Meilleures pratiques d'utilisation de l' AWS CDK in pour TypeScript créer des projets IaC

AWS Conseils prescriptifs

AWS Conseils prescriptifs: Meilleures pratiques d'utilisation de l' AWS CDK in pour TypeScript créer des projets IaC

Table of Contents

Bonnes pratiques d'utilisation du kit AWS CDK TypeScript pour créer des projets IaC

Sandeep Gawande, Mason Cahill, Sandip Gangapadhyay, Siamak Heshmati et Rajneesh Tyagi, Amazon Web Services (AWS)

Février 2024 (historique du document)

Ce guide fournit des recommandations et des bonnes pratiques pour utiliser l'AWS Cloud Development Kit (AWS CDK)in TypeScript pour créer et déployer des projets d'infrastructure en tant que code (IaC) à grande échelle. AWS CDK Il s'agit d'un framework permettant de définir l'infrastructure cloud dans le code et de provisionner cette infrastructure par le biais AWS CloudFormation de celui-ci. Si vous ne disposez pas d'une structure de projet bien définie, la création et la gestion d'une AWS CDK base de code pour des projets de grande envergure peuvent s'avérer difficiles. Pour relever ces défis, certaines organisations utilisent des anti-modèles pour les projets à grande échelle, mais ces modèles peuvent ralentir votre projet et engendrer d'autres problèmes qui ont un impact négatif sur votre organisation. Par exemple, les anti-modèles peuvent compliquer et ralentir l'intégration de développeurs, les corrections de bogues et l'adoption de nouvelles fonctionnalités.

Ce guide propose une alternative à l'utilisation d'anti-modèles et vous explique comment organiser votre code à des fins de capacité de mise à l'échelle, de test et d'alignement sur les bonnes pratiques de sécurité. Vous pouvez utiliser ce guide pour améliorer la qualité du code de vos projets IaC et optimiser l'agilité de votre entreprise. Ce guide est destiné aux architectes, aux responsables techniques, aux ingénieurs d'infrastructure et à toute autre personne cherchant à créer un AWS CDK projet bien conçu pour des projets de grande envergure.

Objectifs

Ce guide peut vous aider à atteindre les résultats métier ciblés suivants :

- Coûts réduits — Vous pouvez utiliser le AWS CDK pour concevoir vos propres composants réutilisables qui répondent aux exigences de sécurité, de conformité et de gouvernance de votre entreprise. Vous pouvez également partager facilement des composants au sein de votre organisation, afin d'amorcer rapidement de nouveaux projets conformes aux bonnes pratiques par défaut.
- Réduction des délais de mise sur le marché : profitez des fonctionnalités habituelles du AWS CDK pour accélérer votre processus de développement. Cela augmente la réutilisabilité pour le déploiement et réduit les efforts de développement.
- Productivité accrue des développeurs : les développeurs peuvent utiliser des langages de programmation familiers pour définir l'infrastructure. Cela aide les développeurs à exprimer et à gérer AWS les ressources. Cela peut améliorer l'efficacité et la collaboration des développeurs.

Bonnes pratiques

Cette section fournit une vue d'ensemble des bonnes pratiques suivantes :

- Organiser le code pour les projets à grande échelle
- Développer des modèles réutilisables
- Créer ou étendre des constructions
- Suivez les TypeScript meilleures pratiques
- Rechercher les vulnérabilités de sécurité et les erreurs de formatage
- Développer et affiner la documentation
- Adopter une approche de développement piloté par les tests
- Utiliser le contrôle des éditions et des versions pour les constructions
- Appliquer la gestion des versions de la bibliothèque

Organiser le code pour les projets à grande échelle

Pourquoi l'organisation du code est-elle importante ?

Il est essentiel que les AWS CDK projets de grande envergure aient une structure bien définie et de haute qualité. À mesure qu'un projet prend de l'ampleur et que le nombre de fonctionnalités et de constructions prises en charge augmente, la navigation du code devient plus difficile. Cette difficulté peut avoir un impact sur la productivité et ralentir l'intégration des développeurs.

