer à votre application
     - Dans la section Rôle IAM, choisissez Créer et utiliser un nouveau rôle de service.
   - Pour associer un rôle de service que vous avez créé précédemment
     a. Dans la section Rôle IAM, choisissez Utiliser un rôle de service existant.
     b. Choisissez le rôle à utiliser dans la liste.
7. Choisissez Suivant.
8. Sur la page Révision, choisissez Enregistrer et déployer.

Utilisation d'un adaptateur de framework

Vous pouvez installer et utiliser n'importe quel adaptateur de construction de framework SSR créé pour être intégré à Amplify Hosting. Chaque infrastructure qui propose un adaptateur détermine la manière dont l'adaptateur est configuré et connecté à son processus de création. En règle générale, vous installerez l'adaptateur en tant que dépendance de développement npm.

Après avoir créé une application avec un framework, consultez la documentation du framework pour savoir comment installer l'adaptateur Amplify Hosting et le configurer dans le fichier de configuration de votre application.

Créez ensuite un `amplify.yml` fichier dans le répertoire racine de votre projet. Dans le `amplify.yml` fichier, définissez le répertoire `baseDirectory` de sortie de compilation de votre application. Le framework exécute l'adaptateur pendant le processus de génération pour transformer la sortie en bundle de déploiement Amplify Hosting.

Le nom du répertoire de sortie de construction peut être n'importe quoi, mais le `.amplify-hosting` nom du fichier a une importance. Amplify recherche d'abord un répertoire défini comme. `baseDirectory` S'il existe, Amplify y recherche la sortie de compilation. Si le répertoire n'existe pas, Amplify recherche la sortie de compilation qu'il contient`.amplify-hosting`, même si elle n'a pas été définie par le client.

Voici un exemple des paramètres de génération d'une application. Le `baseDirectory` est défini sur `.amplify-hosting` pour indiquer que la sortie de compilation se trouve dans le `.amplify-hosting` dossier. Tant que le contenu du `.amplify-hosting` dossier correspond aux spécifications de déploiement d'Amplify Hosting, l'application sera déployée avec succès.

```
version: 1
frontend:
  preBuild:
    commands:
      - npm install
  build:
    commands:
      - npm run build
  artifacts:
    baseDirectory: .amplify-hosting
```

Une fois que votre application est configurée pour utiliser un adaptateur de framework, vous pouvez la déployer sur Amplify Hosting. Pour obtenir les instructions complètes, consultez Déployer une application SSR sur Amplify.

Utilisation de la spécification de déploiement d'Amplify Hosting pour configurer la sortie de compilation

Utilisez la spécification de déploiement Amplify pour configurer la sortie de compilation d'un framework SSR que vous souhaitez intégrer à Amplify Hosting. Si vous êtes l'auteur d'un framework, vous pouvez utiliser la spécification de déploiement pour comprendre comment structurer la sortie de compilation attendue par Amplify. Si vous n'utilisez pas de framework, vous pouvez développer votre propre solution pour générer une sortie de build attendue par Amplify.

Spécification de déploiement d'Amplify Hosting

La spécification de déploiement d'Amplify Hosting est une spécification basée sur un système de fichiers qui définit la structure de répertoire qui facilite les déploiements vers Amplify Hosting.

Un framework peut générer cette structure de répertoire attendue en sortie de sa commande de construction, ce qui lui permet de tirer parti des primitives de service d'Amplify Hosting. Amplify Hosting comprend la structure du bundle de déploiement et le déploie en conséquence.

Voici un exemple de la structure de dossiers qu'Amplify attend pour le bundle de déploiement. À un niveau élevé, il possède un dossier nomméstatic, un dossier nommé compute et un fichier manifeste de déploiement nommédeploy-manifest.json.

```
.amplify-hosting/
### compute/
#   ### default/
#       ### chunks/
#       #   ### app/
#       #       ### _nuxt/
#       #       #   ### index-xxx.mjs
#       #       #   ### index-styles.xxx.js
#       #       ### server.mjs
#       ### node_modules/
#       ### server.js
### static/
#   ### css/
#   #   ### nuxt-google-fonts.css
#   ### fonts/
#   #   ### font.woff2
#   ### _nuxt/
#   #   ### builds/
#   #   #   ### latest.json
#   #   ### entry.xxx.js
#   ### favicon.ico
#   ### robots.txt
### deploy-manifest.json
```

Amplify SSR : support primitif

La spécification de déploiement d'Amplify Hosting définit un contrat qui correspond étroitement aux primitives suivantes.

Actifs statiques

Fournit des frameworks capables d'héberger des fichiers statiques.

Calcul

Fournit des frameworks capables d'exécuter un serveur HTTP Node.js sur le port 3000.

Optimisation de l'image

Fournit aux frameworks un service permettant d'optimiser les images lors de l'exécution.

Règles de routage

Fournit des frameworks dotés d'un mécanisme permettant de mapper les chemins des demandes entrantes vers des cibles spécifiques.

L'.amplify-hosting/staticannuaire

Vous devez placer dans le `.amplify-hosting/static` répertoire tous les fichiers statiques accessibles au public destinés à être servis à partir de l'URL de l'application. Les fichiers contenus dans ce répertoire sont servis via la primitive static assets.

Les fichiers statiques sont accessibles à la racine (/) de l'URL de l'application sans aucune modification de leur contenu, de leur nom de fichier ou de leur extension. En outre, les sous-répertoires sont conservés dans la structure de l'URL et apparaissent avant le nom du fichier. À titre d'exemple, `.amplify-hosting/static/favicon.ico` sera servi depuis `https://myAppId.amplify-hostingapp.com/favicon.ico` et `.amplify-hosting/static/_nuxt/main.js` sera servi depuis `https://myAppId.amplify-hostingapp.com/_nuxt/main.js`

Si un framework permet de modifier le chemin de base de l'application, il doit ajouter le chemin de base aux actifs statiques du `.amplify-hosting/static` répertoire. Par exemple, si le chemin de base est`/folder1/folder2`, la sortie de génération pour un actif statique appelé `main.css` sera`.amplify-hosting/static/folder1/folder2/main.css`.

L'.amplify-hosting/computeannuaire

Une ressource de calcul unique est représentée par un sous-répertoire unique nommé `default` contenu dans le `.amplify-hosting/compute` répertoire. Le chemin est`.amplify-hosting/compute/default`. Cette ressource de calcul correspond à la primitive de calcul d'Amplify Hosting.

Le contenu du `default` sous-répertoire doit être conforme aux règles suivantes.

- Un fichier doit exister à la racine du `default` sous-répertoire pour servir de point d'entrée à la ressource de calcul.

- Le fichier du point d'entrée doit être un module Node.js et il doit démarrer un serveur HTTP qui écoute sur le port 3000.
- Vous pouvez placer d'autres fichiers dans le `default` sous-répertoire et les référencer à partir du code du fichier du point d'entrée.
- Le contenu du sous-répertoire doit être autonome. Le code du module du point d'entrée ne peut référencer aucun module en dehors du sous-répertoire. Notez que les frameworks peuvent regrouper leur serveur HTTP comme ils le souhaitent. Si le processus de calcul peut être lancé avec la `node server.js` commande, où `server.js is` est le nom du fichier d'entrée, depuis le sous-répertoire, Amplify considère que la structure du répertoire est conforme à la spécification de déploiement.

Amplify Hosting regroupe et déploie tous les fichiers du `default` sous-répertoire vers une ressource de calcul provisionnée. Chaque ressource de calcul se voit attribuer 512 Mo de stockage éphémère. Ce stockage n'est pas partagé entre les instances d'exécution, mais est partagé entre les invocations suivantes au sein de la même instance d'exécution. Les instances d'exécution sont limitées à une durée d'exécution maximale de 15 minutes, et le seul chemin accessible en écriture au sein de l'instance d'exécution est le `/tmp` répertoire. La taille compressée de chaque ensemble de ressources de calcul ne peut pas dépasser 220 Mo. Par exemple, le `.amplify/compute/default` sous-répertoire ne peut pas dépasser 220 Mo lorsqu'il est compressé.

le fichier .amplify-hosting/deploy-manifest.json ;

Utilisez le `deploy-manifest.json` fichier pour stocker les détails de configuration et les métadonnées d'un déploiement. Au minimum, un `deploy-manifest.json` fichier doit inclure un `version` attribut, l'`routes`attribut avec une route fourre-tout spécifiée et l'`framework`attribut avec des métadonnées de structure spécifiées.

La définition d'objet suivante illustre la configuration d'un manifeste de déploiement.

```
type DeployManifest = {
  version: 1;
  routes: Route[];
  computeResources?: ComputeResource[];
  imageSettings?: ImageSettings;
  framework: FrameworkMetadata;
};
```

Les rubriques suivantes décrivent les détails et l'utilisation de chaque attribut du manifeste de déploiement.

Utilisation de l'attribut version

L'`version`attribut définit la version de la spécification de déploiement que vous implémentez. Actuellement, la seule version pour la spécification de déploiement d'Amplify Hosting est la version 1. L'exemple JSON suivant illustre l'utilisation de l'`version`attribut.

```
"version": 1
```

Utilisation de l'attribut routes

L'`routes`attribut permet aux frameworks de tirer parti de la primitive des règles de routage d'Amplify Hosting. Les règles de routage fournissent un mécanisme pour acheminer les chemins des demandes entrantes vers une cible spécifique du bundle de déploiement. Les règles de routage dictent uniquement la destination d'une demande entrante et sont appliquées une fois que la demande a été transformée par des règles de réécriture et de redirection. Pour plus d'informations sur la façon dont Amplify Hosting gère les réécritures et les redirections, consultez. Utilisation des redirections

Les règles de routage ne réécrivent ni ne transforment la demande. Si une demande entrante correspond au modèle de chemin d'un itinéraire, la demande est acheminée telle quelle vers la cible de l'itinéraire.

Les règles de routage spécifiées dans le `routes` tableau doivent être conformes aux règles suivantes.

- Un itinéraire fourre-tout doit être spécifié. Un itinéraire fourre-tout possède le `/*` modèle qui correspond à toutes les demandes entrantes.
- Le `routes` tableau peut contenir un maximum de 25 éléments.
- Vous devez spécifier un `Static` itinéraire ou un `Compute` itinéraire.
- Si vous spécifiez un `Static` itinéraire, le `.amplify-hosting/static` répertoire doit exister.
- Si vous spécifiez un `Compute` itinéraire, le `.amplify-hosting/compute` répertoire doit exister.
- Si vous spécifiez un `ImageOptimization` itinéraire, vous devez également spécifier un `Compute` itinéraire. Cela est nécessaire car l'optimisation des images n'est pas encore prise en charge pour les applications purement statiques.

La définition d'objet suivante illustre la configuration de l'`Route`objet.

```
type Route = {
  path: string;
  target: Target;
  fallback?: Target;
}
```

Le tableau suivant décrit les propriétés de l'`Route`objet.

Clé	Type	Obligatoire	Description
path	Chaîne	Oui	Définit un modèle qui correspond aux chemins des demandes entrantes (à l'exception de la chaîne de requête). La longueur maximale du chemin est de 255 caractères. Un tracé doit commencer par la barre oblique`/`. Un chemin peut contenir n'importe lequel des caractères suivants : [A-Z], [a-z], [0-9], [_-.*$/~"'@ : +]. Pour la correspon dance de modèles, seuls les caractères génériques suivants sont pris en charge :

Clé	Type	Obligatoire	Description
			• *(correspond à 0 caractères ou plus) • Le `/*` modèle est appelé modèle fourre-tout et correspond à toutes les demandes entrantes.
cible	Cible	Oui	Un objet qui définit la cible vers laquelle acheminer la demande correspondante. Si un `Compute` itinéraire est spécifié, il `ComputeResource` doit exister un itinéraire correspondant. Si un `ImageOptimization` itinéraire est spécifié, `imageSettings` il doit également être spécifié.

Clé	Type	Obligatoire	Description
repli	Cible	Non	Objet qui définit la cible vers laquelle se rabattre si la cible d'origine renvoie une erreur 404. Le `target` type et le `fallback` type ne peuvent pas être identiques pour un itinéraire spécifiqu e. Par exemple, le repli de `Static` à n'`Static`est pas autorisé. Les solutions de secours ne sont prises en charge que pour les requêtes GET qui n'ont pas de corps. Si un corps est présent dans la demande, il sera supprimé lors du repli.

La définition d'objet suivante illustre la configuration de l'`Target`objet.

```
type Target = {
  kind: TargetKind;
  src?: string;
  cacheControl?: string;
}
```

Le tableau suivant décrit les propriétés de l'`Target`objet.

Clé	Type	Obligatoire	Description
sorte	Type cible	Oui	Et `enum` qui définit le type de cible. Les valeurs valides sont `Static`, `Compute` et `ImageOpti mization` .
src	Chaîne	Oui pour `Compute` Non pour les autres primitives	Chaîne qui indique le nom du sous-répertoire du bundle de déploiement qui contient le code exécutable de la primitive. Valide et obligatoire uniquemen t pour la primitive Compute. La valeur doit pointer vers l'une des ressources de calcul présentes dans le bundle de déploieme nt. Actuellement, la seule valeur prise en charge pour ce champ est`default`.
Contrôle du cache	Chaîne	Non	Chaîne qui indique la valeur de l'en-tête Cache-Con trol à appliquer à la réponse. Valable uniquement pour le Static et les

Clé	Type	Obligatoire	Description
			ImageOptimization primitives. La valeur spécifiée est remplacée par des en-têtes personnalisés. Pour plus d'informations sur les en-têtes clients d'Amplify Hosting, consultez. En-têtes personnalisés **Note** Cet en-tête Cache-Control est uniquement appliqué aux réponses réussies dont le code d'état est défini sur 200 (OK).

La définition d'objet suivante illustre l'utilisation de l'`TargetKind`énumération.

```
enum TargetKind {
  Static = "Static",
  Compute = "Compute",
  ImageOptimization = "ImageOptimization"
}
```

La liste suivante indique les valeurs valides pour l'`TargetKind`énumération.

Statique

Achemine les demandes vers la primitive des actifs statiques.

Calcul

Achemine les demandes vers la primitive de calcul.

ImageOptimization

Achemine les demandes vers la primitive d'optimisation d'image.

L'exemple JSON suivant illustre l'utilisation de l'`routes`attribut avec plusieurs règles de routage spécifiées.

```
"routes": [
    {
      "path": "/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
```

```
    {
      "path": "/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ]
```

Pour plus d'informations sur la spécification des règles de routage dans votre manifeste de déploiement, voir Bonnes pratiques pour configurer les règles de routage

Utilisation de l'attribut ComputerResources

L'`computeResources`attribut permet aux frameworks de fournir des métadonnées sur les ressources de calcul allouées. Chaque ressource de calcul doit être associée à un itinéraire correspondant.

La définition d'objet suivante illustre l'utilisation de l'`ComputeResource`objet.

```
type ComputeResource = {
  name: string;
  runtime: ComputeRuntime;
  entrypoint: string;
};

type ComputeRuntime = 'nodejs16.x' | 'nodejs18.x' | 'nodejs20.x';
```

Le tableau suivant décrit les propriétés de l'`ComputeResource`objet.

Clé	Type	Obligatoire	Description
name	Chaîne	Oui	Spécifie le nom de la ressource de calcul. Le nom doit correspondre au nom du sous-répertoire situé dans le `.amplify-hosting/compute directory` . Pour la version 1 de la spécification de déploiement, la seule valeur valide est `default`.
environnement d'exécution	ComputeRuntime	Oui	Définit le temps d'exécution de la ressource de calcul provisionnée. Les valeurs valides sont `nodejs16.x` , `nodejs18.x` et `nodejs20.x` .
point d'entrée	Chaîne	Oui	Spécifie le nom du fichier de départ à partir duquel le code sera exécuté pour la ressource de calcul spécifiée. Le fichier doit se trouver dans le sous-répertoire

Clé	Type	Obligatoire	Description
			qui représente une ressource de calcul.

Si vous avez une structure de répertoire qui ressemble à la suivante.

```
.amplify-hosting
|---compute
|   |---default
|       |---index.js
```

Le JSON de l'`computeResource`attribut ressemblera à ce qui suit.

```
"computeResources": [
    {
      "name": "default",
      "runtime": "nodejs16.x",
      "entrypoint": "index.js",
    }
  ]
```

Utilisation de l'attribut ImageSettings

L'`imageSettings`attribut permet aux frameworks de personnaliser le comportement de la primitive d'optimisation d'image, qui fournit une optimisation à la demande des images lors de l'exécution.

La définition d'objet suivante illustre l'utilisation de l'`ImageSettings`objet.

```
type ImageSettings = {
  sizes: number[];
  domains: string[];
  remotePatterns: RemotePattern[];
  formats: ImageFormat[];
  minumumCacheTTL: number;
  dangerouslyAllowSVG: boolean;
};

type ImageFormat = 'image/avif' | 'image/webp' | 'image/png' | 'image/jpeg';
```

Le tableau suivant décrit les propriétés de l'`ImageSettings`objet.

Clé	Type	Obligatoire	Description
tailles	Numéro []	Oui	Un tableau de largeurs d'image prises en charge.
domains	Chaîne []	Oui	Un ensemble de domaines externes autorisés qui peuvent utiliser l'optimisation d'image. Laissez le tableau vide pour autoriser uniquemen t le domaine de déploiement à utiliser l'optimisation des images.
Motifs distants	RemotePattern[]	Oui	Un tableau de modèles externes autorisés qui peuvent utiliser l'optimisation de l'image. Similaire aux domaines, mais offre un meilleur contrôle avec les expressions régulière s (regex).
formats	ImageFormat[]	Oui	Un tableau de formats d'image de sortie autorisés.
Cache minimal TL	Nombre	Oui	Durée du cache en secondes pour les images optimisées.

Clé	Type	Obligatoire	Description
Autorise dangereus ement le SVG	Booléen	Oui	Autorise les URL d'image d'entrée SVG. Cette option est désactivée par défaut pour des raisons de sécurité.

La définition d'objet suivante illustre l'utilisation de l'`RemotePattern`objet.

```
type RemotePattern = {
  protocol?: 'http' | 'https';
  hostname: string;
  port?: string;
  pathname?: string;
}
```

Le tableau suivant décrit les propriétés de l'`RemotePattern`objet.

Clé	Type	Obligatoire	Description
protocole ;	Chaîne	Non	Protocole du modèle de télécommande autorisé. Les valeurs valides sont `http` ou `https`.
hostname	Chaîne	Oui	Le nom d'hôte du modèle distant autorisé. Vous pouvez spécifier un littéral ou un caractère générique. Un seul `*` correspon d à un seul sous-

Clé	Type	Obligatoire	Description
			domaine. Un double `**` correspond à un nombre quelconque de sous-domaines. Amplify n'autorise pas les caractères génériques où seul `**` est spécifié.
port	Chaîne	Non	Port du modèle de télécommande autorisé.
chemin d'accès	Chaîne	Non	Le nom du chemin du modèle de télécommande autorisé.

L'exemple suivant illustre l'`imageSettings`attribut.

```
"imageSettings": {
    "sizes": [
      100,
      200
    ],
    "domains": [
      "example.com"
    ],
    "remotePatterns": [
      {
        "protocol": "https",
        "hostname": "example.com",
        "port": "",
        "pathname": "/**",
      }
    ],
    "formats": [
      "image/webp"
    ],
```

```
    "minumumCacheTTL": 60,
    "dangerouslyAllowSVG": false
  }
```

Utilisation de l'attribut framework

Utilisez l'`frameworkattribut pour spécifier les métadonnées du framework.

La définition d'objet suivante illustre la configuration de l'`FrameworkMetadataobjet.

```
type FrameworkMetadata = {
  name: string;
  version: string;
}
```

Le tableau suivant décrit les propriétés de l'`FrameworkMetadataobjet.

Clé	Type	Obligatoire	Description
name	Chaîne	Oui	Le nom du framework.
version	Chaîne	Oui	Version du framework . Il doit s'agir d'une chaîne de version sémantique (semver) valide.

Bonnes pratiques pour configurer les règles de routage

Les règles de routage fournissent un mécanisme pour acheminer les chemins des demandes entrantes vers des cibles spécifiques du bundle de déploiement. Dans un bundle de déploiement, les auteurs du framework peuvent émettre des fichiers vers la sortie de build qui sont déployés sur l'une des cibles suivantes :

- Ressources statiques primitives — Les fichiers sont contenus dans le `.amplify-hosting/static` répertoire.

- Primitive de calcul — Les fichiers sont contenus dans le `.amplify-hosting/compute/default` répertoire.

Les auteurs du framework fournissent également un ensemble de règles de routage dans le fichier manifeste de déploiement. Chaque règle du tableau est comparée à la demande entrante dans un ordre séquentiel, jusqu'à ce qu'il y ait une correspondance. Lorsqu'il existe une règle de correspondance, la demande est acheminée vers la cible spécifiée dans la règle de correspondance. Facultativement, une cible de secours peut être spécifiée pour chaque règle. Si la cible d'origine renvoie une erreur 404, la demande est acheminée vers la cible de secours.

La spécification de déploiement exige que la dernière règle de l'ordre de traversée soit une règle fourre-tout. Une règle fourre-tout est spécifiée avec le `/*` chemin. Si la demande entrante ne correspond à aucune des routes précédentes du tableau de règles de routage, elle est acheminée vers la cible de règles fourre-tout.

Pour les frameworks SSR tels queNuxt.js, la cible de la règle fourre-tout doit être la primitive de calcul. Cela est dû au fait que les applications SSR ont des pages affichées côté serveur avec des itinéraires qui ne sont pas prévisibles au moment de la création. Par exemple, si une Nuxt.js application possède une page `/blog/[slug]` où se `[slug]` trouve un paramètre de route dynamique. La cible de la règle fourre-tout est le seul moyen d'acheminer les demandes vers ces pages.

En revanche, des modèles de trajectoire spécifiques peuvent être utilisés pour cibler des itinéraires connus au moment de la construction. Par exemple, Nuxt.js diffuse les actifs statiques depuis le `/_nuxt` chemin. Cela signifie que le `/_nuxt/*` chemin peut être ciblé par une règle de routage spécifique qui achemine les demandes vers la primitive des actifs statiques.

Routage des dossiers publics

La plupart des frameworks SSR offrent la possibilité de servir des actifs statiques mutables à partir d'un `public` dossier. Les fichiers tels que `favicon.ico` et `robots.txt` sont généralement conservés dans le `public` dossier et sont servis à partir de l'URL racine de l'application. Par exemple, le `favicon.ico` fichier est servi à partir de`https://example.com/favicon.ico`. Notez qu'il n'existe aucun modèle de chemin prévisible pour ces fichiers. Ils sont presque entièrement dictés par le nom du fichier. La seule façon de cibler les fichiers contenus dans le `public` dossier est d'utiliser la méthode fourre-tout. Cependant, la cible de l'itinéraire fourre-tout doit être la primitive de calcul.

Nous recommandons l'une des approches suivantes pour gérer votre `public` dossier.

1. Utilisez un modèle de chemin pour cibler les chemins de requête contenant des extensions de fichiers. Par exemple, vous pouvez l'utiliser `/*.*` pour cibler tous les chemins de demande contenant une extension de fichier.

   Notez que cette approche peut ne pas être fiable. Par exemple, si le `public` dossier contient des fichiers sans extension, ils ne sont pas visés par cette règle. Un autre problème à prendre en compte avec cette approche est que l'application peut avoir des pages avec des points dans leur nom. Par exemple, une page `/blog/2021/01/01/hello.world` sera ciblée par la `/*.*` règle. Ce n'est pas idéal car la page n'est pas un actif statique. Toutefois, vous pouvez ajouter une cible de secours à cette règle pour garantir qu'en cas d'erreur 404 provenant de la primitive statique, la requête revienne à la primitive de calcul.

   ```
   {
       "path": "/*.*",
       "target": {
           "kind": "Static"
       },
       "fallback": {
           "kind": "Compute",
           "src": "default"
       }
   }
   ```

2. Identifiez les fichiers du `public` dossier au moment de la création et émettez une règle de routage pour chaque fichier. Cette approche n'est pas évolutive car la spécification de déploiement impose une limite de 25 règles.

   ```
   {
       "path": "/favicon.ico",
       "target": {
           "kind": "Static"
       }
   },
   {
       "path": "/robots.txt",
       "target": {
           "kind": "Static"
       }
   }
   ```

3. Recommandez aux utilisateurs de votre framework de stocker tous les actifs statiques mutables dans un sous-dossier du `public` dossier.

   Dans l'exemple suivant, l'utilisateur peut stocker tous les actifs statiques modifiables dans le `public/assets` dossier. Ensuite, une règle de routage avec le modèle de chemin `/assets/*` peut être utilisée pour cibler tous les actifs statiques mutables présents dans le `public/assets` dossier.

   ```
   {
       "path": "/assets/*",
       "target": {
           "kind": "Static"
       }
   }
   ```

4. Spécifiez une solution de secours statique pour l'itinéraire fourre-tout. Cette approche présente des inconvénients qui sont décrits plus en détail dans la Routage de secours fourre-tout section suivante.

Routage de secours fourre-tout

Pour les frameworks SSR tels que ceux Nuxt.js où une route fourre-tout est spécifiée pour la cible primitive de calcul, les auteurs du framework peuvent envisager de spécifier une solution de secours statique pour la route fourre-tout afin de résoudre le problème de routage des dossiers. `public` Cependant, ce type de règle de routage interrompt les pages 404 affichées côté serveur. Par exemple, si l'utilisateur final visite une page qui n'existe pas, l'application affiche une page 404 avec un code d'état 404. Toutefois, si la route fourre-tout comporte une solution de secours statique, la page 404 n'est pas affichée. Au lieu de cela, la requête revient à la primitive statique et aboutit toujours à un code d'état 404, mais la page 404 n'est pas rendue.

```
{
    "path": "/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    },
    "fallback": {
        "kind": "Static"
    }
}
```

Routage du chemin de base

Les frameworks qui offrent la possibilité de modifier le chemin de base de l'application sont censés ajouter le chemin de base aux actifs statiques du `.amplify-hosting/static` répertoire. Par exemple, si le chemin de base est`/folder1/folder2`, la sortie de génération pour un actif statique appelé main.css sera`.amplify-hosting/static/folder1/folder2/main.css`.

Cela signifie que les règles de routage doivent également être mises à jour pour refléter le chemin de base. Par exemple, si le chemin de base est`/folder1/folder2`, la règle de routage pour les actifs statiques du `public` dossier sera la suivante.

```
{
    "path": "/folder1/folder2/*.*",
    "target": {
        "kind": "Static"
    }
}
```

De même, les routes côté serveur doivent également être précédées du chemin de base. Par exemple, si le chemin de base est`/folder1/folder2`, la règle de routage de l'`/api`itinéraire ressemblera à ce qui suit.

```
{
    "path": "/folder1/folder2/api/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    }
}
```

Cependant, le chemin de base ne doit pas être ajouté à l'itinéraire fourre-tout. Par exemple, si le chemin de base est le suivant`/folder1/folder2`, l'itinéraire fourre-tout restera le suivant.

```
{
    "path": "/*",
    "target": {
        "kind": "Compute",
        "src": "default"
    }
}
```

Exemples de routes Nuxt.js

Voici un exemple de `deploy-manifest.json` fichier pour une application Nuxt qui montre comment spécifier des règles de routage.

```
{
  "version": 1,
  "routes": [
    {
      "path": "/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/_nuxt/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
```

```
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "entrypoint": "server.js",
      "runtime": "nodejs18.x"
    }
  ],
  "framework": {
    "name": "nuxt",
    "version": "3.8.1"
  }
}
```

Voici un exemple de `deploy-manifest.json` fichier pour Nuxt qui montre comment spécifier des règles de routage, y compris des chemins de base.

```
{
  "version": 1,
  "routes": [
    {
      "path": "/base-path/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/base-path/_nuxt/builds/meta/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
```

```
    {
      "path": "/base-path/_nuxt/builds/*",
      "target": {
        "cacheControl": "public, max-age=1, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/base-path/_nuxt/*",
      "target": {
        "cacheControl": "public, max-age=31536000, immutable",
        "kind": "Static"
      }
    },
    {
      "path": "/base-path/*.*",
      "target": {
        "kind": "Static"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "entrypoint": "server.js",
      "runtime": "nodejs18.x"
    }
  ],
  "framework": {
    "name": "nuxt",
    "version": "3.8.1"
  }
```

```
}
```

Pour plus d'informations sur l'utilisation de l'`routes`attribut, consultez[Utilisation de l'attribut routes](#).

Déploiement d'un serveur Express à l'aide du manifeste de déploiement

Cet exemple explique comment déployer un serveur Express de base à l'aide de la spécification de déploiement d'Amplify Hosting. Vous pouvez utiliser le manifeste de déploiement fourni pour spécifier le routage, les ressources de calcul et d'autres configurations.

Configurer un serveur Express localement avant de le déployer sur Amplify Hosting

1. Créez un nouveau répertoire pour votre projet et installez Express et Typescript.

```
mkdir express-app
cd express-app

# The following command will prompt you for information about your project
npm init

# Install express, typescript and types
npm install express --save
npm install typescript ts-node @types/node @types/express --save-dev
```

2. Ajoutez un `tsconfig.json` fichier à la racine de votre projet avec le contenu suivant.

```
{
  "compilerOptions": {
    "target": "es6",
    "module": "commonjs",
    "outDir": "./dist",
    "strict": true,
    "esModuleInterop": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true
  },
  "include": ["src/**/*.ts"],
  "exclude": ["node_modules"]
}
```

3. Créez un répertoire nommé `src` à la racine de votre projet.

4. Créez un `index.ts` fichier dans le `src` répertoire. Ce sera le point d'entrée de l'application qui démarre un serveur Express. Le serveur doit être configuré pour écouter sur le port 3000.

```
// src/index.ts
import express from 'express';

const app: express.Application = express();
const port = 3000;

app.use(express.text());

app.listen(port, () => {
  console.log(`server is listening on ${port}`);
});

// Homepage
app.get('/', (req: express.Request, res: express.Response) => {
  res.status(200).send("Hello World!");
});

// GET
app.get('/get', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-get-header", "get-header-value").send("get-response-
from-compute");
});

//POST
app.post('/post', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-post-header", "post-header-
value").send(req.body.toString());
});

//PUT
app.put('/put', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-put-header", "put-header-
value").send(req.body.toString());
});

//PATCH
app.patch('/patch', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-patch-header", "patch-header-
value").send(req.body.toString());
});
```

```
// Delete
app.delete('/delete', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-delete-header", "delete-header-value").send();
});
```

5. Ajoutez les scripts suivants à votre `package.json` fichier.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
  "serve": "node dist/index.js"
}
```

6. Créez un répertoire nommé `public` à la racine de votre projet. Créez ensuite un fichier nommé `hello-world.txt` avec le contenu suivant.

```
Hello world!
```

7. Ajoutez un `.gitignore` fichier à la racine de votre projet avec le contenu suivant.

```
.amplify-hosting
dist
node_modules
```

Configurer le manifeste de déploiement d'Amplify

1. Créez un fichier nommé `deploy-manifest.json` dans le répertoire racine de votre projet.
2. Copiez et collez le manifeste suivant dans votre `deploy-manifest.json` fichier.

```
{
  "version": 1,
  "framework": { "name": "express", "version": "4.18.2" },
  "imageSettings": {
    "sizes": [
      100,
      200,
      1920
    ],
    "domains": [],
    "remotePatterns": [],
```

```
    "formats": [],
    "minimumCacheTTL": 60,
    "dangerouslyAllowSVG": false
  },
  "routes": [
    {
      "path": "/_amplify/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/*.*",
      "target": {
        "kind": "Static",
        "cacheControl": "public, max-age=2"
      },
      "fallback": {
        "kind": "Compute",
        "src": "default"
      }
    },
    {
      "path": "/*",
      "target": {
        "kind": "Compute",
        "src": "default"
      }
    }
  ],
  "computeResources": [
    {
      "name": "default",
      "runtime": "nodejs18.x",
      "entrypoint": "index.js"
    }
  ]
}
```

Le manifeste décrit comment Amplify Hosting doit gérer le déploiement de votre application. Les principaux paramètres sont les suivants.

- version — Indique la version de la spécification de déploiement que vous utilisez.
- framework — Ajustez-le pour spécifier la configuration de votre Express serveur.
- ImageSettings — Cette section est facultative pour un Express serveur, sauf si vous gérez l'optimisation des images.
- itinéraires : ils sont essentiels pour diriger le trafic vers les bonnes parties de votre application. L'`"kind": "Compute"`itinéraire dirige le trafic vers la logique de votre serveur.
- ComputerResources — Utilisez cette section pour spécifier le moteur d'exécution et le point d'entrée de votre Express serveur.

Configurez ensuite un script de post-génération qui déplace les artefacts de l'application créée dans le bundle de `.amplify-hosting` déploiement. La structure du répertoire est conforme à la spécification de déploiement d'Amplify Hosting.

Configuration du script de post-construction

1. Créez un répertoire nommé `bin` à la racine de votre projet.
2. Créez un fichier nommé `postbuild.sh` dans le `bin` répertoire. Ajoutez le contenu suivant au fichier `postbuild.sh`.

```
#!/bin/bash

rm -rf ./.amplify-hosting

mkdir -p ./.amplify-hosting/compute

cp -r ./dist ./.amplify-hosting/compute/default
cp -r ./node_modules ./.amplify-hosting/compute/default/node_modules

cp -r public ./.amplify-hosting/static

cp deploy-manifest.json ./.amplify-hosting/deploy-manifest.json
```

3. Ajoutez un `postbuild` script à votre `package.json` fichier. Le fichier doit ressembler à ce qui suit.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
```

```
  "serve": "node dist/index.js",
  "postbuild": "chmod +x bin/postbuild.sh && ./bin/postbuild.sh"
}
```

4. Exécutez la commande suivante pour créer votre application.

```
npm run build
```

5. (Facultatif) Ajustez vos itinéraires pour Express. Vous pouvez modifier les itinéraires de votre manifeste de déploiement pour les adapter à votre serveur Express. Par exemple, si le `public` répertoire ne contient aucun actif statique, il se peut que vous n'ayez besoin que de la route fourre-tout menant à `"path": "/*"` Compute. Cela dépend de la configuration de votre serveur.

La structure finale de votre répertoire doit ressembler à ce qui suit.

```
express-app/
### .amplify-hosting/
#   ### compute/
#   #   ### default/
#   #       ### node_modules/
#   #       ### index.js
#   ### static/
#   #   ### hello.txt
#   ### deploy-manifest.json
### bin/
#   ### .amplify-hosting/
#   #   ### compute/
#   #   #   ### default/
#   #   ### static/
#   ### postbuild.sh*
### dist/
#   ### index.js
### node_modules/
### public/
#   ### hello.txt
### src/
#   ### index.ts
### deploy-manifest.json
### package.json
### package-lock.json
```

```
### tsconfig.json
```

Déployez votre serveur

1. Transférez votre code dans votre dépôt Git, puis déployez votre application sur Amplify Hosting.
2. Mettez à jour vos paramètres de compilation pour qu'ils `baseDirectory` pointent vers ce qui `.amplify-hosting` suit. Au cours de la compilation, Amplify détectera le fichier manifeste dans le `.amplify-hosting` répertoire et déploiera votre serveur Express tel que configuré.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - nvm use 18
        - npm install
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .amplify-hosting
    files:
      - '**/*'
```

3. Pour vérifier que votre déploiement a réussi et que votre serveur fonctionne correctement, accédez à votre application à l'URL par défaut fournie par Amplify Hosting.

Optimisation de l'image pour les applications SSR

Amplify Hosting fournit une fonctionnalité d'optimisation d'image intégrée qui prend en charge toutes les applications SSR. Grâce à l'optimisation d'image d'Amplify, vous pouvez fournir des images de haute qualité au format, aux dimensions et à la résolution adaptés à l'appareil qui y accède, tout en conservant la plus petite taille de fichier possible.

Actuellement, vous pouvez soit utiliser le composant Image Next.js pour optimiser les images à la demande, soit implémenter un chargeur d'images personnalisé. Si vous utilisez Next.js 13 ou une version ultérieure, vous n'avez aucune autre action à effectuer pour utiliser la fonction d'optimisation d'image d'Amplify. Si vous implémentez un chargeur personnalisé, consultezUtilisation d'un chargeur d'images personnalisé.

Utilisation d'un chargeur d'images personnalisé

Si vous utilisez un chargeur d'image personnalisé, Amplify détecte le chargeur dans le `next.config.js` fichier de votre application et n'utilise pas la fonction d'optimisation d'image intégrée. Pour plus d'informations sur les chargeurs personnalisés pris en charge par Next.js, consultez la documentation relative aux images Next.js.

Intégration de l'optimisation des images pour les auteurs de frameworks

Les auteurs du framework peuvent intégrer la fonctionnalité d'optimisation d'image d'Amplify en utilisant la spécification de déploiement d'Amplify Hosting. Pour activer l'optimisation des images, votre manifeste de déploiement doit contenir une règle de routage qui cible le service d'optimisation des images. L'exemple suivant montre comment configurer la règle de routage.

```
// .amplify-hosting/deploy-manifest.json

{
  "routes": [
    {
      "path": "/images/*",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=31536000, immutable"
      }
    }
  ]
}
```

Pour plus d'informations sur la configuration des paramètres d'optimisation des images à l'aide de la spécification de déploiement, consultezSpécification de déploiement d'Amplify Hosting.

Comprendre l'API d'optimisation des images

L'optimisation des images peut être invoquée lors de l'exécution via l'URL de domaine d'une application Amplify, sur le chemin défini par la règle de routage.

```
GET https://{appDomainName}/{path}?{queryParams}
```

L'optimisation des images impose les règles suivantes aux images.

- Amplify ne peut pas optimiser les formats GIF, APNG et SVG ni les convertir dans un autre format.
- Les images SVG ne sont diffusées que si le `dangerouslyAllowSVG` paramètre est activé.
- La largeur ou la hauteur d'une image source ne peut pas dépasser 11 Mo ou 9 000 pixels.
- La limite de taille d'une image optimisée est de 4 Mo.
- Le protocole HTTP ou HTTPS est le seul protocole pris en charge pour le sourcing d'images avec des URL distantes.

En-têtes HTTP

L'en-tête HTTP de la demande Accept est utilisé pour spécifier les formats d'image, exprimés sous forme de types MIME, autorisés par le client (généralement un navigateur Web). Le service d'optimisation d'image tentera de convertir l'image au format spécifié. La valeur spécifiée pour cet en-tête aura une priorité supérieure à celle du paramètre de requête de format. Par exemple, une valeur valide pour l'en-tête Accept est`image/png, image/webp, */*` . Le paramètre de formats spécifié dans le manifeste de déploiement d'Amplify limitera les formats à ceux de la liste. Même si l'en-tête Accept demande un format spécifique, il sera ignoré si le format ne figure pas dans la liste d'autorisation.

Paramètres de demande URI

Le tableau suivant décrit les paramètres de demande d'URI pour l'optimisation des images.

Paramètre de la demande	Type	Obligatoire	Description	Exemple
url	Chaîne	Oui	Un chemin relatif ou une URL absolue vers l'image source. Pour une URL distante, les protocoles http et https sont pris en charge. La valeur doit être codée en URL.	`?url=https%3A%2F%2Fwww.example.com%2Fbuffalo.png`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
width	Nombre	Oui	Largeur en pixels de l'image optimisée.	`?width=800`
height	Nombre	Non	Hauteur en pixels de l'image optimisée. Si ce n'est pas spécifié, l'image sera redimensionnée automatiquement pour correspondre à la largeur.	`?height=600`
ajuster	Valeurs d'énumération :`cover`,,`contain` `inside` `outside`	Non	Comment l'image est redimensionnée pour s'adapter à la largeur et à la hauteur spécifiées.	`?width=800&height=600&fit=cover`
position	Valeurs d'énumération :`center`,,`top`, `bottom` `left`	Non	Une position à utiliser lorsque l'ajustement est `cover` ou`contain`.	`?fit=contain&position=centre`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
trim	Nombre	Non	Découpe les pixels de tous les bords qui contiennent des valeurs similaires à la couleur d'arrière-plan spécifiée pour le pixel en haut à gauche.	`?trim=50`
étendre	Objet	Non	Ajoute des pixels sur les bords de l'image en utilisant la couleur dérivée des pixels du bord le plus proche. Dans le format `{top}_{right}_{bottom}_{left}`, chaque valeur correspond au nombre de pixels à ajouter.	`?extend=10_0_5_0`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
extract	Objet	Non	Recadre l'image dans le rectangle spécifié délimité par le haut, la gauche, la largeur et la hauteur. Le format est {left} _ {top} _ {width} _ {right} où chaque valeur est le nombre de pixels à recadrer.	`?extract=10_0_5_0`
format	Chaîne	Non	Format de sortie souhaité pour l'image optimisée .	`?format=webp`
quality	Nombre	Non	La qualité de l'image, de 1 à 100. Utilisé uniquement lors de la conversion du format de l'image.	`?quality=50`
rotate	Nombre	Non	Fait pivoter l'image selon l'angle spécifié en degrés.	`?rotate=45`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
retourner	Booléen	Non	Reflète l'image verticalement (de haut en bas) sur l'axe X. Cela se produit toujours avant la rotation, le cas échéant.	`?flip`
flop	Booléen	Non	Reflète l'image horizontalement (gauche-droite) sur l'axe Y. Cela se produit toujours avant la rotation, le cas échéant.	`?flop`
affiner	Nombre	Non	La netteté améliore la définition des bords de l'image. Les valeurs valides sont comprises entre 0,000001 et 10.	`?sharpen=1`
median	Nombre	Non	Applique un filtre médian. Cela permet de supprimer le bruit ou de lisser les bords d'une image.	`?sharpen=3`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
brouiller	Nombre	Non	Applique un flou gaussien du sigma spécifié. Les valeurs valides sont comprises entre 0,3 et 1 000.	`?blur=20`
gamma	Nombre	Non	Applique une correction gamma pour améliorer la luminosité perçue d'une image redimensionnée. La valeur doit être comprise entre 1,0 et 3,0.	`?gamma=1`
nier	Booléen	Non	Inverse les couleurs de l'image.	`?negate`
normaliser	Booléen	Non	Améliore le contraste de l'image en étendant sa luminance pour couvrir une plage dynamique complète.	`?normalize`

Paramètre de la demande	Type	Obligatoire	Description	Exemple
seuil	Nombre	Non	Remplace n'importe quel pixel de l'image par un pixel noir si son intensité est inférieure au seuil spécifié. Ou avec un pixel blanc s'il est supérieur au seuil. Les valeurs valides sont comprises entre 0 et 255.	`?threshold=155`
teinte	Chaîne	Non	Teinte l'image à l'aide du RGB fourni tout en préservant la luminance de l'image.	`?tint=#7743CE`
niveaux de gris	Booléen	Non	Transforme l'image en niveaux de gris (noir et blanc).	`?grayscale`

Codes d'état des réponses

La liste suivante décrit les codes d'état de réponse pour l'optimisation des images.

Succès : code d'état HTTP 200

La demande a été traitée avec succès.

BadRequest - Code d'état HTTP 400

- Un paramètre de requête d'entrée n'a pas été spécifié correctement.
- L'URL distante n'est pas répertoriée comme autorisée dans le `remotePatterns` paramètre.
- L'URL distante ne se transforme pas en image.
- La largeur ou la hauteur demandées ne sont pas répertoriées comme autorisées dans le `sizes` paramètre.
- L'image demandée est au format SVG mais le `dangerouslyAllowSvg` paramètre est désactivé.

Introuvable - Code d'état HTTP 404

L'image source n'a pas été trouvée.

Contenu trop volumineux - code d'état HTTP 413

L'image source ou l'image optimisée dépassent la taille maximale autorisée en octets.

Mise en cache

Amplify Hosting met en cache les images optimisées sur notre CDN afin que les demandes suivantes adressées à la même image, avec les mêmes paramètres de requête, soient traitées à partir du cache. Le temps de vie du cache (TTL) est contrôlé par l'`Cache-Control`en-tête. La liste suivante décrit les options qui s'offrent à vous pour spécifier l'`Cache-Control`en-tête.

- En utilisant la `Cache-Control` clé de la règle de routage qui cible l'optimisation de l'image.
- À l'aide d'en-têtes personnalisés définis dans l'application Amplify.
- Pour les images distantes, l'`Cache-Control`en-tête renvoyé par l'image distante est respecté.

La `minimumCacheTTL` valeur spécifiée dans les paramètres d'optimisation de l'image définit la limite inférieure de la cache-control max-age directive. Par exemple, si l'URL d'une image distante répond par un`cache-control s-max-age=10`, mais que la valeur de `minimumCacheTTL` est 60, alors 60 est utilisé.

Prise en charge de la version Node.js pour les applications Next.js

Lorsqu'Amplify crée et déploie une application de calcul Next.js, elle utilise la version Node.js d'exécution qui correspond à la version principale de Node.js celle utilisée pour créer l'application.

Vous pouvez spécifier la Node.js version à utiliser dans la fonction de remplacement du package Live de la console Amplify. Pour plus d'informations sur la configuration des mises à jour des packages en direct, consultezMises à jour des packages en. Vous pouvez également spécifier la Node.js version à l'aide d'autres mécanismes, tels que nvm des commandes. Si vous ne spécifiez pas de version, Amplify utilise par défaut la version actuelle utilisée par le conteneur de compilation Amplify.

Résolution des problèmes liés aux déploiements SSR

Si vous rencontrez des problèmes inattendus lors du déploiement d'une application SSR avec le système de calcul Amplify Hosting, consultez les rubriques de résolution des problèmes suivantes. Si vous ne trouvez pas de solution à votre problème ici, consultez le guide de résolution des problèmes de calcul Web SSR dans le référentiel Amplify GitHub Hosting Issues.

Rubriques

Vous utilisez un adaptateur de framework

Si vous rencontrez des problèmes pour déployer une application SSR qui utilise un adaptateur de framework, consultezAmplify la prise en charge des frameworks SSR.

Les routes de l'API Edge entraînent l'échec de la compilation de votre fichier Next.js

Actuellement, Amplify ne prend pas en charge les routes d'API Next.js Edge. Vous devez utiliser des API et des intergiciels non périphériques lorsque vous hébergez votre application avec Amplify.

La régénération statique incrémentielle à la demande ne fonctionne pas pour votre application

À partir de la version 12.2.0, Next.js prend en charge la régénération statique incrémentielle (ISR) pour purger manuellement le cache Next.js pour une page spécifique. Cependant, Amplify ne prend actuellement pas en charge l'ISR à la demande. Si votre application utilise la revalidation à la demande de Next.js, cette fonctionnalité ne fonctionnera pas lorsque vous déployez votre application sur Amplify.

La sortie de build de votre application dépasse la taille maximale autorisée

Actuellement, la taille de sortie maximale prise en charge par Amplify pour les applications SSR est de 220 Mo. Si vous recevez un message d'erreur indiquant que la taille de la sortie de build de votre application dépasse la taille maximale autorisée, vous devez prendre des mesures pour la réduire.

Pour réduire la taille de la sortie de build d'une application, vous pouvez inspecter les artefacts de build de l'application et identifier les dépendances importantes à mettre à jour ou à supprimer. Tout d'abord, téléchargez les artefacts de construction sur votre ordinateur local. Vérifiez ensuite la taille des répertoires. Par exemple, le `node_modules` répertoire peut contenir des fichiers binaires tels que `@swc` et `@esbuild` qui sont référencés par les fichiers d'exécution du serveur Next.js. Comme ces fichiers binaires ne sont pas nécessaires au moment de l'exécution, vous pouvez les supprimer après la compilation.

Utilisez les instructions suivantes pour télécharger le résultat de compilation d'une application et inspecter la taille des répertoires à l'aide de la AWS Command Line Interface (CLI).

Pour télécharger et inspecter le résultat de compilation d'une application Next.js

1. Ouvrez une fenêtre de terminal et exécutez la commande suivante. Modifiez l'identifiant de l'application, le nom de la branche et l'identifiant de la tâche selon vos propres informations. Pour l'identifiant de la tâche, utilisez le numéro de version de la version échouée que vous étudiez.

   ```
   aws amplify get-job --app-id abcd1234 --branch-name main --job-id 2
   ```

2. Dans la sortie du terminal, recherchez l'URL des artefacts présignés dans la `stepName: "BUILD"` section `jobsteps`,. L'URL est surlignée en rouge dans l'exemple de sortie suivant.

   ```
   "job": {
       "summary": {
   ```

```
        "jobArn": "arn:aws:amplify:us-west-2:111122223333:apps/abcd1234/main/
jobs/0000000002",
        "jobId": "2",
        "commitId": "HEAD",
        "commitTime": "2024-02-08T21:54:42.398000+00:00",
        "startTime": "2024-02-08T21:54:42.674000+00:00",
        "status": "SUCCEED",
        "endTime": "2024-02-08T22:03:58.071000+00:00"
    },
    "steps": [
        {
            "stepName": "BUILD",
            "startTime": "2024-02-08T21:54:42.693000+00:00",
            "status": "SUCCEED",
            "endTime": "2024-02-08T22:03:30.897000+00:00",
            "logUrl": "https://aws-amplify-prod-us-west-2-artifacts.s3.us-
west-2.amazonaws.com/abcd1234/main/0000000002/BUILD/log.txt?X-Amz-Security-
Token=IQoJb3JpZ2luX2V...Example
```

3. Copiez et collez l'URL dans une fenêtre de navigateur. Un `artifacts.zip` fichier est téléchargé sur votre ordinateur local. Il s'agit du résultat de votre build.
4. Exécutez la commande d'utilisation du du disque pour vérifier la taille des répertoires. L'exemple de commande suivant renvoie la taille des `static` répertoires `compute` et.

   ```
   du -csh compute static
   ```

   Voici un exemple de sortie avec des informations de taille pour les `static` répertoires `compute` et.

   ```
    29M    compute
   3.8M    static
    33M    total
   ```

5. Ouvrez le `compute` répertoire et `node_modules` localisez-le. Vérifiez vos dépendances pour les fichiers que vous pouvez mettre à jour ou supprimer afin de réduire la taille du dossier.
6. Si votre application inclut des fichiers binaires qui ne sont pas nécessaires à l'exécution, supprimez-les après la compilation en ajoutant les commandes suivantes à la section build du `amplify.yml` fichier de votre application.

   ```
   - rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
   ```