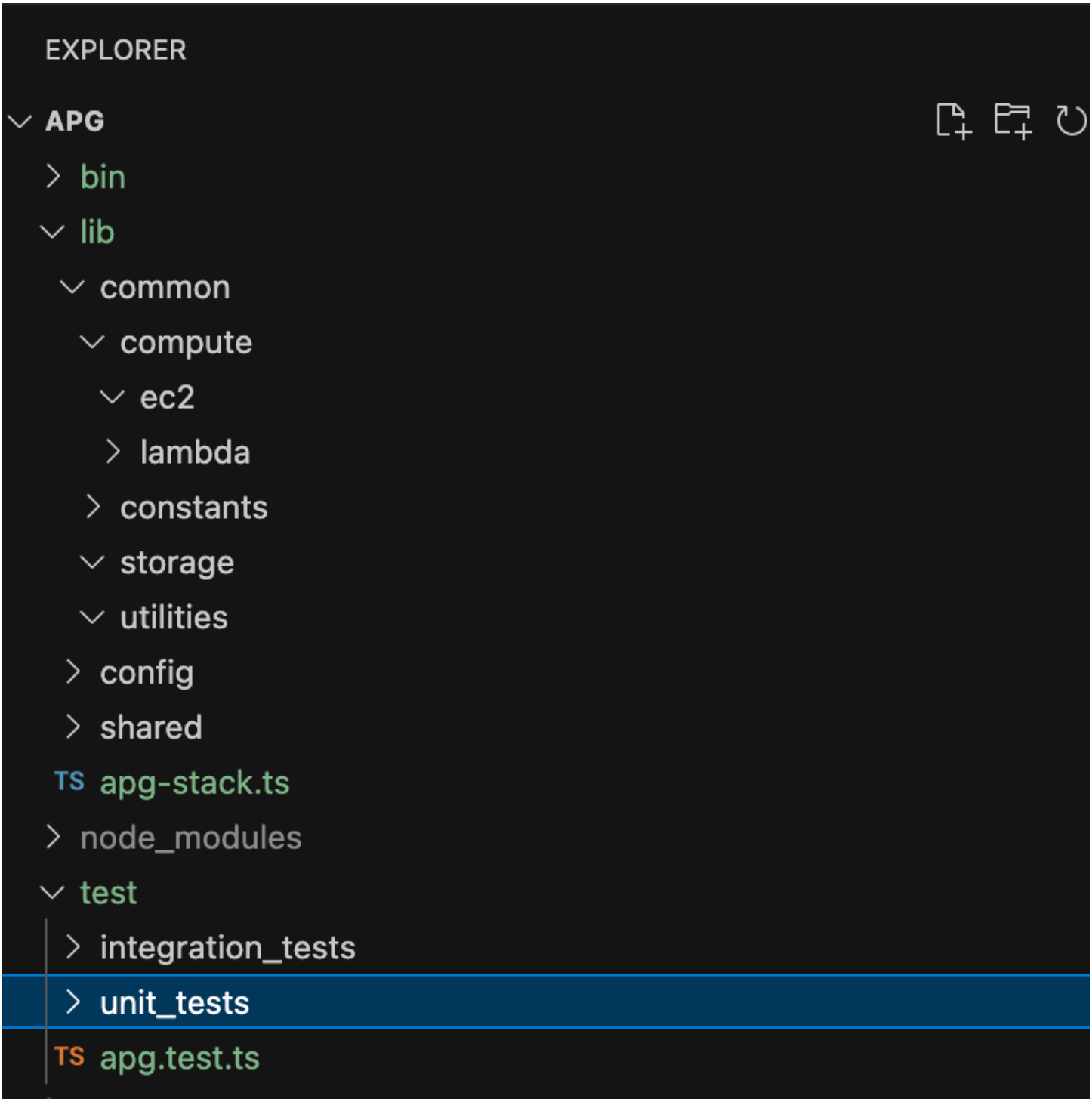
Comment organiser votre code pour le mettre à l'échelle

Pour atteindre un niveau élevé de flexibilité et de lisibilité du code, nous vous recommandons de le diviser en éléments logiques en fonction des fonctionnalités. Cette division reflète le fait que la plupart de vos constructions sont utilisées dans différents domaines métier. Par exemple, vos applications frontales et dorsales peuvent nécessiter une AWS Lambda fonction et consommer le même code source. Les usines peuvent créer des objets sans exposer la logique de création au client et utiliser une interface commune pour faire référence aux objets récemment créés. Vous pouvez utiliser une usine comme modèle efficace pour créer un comportement cohérent dans votre base de code. En outre, une usine peut servir de source de vérité fiable unique pour vous aider à éviter le code répétitif et à simplifier le dépannage.

Pour mieux comprendre le fonctionnement des usines, prenons l'exemple d'un constructeur automobile. Un constructeur automobile n'a pas besoin des connaissances et de l'infrastructure nécessaires pour fabriquer des pneus. Le constructeur automobile sous-traite plutôt cette expertise à un fabricant spécialisé dans les pneus, puis lui commande simplement les pneus en fonction des besoins. Le même principe s'applique au code. Par exemple, vous pouvez créer une usine Lambda capable de créer des fonctions Lambda de haute qualité, puis appeler l'usine Lambda dans votre code chaque fois que vous devez créer une fonction Lambda. De même, vous pouvez utiliser ce même processus d'externalisation pour découpler votre application et créer des composants modulaires.

Exemple d'organisation de code

L' TypeScript exemple de projet suivant, comme le montre l'image suivante, inclut un dossier commun dans lequel vous pouvez conserver toutes vos constructions ou fonctionnalités communes.



Par exemple, le dossier calculer (résidant dans le dossier commun) contient toute la logique des différentes constructions de calcul. Les nouveaux développeurs peuvent facilement ajouter de nouvelles constructions de calcul sans affecter les autres ressources. Toutes les autres constructions n'auront pas besoin de créer de nouvelles ressources en interne. Au lieu de cela, ces constructions

appellent simplement l'usine de constructions communes. Vous pouvez organiser d'autres constructions, telles que le stockage, de la même manière.

Les configurations contiennent des données basées sur l'environnement que vous devez dissocier du dossier commun dans lequel vous conservez la logique. Nous vous recommandons de placer vos données de configuration dans un dossier partagé. Nous vous recommandons également d'utiliser le dossier utilitaires pour servir toutes les fonctions d'assistance et les scripts de nettoyage.

Développer des modèles réutilisables

Les modèles de conception logicielle sont des solutions réutilisables aux problèmes courants du développement logiciel. Ils servent de guide ou de paradigme pour aider les ingénieurs logiciels à créer des produits conformes aux bonnes pratiques. Cette section fournit un aperçu de deux modèles réutilisables que vous pouvez utiliser dans votre AWS CDK base de code : le modèle Abstract Factory et le modèle Chain of Responsibility. Vous pouvez utiliser chaque modèle comme plan et le personnaliser en fonction du problème de conception particulier de votre code. Pour obtenir plus d'informations sur les modèles de conception, veuillez consulter Design Patterns dans la documentation de Refactoring.Guru.

Abstract Factory

Le modèle Abstract Factory propose des interfaces permettant de créer des familles d'objets associés ou dépendants sans spécifier leurs classes concrètes. Ce modèle s'applique aux cas d'utilisation suivants :

- Lorsque le client est indépendant de la façon dont vous créez et composez les objets dans le système.
- Lorsque le système est composé de plusieurs familles d'objets et que ces familles sont conçues pour être utilisées ensemble.
- Lorsque vous devez disposer d'une valeur d'exécution pour construire une dépendance particulière.

Pour plus d'informations sur le modèle Abstract Factory, consultez Abstract Factory TypeScript dans la documentation de Refactoring.Guru.

L'exemple de code suivant montre comment le modèle Abstract Factory peut être utilisé pour créer une usine de stockage Amazon Elastic Block Store (Amazon EBS).

```
abstract class EBSStorage {
    abstract initialize(): void;
}

class ProductEbs extends EBSStorage{
    constructor(value: String) {
        super();
        console.log(value);
    }
    initialize(): void {}
}

abstract class AbstractFactory {
    abstract createEbs(): EBSStorage
}

class EbsFactory extends AbstractFactory {
    createEbs(): ProductEbs{
        return new ProductEbs('EBS Created.')
    }
}

const ebs = new EbsFactory();
ebs.createEbs();
```

Chain of Responsibility

Chain of Responsibility est un modèle de conception comportementale qui vous permet de transmettre une demande à la chaîne des gestionnaires potentiels jusqu'à ce que l'un d'eux la traite. Le modèle Chain of Responsibility s'applique aux cas d'utilisation suivants :

- Lorsque plusieurs objets, déterminés lors de l'exécution, sont candidats pour traiter une demande.
- Lorsque vous ne souhaitez pas spécifier de gestionnaires explicitement dans votre code
- Lorsque vous souhaitez envoyer une demande à l'un des nombreux objets sans spécifier explicitement le destinataire.