```
- rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Voici un exemple de la section des commandes de génération d'un `amplify.yml` fichier dans laquelle ces commandes ont été ajoutées après l'exécution d'une version de production.

```
frontend:
  phases:
    build:
      commands:
         -npm run build

         // After running a production build, delete the files
         - rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
         - rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Votre compilation échoue en raison d'une erreur de mémoire insuffisante

Next.js vous permet de mettre en cache des artefacts de build afin d'améliorer les performances lors des builds suivants. En outre, le AWS CodeBuild conteneur d'Amplify compresse et télécharge ce cache sur Amazon S3, en votre nom, afin d'améliorer les performances de compilation ultérieures. Cela pourrait entraîner l'échec de votre compilation en raison d'une erreur de mémoire insuffisante.

Effectuez les actions suivantes pour empêcher votre application de dépasser la limite de mémoire pendant la phase de création. Tout d'abord, `.next/cache/**/*` supprimez-le de la section cache.paths de vos paramètres de compilation. Supprimez ensuite la variable d'`NODE_OPTIONS`environnement de votre fichier de paramètres de compilation. Définissez plutôt la variable d'`NODE_OPTIONS`environnement dans la console Amplify pour définir la limite de mémoire maximale du nœud. Pour plus d'informations sur la définition des variables d'environnement à l'aide de la console Amplify, consultez. Définir les variables d'environnement

Après avoir apporté ces modifications, réessayez de créer. En cas de succès, ajoutez-le à `.next/cache/**/*` nouveau à la section cache.paths de votre fichier de paramètres de compilation.

Pour plus d'informations sur la configuration du cache Next.js afin d'améliorer les performances de compilation, consultez AWS CodeBuild sur le site Web Next.js.

La taille de la réponse HTTP est trop grande

Actuellement, la taille de réponse maximale prise en charge par Amplify pour les applications Next.js 12 et 13 utilisant la plate-forme Web Compute est de 5,72 Mo. Les réponses dépassant cette limite renvoient 504 erreurs sans aucun contenu aux clients.

Amplify le support pour Next.js SSR

Amplify prend en charge le déploiement et l'hébergement d'applications Web rendues côté serveur (SSR) créées à l'aide de Next.js uniquement. Next.js est un framework React pour développer des SPA avec JavaScript. Vous pouvez déployer des applications créées avec Next.js 13 avec des fonctionnalités telles que l'optimisation des images et le middleware.

Les développeurs peuvent utiliser Next.js pour combiner la génération de sites statiques (SSG) et le SSR dans un seul projet. Les pages SSG sont prérendues au moment de la création, et les pages SSR sont prérendues au moment de la demande.

Le prérendu peut améliorer les performances et l'optimisation des moteurs de recherche. Comme Next.js préaffiche toutes les pages du serveur, le contenu HTML de chaque page est prêt lorsqu'il atteint le navigateur du client. Ce contenu peut également être chargé plus rapidement. Des temps de chargement plus rapides améliorent l'expérience de l'utilisateur final avec un site Web et ont un impact positif sur le classement SEO du site. Le pré-rendu améliore également le référencement en permettant aux robots des moteurs de recherche de trouver et d'explorer facilement le contenu HTML d'un site Web.

Next.js fournit un support analytique intégré pour mesurer divers indicateurs de performance, tels que le délai jusqu'au premier octet (TTFB) et le premier contenu de peinture (FCP). Pour plus d'informations sur Next.js, consultez Getting started on the Next.js website.

Support des fonctionnalités de Next.js

Amplify Hosting Compute gère entièrement le rendu côté serveur (SSR) pour les applications créées avec Next.js 12 et 13. Si vous avez déployé une application Next.js sur Amplify avant la sortie d'Amplify Hosting Compute, votre application utilise l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). Amplify Hosting Compute ne prend pas en charge les applications créées à l'aide de Next.js version 11 ou antérieure. Nous vous recommandons vivement de migrer vos applications Next.js 11 vers le fournisseur SSR géré par le calcul Amplify Hosting.

La liste suivante décrit les fonctionnalités spécifiques prises en charge par le fournisseur de SSR de calcul Amplify Hosting.

Fonctionnalités prises en charge

- Pages rendues côté serveur (SSR)
- Pages statiques
- Routes d'API
- Routes dynamiques
- Suivez tous les itinéraires
- SSG (génération statique)
- Régénération statique incrémentielle (ISR)
- Routage de sous-chemins internationalisé (i18n)
- Routage de domaine internationalisé (i18n)
- Intergiciel
- Variables d’environnement
- Optimisation de l'image
- Répertoire de l'application Next.js 13

Fonctions non prises en charge

- Routes d'API Edge (le middleware Edge n'est pas pris en charge)
- Régénération statique incrémentielle (ISR) à la demande
- Détection automatique des paramètres régionaux internationalisée (i18n)
- Diffusion de Next.js
- Exécution d'un intergiciel sur des actifs statiques et des images optimisées

Images du fichier Next.js

La taille de sortie maximale d'une image ne doit pas dépasser 4,3 Mo. Vous pouvez stocker un fichier image plus volumineux quelque part et utiliser le composant Image Next.js pour le redimensionner et l'optimiser au format Webp ou AVIF, puis l'utiliser dans une taille plus petite.

Notez que la documentation Next.js vous conseille d'installer le module de traitement d'image Sharp pour permettre à l'optimisation des images de fonctionner correctement en production. Toutefois, cela n'est pas nécessaire pour les déploiements d'Amplify. Amplify déploie automatiquement Sharp pour vous.

Tarification des applications Next.js SSR

Lors du déploiement de votre application SSR Next.js 12 ou version ultérieure, Amplify Hosting gère les ressources nécessaires pour déployer l'application SSR à votre place. Pour plus d'informations sur les frais de calcul d'Amplify Hosting, consultezAWS Amplify la section Tarification.

Déploiement d'une application SSR Next.js avec Amplify

Par défaut, Amplify déploie de nouvelles applications SSR à l'aide du service de calcul d'Amplify Hosting avec prise en charge de Next.js 12 et 13. Amplify Hosting Compute gère entièrement les ressources nécessaires au déploiement d'une application SSR. Les applications SSR de votre compte Amplify que vous avez déployées avant le 17 novembre 2022 utilisent le fournisseur SSR Classic (Next.js 11 uniquement).

Nous vous recommandons vivement de migrer les applications utilisant le SSR classique (Next.js 11 uniquement) vers le fournisseur de SSR de calcul Amplify Hosting. Amplify n'effectue pas de migrations automatiques pour vous. Vous devez migrer manuellement votre application, puis lancer une nouvelle version pour terminer la mise à jour. Pour obtenir des instructions, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

Suivez les instructions ci-dessous pour déployer une nouvelle application SSR.

Pour déployer une application SSR sur Amplify à l'aide du fournisseur de calcul SSR d'Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web.
3. Sélectionnez votre fournisseur GitHub, Bitbucket ou de AWS CodeCommit dépôt GitLab, puis choisissez Continuer.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :

    a. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel à connecter.

b. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.

c. Choisissez Suivant.

5. L'application nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

   - Pour permettre à Amplify de créer automatiquement un rôle et de l'associer à votre application

     - Dans la section Rôle IAM, choisissez Créer et utiliser un nouveau rôle de service.

   - Pour associer un rôle de service que vous avez créé précédemment

     a. Dans la section Rôle IAM, choisissez Utiliser un rôle de service existant.

     b. Choisissez le rôle à utiliser dans la liste.

6. Choisissez Suivant.

7. Sur la page Révision, choisissez Enregistrer et déployer.

Paramètres du fichier Package.json

Lorsque vous déployez une application Next.js, Amplify inspecte le script de génération de l'application dans le `package.json` fichier pour détecter si l'application est SSR ou SSG.

Voici un exemple de script de génération pour une application SSR Next.js. Le script de compilation `"next build"` indique que l'application prend en charge les pages SSG et SSR.

```
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start"
},
```

Voici un exemple de script de génération pour une application SSG Next.js. Le script de génération `"next build && next export"` indique que l'application ne prend en charge que les pages SSG.

```
"scripts": {
  "dev": "next dev",
  "build": "next build && next export",
```

```
  "start": "next start"
},
```

Amplifier les paramètres de construction

Après avoir inspecté le `package.json` fichier de votre application pour déterminer si vous déployez une application SSG ou SSR, Amplify vérifie les paramètres de compilation de l'application. Vous pouvez enregistrer les paramètres de compilation dans la console Amplify ou dans un `amplify.yml` fichier à la racine de votre référentiel. Pour de plus amples informations, veuillez consulter Configuration des paramètres de compilation.

Si Amplify détecte que vous déployez une application SSR Next.js et qu'aucun `amplify.yml` fichier n'est présent, il génère une spécification de construction pour l'application et la définit sur. `baseDirectory .next` Si vous déployez une application contenant un `amplify.yml` fichier, les paramètres de génération du fichier remplacent ceux de la console. Par conséquent, vous devez définir manuellement le « `baseDirectory` to » `.next` dans le fichier.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur `.next`. Cela indique que les artefacts de construction sont destinés à une application Next.js qui prend en charge les pages SSG et SSR.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Si Amplify détecte que vous déployez une application SSG, il génère une spécification de construction pour l'application et la définit sur. `baseDirectory out` Si vous déployez une

application dans laquelle un `amplify.yml` fichier est présent, vous devez définir manuellement le `baseDirectory` to `out` dans le fichier.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`out`. Cela indique que les artefacts de construction sont destinés à une application Next.js qui ne prend en charge que les pages SSG.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: out
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute

Lorsque vous déployez une nouvelle application Next.js, Amplify utilise par défaut la dernière version prise en charge de Next.js. Actuellement, le fournisseur de SSR de calcul Amplify Hosting prend en charge la version 13 de Next.js.

La console Amplify détecte les applications de votre compte qui ont été déployées avant la sortie du service de calcul Amplify Hosting avec prise en charge complète de Next.js 12 et 13. La console affiche une bannière d'information identifiant les applications dotées de branches déployées à l'aide de l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). Nous vous recommandons vivement de migrer vos applications vers le fournisseur de calcul SSR d'Amplify Hosting.

Vous devez migrer manuellement l'application et toutes ses branches de production en même temps. Une application ne peut pas contenir à la fois des branches Classic (Next.js 11 uniquement) et Next.js 12 ou 13.

Suivez les instructions suivantes pour migrer une application vers le fournisseur de calcul SSR d'Amplify Hosting.

Pour migrer une application vers le fournisseur de calcul SSR d'Amplify Hosting

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application Next.js que vous souhaitez migrer.

> **Note**
>
> Avant de migrer une application dans la console Amplify, vous devez d'abord mettre à jour le fichier package.json de l'application pour utiliser la version 12 ou 13 de Next.js.

3. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
4. Sur la page d'accueil de l'application, la console affiche une bannière si l'application possède des branches déployées à l'aide du fournisseur SSR Classic (Next.js 11 uniquement). Sur la bannière, choisissez Migrer.
5. Dans la fenêtre de confirmation de la migration, sélectionnez les trois instructions et choisissez Migrer.
6. Amplify créera et redéploiera votre application pour terminer la migration.

Annulation d'une migration SSR

Lorsque vous déployez une application Next.js, Amplify Hosting détecte les paramètres de votre application et définit la valeur de plate-forme interne de l'application. Il existe trois valeurs de plateforme valides. Une application SSG est définie sur la valeur `WEB` de la plateforme. Une application SSR utilisant Next.js version 11 est définie sur la valeur `WEB_DYNAMIC` de la plateforme. Une application SSR Next.js 12 ou 13 est définie sur la valeur `WEB_COMPUTE` de la plate-forme.

Lorsque vous migrez une application en suivant les instructions de la section précédente, Amplify change la valeur de plateforme de votre application de à`WEB_DYNAMIC`. `WEB_COMPUTE` Une fois la migration vers Amplify Hosting terminée, vous ne pouvez pas annuler la migration dans la console. Pour annuler la migration, vous devez utiliser le AWS Command Line Interface pour redéfinir la plateforme de l'application. `WEB_DYNAMIC` Ouvrez une fenêtre de terminal et entrez la commande suivante pour mettre à jour l'ID de l'application et la région avec vos informations uniques.

```
aws amplify update-app --app-id abcd1234 --platform WEB_DYNAMIC --region us-west-2
```

Ajout de la fonctionnalité SSR à une application Next.js statique

Vous pouvez ajouter la fonctionnalité SSR à une application statique (SSG) Next.js existante déployée avec Amplify. Avant de commencer le processus de conversion de votre application SSG en SSR, mettez-la à jour pour utiliser la version 12 ou 13 de Next.js et ajoutez la fonctionnalité SSR. Ensuite, vous devrez effectuer les étapes suivantes.

1. Utilisez le AWS Command Line Interface pour modifier le type de plateforme de l'application.
2. Ajoutez un rôle de service à l'application.
3. Mettez à jour le répertoire de sortie dans les paramètres de compilation de l'application.
4. Mettez à jour le `package.json` fichier de l'application pour indiquer que celle-ci utilise le SSR.

Mettre à jour la plateforme

Il existe trois valeurs valides pour le type de plateforme. Une application SSG est définie sur le type `WEB` de plateforme. Une application SSR utilisant Next.js version 11 est définie sur le type `WEB_DYNAMIC` de plate-forme. Pour les applications déployées sur Next.js 12 ou 13 à l'aide du SSR géré par Amplify Hosting Compute, le type de plateforme est défini sur. `WEB_COMPUTE`

Lorsque vous avez déployé votre application en tant qu'application SSG, Amplify a défini le type de plateforme sur. `WEB` Utilisez le AWS CLI pour modifier la plateforme de votre application`WEB_COMPUTE`. Ouvrez une fenêtre de terminal et entrez la commande suivante, en mettant à jour le texte en rouge avec votre identifiant d'application unique et votre région.

```
aws amplify update-app --app-id abcd1234 --platform WEB_COMPUTE --region us-west-2
```

Ajouter un rôle de service

Un rôle de service est le rôle AWS Identity and Access Management (IAM) qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Suivez ces étapes pour ajouter un rôle de service à une application SSG déjà déployée avec Amplify.

Pour ajouter un rôle de service

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Si vous n'avez pas encore créé de rôle de service dans votre compte Amplify, consultez Ajouter un rôle de service pour terminer cette étape préalable.

3. Choisissez l'application statique Next.js à laquelle vous souhaitez ajouter un rôle de service.
4. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
5. Sur la page des détails de l'application, choisissez Modifier
6. Pour Rôle de service, choisissez le nom d'un rôle de service existant ou le nom du rôle de service que vous avez créé à l'étape 2.
7. Choisissez Enregistrer.

Mettre à jour les paramètres de compilation

Avant de redéployer votre application avec la fonctionnalité SSR, vous devez mettre à jour les paramètres de compilation de l'application afin de définir le répertoire de sortie sur. `.next` Vous pouvez modifier les paramètres de compilation dans la console Amplify ou dans un `amplify.yml` fichier stocké dans votre dépôt. Pour plus d'informations, veuillez consulter Configuration des paramètres de compilation.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`.next`.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Mettez à jour le fichier package.json

Après avoir ajouté un rôle de service et mis à jour les paramètres de compilation, mettez à jour le `package.json` fichier de l'application. Comme dans l'exemple suivant, définissez le script de

génération sur "`next build`" pour indiquer que l'application Next.js prend en charge les pages SSG et SSR.

```
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start"
},
```

Amplify détecte la modification du `package.json` fichier dans votre dépôt et redéploie l'application avec la fonctionnalité SSR.

Rendre les variables d'environnement accessibles aux environnements d'exécution côté serveur

Amplify Hosting prend en charge l'ajout de variables d'environnement aux versions de votre application en les définissant dans la configuration du projet dans la console Amplify. Cependant, un composant serveur Next.js n'a pas accès à ces variables d'environnement par défaut. Ce comportement est intentionnel pour protéger les secrets stockés dans les variables d'environnement que votre application utilise pendant la phase de génération.

Pour rendre des variables d'environnement spécifiques accessibles à Next.js, vous pouvez modifier le fichier de spécification de build Amplify afin de les définir dans les fichiers d'environnement reconnus par Next.js. Cela permet à Amplify de charger ces variables d'environnement avant de créer l'application. L'exemple de spécification de construction suivant montre comment ajouter des variables d'environnement dans la section des commandes de construction.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - env | grep -e DB_HOST -e DB_USER -e DB_PASS >> .env.production
        - env | grep -e NEXT_PUBLIC_ >> .env.production
        - npm run build
  artifacts:
    baseDirectory: .next
```

```
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
      - .next/cache/**/*
```

Dans cet exemple, la section des commandes de génération inclut deux commandes qui écrivent des variables d'environnement dans le `.env.production` fichier avant l'exécution de la compilation de l'application. Amplify Hosting permet à votre application d'accéder à ces variables lorsque l'application reçoit du trafic.

La ligne suivante, tirée de la section des commandes de génération de l'exemple précédent, montre comment prendre une variable spécifique de l'environnement de génération et l'ajouter au `.env.production` fichier.

```
- env | grep -e DB_HOST -e DB_USER -e DB_PASS >> .env.production
```

Si les variables existent dans votre environnement de génération, le `.env.production` fichier contiendra les variables d'environnement suivantes.

```
DB_HOST=localhost
DB_USER=myuser
DB_PASS=mypassword
```

La ligne suivante, tirée de la section des commandes de construction de l'exemple précédent, montre comment ajouter une variable d'environnement avec un préfixe spécifique au `.env.production` fichier. Dans cet exemple, toutes les variables avec le préfixe `NEXT_PUBLIC_` sont ajoutées.

```
- env | grep -e NEXT_PUBLIC_ >> .env.production
```

Si plusieurs variables avec le `NEXT_PUBLIC_` préfixe existent dans l'environnement de construction, le `.env.production` fichier ressemblera à ce qui suit.

```
NEXT_PUBLIC_ANALYTICS_ID=abcdefghijk
NEXT_PUBLIC_GRAPHQL_ENDPOINT=uowelalsmlsadf
NEXT_PUBLIC_SEARCH_KEY=asdfiojslf
NEXT_PUBLIC_SEARCH_ENDPOINT=https://search-url
```

Variables d'environnement SSR pour monorepos

Si vous déployez une application SSR dans un monorepo et que vous souhaitez rendre des variables d'environnement spécifiques accessibles à Next.js, vous devez préfixer le `.env.production` fichier avec la racine de votre application. L'exemple de spécification de construction suivant pour une application Next.js dans un monorepo Nx montre comment ajouter des variables d'environnement dans la section des commandes de génération.

```
version: 1
applications:
  - frontend:
      phases:
        preBuild:
          commands:
            - npm ci
        build:
          commands:
            - env | grep -e DB_HOST -e DB_USER -e DB_PASS >> apps/app/.env.production
            - env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
            - npx nx build app
      artifacts:
        baseDirectory: dist/apps/app/.next
        files:
          - '**/*'
      cache:
        paths:
          - node_modules/**/*
      buildPath: /
    appRoot: apps/app
```

Les lignes suivantes de la section des commandes de génération de l'exemple précédent montrent comment prendre des variables spécifiques de l'environnement de génération et les ajouter au `.env.production` fichier d'une application dans un monorepo avec la racine de l'application. `apps/app`

```
- env | grep -e DB_HOST -e DB_USER -e DB_PASS >> apps/app/.env.production
- env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
```

Déploiement d'une application Next.js dans un monorepo

Amplify prend en charge les applications en monorepos génériques ainsi que les applications en monorepos créées à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo. Lorsque vous déployez votre application, Amplify détecte automatiquement le framework de construction monorepo que vous utilisez. Amplify applique automatiquement les paramètres de génération pour les applications dans un espace de travail npm, un espace de travail Yarn ou Nx. Notez que les applications pnpm et Turborepo nécessitent une configuration supplémentaire. Pour de plus amples informations, veuillez consulter Paramètres de construction de Monorepo.

Pour un exemple détaillé de Nx, consultez le billet de blog Share code between Next.js apps with Nx on AWS Amplify Hosting.

Amazon CloudWatch Logs pour les applications SSR

Amplify envoie des informations sur votre environnement d'exécution Next.js à Amazon CloudWatch Logs dans votre. Compte AWSLorsque vous déployez une application SSR, celle-ci nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

Si vous choisissez d'autoriser Amplify à créer un rôle IAM pour vous, le rôle sera déjà autorisé à créer des journaux. CloudWatch Si vous créez votre propre rôle IAM, vous devrez ajouter les autorisations suivantes à votre politique pour permettre à Amplify d'accéder à Amazon CloudWatch Logs.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
logs:PutLogEvents
```

Pour de plus amples informations sur les rôles de service, veuillez consulter Ajouter un rôle de service.

Prise en charge du SSR Amplify Next.js 11

Si vous avez déployé une application Next.js sur Amplify avant la sortie d'Amplify Hosting Compute le 17 novembre 2022, votre application utilise l'ancien fournisseur SSR d'Amplify, Classic (Next.js 11 uniquement). La documentation de cette section s'applique uniquement aux applications déployées à l'aide du fournisseur SSR Classic (Next.js 11 uniquement).

> **Note**
>
> Nous vous recommandons vivement de migrer vos applications Next.js 11 vers le fournisseur SSR géré par le calcul Amplify Hosting. Pour de plus amples informations, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

La liste suivante décrit les fonctionnalités spécifiques prises en charge par le fournisseur SSR Amplify Classic (Next.js 11 uniquement).

Fonctionnalités prises en charge

- Pages rendues côté serveur (SSR)
- Pages statiques
- Routes d'API
- Routes dynamiques
- Suivez tous les itinéraires
- SSG (génération statique)
- Régénération statique incrémentielle (ISR)
- Routage de sous-chemins internationalisé (i18n)
- Variables d'environnement

Fonctions non prises en charge

- Optimisation de l'image
- Régénération statique incrémentielle (ISR) à la demande
- Routage de domaine internationalisé (i18n)
- Détection automatique des paramètres régionaux internationalisée (i18n)
- Intergiciel
- Intergiciel Edge
- Routes de l'API Edge

Tarification des applications Next.js 11 SSR

Lors du déploiement de votre application Next.js 11 SSR, Amplify crée des ressources backend supplémentaires dans AWS votre compte, notamment :

- Un bucket Amazon Simple Storage Service (Amazon S3) qui stocke les ressources des actifs statiques de votre application. Pour plus d'informations sur les frais d'Amazon S3, consultez la section Tarification d'Amazon S3.
- Une CloudFront distribution Amazon pour diffuser l'application. Pour plus d'informations sur CloudFront les frais, consultez Amazon CloudFront Pricing.
- Quatre fonctions Lambda @Edge pour personnaliser le contenu diffusé. CloudFront

AWS Identity and Access Management autorisations pour les applications Next.js 11 SSR

Amplify nécessite des autorisations AWS Identity and Access Management (IAM) pour déployer une application SSR. Si vous ne disposez pas des autorisations minimales requises, vous recevrez un message d'erreur lorsque vous tenterez de déployer votre application SSR. Pour fournir à Amplify les autorisations requises, vous devez spécifier un rôle de service.

Pour créer un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom, voir. Ajouter un rôle de service Ces instructions montrent comment créer un rôle qui associe la politique `AdministratorAccess-Amplify` gérée.

La politique `AdministratorAccess-Amplify` gérée donne accès à de multiples AWS services, y compris aux actions IAM, et doit être considérée comme aussi puissante que la `AdministratorAccess` politique. Cette politique fournit plus d'autorisations que ce qui est nécessaire pour déployer votre application SSR.

Il est recommandé de suivre la meilleure pratique consistant à accorder le moindre privilège et à réduire les autorisations accordées au rôle de service. Au lieu d'accorder des autorisations d'accès d'administrateur à votre rôle de service, vous pouvez créer votre propre politique IAM gérée par le client qui accorde uniquement les autorisations requises pour déployer votre application SSR. Reportez-vous à la section Création de politiques IAM dans le Guide de l'utilisateur IAM pour obtenir des instructions sur la création d'une politique gérée par le client.

Si vous créez votre propre politique, reportez-vous à la liste suivante des autorisations minimales requises pour déployer une application SSR.

```
acm:DescribeCertificate
acm:ListCertificates
acm:RequestCertificate
cloudfront:CreateCloudFrontOriginAccessIdentity
cloudfront:CreateDistribution
cloudfront:CreateInvalidation
cloudfront:GetDistribution
cloudfront:GetDistributionConfig
cloudfront:ListCloudFrontOriginAccessIdentities
cloudfront:ListDistributions
cloudfront:ListDistributionsByLambdaFunction
cloudfront:ListDistributionsByWebACLId
cloudfront:ListFieldLevelEncryptionConfigs
cloudfront:ListFieldLevelEncryptionProfiles
cloudfront:ListInvalidations
cloudfront:ListPublicKeys
cloudfront:ListStreamingDistributions
cloudfront:UpdateDistribution
cloudfront:TagResource
cloudfront:UntagResource
cloudfront:ListTagsForResource
cloudfront:DeleteDistribution
iam:AttachRolePolicy
iam:CreateRole
iam:CreateServiceLinkedRole
iam:GetRole
iam:PutRolePolicy
iam:PassRole
iam:UpdateAssumeRolePolicy
iam:DeleteRolePolicy
lambda:CreateFunction
lambda:EnableReplication
lambda:DeleteFunction
lambda:GetFunction
lambda:GetFunctionConfiguration
lambda:PublishVersion
lambda:UpdateFunctionCode
lambda:UpdateFunctionConfiguration
lambda:ListTags
lambda:TagResource
lambda:UntagResource
lambda:ListEventSourceMappings
lambda:CreateEventSourceMapping
```

```
route53:ChangeResourceRecordSets
route53:ListHostedZonesByName
route53:ListResourceRecordSets
s3:CreateBucket
s3:GetAccelerateConfiguration
s3:GetObject
s3:ListBucket
s3:PutAccelerateConfiguration
s3:PutBucketPolicy
s3:PutObject
s3:PutBucketTagging
s3:GetBucketTagging
sqs:CreateQueue
sqs:DeleteQueue
sqs:GetQueueAttributes
sqs:SetQueueAttributes
amplify:GetApp
amplify:GetBranch
amplify:UpdateApp
amplify:UpdateBranch
```

Résolution des problèmes liés aux déploiements de Next.js 11 SSR

Si vous rencontrez des problèmes inattendus lors du déploiement d'une application SSR classique (Next.js 11 uniquement) avec Amplify, consultez les rubriques de résolution des problèmes suivantes.

Rubriques

- Les images de votre application Next.js ne se chargent pas
- Régions non prises en charge

Votre répertoire de sortie est remplacé

Le répertoire de sortie d'une application Next.js déployée avec Amplify doit être défini sur. `.next` Si le répertoire de sortie de votre application est remplacé, vérifiez le `next.config.js` fichier. Pour que le répertoire de sortie de build soit défini par défaut sur`.next`, supprimez la ligne suivante du fichier :

```
distDir: 'build'
```

Vérifiez que le répertoire de sortie est défini sur `.next` dans vos paramètres de compilation. Pour plus d'informations sur l'affichage des paramètres de compilation de votre application, consultezConfiguration des paramètres de compilation.

Voici un exemple des paramètres de génération d'une application où le paramètre `baseDirectory` est défini sur`.next`.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Vous obtenez une erreur 404 après le déploiement de votre site SSR

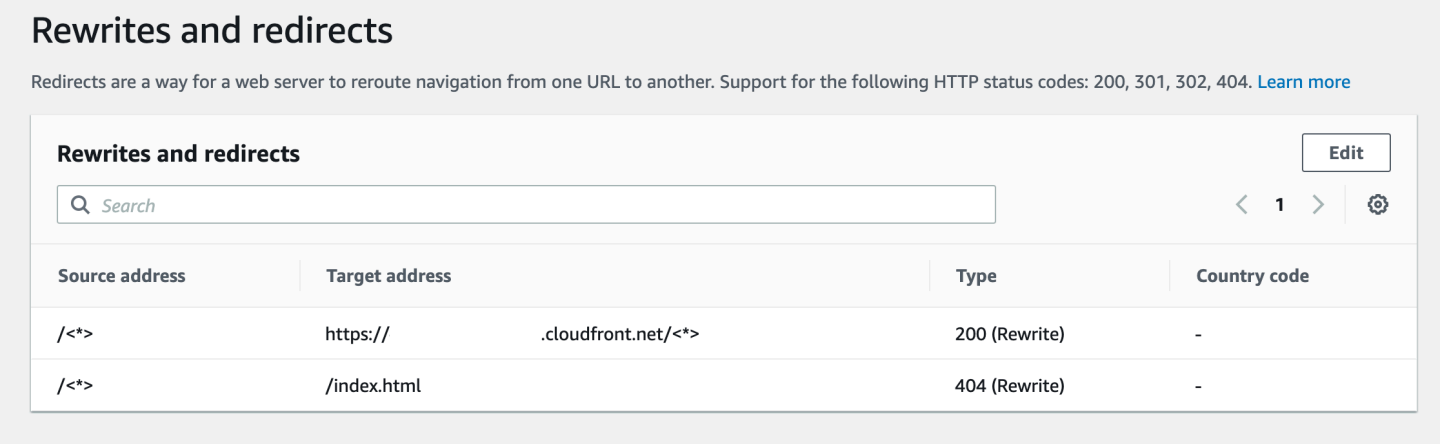
Si vous recevez une erreur 404 après le déploiement de votre site, le problème peut être dû au remplacement de votre répertoire de sortie. Pour vérifier votre `next.config.js` fichier et vérifier le

répertoire de sortie de compilation correct dans les spécifications de compilation de votre application, suivez les étapes décrites dans la rubrique précédente,Votre répertoire de sortie est remplacé.

Il manque la règle de réécriture pour les distributions CloudFront SSR dans votre application

Lorsque vous déployez une application SSR, Amplify crée une règle de réécriture pour CloudFront vos distributions SSR. Si vous ne pouvez pas accéder à votre application dans un navigateur Web, vérifiez que la règle de CloudFront réécriture existe pour votre application dans la console Amplify. S'il est absent, vous pouvez l'ajouter manuellement ou redéployer votre application.

Pour afficher ou modifier les règles de réécriture et de redirection d'une application dans la console Amplify, dans le volet de navigation, choisissez Paramètres de l'application, puis Réécritures et redirections. La capture d'écran suivante montre un exemple des règles de réécriture qu'Amplify crée pour vous lorsque vous déployez une application SSR. Notez que dans cet exemple, il existe une règle de CloudFront réécriture.



Votre application est trop volumineuse pour être déployée

Amplify limite la taille d'un déploiement SSR à 50 Mo. Si vous essayez de déployer une application SSR Next.js sur Amplify et que vous obtenez `RequestEntityTooLargeException` une erreur, cela signifie que votre application est trop volumineuse pour être déployée. Vous pouvez essayer de contourner ce problème en ajoutant du code de nettoyage du cache à votre `next.config.js` fichier.

Voici un exemple de code contenu dans le `next.config.js` fichier qui effectue le nettoyage du cache.

```
module.exports = {
    webpack: (config, { buildId, dev, isServer, defaultLoaders, webpack }) => {
```

```
        config.optimization.splitChunks.cacheGroups = { }
        config.optimization.minimize = true;
        return config
      },
}
```

Votre compilation échoue en raison d'une erreur de mémoire insuffisante

Next.js vous permet de mettre en cache des artefacts de build afin d'améliorer les performances lors des builds suivants. En outre, le AWS CodeBuild conteneur d'Amplify compresse et télécharge ce cache sur Amazon S3, en votre nom, afin d'améliorer les performances de compilation ultérieures. Cela pourrait entraîner l'échec de votre compilation en raison d'une erreur de mémoire insuffisante.

Effectuez les actions suivantes pour empêcher votre application de dépasser la limite de mémoire pendant la phase de création. Tout d'abord, `.next/cache/**/*` supprimez-le de la section cache.paths de vos paramètres de compilation. Supprimez ensuite la variable d'`NODE_OPTIONS`environnement de votre fichier de paramètres de compilation. Définissez plutôt la variable d'`NODE_OPTIONS`environnement dans la console Amplify pour définir la limite de mémoire maximale du nœud. Pour plus d'informations sur la définition des variables d'environnement à l'aide de la console Amplify, consultez. Définir les variables d'environnement

Après avoir apporté ces modifications, réessayez de créer. En cas de succès, ajoutez-le à `.next/cache/**/*` nouveau à la section cache.paths de votre fichier de paramètres de compilation.

Pour plus d'informations sur la configuration du cache Next.js afin d'améliorer les performances de compilation, consultez AWS CodeBuild sur le site Web Next.js.

Votre application possède à la fois des branches SSR et SSG

Vous ne pouvez pas déployer une application qui possède à la fois des branches SSR et SSG. Si vous devez déployer à la fois des branches SSR et SSG, vous devez déployer une application qui utilise uniquement des branches SSR et une autre qui utilise uniquement des branches SSG.

Votre application stocke les fichiers statiques dans un dossier avec un chemin réservé

Next.js peut servir des fichiers statiques à partir d'un dossier nommé `public` qui est stocké dans le répertoire racine du projet. Lorsque vous déployez et hébergez une application Next.js avec Amplify, votre projet ne peut pas inclure de dossiers avec le chemin. `public/static` Amplify réserve le `public/static` chemin à utiliser lors de la distribution de l'application. Si votre application inclut ce chemin, vous devez renommer le `static` dossier avant de le déployer avec Amplify.

Votre application a atteint une CloudFront limite

CloudFront les quotas de service limitent votre AWS compte à 25 distributions associées à des fonctions Lambda @Edge. Si vous dépassez ce quota, vous pouvez soit supprimer les CloudFront distributions non utilisées de votre compte, soit demander une augmentation du quota. Pour plus d'informations, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Les variables d'environnement ne sont pas transmises aux fonctions Lambda

Les variables d'environnement que vous spécifiez dans la console Amplify pour une application SSR ne sont pas répercutées sur les fonctions de l'application. AWS Lambda ConsultezRendre les variables d'environnement accessibles aux environnements d'exécution côté serveur, pour des instructions détaillées sur la façon d'ajouter des variables d'environnement auxquelles vous pouvez faire référence à partir de vos fonctions Lambda.

Les fonctions Lambda @Edge sont créées dans la région USA Est (Virginie du Nord)

Lorsque vous déployez une application Next.js, Amplify crée des fonctions Lambda @Edge pour personnaliser le contenu diffusé. CloudFront Les fonctions Lambda @Edge sont créées dans la région USA Est (Virginie du Nord), et non dans la région où votre application est déployée. Il s'agit d'une restriction Lambda @Edge. Pour plus d'informations sur les fonctions Lambda @Edge, consultez la section Restrictions relatives aux fonctions Edge dans le manuel Amazon CloudFront Developer Guide.

Votre application Next.js utilise des fonctionnalités non prises en charge

Les applications déployées avec Amplify prennent en charge les versions principales de Next.js jusqu'à la version 11. Pour une liste détaillée des fonctionnalités de Next.js prises en charge et non prises en charge par Amplify, voir. supported features

Lorsque vous déployez une nouvelle application Next.js, Amplify utilise par défaut la dernière version prise en charge de Next.js. Si vous avez une application Next.js existante que vous avez déployée sur Amplify avec une ancienne version de Next.js, vous pouvez migrer l'application vers le fournisseur de calcul SSR d'Amplify Hosting. Pour obtenir des instructions, veuillez consulter Migration d'une application Next.js 11 SSR vers Amplify Hosting Compute.

Les images de votre application Next.js ne se chargent pas

Lorsque vous ajoutez des images à votre application Next.js à l'aide du `next/image` composant, la taille de l'image ne peut pas dépasser 1 Mo. Lorsque vous déployez l'application sur Amplify, les

images de plus de 1 Mo renvoient une erreur 503. Cela est dû à une limite Lambda @Edge qui limite la taille d'une réponse générée par une fonction Lambda, y compris les en-têtes et le corps, à 1 Mo.

La limite de 1 Mo s'applique aux autres artefacts de votre application, tels que les fichiers PDF et les documents.

Régions non prises en charge

Amplify ne prend pas en charge le déploiement de l'application SSR classique (Next.js 11 uniquement) dans toutes les régions AWS où Amplify est disponible. Le SSR classique (Next.js 11 uniquement) n'est pas pris en charge dans les régions suivantes : Europe (Milan) eu-south-1, Moyen-Orient (Bahreïn) me-south-1 et Asie-Pacifique (Hong Kong) ap-east-1.

Configuration de domaines personnalisés

Vous pouvez connecter une application que vous avez déployée avec Amplify Hosting à un domaine personnalisé. Lorsque vous utilisez Amplify pour déployer votre application Web, Amplify l'héberge pour vous sur le `amplifyapp.com` domaine par défaut avec une URL telle que. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com` Lorsque vous connectez votre application à un domaine personnalisé, les utilisateurs voient que votre application est hébergée sur une URL personnalisée, telle que`https://www.example.com`.

Vous pouvez acheter un domaine personnalisé auprès d'un bureau d'enregistrement de domaines accrédité tel qu'Amazon Route 53 ou GoDaddy. Route 53 est le service Web DNS (Domain Name System) d'Amazon. Pour plus d'informations sur l'utilisation de Route 53, consultez Qu'est-ce qu'Amazon Route 53 ? Pour obtenir la liste des bureaux d'enregistrement de domaines accrédités tiers, consultez le répertoire des bureaux d'enregistrement accrédités sur le site Web de l'ICANN.

Lorsque vous configurez votre domaine personnalisé, vous pouvez utiliser le certificat géré par défaut fourni par Amplify pour vous ou vous pouvez utiliser votre propre certificat personnalisé. Vous pouvez modifier le certificat utilisé pour le domaine à tout moment. Pour obtenir des informations détaillées sur la gestion des certificats, consultezUtilisation de certificats SSL/TLS.

Avant de procéder à la configuration d'un domaine personnalisé, vérifiez que vous remplissez les conditions préalables suivantes.

- Vous êtes propriétaire d'un nom de domaine enregistré.
- Vous avez un certificat émis par ou importé dans AWS Certificate Manager.
- Vous avez déployé votre application sur Amplify Hosting.

  Pour plus d'informations sur la réalisation de cette étape, voir Commencer avec le code existant.
- Vous avez une connaissance de base des domaines et de la terminologie du DNS.

  Pour plus d'informations sur les domaines et le DNS, consultezComprendre la terminologie et les concepts du DNS.

Rubriques

- Comprendre la terminologie et les concepts du DNS
- Utilisation de certificats SSL/TLS
- Ajouter un domaine personnalisé géré par Amazon Route 53

- Ajouter un domaine personnalisé géré par un fournisseur DNS tiers
- Ajoutez un domaine personnalisé géré par GoDaddy
- Ajouter un domaine personnalisé géré par Google Domains
- Mettre à jour le certificat SSL/TLS d'un domaine
- Gérer les sous-domaines
- Sous-domaines Wildcard
- Configurer des sous-domaines automatiques pour un domaine personnalisé Amazon Route 53
- Résolution des problèmes liés aux domaines personnalisés

Comprendre la terminologie et les concepts du DNS

Si vous ne connaissez pas les termes et concepts associés au système de noms de domaine (DNS), les rubriques suivantes peuvent vous aider à comprendre les procédures d'ajout de domaines personnalisés.

Terminologie DNS

Vous trouverez ci-dessous une liste de termes communs au DNS. Ils peuvent vous aider à comprendre les procédures d'ajout de domaines personnalisés.

CNAME

Un nom d'enregistrement canonique (CNAME) est un type d'enregistrement DNS qui masque le domaine d'un ensemble de pages Web et les fait apparaître comme si elles se trouvaient ailleurs. Un CNAME pointe un sous-domaine vers un nom de domaine complet (FQDN). Par exemple, vous pouvez créer un nouvel enregistrement CNAME pour mapper le sous-domaine www.example.com, où www est le sous-domaine, au domaine FQDN branch-name.d1m7bkiki6tdw1.cloudfront.net attribué à votre application dans la console Amplify.

ANAME

Un enregistrement ANAME est similaire à un enregistrement CNAME, mais au niveau de la racine. Un ANAME pointe la racine de votre domaine vers un FQDN. Ce FQDN pointe vers une adresse IP.

Serveur de noms

Un serveur de noms est un serveur sur Internet spécialisé dans le traitement des requêtes concernant l'emplacement des différents services d'un nom de domaine. Si vous configurez

votre domaine dans Amazon Route 53, une liste de serveurs de noms est déjà attribuée à votre domaine.

Record NS

Un enregistrement NS pointe vers des serveurs de noms qui consultent les détails de votre domaine.

Vérification DNS

Un système de noms de domaine (DNS) est comme un annuaire téléphonique qui traduit des noms de domaine lisibles par l'homme en adresses IP adaptées aux ordinateurs. Lorsque vous tapez **`https://google.com`** dans un navigateur, une opération de recherche est effectuée dans le fournisseur DNS pour trouver l'adresse IP du serveur qui héberge le site Web.

Les fournisseurs DNS contiennent des enregistrements de domaines et leurs adresses IP correspondantes. Les enregistrements DNS les plus couramment utilisés sont les enregistrements CNAME, ANAME et NS.

Amplify utilise un enregistrement CNAME pour vérifier que vous êtes bien le propriétaire de votre domaine personnalisé. Si vous hébergez votre domaine avec Route 53, la vérification est effectuée automatiquement en votre nom. Toutefois, si vous hébergez votre domaine chez un fournisseur tiers tel que GoDaddy, vous devez mettre à jour manuellement les paramètres DNS de votre domaine et ajouter un nouvel enregistrement CNAME fourni par Amplify.

Processus d'activation du domaine personnalisé d'Amplify Hosting

Lorsque vous ajoutez un domaine personnalisé avec Amplify Hosting, vous devez effectuer un certain nombre d'étapes avant de pouvoir afficher votre application à l'aide de votre domaine personnalisé. La liste suivante décrit chaque étape du processus de configuration du domaine.

Création de SSL/TLS

Si vous utilisez un certificat géré, AWS Amplify émet un certificat SSL/TLS pour configurer un domaine personnalisé sécurisé.

Configuration et vérification SSL/TLS

Avant d'émettre un certificat géré, Amplify vérifie que vous êtes le propriétaire du domaine. Pour les domaines gérés par Amazon Route 53, Amplify met automatiquement à jour l'enregistrement de vérification DNS. Pour les domaines gérés en dehors de Route 53, vous devez ajouter

manuellement l'enregistrement de vérification DNS fourni dans la console Amplify dans votre domaine auprès d'un fournisseur DNS tiers.

Si vous utilisez un certificat personnalisé, vous êtes responsable de la validation de la propriété du domaine.

Activation du domaine

Le domaine a été vérifié avec succès. Pour les domaines gérés en dehors de Route 53, vous devez ajouter manuellement les enregistrements CNAME fournis dans la console Amplify dans votre domaine auprès d'un fournisseur DNS tiers.

Utilisation de certificats SSL/TLS

Un certificat SSL/TLS est un document numérique qui permet aux navigateurs Web d'identifier et d'établir des connexions réseau cryptées vers des sites Web à l'aide du protocole SSL/TLS sécurisé. Lorsque vous configurez votre domaine personnalisé, vous pouvez utiliser le certificat géré par défaut fourni par Amplify pour vous ou vous pouvez utiliser votre propre certificat personnalisé.

Avec un certificat géré, Amplify émet un certificat SSL/TLS pour tous les domaines connectés à votre application afin que tout le trafic soit sécurisé via HTTPS/2. Le certificat par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé dans les paramètres DNS de votre fournisseur de domaine. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

Pour utiliser un certificat personnalisé, vous devez obtenir un certificat auprès de l'autorité de certification tierce de votre choix. Importez ensuite le certificat dans AWS Certificate Manager. ACM est un service qui vous permet d'approvisionner, de gérer et de déployer facilement des certificats SSL/TLS publics et privés à utiliser avec vos ressources internes Services AWS connectées. Assurez-vous de demander ou d'importer le certificat dans la région USA Est (Virginie du Nord) (us-east-1).

Assurez-vous que votre certificat personnalisé couvre tous les sous-domaines que vous prévoyez d'ajouter. Vous pouvez utiliser un caractère générique au début de votre nom de domaine pour couvrir plusieurs sous-domaines. Par exemple, si votre domaine l'est`example.com`, vous pouvez inclure le domaine `*.example.com` générique. Cela couvrira les sous-domaines tels que `product.example.com` et. `api.example.com`

Une fois que votre certificat personnalisé sera disponible dans ACM, vous pourrez le sélectionner lors du processus de configuration du domaine. Pour obtenir des instructions sur l'importation de certificats dans AWS Certificate Manager, consultez la section Importation de certificats AWS Certificate Manager dans le guide de AWS Certificate Manager l'utilisateur.

Si vous renouvelez ou réimportez votre certificat personnalisé dans ACM, Amplify actualise les données du certificat associées à votre domaine personnalisé. Dans le cas de certificats importés, ACM ne gère pas les renouvellements automatiquement. Vous êtes responsable du renouvellement de vos certificats personnalisés et de leur réimportation.

Vous pouvez modifier le certificat utilisé pour un domaine à tout moment. Par exemple, vous pouvez passer du certificat géré par défaut à un certificat personnalisé ou passer d'un certificat personnalisé à un certificat géré. En outre, vous pouvez remplacer le certificat personnalisé utilisé par un autre certificat personnalisé. Pour obtenir des instructions sur la mise à jour des certificats, voir Mettre à jour le certificat SSL/TLS pour un domaine.

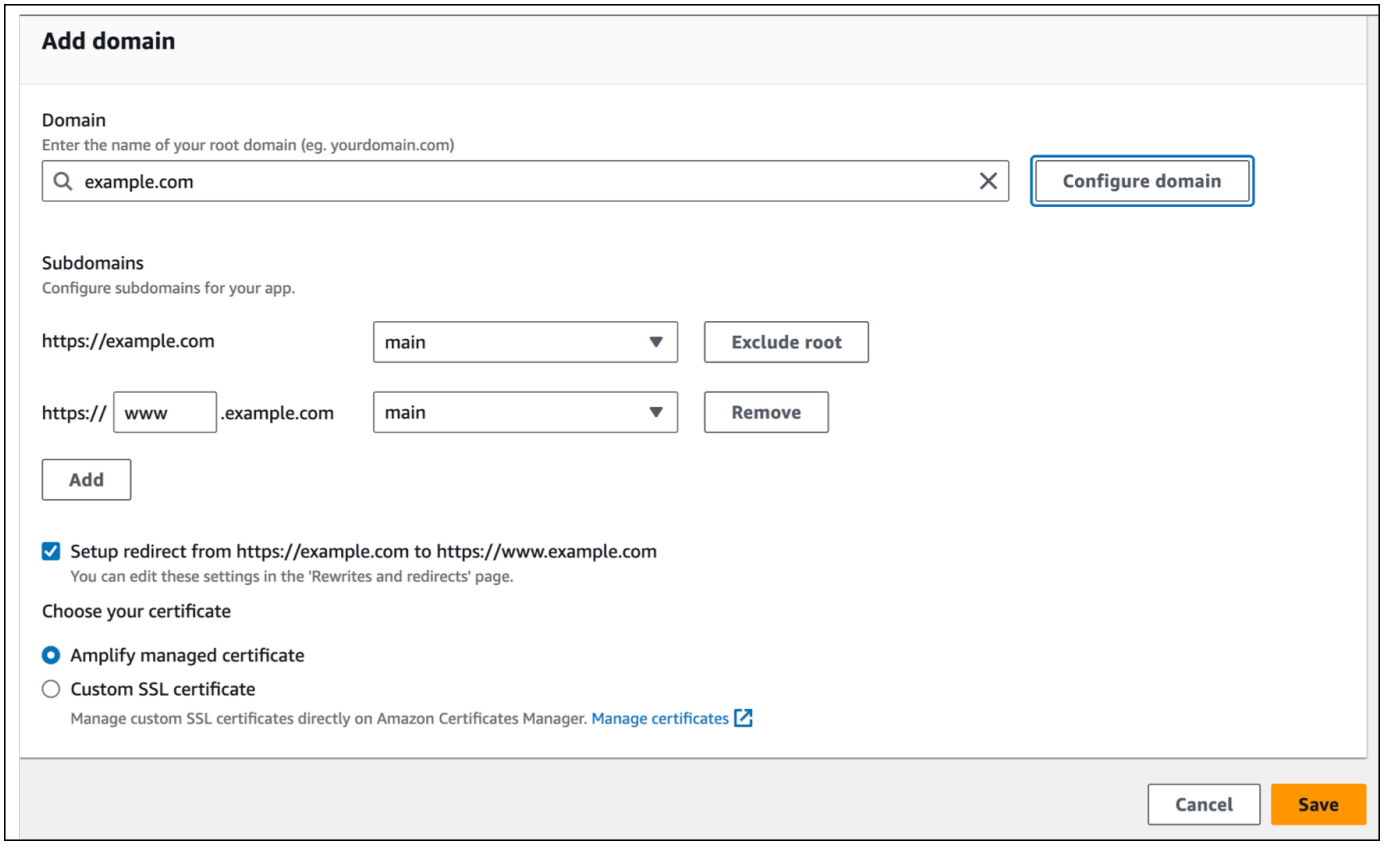
Ajouter un domaine personnalisé géré par Amazon Route 53

Pour ajouter un domaine personnalisé géré par Route 53

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez connecter à un domaine personnalisé.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Gestion des domaines.
4. Sur la page Gestion des domaines, choisissez Ajouter un domaine.
5. Dans le champ Domaine, entrez votre domaine racine. Par exemple, si le nom de votre domaine est https://example.com, entrez exemple.com pour Domaine.

   Lorsque vous commencez à taper, tous les domaines racines que vous gérez déjà dans Route 53 apparaissent dans la liste. Vous pouvez choisir le domaine que vous souhaitez utiliser dans la liste. Si vous ne possédez pas encore le domaine et qu'il est disponible, vous pouvez l'acheter sur Amazon Route 53.
6. Après avoir saisi votre nom de domaine, choisissez Configurer le domaine.
7. Par défaut, Amplify crée automatiquement deux entrées de sous-domaine pour votre domaine. Par exemple, si votre nom de domaine est exemple.com, vous verrez les sous-domaines https://www.example.com et https://example.com avec une redirection configurée du domaine racine vers le sous-domaine www.

(Facultatif) Vous pouvez modifier la configuration par défaut si vous souhaitez uniquement ajouter des sous-domaines. Pour modifier la configuration par défaut, choisissez Réécritures et redirections dans le volet de navigation, puis configurez votre domaine.



8. Choisissez le certificat SSL/TLS à utiliser. Vous pouvez soit utiliser le certificat géré par défaut fourni par Amplify pour vous, soit un certificat tiers personnalisé dans lequel vous avez importé. AWS Certificate Manager
   - Utilisez le certificat géré par Amplify par défaut.
     - Choisissez le certificat géré Amplify.
   - Utilisez un certificat tiers personnalisé.
     a. Choisissez un certificat SSL personnalisé.
     b. Sélectionnez le certificat à utiliser dans la liste.
9. Choisissez Enregistrer.

> **Note**
>
> La propagation du DNS et l'émission du certificat peuvent prendre jusqu'à 24 heures. Pour obtenir de l'aide sur la résolution des erreurs qui se produisent, consultezRésolution des problèmes liés aux domaines personnalisés.

Ajouter un domaine personnalisé géré par un fournisseur DNS tiers

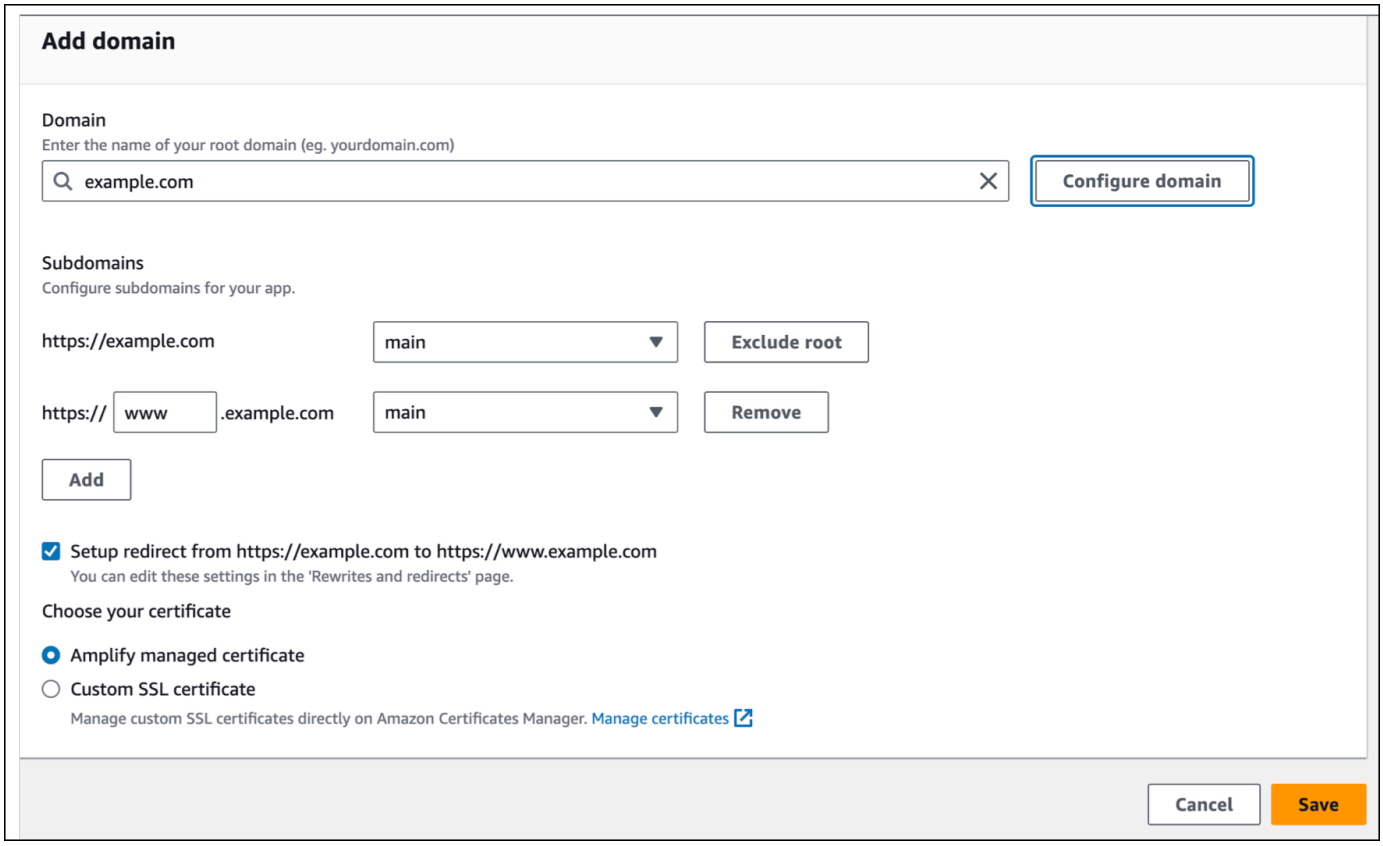
Si vous n'utilisez pas Amazon Route 53 pour gérer votre domaine, vous pouvez ajouter un domaine personnalisé géré par un fournisseur DNS tiers à votre application déployée avec Amplify.

Si vous utilisez GoDaddy Google Domains, consultez the section called "Ajoutez un domaine personnalisé géré par GoDaddy" ou consultez the section called "Ajouter un domaine personnalisé géré par Google Domains" les procédures spécifiques à ces fournisseurs.

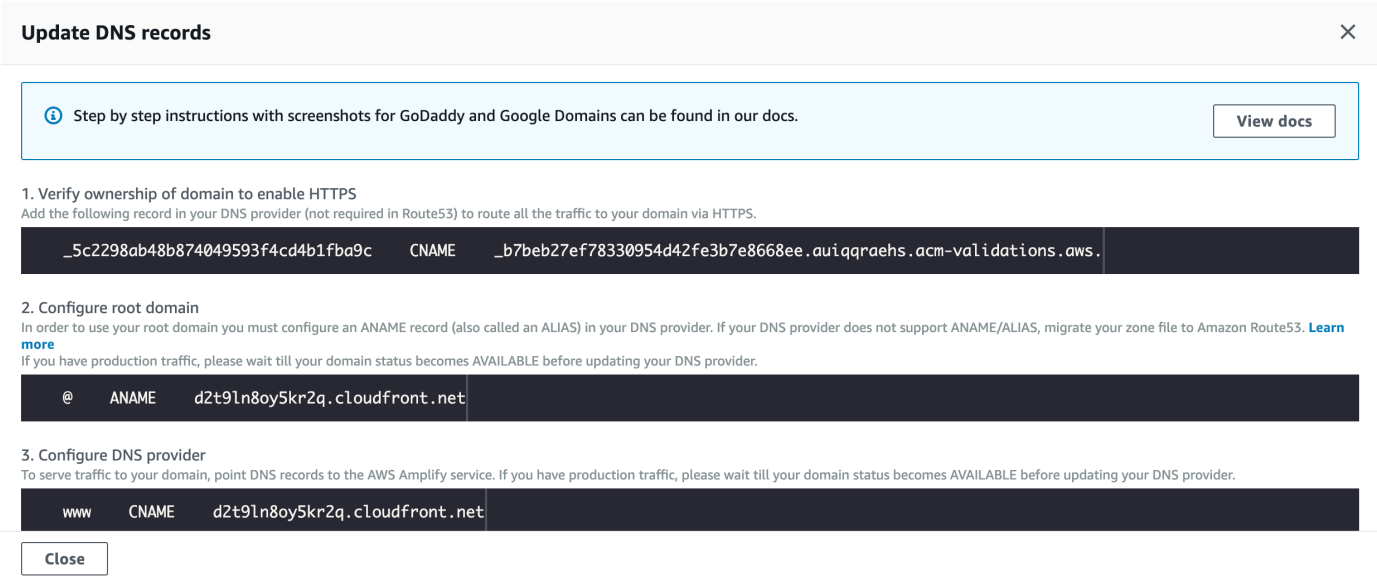
Pour ajouter un domaine personnalisé géré par un fournisseur DNS tiers

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un domaine personnalisé.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Gestion des domaines.
4. Sur la page Gestion des domaines, choisissez Ajouter un domaine.
5. Pour Domaine, entrez le nom de votre domaine racine, puis choisissez Configurer le domaine. Par exemple, si le nom de votre domaine est https://example.com, entrez**`example.com`**.
6. Par défaut, Amplify crée automatiquement deux entrées de sous-domaine pour votre domaine. Par exemple, si votre nom de domaine est exemple.com, vous verrez les sous-domaines https://www.example.com et https://example.com avec une redirection configurée du domaine racine vers le sous-domaine www.

   (Facultatif) Vous pouvez modifier la configuration par défaut si vous souhaitez uniquement ajouter des sous-domaines. Pour modifier la configuration par défaut, choisissez Réécritures et redirections dans le volet de navigation et configurez votre domaine.



7. Choisissez le certificat SSL/TLS à utiliser. Vous pouvez soit utiliser le certificat géré par défaut fourni par Amplify pour vous, soit un certificat tiers personnalisé dans lequel vous avez importé. AWS Certificate Manager
   - Utilisez le certificat géré par Amplify par défaut.
     - Choisissez le certificat géré Amplify.
   - Utilisez un certificat tiers personnalisé.
     a. Choisissez un certificat SSL personnalisé.
     b. Sélectionnez le certificat à utiliser dans la liste.
8. Choisissez Enregistrer.
9. Dans le menu Actions, choisissez Afficher les enregistrements DNS. À l'étape suivante, vous utiliserez ces enregistrements DNS pour mettre à jour vos enregistrements DNS auprès de votre fournisseur de domaine tiers.



10. Effectuez l'une des actions suivantes :

    - Si vous utilisez GoDaddy, rendez-vous surAjoutez un domaine personnalisé géré par GoDaddy.
    - Si vous utilisez Google Domains, rendez-vous surAjouter un domaine personnalisé géré par Google Domains.
    - Si vous utilisez un autre fournisseur DNS tiers, passez à l'étape suivante de cette procédure.

11. Accédez au site Web de votre fournisseur DNS, connectez-vous à votre compte et recherchez les paramètres de gestion DNS de votre domaine.

12. Configurez un CNAME pour pointer vers le serveur de AWS validation. Par exemple, si le serveur de validation est _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws, entrez _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws. Amplify utilise ces informations pour vérifier la propriété de votre domaine et générer un certificat SSL/TLS pour votre domaine. Une fois qu'Amplify aura validé la propriété de votre domaine, tout le trafic sera diffusé via HTTPS/2.

> **Note**
>
> Le certificat Amplify par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

> **Important**
>
> Il est important que vous exécutiez cette étape peu après avoir ajouté votre domaine personnalisé dans la console Amplify. L' AWS Certificate Manager (ACM) commence immédiatement à essayer de vérifier la propriété. Au fil du temps, les contrôles deviennent moins fréquents. Si vous ajoutez ou mettez à jour vos enregistrements CNAME quelques heures après avoir créé votre application, celle-ci peut rester bloquée dans l'état d'attente de vérification.

13. Configurez un deuxième enregistrement CNAME (par exemple, https://*.example.com) pour faire pointer vos sous-domaines vers le domaine Amplify. Si vous avez du trafic de production, nous vous recommandons de mettre à jour cet enregistrement CNAME une fois que le statut de votre domaine sera indiqué comme DISPONIBLE dans la console Amplify.
14. Configurez l'enregistrement ANAME/ALIAS pour qu'il pointe vers le domaine racine de votre **`amplifyapp`** domaine (par exemple https://example.com). Un enregistrement ANAME pointe la racine de votre domaine vers un nom d'hôte. Si vous avez du trafic de production, nous vous recommandons de mettre à jour votre enregistrement ANAME une fois que le statut de votre domaine sera indiqué comme DISPONIBLE dans la console. Pour les fournisseurs DNS qui ne prennent pas en charge ANAME/ALIAS, nous recommandons vivement de migrer votre DNS vers Route 53. Pour plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS.

> **Note**
>
> La vérification de la propriété du domaine et la propagation du DNS pour les domaines tiers peuvent prendre jusqu'à 48 heures. Pour obtenir de l'aide pour résoudre les erreurs qui se produisent, consultez la section Résolution des problèmes liés aux domaines personnalisés.

Ajoutez un domaine personnalisé géré par GoDaddy

Pour ajouter un domaine personnalisé géré par GoDaddy

1. Suivez les étapes 1 à 9 de la procéduresection called “Ajouter un domaine personnalisé géré par un fournisseur DNS tiers”.

2. Connectez-vous à votre GoDaddy compte.
3. Dans votre liste de domaines, recherchez le domaine à ajouter et choisissez Gérer le DNS.
4. Sur la page DNS, GoDaddy affiche la liste des enregistrements de votre domaine dans la section Enregistrements DNS. Vous devez ajouter deux nouveaux enregistrements CNAME.
5. Créez le premier enregistrement CNAME pour faire pointer vos sous-domaines vers le domaine Amplify.

   a. Dans la section Enregistrements DNS, choisissez Ajouter un nouvel enregistrement.

   b. Pour Type, choisissez CNAME.

   c. Dans Nom, entrez uniquement le sous-domaine. Par exemple, si votre sous-domaine est www.exemple.com, entrez www pour Nom.

   d. Pour Value, examinez vos enregistrements DNS dans la console Amplify, puis entrez la valeur. Si la console Amplify affiche le domaine de votre application sous la forme xxxxxxxxxxxxxx.cloudfront.net, entrez xxxxxxxxxxxxxx.cloudfront.net pour Value.

   e. Choisissez Enregistrer.
6. Créez le deuxième enregistrement CNAME qui pointe vers le serveur de validation AWS Certificate Manager (ACM). Un seul ACM validé génère un certificat SSL/TLS pour votre domaine.

   a. Pour Type, choisissez CNAME.

   b. Dans Nom, entrez le sous-domaine.

      Par exemple, si l'enregistrement DNS de la console Amplify permettant de vérifier la propriété de votre sous-domaine est _c3e2d7eaf1e656b73f46cd6980fdc0e.example.com, entrez uniquement _c3e2d7eaf1e656b73f46cd6980fdc0e pour Nom.

   c. Pour Value, entrez le certificat de validation ACM.

      Par exemple, si le serveur de validation est _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws, entrez _cjhwou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws pour Value.

   d. Choisissez Enregistrer.

> **Note**
>
> Le certificat Amplify par défaut généré par AWS Certificate Manager (ACM) est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

7. Cette étape n'est pas obligatoire pour les sous-domaines. GoDaddy ne prend pas en charge les enregistrements ANAME/ALIAS. Pour les fournisseurs DNS ne prenant pas en charge ANAME/ALIAS, nous vous recommandons vivement de migrer votre DNS vers Amazon Route 53. Pour plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS.

   Si vous souhaitez rester GoDaddy votre fournisseur et mettre à jour le domaine racine, ajoutez le transfert et configurez un transfert de domaine :

   a. Sur la page DNS, recherchez le menu en haut de la page et choisissez Forwarding.
   b. Dans la section Domaine, choisissez Ajouter un transfert.
   c. Choisissez http ://, puis entrez le nom du sous-domaine vers lequel vous souhaitez transférer (par exemple, www.example.com) pour l'URL de destination.
   d. Pour le type de transfert, choisissez Temporary (302).
   e. Choisissez, Enregistrer.

Ajouter un domaine personnalisé géré par Google Domains

Pour ajouter un domaine personnalisé géré par Google Domains

1. Suivez les étapes 1 à 9 de la procédure pour ajouter un domaine personnalisé géré par un fournisseur DNS tiers.
2. Connectez-vous à votre compte sur https://domains.google.com et choisissez Mes domaines dans le volet de navigation de gauche.
3. Dans votre liste de domaines, recherchez le domaine à ajouter et choisissez Gérer.
4. Dans le volet de navigation de gauche, choisissez DNS. Google affiche les enregistrements de ressources de votre domaine. Vous devez ajouter deux nouveaux enregistrements CNAME.

5. Créez le premier enregistrement CNAME pour pointer tous les sous-domaines vers le domaine Amplify comme suit :

   a. Pour Nom d'hôte, entrez uniquement le nom du sous-domaine. Par exemple, si votre sous-domaine est www.exemple.com, entrez www comme nom d'hôte.

   b. Pour Type, choisissez CNAME.

   c. Pour Data, entrez la valeur disponible dans la console Amplify.

      Si la console Amplify affiche le domaine de votre application sous la forme d111111abcdef8.cloudfront.net, entrez d111111abcdef8.cloudfront.net pour les données.

6. Créez le deuxième enregistrement CNAME qui pointe vers le serveur de validation AWS Certificate Manager (ACM). Un seul ACM validé génère un certificat SSL/TLS pour votre domaine.

   a. Dans Nom d'hôte, entrez le sous-domaine.

      Par exemple, si l'enregistrement DNS de la console Amplify permettant de vérifier la propriété de votre sous-domaine est _c3e2d7eaf1e656b73f46cd6980fdc0e.example.com, entrez uniquement _c3e2d7eaf1e656b73f46cd6980fdc0e pour le nom d'hôte.

   b. Pour Type, choisissez CNAME.

   c. Pour Data, entrez le certificat de validation ACM.

      Par exemple, si le serveur de validation est _cf1z2npwt9vzexample93c1j4xzc92wl.2te3iym6kzr.acm-validations.aws. , entrez _cf1z2npwt9vzexample93c1j4xzc92wl.2te3iym6kzr.acm-validations.aws. pour Data.

7. Choisissez Enregistrer.

> **Note**
>
> Le certificat Amplify ; généré par AWS Certificate Manager (ACM) par défaut est valide pendant 13 mois et se renouvelle automatiquement tant que votre application est hébergée chez Amplify. Amplify ne peut pas renouveler le certificat si l'enregistrement de vérification CNAME a été modifié ou supprimé. Vous devez supprimer et ajouter à nouveau le domaine dans la console Amplify.

8. La prise en charge des enregistrements ANAME/ALIAS par Google Domains est en cours de prévisualisation. Pour les fournisseurs DNS ne prenant pas en charge ANAME/ALIAS,

nous vous recommandons vivement de migrer votre DNS vers Amazon Route 53. Pour plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS. Si vous souhaitez conserver Google Domains en tant que fournisseur et mettre à jour le domaine racine, configurez un sous-domaine redirigé. Localisez la page du site Web correspondant à votre domaine Google. Choisissez ensuite le domaine de transfert et configurez votre transfert sur la page de transfert Web.

> **Note**
>
> Les mises à jour de vos paramètres DNS pour un domaine Google peuvent prendre jusqu'à 48 heures pour prendre effet. Pour obtenir de l'aide sur la résolution des erreurs qui se produisent, consultez la section Résolution des problèmes liés aux domaines personnalisés.

Mettre à jour le certificat SSL/TLS d'un domaine

Vous pouvez modifier le certificat SSL/TLS utilisé pour un domaine à tout moment. Par exemple, vous pouvez passer d'un certificat géré à un certificat personnalisé. Vous pouvez également modifier le certificat personnalisé utilisé pour le domaine. Pour plus d'informations sur les certificats, consultez la section Utilisation de certificats SSL/TLS.

Suivez la procédure ci-dessous pour mettre à jour le type de certificat ou le certificat personnalisé utilisé pour un domaine.

Pour mettre à jour le certificat d'un domaine

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application que vous souhaitez mettre à jour.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion du domaine, choisissez Gérer le domaine.
5. Sur la page de détails de votre domaine, recherchez la section Choisissez votre certificat. La procédure de mise à jour de votre certificat varie en fonction du type de modification que vous souhaitez apporter.
   - Pour passer d'un certificat personnalisé au certificat géré par Amplify par défaut
     - Choisissez le certificat géré Amplify.

- Pour passer d'un certificat géré à un certificat personnalisé

  a. Choisissez un certificat SSL personnalisé.

  b. Sélectionnez le certificat à utiliser dans la liste.

- Pour remplacer un certificat personnalisé par un autre certificat personnalisé

  - Pour le certificat SSL personnalisé, sélectionnez le nouveau certificat à utiliser dans la liste.

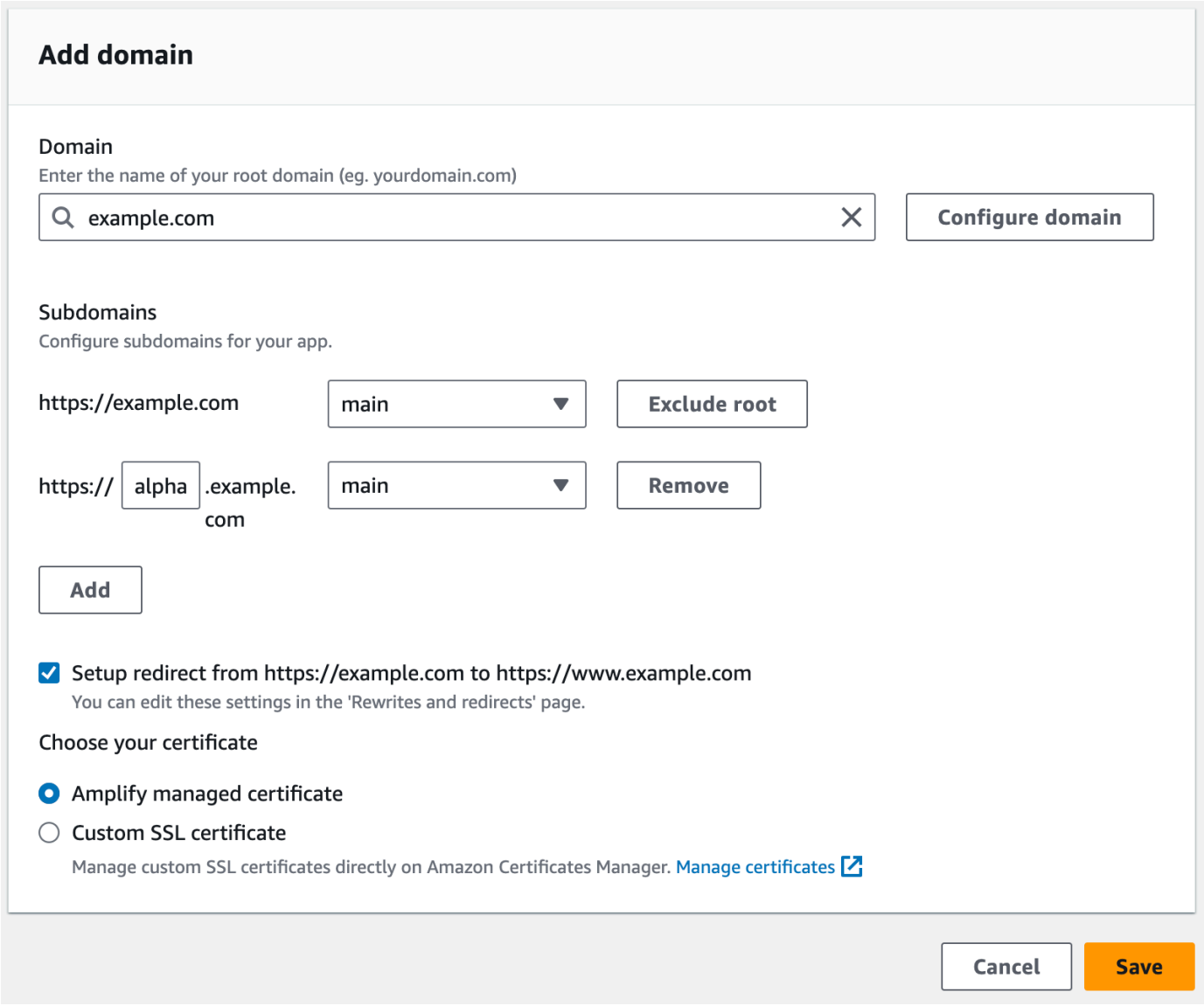
6. Choisissez Mettre à jour. Les détails du statut du domaine indiqueront qu'Amplify a lancé le processus de création SSL pour un certificat géré ou le processus de configuration pour un certificat personnalisé.

Gérer les sous-domaines

Un sous-domaine est la partie de votre URL qui apparaît avant votre nom de domaine. Par exemple, www est le sous-domaine de www.amazon.com et aws est le sous-domaine de aws.amazon.com. Si vous possédez déjà un site Web de production, vous souhaiterez peut-être connecter uniquement un sous-domaine. Les sous-domaines peuvent également être à plusieurs niveaux, par exemple beta.alpha.example.com possède le sous-domaine à plusieurs niveaux beta.alpha.

Pour ajouter un sous-domaine uniquement

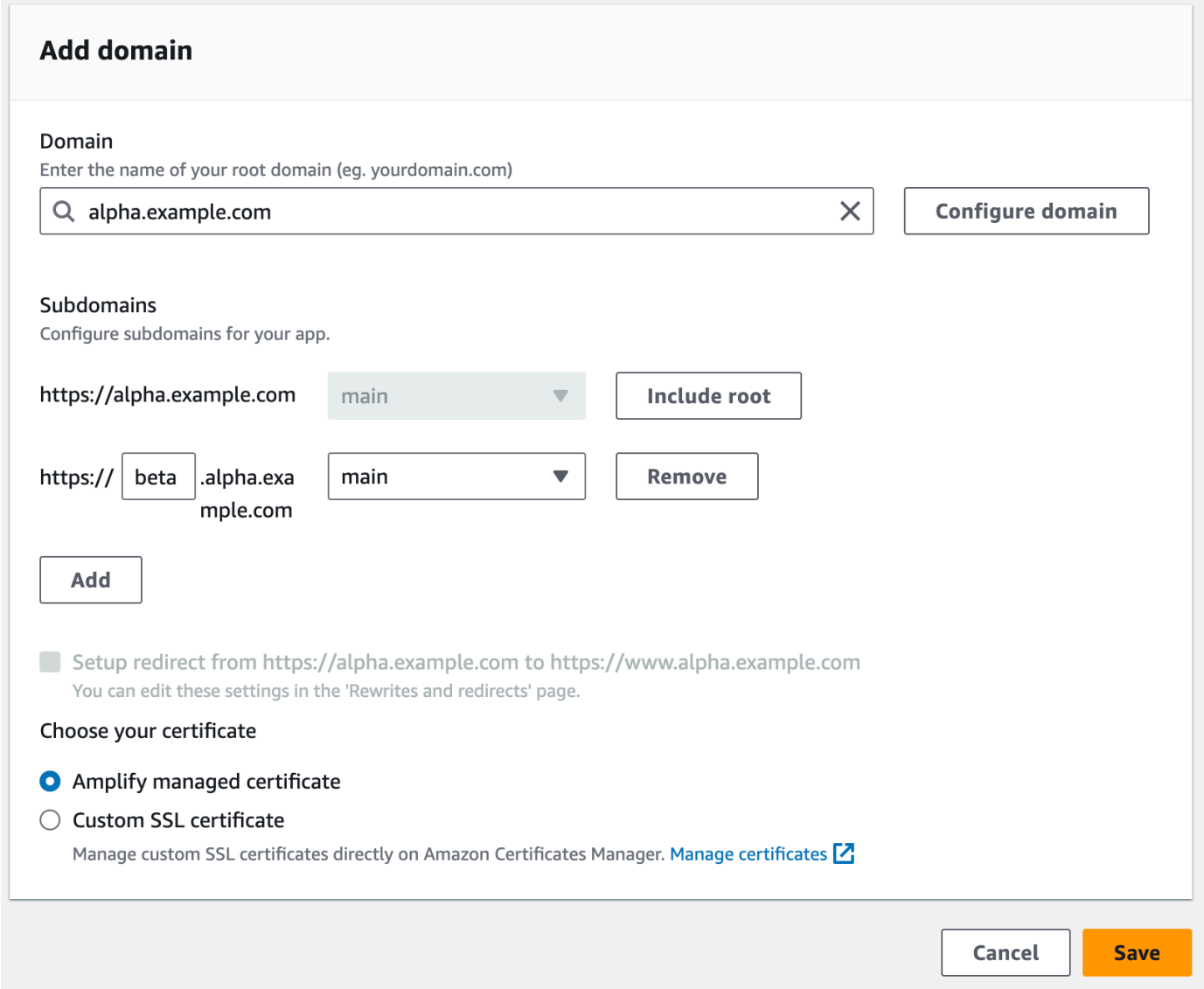
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un sous-domaine.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion des domaines, choisissez Ajouter un domaine.
5. Pour Domaine, entrez le nom de votre domaine racine, puis choisissez Configurer le domaine. Par exemple, si le nom de votre domaine est https://example.com, entrez exemple.com pour Domaine.
6. Choisissez Exclure la racine et modifiez le nom du sous-domaine. Par exemple, si le domaine est exemple.com, vous pouvez le modifier pour ajouter uniquement le sous-domaine alpha, comme indiqué dans la capture d'écran suivante.



Pour ajouter un sous-domaine à plusieurs niveaux

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez ajouter un sous-domaine multiniveaux.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion des domaines, choisissez Ajouter un domaine.
5. Pour Domaine, entrez le nom d'un domaine avec un sous-domaine, choisissez Exclure la racine et modifiez le sous-domaine pour ajouter un nouveau niveau.

Par exemple, si vous avez un domaine appelé alpha.exemple.com et que vous souhaitez créer un sous-domaine à plusieurs niveaux beta.alpha.example.com, vous devez entrer beta comme valeur de sous-domaine, comme indiqué dans la capture d'écran suivante.



Pour ajouter ou modifier un sous-domaine

Après avoir ajouté un domaine personnalisé à une application, vous pouvez modifier un sous-domaine existant ou en ajouter un nouveau.

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez gérer les sous-domaines.

3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion du domaine, choisissez Gérer le domaine.
5. Dans Modifier le domaine, vous pouvez modifier vos sous-domaines existants selon vos besoins.
6. (Facultatif) Pour ajouter un nouveau sous-domaine, choisissez Ajouter.
7. Choisissez Mettre à jour pour enregistrer vos modifications.

Sous-domaines Wildcard

Amplify Hosting prend désormais en charge les sous-domaines génériques. Un sous-domaine générique est un sous-domaine fourre-tout qui vous permet de rediriger des sous-domaines existants et inexistants vers une branche spécifique de votre application. Lorsque vous utilisez un caractère générique pour associer tous les sous-domaines d'une application à une branche spécifique, vous pouvez proposer le même contenu aux utilisateurs de votre application dans n'importe quel sous-domaine et éviter de configurer chaque sous-domaine individuellement.

Pour créer un sous-domaine générique, spécifiez un astérisque (*) comme nom de sous-domaine. Par exemple, si vous spécifiez le sous-domaine générique `*.example.com` pour une branche spécifique de votre application, toute URL se terminant par exemple.com sera acheminée vers la branche. Dans ce cas, les demandes concernant `dev.example.com` et `prod.example.com` seront acheminées vers le `*.example.com` sous-domaine.

Notez qu'Amplify prend en charge les sous-domaines génériques uniquement pour un domaine personnalisé. Vous ne pouvez pas utiliser cette fonctionnalité avec le `amplifyapp.com` domaine par défaut.

Les exigences suivantes s'appliquent aux sous-domaines génériques :

- Le nom du sous-domaine doit être indiqué uniquement par un astérisque (*).
- Vous ne pouvez pas utiliser un caractère générique pour remplacer une partie du nom d'un sous-domaine, comme ceci : *domain.example.com.
- Vous ne pouvez pas remplacer un sous-domaine au milieu d'un nom de domaine, comme ceci : sous-domain.*.example.com.
- Par défaut, tous les certificats fournis par Amplify couvrent tous les sous-domaines d'un domaine personnalisé.

Pour ajouter ou supprimer un sous-domaine générique

Après avoir ajouté un domaine personnalisé à une application, vous pouvez ajouter un sous-domaine générique pour une branche d'application.

1. Connectez-vous à la console Amplify Hosting AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez gérer les sous-domaines génériques.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion du domaine, choisissez Gérer le domaine.
5. Dans Modifier le domaine, vous pouvez ajouter ou supprimer des sous-domaines génériques.
   - Pour ajouter un nouveau sous-domaine générique
     a. Choisissez Ajouter.
     b. Pour le sous-domaine, entrez un*****.
     c. Pour la branche de votre application, sélectionnez un nom de branche dans la liste.

        Dans l'exemple de capture d'écran suivant, le sous-domaine `*.example.com` générique a été créé pour la `dev` branche de l'application.

     d. Choisissez Mettre à jour pour enregistrer vos modifications.
   - Pour supprimer un sous-domaine générique
     a. Choisissez Supprimer à côté du nom du sous-domaine. Le trafic vers un sous-domaine qui n'est pas configuré explicitement s'arrête et Amplify Hosting renvoie un code d'état 404 à ces demandes.
     b. Choisissez Mettre à jour pour enregistrer vos modifications.

Configurer des sous-domaines automatiques pour un domaine personnalisé Amazon Route 53

Une fois qu'une application est connectée à un domaine personnalisé dans Route 53, Amplify vous permet de créer automatiquement des sous-domaines pour les succursales nouvellement connectées. Par exemple, si vous connectez votre branche de développement, Amplify peut créer automatiquement dev.exampledomain.com. Lorsque vous supprimez une branche, tous les sous-domaines associés sont automatiquement supprimés.

Pour configurer la création automatique de sous-domaines pour les succursales nouvellement connectées

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez une application connectée à un domaine personnalisé géré dans Route 53.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Gestion des domaines.
4. Sur la page Gestion du domaine, choisissez Gérer le domaine.
5. Cochez la case Détection automatique des sous-domaines en bas à gauche.

> **Note**
>
> Cette fonctionnalité n'est disponible que pour les domaines racine, par exemple exempledomain.com. La console Amplify n'affiche pas cette case à cocher si votre domaine est déjà un sous-domaine, tel que dev.exampledomain.com.

Aperçus Web avec sous-domaines

Une fois que vous avez activé la détection automatique des sous-domaines en suivant les instructions précédentes, les aperçus Web par pull request de votre application seront également accessibles avec les sous-domaines créés automatiquement. Lorsqu'une pull request est fermée, la branche et le sous-domaine associés sont automatiquement supprimés. Pour plus d'informations sur la configuration des aperçus Web pour les pull requests, consultezAperçus Web pour les pull requests.

Résolution des problèmes liés aux domaines personnalisés

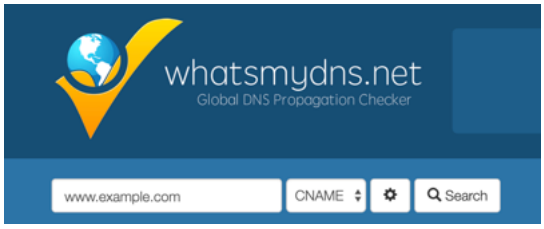
Si vous rencontrez des problèmes lors de l'ajout d'un domaine personnalisé à une application dans la AWS Amplify console, consultez les rubriques suivantes de cette section pour obtenir de l'aide à la résolution des problèmes.

Si vous ne trouvez pas de solution à votre problème ici, contactez AWS Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

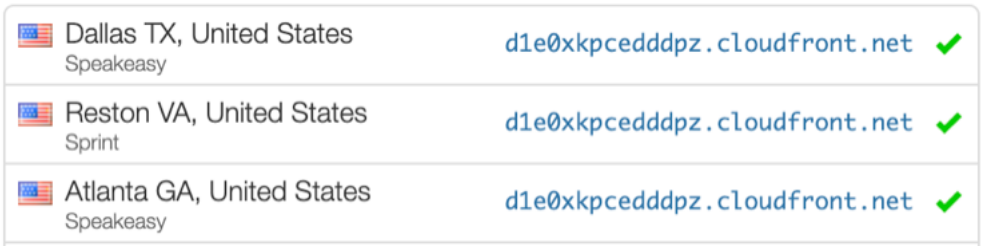
Rubriques

Comment vérifier que la résolution de mon enregistrement CNAME fonctionne ?

1. Après avoir mis à jour vos enregistrements DNS auprès de votre fournisseur de domaine tiers, vous pouvez utiliser un outil tel que dig ou un site Web gratuit tel que https://www.whatsmydns.net/ pour vérifier que votre enregistrement CNAME est correctement résolu. La capture d'écran suivante montre comment utiliser whatsmydns.net pour vérifier votre enregistrement CNAME pour le domaine www.example.com.



2. Choisissez Rechercher, et whatsmydns.net affiche les résultats de votre CNAME. La capture d'écran suivante est un exemple de liste de résultats qui vérifient que le CNAME correspond correctement à une URL cloudfront.net.



Mon domaine hébergé chez un tiers est bloqué dans l'état En attente de vérification

1. Si votre domaine personnalisé est bloqué dans l'état En attente de vérification, vérifiez que vos CNAME enregistrements sont en cours de résolution. Consultez la rubrique de résolution des problèmes précédente, Comment puis-je vérifier que mes problèmes sont résolusCNAME, pour obtenir des instructions sur l'exécution de cette tâche.
2. Si vos CNAME enregistrements ne sont pas résolus, vérifiez que l'CNAMEentrée existe dans vos paramètres DNS auprès de votre fournisseur de domaine.

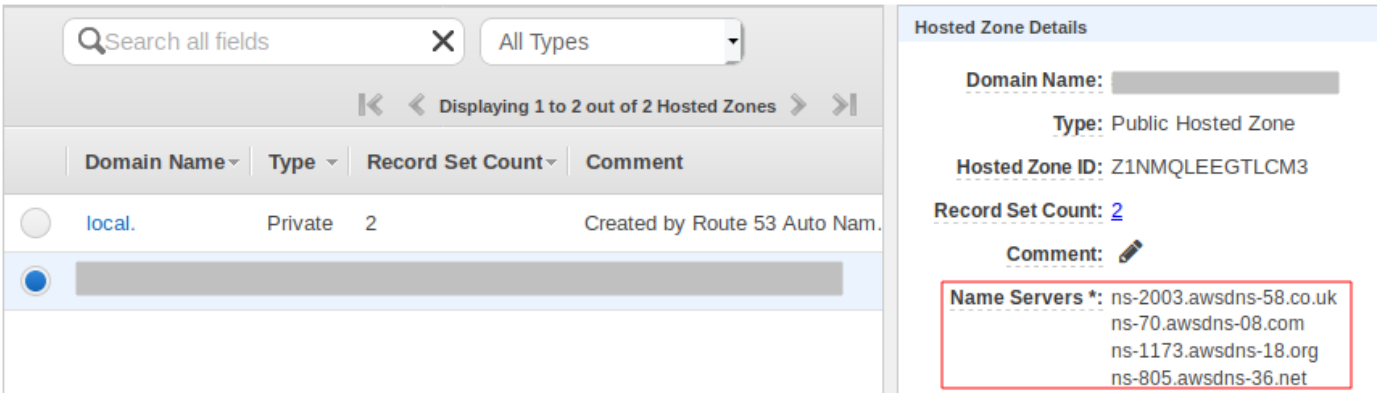
   > **Important**
   >
   > Il est important de mettre à jour vos CNAME enregistrements dès que vous créez votre domaine personnalisé. Une fois votre application créée dans la console Amplify, votre CNAME dossier est vérifié toutes les quelques minutes pour déterminer s'il est résolu. Si le problème persiste au bout d'une heure, la vérification est effectuée toutes les quelques heures, ce qui peut retarder la mise en service de votre domaine. Si vous avez ajouté ou mis à jour vos CNAME enregistrements quelques heures après avoir créé votre application, il est fort probable que votre application reste bloquée dans l'état En attente de vérification.

3. Si vous avez vérifié que l'CNAMEenregistrement existe, il se peut qu'il y ait un problème avec votre fournisseur DNS. Vous pouvez soit contacter le fournisseur DNS pour savoir pourquoi la vérification DNS ne résout pas le problème, CNAME soit migrer votre DNS vers Route 53. Pour plus d'informations, consultez Faire d'Amazon Route 53 le service DNS d'un domaine existant.

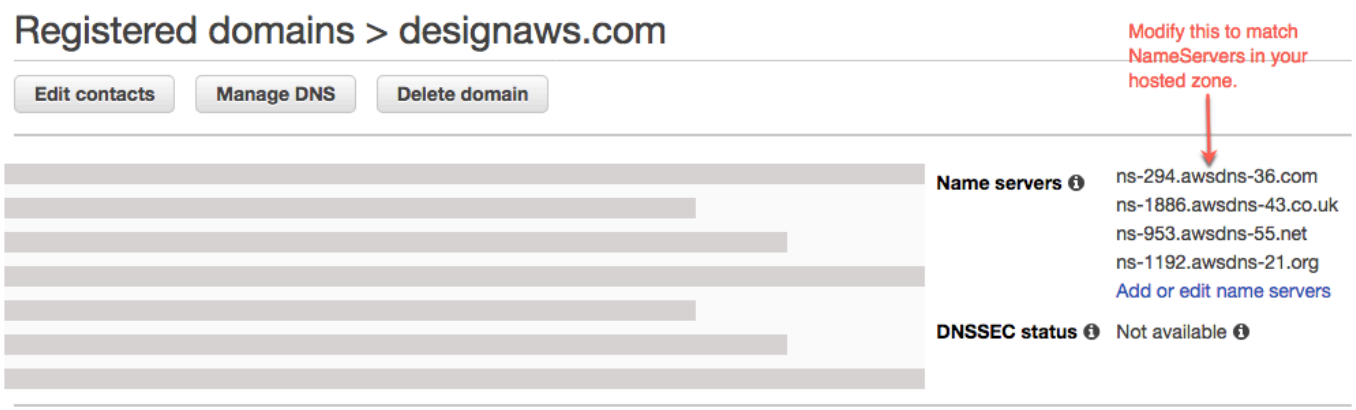
Mon domaine hébergé par Amazon Route 53 est bloqué dans l'état En attente de vérification

Si vous avez transféré votre domaine vers Amazon Route 53, il est possible que votre domaine possède des serveurs de noms différents de ceux émis par Amplify lors de la création de votre application. Procédez comme suit pour diagnostiquer la cause de l'erreur.

1. Connectez-vous à la console Amazon Route 53
2. Dans le volet de navigation, choisissez Hosted Zones, puis choisissez le nom du domaine que vous connectez.
3. Enregistrez les valeurs du serveur de noms dans la section Détails de la zone hébergée. Vous avez besoin de ces valeurs pour passer à l'étape suivante. La capture d'écran suivante de la console Route 53 montre l'emplacement des valeurs du serveur de noms dans le coin inférieur droit.



4. Dans le panneau de navigation, choisissez Registered domains (Domaines membres). Vérifiez que les serveurs de noms affichés dans la section Domaines enregistrés correspondent aux valeurs des serveurs de noms que vous avez enregistrées à l'étape précédente dans la section Détails de la zone hébergée. S'ils ne correspondent pas, modifiez les valeurs du serveur de noms pour qu'elles correspondent aux valeurs de votre zone hébergée. La capture d'écran suivante de la console Route 53 montre l'emplacement des valeurs du serveur de noms sur le côté droit.



5. Si cela ne résout pas le problème, contactez AWS Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

Je reçois une erreur CNAME AlreadyExistsException

Si une AlreadyExistsException erreur CNAME s'affiche, cela signifie que l'un des noms d'hôte auxquels vous avez essayé de vous connecter (un sous-domaine ou le domaine apex) est déjà déployé sur une autre distribution Amazon CloudFront . Procédez comme suit pour diagnostiquer la cause de l'erreur.

1. Connectez-vous à la CloudFrontconsole Amazon et vérifiez que ce domaine n'est pas déployé sur une autre distribution. Un seul CNAME enregistrement peut être joint à une CloudFront distribution à la fois.
2. Si vous avez déjà déployé le domaine sur une CloudFront distribution, vous devez le supprimer.
   a. Choisissez Distributions dans le menu de navigation de gauche.
   b. Sélectionnez le nom de la distribution à modifier.
   c. Choisissez l'onglet Général. Dans la section Settings (Paramètres), choisissez Edit (Modifier).
   d. Supprimez le nom de domaine du nom de domaine alternatif (CNAME). Choisissez ensuite Enregistrer les modifications.
3. Vérifiez si ce domaine est connecté à une autre application Amplify que vous possédez. Si tel est le cas, assurez-vous que vous n'essayez pas de réutiliser l'un des noms d'hôte. Si vous utilisez www.example.com pour une autre application, vous ne pouvez pas utiliser www.example.com avec l'application que vous êtes en train de connecter. Vous pouvez utiliser d'autres sous-domaines, tels que blog.example.com.

4. Si ce domaine a été connecté avec succès à une autre application puis supprimé au cours de la dernière heure, réessayez au bout d'une heure au moins. Si cette exception persiste après 6 heures, contactez AWS Support. Pour obtenir plus d'informations, consultez la section Creating a support case (Création d'un cas de support) dans le Guide de l'utilisateur AWS Support .

Je reçois un message d'erreur « Vérification supplémentaire requise »

Si le message d'erreur « Vérification supplémentaire requise » s'affiche, cela signifie que AWS Certificate Manager (ACM) a besoin d'informations supplémentaires pour traiter cette demande de certificat. Cela peut se produire en tant que mesure de protection contre la fraude, par exemple lorsque le domaine se classe dans les 1 000 meilleurs sites Web d'Alexa. Pour fournir ces informations, utilisez le Centre de support pour contacter AWS Support. Si vous n'avez pas de plan de support, publiez un nouveau fil de discussion dans le forum de discussion ACM.

> **Note**
>
> Vous ne pouvez pas demander de certificat pour des noms de domaine qui sont la propriété d'Amazon, par exemple ceux qui se terminent par amazonaws.com, cloudfront.net ou elasticbeanstalk.com.

Je reçois une erreur 404 sur l' CloudFront URL

Pour gérer le trafic, Amplify Hosting pointe vers une CloudFront URL via un enregistrement CNAME. Lors du processus de connexion d'une application à un domaine personnalisé, la console Amplify affiche l' CloudFrontURL de l'application. Toutefois, vous ne pouvez pas accéder directement à votre application à l'aide de cette CloudFront URL. Elle renvoie une erreur 404. Votre application est résolue uniquement à l'aide de l'URL de l'application Amplify (par exemple) ou de votre domaine personnalisé (par exemple`www.example.com`). `https://main.d5udybEXAMPLE.amplifyapp.com`

Amplify doit acheminer les demandes vers la branche déployée appropriée et utilise le nom d'hôte pour ce faire. Par exemple, vous pouvez configurer le domaine `www.example.com` qui pointe vers la branche principale d'une application, mais également configurer `dev.example.com` qui pointe vers la branche de développement de la même application. Par conséquent, vous devez visiter votre application en fonction de ses sous-domaines configurés afin qu'Amplify puisse acheminer les demandes en conséquence.

Je reçois des erreurs de certificat SSL ou HTTPS lorsque je visite mon domaine

Si des enregistrements DNS d'autorisation d'autorité de certification (CAA) sont configurés auprès de votre fournisseur DNS tiers, AWS Certificate Manager (ACM) risque de ne pas être en mesure de mettre à jour ou de réémettre les certificats intermédiaires pour votre certificat SSL de domaine personnalisé. Pour résoudre ce problème, vous devez ajouter un enregistrement CAA pour approuver au moins un des domaines d'autorité de certification d'Amazon. La procédure suivante décrit les étapes à suivre.

Pour ajouter un enregistrement CAA afin de faire confiance à une autorité de certification Amazon

1. Configurez un enregistrement CAA auprès de votre fournisseur de domaine pour faire confiance à au moins un des domaines d'autorité de certification d'Amazon. Pour plus d'informations sur la configuration de l'enregistrement CAA, consultez la section Problèmes d'autorisation de l'autorité de certification (CAA) dans le guide de AWS Certificate Manager l'utilisateur.
2. Utilisez l'une des méthodes suivantes pour mettre à jour votre certificat SSL :
   - Effectuez une mise à jour manuelle à l'aide de la console Amplify.

     > **Note**
     >
     > Cette méthode entraînera une interruption de service de votre domaine personnalisé.

     a. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
     b. Choisissez l'application à laquelle vous souhaitez ajouter un enregistrement CAA.
     c. Dans le volet de navigation, choisissez Paramètres de l'application, Gestion des domaines.
     d. Sur la page Gestion du domaine, supprimez le domaine personnalisé.
     e. Connectez à nouveau votre application au domaine personnalisé. Ce processus émet un nouveau certificat SSL et ses certificats intermédiaires peuvent désormais être gérés par ACM.

        Pour reconnecter votre application à votre domaine personnalisé, appliquez l'une des procédures suivantes correspondant au fournisseur de domaine que vous utilisez.

    - Ajouter un domaine personnalisé géré par Amazon Route 53.
    - Ajouter un domaine personnalisé géré par un fournisseur DNS tiers.
    - Ajoutez un domaine personnalisé géré par GoDaddy.
    - Ajouter un domaine personnalisé géré par Google Domains.
- Contactez-nous AWS Support pour que votre certificat SSL soit réémis.

Configuration des paramètres de compilation

Lorsque vous déployez une application avec Amplify Hosting, celle-ci détecte automatiquement le framework frontal et les paramètres de build associés en inspectant le `package.json` fichier dans votre référentiel. Vous disposez des options suivantes pour stocker les paramètres de compilation de votre application :

- Enregistrez les paramètres de build dans la console Amplify - La console Amplify détecte automatiquement les paramètres de build et les enregistre afin qu'ils soient accessibles via la console Amplify. Amplify applique ces paramètres à toutes vos branches, sauf si un `amplify.yml` fichier est stocké dans votre référentiel.
- Enregistrez les paramètres de compilation dans votre dépôt : téléchargez le `amplify.yml` fichier et ajoutez-le à la racine de votre dépôt.

Vous pouvez modifier les paramètres de compilation d'une application dans la console Amplify en choisissant Paramètres de l'application, Paramètres de génération. Les paramètres de génération sont appliqués à toutes les branches de votre application, à l'exception des branches dont un `amplify.yml` fichier est enregistré dans le référentiel.

> **Note**
>
> Les paramètres de compilation ne sont visibles dans le menu des paramètres de l'application de la console Amplify que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel git. Pour obtenir des instructions sur ce type de déploiement, voir Commencer avec le code existant.

Commandes et paramètres de spécification de construction

La spécification de construction YAML contient un ensemble de commandes de construction et de paramètres associés qu'Amplify utilise pour exécuter votre build. La liste suivante décrit ces paramètres et leur mode d'utilisation.

version

Le numéro de version d'Amplify YAML.

par Root

Le chemin dans le référentiel dans lequel réside cette application. Ignoré sauf si plusieurs applications sont définies.

env

Ajoutez des variables d'environnement à cette section. Vous pouvez également ajouter des variables d'environnement à l'aide de la console.

dorsal

Exécutez les commandes Amplify CLI pour provisionner un backend, mettre à jour des fonctions Lambda ou des schémas GraphQL dans le cadre d'un déploiement continu. Découvrez comment déployer un backend avec votre frontend.

frontend

Exécutez les commandes de construction du frontend.

test

Exécutez des commandes pendant une phase de test. Découvrez comment ajouter des tests à votre application.

phases de construction

Le frontend, le backend et le test comportent trois phases qui représentent les commandes exécutées au cours de chaque séquence de construction.

- PreBuild - Le script PreBuild s'exécute avant le début de la compilation proprement dite, mais après qu'Amplify ait installé les dépendances.
- génération : vos commandes de génération.
- PostBuild - Le script post-build s'exécute une fois la compilation terminée et Amplify a copié tous les artefacts nécessaires dans le répertoire de sortie.

chemin de construction

Le chemin à utiliser pour exécuter le build. Amplify utilise ce chemin pour localiser les artefacts de votre build. Si vous ne spécifiez pas de chemin, Amplify utilise la racine de l'application monorepo, par exemple. `apps/app`

artéfacts > répertoire de base

Le répertoire dans lequel se trouvent vos artefacts de build.

artéfacts > fichiers

Spécifiez les fichiers à partir de vos artefacts que vous souhaitez déployer. Entrez `**/*` pour inclure tous les fichiers.

cache

Le champ de cache de buildspec est utilisé pour mettre en cache les dépendances au moment de la création, telles que le dossier node_modules, et est automatiquement suggéré en fonction du gestionnaire de packages et du framework dans lesquels l'application du client est intégrée. Lors de la première génération, tous les chemins présents sont mis en cache, et lors des versions suivantes, nous regonflons le cache et utilisons ces dépendances mises en cache dans la mesure du possible pour accélérer le temps de construction.

L'exemple de spécification de construction suivant illustre la syntaxe YAML de base :

Syntaxe YAML des spécifications de construction

```
version: 1
env:
  variables:
    key: value
backend:
  phases:
    preBuild:
      commands:
        - *enter command*
    build:
      commands:
        - *enter command*
    postBuild:
        commands:
        - *enter command*
frontend:
  buildpath:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
```

```
        - npm run build
  artifacts:
    files:
        - location
        - location
    discard-paths: yes
    baseDirectory: location
  cache:
    paths:
        - path
        - path
test:
  phases:
    preTest:
      commands:
        - *enter command*
    test:
      commands:
        - *enter command*
    postTest:
      commands:
        - *enter command*
  artifacts:
    files:
        - location
        - location
    configFilePath: *location*
    baseDirectory: *location*
```

Paramètres de construction spécifiques à la branche

Vous pouvez utiliser des scripts shell bash pour définir des paramètres de build spécifiques à la branche. Par exemple, le script suivant utilise la variable d'environnement système $AWS_BRANCH pour exécuter un ensemble de commandes si le nom de la branche est main et un autre ensemble de commandes si le nom de la branche est dev.