Pour plus d'informations sur le modèle de chaîne de responsabilité, voir Chaîne de responsabilité TypeScript dans la documentation de Refactoring.Guru.

Le code suivant montre un exemple de la manière dont le modèle Chain of Responsibility est utilisé pour générer une série d'actions nécessaires à l'exécution de la tâche.

```
interface Handler {
    setNext(handler: Handler): Handler;
    handle(request: string): string;
}
abstract class AbstractHandler implements Handler
{
    private nextHandler: Handler;
    public setNext(handler: Handler): Handler {
        this.nextHandler = handler;
        return handler;
    }

    public handle(request: string): string {
        if (this.nextHandler) {
            return this.nextHandler.handle(request);
        }
        return '';
    }
}

class KMSHandler extends AbstractHandler {
    public handle(request: string): string {
        return super.handle(request);
    }
}
```

Créer ou étendre des constructions

Qu'est-ce qu'une construction ?

Une construction est l'élément de base d'une AWS CDK application. Une construction peut représenter une AWS ressource unique, telle qu'un bucket Amazon Simple Storage Service (Amazon S3), ou il peut s'agir d'une abstraction de niveau supérieur composée de plusieurs ressources connexes. AWS Les composants d'une construction peuvent inclure une file d'attente de travail avec sa capacité de calcul associée, ou une tâche planifiée avec des ressources de surveillance et un tableau de bord. AWS CDK Il inclut une collection de constructions appelée AWS Construct Library. La bibliothèque contient des constructions pour chaque AWS service. Vous pouvez utiliser

le Construct Hub pour découvrir des constructions supplémentaires provenant de AWS tiers et de la communauté open source AWS CDK .

Quels sont les différents types de construction ?

Il existe trois types de constructions différents pour : AWS CDK

- Constructions L1 — Les constructions de couche 1, ou L1, sont exactement les ressources définies par CloudFormation —ni plus, ni moins. Vous devez fournir vous-même les ressources nécessaires à la configuration. Ces constructions L1 sont très basiques et doivent être configurées manuellement. Les constructions L1 ont un `Cfn` préfixe et correspondent directement aux spécifications. CloudFormation AWS services Les nouveaux sont pris en charge AWS CDK dès que ces services sont CloudFormation pris en charge. CfnBucketest un bon exemple de construction L1. Cette classe représente un compartiment S3 dans lequel vous devez configurer explicitement toutes les propriétés. Nous vous recommandons de n'utiliser une construction L1 que si vous ne trouvez pas la construction L2 ou L3 correspondante.
- Constructions L2 : les constructions Layer 2, ou L2, ont un code réutilisable et une logique de collage communs. Ces constructions comportent des valeurs par défaut pratiques et réduisent la quantité de connaissances que vous devez connaître à leur sujet. Les constructions L2 utilisent des API basées sur l'intention pour créer vos ressources et encapsulent généralement les modules L1 correspondants. Un bon exemple de construction L2 est Compartiment. Cette classe crée un compartiment S3 avec des propriétés et des méthodes par défaut telles que le bucket. addLifeCycleRule (), qui ajoute une règle de cycle de vie au compartiment.
- Constructions L3 : une construction Layer 3, ou L3, est appelée un modèle. Les constructions L3 sont conçues pour vous aider à effectuer des tâches courantes AWS, impliquant souvent plusieurs types de ressources. Elles sont encore plus spécifiques et mieux définies que les constructions L2 et répondent à un cas d'utilisation spécifique. Par exemple, le aws-ecs-patterns. ApplicationLoadBalancedFargateServiceconstruct représente une architecture qui inclut un cluster de AWS Fargate conteneurs utilisant un Application Load Balancer. Un autre exemple est le aws-apigateway. LambdaRestApiconstruire. Cette construction représente une API Amazon API Gateway qui est soutenue par une fonction Lambda.

À mesure que les niveaux de construction augmentent, de plus en plus d'hypothèses sont formulées quant à la manière dont ces constructions seront utilisées. Cela vous permet de fournir des interfaces avec des valeurs par défaut plus efficaces pour des cas d'utilisation très spécifiques.