```
frontend:
  phases:
    build:
      commands:
        - if [ "${AWS_BRANCH}" = "main" ]; then echo "main branch"; fi
```

```
        - if [ "${AWS_BRANCH}" = "dev" ]; then echo "dev branch"; fi
```

Navigation vers un sous-dossier

Pour monorepos, les utilisateurs veulent pouvoir accéder à un `cd` dossier pour exécuter le build. Une fois que vous avez exécuté la `cd` commande, elle s'applique à toutes les étapes de votre build, vous n'avez donc pas besoin de répéter la commande en plusieurs phases.

```
version: 1
env:
  variables:
    key: value
frontend:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
        - npm run build
```

Déploiement du backend avec le front-end

La `amplifyPush` commande est un script d'assistance qui vous aide dans les déploiements du backend. Les paramètres de build ci-dessous déterminent automatiquement l'environnement backend approprié à déployer pour la branche active.

```
version: 1
env:
  variables:
    key: value
backend:
  phases:
    build:
      commands:
        - amplifyPush --simple
```

Configuration du dossier de sortie

Les paramètres de génération suivants définissent le répertoire de sortie sur le dossier public.

```
frontend:
  phases:
    commands:
      build:
        - yarn run build
  artifacts:
    baseDirectory: public
```

Installation de packages dans le cadre d'une compilation

Vous pouvez utiliser les `yarn` commandes `npm` or pour installer des packages lors de la compilation.

```
frontend:
  phases:
    build:
      commands:
        - npm install -g <package>
        - <package> deploy
        - yarn run build
  artifacts:
    baseDirectory: public
```

Utilisation d'un registre npm privé

Vous pouvez ajouter des références à un registre privé dans vos paramètres de génération ou l'ajouter en tant que variable d'environnement.

```
build:
  phases:
    preBuild:
      commands:
        - npm config set <key> <value>
        - npm config set registry https://registry.npmjs.org
        - npm config set always-auth true
        - npm config set email hello@amplifyapp.com
```

```
        - yarn install
```

Installation de packages de système d'exploitation

L'image AL2023 d'Amplify exécute votre code avec un nom d'utilisateur non privilégié. `amplify` Amplify accorde à cet utilisateur les privilèges nécessaires pour exécuter les commandes du système d'exploitation à l'aide de la commande Linux. `sudo` Si vous souhaitez installer des packages de système d'exploitation pour les dépendances manquantes, vous pouvez utiliser des commandes telles que `yum` et `rpm` avec`sudo`.

L'exemple de section de construction suivant illustre la syntaxe d'installation d'un package de système d'exploitation à l'aide de la `sudo` commande.

```
build:
  phases:
    preBuild:
      commands:
        - sudo yum install -y <package>
```

Stockage des paires clé-valeur pour chaque build

`envCache`Fournit un stockage des valeurs clés au moment de la construction. Les valeurs stockées dans le ne `envCache` peuvent être modifiées que lors d'une construction et peuvent être réutilisées lors de la prochaine génération. À l'aide de`envCache`, nous pouvons stocker des informations sur l'environnement déployé et les mettre à la disposition du conteneur de construction lors de versions successives. Contrairement aux valeurs stockées dans le`envCache`, les modifications apportées aux variables d'environnement au cours d'une génération ne sont pas conservées dans les versions futures.

Exemple d'utilisation :

```
envCache --set <key> <value>
envCache --get <key>
```

Ignorer le build pour un commit

Pour ignorer une compilation automatique sur un commit particulier, incluez le texte [skip-cd] à la fin du message de validation.

Désactiver les builds automatiques

Vous pouvez configurer Amplify pour désactiver les builds automatiques à chaque validation de code. Pour le configurer, choisissez Paramètres de l'application, Général, puis faites défiler l'écran jusqu'à la section Branches qui répertorie les branches connectées. Sélectionnez une branche, puis choisissez Action, Désactiver la construction automatique. Toute autre validation ayant lieu dans cette branche ne déclenchera plus la création automatique de builds.

Activer ou désactiver la création et le déploiement du frontend basé sur les différences

Vous pouvez configurer Amplify pour utiliser des versions frontales basées sur les différences. Si cette option est activée, au début de chaque build, Amplify tente d'exécuter une différence sur votre `appRoot` dossier ou sur le `/src/` dossier par défaut. Si Amplify ne trouve aucune différence, il ignore les étapes de création, de test (si configuré) et de déploiement du frontend, et ne met pas à jour votre application hébergée.

Pour configurer la création et le déploiement d'un frontend basé sur Diff

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle configurer la création et le déploiement du frontend basé sur les différences.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Variables d'environnement.
4. Dans la section Variables d'environnement, sélectionnez Gérer les variables.
5. La procédure de configuration de la variable d'environnement varie selon que vous activez ou désactivez la création et le déploiement du frontend basé sur le diff.
   - Pour activer la création et le déploiement d'une interface basée sur les différences
     a. Dans la section Gérer les variables, sous Variable, entrez`AMPLIFY_DIFF_DEPLOY`.
     b. Pour le champ Valeur, saisissez `true`.
   - Pour désactiver la création et le déploiement du frontend basé sur les différences
     - Effectuez l'une des actions suivantes :
       - Dans la section Gérer les variables, recherchez`AMPLIFY_DIFF_DEPLOY`. Pour le champ Valeur, saisissez `false`.
       - Supprimez la variable d'`AMPLIFY_DIFF_DEPLOY`environnement.

6. Choisissez Enregistrer.

Vous pouvez éventuellement définir la variable d'`AMPLIFY_DIFF_DEPLOY_ROOT`environnement pour remplacer le chemin par défaut par un chemin relatif à la racine de votre dépôt, tel que. `dist`

Activer ou désactiver les versions de backend basées sur les différences

Vous pouvez configurer Amplify Hosting pour utiliser des versions de backend basées sur les différences à l'aide de la variable d'environnement. `AMPLIFY_DIFF_BACKEND` Lorsque vous activez les versions de backend basées sur le diff, au début de chaque build, Amplify tente d'exécuter un diff sur le `amplify` dossier de votre référentiel. Si Amplify ne trouve aucune différence, il ignore l'étape de création du backend et ne met pas à jour vos ressources backend. Si votre projet ne contient aucun `amplify` dossier dans votre référentiel, Amplify ignore la valeur de la `AMPLIFY_DIFF_BACKEND` variable d'environnement.

Si des commandes personnalisées sont actuellement spécifiées dans les paramètres de génération de votre phase de backend, les builds de backend conditionnels ne fonctionneront pas. Si vous souhaitez que ces commandes personnalisées s'exécutent, vous devez les déplacer vers la phase frontale de vos paramètres de génération dans le `amplify.yml` fichier de votre application.

Pour configurer des versions de backend basées sur les différences

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle configurer les versions de backend basées sur les différences.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Variables d'environnement.
4. Dans la section Variables d'environnement, sélectionnez Gérer les variables.
5. La procédure de configuration de la variable d'environnement varie selon que vous activez ou désactivez les versions de backend basées sur les différences.
   - Pour activer les builds de backend basés sur les différences
     a. Dans la section Gérer les variables, sous Variable, entrez`AMPLIFY_DIFF_BACKEND`.
     b. Pour le champ Valeur, saisissez `true`.
   - Pour désactiver les builds de backend basés sur les différences

- Effectuez l'une des actions suivantes :
  - Dans la section Gérer les variables, recherchez`AMPLIFY_DIFF_BACKEND`. Pour le champ Valeur, saisissez `false`.
  - Supprimez la variable d'`AMPLIFY_DIFF_BACKEND`environnement.

6. Choisissez Enregistrer.

Paramètres de construction de Monorepo

Lorsque vous stockez plusieurs projets ou microservices dans un seul référentiel, cela s'appelle un monorepo. Vous pouvez utiliser Amplify Hosting pour déployer des applications dans un monorepo sans créer plusieurs configurations de build ou de branche.

Amplify prend en charge les applications en monorepos génériques ainsi que les applications en monorepos créées à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo. Lorsque vous déployez votre application, Amplify détecte automatiquement l'outil de génération monorepo que vous utilisez. Amplify applique automatiquement les paramètres de génération pour les applications dans un espace de travail npm, un espace de travail Yarn ou Nx. Les applications Turborepo et pnpm nécessitent une configuration supplémentaire. Pour plus d'informations, consultez Configuration des applications Turborepo et pnpm monorepo.

Vous pouvez enregistrer les paramètres de compilation d'un monorepo dans la console Amplify ou vous pouvez télécharger le `amplify.yml` fichier et l'ajouter à la racine de votre référentiel. Amplify applique les paramètres enregistrés dans la console à toutes vos branches sauf s'il trouve un `amplify.yml` fichier dans votre référentiel. Lorsqu'un `amplify.yml` fichier est présent, ses paramètres remplacent tous les paramètres de compilation enregistrés dans la console Amplify.

Syntaxe YAML de la spécification de construction Monorepo

La syntaxe YAML pour une spécification de build monorepo est différente de la syntaxe YAML pour un dépôt contenant une seule application. Pour un monorepo, vous déclarez chaque projet dans une liste d'applications. Vous devez fournir la `appRoot` clé supplémentaire suivante pour chaque application que vous déclarez dans la spécification de construction de votre monorepo :

par Root

Racine, au sein du référentiel, dans laquelle l'application démarre. Cette clé doit exister et avoir la même valeur que la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement. Pour obtenir

des instructions sur la définition de cette variable d'environnement, consultezConfiguration de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT.

L'exemple de spécification de construction de monorepo suivant montre comment déclarer plusieurs applications Amplify dans le même dépôt. Les deux applications`react-app`, et `angular-app` sont déclarées dans la `applications` liste. La `appRoot` clé de chaque application indique que l'application se trouve dans le dossier `apps` racine du dépôt.

L'`buildpath`attribut est défini pour exécuter et créer l'application `/` à partir de la racine du projet monorepo.

Syntaxe YAML de la spécification de construction Monorepo

```
version: 1
applications:
  - appRoot: apps/react-app
    env:
      variables:
        key: value
    backend:
      phases:
        preBuild:
          commands:
            - *enter command*
        build:
          commands:
            - *enter command*
        postBuild:
            commands:
            - *enter command*
    frontend:
      buildPath: / # Run install and build from the monorepo project root
      phases:
        preBuild:
          commands:
            - *enter command*
            - *enter command*
        build:
          commands:
            - *enter command*
      artifacts:
        files:
```

```
            - location
            - location
        discard-paths: yes
        baseDirectory: location
      cache:
        paths:
            - path
            - path
    test:
      phases:
        preTest:
          commands:
            - *enter command*
        test:
          commands:
            - *enter command*
        postTest:
          commands:
            - *enter command*
      artifacts:
        files:
            - location
            - location
        configFilePath: *location*
        baseDirectory: *location*
  - appRoot: apps/angular-app
    env:
      variables:
        key: value
    backend:
      phases:
        preBuild:
          commands:
            - *enter command*
        build:
          commands:
            - *enter command*
        postBuild:
            commands:
            - *enter command*
    frontend:
      phases:
        preBuild:
          commands:
```

```
            - *enter command*
            - *enter command*
        build:
          commands:
            - *enter command*
      artifacts:
        files:
            - location
            - location
        discard-paths: yes
        baseDirectory: location
      cache:
        paths:
            - path
            - path
    test:
      phases:
        preTest:
          commands:
            - *enter command*
        test:
          commands:
            - *enter command*
        postTest:
          commands:
            - *enter command*
      artifacts:
        files:
            - location
            - location
        configFilePath: *location*
        baseDirectory: *location*
```

Configuration de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT

Lorsque vous déployez une application stockée dans un monorepo, la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement de l'application doit avoir la même valeur que le chemin de la racine de l'application, par rapport à la racine de votre dépôt. Par exemple, un monorepo nommé `ExampleMonorepo` avec un dossier racine nommé`apps`, qui contient, `app1``app2`, et `app3` possède la structure de répertoires suivante :

```
ExampleMonorepo
  apps
    app1
    app2
    app3
```

Dans cet exemple, la valeur de la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement for `app1` est`apps/app1`.

Lorsque vous déployez une application monorepo à l'aide de la console Amplify, celle-ci définit automatiquement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement en utilisant la valeur que vous spécifiez pour le chemin d'accès à la racine de l'application. Toutefois, si votre application monorepo existe déjà dans Amplify ou est déployée à l'aide de AWS CloudFormation, vous devez définir manuellement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement dans la section Variables d'environnement de la console Amplify.

Définition automatique de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT lors du déploiement

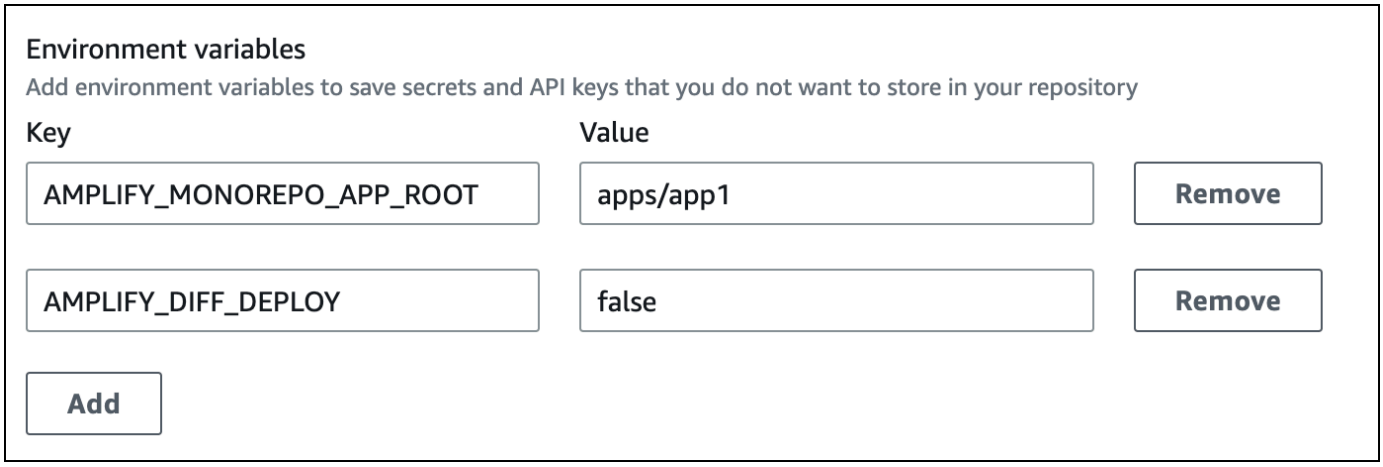
Les instructions suivantes montrent comment déployer une application monorepo avec la console Amplify. Amplify définit automatiquement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement à l'aide du dossier racine de l'application que vous spécifiez dans la console.

Pour déployer une application monorepo avec la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez Nouvelle application, Héberger une application Web dans le coin supérieur droit.
3. Sur la page Héberger votre application Web, choisissez votre fournisseur Git, puis choisissez Continuer.
4. Sur la page Ajouter une branche de référentiel, procédez comme suit :
    a. Choisissez le nom de votre dépôt dans la liste des référentiels récemment mis à jour.
    b. Pour Branch, choisissez le nom de la branche à utiliser.
    c. Sélectionnez Connecter un monorepo ? Choisissez un dossier.
    d. Entrez le chemin d'accès à votre application dans votre monorepo, par exemple,. **`apps/app1`**

e. Choisissez Suivant.

5. Sur la page des paramètres de génération, vous pouvez utiliser les paramètres par défaut ou personnaliser les paramètres de génération de votre application. Dans l'exemple de capture d'écran suivant, dans la section Variables d'environnement, Amplify est défini sur`apps/app1`, `AMPLIFY_MONOREPO_APP_ROOT` en utilisant le chemin que vous avez spécifié à l'étape 4d.



6. Choisissez Suivant.
7. Sur la page Révision, choisissez Enregistrer et déployer.

Définition de la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT pour une application existante

Utilisez les instructions suivantes pour définir manuellement la variable d'`AMPLIFY_MONOREPO_APP_ROOT`environnement pour une application déjà déployée sur Amplify ou créée à l'aide de. CloudFormation

Pour définir la variable d'environnement AMPLIFY_MONOREPO_APP_ROOT pour une application existante

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez le nom de l'application pour laquelle définir la variable d'environnement.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis choisissez Variables d'environnement.
4. Sur la page Variables d'environnement, sélectionnez Gérer les variables.
5. Dans la section Gérer les variables, procédez comme suit :

   a. Choisissez Ajouter une variable.

b. Pour Variable, entrez la clé`AMPLIFY_MONOREPO_APP_ROOT`.

c. Pour Value, entrez le chemin d'accès à l'application, par exemple**`apps/app1`**.

d. Pour Branch, Amplify applique par défaut la variable d'environnement à toutes les branches.

6. Choisissez Enregistrer.

Configuration des applications Turborepo et pnpm monorepo

Les outils de génération monorepo de Turborepo et pnpm workspace obtiennent des informations de configuration à partir de fichiers. `.npmrc` Lorsque vous déployez une application monorepo créée avec l'un de ces outils, vous devez avoir un `.npmrc` fichier dans le répertoire racine de votre projet.

Dans le `.npmrc` fichier, définissez l'éditeur de liens pour l'installation des packages Node sur`hoisted`. Vous pouvez copier la ligne suivante dans votre fichier.

```
node-linker=hoisted
```

Pour plus d'informations sur *`.npmrc`* les fichiers et les paramètres, consultez pnpm .npmrc dans la documentation de pnpm.

Pnpm n'est pas inclus dans le conteneur de build par défaut d'Amplify. Pour les applications pnpm workspace et Turborepo, vous devez ajouter une commande pour installer pnpm dans la `preBuild` phase des paramètres de compilation de votre application.

L'exemple d'extrait suivant d'une spécification de construction montre une `preBuild` phase avec une commande pour installer pnpm.

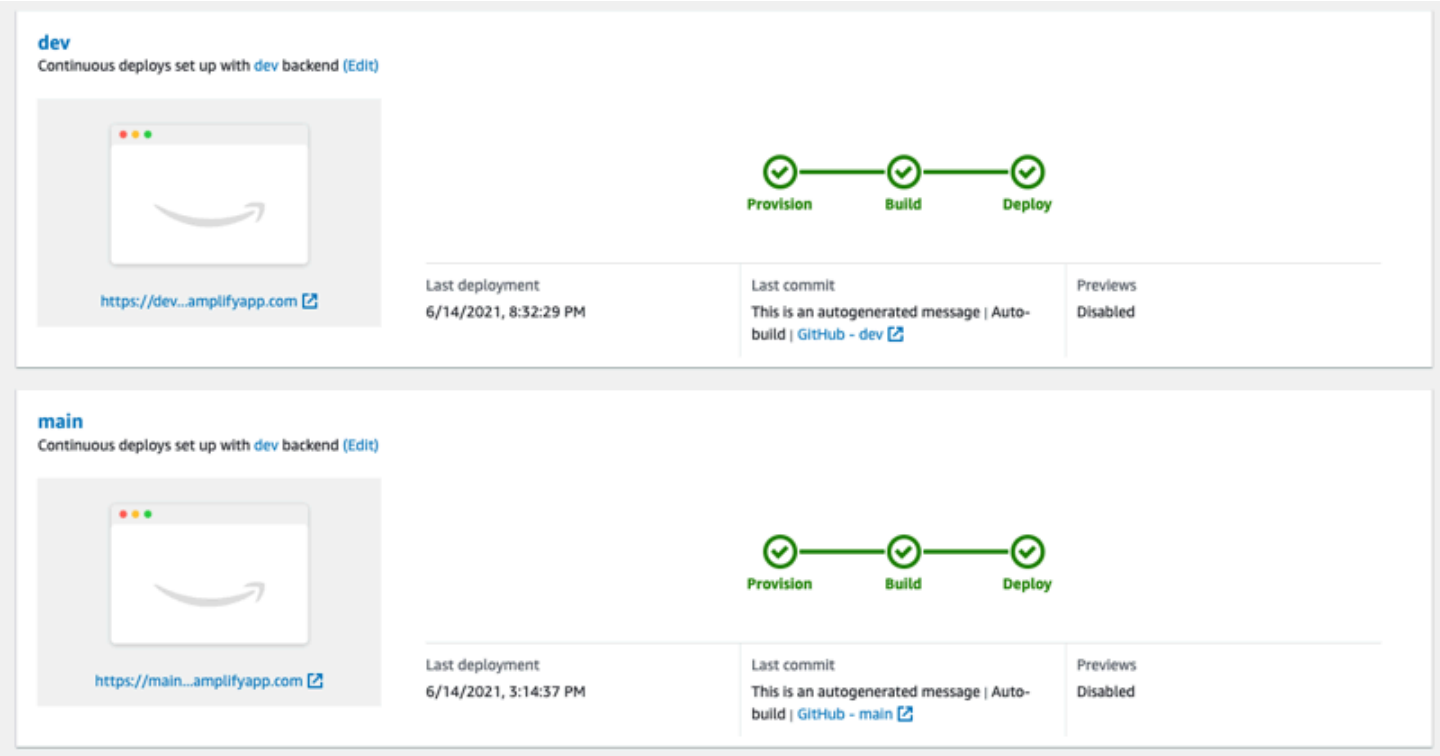
```
version: 1
applications:
  - frontend:
      phases:
        preBuild:
          commands:
            - npm install -g pnpm
```

Flux de travail de déploiements de branches de fonctionnalités et flux de travail d'équipe

Amplify Hosting est conçu pour fonctionner avec les branches de fonctionnalités et les GitFlow flux de travail. Amplify utilise les branches Git pour créer de nouveaux déploiements chaque fois qu'un développeur connecte une nouvelle branche dans son référentiel. Après avoir connecté votre première branche, vous pouvez créer un autre déploiement en ajoutant une branche comme suit :

1. Sur la page de la liste des branches, choisissez Connecter une branche.
2. Choisissez une branche dans votre référentiel.
3. Enregistrez, puis déployez votre application.

Votre application dispose désormais de deux déploiements disponibles sur https://main.appid.amplifyapp.com et https://dev.appid.amplifyapp.com. Cela peut varier par rapport àteam-to-team, mais généralement, la branche principale publie le code de publication et constitue votre branche de production. La branche de développement sert de branche d'intégration pour tester les nouvelles fonctionnalités. Cela permet aux bêta-testeurs de tester des fonctionnalités inédites lors du déploiement de la branche de développement, sans affecter les utilisateurs finaux de production participant au déploiement de la branche principale.



Rubriques

- Flux de travail d'équipe avec les environnements principaux d'Amplify
- Déploiements de branches de fonctionnalités basés sur des modèles
- Génération automatique de la configuration Amplify au moment de la construction
- Configurations conditionnelles du backend
- Utilisez les backends Amplify dans toutes les applications

Flux de travail d'équipe avec les environnements principaux d'Amplify

Le déploiement d'une branche de fonctionnalités se compose d'un frontend et d'un environnement dorsal optionnel. Le frontend est créé et déployé sur un réseau mondial de diffusion de contenu (CDN), tandis que le backend est déployé par Amplify Studio ou l'interface de ligne de commande Amplify sur. AWS Pour plus d'informations sur ce scénario de déploiement, consultezDémarrez avec les déploiements continus Fullstack.

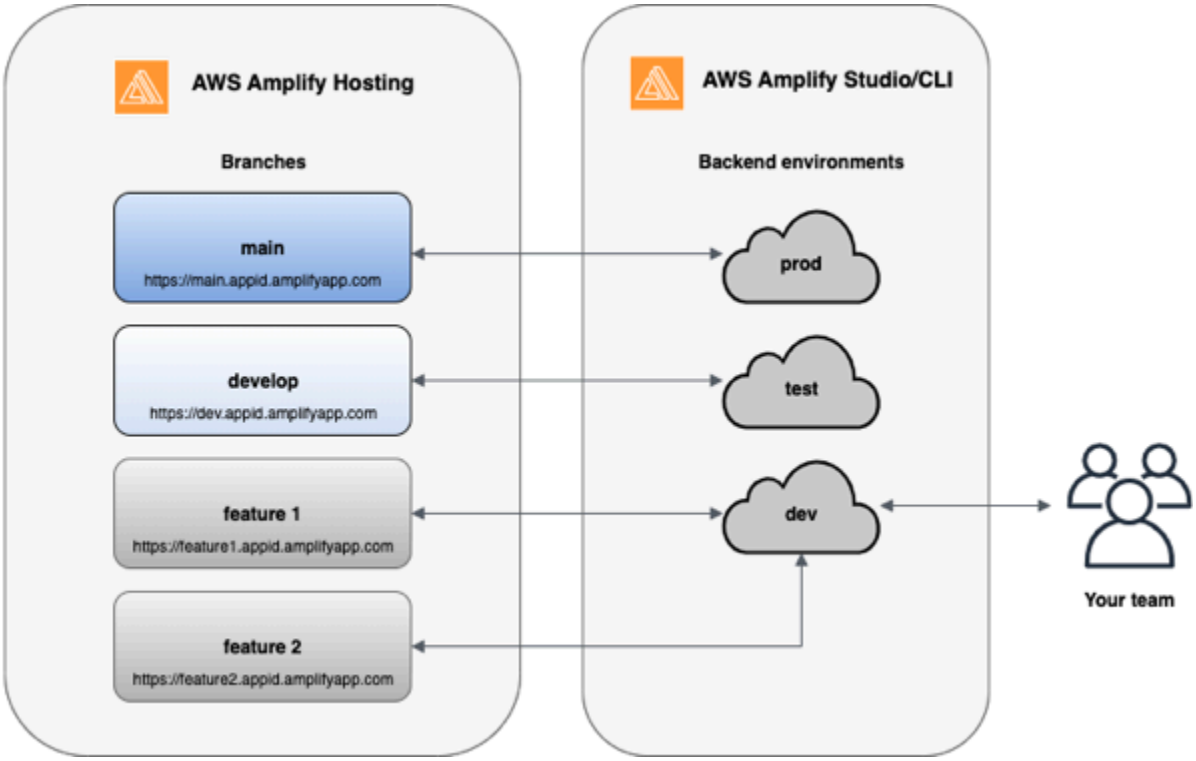
> **Note**
>
> Vous pouvez facilement réutiliser les environnements principaux d'Amplify dans vos applications Amplify. Pour plus d'informations, veuillez consulter Utilisez les backends Amplify dans toutes les applications.

Amplify Hosting déploie en continu des ressources dorsales telles que les API GraphQL et les fonctions Lambda avec vos déploiements de branches de fonctionnalités. Vous pouvez utiliser les modèles de branchement suivants pour déployer votre backend et votre frontend avec Amplify Hosting.

Rubriques

Flux de travail de branche de fonctionnalités

- Créez des environnements principaux de production, de test et de développement avec Amplify Studio ou l'interface de ligne de commande Amplify.
- Mappez le backend prod à la branche principale.
- Mappez le backend de test à la branche de développement.
- Les membres de l'équipe peuvent utiliser l'environnement principal de développement pour tester des branches de fonctionnalités individuelles.



1. Installez l'interface de ligne de commande Amplify pour lancer un nouveau projet Amplify.

```
npm install -g @aws-amplify/cli
```

2. Créez un environnement backend prod pour votre projet. Si vous n'avez pas de projet, créez-en un à l'aide d'outils d'amorçage tels que Gatsby create-react-app ou Gatsby.

```
create-react-app next-unicorn
cd next-unicorn
amplify init
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: prod
...
amplify push
```

3. Ajoutez des environnements backen de test et de développement.

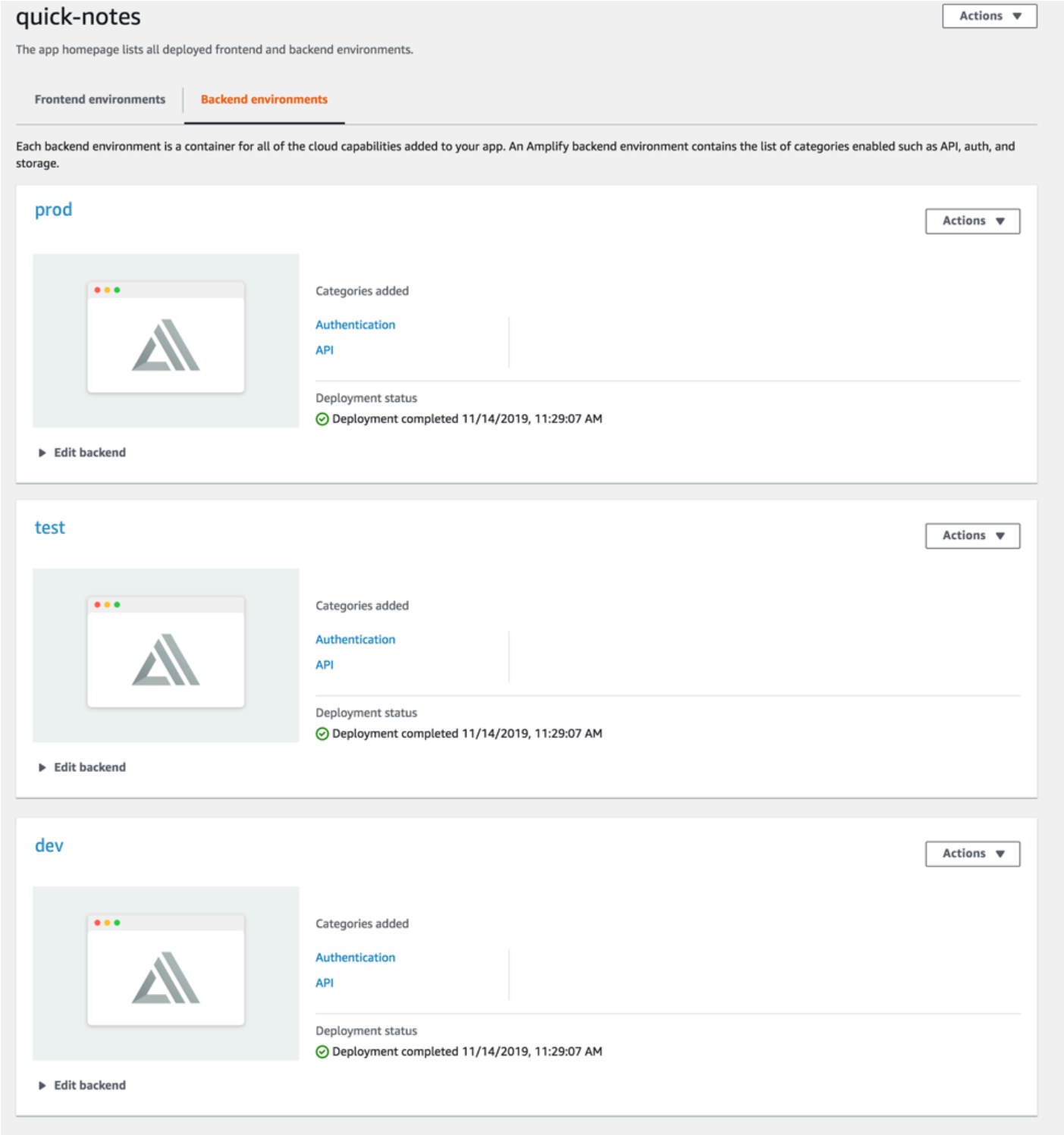
```
amplify env add
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: test
...
amplify push
```

```
amplify env add
 ? Do you want to use an existing environment? (Y/n): n
 ? Enter a name for the environment: dev
...
amplify push
```

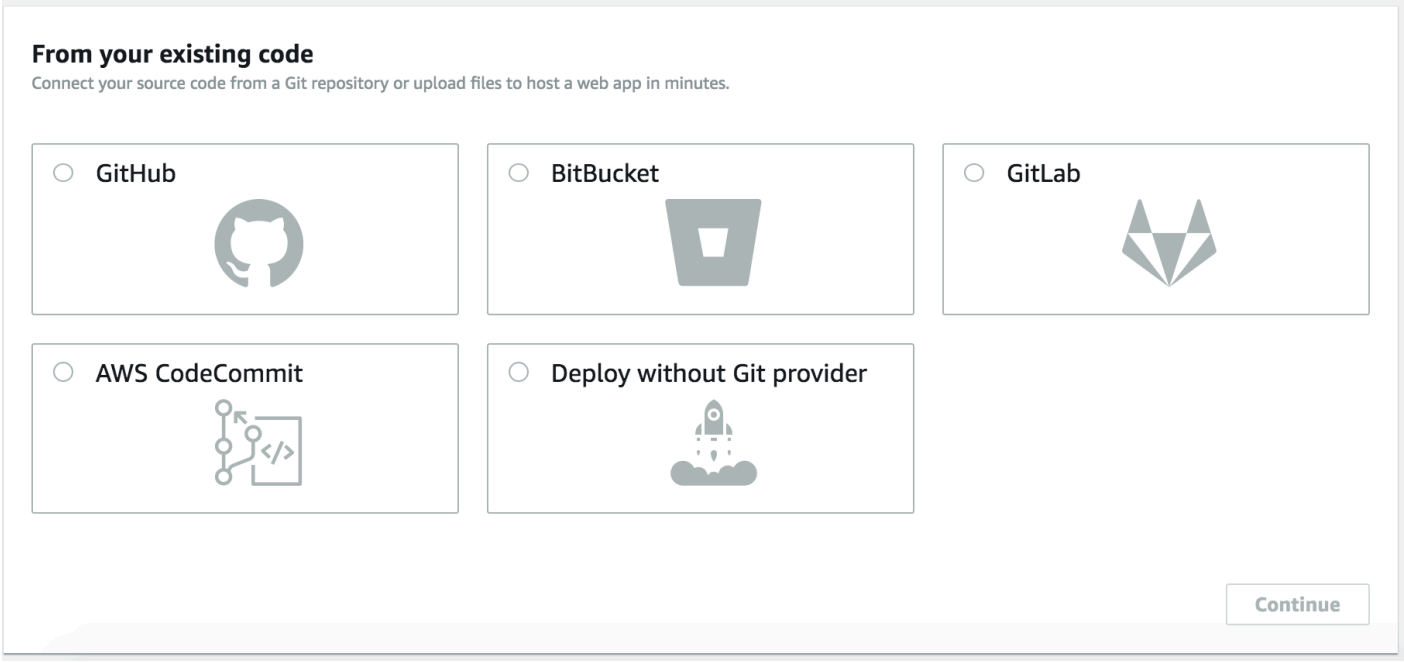
4. Envoyez le code vers le dépôt Git de votre choix (dans cet exemple, nous supposerons que vous avez accédé au répertoire principal).

```
git commit -am 'Added dev, test, and prod environments'
git push origin main
```

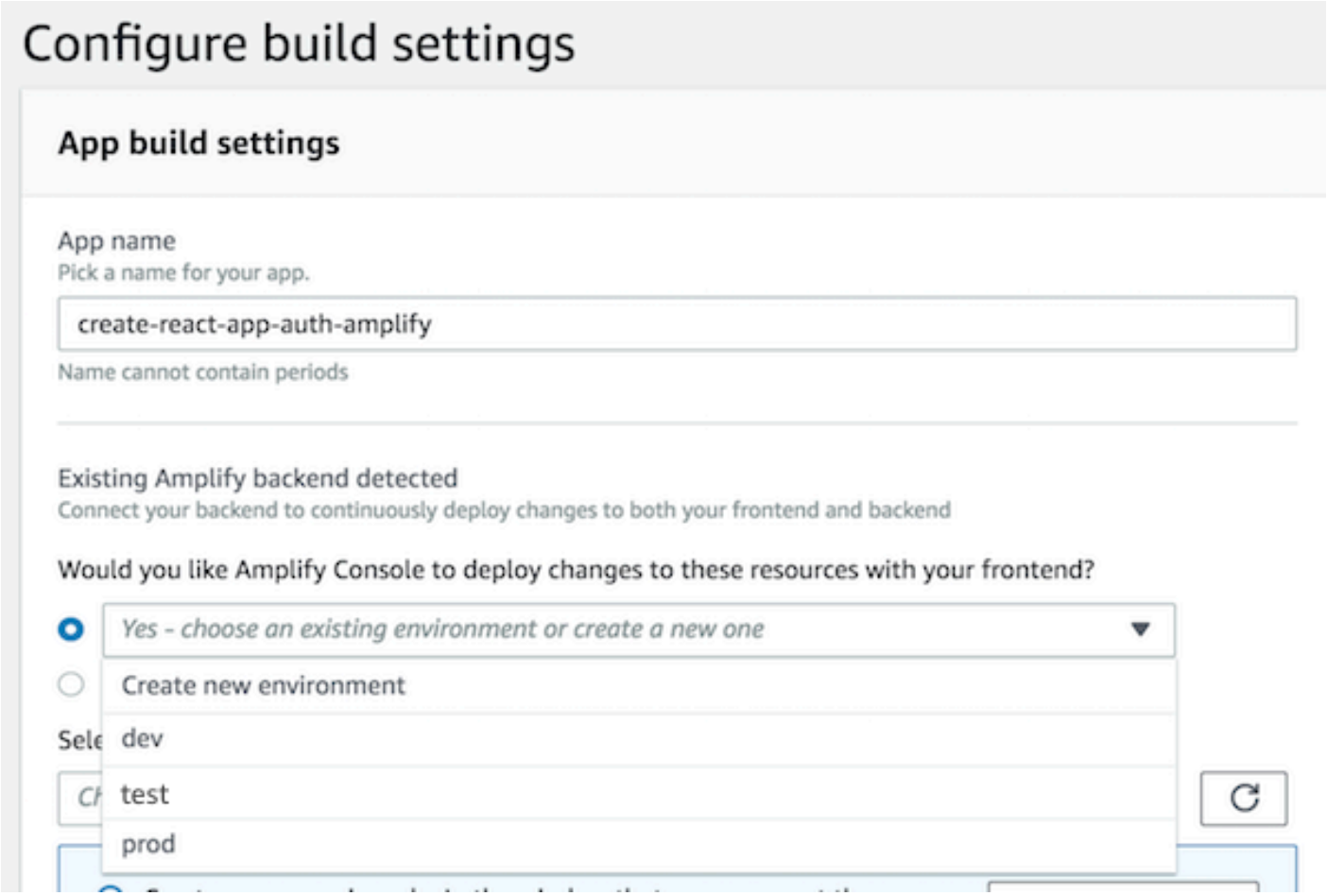
5. Accédez à Amplify dans le AWS Management Console pour voir votre environnement principal actuel. Naviguez d'un niveau vers le haut à partir du fil de navigation pour afficher la liste de tous les environnements principaux créés dans l'onglet Environnements principaux.



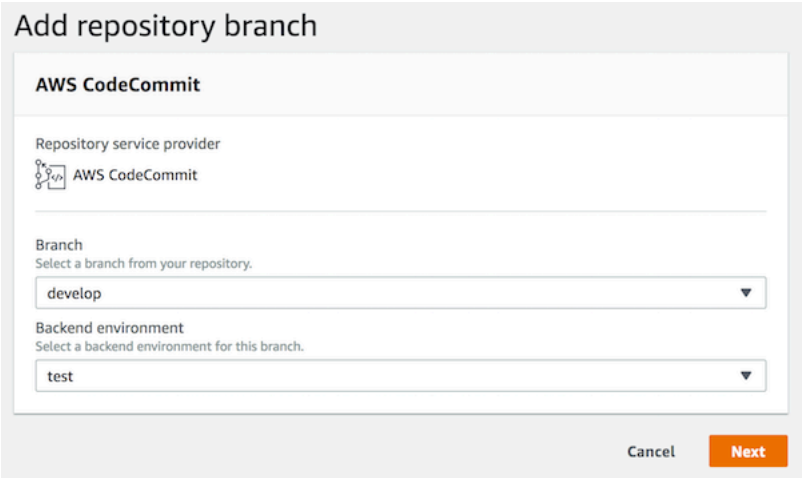
6. Passez à l'onglet Environnements frontaux et connectez votre fournisseur de référentiel et votre branche principale.



7. Dans l'écran des paramètres de génération, sélectionnez un environnement principal existant pour configurer un déploiement continu avec la branche principale. Choisissez prod dans la liste déroulante et attribuez le rôle de service à Amplify. Choisissez Save and deploy (Enregistrer et déployer). Une fois la compilation terminée, vous obtiendrez un déploiement de branche principale disponible à l'adresse https://main.appid.amplifyapp.com.



8. Connectez la branche de développement dans Amplify (en supposant que la branche de développement et la branche principale sont identiques à ce stade). Choisissez l'environnement backend test.



9. Amplify est désormais configuré. Vous pouvez commencer à travailler sur les nouvelles fonctionnalités d'une branche. Pour ajouter une fonctionnalité backend, utilisez l'environnement backend dev à partir de votre poste de travail local.

```
git checkout -b newinternet
amplify env checkout dev
amplify add api
...
amplify push
```

10. Une fois que vous avez fini de travailler sur cette fonctionnalité, validez le code et créez une demande d'extraction afin d'effectuer un examen en interne.

```
git commit -am 'Decentralized internet v0.1'
git push origin newinternet
```

11. Pour prévisualiser à quoi ressembleront les modifications, accédez à la console Amplify et connectez votre branche de fonctionnalités. Remarque : Si vous l'avez AWS CLI installé sur votre système (et non sur l'interface de ligne de commande Amplify), vous pouvez connecter une branche directement depuis votre terminal. Pour localiser l'ID de votre application, accédez à App Settings (Paramètres de l'application) > General (Général) > AppARN : arn:aws:amplify:<région>:<région>:apps/<IDapp>.

```
aws amplify create-branch --app-id <appid> --branch-name <branchname>
aws amplify start-job --app-id <appid> --branch-name <branchname> --job-type RELEASE
```

12. La fonctionnalité sera accessible à l'adresse https://newinternet.appid.amplifyapp.com. Partagez-la avec vos collègues à votre convenance. Si tout semble correct, fusionnez la branche PR avec la branche de développement.

```
git checkout develop
git merge newinternet
git push
```

13. Cela lancera une version qui mettra à jour le backend ainsi que le frontend dans Amplify avec un déploiement de branche sur https://dev.appid.amplifyapp.com. Vous pouvez partager ce lien avec les intervenants internes afin qu'ils puissent examiner cette nouvelle fonctionnalité.

14. Supprimez votre branche de fonctionnalités de Git, Amplify et supprimez l'environnement principal du cloud (vous pouvez toujours en créer une nouvelle en exécutant « amplify env checkout prod » et « amplify env add »).

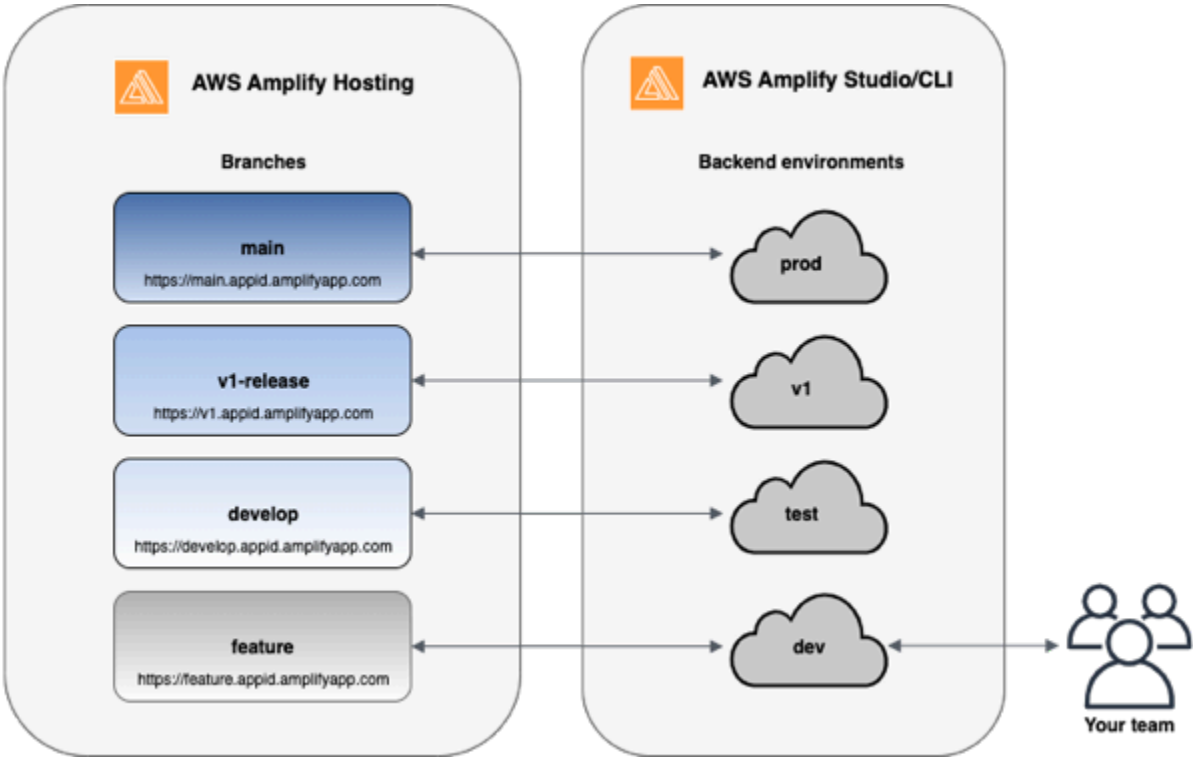
```
git push origin --delete newinternet
aws amplify delete-branch --app-id <appid> --branch-name <branchname>
```

```
amplify env remove dev
```

Flux de travail dans GitFlow

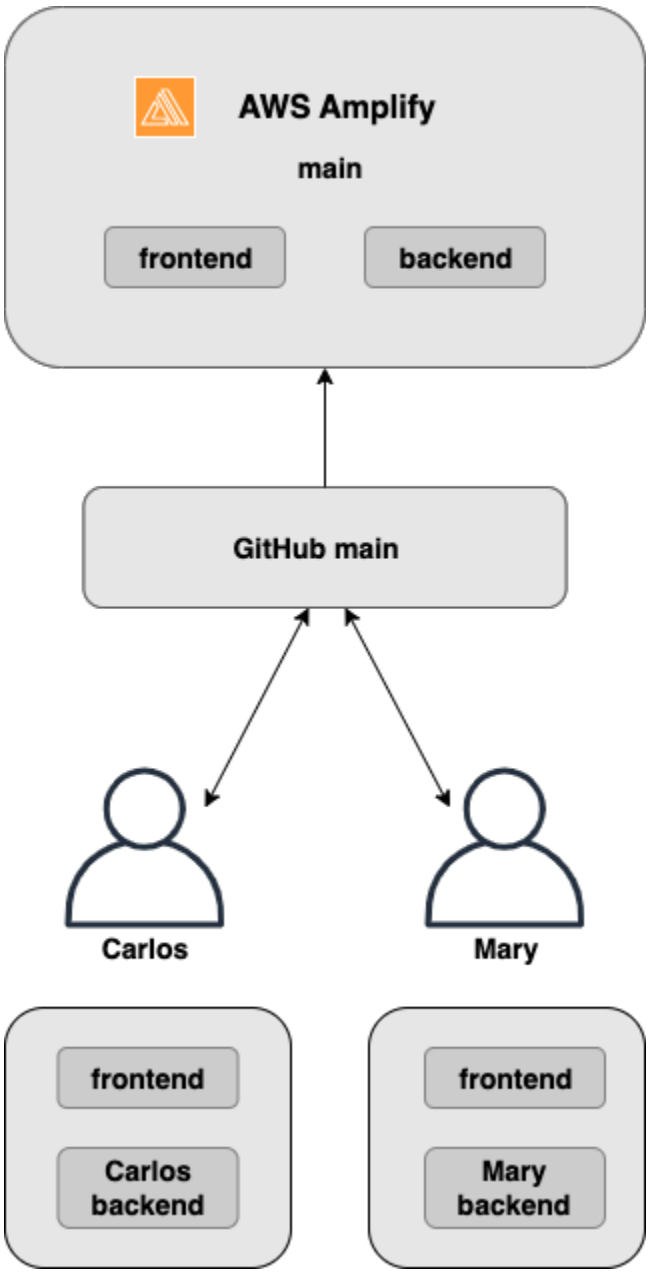
GitFlowutilise deux branches pour enregistrer l'historique du projet. La branche principale suit uniquement le code de publication, et la branche de développement est utilisée comme branche d'intégration pour les nouvelles fonctionnalités. GitFlowsimplifie le développement parallèle en isolant les nouveaux développements des travaux achevés. Le nouveau développement (comme les fonctionnalités et les correctifs de bogues non urgents) s'effectue dans les branches de fonctionnalités. Lorsque le développeur juge que le code peut être publié, la branche de fonctionnalités est fusionnée dans la branche de développement pour l'intégration. Les seules validations apportées à la branche principale sont les fusions à partir des branches des versions et des branches de correctifs (pour corriger des bogues urgents).

Le schéma ci-dessous montre une configuration recommandée avecGitFlow. Vous pouvez suivre le même processus, comme décrit dans la section ci-dessus sur les flux de travail des branches de fonctionnalités.



Un sandbox pour chaque développeur

- Chaque développeur d'une équipe crée un environnement de test (sandbox) dans le cloud qui est distinct de son ordinateur local. Cela permet aux développeurs de travailler indépendamment les uns des autres sans annuler les modifications apportées par les autres membres de l'équipe.
- Chaque branche d'Amplify possède son propre backend. Cela garantit que l'Amplify utilise le référentiel Git comme source unique de vérité à partir de laquelle déployer les modifications, plutôt que de compter sur les développeurs de l'équipe pour mettre manuellement leur backend ou leur front-end en production depuis leurs ordinateurs locaux.



1. Installez l'interface de ligne de commande Amplify pour lancer un nouveau projet Amplify.

```
npm install -g @aws-amplify/cli
```

2. Initialisez un environnement de backend Mary pour votre projet. Si vous n'avez pas de projet, créez-en un à l'aide d'outils d'amorçage tels que Gatsby create-react-app ou Gatsby.

```
cd next-unicorn
amplify init
 ? Do you want to use an existing environment? (Y/n): n
```

```
 ? Enter a name for the environment: mary
...
amplify push
```

3. Envoyez le code vers le dépôt Git de votre choix (dans cet exemple, nous supposerons que vous l'avez poussé vers main).

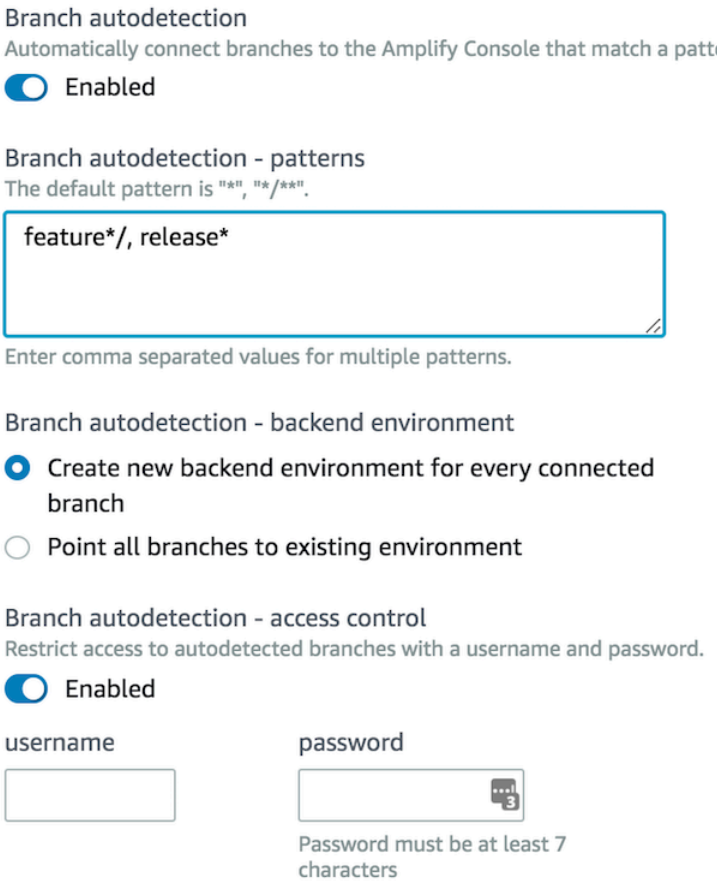
```
git commit -am 'Added mary sandbox'
git push origin main
```

4. Connectez votre dépôt > principal à Amplify.
5. La console Amplify détectera les environnements principaux créés par l'interface de ligne de commande Amplify. Choisissez Créer un nouvel environnement dans la liste déroulante et accordez le rôle de service à Amplify. Choisissez Save and deploy (Enregistrer et déployer). Une fois la compilation terminée, vous obtiendrez un déploiement de branche principale disponible sur https://main.appid.amplifyapp.com avec un nouvel environnement principal lié à la branche.
6. Connectez la branche de développement dans Amplify (en supposant que la branche de développement et la branche principale sont identiques à ce stade) et choisissez Créer un nouvel environnement. Une fois la création du build terminée, vous obtenez un déploiement de branche de développement sous https://develop.appid.amplifyapp.com avec un nouvel environnement backend relié à cette branche.

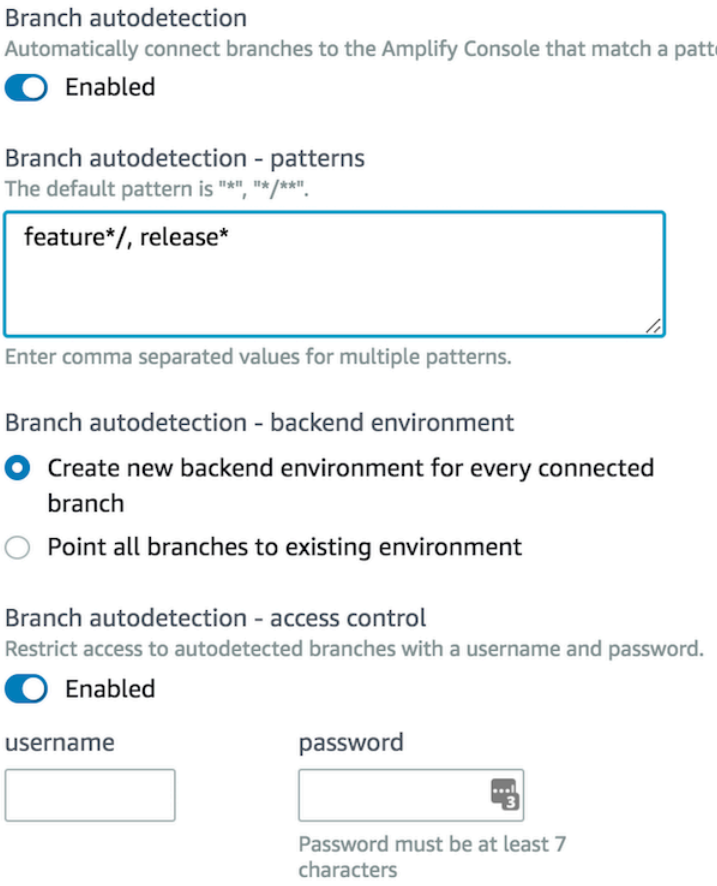
Déploiements de branches de fonctionnalités basés sur des modèles

Les déploiements de branches basés sur des modèles vous permettent de déployer automatiquement des branches qui correspondent à un modèle spécifique pour Amplify. Les équipes produit qui utilisent des branches de fonctionnalités ou des GitFlow flux de travail pour leurs versions peuvent désormais définir des modèles tels que « release** » pour déployer automatiquement des branches Git commençant par « release » vers une URL partageable. Cet article de blog décrit l'utilisation de cette fonctionnalité avec différents flux de travail d'équipe.

1. Choisissez App Settings > General > Edit (Paramètres de l'application > Général > Modifier).
2. Activez le commutateur de détection automatique de branches.



1. Définissez des modèles pour déployer automatiquement les branches.
   - **`*`**— Déploie toutes les branches de votre référentiel.
   - **`release*`**'— Déploie toutes les branches qui commencent par le mot « release ».
   - **`release*/`**— Déploie toutes les branches qui correspondent à un modèle « release /».
   - Spécifiez plusieurs modèles dans une liste séparée par des virgules. Par exemple, `release*, feature*`.
2. Pour configurer la protection automatique par mot de passe pour toutes les branches créées automatiquement, localisez le paramètre Branch autodetection - access control (Détection automatique de branche - Contrôle d'accès) et activez-le.
3. Pour les applications développées avec un backend Amplify, vous pouvez choisir de créer un autre environnement ou de renvoyer toutes les branches vers un environnement backend existant.



Déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé

Vous pouvez utiliser des déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé Amazon Route 53.

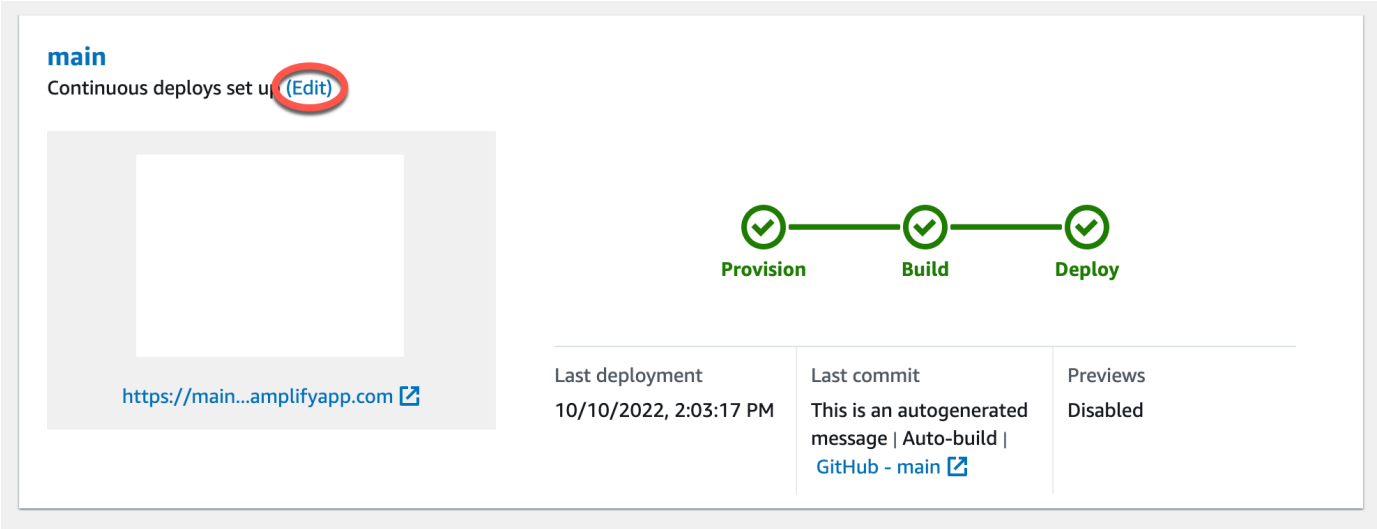
- Pour obtenir des instructions sur la configuration de déploiements de branches de fonctionnalités basés sur des modèles, voir Configurer des sous-domaines automatiques pour un domaine personnalisé Amazon Route 53
- Pour obtenir des instructions sur la connexion d'une application Amplify à un domaine personnalisé géré dans Route 53, voir Ajouter un domaine personnalisé géré par Amazon Route 53
- Pour plus d'informations sur l'utilisation de Route 53, consultez Qu'est-ce qu'Amazon Route 53 ?

Génération automatique de la configuration Amplify au moment de la construction

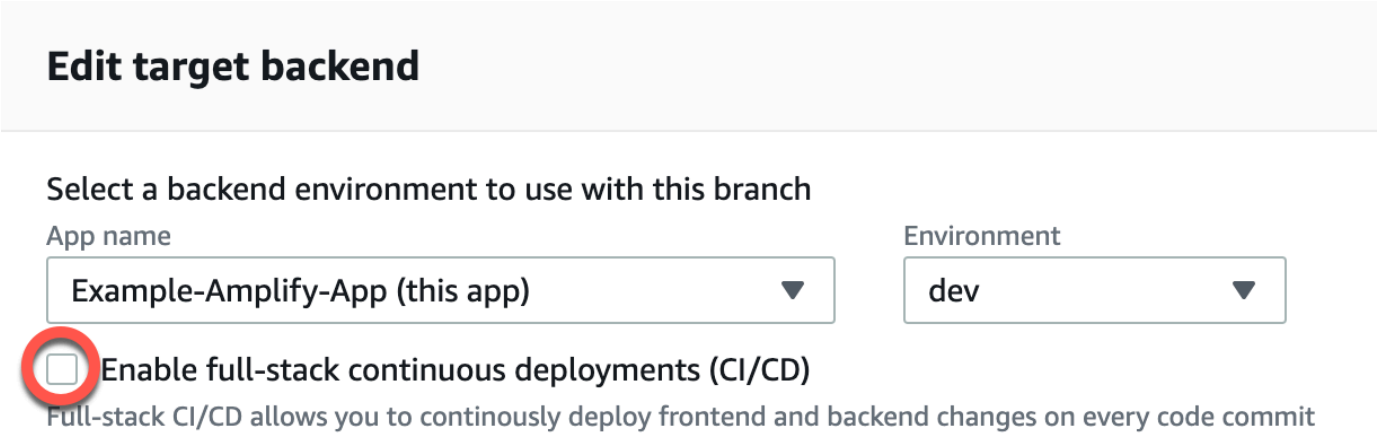
Amplify prend en charge la génération automatique du fichier de configuration Amplify au moment de la construction. `aws-exports.js` En désactivant les déploiements CI/CD complets, vous permettez à votre application de générer automatiquement le `aws-exports.js` fichier et vous vous assurez qu'aucune mise à jour n'est apportée à votre backend au moment de la création.

Pour générer automatiquement au **`aws-exports.js`** moment de la construction

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à modifier.
3. Choisissez l'onglet Environnements d'hébergement.
4. Localisez la branche à modifier et choisissez Modifier.



5. Sur la page Modifier le backend cible, décochez la case Activer les déploiements continus complets (CI/CD) pour désactiver le CI/CD full-stack pour ce backend.



6. Sélectionnez un rôle de service existant pour accorder à Amplify les autorisations nécessaires pour apporter des modifications au backend de votre application. Si vous devez créer un rôle de service, choisissez Créer un nouveau rôle. Pour plus d'informations sur la création d'un rôle de service, consultez la section Ajouter un rôle de service.
7. Choisissez Save (Enregistrer). Amplify appliquera ces modifications la prochaine fois que vous créerez l'application.

Configurations conditionnelles du backend

Amplify prend en charge les versions conditionnelles du backend sur toutes les branches d'une application. Pour configurer des versions conditionnelles du backend, définissez la variable d'`AMPLIFY_DIFF_BACKEND`environnement sur`true`. L'activation des versions conditionnelles du backend permettra d'accélérer les versions où les modifications ne sont apportées qu'au frontend.

Lorsque vous activez les versions de backend basées sur des différences, au début de chaque génération, Amplify tente d'exécuter un diff sur le `amplify` dossier de votre référentiel. Si Amplify ne trouve aucune différence, il ignore l'étape de création du backend et ne met pas à jour les ressources de votre backend. Si votre projet ne contient aucun `amplify` dossier dans votre référentiel, Amplify ignore la valeur de la variable d'`AMPLIFY_DIFF_BACKEND`environnement. Pour obtenir des instructions sur la définition de la variable d'`AMPLIFY_DIFF_BACKEND`environnement, reportez-vous à la sectionActiver ou désactiver les versions de backend basées sur les différences.

Si des commandes personnalisées sont actuellement spécifiées dans les paramètres de génération de votre phase de backend, les versions conditionnelles du backend ne fonctionneront pas. Si vous souhaitez que ces commandes personnalisées s'exécutent, vous devez les déplacer vers la phase frontale de vos paramètres de génération dans le `amplify.yml` fichier de votre

application. Pour plus d'informations sur la mise à jour du `amplify.yml` fichier, reportez-vous à la sectionCommandes et paramètres de spécification de construction.

Utilisez les backends Amplify dans toutes les applications

Amplify vous permet de réutiliser facilement les environnements principaux existants pour toutes vos applications dans une région donnée. Vous pouvez le faire lorsque vous créez une nouvelle application, connectez une nouvelle branche à une application existante ou mettez à jour une interface existante pour qu'elle pointe vers un autre environnement principal.

Réutilisez les backends lors de la création d'une nouvelle application

Pour réutiliser un backend lors de la création d'une nouvelle application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Pour créer un nouveau backend à utiliser pour cet exemple, procédez comme suit :
    a. Dans le volet de navigation, choisissez Toutes les applications.
    b. Choisissez Nouvelle application, puis Créez une application.
    c. Entrez un nom pour votre application, par exemple**`Example-Amplify-App`**.
    d. Choisissez Confirmer le déploiement.
3. Pour connecter un frontend à votre nouveau backend, choisissez l'onglet Environnements d'hébergement.
4. Choisissez votre fournisseur Git, puis sélectionnez Connect branch.
5. Sur la page Ajouter une branche de référentiel, pour Référentiels récemment mis à jour, choisissez le nom de votre référentiel. Pour Branch, sélectionnez la branche dans votre référentiel pour vous connecter.
6. Sur la page Paramètres de construction, procédez comme suit :
    a. Pour le nom de l'application, sélectionnez l'application à utiliser pour ajouter un environnement principal. Vous pouvez choisir l'application actuelle ou toute autre application de la région actuelle.
    b. Pour Environnement, sélectionnez le nom de l'environnement principal à ajouter. Vous pouvez utiliser un environnement existant ou en créer un nouveau.
    c. Par défaut, le CI/CD full-stack est désactivé. La désactivation du CI/CD full-stack entraîne l'exécution de l'application en mode pull uniquement. Au moment de la génération, Amplify

générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.

d. Sélectionnez un rôle de service existant pour accorder à Amplify les autorisations nécessaires pour apporter des modifications au backend de votre application. Si vous devez créer un rôle de service, choisissez Créer un nouveau rôle. Pour plus d'informations sur la création d'un rôle de service, consultez la section Ajouter un rôle de service.

e. Choisissez Suivant.

7. Choisissez Save and deploy (Enregistrer et déployer).

Réutilisez les backends lors de la connexion d'une branche à une application existante

Pour réutiliser un backend lors de la connexion d'une branche à une application Amplify existante

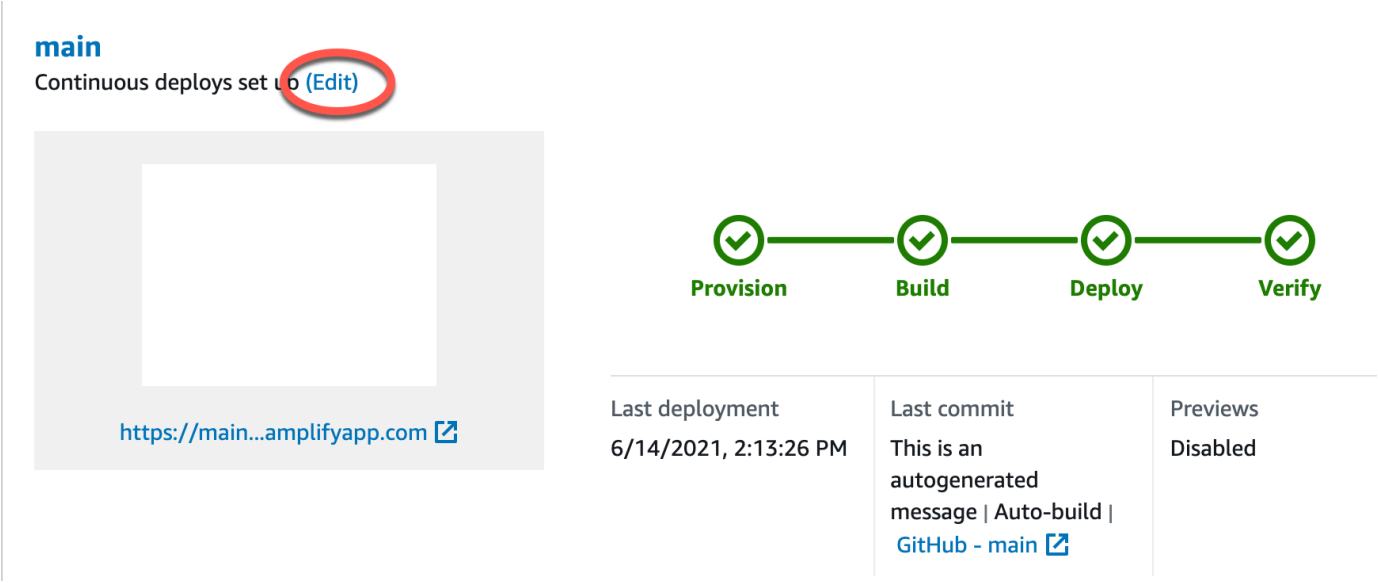
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application à laquelle vous souhaitez connecter une nouvelle succursale.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
4. Dans la section Branches, choisissez Connecter une branche.
5. Sur la page Ajouter une branche de référentiel, pour Branch, sélectionnez la branche de votre référentiel à connecter.
6. Pour le nom de l'application, sélectionnez l'application à utiliser pour ajouter un environnement principal. Vous pouvez choisir l'application actuelle ou toute autre application de la région actuelle.
7. Pour Environnement, sélectionnez le nom de l'environnement principal à ajouter. Vous pouvez utiliser un environnement existant ou en créer un nouveau.
8. Si vous devez configurer un rôle de service pour accorder à Amplify les autorisations nécessaires pour apporter des modifications au backend de votre application, la console vous invite à effectuer cette tâche. Pour plus d'informations sur la création d'un rôle de service, consultez la section Ajouter un rôle de service.
9. Par défaut, le CI/CD full-stack est désactivé. La désactivation du CI/CD full-stack entraîne l'exécution de l'application en mode pull uniquement. Au moment de la génération, Amplify générera automatiquement le `aws-exports.js` fichier uniquement, sans modifier votre environnement principal.
10. Choisissez Suivant.

11. Choisissez Save and deploy (Enregistrer et déployer).

Modifier un frontend existant pour pointer vers un autre backend

Pour modifier un frontend (application Amplify) afin qu'elle pointe vers un autre backend

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez modifier le backend.
3. Choisissez l'onglet Environnements d'hébergement.
4. Localisez la branche à modifier et choisissez Modifier.



5. Sur la page Sélectionnez un environnement principal à utiliser avec cette branche, pour Nom de l'application, sélectionnez l'application frontale pour laquelle vous souhaitez modifier l'environnement principal. Vous pouvez choisir l'application actuelle ou toute autre application de la région actuelle.
6. Pour l'environnement principal, sélectionnez le nom de l'environnement principal à ajouter.
7. Par défaut, le CI/CD full-stack est activé. Décochez cette option pour désactiver le CI/CD full-stack pour ce backend. La désactivation du CI/CD full-stack entraîne l'exécution de l'application en mode pull uniquement. Au moment de la génération, Amplify génère automatiquement le `aws-exports.js` fichier uniquement, sans modifier l'environnement principal.
8. Choisissez Save (Enregistrer). Amplify appliquera ces modifications la prochaine fois que vous créerez l'application.

Déploiements

Les déploiements manuels vous permettent de publier votre application Web avec Amplify Hosting sans connecter un fournisseur Git. Vous pouvez glisser-déposer un dossier depuis votre bureau et héberger votre site en quelques secondes. Vous pouvez également référencer des ressources dans un compartiment Amazon S3 ou spécifier une URL publique vers l'emplacement où vos fichiers sont stockés.

Pour Amazon S3, vous pouvez également configurer desAWS Lambda déclencheurs pour mettre à jour votre site chaque fois que de nouvelles ressources sont téléchargées. Consultez l'article de blog Déployer des fichiers stockés sur Amazon S3, Dropbox ou votre ordinateur de bureau vers laAWS Amplify console pour plus de détails sur la configuration de ce scénario.

Amplify Hosting ne prend pas en charge les déploiements manuels pour les applications de rendu côté serveur (SSR). Pour plus d'informations, veuillez consulter Déployez des applications rendues côté serveur avec Amplify Hosting.

Déploiement manuel par glisser-déposer

Pour déployer manuellement une application par glisser-déposer

1. Connectez-vous à la console AmplifyAWS Management Console et ouvrez la console Amplify.
2. La façon d'accéder à la page Hébergez votre application Web varie selon que vous partez de la page d'accueil d'Amplify ou de la page Toutes les applications.
   - Depuis la page d'accueil d'Amplify
     a. Sélectionnez Get started (Démarrer).
     b. Dans la section Livrer, choisissez Commencer.
   - Depuis la page Toutes les applications
     - Dans le coin supérieur droit, choisissez Nouvelle application, Héberger l'application Web
3. Sur la page Héberger votre application Web, choisissez Déployer sans fournisseur Git. Choisissez ensuite Continue (Continuer).
4. Dans la section Démarrer un déploiement manuel, dans Nom de l'application, entrez le nom de votre application.

5. Dans le champ Nom de l'environnement, entrez un nom significatif pour l'environnement, tel que**`development`** ou**`production`**.
6. Pour Méthode, choisissez Drag and drop.
7. Faites glisser et déposez des fichiers depuis votre bureau vers la zone de dépôt ou utilisez Choisir des fichiers pour sélectionner les fichiers sur votre ordinateur. Les fichiers que vous glissez et déposez ou que vous sélectionnez peuvent être un dossier ou un fichier zip contenant la racine de votre site.
8. Choisissez Save and deploy (Enregistrer et déployer).

Déploiement manuel d'Amazon S3 ou d'URL

Pour déployer manuellement une application depuis Amazon S3 ou une URL publique

1. Connectez-vous à la console AmplifyAWS Management Console et ouvrez la console Amplify.
2. En haut de la page, choisissez Commencer.
3. Dans la section Livrer, choisissez Commencer.
4. Sur la page Héberger votre application Web, choisissez Déployer sans fournisseur Git. Choisissez ensuite Continue (Continuer).
5. Dans la section Démarrer un déploiement manuel, dans Nom de l'application, entrez le nom de votre application.
6. Dans le champ Nom de l'environnement, entrez un nom significatif pour l'environnement, tel que**`development`** ou**`production`**.
7. Dans le champ Méthode, choisissez Amazon S3 ou n'importe quelle URL.
8. La procédure de téléchargement de vos fichiers dépend de la méthode de téléchargement.
   - Amazon S3
     a. Dans le compartiment, sélectionnez le nom du compartiment Amazon S3 dans la liste. Les listes de contrôle d'accès (ACL) doivent être activées pour le compartiment que vous sélectionnez. Pour plus d'informations, veuillez consulter Résolutions des problèmes d'accès aux compartiments Amazon S3.
     b. Dans Fichier Zip, sélectionnez le nom du fichier zip à déployer.
   - N'importe quelle URL
     - Dans le champ URL de la ressource, entrez l'URL du fichier compressé à déployer.

9. Choisissez Save and deploy (Enregistrer et déployer).

> **Note**
>
> Lorsque vous créez le dossier zip, assurez-vous de compresser le contenu de votre sortie de compilation et non le dossier de niveau supérieur. Par exemple, si la sortie de votre build génère un dossier nommé « build » ou « public », naviguez d'abord dans ce dossier, sélectionnez tout le contenu et compressez-le à partir de là. Si vous ne le faites pas, vous verrez un message d'erreur « Accès refusé » car le répertoire racine du site ne sera pas initialisé correctement.

Résolutions des problèmes d'accès aux compartiments Amazon S3

Lorsque vous créez un compartiment Amazon S3, vous utilisez son paramètre de propriété d'objets Amazon S3 pour contrôler si les listes de contrôle d'accès (ACL) sont dans le compartiment. Pour déployer manuellement une application sur Amplify à partir d'un compartiment Amazon S3, les ACL doivent être activées sur le compartiment.

Si vous obtenez une`AccessControlList` erreur lorsque vous déployez à partir d'un compartiment Amazon S3, cela signifie que le compartiment a été créé avec des ACL désactivées et que vous devez les activer dans la console Amazon S3. Pour obtenir des instructions, consultez la section Définition de la propriété d'un objet sur un compartiment existant dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Déployer vers le bouton Amplify

Le bouton Déployer vers Amplify Hosting vous permet de partager GitHub des projets publiquement ou au sein de votre équipe. Voici une image du bouton :

Ajouter le bouton Déployer vers Amplify Hosting à un référentiel ou à un blog

Ajoutez le bouton à votre fichier GitHub README.md, à votre billet de blog ou à toute autre page de balisage qui affiche du code HTML. Le bouton comporte les deux éléments suivants :

1. Une image SVG située à l'URL `https://oneclick.amplifyapp.com/button.svg`
2. L'URL de la console Amplify avec un lien vers votre GitHub référentiel. Vous pouvez soit copier l'URL de votre référentiel, par exemple`https://github.com/username/repository`, soit fournir un lien profond vers un dossier spécifique, tel que`https://github.com/username/repository/tree/branchname/folder`. Amplify Hosting déploiera la branche par défaut dans votre référentiel. D'autres branches pourront être connectées une fois que l'application sera connectée.

Utilisez l'exemple suivant pour ajouter le bouton à un fichier Markdown, tel que votre fichier GitHub README.md. `https://github.com/username/repository`Remplacez-le par l'URL de votre référentiel.

```
[![amplifybutton](https://oneclick.amplifyapp.com/button.svg)](https://
console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/username/
repository)
```

Utilisez l'exemple suivant pour ajouter le bouton à n'importe quel document HTML. `https://github.com/username/repository`Remplacez-le par l'URL de votre référentiel.

```
<a href="https://console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/
username/repository">
    <img src="https://oneclick.amplifyapp.com/button.svg" alt="Deploy to Amplify
 Hosting">
```

```
</a>
```

Configuration de l'accès Amplify aux GitHub référentiels

Amplify utilise désormais la fonctionnalité GitHub Apps pour autoriser Amplify à accéder en lecture seule aux GitHub référentiels. Avec l' GitHub application Amplify, les autorisations sont plus précises, ce qui vous permet d'accorder à Amplify l'accès uniquement aux référentiels que vous spécifiez. Pour en savoir plus sur les GitHub applications, consultez la section À propos GitHub des applications sur le GitHub site Web.

Lorsque vous connectez une nouvelle application stockée dans un GitHub référentiel, Amplify utilise par défaut l' GitHub application pour accéder au référentiel. Toutefois, les applications Amplify existantes que vous avez précédemment connectées à partir GitHub de dépôts utilisent OAuth pour y accéder. CI/CD continuera de fonctionner pour ces applications, mais nous vous recommandons vivement de les migrer pour utiliser la nouvelle GitHub application Amplify.

Lorsque vous déployez une nouvelle application ou migrez une application existante à l'aide de la console Amplify, vous êtes automatiquement dirigé vers l'emplacement d'installation de l' GitHub application Amplify. Pour accéder manuellement à la page d'accueil d'installation de l'application, ouvrez un navigateur Web et accédez à l'application par région. Utilisez le format`https://github.com/apps/aws-amplify-`*`REGION`* en remplaçant *`REGION`* par la région dans laquelle vous allez déployer votre application Amplify. Par exemple, pour installer l' GitHub application Amplify dans la région USA Ouest (Oregon), accédez à https://github.com/apps/aws-amplify-us-west -2.

Rubriques

- Installation et autorisation de l' GitHub application Amplify pour un nouveau déploiement
- Migration d'uneOAuth application existante vers l' GitHub application Amplify
- Configuration de l' GitHub application Amplify pour lesAWS CloudFormation déploiements, la CLI et le SDK
- Configuration des aperçus Web avec l' GitHub application Amplify

Installation et autorisation de l' GitHub application Amplify pour un nouveau déploiement

Lorsque vous déployez une nouvelle application sur Amplify à partir du code existant dans un GitHub dépôt, suivez les instructions suivantes pour installer et autoriser l' GitHub application.

Pour installer et autoriser l' GitHub application Amplify

1. Connectez-vous à la console AmplifyAWS Management Console et ouvrez la console Amplify.
2. Sur la page Toutes les applications, choisissez Nouvelle application, puis Héberger l'application Web.
3. Sur la page Commencer à utiliser Amplify Hosting, choisissez GitHub, puis choisissez Continuer.
4. S'il s'agit de la première connexion à un GitHub référentiel, une nouvelle page s'ouvre dans votre navigateur sur GitHub .com, demandant l'autorisation d'autoriser l'AWS Amplifyaccès à votre GitHub compte. Choisissez Authorize (Autoriser).
5. Ensuite, vous devez installer l' GitHub application Amplify sur votre GitHub compte. Une page s'ouvre sur GitHub.com demandant l'autorisation d'installer etAWS Amplify d'autoriser votre GitHub compte.
6. Sélectionnez le GitHub compte sur lesquels vous souhaitez installer l' GitHub application Amplify.
7. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
   - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le référentiel de l'application que vous migrez dans les référentiels que vous sélectionnez.
8. Choisissez Installer et autoriser.
9. Vous êtes redirigé vers la page Ajouter une branche de référentiel pour votre application dans la console Amplify.
10. Dans la liste des référentiels récemment mis à jour, sélectionnez le nom du référentiel auquel vous souhaitez vous connecter.
11. Dans la liste Branche, sélectionnez le nom de la branche du référentiel à connecter.
12. Choisissez Next (Suivant).
13. Sur la page Configurer les paramètres de génération, choisissez Suivant.
14. Sur la page Révision, choisissez Enregistrer et déployer.

Migration d'uneOAuth application existante vers l' GitHub application Amplify

Les applications Amplify existantes que vous avez précédemment connectées à partir de GitHub référentiels utilisent OAuth pour accéder aux référentiels. Nous vous recommandons vivement de procéder à la migration de ces applications pour utiliser l' GitHubapplication Amplify.

Suivez les instructions suivantes pour migrer une application et supprimer le webhook OAuth correspondant dans votre GitHub compte. Notez que la procédure de migration varie selon que l' GitHub application Amplify est déjà installée ou non. Après avoir migré votre première application, installé et autorisé l' GitHub application, il vous suffit de mettre à jour les autorisations du référentiel pour les migrations d'applications suivantes.

Pour migrer une application depuis OAuth vers l' GitHub application

1. Connectez-vous à la console AmplifyAWS Management Console et ouvrez la console Amplify.
2. Choisissez l'application que vous souhaitez procéder à la migration.
3. Sur la page d'informations de l'application, recherchez le message bleu Migrer vers notre GitHub application et choisissez Démarrer la migration.
4. Sur la page Installer et autoriser GitHub l'application, choisissez Configurer GitHub l'application.
5. Une nouvelle page s'ouvre dans votre navigateur sur GitHub .com, demandant l'autorisation d'accéderAWS Amplify à votre GitHub compte. Choisissez Authorize (Autoriser).
6. Sélectionnez le GitHub compte sur lesquels vous souhaitez installer l' GitHub application Amplify.
7. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
   - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le référentiel de l'application que vous migrez dans les référentiels que vous sélectionnez.
8. Choisissez Installer et autoriser.
9. Vous êtes redirigé vers la page Installer et autoriser GitHub l'application de votre application dans la console Amplify. Si GitHub l'autorisation réussit, vous voyez un message de réussite. Choisissez Next.
10. Sur la page Installation complète, choisissez Installation complète. Cette étape supprime votre webhook existant, en crée un nouveau et termine la migration.

Configuration de l' GitHub application Amplify pour lesAWS CloudFormation déploiements, la CLI et le SDK

Les applications Amplify existantes que vous avez précédemment connectées à partir de GitHub référentiels utilisent OAuth pour accéder aux référentiels. Cela peut inclure des applications que vous avez déployées à l'aide de l'interface de ligne de commande (CLI) Amplify ou des kits SDK.AWS CloudFormation Nous vous recommandons vivement de procéder à la migration de ces applications pour utiliser la nouvelle GitHub application Amplify. La migration doit être effectuée dans la console Amplify duAWS Management Console. Pour des instructions, consultez Migration d'uneOAuth application existante vers l' GitHub application Amplify.

Vous pouvez utiliserAWS CloudFormation l'interface de ligne de commande Amplify et les SDK pour déployer une nouvelle application Amplify qui utilise l' GitHub application pour accéder aux dépôts. Ce processus nécessite que vous installiez d'abord l' GitHub application Amplify GitHub sur votre compte. Ensuite, vous devrez générer un jeton d'accès personnel sur votre GitHub compte. Enfin, déployez l'application et spécifiez le jeton d'accès personnel.

Installez l' GitHub application Amplify sur votre compte

1. Ouvrez un navigateur Web et accédez à l'emplacement d'installation de l' GitHubapplication Amplify dans laAWS région où vous allez déployer votre application.

   Utilisez le format`https://github.com/apps/aws-amplify-REGION/installations/new` en remplaçant *`REGION`* par votre propre saisie. Par exemple, si vous installez votre application dans la région USA Ouest (Oregon), spécifiez`https://github.com/apps/aws-amplify-us-west-2/installations/new`.
2. Sélectionnez le GitHub compte sur lesquels vous souhaitez installer l' GitHub application Amplify.
3. Effectuez l'une des actions suivantes :
   - Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
   - Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le référentiel de l'application que vous migrez dans les référentiels que vous sélectionnez.
4. Choisissez Installer.

Générez un jeton d'accès personnel sur votre GitHub compte

1. Connectez-vous à votre GitHub compte.
2. Dans le coin supérieur droit, localisez votre photo de profil et choisissez Paramètres dans le menu.
3. Dans le menu de navigation de gauche, sélectionnez Developer settings (Paramètres du développeur).
4. Sur la page GitHub Applications, dans le menu de navigation de gauche, choisissez Jetons d'accès personnels.
5. Sur la page des jetons d'accès personnels, choisissez Générer un nouveau jeton.
6. Sur la page Nouveau jeton d'accès personnel, dans le champ Note, entrez un nom descriptif pour le jeton.
7. Dans la section Sélectionner les étendues, sélectionnez admin:repo_hook.
8. Choisissez Generate token (Générer le jeton).
9. Copiez et enregistrez le jeton d'accès personnel. Vous devrez le fournir lorsque vous déployez une application Amplify à l'aide de la CLI ou des SDK.AWS CloudFormation

Une fois que l' GitHub application Amplify est installée sur votre GitHub compte et que vous avez généré un jeton d'accès personnel, vous pouvez déployer une nouvelle application à l'aide de l'interface de ligne de commande Amplify ou des SDK.AWS CloudFormation Utilisez le`accessToken` champ pour spécifier le jeton d'accès personnel que vous avez créé dans la procédure précédente. Pour plus d'informations, consultez CreateAppla référence de l'API Amplify et AWS::Amplify::Apple guide deAWS CloudFormation l'utilisateur.

La commande CLI suivante déploie une nouvelle application Amplify qui utilise l' GitHub application pour accéder au référentiel. Remplacez *myapp-using-githubapphttps://github.com/Myaccount/react-app* et *MY_TOKEN* par vos propres informations.

```
aws amplify create-app --name myapp-using-githubapp --repository https://github.com/
Myaccount/react-app --access-token MY_TOKEN
```

Configuration des aperçus Web avec l' GitHub application Amplify

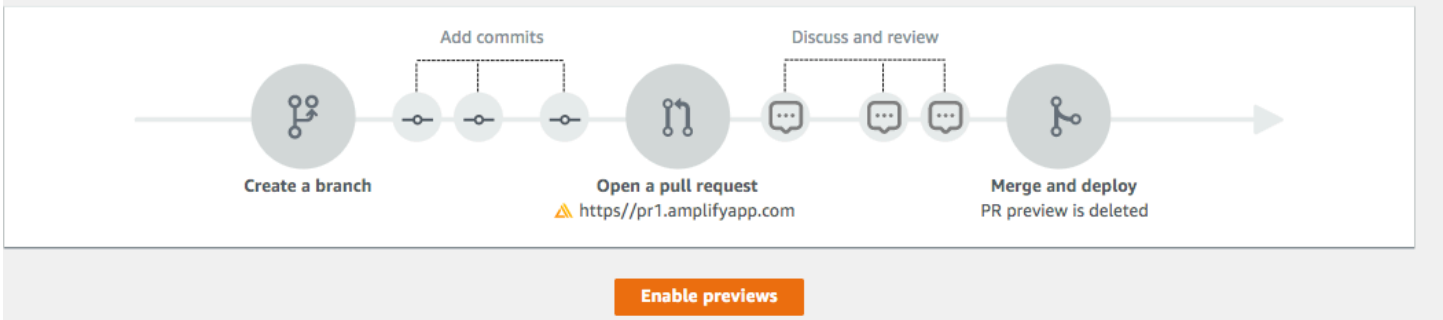
Un aperçu Web déploie chaque pull request (PR) envoyée à votre GitHub référentiel vers une URL d'aperçu unique. Les aperçus utilisent désormais l' GitHub application Amplify pour accéder à votre GitHub référentiel. Pour obtenir des instructions sur l'installation et l'autorisation de l' GitHub application pour les aperçus Web, consultezActiver les aperçus Web.

Aperçus Web pour les pull requests

Les aperçus Web offrent aux équipes de développement et d'assurance qualité (QA) un moyen de prévisualiser les modifications issues des pull requests (PR) avant de fusionner le code vers une branche de production ou d'intégration. Les pull requests vous permettent d'informer les autres utilisateurs des modifications que vous avez apportées à une branche d'un référentiel. Une fois qu'une pull request est ouverte, vous pouvez discuter et examiner les modifications potentielles avec vos collaborateurs et ajouter des validations de suivi avant que vos modifications ne soient fusionnées dans la branche de base.

> **Note**
>
> Actuellement, la branche préliminaire d'Amplify prend en charge GitLab BitBucket, etAWS CodeCommit n'est pas totalement compatible avec les fonctionnalités GitHub. La variable d'`AWS_PULL_REQUEST_ID`environnement n'est disponible que si vous l'utilisez en GitHub tant que fournisseur de référentiel.

Un aperçu Web déploie chaque pull request envoyée à votre référentiel vers une URL d'aperçu unique qui est complètement différente de l'URL utilisée par votre site principal. Pour les applications dont les environnements dorsaux sont provisionnés à l'aide de la CLI Amplify ou d'Amplify Studio, chaque pull request (référentiels Git privés uniquement) génère un backend éphémère qui est supprimé lorsque le PR est fermé.



> **Important**
>
> Pour des raisons de sécurité, vous pouvez activer les aperçus Web sur toutes les applications dotées de référentiels privés, mais pas sur toutes les applications dotées de référentiels publics. Si votre référentiel Git est public, vous pouvez configurer des aperçus uniquement pour les applications qui ne nécessitent pas de rôle de service IAM.

Par exemple, les applications dotées de backends et les applications déployées sur la plate-formeWEB_COMPUTE d'hébergement nécessitent un rôle de service IAM. Par conséquent, vous ne pouvez pas activer les aperçus Web pour ces types d'applications si leur référentiel est public.
Amplify applique cette restriction pour empêcher des tiers de soumettre du code arbitraire qui s'exécuterait en utilisant les autorisations de rôle IAM de votre application.

Activer les aperçus Web

Pour les applications stockées dans un GitHub référentiel, les aperçus utilisent l' GitHub application Amplify pour accéder au référentiel. Si vous activez les aperçus Web sur une application Amplify existante que vous avez précédemment déployée à partir d'un GitHub référentiel à l'aide d'OAuth pour y accéder, vous devez d'abord migrer l'application pour utiliser l' GitHub application Amplify. Pour obtenir des instructions de migration, reportez-vous à la sectionMigration d'uneOAuth application existante vers l' GitHub application Amplify.

Pour activer les aperçus Web pour les pull requests

1. Choisissez Paramètres de l'application, Aperçus, puis sélectionnez Activer les aperçus.

   > **Note**
   >
   > Les aperçus ne sont visibles dans le menu des paramètres de l'application que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel Git. Pour obtenir des instructions sur ce type de déploiement, consultez la section Mise en route avec le code existant.

2. Pour les GitHub référentiels uniquement, procédez comme suit pour installer et autoriser l' GitHub application Amplify sur votre compte :

   a. Dans la fenêtre Installer GitHub l'application pour activer les aperçus, choisissez Installer GitHub l'application.

   b. Sélectionnez le GitHub compte sur lequel vous souhaitez configurer l' GitHubapplication Amplify.

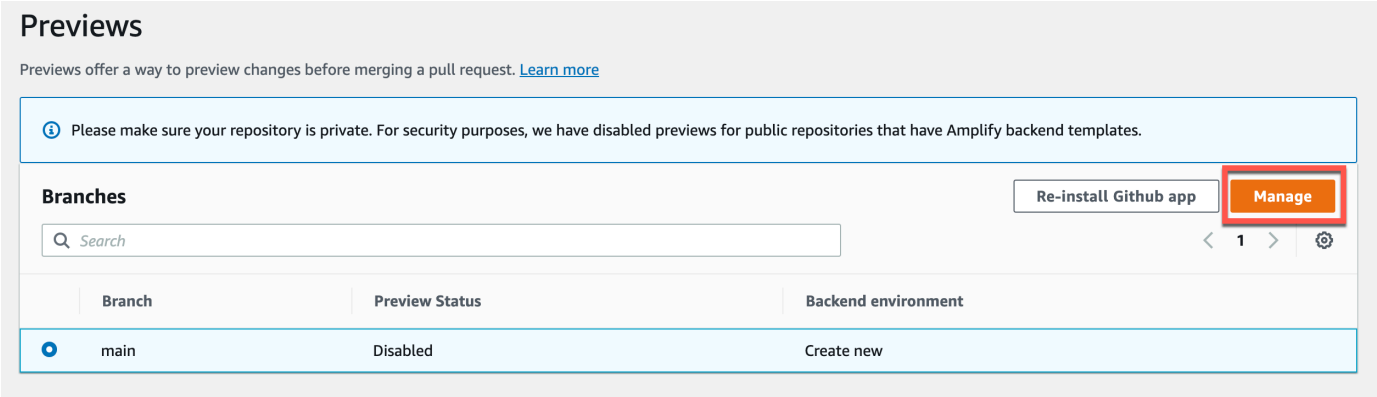
   c. Une page s'ouvre sur GitHub.com pour configurer les autorisations de dépôt pour votre compte.

d. Effectuez l'une des actions suivantes :

- Pour appliquer l'installation à tous les référentiels, choisissez Tous les référentiels.
- Pour limiter l'installation aux référentiels spécifiques que vous sélectionnez, choisissez Ne sélectionner que les référentiels. Assurez-vous d'inclure le référentiel de l'application pour laquelle vous activez les aperçus Web dans les référentiels que vous sélectionnez.

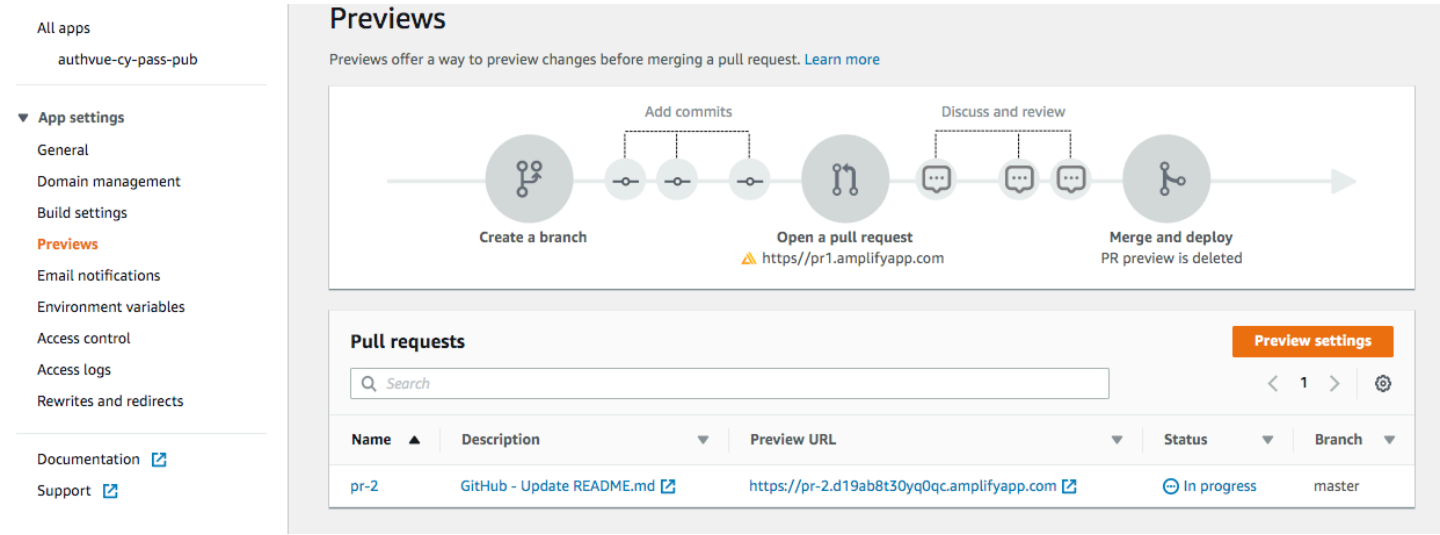
e. Choisissez Enregistrer

3. Après avoir activé les aperçus pour votre référentiel, revenez à la console Amplify pour activer les aperçus pour des branches spécifiques. Sur la page Aperçus, sélectionnez une branche dans la liste et choisissez Gérer.

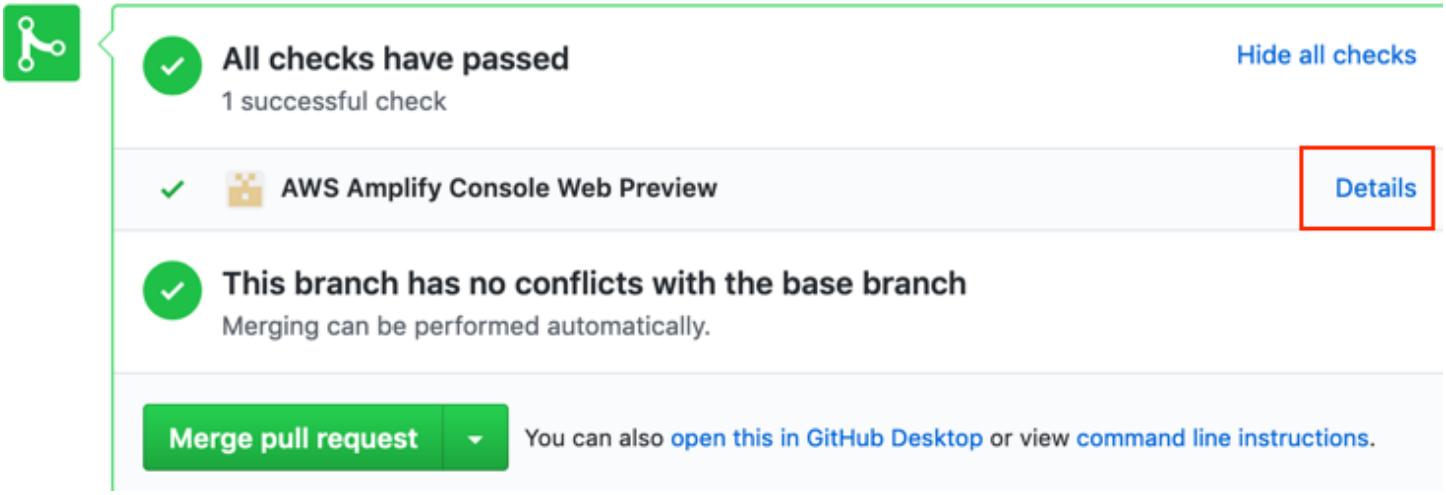


4. Dans la fenêtre Gérer les paramètres d'aperçu pour les succursales, activez les aperçus des demandes d'extraction.
5. Pour obtenir des instructions sur la configuration complète, veuillez consulter l', veuillez consulter l', veuillez consulter :
   - Choisissez Créer un nouvel environnement de backend pour chaque Pull Request. Cette option vous permet de tester les modifications sans impact sur la production.
   - Choisissez Pointer toutes les Pull Requests de cette branche vers un environnement existant.
6. Choisissez Confirm (Confirmer).

La prochaine fois que vous soumettrez une pull request pour la branche, Amplify crée et déploie votre PR sur une URL d'aperçu.



Pour les GitHub référentiels uniquement, vous pouvez accéder à un aperçu de votre URL directement à partir de la pull request de votre GitHub compte.



Une fois la pull request fermée, l'URL d'aperçu est supprimée et tout environnement de backend temporaire lié à la pull request est supprimé.

Accès à l'aperçu Web avec sous-domaines

Les aperçus Web issus des pull requests sont accessibles avec les sous-domaines d'une application Amplify connectée à un domaine personnalisé géré par Amazon Route 53. Lorsque la pull request est fermée, les branches et les sous-domaines associés à la pull request sont automatiquement supprimés. Il s'agit du comportement par défaut pour les aperçus Web une fois que vous avez configuré des déploiements de branches de fonctionnalités basés sur des modèles pour votre application. Pour obtenir des instructions sur la configuration de sous-domaines automatiques,

consultezConfigurer des sous-domaines automatiques pour un domaine personnalisé Amazon Route 53.

Ajoutez des tests Cypress de bout en bout à votre application Amplify

Vous pouvez exécuter des tests de bout en bout (E2E) lors de la phase de test de votre application Amplify pour détecter les régressions avant de mettre le code en production. La phase de test peut être configurée dans la spécification de construction YAML. Actuellement, vous ne pouvez exécuter que le framework de test Cypress lors d'une compilation.

Tutoriel : Configurez des tests de bout en bout avec Cypress

Cypress est un framework de test JavaScript basé sur lequel vous pouvez exécuter des tests E2E sur un navigateur. Pour un didacticiel expliquant comment configurer des tests E2E, consultez l'article de blog Exécuter des tests Cypress de bout en bout pour votre déploiement CI/CD complet avec Amplify.

Ajoutez des tests à votre application Amplify existante

Vous pouvez ajouter des tests Cypress à une application existante en mettant à jour les paramètres de génération de l'application dans la console Amplify. La spécification de compilation YAML contient un ensemble de commandes de compilation et de paramètres associés qu'Amplify utilise pour exécuter votre build. Utilisez cette `test` étape pour exécuter toutes les commandes de test au moment de la génération. Pour les tests E2E, Amplify Hosting propose une intégration plus approfondie avec Cypress qui vous permet de générer un rapport d'interface utilisateur pour vos tests.

La liste suivante décrit les paramètres de test et la façon dont ils sont utilisés.

Pré-test

> Installez les dépendances requises pour exécuter les tests Cypress. Amplify Hosting utilise mochawesome pour générer un rapport afin de consulter les résultats de vos tests et d'attendre de configurer le serveur localhost pendant la construction.

test

> Exécutez les commandes Cypress pour effectuer des tests à l'aide de mochawesome.

Après le test

Le rapport mochawesome est généré à partir du JSON de sortie. Notez que si vous utilisez Yarn, vous devez exécuter cette commande en mode silencieux pour générer le rapport mochawesome. Pour Yarn, vous pouvez utiliser la commande suivante.