Comment créer votre propre construction

Pour définir votre propre construction, vous devez suivre une approche spécifique. Cela est dû au fait que toutes les constructions étendent la classe `Construct`. La classe `Construct` est la composante de base de l'arbre de construction. Les constructions sont implémentées dans des classes qui étendent la classe de base `Construct`. Toutes les constructions prennent trois paramètres lorsqu'elles sont initialisées :

- Portée — Le parent ou le propriétaire d'une construction, qu'il s'agisse d'une pile ou d'une autre construction, détermine sa place dans l'arbre des constructions. Vous devez généralement transmettre `this` (ou `self` en Python), qui représente l'objet actuel, pour la portée.
- ID : identifiant qui doit être unique dans cette portée. L'identifiant sert d'espace de noms pour tout ce qui est défini dans la construction actuelle et est utilisé pour attribuer des identités uniques, telles que les noms de ressources et les identifiants CloudFormation logiques.
- Accessoires — Ensemble de propriétés qui définissent la configuration initiale de la construction.

L'exemple suivant montre comment définir une construction.

```
import { Construct } from 'constructs';

export interface CustomProps {
  // List all the properties
  Name: string;
}
export class MyConstruct extends Construct {
  constructor(scope: Construct, id: string, props: CustomProps) {
    super(scope, id);

    // TODO
  }
}
```

Création ou extension d'une construction L2

Une construction L2 représente un « composant cloud » et encapsule tout ce qui CloudFormation doit être nécessaire pour créer le composant. Une construction L2 peut contenir une ou plusieurs AWS ressources, et vous êtes libre de la personnaliser vous-même. L'avantage de créer ou d'étendre une

construction L2 est que vous pouvez réutiliser les composants dans des CloudFormation piles sans redéfinir le code. Vous pouvez simplement importer la construction en tant que classe.

Lorsqu'il existe une relation « existe » avec une construction existante, vous pouvez étendre une construction existante pour ajouter des fonctionnalités par défaut supplémentaires. Il est recommandé de réutiliser les propriétés de la construction L2 existante. Vous pouvez remplacer les propriétés en les modifiant directement dans le constructeur.

L'exemple suivant montre comment s'aligner sur les bonnes pratiques et étendre une construction L2 existante appelée `s3.Bucket`. L'extension définit les propriétés par défaut, telles que `versioned`, `publicReadAccess`, `blockPublicAccess`, pour s'assurer que tous les objets (dans cet exemple, les compartiments S3) créés à partir de cette nouvelle construction auront toujours ces valeurs par défaut définies.

```
import * as s3 from 'aws-cdk-lib/aws-s3';
import { Construct } from 'constructs';
export class MySecureBucket extends s3.Bucket {
  constructor(scope: Construct, id: string, props?: s3.BucketProps) {

    super(scope, id, {
      ...props,
      versioned: true,
      publicReadAccess: false,
      blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL
    });
  }
}
```

Création d'une construction L3

La composition est le meilleur choix lorsqu'il existe une relation « a » avec une composition de construction existante. La composition signifie que vous générez votre propre construction par-dessus d'autres constructions existantes. Vous pouvez créer votre propre modèle pour encapsuler toutes les ressources et leurs valeurs par défaut dans une seule construction L3 de niveau supérieur pouvant être partagée. L'avantage de créer vos propres constructions L3 (modèles) est que vous pouvez réutiliser les composants dans des piles sans redéfinir le code. Vous pouvez simplement importer la construction en tant que classe. Ces modèles sont conçus pour aider les consommateurs à fournir de manière concise plusieurs ressources basées sur des modèles communs avec une quantité limitée de connaissances.

L'exemple de code suivant crée une AWS CDK construction appelée`ExampleConstruct`. Vous pouvez utiliser cette construction comme modèle pour définir vos composants cloud.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';

export interface ExampleConstructProps {
  //insert properties you wish to expose
}

export class ExampleConstruct extends Construct {
  constructor(scope: Construct, id: string, props: ExampleConstructProps) {
    super(scope, id);
    //Insert the AWS components you wish to integrate
  }
}
```

L'exemple suivant montre comment importer la nouvelle construction dans votre AWS CDK application ou votre pile.

```
import { ExampleConstruct } from './lib/construct-name';
```

L'exemple suivant montre comment instancier une instance de la construction que vous avez étendue à partir de la classe de base.