```
yarn run --silent mochawesome-merge cypress/report/mochawesome-report/
mochawesome*.json > cypress/report/mochawesome.json
```

Artefacts> Répertoire de base

Le répertoire à partir duquel les tests sont exécutés.

artéfacts> configFilePath

Les données du rapport de test générées.

artéfacts>fichiers

Les artefacts générés (captures d'écran et vidéos) sont disponibles en téléchargement.

L'exemple suivant extrait d'un `amplify.yml` fichier de spécification de build montre comment ajouter des tests Cypress à votre application.

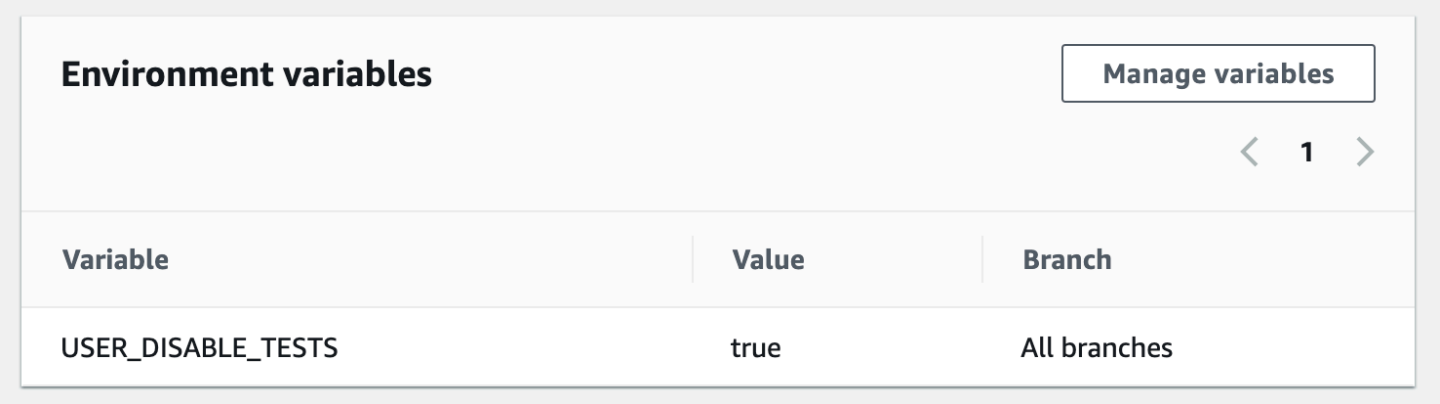
```
test:
  phases:
    preTest:
      commands:
        - npm ci
        - npm install -g pm2
        - npm install -g wait-on
        - npm install mocha mochawesome mochawesome-merge mochawesome-report-generator
        - pm2 start npm -- start
        - wait-on http://localhost:3000
    test:
      commands:
        - 'npx cypress run --reporter mochawesome --reporter-options
 "reportDir=cypress/report/mochawesome-
report,overwrite=false,html=false,json=true,timestamp=mmddyyyy_HHMMss"'
    postTest:
      commands:
```

```
        - npx mochawesome-merge cypress/report/mochawesome-report/mochawesome*.json >
cypress/report/mochawesome.json
        - pm2 kill
 artifacts:
   baseDirectory: cypress
   configFilePath: '**/mochawesome.json'
   files:
     - '**/*.png'
     - '**/*.mp4'
```

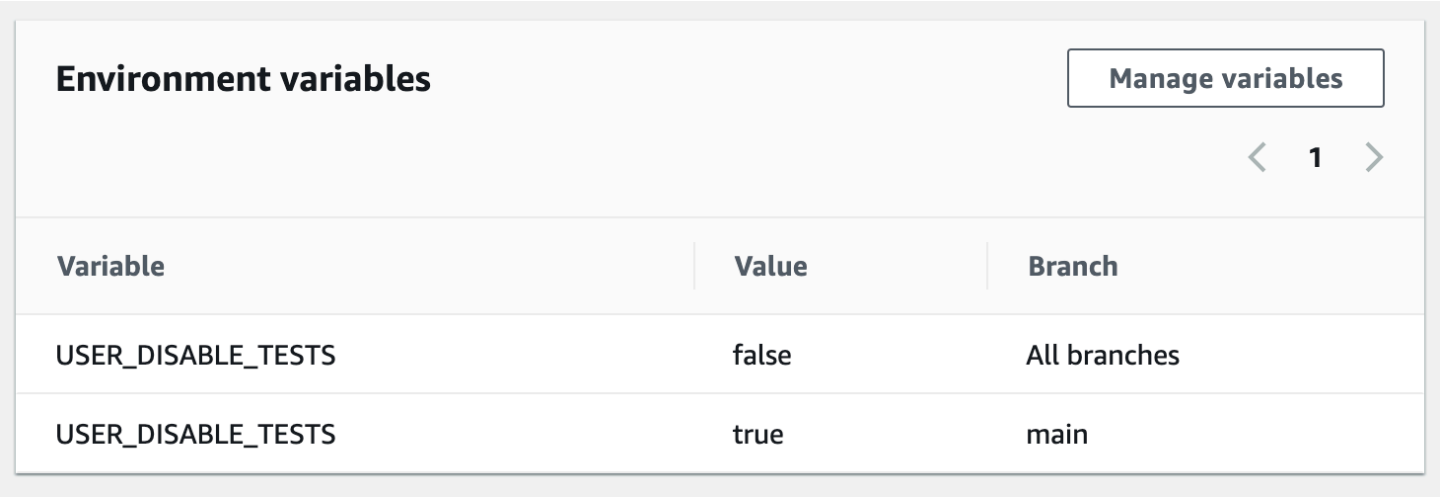
Désactivation des tests

Une fois que la configuration de test a été ajoutée à vos `amplify.yml` paramètres de compilation, l'`test`étape s'exécute pour chaque build, sur chaque branche. Si vous souhaitez désactiver globalement l'exécution des tests ou uniquement exécuter des tests pour des branches spécifiques, vous pouvez utiliser la variable d'USER_DISABLE_TESTSenvironnement sans modifier vos paramètres de génération.

Pour désactiver globalement les tests pour toutes les branches, ajoutez la variable d'USER_DISABLE_TESTSenvironnement avec une valeur de `true` pour toutes les branches. La capture d'écran suivante montre la section des variables d'environnement de la console Amplify avec les tests désactivés pour toutes les branches.



Pour désactiver les tests pour une branche spécifique, ajoutez la variable d'USER_DISABLE_TESTSenvironnement avec la valeur `false` pour toutes les branches, puis ajoutez un remplacement pour chaque branche que vous souhaitez désactiver avec une valeur de`true`. Dans la capture d'écran suivante, les tests sont désactivés sur la branche principale et activés pour toutes les autres branches.



Si vous désactivez les tests avec cette variable, l'étape de test sera complètement ignorée lors de la construction. Pour réactiver les tests, définissez cette valeur sur `false` ou supprimez la variable d'environnement.

Utilisation des redirections

Les redirections permettent à un serveur web de réacheminer la navigation d'une URL vers une autre. Les redirections sont souvent utilisées pour personnaliser l'apparence d'une URL, pour éviter les liens brisés, pour déplacer l'emplacement d'hébergement d'une application ou d'un site sans modifier son adresse et pour remplacer l'URL demandée par le formulaire requis par une application Web.

Types de redirections

Amplify prend en charge les types de redirection suivants dans la console.

Redirection permanente (301)

Les redirections 301 sont conçues pour les modifications durables apportées à la destination d'une adresse web. L'historique de classement du moteur de recherche de l'adresse d'origine s'applique à la nouvelle adresse de destination. La redirection se produit côté client, une barre de navigation de navigateur affiche ainsi l'adresse de destination après la redirection.

Les raisons courantes d'utilisation de redirections 301 incluent les suivantes :

- Pour éviter un lien brisé lorsque l'adresse d'une page change.
- Pour éviter un lien brisé lorsqu'un utilisateur fait une faute de frappe prévisible dans une adresse.

Redirection temporaire (302)

Les redirections 302 sont conçues pour les modifications temporaires apportées à la destination d'une adresse web. L'historique de classement de l'adresse d'origine dans les moteurs de recherche ne s'applique pas à la nouvelle adresse de destination. La redirection se produit côté client, une barre de navigation de navigateur affiche ainsi l'adresse de destination après la redirection.

Les raisons courantes d'utilisation de redirections 302 incluent les suivantes :

- Pour fournir une destination de détour lorsque des réparations sont effectuées sur l'adresse d'origine.
- Fournir des pages de test pour la comparaison A/B d'une interface utilisateur.

> **Note**
>
> Si votre application renvoie une réponse 302 inattendue, l'erreur est probablement due aux modifications que vous avez apportées à la redirection et à la configuration personnalisée de l'en-tête de votre application. Pour résoudre ce problème, vérifiez que vos en-têtes personnalisés sont valides, puis réactivez la règle de réécriture 404 par défaut pour votre application.

Réécriture (200)

Les redirections 200 (réécritures) sont conçues pour afficher le contenu de l'adresse de destination comme s'il était servi à partir de l'adresse d'origine. L'historique de classement du moteur de recherche continue à s'appliquer à l'adresse d'origine. La redirection se produit côté serveur, une barre de navigation de navigateur affiche ainsi l'adresse d'origine après la redirection. Les raisons courantes d'utilisation de redirections 200 incluent les suivantes :

- Pour rediriger l'ensemble d'un site vers un nouvel emplacement d'hébergement sans modifier l'adresse du site.
- Pour rediriger l'ensemble du trafic vers une application web monopage (SPA) vers sa page index.html pour traitement par une fonction de routeur côté client.

Introuvable (404)

Les redirections 404 se produisent lorsqu'une demande pointe vers une adresse qui n'existe pas. La page de destination d'une redirection 404 s'affiche au lieu de celle demandée. Les raisons courantes d'une redirection 404 incluent les suivantes :

- Pour éviter un message de lien brisé lorsqu'un utilisateur saisit une URL incorrecte.
- Pour pointer des requêtes de pages inexistantes d'une application web vers sa page index.html pour traitement par une fonction de routeur côté client.

Création et modification de redirections

Vous pouvez créer et modifier des redirections pour une application dans la console Amplify. Avant de commencer, vous aurez besoin des informations suivantes concernant les différentes parties d'une redirection.

Une adresse originale

Adresse demandée par l'utilisateur.

Une adresse de destination

L'adresse qui diffuse réellement le contenu que l'utilisateur voit.

Un type de redirection

Les types incluent une redirection permanente (301), une redirection temporaire (302), une réécriture (200) ou une redirection introuvable (404).

Un code de pays à deux lettres (facultatif)

Une valeur que vous pouvez inclure pour segmenter l'expérience utilisateur de votre application par région géographique.

Pour créer une redirection dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez créer une redirection.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis sélectionnez Réécritures et redirections.
4. Dans la section Réécritures et redirections, choisissez Modifier.
5. La procédure d'ajout d'une redirection varie selon que vous souhaitez ajouter des règles individuellement ou effectuer une modification en bloc :
   - Pour créer une redirection individuelle, choisissez Ajouter une règle.
     a. Pour Adresse source, entrez l'adresse d'origine demandée par l'utilisateur.
     b. Pour Adresse cible, entrez l'adresse de destination qui affiche le contenu à l'utilisateur.
     c. Pour Type, choisissez le type de redirection dans la liste.
     d. (Facultatif) Pour le code de pays, entrez une condition de code de pays à deux lettres.

- Pour modifier les redirections en bloc, choisissez Ouvrir un éditeur de texte.
    - Ajoutez ou mettez à jour manuellement des redirections dans l'éditeur JSON d'ajout groupé de réécritures et de redirections.

6. Choisissez Enregistrer.

Ordre des redirections

Les redirections sont exécutées à partir du haut de la liste. Assurez-vous que votre ordre donne l'effet voulu. Par exemple, avec l'ordre de redirections suivant, toutes les requêtes d'un chemin donné sous /docs/ sont redirigées vers le même chemin sous /documents/, sauf /docs/specific-filename.html qui redirige vers /documents/different-filename.html :

```
/docs/specific-filename.html /documents/different-filename.html 301
/docs/<*> /documents/<*>
```

L'ordre de redirections suivant ignore la redirection de specific-filename.html vers different-filename.html :

```
/docs/<*> /documents/<*>
/docs/specific-filename.html /documents/different-filename.html 301
```

Paramètres Query (Requête)

Vous pouvez utiliser les paramètres de requête pour mieux contrôler vos correspondances d'URL. Amplify transmet tous les paramètres de requête vers le chemin de destination pour les redirections 301 et 302, avec les exceptions suivantes :

- Si l'adresse d'origine inclut une chaîne de requête définie sur une valeur spécifique, Amplify ne transmet pas les paramètres de requête. Dans ce cas, la redirection s'applique uniquement aux demandes adressées à l'URL de destination avec la valeur de requête spécifiée.
- Si l'adresse de destination de la règle correspondante comporte des paramètres de requête, les paramètres de requête ne sont pas transférés. Par exemple, si l'adresse de destination de la redirection est`https://example-target.com?q=someParam`, les paramètres de requête ne sont pas transmis.

Redirections et réécritures simples

Cette section inclut des exemples de code pour les scénarios de redirection courants.

> **Note**
>
> La correspondance du domaine d'adresse d'origine ne fait pas la distinction majuscules/minuscules.

Vous pouvez utiliser l'exemple de code suivant pour rediriger définitivement une page spécifique à une nouvelle adresse.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/original.html`	`/destination.html`	`permanent redirect (301)`	

JSON [{"source": "/original.html", "status": "301", "target": "/destination.html", "condition": null}]

Vous pouvez utiliser l'exemple de code suivant pour rediriger n'importe quel chemin dans un dossier vers le même chemin dans un autre dossier.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>`	`/documents/<*>`	`permanent redirect (301)`	

JSON [{"source": "/docs/<*>", "status": "301", "target": "/documents/<*>", "condition": null}]

Vous pouvez utiliser l'exemple de code suivant pour rediriger l'ensemble du trafic vers index.html en tant que réécriture. Dans ce scénario, la réécriture permet d'indiquer à l'utilisateur qu'il a accédé à l'adresse d'origine.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/<*>`	`/index.html`	`rewrite (200)`	

JSON [{"source": "/<*>", "status": "200", "target": "/index.html", "condition": null}]

Vous pouvez utiliser l'exemple de code suivant pour utiliser une réécriture afin de modifier le sous-domaine qui est présenté à l'utilisateur.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`https://mydomain.com`	`https://www.mydomain.com`	`rewrite (200)`	

JSON [{"source": "https://mydomain.com", "status": "200", "target": "https://www.mydomain.com", "condition": null}]

Vous pouvez utiliser l'exemple de code suivant pour rediriger vers un autre domaine avec un préfixe de chemin.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`https://mydomain.com`	`https://www.mydomain.com/documents`	`temporary redirect (302)`	

JSON [{"source": "https://mydomain.com", "status": "302", "target": "https://www.mydomain.com/documents/", "condition": null}]

Vous pouvez utiliser l'exemple de code suivant pour rediriger les chemins d'un dossier introuvable vers une page 404 personnalisée.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/<*>`	`/404.html`	`not found (404)`	

JSON [{"source": "/<*>", "status": "404", "target": "/404.html", "condition": null}]

Redirections pour les applications Web à page unique (SPA)

La plupart des infrastructures SPA prennent en charge HTML5 history.pushState() pour modifier l'emplacement du navigateur sans déclencher de requête serveur. Cela fonctionne pour les utilisateurs qui commencent leur transition à partir de la racine (ou /index.html), mais échoue pour les utilisateurs qui accèdent directement à une autre page.

L'exemple suivant utilise des expressions régulières pour configurer une réécriture 200 pour tous les fichiers dans index.html, à l'exception des extensions de fichier spécifiées dans l'expression régulière.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`</^[^.]+$\|\.(?!(css\|gif\|ico\|jpg\|js\|png\|txt\|svg\|woff\|woff2\|ttf\|map\|json\|webp)$)([^.]+$)/>`	`/index.html`	`200`	

JSON [{"source": "</^[^.]+$|\.(?!(css|gif|ico|jpg|js|png|txt|svg|woff|woff2|ttf|map|json|webp)$)([^.]+$)/>", "status": "200", "target": "/index.html", "condition": null}]

Réécriture du proxy inversé

L'exemple suivant utilise une réécriture pour transférer du contenu provenant d'un autre emplacement afin que l'utilisateur sache que le domaine n'a pas changé.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/images/<*>`	`https://images.otherdomain.com/<*>`	`rewrite (200)`	

JSON [{"source": "/images/<*>", "status": "200", "target": "https://images.otherdomain.com/<*>", "condition": null}]

Barres obliques et URL propres

Pour créer des structures d'URL propres telles que about au lieu de about.html, des générateurs sur site statiques tels que Hugo génèrent des répertoires pour les pages avec un index.html (/about/index.html). Amplify crée automatiquement des URL propres en ajoutant une barre oblique lorsque cela est nécessaire. Le tableau ci-dessous présente différents scénarios :

Entrées utilisateur dans le navigateur	URL dans la barre d'adresse	Document affiché
`/about`	`/about`	`/about.html`
`/about (when about.html returns 404)`	`/about/`	`/about/index.html`
`/about/`	`/about/`	`/about/index.html`

Espaces réservés

Vous pouvez utiliser l'exemple de code suivant pour rediriger des chemins dans une structure de dossiers vers une structure correspondante dans un autre dossier.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<year>/<month>/<date>/<itemid>`	`/documents/<year>/<month>/<date>/<itemid>`	`permanent redirect (301)`	

JSON [{"source": "/docs/<year>/<month>/<date>/<itemid>", "status": "301", "target": "/documents/<year>/<month>/<date>/<itemid>", "condition": null}]

Chaînes de requête et paramètres de chemin

Vous pouvez utiliser l'exemple de code suivant pour rediriger un chemin vers un dossier avec un nom qui correspond à la valeur d'un élément de la chaîne de requête dans l'adresse d'origine :

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs?id=<my-blog-id-value`	`/documents/<my-blog-post-id-value>`	`permanent redirect (301)`	

JSON [{"source": "/docs?id=<my-blog-id-value>", "status": "301", "target": "/documents/<my-blog-id-value>", "condition": null}]

> **Note**
>
> Amplify transmet tous les paramètres de chaîne de requête au chemin de destination pour les redirections 301 et 302. Toutefois, si l'adresse d'origine inclut une chaîne de requête définie sur une valeur spécifique, comme illustré dans cet exemple, Amplify ne transmet pas les paramètres de requête. Dans ce cas, la redirection s'applique uniquement aux demandes adressées à l'adresse de destination avec la valeur de requête spécifiée`id`.

Vous pouvez utiliser l'exemple de code suivant pour rediriger tous les chemins introuvables à un niveau donné d'une structure de dossiers vers index.html dans un dossier spécifique.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/documents/<folder>/<child-folder>/<grand-child-folder>`	`/documents/index.html`	`not found (404)`	

JSON [{"source": "/documents/<x>/<y>/<z>", "status": "404", "target": "/documents/index.html", "condition": null}]

Redirections basées sur les régions

Vous pouvez utiliser l'exemple de code suivant pour rediriger des requêtes en fonction de la région.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/documents`	`/documents/us/`	`temporary redirect (302)`	`<US>`

JSON [{"source": "/documents", "status": "302", "target": "/documents/us/", "condition": "<US>"}]

Expressions génériques dans les redirections et les réécritures

Vous pouvez utiliser l'expression générique`<*>`, dans l'adresse d'origine pour une redirection ou une réécriture. Vous devez placer l'expression à la fin de l'adresse d'origine et elle doit être unique. Amplify ignore les adresses d'origine qui incluent plusieurs expressions génériques, ou les utilise à un emplacement différent.

Voici un exemple de redirection valide avec une expression générique.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>`	`/documents/<*>`	`permanent redirect (301)`	

Les deux exemples suivants illustrent des redirections non valides avec des expressions génériques.

Adresse d'origine	Adresse de destination	Type de redirection	Code pays
`/docs/<*>/content`	`/documents/<*>/content`	`permanent redirect (301)`	
`/docs/<*>/content/<*>`	`/documents/<*>/content/<*>`	`permanent redirect (301)`	

Restreindre l'accès aux succursales

Si vous travaillez sur des fonctionnalités inédites, vous pouvez protéger par mot de passe les branches de fonctionnalités qui ne sont pas prêtes à être accessibles au public. Lorsque le contrôle d'accès est défini sur une branche, les utilisateurs sont invités à saisir un nom d'utilisateur et un mot de passe lorsqu'ils tentent d'accéder à l'URL de la branche.

Pour définir des mots de passe pour les branches de fonctionnalités

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir les mots de passe des branches fonctionnelles.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Contrôle d'accès.
4. Dans la section Paramètres de contrôle d'accès, choisissez Gérer l'accès.
5. Effectuez l'une des opérations suivantes dans les paramètres de contrôle d'accès :
   - Pour définir un nom d'utilisateur et un mot de passe applicables à toutes les succursales connectées, activez Appliquer un mot de passe global. Par exemple, si des branches principale, dev et feature sont connectées, vous pouvez utiliser un mot de passe global pour définir le même nom d'utilisateur et le même mot de passe pour toutes les branches.
   - Pour appliquer un nom d'utilisateur et un mot de passe à une succursale individuelle, désactivez Appliquer un mot de passe global. Pour la succursale pour laquelle vous souhaitez définir un nom d'utilisateur et un mot de passe uniques, choisissez le mot de passe restreint requis pour le paramètre Accès et entrez un nom d'utilisateur et un mot de passe.
6. Si vous gérez le contrôle d'accès pour une application de rendu côté serveur (SSR), redéployez l'application en effectuant une nouvelle compilation à partir de votre dépôt Git. Cette étape est nécessaire pour permettre à Amplify d'appliquer vos paramètres de contrôle d'accès.

Variables d'environnement

Les variables d'environnement sont des paires clé-valeur que vous pouvez ajouter aux paramètres de votre application pour les mettre à la disposition d'Amplify Hosting. La meilleure pratique consiste à utiliser des variables d'environnement pour exposer les données de configuration des applications. Toutes les variables d'environnement que vous ajoutez sont chiffrées pour empêcher tout accès non autorisé.

Amplify ne vous permet pas de créer des variables d'environnement avec un `AWS` préfixe. Ce préfixe est réservé à un usage interne d'Amplify uniquement.

> **Important**
>
> N'utilisez pas de variables d'environnement pour stocker des secrets. Stockez les secrets dans un secret d'environnement créé à l'aide du AWS Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Secrets environnementaux.

Amplifier les variables d'environnement

Les variables d'environnement suivantes sont accessibles par défaut dans la console Amplify.

Nom de variable	Description	Exemple de valeur
_BUILD_TIMEOUT	Durée du délai d'expiration de la construction en minutes	`30`
_LIVE_UPDATES	L'outil sera mis à niveau vers la dernière version.	`[{"name":"Amplify CLI","pkg":"@aws-amplify/cli","type":"npm","version":"latest"}]`
USER_DISABLE_TESTS	L'étape de test est ignorée lors d'une génération. Vous pouvez désactiver les tests pour toutes les branches ou	`true`

Nom de variable	Description	Exemple de valeur
	pour des branches spécifiques d'une application. Cette variable d'environnement est utilisée pour les applications qui effectuent des tests pendant la phase de création. Pour plus d'informations sur la définition de cette variable, consultezDésactivation des tests.	
AWS_APP_ID	ID d'application du build actuel	`abcd1234`
AWS_BRANCH	Nom de branche du build actuel	`main`, `develop`, `beta`, `v2.0`
AWS_BRANCH_ARN	La branche Amazon Resource Name (ARN) de la version actuelle	`aws:arn:amplify:us-west-2:123456789012:appname/branch/...`
AWS_CLONE_URL	URL clone utilisée pour extraire le contenu du référentiel git	`git@github.com:<user-name>/<repo-name>.git`
AWS_COMMIT_ID	L'ID de validation de la version actuelle « HEAD » pour les reconstructions	`abcd1234`
AWS_JOB_ID	ID de tâche du build actuel. Cela inclut un rembourrage de « 0 » afin qu'il ait toujours la même longueur.	`0000000001`

Nom de variable	Description	Exemple de valeur
AWS_PULL_REQUEST_ID	L'ID de pull request de la version de prévisualisation Web. Cette variable d'environnement n'est disponible que si vous l'utilisez en GitHub tant que fournisseur de référentiel.	1
AMPLIFY_AMAZON_CLIENT_ID	L'identifiant du client Amazon	`123456`
AMPLIFY_AMAZON_CLIENT_SECRET	Le secret du client Amazon	`example123456`
AMPLIFY_FACEBOOK_CLIENT_ID	L'identifiant du client Facebook	`123456`
AMPLIFY_FACEBOOK_CLIENT_SECRET	Le secret du client de Facebook	`example123456`
AMPLIFY_GOOGLE_CLIENT_ID	L'identifiant du client Google	`123456`
AMPLIFY_GOOGLE_CLIENT_SECRET	Le secret du client de Google	`example123456`
AMPLIFY_DIFF_DEPLOY	Activez ou désactivez le déploiement frontal basé sur les différences. Pour de plus amples informations, veuillez consulter Activer ou désactiver la création et le déploiement du frontend basé sur les différences.	`true`

Nom de variable	Description	Exemple de valeur
AMPLIFY_DIFF_DEPLOY_ROOT	Le chemin à utiliser pour les comparaisons de déploiements frontaux basées sur les différences, par rapport à la racine de votre référentiel.	`dist`
AMPLIFY_DIFF_BACKEND	Activez ou désactivez les versions de backend basées sur les différences. Pour plus d'informations, consultez Activer ou désactiver les versions de backend basées sur les différences.	`true`
AMPLIFY_BACKEND_PULL_ONLY	Amplify gère cette variable d'environnement. Pour plus d'informations, consultez Modifier un frontend existant pour pointer vers un autre backend.	`true`
AMPLIFY_BACKEND_APP_ID	Amplify gère cette variable d'environnement. Pour plus d'informations, consultez Modifier un frontend existant pour pointer vers un autre backend.	`abcd1234`
AMPLIFY_SKIP_BACKEND_BUILD	Si votre spécification de build ne contient aucune section de backend et que vous souhaitez désactiver les builds de backend, définissez cette variable d'environnement sur. `true`	`true`

Nom de variable	Description	Exemple de valeur
AMPLIFY_ENABLE_DEBUG_OUTPUT	Définissez cette variable sur `true` pour imprimer une trace de pile dans les journaux. Cela est utile pour corriger les erreurs de compilation du backend.	`true`
AMPLIFY_MONOREPO_APP_ROOT	Le chemin à utiliser pour spécifier la racine d'une application monorepo, par rapport à la racine de votre dépôt.	`apps/react-app`
AMPLIFY_USERPOOL_ID	L'ID du groupe d'utilisateurs Amazon Cognito importé pour l'authentification	`us-west-2_example`
AMPLIFY_WEBCLIENT_ID	L'ID du client d'application à utiliser par les applications Web Le client de l'application doit être configuré pour accéder au groupe d'utilisateurs Amazon Cognito spécifié par la variable d'environnement AMPLIFY_USERPOOL_ID.	`123456`

Nom de variable	Description	Exemple de valeur
AMPLIFY_NATIVECLIENT_ID	L'ID du client d'application à utiliser par les applications natives Le client de l'application doit être configuré pour accéder au groupe d'utilisateurs Amazon Cognito spécifié par la variable d'environnement AMPLIFY_USERPOOL_ID.	`123456`
AMPLIFY_IDENTITYPOOL_ID	L'ID du pool d'identités Amazon Cognito	`example-identitypool-id`
AMPLIFY_PERMISSIONS_BOUNDARY_ARN	L'ARN que la politique IAM doit utiliser comme limite d'autorisations qui s'applique à tous les rôles IAM créés par Amplify. Pour plus d'informations, consultez la section Limite des autorisations IAM pour les rôles générés par Amplify.	`arn:aws:iam::123456789012:policy/example-policy`
AMPLIFY_DESTRUCTIVE_UPDATES	Définissez cette variable d'environnement sur true pour permettre à une API GraphQL d'être mise à jour avec des opérations de schéma susceptibles de provoquer une perte de données. Pour plus d'informations, consultez la section Mettre à jour le schéma.	`true`

> **Note**
>
> Les variables d'`AMPLIFY_AMAZON_CLIENT_SECRET`environnement `AMPLIFY_AMAZON_CLIENT_ID` et sont des jetons OAuth, et non une clé d' AWS accès ou une clé secrète.

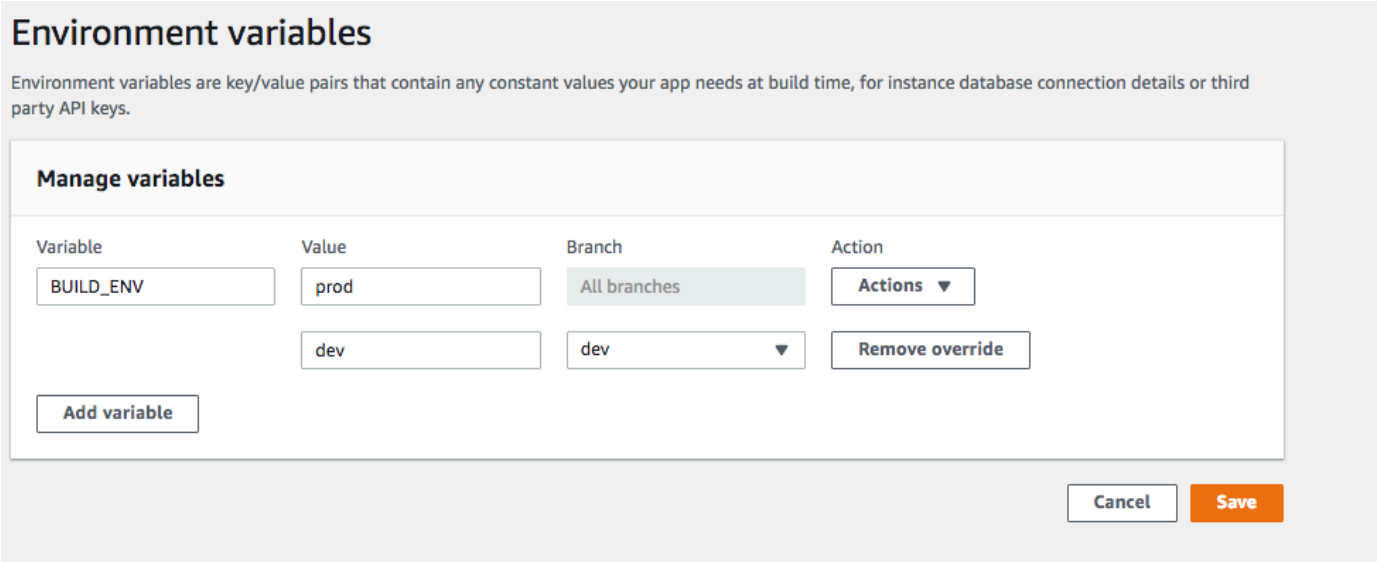
Définir les variables d'environnement

Utilisez les instructions suivantes pour définir les variables d'environnement d'une application dans la console Amplify.

> **Note**
>
> Les variables d'environnement ne sont visibles dans le menu des paramètres de l'application de la console Amplify que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel git. Pour obtenir des instructions sur ce type de déploiement, voir Commencer avec le code existant.

Pour définir des variables d'environnement

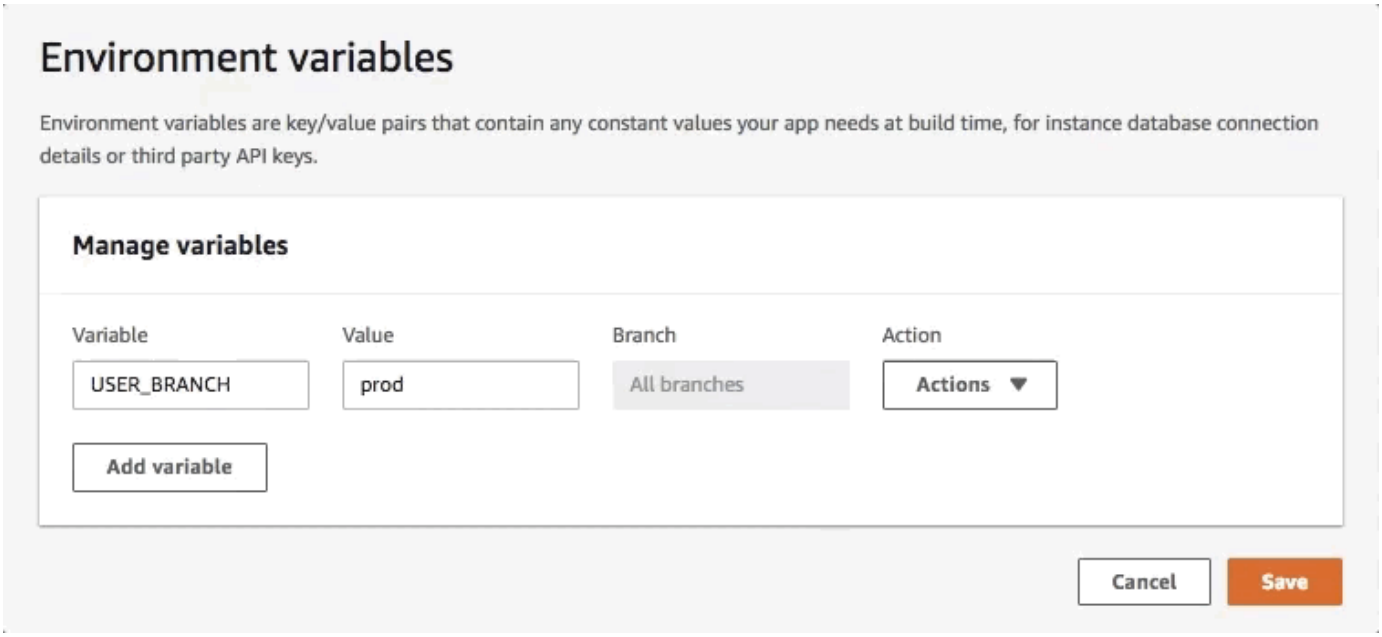
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Dans la console Amplify, choisissez Paramètres de l'application, puis choisissez Variables d'environnement.
3. Dans la section Variables d'environnement, sélectionnez Gérer les variables.
4. Dans la section Gérer les variables, sous Variable, entrez votre clé. Dans Valeur, entrez votre valeur. Par défaut, Amplify applique les variables d'environnement à toutes les branches, de sorte que vous n'avez pas à saisir à nouveau les variables lorsque vous connectez une nouvelle branche.



5. (Facultatif) Pour personnaliser une variable d'environnement spécifiquement pour une branche, ajoutez une dérogation de branche comme suit :

   a. Choisissez Actions, puis sélectionnez Ajouter un remplacement de variable.

   b. Vous disposez maintenant d'un ensemble de variables d'environnement spécifique à votre branche.



6. Choisissez Enregistrer.

Accédez aux variables d'environnement au moment de la création

Pour accéder à une variable d'environnement pendant une génération, modifiez vos paramètres de génération pour inclure la variable d'environnement dans vos commandes de génération.

Pour modifier les paramètres de compilation afin d'inclure une variable d'environnement

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Dans la console Amplify, choisissez Paramètres de l'application, puis sélectionnez Paramètres de génération.
3. Dans la section Spécification de construction de l'application, choisissez Modifier.
4. Ajoutez la variable d'environnement à votre commande de génération. Vous devez maintenant être en mesure d'accéder à votre variable d'environnement lors de la génération suivante. Cet exemple modifie le comportement du npm (BUILD_ENV) et ajoute un jeton d'API (TWITCH_CLIENT_ID) pour un service externe à un fichier d'environnement pour une utilisation ultérieure.

```
build:
  commands:
    - npm run build:$BUILD_ENV
    - echo "TWITCH_CLIENT_ID=$TWITCH_CLIENT_ID" >> backend/.env
```

Chaque commande de votre configuration de build s'exécute dans un shell Bash. Pour plus d'informations sur l'utilisation des variables d'environnement dans Bash, consultez les extensions Shell dans le manuel GNU Bash.

Rendre les variables d'environnement accessibles aux environnements d'exécution côté serveur

Par défaut, un composant serveur Next.js n'a pas accès aux variables d'environnement de votre application. Ce comportement est intentionnel pour protéger les secrets stockés dans les variables d'environnement que votre application utilise pendant la phase de génération.

Pour rendre des variables d'environnement spécifiques accessibles à Next.js, vous devez modifier le fichier de spécification de build Amplify afin de définir les variables d'environnement dans les fichiers d'environnement reconnus par Next.js. Cela permet à Amplify de charger les variables

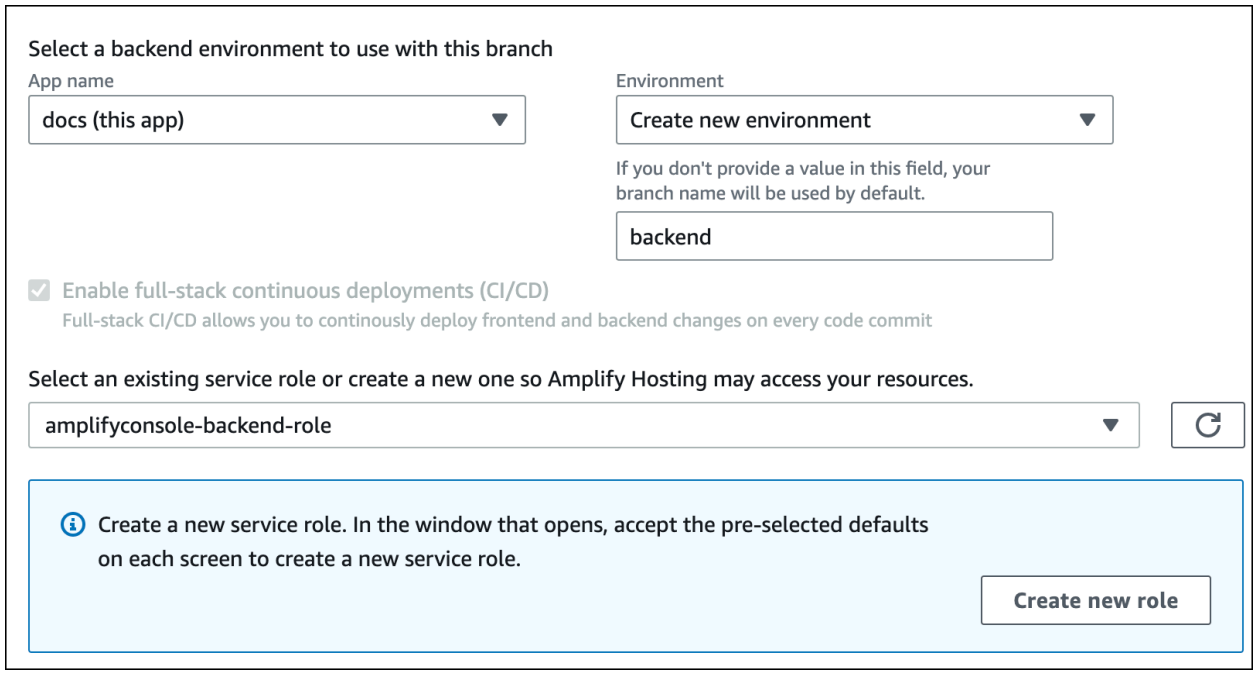
d'environnement avant de créer l'application. Pour plus d'informations sur la modification de votre spécification de construction, consultez les exemples d'ajout de variables d'environnement dans la section des commandes de construction.

Création d'un nouvel environnement principal avec des paramètres d'authentification pour la connexion sociale

Pour connecter une succursale à une application

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. La procédure de connexion d'une branche à une application varie selon que vous connectez une succursale à une nouvelle application ou à une application existante.
   - Connecter une succursale à une nouvelle application

     a. Sur la page des paramètres de génération, recherchez la section Sélectionnez un environnement principal à utiliser avec cette branche. Pour Environnement, choisissez Créer un nouvel environnement et entrez le nom de votre environnement principal. La capture d'écran suivante montre la section Sélectionnez un environnement principal à utiliser avec cette branche de la page des paramètres de génération avec le **backend** nom de l'environnement principal saisi.



     b. Développez la section Paramètres avancés sur la page des paramètres de création et ajoutez des variables d'environnement pour les clés de connexion aux réseaux

sociaux. Par exemple, **`AMPLIFY_FACEBOOK_CLIENT_SECRET`** est une variable d'environnement valide. Pour la liste des variables d'environnement du système Amplify disponibles par défaut, consultez le tableau dans. Amplifier les variables d'environnement

- Connecter une succursale à une application existante

  a. Si vous connectez une nouvelle branche à une application existante, définissez les variables d'environnement de connexion sociale avant de connecter la branche. Dans le volet de navigation, choisissez Paramètres de l'application, Variables d'environnement.

  b. Dans la section Variables d'environnement, sélectionnez Gérer les variables.

  c. Dans la section Gérer les variables, choisissez Ajouter une variable.

  d. Pour Variable (clé), entrez votre identifiant client. Dans Value, entrez le secret de votre client.

  e. Choisissez Enregistrer.

Variables d'environnement du framework frontal

Si vous développez votre application avec un framework frontal qui prend en charge ses propres variables d'environnement, il est important de comprendre que celles-ci ne sont pas identiques aux variables d'environnement que vous configurez dans la console Amplify. Par exemple, React (préfixé REACT_APP) et Gatsby (préfixé GATSBY) vous permettent de créer des variables d'environnement d'exécution que ces frameworks intègrent automatiquement dans la version de production de votre frontend. Pour comprendre les effets de l'utilisation de ces variables d'environnement pour stocker des valeurs, reportez-vous à la documentation du framework d'interface que vous utilisez.

Le stockage de valeurs sensibles, telles que les clés d'API, dans ces variables d'environnement préfixées par le framework frontal n'est pas une bonne pratique et est fortement déconseillé. Pour un exemple d'utilisation des variables d'environnement de temps de construction d'Amplify à cette fin, voirAccédez aux variables d'environnement au moment de la création.

Secrets environnementaux

Les secrets d'environnement sont similaires aux variables d'environnement, mais il s'agit de paires clé-valeur du magasin de paramètres AWS Systems Manager (SSM) qui peuvent être chiffrées. Certaines valeurs doivent être chiffrées, comme la clé privée de connexion avec Apple pour Amplify.

Définissez les secrets de l'environnement

Suivez les instructions suivantes pour définir un secret d'environnement pour une application Amplify à l'aide de la AWS Systems Manager console.

Pour définir un secret d'environnement

1. Connectez-vous à la AWS Systems Manager console AWS Management Console et ouvrez-la.
2. Dans le volet de navigation, choisissez Application Management, puis Parameter Store.
3. Sur la page AWS Systems Manager Parameter Store, choisissez Create parameter.
4. Sur la page Créer un paramètre, dans la section Détails du paramètre, procédez comme suit :
   a. Pour Nom, entrez un paramètre au format**`/amplify/{your_app_id}/{your_backend_environment_name}/{your_parameter_name}`**.
   b. Dans le champ Type, sélectionnez SecureString.
   c. Pour la source de clé KMS, choisissez Mon compte actuel pour utiliser la clé par défaut pour votre compte.
   d. Dans Valeur, entrez votre valeur secrète à chiffrer.
5. Choisissez Créer un paramètre.

> **Note**
>
> Amplify n'a accès qu'aux clés situées sous le build `/amplify/{your_app_id}/{your_backend_environment_name}` de l'environnement spécifique. Vous devez spécifier la valeur par défaut AWS KMS key pour permettre à Amplify de déchiffrer la valeur.

Accédez aux secrets de l'environnement

L'accès à un secret d'environnement pendant une construction est similaire à l'accès à des variables d'environnement, sauf que les secrets d'environnement sont stockés `process.env.secrets` sous forme de chaîne JSON.

Amplifiez les secrets de l'environnement

Spécifiez un paramètre Systems Manager au format`/amplify/{your_app_id}/{your_backend_environment_name}/AMPLIFY_SIWA_CLIENT_ID`.

Vous pouvez utiliser les secrets d'environnement suivants, accessibles par défaut dans la console Amplify.

Nom de variable	Description	Exemple de valeur
AMPLIFY_SIWA_CLIENT_ID	La connexion à l'aide de l'identifiant client Apple	`com.yourapp.auth`
AMPLIFY_SIWA_TEAM_ID	La connexion à l'aide de l'identifiant d'équipe Apple	`ABCD123`
AMPLIFY_SIWA_KEY_ID	La connexion à l'aide de l'identifiant Apple Key	`ABCD123`
AMPLIFY_SIWA_PRIVATE_KEY	La clé privée de connexion avec Apple	-----COMMENCER LA CLÉ PRIVÉE----- ****..... -----FIN DE LA CLÉ PRIVÉE-----

En-têtes personnalisés

Les en-têtes HTTP personnalisés vous permettent de spécifier des en-têtes pour chaque réponse HTTP. Les en-têtes de réponse peuvent être utilisés à des fins de débogage, de sécurité et d'information. Vous pouvez spécifier des en-têtes dans le `customHttp.yml` fichier d'une application ou en le téléchargeant et en le modifiantAWS Management Console, puis en l'enregistrant dans le répertoire racine du projet. Pour connaître les procédures détaillées, consultez Définition d'en-têtes personnalisés.

Auparavant, les en-têtes HTTP personnalisés étaient spécifiés pour une application soit en modifiant la spécification de construction (buildspec) dans le, AWS Management Console soit en téléchargeant et en mettant à jour le `amplify.yml` fichier et en l'enregistrant dans le répertoire racine du projet. Les en-têtes personnalisés spécifiés de cette manière doivent être migrés hors du buildspec et du fichier. `amplify.yml` Pour des instructions, consultez Migration d'en-têtes personnalisés.

En-tête personnalisé au format YAML

Spécifiez des en-têtes personnalisés à l'aide du format YAML suivant :