```
import { ExampleConstruct } from './lib/construct-name';

new ExampleConstruct(this, 'newConstruct', {
  //insert props which you exposed in the interface `ExampleConstructProps`
});
```

Pour plus d'informations, consultez AWS CDK Workshop dans la documentation de l' AWS CDK atelier.

Trappe de secours

Vous pouvez utiliser une trappe d'échappement dans le AWS CDK pour monter d'un niveau d'abstraction afin d'accéder au niveau inférieur des constructions. Les trappes d'évacuation sont utilisées pour étendre la construction aux fonctionnalités qui ne sont pas exposées dans la version actuelle de AWS mais disponibles dans CloudFormation.

Nous vous recommandons d'utiliser une trappe de secours dans les situations suivantes :

- Une AWS service fonctionnalité est disponible via CloudFormation, mais il n'existe aucune `Construct` structure pour celle-ci.
- Une AWS service fonctionnalité est disponible CloudFormation et il existe `Construct` des constructions pour le service, mais celles-ci n'exposent pas encore la fonctionnalité. Comme `Construct` les concepts sont développés « à la main », ils peuvent parfois être en retard par rapport aux concepts liés aux CloudFormation ressources.

L'exemple de code suivant montre un cas d'utilisation courant d'une trappe de secours. Dans cet exemple, la fonctionnalité qui n'est pas encore implémentée dans la construction de niveau supérieur permet d'ajouter `httpPutResponseHopLimit` pour la mise à l'échelle automatique de `LaunchConfiguration`.

```
const launchConfig = autoscaling.onDemandASG.node.findChild("LaunchConfig") as
 CfnLaunchConfiguration;
            launchConfig.metadataOptions = {
                   httpPutResponseHopLimit: autoscalingConfig.httpPutResponseHopLimit||
 2
            }
```

L'exemple de code précédent montre le flux de travail suivant :

1. Vous définissez votre `AutoScalingGroup` à l'aide d'une construction L2. La construction L2 ne prend pas en charge la mise à jour du`httpPutResponseHopLimit`, vous devez donc utiliser une trappe d'évacuation.
2. Vous accédez à la propriété `node.defaultChild` sur la construction `AutoScalingGroup` L2 et l'ajoutez comme ressource `CfnLaunchConfiguration`.
3. Vous pouvez à présent définir la propriété `launchConfig.metadataOptions` sur la `CfnLaunchConfiguration` L1.

Ressources personnalisées

Vous pouvez utiliser des ressources personnalisées pour écrire une logique de provisionnement personnalisée dans des modèles qui CloudFormation s'exécutent chaque fois que vous créez, mettez à jour (si vous avez modifié la ressource personnalisée) ou supprimez des piles. Par exemple, vous pouvez utiliser une ressource personnalisée si vous souhaitez inclure des ressources qui ne sont pas

disponibles dans le AWS CDK. De cette façon, vous pouvez continuer à gérer toutes les ressources connexes dans une seule pile.

La génération d'une ressource personnalisée implique l'écriture d'une fonction Lambda qui répond aux événements du cycle de vie CREATE, UPDATE et DELETE d'une ressource. Si votre ressource personnalisée ne doit effectuer qu'un seul appel d'API, envisagez d'utiliser la AwsCustomResourceconstruction. Cela permet d'effectuer des appels au SDK arbitraires lors d'un CloudFormation déploiement. Sinon, nous vous suggérons d'écrire votre propre fonction Lambda pour effectuer le travail que vous devez exécuter.

Pour plus d'informations sur les ressources personnalisées, consultez la section Ressources personnalisées dans la CloudFormation documentation. Pour un exemple d'utilisation d'une ressource personnalisée, consultez le référentiel de ressources personnalisées sur GitHub.

L'exemple suivant montre comment créer une classe de ressources personnalisée pour lancer une fonction Lambda et envoyer CloudFormation un signal de réussite ou d'échec.