```
customHeaders:
  - pattern: '*.json'
    headers:
    - key: 'custom-header-name-1'
      value: 'custom-header-value-1'
    - key: 'custom-header-name-2'
      value: 'custom-header-value-2'
  - pattern: '/path/*'
    headers:
    - key: 'custom-header-name-1'
      value: 'custom-header-value-2'
```

Pour un monorepo, utilisez le format YAML suivant :

```
applications:
  - appRoot: app1
    customHeaders:
    - pattern: '**/*'
      headers:
      - key: 'custom-header-name-1'
```

```
        value: 'custom-header-value-1'
  - appRoot: app2
    customHeaders:
    - pattern: '/path/*.json'
      headers:
      - key: 'custom-header-name-2'
        value: 'custom-header-value-2'
```

Lorsque vous ajoutez des en-têtes personnalisés à votre application, vous devez spécifier vos propres valeurs pour les éléments suivants :

pattern

Les en-têtes personnalisés sont appliqués à tous les chemins de fichiers URL qui correspondent au modèle.

headers

Définit les en-têtes qui correspondent au modèle de fichier.

key

Nom de l'en-tête personnalisé.

value

Valeur de l'en-tête personnalisé.

Pour en savoir plus sur les en-têtes HTTP, consultez la liste des en-têtes HTTP de Mozilla.

Définition d'en-têtes personnalisés

Il existe deux manières de spécifier des en-têtes HTTP personnalisés pour une AWS Amplify application. Vous pouvez spécifier des en-têtes dans le AWS Management Console ou vous pouvez spécifier des en-têtes en téléchargeant et en modifiant le `customHttp.yml` fichier d'une application, puis en l'enregistrant dans le répertoire racine de votre projet.

Pour définir des en-têtes personnalisés pour une application dans AWS Management Console

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.

3. Dans le volet de navigation, choisissez Paramètres de l'application, En-têtes personnalisés.
4. Dans la section Spécification de l'en-tête personnalisé, choisissez Modifier.
5. Dans la fenêtre d'édition, entrez les informations relatives à vos en-têtes personnalisés à l'aide du format YAML d'en-tête personnalisé.

   a. Pour`pattern`, entrez le modèle correspondant.

   b. Pour`key`, entrez le nom de l'en-tête personnalisé.

   c. Pour`value`, entrez la valeur de l'en-tête personnalisé.
6. Choisissez Save (Enregistrer).
7. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
   - Pour une application CI/CD, accédez à la branche à déployer et choisissez Redéployer cette version. Vous pouvez également créer une nouvelle version à partir de votre dépôt Git.
   - Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify.

Pour définir des en-têtes personnalisés à l'aide du fichier CustomHttp.yml

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Paramètres de l'application, En-têtes personnalisés.
4. Dans la section Spécification de l'en-tête personnalisé, choisissez Télécharger.
5. Ouvrez le `customHttp.yml` fichier téléchargé dans l'éditeur de code de votre choix et saisissez les informations relatives à vos en-têtes personnalisés à l'aide du format YAML d'en-tête personnalisé.

   a. Pour`pattern`, entrez le modèle correspondant.

   b. Pour`key`, entrez le nom de l'en-tête personnalisé.

   c. Pour`value`, entrez la valeur de l'en-tête personnalisé.
6. Enregistrez le `customHttp.yml` fichier modifié dans le répertoire racine de votre projet. Si vous travaillez avec un monorepo, enregistrez le `customHttp.yml` fichier à la racine de votre dépôt.
7. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
   - Pour une application CI/CD, effectuez une nouvelle compilation à partir de votre référentiel Git qui inclut le nouveau `customHttp.yml` fichier.

- Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify et incluez le nouveau `customHttp.yml` fichier avec les artefacts que vous chargez.

> **Note**
>
> Les en-têtes personnalisés définis dans le `customHttp.yml` fichier et déployés dans le répertoire racine de l'application remplaceront les en-têtes personnalisés définis dans la section En-têtes personnalisés du. AWS Management Console

Migration d'en-têtes personnalisés

Auparavant, les en-têtes HTTP personnalisés étaient spécifiés pour une application soit en modifiant le buildspec dans le, AWS Management Console soit en téléchargeant et en mettant à jour le `amplify.yml` fichier et en l'enregistrant dans le répertoire racine du projet. Il est fortement recommandé de migrer vos en-têtes personnalisés hors du buildspec et du fichier. `amplify.yml`

Spécifiez vos en-têtes personnalisés dans la section En-têtes personnalisés du AWS Management Console ou en téléchargeant et en modifiant le `customHttp.yml` fichier.

Pour migrer les en-têtes personnalisés stockés dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application sur laquelle effectuer la migration des en-têtes personnalisés.
3. Dans le volet de navigation, choisissez Paramètres de l'application, puis Paramètres de création. Dans la section Spécification de création de l'application, vous pouvez consulter les spécifications de construction de votre application.
4. Choisissez Télécharger pour enregistrer une copie de votre buildspec actuel. Vous pourrez consulter cette copie ultérieurement si vous avez besoin de récupérer des paramètres.
5. Lorsque le téléchargement est terminé, choisissez Modifier.
6. Prenez note des informations d'en-tête personnalisées figurant dans le fichier, car vous les utiliserez ultérieurement à l'étape 9. Dans la fenêtre d'édition, supprimez tous les en-têtes personnalisés du fichier et choisissez Enregistrer.
7. Dans le volet de navigation, choisissez Paramètres de l'application, En-têtes personnalisés.

8. Dans la section Spécification de l'en-tête personnalisé, choisissez Modifier.
9. Dans la fenêtre d'édition, entrez les informations relatives à vos en-têtes personnalisés que vous avez supprimés à l'étape 6.
10. Choisissez Save (Enregistrer).
11. Redéployez n'importe quelle branche à laquelle vous souhaitez appliquer les nouveaux en-têtes personnalisés.

Pour migrer des en-têtes personnalisés d'amplify.yml vers CustomHttp.yml

1. Accédez au `amplify.yml` fichier actuellement déployé dans le répertoire racine de votre application.
2. Ouvrez `amplify.yml` dans l'éditeur de code de votre choix.
3. Prenez note des informations d'en-tête personnalisées figurant dans le fichier, car vous les utiliserez ultérieurement à l'étape 8. Supprimez les en-têtes personnalisés du fichier. Enregistrez et fermez le fichier .
4. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
5. Choisissez l'application pour laquelle vous souhaitez définir des en-têtes personnalisés.
6. Dans le volet de navigation, choisissez Paramètres de l'application, En-têtes personnalisés.
7. Dans la section Spécification de l'en-tête personnalisé, choisissez Télécharger.
8. Ouvrez le `customHttp.yml` fichier téléchargé dans l'éditeur de code de votre choix et saisissez les informations relatives aux en-têtes personnalisés que vous avez supprimés `amplify.yml` à l'étape 3.
9. Enregistrez le `customHttp.yml` fichier modifié dans le répertoire racine de votre projet. Si vous travaillez avec un monorepo, enregistrez le fichier à la racine de votre dépôt.
10. Redéployez l'application pour appliquer les nouveaux en-têtes personnalisés.
    - Pour une application CI/CD, effectuez une nouvelle compilation à partir de votre référentiel Git qui inclut le nouveau `customHttp.yml` fichier.
    - Pour une application à déploiement manuel, déployez à nouveau l'application dans la console Amplify et incluez le nouveau `customHttp.yml` fichier contenant les artefacts que vous chargez.

> **Note**
>
> Les en-têtes personnalisés définis dans le `customHttp.yml` fichier et déployés dans le répertoire racine de l'application remplaceront les en-têtes personnalisés définis dans la section En-têtes personnalisés du. AWS Management Console

En-têtes personnalisés Monorepo

Lorsque vous spécifiez des en-têtes personnalisés pour une application dans un monorepo, tenez compte des exigences de configuration suivantes :

- Il existe un format YAML spécifique pour un monorepo. Pour connaître la syntaxe correcte, reportez-vous à la sectionEn-tête personnalisé au format YAML.
- Vous pouvez spécifier des en-têtes personnalisés pour une application dans un monorepo à l'aide de la section En-têtes personnalisés du. AWS Management Console Notez que vous devez redéployer votre application pour appliquer les nouveaux en-têtes personnalisés.
- Au lieu d'utiliser la console, vous pouvez spécifier des en-têtes personnalisés pour une application dans un monorepo dans un fichier. `customHttp.yml` Vous devez enregistrer le `customHttp.yml` fichier à la racine de votre référentiel, puis redéployer l'application pour appliquer les nouveaux en-têtes personnalisés. Les en-têtes personnalisés spécifiés dans le `customHttp.yml` fichier remplacent tous les en-têtes personnalisés spécifiés à l'aide de la section En-têtes personnalisés du. AWS Management Console

Exemple d'en-têtes de sécurité

Les en-têtes de sécurité personnalisés permettent d'appliquer le protocole HTTPS, de prévenir les attaques XSS et de protéger votre navigateur contre le clickjacking. Utilisez la syntaxe YAML suivante pour appliquer des en-têtes de sécurité personnalisés à votre application.

```
customHeaders:
  - pattern: '**'
    headers:
      - key: 'Strict-Transport-Security'
        value: 'max-age=31536000; includeSubDomains'
      - key: 'X-Frame-Options'
        value: 'SAMEORIGIN'
```

```
      - key: 'X-XSS-Protection'
        value: '1; mode=block'
      - key: 'X-Content-Type-Options'
        value: 'nosniff'
      - key: 'Content-Security-Policy'
        value: "default-src 'self'"
```

Exemple d'en-tête de contrôle du cache

Vous pouvez ajuster manuellement la `s-maxage` directive pour mieux contrôler les performances et la disponibilité du déploiement de votre application. Par exemple, pour augmenter la durée pendant laquelle votre contenu reste en cache à la périphérie, vous pouvez augmenter manuellement la durée de vie (TTL) en la mettant à jour `s-maxage` vers une valeur supérieure à la valeur par défaut de 600 secondes (10 minutes).

Pour spécifier une valeur personnalisée pour`s-maxage`, utilisez le format YAML suivant. Cet exemple montre comment conserver le contenu associé en cache à la périphérie pendant 3 600 secondes (une heure).

```
customHeaders:
  - pattern: '/img/*'
    headers:
      - key: 'Cache-Control'
        value: 's-maxage=3600'
```

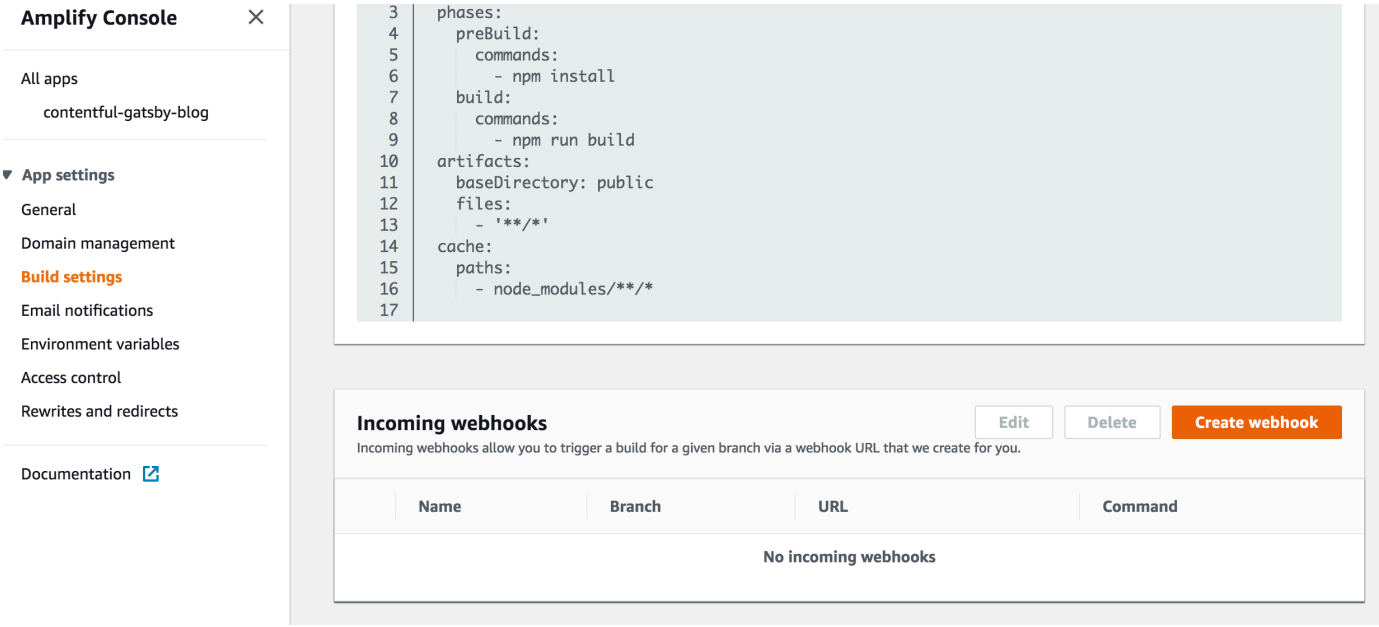
Pour plus d'informations sur le contrôle des performances des applications à l'aide des en-têtes, reportez-vous Utilisation d'en-têtes pour contrôler la durée du cache à la section.

Webhooks entrants

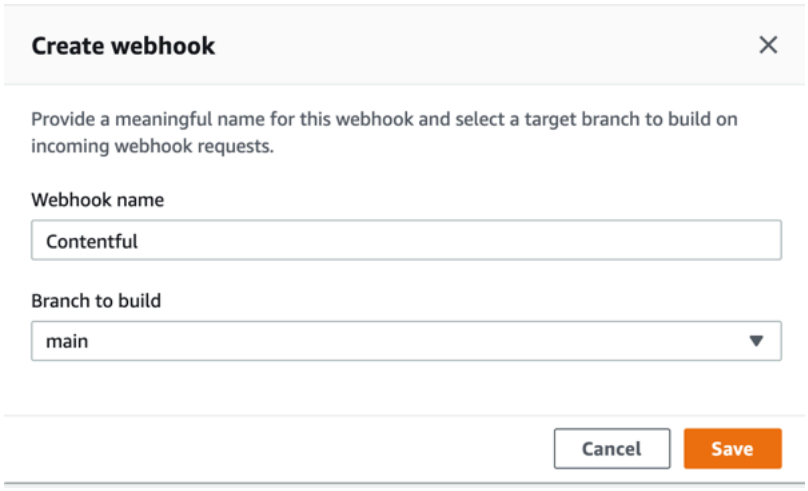
Configurez un webhook entrant dans la console Amplify pour déclencher une compilation sans envoyer de code dans votre dépôt Git. Vous pouvez utiliser des déclencheurs Webhook avec des outils CMS sans tête (tels que Contentful ou GraphCMS) pour démarrer un build chaque fois que le contenu change, ou pour effectuer des builds quotidiens à l'aide de services tels que Zapier.

Pour créer un webhook entrant

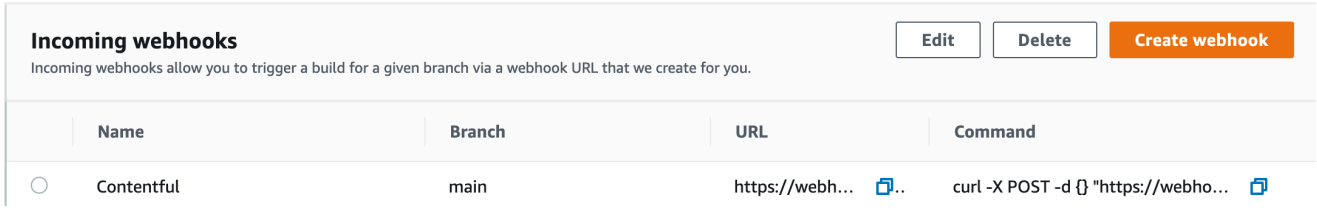
1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez créer un webhook.
3. Dans le volet de navigation, choisissez Paramètres de construction.
4. Sur la page des paramètres de création, faites défiler la page jusqu'à la section Webhooks entrants et choisissez Create Webhook.



5. Dans la boîte de dialogue Créer un webhook, procédez comme suit :
   a. Pour le nom du webhook, entrez le nom du webhook.
   b. Pour Branch to build, sélectionnez la branche à compiler sur les requêtes webhook entrantes.
   c. Choisissez Enregistrer.



6. Dans la section Webhooks entrants, effectuez l'une des opérations suivantes :
   - Copiez l'URL du webhook et fournissez-la à un outil CMS headless ou à un autre service pour déclencher les builds.
   - Exécutez la commande curl dans une fenêtre de terminal pour déclencher une nouvelle génération.



Surveillance

AWS Amplify émet des métriques via Amazon CloudWatch et fournit des journaux d'accès contenant des informations détaillées sur les demandes adressées à votre application. Consultez les rubriques de cette section pour savoir comment utiliser ces statistiques et journaux pour surveiller votre application.

Rubriques

Surveillance avec CloudWatch

AWS Amplify est intégré à Amazon CloudWatch, ce qui vous permet de surveiller les métriques de vos applications Amplify en temps quasi réel. Vous pouvez créer des alarmes qui envoient des notifications lorsqu'une métrique dépasse un seuil que vous avez défini. Pour plus d'informations sur le fonctionnement du CloudWatch service, consultez le guide de CloudWatch l'utilisateur Amazon.

Métriques

Amplify prend en charge six CloudWatch métriques dans l'espace de `AWS/AmplifyHosting` noms pour surveiller le trafic, les erreurs, le transfert de données et la latence de vos applications. Ces mesures sont agrégées à intervalles d'une minute. CloudWatch les indicateurs de surveillance sont gratuits et ne sont pas pris en compte dans les quotas CloudWatch de service.

Les statistiques disponibles ne sont pas toutes applicables à tous les indicateurs. Dans le tableau suivant, les statistiques les plus pertinentes sont répertoriées dans la description de chaque métrique.

Métriques	Description
Requêtes	Le nombre total de demandes de visiteurs reçues par votre application. La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total de demandes.

Métriques	Description
BytesDownloaded	La quantité totale de données transférées depuis votre application (téléchargées) en octets par les utilisateurs pour `GETHEAD`, et les `OPTIONS` demandes. La statistique la plus pertinente est`Sum`.
BytesUploaded	La quantité totale de données transférées dans votre application (téléchargées) en octets lors de l'utilisation `POST` et des `PUT` demandes. La statistique la plus pertinente est`Sum`.
4XXErrors	Nombre de requêtes ayant renvoyé une erreur dans la plage de codes d'état HTTP comprise entre 400 et 499. La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total d'occurrences de ces erreurs.
5XXErrors	Nombre de demandes ayant renvoyé une erreur dans la plage de codes d'état HTTP comprise entre 500 et 599. La statistique la plus pertinente est`Sum`. Utilisez les `Sum` statistiques pour obtenir le nombre total d'occurrences de ces erreurs.

Métriques	Description
Latence	Temps écoulé jusqu'au premier octet, en secondes. Il s'agit du délai total entre le moment où Amplify Hosting reçoit une demande et le moment où il renvoie une réponse au réseau. Cela n'inclut pas la latence réseau rencontrée pour qu'une réponse atteigne l'appareil du spectateur. Les statistiques les plus pertinentes sont `Average` `MaximumMinimum`,`p10`,`p50`,`p90`,`p95`, etp`100`. Utilisez les `Average` statistiques pour évaluer les latences attendues.

Amplify fournit les dimensions CloudWatch métriques suivantes.

Dimension	Description
Appli	Les données métriques sont fournies par l'application.
Compte AWS	Les données métriques sont fournies dans toutes les applications du Compte AWS.

Vous pouvez accéder aux CloudWatch métriques AWS Management Console à l'adresse https://console.aws.amazon.com/cloudwatch/. Vous pouvez également accéder aux métriques dans la console Amplify en suivant la procédure suivante.

Pour accéder aux métriques dans la console Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application dont vous souhaitez consulter les statistiques.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Surveillance.

4. Sur la page Surveillance, choisissez Metrics.

Alertes

Vous pouvez créer des CloudWatch alarmes dans la console Amplify qui envoient des notifications lorsque des critères spécifiques sont remplis. Une alarme surveille une seule CloudWatch métrique et envoie une notification Amazon Simple Notification Service lorsque la métrique dépasse le seuil pour un certain nombre de périodes d'évaluation.

Vous pouvez créer des alarmes plus avancées qui utilisent des expressions mathématiques métriques dans la CloudWatch console ou à l'aide CloudWatch des API. Par exemple, vous pouvez créer une alarme qui vous avertit lorsque le pourcentage 4XXErrors dépasse 15 % pendant trois périodes consécutives. Pour plus d'informations, consultez la section Création CloudWatch d'une alarme basée sur une expression mathématique métrique dans le guide de CloudWatch l'utilisateur Amazon.

La CloudWatch tarification standard s'applique aux alarmes. Pour plus d'informations, consultez les CloudWatchtarifs Amazon.

Utilisez la procédure suivante pour créer une alarme dans la console Amplify.

Pour créer une CloudWatch alarme pour une métrique Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application sur laquelle vous souhaitez activer une alarme.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Surveillance.
4. Sur la page Surveillance, sélectionnez Alarmes.
5. Sélectionnez Créer une alerte.
6. Dans la fenêtre Créer une alarme, configurez votre alarme comme suit :

    a. Pour Metric, choisissez le nom de la métrique à surveiller dans la liste.
    b. Dans Nom de l'alarme, entrez un nom significatif pour l'alarme. Par exemple, si vous surveillez des demandes, vous pouvez nommer l'alarme**`HighTraffic`**. Le nom ne doit contenir que des caractères ASCII.
    c. Pour configurer les notifications, effectuez l'une des opérations suivantes :
        - i. Choisissez Nouveau pour configurer une nouvelle rubrique Amazon SNS.
            ii. Dans Adresse e-mail, entrez l'adresse e-mail du destinataire des notifications.

iii. Choisissez Ajouter une nouvelle adresse e-mail pour ajouter des destinataires supplémentaires.

- i. Choisissez Existing pour réutiliser une rubrique Amazon SNS.
  ii. Pour le sujet SNS, sélectionnez le nom d'un sujet Amazon SNS existant dans la liste.

d. Pour Whenever the Statistic of Metric, définissez les conditions de votre alarme comme suit :

i. Spécifiez si la métrique doit être supérieure, inférieure ou égale à la valeur du seuil.
ii. Spécifiez la valeur de seuil.
iii. Spécifiez le nombre de périodes d'évaluation consécutives qui doivent être en état d'alarme pour déclencher l'alarme.
iv. Spécifiez la durée de la période d'évaluation.

e. Sélectionnez Créer une alerte.

> **Note**
>
> Chaque destinataire Amazon SNS que vous spécifiez reçoit un e-mail de confirmation de la part de AWS Notifications. L'e-mail contient un lien que le destinataire doit suivre pour confirmer son abonnement et recevoir des notifications.

Amazon CloudWatch Logs pour les applications SSR

Amplify envoie des informations sur votre environnement d'exécution Next.js à Amazon CloudWatch Logs dans votre. Compte AWSLorsque vous déployez une application SSR, celle-ci nécessite un rôle de service IAM qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Vous pouvez soit autoriser le calcul d'Amplify Hosting à créer automatiquement un rôle de service pour vous, soit spécifier un rôle que vous avez créé.

Si vous choisissez d'autoriser Amplify à créer un rôle IAM pour vous, le rôle sera déjà autorisé à créer des journaux. CloudWatch Si vous créez votre propre rôle IAM, vous devrez ajouter les autorisations suivantes à votre politique pour permettre à Amplify d'accéder à Amazon CloudWatch Logs.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
```

```
logs:PutLogEvents
```

Pour de plus amples informations sur les rôles de service, veuillez consulter Ajouter un rôle de service. Pour plus d'informations sur le déploiement d'applications rendues côté serveur, consultez. Déployez des applications rendues côté serveur avec Amplify Hosting

Journaux d'accès

Amplify stocke les journaux d'accès pour toutes les applications que vous hébergez dans Amplify. Les journaux d'accès contiennent des informations sur les demandes adressées à vos applications hébergées. Amplify conserve tous les journaux d'accès à une application jusqu'à ce que vous supprimiez l'application. Tous les journaux d'accès à une application sont disponibles dans la console Amplify. Cependant, chaque demande individuelle de journaux d'accès est limitée à une période de deux semaines que vous spécifiez.

Amplify ne réutilise jamais les CloudFront distributions entre clients. Amplify crée CloudFront des distributions à l'avance afin que vous n'ayez pas à attendre qu'une CloudFront distribution soit créée lorsque vous déployez une nouvelle application. Avant que ces distributions ne soient attribuées à une application Amplify, elles peuvent recevoir du trafic provenant de robots. Toutefois, ils sont configurés pour toujours répondre comme Introuvables avant d'être assignés. Si les journaux d'accès de votre application contiennent des entrées relatives à une période antérieure à la création de votre application, ces entrées sont liées à cette activité.

> **Important**
>
> Il est recommandé d'utiliser les journaux pour comprendre la nature des demandes de votre contenu, et non comme comptabilisation complète de toutes les demandes. Amplify fournit des journaux d'accès dans la mesure du possible. L'entrée du journal pour une demande particulière peut être fournie bien après le traitement réel de la demande et, dans de rares cas, une entrée du journal peut ne pas être fournie du tout. Lorsqu'une entrée de journal est omise des journaux d'accès, le nombre d'entrées dans les journaux d'accès ne correspond pas à l'utilisation indiquée dans les rapports AWS de facturation et d'utilisation.

Pour récupérer les journaux d'accès d'une application, procédez comme suit.

Pour consulter les journaux d'accès

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.

2. Choisissez l'application pour laquelle vous souhaitez consulter les journaux d'accès.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Surveillance.
4. Sur la page Surveillance, choisissez Access logs.
5. Choisissez Modifier la plage horaire.
6. Dans la fenêtre Modifier la plage horaire, pour Date de début, spécifiez le premier jour de l'intervalle de deux semaines pour lequel vous souhaitez récupérer les journaux. Pour Heure de début, choisissez l'heure du premier jour pour commencer la récupération du journal.
7. La console Amplify affiche les journaux pour la plage de temps spécifiée dans la section Journaux d'accès. Choisissez Télécharger pour enregistrer les journaux au format CSV.

Analyse des journaux d'accès

Pour analyser les journaux d'accès, vous pouvez stocker les fichiers CSV dans un compartiment Amazon S3. L'un des moyens d'analyser vos journaux d'accès consiste à utiliser Athena. Athena est un service de requêtes interactif qui peut vous aider à analyser les données pour AWS les services. Vous pouvez suivre les step-by-step instructions ici pour créer une table. Une fois votre table créée, vous pouvez interroger les données comme suit.

```
SELECT SUM(bytes) AS total_bytes
FROM logs
WHERE "date" BETWEEN DATE '2018-06-09' AND DATE '2018-06-11'
LIMIT 100;
```

Notifications

Vous pouvez configurer des notifications pour une AWS Amplify application afin d'alerter les parties prenantes ou les membres de l'équipe en cas de réussite ou d'échec d'une compilation. Amplify Hosting crée une rubrique Amazon Simple Notification Service (SNS) dans votre compte et l'utilise pour configurer les notifications par e-mail. Les notifications peuvent être configurées pour s'appliquer à toutes les branches ou à des branches spécifiques d'une application Amplify.

Notifications par e-mail

Utilisez les procédures suivantes pour configurer des notifications par e-mail pour toutes les branches ou pour des branches spécifiques d'une application Amplify.

Pour configurer les notifications par e-mail pour une application Amplify

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez configurer les notifications par e-mail.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Notifications, puis dans la section Notifications par e-mail, choisissez Ajouter une notification.
4. Procédez de l'une des manières suivantes dans la section Gérer les notifications :
   - Pour envoyer des notifications pour une seule succursale, pour E-mail, entrez l'adresse e-mail à laquelle envoyer les notifications. Pour Branche, sélectionnez le nom de la succursale pour laquelle vous souhaitez envoyer des notifications.
   - Pour envoyer des notifications à toutes les succursales connectées, pour E-mail, entrez l'adresse e-mail à laquelle envoyer les notifications. Pour Branche, choisissez Toutes les succursales.
5. Lorsque vous avez terminé, choisissez Save (Enregistrer)

Images de build personnalisées et mises à jour de packages en direct

Rubriques

Images de construction personnalisées

Vous pouvez utiliser une image de construction personnalisée pour fournir un environnement de génération personnalisé pour une application Amplify. Si vous avez des dépendances spécifiques dont l'installation prend du temps lors d'une compilation à l'aide du conteneur par défaut d'Amplify, vous pouvez créer votre propre image Docker et la référencer lors d'une compilation. Les images peuvent être hébergées sur Amazon Elastic Container Registry Public.

> **Note**
>
> Les paramètres de compilation ne sont visibles dans le menu des paramètres de l'application de la console Amplify que lorsqu'une application est configurée pour un déploiement continu et connectée à un référentiel git. Pour obtenir des instructions sur ce type de déploiement, voir Commencer avec le code existant.

Exigences relatives aux images de construction personnalisées

Pour qu'une image de construction personnalisée fonctionne comme une image de génération Amplify, elle doit répondre aux exigences suivantes :

1. Une distribution Linux qui supporte la bibliothèque GNU C (glibc), telle qu'Amazon Linux, compilée pour l'architecture x86-64.
2. cURL : lorsque vous lancez l'image personnalisée, l'exécuteur de build est téléchargé dans votre conteneur. C'est pourquoi il est nécessaire d'indiquer l'attribut cURL. Si cette dépendance est absente, le build échoue instantanément sans aucune sortie car notre build-runner n'est pas en mesure de produire de sortie.

3. Git : afin de pouvoir cloner le référentiel Git, Git doit être installé dans l'image. Si cette dépendance est absente, l'étape de clonage du référentiel échouera.
4. OpenSSH : afin de cloner votre dépôt en toute sécurité, nous avons besoin qu'OpenSSH configure temporairement la clé SSH pendant la compilation. Le package OpenSSH fournit les commandes dont le lanceur de compilation a besoin pour ce faire.
5. Bash et The Bourne Shell : ces deux utilitaires sont utilisés pour exécuter des commandes au moment de la construction. S'ils ne sont pas installés, vos builds risquent d'échouer avant de démarrer.
6. Node.js+npm : Notre lanceur de build n'installe pas Node. Il repose plutôt sur l'installation de Node et de NPM dans l'image. Cette exigence ne s'applique que pour les builds impliquant des packages NPM ou des commandes Node spécifiques. Cependant, nous vous recommandons vivement de les installer car lorsqu'ils sont présents, le lanceur de build Amplify peut utiliser ces outils pour améliorer l'exécution de la compilation. La fonction de remplacement de package d'Amplify utilise NPM pour installer le package Hugo-Extended lorsque vous définissez une dérogation pour Hugo.

Les packages suivants ne sont pas obligatoires, mais nous vous recommandons vivement de les installer.

1. NVM (Node Version Manager): Nous vous recommandons d'installer ce gestionnaire de version si vous devez gérer différentes versions deNode. Lorsque vous définissez une dérogation, la fonction de remplacement de package d'Amplify permet de modifier les versions NVM de Node.js avant chaque build.
2. Wget: Amplify peut utiliser l'Wgetutilitaire pour télécharger des fichiers pendant le processus de compilation. Nous vous recommandons de l'installer dans votre image personnalisée.
3. Tar : Amplify peut utiliser l'Tarutilitaire pour décompresser les fichiers téléchargés pendant le processus de compilation. Nous vous recommandons de l'installer dans votre image personnalisée.

Configuration d'une image de build personnalisée

Pour configurer une image de build personnalisée hébergée sur Amazon ECR

1. Consultez Getting started dans le guide de l'utilisateur Amazon ECR Public pour configurer un référentiel Amazon ECR Public avec une image Docker.

2. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
3. Choisissez l'application pour laquelle vous souhaitez configurer une image de build personnalisée.
4. Dans le volet de navigation, choisissez Paramètres de l'application, Paramètres de création.
5. Sur la page Paramètres de création, dans la section Paramètres de création d'image, choisissez Modifier.
6. Dans la boîte de dialogue Modifier les paramètres de l'image de construction, développez le menu Créer une image, puis choisissez Créer une image.
7. Entrez le nom du dépôt Amazon ECR Public que vous avez créé à la première étape. C'est ici que votre image de build est hébergée. Par exemple, si le nom de votre dépôt est ecr-examplerepo, vous devez entrer. **`public.ecr.aws/xxxxxxxx/ecr-examplerepo`**
8. Choisissez Enregistrer.

Mises à jour des packages en

Les mises à jour des packages en direct vous permettent de spécifier les versions des packages et les dépendances à utiliser dans l'image de compilation par défaut d'Amplify. L'image de construction par défaut est fournie avec plusieurs packages et dépendances préinstallés (par exemple Hugo, Amplify CLI, Yarn, etc.). Avec les mises à jour de packages en direct, vous pouvez remplacer la version de ces dépendances et spécifier une version spécifique ou vous assurer que la dernière version est toujours installée.

Si les mises à jour des packages en direct sont activées, avant l'exécution de votre build, le build runner met d'abord à jour (ou rétrograde) les dépendances spécifiées. Cela augmente le temps de création proportionnellement au temps nécessaire pour mettre à jour les dépendances, mais l'avantage est que vous pouvez vous assurer que la même version d'une dépendance est utilisée pour créer votre application.

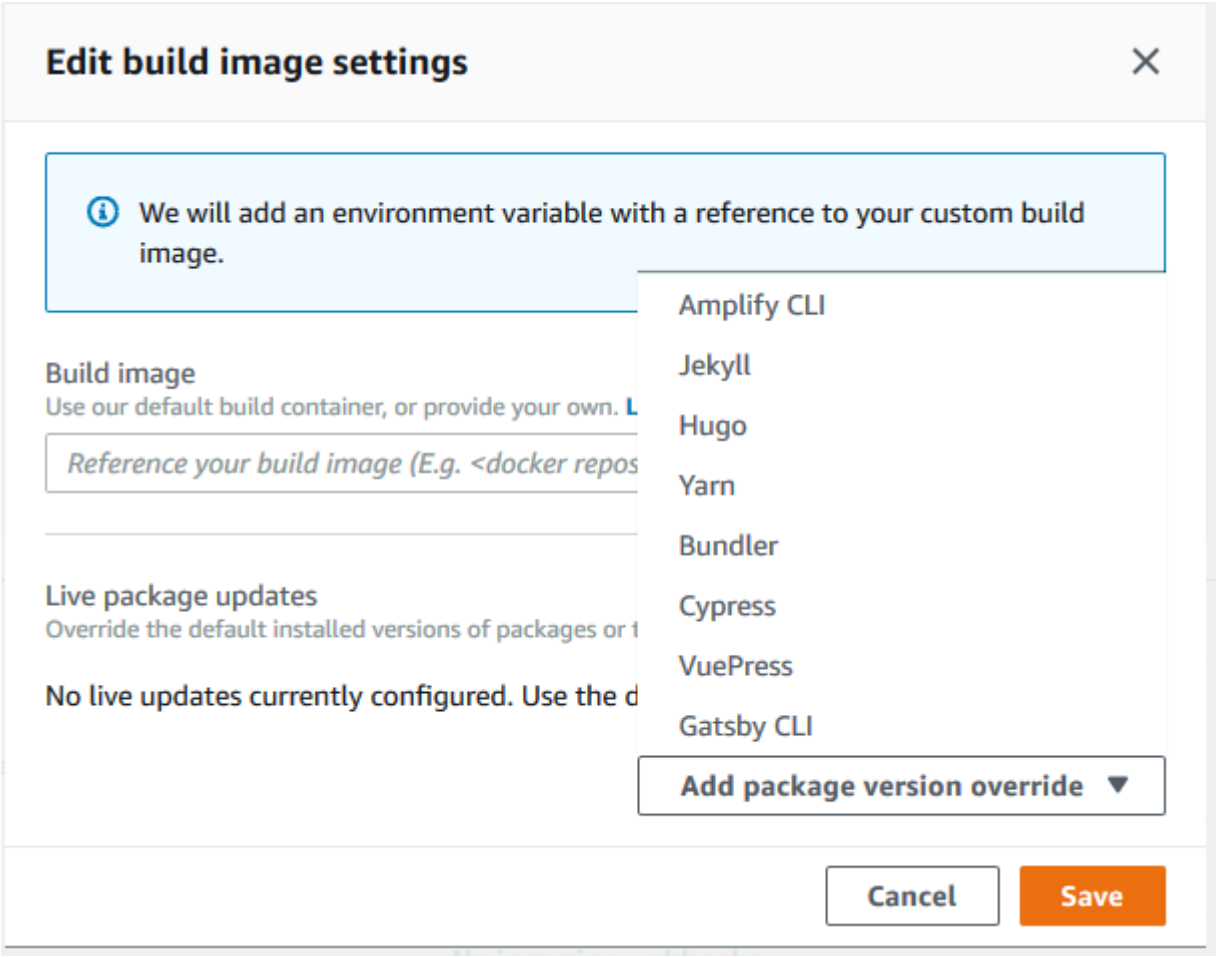
> **Warning**
>
> La configuration de la version de Node.js sur la dernière version entraîne l'échec des builds. Vous devez plutôt spécifier une version exacte de Node.js, telle que `1821.5`, ou`v0.1.2`.

Configuration des mises à jour de packages en direct

Pour configurer les mises à jour de packages en direct

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez configurer les mises à jour des packages en direct.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Paramètres de création.
4. Sur la page Paramètres de création, dans la section Paramètres de création d'image, choisissez Modifier.
5. Dans la boîte de dialogue Modifier les paramètres de l'image de construction, développez la liste de remplacement de la version du package et choisissez le package que vous souhaitez modifier.

   La capture d'écran suivante montre la boîte de dialogue Modifier les paramètres de l'image de construction avec la liste étendue de remplacement de la version d'ajout d'un package.



6. Pour Version, conservez la dernière version par défaut ou entrez une version spécifique de la dépendance. Si vous utilisez la dernière version, la dépendance sera toujours mise à niveau vers la dernière version disponible.
7. Choisissez Enregistrer.

Ajouter un rôle de service

Amplify a besoin d'autorisations pour déployer des ressources de backend avec votre front-end. Pour cela, vous utilisez un rôle de service. Un rôle de service est le rôle AWS Identity and Access Management (IAM) qu'Amplify assume lorsqu'il appelle d'autres services en votre nom. Dans ce guide, vous allez créer un rôle de service Amplify doté d'autorisations administratives de compte et autorisant explicitement un accès direct aux ressources dont les applications Amplify ont besoin pour déployer toutes les ressources Amplify Studio ou CLI, ainsi que pour créer et gérer des backends. Pour plus d'informations sur Amplify Studio, consultez la section Mise en route dans la documentation Amplify. Pour plus d'informations sur l'Amplify CLI, voir Amplify CLI dans la documentation Amplify.

Étape 1 : connectez-vous à la console IAM

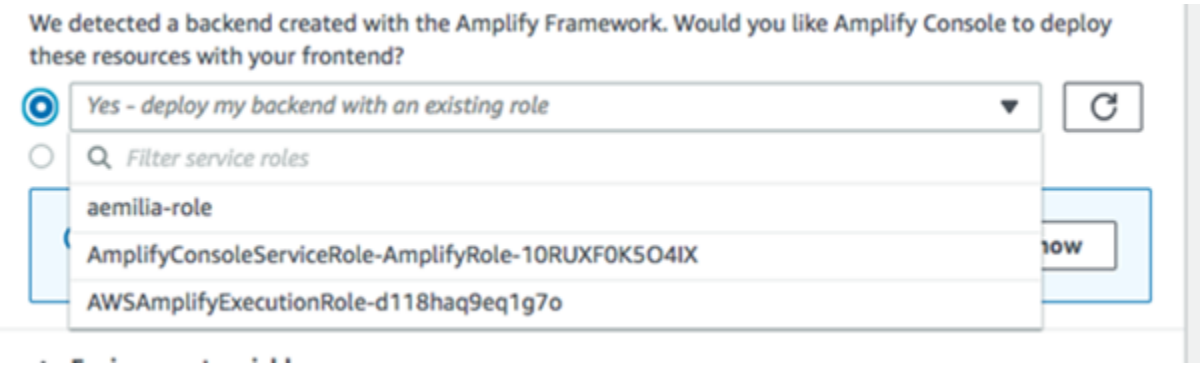
Ouvrez la console IAM et choisissez Rôles dans la barre de navigation de gauche, puis choisissez Créer un rôle.

Étape 2 : Création d'un rôle Amplify

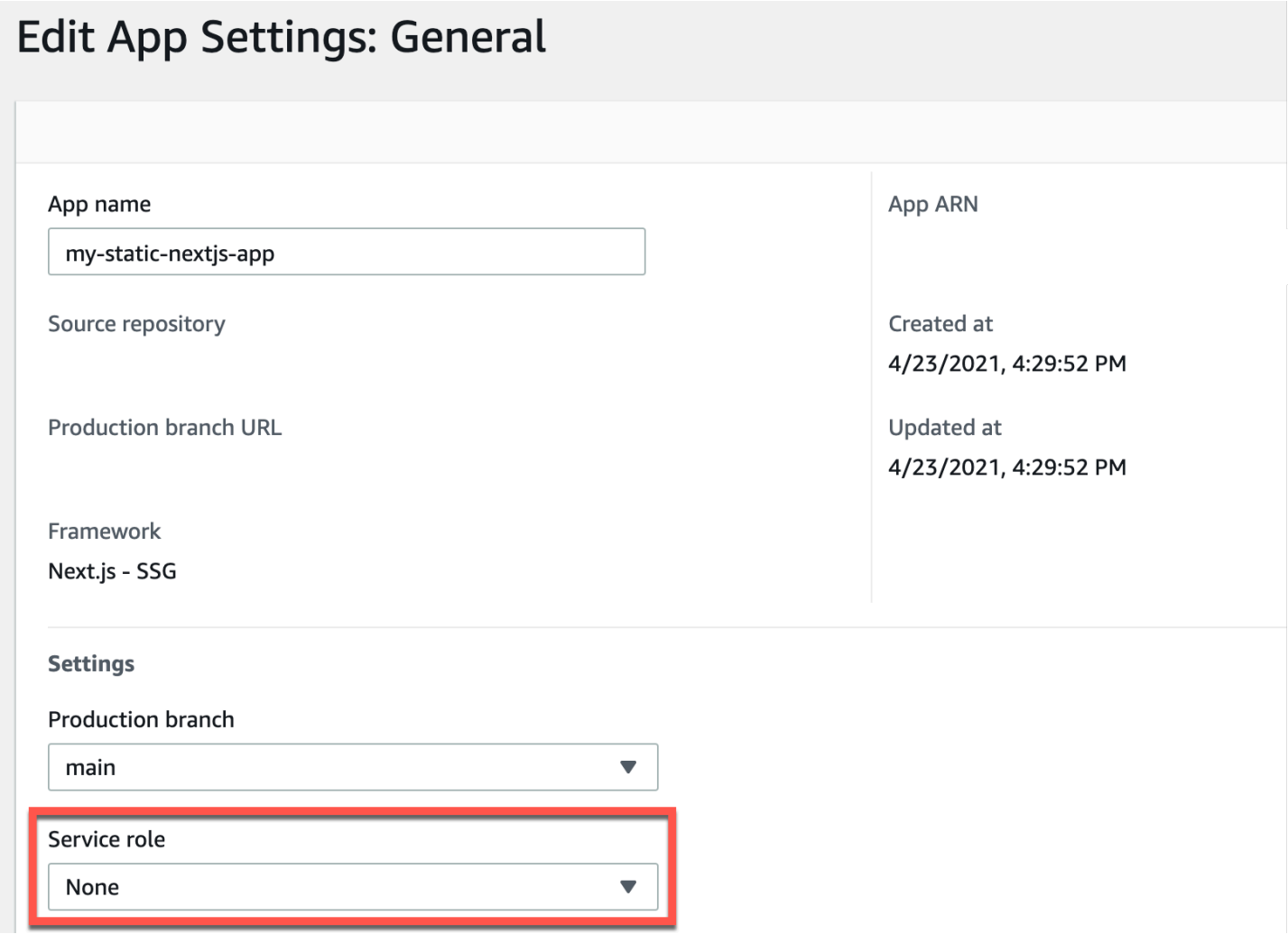
Dans l'écran de sélection des rôles, recherchez Amplify et choisissez le rôle Amplify-Backend Deployment. Acceptez toutes les valeurs par défaut et choisissez un nom pour votre rôle, tel que AmplifyConsoleServiceRole- AmplifyRole.

Étape 3 : Retournez à la console Amplify

Ouvrez la console Amplify. Si vous êtes en train de déployer une nouvelle application, choisissez Actualiser, puis choisissez le rôle que vous venez de créer. Cela devrait ressembler à AmplifyConsoleServiceRole- AmplifyRole.



Si vous possédez déjà une application, vous trouverez le paramètre du rôle de service dans App Settings > General (Paramètres de l'application > Général). À partir de là, choisissez Edit (Modifier) dans l'angle supérieur droit. Choisissez le rôle de service que vous venez de créer dans la liste déroulante, puis choisissez Save (Enregistrer).



La console Amplify est désormais autorisée à déployer les ressources du backend.

Prévention de l'adjoint confus

Le problème de l'adjoint confus est un problème de sécurité dans lequel une entité qui n'a pas l'autorisation d'effectuer une action peut contraindre une entité plus privilégiée à effectuer cette action. Pour plus d'informations, consultez Prévention du cas de figure de l'adjoint désorienté entre services.

Actuellement, la politique de confiance par défaut pour le rôle de `Amplify-Backend Deployment` service applique les clés de condition `aws:SourceArn` contextuelle et `aws:SourceAccount`

globale afin d'éviter toute confusion chez les adjoints. Toutefois, si vous avez déjà créé un `Amplify-Backend Deployment` rôle dans votre compte, vous pouvez mettre à jour la politique de confiance du rôle afin d'ajouter ces conditions afin de vous protéger contre la confusion chez les adjoints.

Utilisez l'exemple suivant pour restreindre l'accès aux applications de votre compte. Remplacez le texte rouge en italique dans l'exemple par vos propres informations.

```
"Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
```

Pour obtenir des instructions sur la modification de la politique de confiance pour un rôle à l'aide deAWS Management Console, consultez la section Modification d'un rôle (console) dans le guide de l'utilisateur IAM.

Gestion des performances des applications

L'architecture d'hébergement par défaut d'Amplify optimise l'équilibre entre les performances d'hébergement et la disponibilité du déploiement. Pour la plupart des clients, nous recommandons d'utiliser l'architecture par défaut.

Pour les clients expérimentés qui ont besoin d'un contrôle plus précis des performances d'une application, Amplify Hosting prend en charge le mode performance. Le mode performance optimise les performances d'hébergement en conservant le contenu en cache à la périphérie du réseau de diffusion de contenu (CDN) pendant un intervalle plus long. Lorsque le mode performance est activé, le déploiement et la disponibilité de la configuration de l'hébergement ou des modifications de code peuvent prendre jusqu'à 10 minutes. Pour plus d'informations, consultez the section called "Activer le mode performance".

Activer le mode performance

Utilisez la procédure suivante pour activer le mode performance pour une application déployée sur Amplify Hosting.

Pour activer le mode performance pour une application

1. Connectez-vous à la console Amplify AWS Management Console et ouvrez-la.
2. Choisissez l'application pour laquelle vous souhaitez activer le mode performance.
3. Dans le volet de navigation, choisissez Paramètres de l'application, Général.
4. Dans le volet Général, faites défiler la page jusqu'à la section Branches. Sélectionnez la branche pour laquelle vous souhaitez activer le mode performance.
5. Choisissez Action, puis Activer le mode performance.
6. Dans la boîte de dialogue Activer le mode performance, choisissez Activer le mode performance.

Utilisation d'en-têtes pour contrôler la durée du cache

L'`Cache-Control`en-tête `max-age` et les `s-maxage` directives HTTP affectent la durée de mise en cache du contenu de votre application. La `max-age` directive indique au navigateur pendant combien de temps (en secondes) vous souhaitez que le contenu reste dans le cache avant qu'il ne soit actualisé depuis le serveur d'origine. La `s-maxage` directive remplace `max-age` et vous

permet de spécifier la durée (en secondes) pendant laquelle vous souhaitez que le contenu reste sur la périphérie du CDN avant qu'il ne soit actualisé depuis le serveur d'origine. Les applications hébergées avec Amplify respectent et réutilisent les en-têtes de `Cache-Control` demande envoyés par les clients, sauf s'ils sont remplacés par un en-tête personnalisé que vous définissez.

Vous pouvez ajuster manuellement la `s-maxage` directive pour mieux contrôler les performances et la disponibilité du déploiement de votre application. Par exemple, pour augmenter la durée pendant laquelle votre contenu reste en cache à la périphérie, vous pouvez augmenter manuellement le temps de vie (TTL) en le mettant à jour `s-maxage` à une valeur supérieure à la valeur par défaut de 600 secondes (10 minutes).

> **Note**
>
> Lorsque le mode performance est activé pour une application, Amplify augmente le TTL maximal que vous pouvez définir pour l'application à l'aide d'un en-tête personnalisé, de 10 minutes (600 secondes) à un jour (86 400 secondes). Amplify plafonne `s-maxage` ce que vous pouvez définir à l'aide d'un en-tête personnalisé par jour. Par exemple, si vous définissez `s-maxage` une semaine (604 800 secondes), Amplify utilise le TTL maximum d'un jour.

Vous pouvez définir des en-têtes personnalisés pour une application dans la section En-têtes personnalisés de la console Amplify. Pour un exemple de YAML format, voirExemple d'en-tête de contrôle du cache.

Journalisation des appels d'API Amplify à l'aide d'AWS CloudTrail

AWS Amplifyest intégré àAWS CloudTrail, service qui enregistre les actions effectuées par un utilisateur, un rôle ou unAWS service dans Amplify. CloudTrail capture les appels d'API pour Amplify en tant qu'événements. Les appels capturés incluent des appels de la console Amplify et les appels de code vers les opérations d'API Amplify. Si vous créez un journal d'activité, vous pouvez activer la livraison continue d' CloudTrailévénements à un compartiment Amazon S3, y compris des événements pour Amplify. Si vous ne configurez pas de journal d'activité, vous pouvez toujours afficher les événements les plus récents dans la CloudTrail console dans Event history (Historique des événements). À l'aide des informations CloudTrail collectées, vous pouvez déterminer la demande qui a été envoyée à Amplify, l'adresse IP à partir de laquelle la demande a été effectuée, l'auteur de la demande, la date de la demande, ainsi que d'autres informations.

Pour en savoir plus CloudTrail, consultez le Guide deAWS CloudTrail l'utilisateur.

Amplify les informations dans CloudTrail

CloudTrail est activé par défaut sur votreAWS compte. Lorsqu'une activité a lieu dans Amplify, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements deAWS service dans Event history (Historique des événements). Vous pouvez afficher, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements dans le Guide deAWS CloudTrail l'utilisateur.

Pour un enregistrement continu des événements dans votreAWS compte, y compris les événements pour Amplify, créez un journal de suivi. Un journal CloudTrail de suivi permet de livrer des fichiers journaux dans 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'autresAWS services pour analyser plus en profondeur les données d'événement collectées dans les CloudTrail journaux et prendre les mesures nécessaires. Pour de plus amples informations, veuillez consulter les rubriques suivantes dans le Guide de l'utilisateur AWS CloudTrail :

- Créer un journal d'activité pour votre compte AWS

- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et Réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les opérations Amplify sont enregistrées CloudTrail et documentées dans la référence de l'API deAWS Amplify console, la référence de l'APIAWS Amplify Admin et la référence de l'API Amplify UI Builder. Par exemple, les appels à`DeleteApp` et les`CreateAppDeleteBackendEnvironment` opérations 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 :

- La demande a-t-elle été effectuée avec les informations d'identification utilisateur racine ouAWS Identity and Access Management (IAM).
- La demande a-t-elle été effectuée avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré.
- La demande a-t-elle été faite par un autreAWS service ?

Pour plus d'informations, consultez l'élémentCloudTrail userIdentity dans le Guide deAWS CloudTrail l'utilisateur.

Présentation Amplify de fichier journal

Un journal d'activité est une configuration qui permet d'envoyer des événements sous forme de fichiers journaux à un compartiment Simple Storage Service (Amazon S3) que vous spécifiez. CloudTrail Les fichiers journaux peuvent contenir une ou plusieurs entrées de journal. Un événement représente une demande individuelle émise à partir d'une source quelconque et comprend 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 série ordonnée retraçant les appels d'API publics. Ils ne suivent aucun ordre précis.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`ListApps`opération deAWS Amplify référence d'API de journal.

```
{
```

```
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-01-12T05:48:10Z"
            }
        }
    },
    "eventTime": "2021-01-12T06:47:29Z",
    "eventSource": "amplify.amazonaws.com",
    "eventName": "ListApps",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
 Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
 java/1.8.0_275 vendor/Oracle_Corporation",
    "requestParameters": {
        "maxResults": "100"
    },
    "responseElements": null,
    "requestID": "1c026d0b-3397-405a-95aa-aa43aexample",
    "eventID": "c5fca3fb-d148-4fa1-ba22-5fa63example",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "444455556666"
}
```


L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`ListBackendJobs`opération de référenceAWS Amplify d'API de l'interface utilisateur.


```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-01-13T00:47:25Z"
            }
        }
    },
    "eventTime": "2021-01-13T01:15:43Z",
    "eventSource": "amplifybackend.amazonaws.com",
    "eventName": "ListBackendJobs",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
java/1.8.0_275 vendor/Oracle_Corporation",
    "requestParameters": {
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging"
    },
    "responseElements": {
        "jobs": [
            {
                "appId": "d23mv2oexample",
                "backendEnvironmentName": "staging",
                "jobId": "ed63e9b2-dd1b-4bf2-895b-3d5dcexample",
                "operation": "CreateBackendAuth",
                "status": "COMPLETED",
                "createTime": "1610499932490",
                "updateTime": "1610500140053"
            },
            {
                "appId": "d23mv2oexample",
                "backendEnvironmentName": "staging",
                "jobId": "06904b10-a795-49c1-92b7-185dfexample",
                "operation": "CreateBackend",
```

```
                "status": "COMPLETED",
                "createTime": "1610499657938",
                "updateTime": "1610499704458"
            }
        ],
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging"
    },
    "requestID": "7adfabd6-98d5-4b11-bd39-c7deaexample",
    "eventID": "68769310-c96c-4789-a6bb-68b52example",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "444455556666"
}
```

Sécurité dans Amplify

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez de centres de données et d'architectures réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cela comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à AWS Amplify, voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amplify. Les rubriques suivantes vous montrent comment configurer Amplify pour atteindre vos objectifs de sécurité et de conformité. Vous apprenez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources Amplify.

Rubriques

Identity and Access Management pour Amplify

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 Amplify. 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 Amplify.

Utilisateur du service : si vous utilisez le service Amplify 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 d'Amplify 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 pouvez pas accéder à une fonctionnalité dans Amplify, consultez. Résolution des problèmes Amplify identity and access

Administrateur du service — Si vous êtes responsable des ressources Amplify dans votre entreprise, vous avez probablement un accès complet à Amplify. C'est à vous de déterminer les fonctionnalités et ressources d'Amplify auxquelles les utilisateurs de votre 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 IAM avec Amplify, consultez. Comment Amplify fonctionne 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 Amplify. Pour voir des exemples de politiques basées sur l'identité Amplify que vous pouvez utiliser dans IAM, consultez. Exemples de politiques basées sur l'identité pour Amplify

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 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 Groupes d'utilisateurs IAM 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 des autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur IAM 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 Amplify fonctionne avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amplify, découvrez quelles fonctionnalités IAM peuvent être utilisées avec Amplify.

Fonctionnalités IAM que vous pouvez utiliser avec Amplify

Fonction IAM	Amplifier le support
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 (identifications dans les politiques)	Partielle
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Fonctions de service	Oui
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont Amplify 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 Amplify

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 Groupes d'utilisateurs IAM 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 Amplify

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Politiques basées sur les ressources dans Amplify

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 politiques pour Amplify

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 une liste des actions Amplify, voir Actions définies par AWS Amplify dans la référence d'autorisation de service.

Les actions politiques dans Amplify utilisent le préfixe suivant avant l'action :

```
amplify
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "amplify:action1",
      "amplify:action2"
         ]
```

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Ressources politiques pour Amplify

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": "*"
```

Pour une liste des types de ressources Amplify et de leurs ARN, voir Types de ressources définis par AWS Amplify dans la référence d'autorisation de service. Pour savoir grâce à quelles actions vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par AWS Amplify.

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Clés de conditions de politique pour Amplify

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.

Pour une liste des clés de condition Amplify, voir Clés de condition pour AWS Amplify dans la référence d'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez la section Actions définies par AWS Amplify.

Pour consulter des exemples de politiques basées sur l'identité Amplify, consultez. Exemples de politiques basées sur l'identité pour Amplify

Listes de contrôle d'accès (ACL) dans Amplify

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 Amplify

Prise en charge d'ABAC (identifications dans les politiques)	Partielle

Le contrôle d'accès basé sur les attributs (ABAC) est une politique 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 étiquettes, vous devez fournir les informations d'étiquette 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 avec Amplify

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.

Transférer les sessions d'accès pour Amplify

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.

Rôles de service pour Amplify

Prend en charge les fonctions de service	Oui

Une fonction du 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.

> **Warning**
>
> La modification des autorisations pour un rôle de service peut interrompre les fonctionnalités d'Amplify. Modifiez les rôles de service uniquement lorsque Amplify fournit des instructions à cet effet.

Rôles liés à un service pour Amplify

Prend en charge les rôles liés à un service	Non

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 à un service, consultez la section AWS Services compatibles avec IAM dans le Guide de l'utilisateur d'IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Cliquez sur le lien Oui pour consulter la documentation relative aux rôles liés à un service pour ce service.

Exemples de politiques basées sur l'identité pour Amplify

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier des ressources Amplify. 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 doit 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 Amplify, y compris le format des ARN pour chacun des types de ressources, voir Actions, ressources et clés de condition AWS Amplify dans la référence d'autorisation de service.

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console Amplify
- 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 Amplify 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 de plus amples 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 de plus amples 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 de plus amples 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 Amplify

Pour accéder à la AWS Amplify console, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et d'afficher les détails des ressources Amplify 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.

Avec la sortie d'Amplify Studio, la suppression d'une application ou d'un backend nécessite à la fois `amplify` des autorisations et des autorisations. `amplifybackend` Si une politique IAM fournit uniquement des `amplify` autorisations, un utilisateur reçoit une erreur d'autorisation lorsqu'il tente de supprimer une application. Si vous êtes un administrateur qui rédige des politiques, déterminez les autorisations appropriées à accorder aux utilisateurs qui doivent effectuer des actions de suppression.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console Amplify, associez également la politique Amplify `ConsoleAccess` ou `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": "*"
        }
    ]
}
```

AWS politiques gérées pour AWS Amplify

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez la section Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

Politique gérée par AWS : AdministratorAccess -Amplify

Vous pouvez associer la politique `AdministratorAccess-Amplify` à vos identités IAM. Amplify associe également cette politique à un rôle de service qui permet à Amplify d'effectuer des actions en votre nom.

Lorsque vous déployez un backend dans la console Amplify, vous devez créer `Amplify-Backend Deployment` un rôle de service qu'Amplify utilise pour créer et gérer des ressources. AWS IAM associe la politique `AdministratorAccess-Amplify` gérée au rôle de `Amplify-Backend Deployment` service.

Cette politique accorde des autorisations administratives aux comptes tout en autorisant explicitement l'accès direct aux ressources dont les applications Amplify ont besoin pour créer et gérer les backends.

Détails de l'autorisation

Cette politique donne accès à plusieurs AWS services, y compris aux actions IAM. Ces actions permettent aux identités soumises à cette politique d'utiliser AWS Identity and Access

Management pour créer d'autres identités avec n'importe quelle autorisation. Cela permet une escalade des autorisations et cette politique doit être considérée comme aussi puissante que la `AdministratorAccess` politique.

Cette politique accorde l'autorisation `iam:PassRole` d'action pour toutes les ressources. Cela est nécessaire pour prendre en charge la configuration des groupes d'utilisateurs Amazon Cognito.

Pour consulter les autorisations associées à cette politique, voir AdministratorAccess-Amplify dans le manuel AWS Managed Policy Reference.

AWS politique gérée : AmplifyBackendDeployFullAccess

Vous pouvez associer la politique `AmplifyBackendDeployFullAccess` à vos identités IAM.

Cette politique accorde à Amplify des autorisations d'accès complètes pour déployer les ressources principales d'Amplify (Amazon AWS AppSync Cognito, Amazon S3 et autres services connexes) via le. AWS Cloud Development Kit (AWS CDK) Les autorisations sont reportées aux AWS CDK rôles dotés des autorisations `AdministratorAccess` politiques nécessaires.

Pour consulter les autorisations associées à cette politique, reportez-vous AmplifyBackendDeployFullAccessà la référence des politiques AWS gérées.

Amplifier les mises à jour des politiques gérées AWS

Consultez les détails des mises à jour des politiques AWS gérées pour Amplify depuis que ce service a commencé à suivre ces modifications. Pour obtenir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS de la page Historique du document pour AWS Amplify.

Modification	Description	Date
AmplifyBackendDeployFullAccess – Mise à jour d'une stratégie existante	Ajoutez l'action `cloudformation:DeleteStack` politique pour prendre en charge la suppression de la pile lorsque l'`DeleteBranch` API est appelée.	5 avril 2024

Modification	Description	Date
	Ajoutez l'action de `lambda:GetFunction` politique pour prendre en charge les fonctions de hotswapping. Ajoutez l'action de `lambda:UpdateFunctionConfiguration` politique pour prendre en charge les mises à jour de la fonction Lambda.	
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez les `cloudformation:UnTagResource` autorisations `cloudformation:TagResource` et pour prendre en charge les appels aux AWS CloudFormation API.	4 avril 2024
AmplifyBackendDeployFullAccess - mise à jour d'une politique existante	Ajoutez l'action `lambda:InvokeFunction` politique pour prendre en charge le AWS Cloud Development Kit (AWS CDK) hotswapping. Il AWS CDK fait des appels directs à une fonction Lambda pour effectuer le hotswapping des actifs Amazon S3. Ajoutez l'action de `lambda:UpdateFunctionCode` politique pour prendre en charge les fonctions de hotswapping.	2 janvier 2024

Modification	Description	Date
AmplifyBackendDeployFullAccess - mise à jour d'une politique existante	Ajoutez des actions politiques pour soutenir l'`UpdateApiKey` opération. Cela est nécessaire pour permettre le déploiement réussi de l'application après avoir quitté et redémarré le sandbox sans supprimer de ressources.	17 novembre 2023
AmplifyBackendDeployFullAccess - mise à jour d'une politique existante	Ajoutez l'`amplify:GetBackendEnvironment` autorisation de prendre en charge le déploiement de l'application Amplify.	6 novembre 2023
AmplifyBackendDeployFullAccess – Nouvelle politique	Amplify a ajouté une nouvelle politique avec les autorisations minimales requises pour déployer les ressources du backend Amplify.	8 octobre 2023
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez l'`ecr:DescribeRepositories` autorisation requise par l'interface de ligne de commande (CLI) Amplify.	1er juin 2023

Modification	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action politique pour soutenir la suppressi on des balises d'une AWS AppSync ressource. Ajoutez une action politique pour soutenir la ressource Amazon Polly. Ajoutez une action politique pour prendre en charge la mise à jour de la configuration du OpenSearch domaine. Ajoutez une action politique pour soutenir la suppression des balises d'un AWS Identity and Access Management rôle. Ajoutez une action politique pour prendre en charge la suppression de balises d'une ressource Amazon DynamoDB. Ajoutez les `cloudfron t:GetCloudFrontOri ginAccessIdentityC onfig` autorisations `cloudfront:GetClou dFrontOriginAccess Identity` et au bloc d'instructions pour prendre `CLISDKCalls` en charge les flux de travail de publication et d'hébergement d'Amplify.	24 février 2023

Modification	Description	Date
	Ajoutez l'`s3:PutBucketPublicAccessBlock` autorisation au `CLIManageviaCFNPolicy` bloc d'instructions pour permettre à Amazon S3 de AWS CLI respecter les meilleures pratiques de sécurité d'Amazon S3, qui consistent à activer la fonctionnalité Amazon S3 Block Public Access sur les compartiments internes. Ajoutez l'`cloudformation:DescribeStacks` autorisation au bloc d'instructions pour `CLISDKCalls` permettre de récupérer les AWS CloudFormation piles des clients lors de nouvelles tentatives dans le processeur principal Amplify afin d'éviter de dupliquer les exécutions en cas de mise à jour d'une pile. Ajoutez l'`cloudformation:ListStacks` autorisation au bloc de `CLICloudformationPolicy` déclarations. Cette autorisation est requise pour prendre pleinement en charge	

Modification	Description	Date
	l' CloudFormation DescribeS tacks action.	
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politiques pour permettre à la fonctionn alité de rendu côté serveur Amplify de transférer les métriques de l'application vers celles du client CloudWatch . Compte AWS	30 août 2022
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politiques pour bloquer l'accès public au compartiment Amazon S3 du déploiement d'Amplify.	27 avril 2022

Modification	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action pour permettre aux clients de supprimer leurs applications de rendu côté serveur (SSR). Cela permet également de supprimer correctement la CloudFront distribution correspondante. Ajoutez une action pour permettre aux clients de spécifier une fonction Lambda différente pour gérer les événements provenant d'une source d'événements existante à l'aide de la CLI Amplify. Avec ces modifications, AWS Lambda vous serez en mesure d'effectuer l'UpdateEventSourceMappingaction.	17 avril 2022
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez une action de politique pour activer les actions Amplify UI Builder sur toutes les ressources.	2 décembre 2021

Modification	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions politiques pour soutenir la fonctionnalité d'authentification Amazon Cognito qui utilise des fournisseurs d'identité sociale. Ajoutez une action de politique pour prendre en charge les couches Lambda. Ajoutez une action politique pour soutenir la catégorie Amplify Storage.	8 novembre 2021

Modification	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Ajoutez des actions Amazon Lex pour soutenir la catégorie Amplify Interactions. Ajoutez des actions Amazon Rekognition pour soutenir la catégorie Amplify Predictions. Ajoutez une action Amazon Cognito pour prendre en charge la configuration MFA sur les groupes d'utilisateurs Amazon Cognito. Ajoutez CloudFormation des actions au support AWS CloudFormation StackSets. Ajoutez des actions Amazon Location Service pour soutenir la catégorie Amplify Geo. Ajoutez une action Lambda pour prendre en charge les couches Lambda dans Amplify. Ajoutez des actions de CloudWatch journalisation pour prendre en charge CloudWatch les événements. Ajoutez des actions Amazon S3 pour soutenir la catégorie Amplify Storage.	27 septembre 2021

Modification	Description	Date
	Ajoutez des actions politiques pour prendre en charge les applications de rendu côté serveur (SSR).	

Modification	Description	Date
AdministratorAccess-Amplify — Mise à jour d'une politique existante	Consolidez toutes les actions Amplify en une seule `amplify:*` action. Ajoutez une action Amazon S3 pour prendre en charge le chiffrement des compartiments Amazon S3 des clients. Ajoutez des actions de limite d'autorisation IAM pour prendre en charge les applications Amplify dont les limites d'autorisation sont activées. Ajoutez des actions Amazon SNS pour faciliter l'affichage des numéros de téléphone d'origine et l'affichage, la création, la vérification et la suppression des numéros de téléphone de destination. Amplify Studio : ajoutez Amazon Cognito AWS Lambda, IAM et des actions de politique pour permettre la gestion des backends dans la console Amplify AWS CloudFormation et Amplify Studio. Ajoutez une déclaration de politique AWS Systems Manager (SSM) pour gérer	28 juillet 2021

Modification	Description	Date
	les secrets de l'environnement Amplify. Ajoutez une AWS CloudFormation `ListResources` action pour prendre en charge les couches Lambda pour les applications Amplify.	
Amplify a commencé à suivre les modifications	Amplify a commencé à suivre les modifications apportées à ses politiques AWS gérées.	28 juillet 2021

Résolution des problèmes Amplify identity and access

Utilisez les informations suivantes pour vous aider à diagnostiquer et à résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation d'Amplify et d'IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Amplify
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures à mon AWS compte à accéder à mes ressources Amplify

Je ne suis pas autorisé à effectuer une action dans Amplify

Si vous recevez une erreur qui indique que vous n'êtes pas autorisé à effectuer une action, vos politiques doivent être mises à jour afin de vous permettre d'effectuer l'action.

L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `amplify:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 amplify:GetWidget on resource: my-example-widget
```

Dans ce cas, la politique qui s'applique à l'utilisateur `mateojackson` doit être mise à jour pour autoriser l'accès à la ressource *`my-example-widget`* à l'aide de l'action `amplify:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Avec la sortie d'Amplify Studio, la suppression d'une application ou d'un backend nécessite à la fois `amplify` des autorisations et des autorisations. `amplifybackend` Si un administrateur a rédigé une politique IAM qui fournit uniquement des `amplify` autorisations, vous recevrez une erreur d'autorisation lorsque vous tenterez de supprimer une application.

L'exemple d'erreur suivant se produit lorsque l'utilisateur `mateojackson` IAM essaie d'utiliser la console pour supprimer une *`example-amplify-app`* ressource fictive mais ne dispose pas des `amplifybackend:`*`RemoveAllBackends`* autorisations nécessaires.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 amplifybackend;:RemoveAllBackends on resource: example-amplify-app
```

Dans ce cas, Mateo demande à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource *`example-amplify-app`* à l'aide de l'action `amplifybackend:`*`RemoveAllBackends`*.

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 transmettre un rôle à Amplify.

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 Amplify. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle 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 autoriser des personnes extérieures à mon AWS compte à accéder à mes ressources Amplify

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 Amplify prend en charge ces fonctionnalités, consultez. Comment Amplify fonctionne 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 intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

Protection des données dans Amplify

AWS Amplify est conforme au modèle de responsabilité AWS partagée modèle de de , qui inclut des réglementations et des directives pour la protection des données. AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS services. AWS conserve le contrôle des données

hébergées sur cette infrastructure, y compris les contrôles de configuration de sécurité pour le traitement du contenu client et des données personnelles. AWS les clients et les partenaires APN, agissant en tant que contrôleurs ou sous-traitants de données, sont responsables de toutes les données personnelles qu'ils placent dans le AWS Cloud.

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
- 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 au sein AWS des services.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données personnelles stockées dans Amazon S3.

Nous vous recommandons vivement de ne jamais placer d'informations identifiables sensibles, telles que les numéros de compte de vos clients, dans des champs de formulaire comme Nom. Cela inclut lorsque vous travaillez avec Amplify ou d'autres AWS services à l'aide de la console, de l'API ou AWS des AWS CLI SDK. Toutes les données que vous entrez dans Amplify ou dans d'autres services peuvent être récupérées pour être incluses dans les journaux de diagnostic. Lorsque vous fournissez une URL à un serveur externe, n'incluez pas les informations d'identification non chiffrées dans l'URL pour valider votre demande adressée au serveur.

Pour en savoir plus sur la protection des données, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD sur le Blog sur la sécurité d'AWS .

Chiffrement au repos

Le chiffrement au repos consiste à protéger vos données contre tout accès non autorisé en chiffrant les données stockées. Amplify chiffre les artefacts de construction d'une application par défaut en utilisant pour Amazon AWS KMS keys S3 qui sont gérés par le. AWS Key Management Service

Amplify utilise Amazon CloudFront pour proposer votre application à vos clients. CloudFront utilise des SSD chiffrés pour les points de présence de localisation périphériques (POP) et des volumes

EBS chiffrés pour les caches périphériques régionaux (REC). Le code de fonction et la configuration dans CloudFront Functions sont toujours stockés dans un format crypté sur les SSD chiffrés sur les POP des emplacements périphériques et dans les autres emplacements de stockage utilisés par. CloudFront

Chiffrement en transit

Le chiffrement en transit consiste à protéger vos données contre l'interception pendant qu'elles se déplacent entre les points de terminaison de communication. Amplify Hosting fournit le cryptage des données en transit par défaut. Toutes les communications entre les clients et Amplify et entre Amplify et ses dépendances en aval sont protégées par des connexions TLS signées à l'aide du processus de signature Signature Version 4. Tous les points de terminaison Amplify Hosting utilisent des certificats SHA-256 gérés par une autorité de certification privée. AWS Certificate Manager Pour de plus amples informations, veuillez consulter Processus de signature Signature Version 4 et Présentation d'ACM PCA.

Gestion des clés de chiffrement

AWS Key Management Service (KMS) est un service géré permettant de créer et de contrôler AWS KMS keys les clés de chiffrement utilisées pour chiffrer les données des clients. AWS Amplify génère et gère des clés cryptographiques pour chiffrer les données pour le compte des clients. Il n'y a aucune clé de chiffrement à gérer.

Validation de conformité pour AWS Amplify

Des auditeurs tiers évaluent la sécurité et AWS Amplify la conformité de plusieurs programmes de AWS conformité. Il s'agit notamment du SOC, du PCI, de l'ISO, de l'HIPAA, du MTCS, du C5, du K-ISMS, de l'ENS High, de l'OSPAR, du HITRUST CSF et de la FINMA.

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et

réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur AWS la sécurité et la conformité.
- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : ce livre blanc décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.

> **Note**
>
> Tous ne Services AWS sont pas éligibles à la loi HIPAA. Pour plus d'informations, consultez le HIPAA Eligible Services Reference.

- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Guides de conformité destinés aux clients — Comprenez le modèle de responsabilité partagée sous l'angle de la conformité. Les guides résument les meilleures pratiques en matière de sécurisation Services AWS et décrivent les directives relatives aux contrôles de sécurité dans de nombreux cadres (notamment le National Institute of Standards and Technology (NIST), le Payment Card Industry Security Standards Council (PCI) et l'Organisation internationale de normalisation (ISO)).
- Évaluation des ressources à l'aide des règles du guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.
- AWS Audit Manager— Cela vous Service AWS permet d'auditer en permanence votre AWS utilisation afin de simplifier la gestion des risques et la conformité aux réglementations et aux normes du secteur.

Sécurité de l'infrastructure dans AWS Amplify

En tant que service géré, AWS Amplify il est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à Amplify via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et 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.

Enregistrement et surveillance des événements de sécurité dans Amplify

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amplify et de vos autres AWS solutions. AWS fournit les outils de surveillance suivants pour surveiller Amplify, signaler un problème et prendre des mesures automatiques le cas échéant :

- Amazon CloudWatch surveille en temps réel vos AWS ressources et les applications que vous utilisez AWS. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés et définir des alarmes qui vous avertissent ou prennent des mesures lorsqu'une certaine métrique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos instances Amazon Elastic Compute Cloud (Amazon EC2) et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations sur l'utilisation CloudWatch des métriques et des alarmes avec Amplify, consultez. Surveillance

- 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. AWS 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 AWS compte de votre compte et transmet les fichiers journaux à un compartiment Amazon Simple Storage Service (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 plus d'informations, consultez Journalisation des appels d'API Amplify à l'aide d'AWS CloudTrail.
- Amazon EventBridge est un service de bus d'événements sans serveur qui permet de connecter facilement vos applications à des données provenant de diverses sources. EventBridge fournit un flux de données en temps réel à partir de vos propres applications, applications oftware-as-a S-Service (SaaS) et AWS services, et achemine ces données vers des cibles telles que AWS Lambda. Cela vous permet de surveiller les événements qui se produisent dans les services et de créer des architectures axées sur les événements. Pour plus d'informations, consultez le guide de EventBridge l'utilisateur Amazon.

Prévention du cas de figure de l'adjoint désorienté entre services

Le problème de l'adjoint confus est un problème de sécurité dans lequel une entité qui n'a pas l'autorisation d'effectuer une action peut contraindre une entité plus privilégiée à effectuer cette action. En AWS, l'usurpation d'identité interservices peut entraîner la confusion des adjoints. L'usurpation d'identité entre services peut se produire lorsqu'un service (le service appelant) appelle un autre service (le service appelé). Le service appelant peut être manipulé et ses autorisations utilisées pour agir sur les ressources d'un autre client auxquelles on ne serait pas autorisé d'accéder autrement. Pour éviter cela, AWS fournit des outils qui vous aident à protéger vos données pour tous les services avec des principaux de service qui ont eu accès aux ressources de votre compte.

Nous recommandons d'utiliser les clés de contexte de condition `aws:SourceAccount`globale `aws:SourceArn`et les clés contextuelles dans les politiques de ressources afin de limiter les autorisations qui AWS Amplify accordent un autre service à la ressource. Si vous utilisez les deux clés de contexte de condition globale, la valeur `aws:SourceAccount` et le compte de la valeur

`aws:SourceArn` doit utiliser le même ID de compte lorsqu'il est utilisé dans la même déclaration de stratégie.

La valeur de `aws:SourceArn` doit être l'ARN de branche de l'application Amplify. Spécifiez cette valeur dans le format`arn:`*Partition*`:amplify:`*Region*`:`*Account*`:apps/`*AppId*`/branches/`*BranchName*.

Le moyen le plus efficace de se protéger contre le problème de député confus consiste à utiliser la clé de contexte de condition globale `aws:SourceArn` avec l'ARN complet de la ressource. Si vous ne connaissez pas l'ARN complet de la ressource ou si vous spécifiez plusieurs ressources, utilisez la clé de contexte de condition globale `aws:SourceArn` avec des caractères génériques (*) pour les parties inconnues de l'ARN. Par exemple, `arn:aws:`*servicename*`::`*123456789012*`:*`.

L'exemple suivant montre une politique de confiance dans les rôles que vous pouvez appliquer pour limiter l'accès à n'importe quelle application Amplify de votre compte et éviter le problème de confusion des adjoints. Pour utiliser cette politique, remplacez le texte en italique rouge dans l'exemple de politique par vos propres informations.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": [
          "amplify.me-south-1.amazonaws.com",
          "amplify.eu-south-1.amazonaws.com",
          "amplify.ap-east-1.amazonaws.com",
          "amplifybackend.amazonaws.com",
          "amplify.amazonaws.com"
        ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
```

```
}
```

L'exemple suivant montre une politique de confiance dans les rôles que vous pouvez appliquer pour limiter l'accès à une application Amplify spécifiée dans votre compte et éviter le problème de confusion des adjoints. Pour utiliser cette politique, remplacez le texte en italique rouge dans l'exemple de politique par vos propres informations.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": [
          "amplify.me-south-1.amazonaws.com",
          "amplify.eu-south-1.amazonaws.com",
          "amplify.ap-east-1.amazonaws.com",
          "amplifybackend.amazonaws.com",
          "amplify.amazonaws.com"
        ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/d123456789/
branches/*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

Bonnes pratiques de sécurité pour Amplify

Amplify fournit un certain nombre de fonctionnalités de sécurité à prendre en compte lorsque vous développez et mettez en œuvre vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou

suffisantes pour votre environnement, considérez-les comme des recommandations utiles plutôt que comme des prescriptions.

Utilisation de cookies avec le domaine par défaut Amplify

Lorsque vous utilisez Amplify pour déployer une application Web, Amplify l'héberge pour vous sur le domaine par défaut. `amplifyapp.com` Vous pouvez afficher votre application sur une URL au format. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com`

Pour renforcer la sécurité de vos applications Amplify, le domaine amplifyapp.com est enregistré dans la liste des suffixes publics (PSL). Pour plus de sécurité, nous vous recommandons d'utiliser des cookies avec un `__Host-` préfixe si vous devez définir des cookies sensibles dans le nom de domaine par défaut de vos applications Amplify. Cette pratique vous aidera à protéger votre domaine contre les tentatives de falsification de requêtes intersites (CSRF). Pour plus d'informations, consultez la page Set-Cookie du Mozilla Developer Network.

Quotas du service Amplify Hosting

Les quotas de service pour l' AWS Amplify hébergement sont les suivants. Les quotas de service (précédemment appelés limites) sont le nombre maximum de ressources de service ou d'opérations pour votre Compte AWS.

Comptes AWS Les nouveautés ont réduit les quotas d'applications et de tâches simultanées. AWS augmente automatiquement ces quotas en fonction de votre utilisation. Vous pouvez également demander une augmentation de quota.

La console Service Quotas fournit des informations sur les quotas de votre compte. Vous pouvez utiliser la console Service Quotas pour afficher les quotas par défaut et demander des augmentations de quota pour les quotas ajustables. Pour de plus amples informations, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Nom	Par défaut	Ajusta	Description
Applications	Chaque région prise en charge : 25	Oui	Le nombre maximum d'applications que vous pouvez créer dans AWS Amplify Console dans ce compte dans la région actuelle.
Branches par application	Chaque région prise en charge : 50	Non	Nombre maximal de branches par application que vous pouvez créer dans ce compte dans la région actuelle.
Taille de l'artefact de génération	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un artefact de génération d'applica tion. Un artefact de build est déployé par AWS Amplify Console après une compilation.

Nom	Par défaut	Ajusta	Description
Taille de l'artefact du cache	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un artefact de cache.
Tâches simultanées	Chaque Région prise en charge : 5	Oui	Le nombre maximum de tâches simultanées que vous pouvez créer dans ce compte dans la région actuelle.
Domaines par application	Chaque Région prise en charge : 5	Oui	Nombre maximal de domaines par application que vous pouvez créer dans ce compte dans la région actuelle.
Taille de l'artefact du cache d'environnement	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) de l'artefact du cache d'environnement.
Taille du fichier ZIP de déploiement manuel	Chaque Région prise en charge : 5 giga-octets	Non	Taille maximale (en Go) d'un fichier ZIP de déploiement manuel.
Nombre maximum de créations d'applications par heure	Chaque région prise en charge : 25	Non	Le nombre maximum d'applications que vous pouvez créer dans AWS Amplify Console par heure sur ce compte dans la région actuelle.

Nom	Par défaut	Ajusta	Description
Demander des jetons par seconde	Chaque région prise en charge : 20 000	Oui	Le nombre maximum de jetons de demande par seconde pour une application. Amplify Hosting alloue des jetons aux demandes en fonction de la quantité de ressources (temps de traitement et transfert de données) qu'elles consomment.
Sous-domaines par domaine	Chaque région prise en charge : 50	Non	Nombre maximal de sous-domaines par domaine que vous pouvez créer dans ce compte dans la région actuelle.
Webhooks par application	Chaque Région prise en charge : 50	Oui	Nombre maximal de webhooks par applicati on que vous pouvez créer dans ce compte dans la région actuelle.

Pour plus d'informations sur les quotas de service Amplify, consultez la section AWS Amplify Points de terminaison et quotas dans le. Références générales AWS

Résolution des problèmes liés à Amplify Hosting

Si vous rencontrez des erreurs ou des problèmes de déploiement lorsque vous travaillez avec Amplify Hosting, consultez les rubriques de cette section.

Rubriques

Résolution des problèmes généraux liés à Amplify

Les informations suivantes peuvent vous aider à résoudre les problèmes généraux liés à Amplify Hosting.

Rubriques

Code d'état HTTP 429 (trop de requêtes)

Amplify contrôle le nombre de requêtes par seconde (RPS) adressées à votre site Web en fonction du temps de traitement et du transfert de données consommés par les demandes entrantes. Si votre application renvoie un code d'état HTTP 429, les demandes entrantes dépassent le temps de traitement et de transfert de données alloué à votre application. Cette limite d'applications est gérée par le quota de `REQUEST_TOKENS_PER_SECOND` service d'Amplify. Pour de plus amples informations sur les quotas, veuillez consulter Quotas du service Amplify Hosting.

Pour résoudre ce problème, nous vous recommandons d'optimiser votre application afin de réduire la durée des demandes et le transfert de données afin d'augmenter le RPS de l'application. Par exemple, avec les mêmes 20 000 jetons, une page SSR hautement optimisée qui répond dans les 100 millisecondes peut supporter un RPS plus élevé qu'une page avec une latence supérieure à 200 millisecondes.

De même, une application qui renvoie une taille de réponse de 1 Mo consommera plus de jetons qu'une application qui renvoie une taille de réponse de 250 Ko.

Nous vous recommandons également de tirer parti du CloudFront cache Amazon en configurant des en-têtes de contrôle du cache qui maximisent le temps pendant lequel une réponse donnée est conservée dans le cache. Les demandes traitées depuis le CloudFront cache ne sont pas prises en compte dans le calcul de la limite de débit. Chaque CloudFront distribution peut traiter jusqu'à 250 000 requêtes par seconde, ce qui vous permet de faire évoluer votre application de manière très élevée en utilisant le cache. Pour plus d'informations sur le CloudFront cache, consultez Optimisation de la mise en cache et de la disponibilité dans le manuel Amazon CloudFront Developer Guide.

Résolution des problèmes liés à l'image de build d'Amazon Linux 2023

Les informations suivantes peuvent vous aider à résoudre les problèmes liés à l'image de build Amazon Linux 2023 (AL2023).

Rubriques

- Comment exécuter les fonctions Amplify avec le moteur d'exécution Python
- Comment exécuter des commandes qui nécessitent des privilèges de superutilisateur ou de root

Comment exécuter les fonctions Amplify avec le moteur d'exécution Python

Amplify Hosting utilise désormais l'image de build Amazon Linux 2023 par défaut lorsque vous déployez une nouvelle application. AL2023 est préinstallé avec les versions 3.8, 3.9, 3.10 et 3.11 de Python.

Pour des raisons de rétrocompatibilité avec l'image Amazon Linux 2, l'image de build AL2023 contient des liens symboliques préinstallés vers les anciennes versions de Python. Par conséquent, vous n'avez plus besoin de mettre à jour les commandes de génération dans les spécifications de construction de votre application à l'aide des instructions disponibles dans la FAQ d'Amplify Hosting GitHub .

Par défaut, la version 3.10 de Python est utilisée dans le monde entier. Pour créer vos fonctions à l'aide d'une version spécifique de Python, exécutez les commandes suivantes dans le fichier de spécification de construction de votre application.

```
version: 1
backend:
  phases:
```

```
    build:
      commands:
        # use a python version globally
        - pyenv global 3.11
        # verify python version
        - python --version
        # install pipenv
        - pip install --user pipenv
        # add to path
        - export PATH=$PATH:/root/.local/bin
        # verify pipenv version
        - pipenv --version
        - amplifyPush --simple
```

Comment exécuter des commandes qui nécessitent des privilèges de superutilisateur ou de root

Si vous utilisez l'image de build Amazon Linux 2023 et que vous recevez une erreur lors de l'exécution de commandes système nécessitant des privilèges de superutilisateur ou de superutilisateur, vous devez exécuter ces commandes à l'aide de la `sudo` commande Linux. Par exemple, si une erreur s'affiche lors de l'exécution`yum install -y gcc`, utilisez`sudo yum install -y gcc`.

L'image de build d'Amazon Linux 2 utilisait l'utilisateur root, mais l'image AL2023 d'Amplify exécute votre code avec un utilisateur personnalisé`amplify`. Amplify accorde à cet utilisateur les privilèges nécessaires pour exécuter des commandes à l'aide de la commande Linux. `sudo` Il s'agit d'une bonne pratique à utiliser `sudo` pour les commandes qui nécessitent des privilèges de superutilisateur.

Résolution des problèmes liés aux domaines personnalisés

Si vous rencontrez des problèmes lors de la connexion d'un domaine personnalisé à votre application Amplify, consultez Résolution des problèmes liés aux domaines personnalisés l'aide.

Résolution des problèmes liés aux applications rendues côté serveur

Si vous rencontrez des problèmes lors du déploiement d'une application SSR sur Amplify, Résolution des problèmes liés aux déploiements SSR consultez l'aide.

AWS AmplifyRéférence d'hébergement

Consultez les rubriques de cette section pour trouver des documents de référence détaillés surAWS Amplify.

Rubriques

Prise en charge de AWS CloudFormation

UtiliserAWS CloudFormationdes modèles pour fournir des ressources Amplify, permettant ainsi des déploiements d'applications Web reproductibles et fiables.AWS CloudFormationfournit un langage commun vous permettant de décrire et de provisionner toutes les ressources d'infrastructure de votre environnement cloud et simplifie le déploiement sur plusieursAWScomptes et/ou régions en quelques clics.

Pour Amplify Hosting, consultez leAmplifierCloudFormationdocumentation. Pour Amplify Studio, consultez leGénérateur d'interface utilisateur AmplifyCloudFormationdocumentation.

Prise en charge de AWS Command Line Interface

Utilisez leAWS Command Line Interfacepour créer des applications Amplify par programmation à partir de la ligne de commande. Pour plus d'informations, consultez leAWS CLIdocumentation.

Support pour le balisage des ressources

Vous pouvez utiliser leAWS Command Line Interfacepour baliser les ressources Amplify. Pour plus d'informations, consultez leAWS CLIdocumentation sur les balises.

API d'hébergement Amplify

Cette référence fournit des descriptions des actions et des types de données pour l'API d'hébergement Amplify. Pour plus d'informations, consultez leRéférence de l'API Amplifydocumentation.

Historique du document pour AWS Amplify

Le tableau suivant décrit les modifications importantes apportées à la documentation depuis la dernière version de AWS Amplify.

- Dernière mise à jour de la documentation : 5 avril 2024

Modification	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	5 avril 2024
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	4 avril 2024
Nouveau chapitre sur le dépannage	Ajout du Résolution des problèmes liés à Amplify Hosting chapitre décrivant comment résoudre les problèmes que vous rencontrez avec les applications déployées sur Amplify Hosting.	2 avril 2024
Nouveau support pour les certificats SSL/TLS personnalisés	Ajout d'une Utilisation de certificats SSL/TLS rubrique au Configuration de domaines personnalisés chapitre pour	20 février 2024

Modification	Description	Date
	décrire la prise en charge par Amplify des certificats SSL/TLS personnalisés lors de la connexion d'une application à un domaine personnalisé.	
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	2 janvier 2024
Nouveau support pour les frameworks SSR	Ajout de la Amplify la prise en charge des frameworks SSR rubrique décrivant le support d'Amplify pour tout framework SSR basé sur JavaScript avec un adaptateur open source.	19 novembre 2023
Nouveau support pour le lancement de la fonctionnalité d'optimisation des images	Ajout d'une Optimisation de l'image pour les applications SSR rubrique décrivant la prise en charge intégrée de l'optimisation des images pour les applications rendues côté serveur.	19 novembre 2023
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	17 novembre 2023

Modification	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	6 novembre 2023
Nouvelle rubrique sur les sous-domaines génériques	Ajout d'une Sous-domaines Wildcard rubrique décrivant la prise en charge des sous-domaines génériques sur les domaines personnalisés.	6 novembre 2023
Nouvelle politique gérée par	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire la nouvelle politique AmplifyBackendDeployFullAccess AWS gérée pour Amplify.	8 octobre 2023
Nouveau support pour les frameworks monorepo : lancement des fonctionnalités	Mise à jour de la Paramètres de construction de Monorepo rubrique pour décrire la prise en charge du déploiement d'applications dans des monorepos créés à l'aide de npm workspace, pnpm workspace, Yarn workspace, Nx et Turborepo.	19 juin 2023

Modification	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	1er juin 2023
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	24 février 2023
Chapitre sur le rendu côté serveur mis à jour	Le Déployez des applications rendues côté serveur avec Amplify Hosting chapitre a été mis à jour pour décrire les modifications récentes apportées à la prise en charge par Amplify des versions 12 et 13 de Next.js.	17 novembre 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	30 août 2022

Modification	Description	Date
Rubrique sur les politiques gérées mise à jour	Mise à jour de la Démarrez avec les déploiements continus Fullstack rubrique pour décrire comment déployer un backend à l'aide d'Amplify Studio.	23 août 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	27 avril 2022
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	17 avril 2022
Lancement d'une nouvelle fonctionnalité de GitHub l'application	Ajout d'une Configuration de l'accès Amplify aux GitHub référentiels rubrique décrivant la nouvelle GitHub application permettant d'autoriser Amplify à accéder à GitHub votre référentiel.	5 avril 2022

Modification	Description	Date
Lancement de la nouvelle fonctionnalité Amplify Studio	Mise à jour de la Bienvenue chez AWS Amplify Hosting rubrique pour décrire les mises à jour d'Amplify Studio qui fournissent un concepteur visuel pour créer des composants d'interface utilisateur que vous pouvez connecter à vos données principales.	2 décembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify afin de prendre en charge Amplify Studio.	2 décembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	8 novembre 2021
Rubrique sur les politiques gérées mise à jour	Mise à jour de la AWS politiques gérées pour AWS Amplify rubrique pour décrire les modifications récentes apportées aux politiques AWS gérées pour Amplify.	27 septembre 2021

Modification	Description	Date
Nouvelle rubrique sur les politiques gérées	Ajout d'une AWS politiques gérées pour AWS Amplify rubrique décrivant les politiques AWS gérées pour Amplify et les modifications récentes apportées à ces politiques.	28 juillet 2021
Chapitre sur le rendu côté serveur mis à jour	Le Déployez des applications rendues côté serveur avec Amplify Hosting chapitre a été mis à jour pour décrire le nouveau support de Next.js version 10. x. x et Next.js version 11.	22 juillet 2021
Chapitre sur la configuration des paramètres de construction mis à jour	Ajout d'une Paramètres de construction de Monorepo rubrique décrivant comment configurer les paramètres de compilation et la nouvelle variable d'`AMPLIFY_MONOREPO_APP_ROOT` environnement lors du déploiement d'une application monorepo avec Amplify.	20 juillet 2021

Modification	Description	Date
Chapitre actualisé sur les déploiements dans les branches de fonctionnalités	Ajout d'une Génération automatique de la configuration Amplify au moment de la construction rubrique décrivant comment générer automatiquement le `aws-exports.js` fichier au moment de la création. Ajout d'une Configurations conditionnelles du backend rubrique décrivant comment activer les builds de backend conditionnels. Ajout d'une Utilisez les backends Amplify dans toutes les applications rubrique décrivant comment réutiliser les backends existants lorsque vous créez une nouvelle application, connectez une nouvelle branche à une application existante ou mettez à jour un frontend existant pour qu'il pointe vers un environnement de backend différent.	30 Juin 2021

Modification	Description	Date
Chapitre sur la sécurité mis à jour	Ajout d'une Protection des données dans Amplify rubrique décrivant comment appliquer le modèle de responsabilité partagée et comment Amplify utilise le chiffrement pour protéger vos données au repos et en transit.	3 juin 2021
Nouveau support pour le lancement de la fonctionnalité SSR	Ajout du Déployez des applications rendues côté serveur avec Amplify Hosting chapitre décrivant le support d'Amplify pour les applications Web qui utilisent le rendu côté serveur (SSR) et sont créées avec Next.js.	18 mai 2021
Nouveau chapitre sur la sécurité	Ajout du Sécurité dans Amplify chapitre décrivant comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amplify et comment configurer Amplify pour atteindre vos objectifs de sécurité et de conformité.	26 mars 2021

Modification	Description	Date
Rubrique sur les builds personnalisés mise à jour	Mise à jour de la rubrique Images de construction personnalisées et mises à jour des packages en direct pour décrire comment configure r une image de construction personnalisée hébergée dans Amazon Elastic Container Registry Public.	12 mars 2021
Rubrique de surveillance mise à jour	Mise à jour de la rubrique Surveillance pour décrire comment accéder aux données CloudWatch des métriques Amazon et définir des alarmes.	2 février 2021
Nouveau sujet de CloudTrail journalisation	Ajout de la AWS CloudTrail rubrique Logging Amplify API using pour décrire comment AWS CloudTrail capture et enregistre toutes les actions d'API pour la référence d'API de AWS Amplify console et la référence d'API AWS Amplify d'interface utilisateur d'adminis tration.	2 février 2021

Modification	Description	Date
Lancement de nouvelles fonctionnalités de l'interface utilisateur d'administration	Mise à jour de la Bienvenue chez AWS Amplify Hosting rubrique pour décrire la nouvelle interface utilisateur d'administration qui fournit une interface visuelle permettant aux développeurs Web et mobiles de créer et de gérer des backends d'applications en dehors de. AWS Management Console	1er décembre 2020
Lancement d'une nouvelle fonctionnalité du mode performance	Mise à jour de la rubrique Gestion des performances des applications pour décrire comment activer le mode performance afin d'optimiser les performances d'hébergement plus rapides.	4 novembre 2020
Mise à jour de la rubrique sur les en-têtes personnalisés	Mise à jour de la rubrique En-têtes personnalisés pour décrire comment définir des en-têtes personnalisés pour une application Amplify à l'aide de la console ou en modifiant un fichier YML.	28 octobre 2020

Modification	Description	Date
Lancement de la nouvelle fonctionnalité de sous-domaines automatiques	Ajout de la rubrique Configurer des sous-domaines automatiques pour un domaine personnalisé Route 53 afin de décrire comment utiliser les déploiements de branches de fonctionnalités basés sur des modèles pour une application connectée à un domaine personnalisé Amazon Route 53. Ajout de la rubrique Accès à l'aperçu Web avec les sous-domaines pour décrire comment configurer les aperçus Web à partir de pull requests afin qu'ils soient accessibles avec les sous-domaines.	20 juin 2020
Nouveau sujet de notifications	Ajout de la rubrique Notifications pour décrire comment configurer des notifications par e-mail pour une application Amplify afin d'avertir les parties prenantes ou les membres de l'équipe en cas de réussite ou d'échec d'un build.	20 juin 2020

Modification	Description	Date
Mise à jour de la rubrique sur les domaines personnalisés	Mise à jour de la Configuration de domaines personnalisés rubrique afin d'améliorer les procédures d'ajout de domaines personnalisés dans Amazon Route 53 et Google Domains. GoDaddy Cette mise à jour inclut également de nouvelles informations de dépannage pour la configuration de domaines personnalisés.	12 mai 2020
AWS Amplify libération	Cette version présente Amplify.	26 novembre 2018

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.