```
import cdk = require('aws-cdk-lib');
import customResources = require('aws-cdk-lib/custom-resources');
import lambda = require('aws-cdk-lib/aws-lambda');
import { Construct } from 'constructs';

import fs = require('fs');

export interface MyCustomResourceProps {
  /**
   * Message to echo
   */
  message: string;
}

export class MyCustomResource extends Construct {
  public readonly response: string;

  constructor(scope: Construct, id: string, props: MyCustomResourceProps) {
    super(scope, id);

    const fn = new lambda.SingletonFunction(this, 'Singleton', {
      uuid: 'f7d4f730-4ee1-11e8-9c2d-fa7ae01bbebc',
      code: new lambda.InlineCode(fs.readFileSync('custom-resource-handler.py',
 { encoding: 'utf-8' })),
      handler: 'index.main',
```

```
      timeout: cdk.Duration.seconds(300),
      runtime: lambda.Runtime.PYTHON_3_6,
    });

    const provider = new customResources.Provider(this, 'Provider', {
      onEventHandler: fn,
    });

    const resource = new cdk.CustomResource(this, 'Resource', {
      serviceToken: provider.serviceToken,
      properties: props,
    });

    this.response = resource.getAtt('Response').toString();
    }
}
```

L'exemple suivant montre la logique principale de la ressource personnalisée.

```
def main(event, context):
    import logging as log
    import cfnresponse
    log.getLogger().setLevel(log.INFO)

    # This needs to change if there are to be multiple resources in the same stack
    physical_id = 'TheOnlyCustomResource'

    try:
        log.info('Input event: %s', event)

        # Check if this is a Create and we're failing Creates
        if event['RequestType'] == 'Create' and
 event['ResourceProperties'].get('FailCreate', False):
            raise RuntimeError('Create failure requested')

        # Do the thing
        message = event['ResourceProperties']['Message']
        attributes = {
            'Response': 'You said "%s"' % message
        }

        cfnresponse.send(event, context, cfnresponse.SUCCESS, attributes, physical_id)
    except Exception as e:
```

```
        log.exception(e)
        # cfnresponse's error message is always "see CloudWatch"
        cfnresponse.send(event, context, cfnresponse.FAILED, {}, physical_id)
```

L'exemple suivant montre comment la AWS CDK pile appelle la ressource personnalisée.

```
import cdk = require('aws-cdk-lib');
import { MyCustomResource } from './my-custom-resource';

/**
 * A stack that sets up MyCustomResource and shows how to get an attribute from it
 */
class MyStack extends cdk.Stack {
  constructor(scope: cdk.App, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    const resource = new MyCustomResource(this, 'DemoResource', {
      message: 'CustomResource says hello',
    });

    // Publish the custom resource output
    new cdk.CfnOutput(this, 'ResponseMessage', {
      description: 'The message that came back from the Custom Resource',
      value: resource.response
    });
  }
}

const app = new cdk.App();
new MyStack(app, 'CustomResourceDemoStack');
app.synth();
```

Suivez les TypeScript meilleures pratiques

TypeScript est un langage qui étend les fonctionnalités de JavaScript. C'est un langage fortement typé et orienté objet. Vous pouvez l'utiliser TypeScript pour spécifier les types de données transmis dans votre code et vous pouvez signaler des erreurs lorsque les types ne correspondent pas. Cette section fournit une vue d'ensemble des TypeScript meilleures pratiques.

Description de vos données

Vous pouvez l'utiliser TypeScript pour décrire la forme des objets et des fonctions de votre code. L'utilisation du type `any` équivaut à se désinscrire de la vérification du type pour une variable. Nous vous recommandons d'éviter d'utiliser `any` dans votre code. Voici un exemple.

```
type Result = "success" | "failure"
function verifyResult(result: Result) {
    if (result === "success") {
        console.log("Passed");
    } else {
        console.log("Failed")
    }
}
```

Utilisation d'énumérations

Vous pouvez utiliser des énumérations pour définir un ensemble de constantes nommées et définir des normes qui peuvent être réutilisées dans votre base de code. Nous vous recommandons d'exporter vos énumérations une seule fois au niveau global, puis de laisser les autres classes importer et utiliser les énumérations. Supposons que vous souhaitiez créer un ensemble d'actions possibles pour capturer les événements dans votre base de code. TypeScript fournit des énumérations numériques et basées sur des chaînes. L'exemple suivant utilise une énumération.

```
enum EventType {
    Create,
    Delete,
    Update
}

class InfraEvent {
    constructor(event: EventType) {
        if (event === EventType.Create) {
            // Call for other function
            console.log(`Event Captured :${event}`);
        }
    }
}

let eventSource: EventType = EventType.Create;
```

```
const eventExample = new InfraEvent(eventSource)
```

Utilisation d'interfaces

Une interface est un contrat pour la classe. Si vous créez un contrat, vos utilisateurs doivent le respecter. Dans l'exemple suivant, une interface est utilisée pour normaliser les `props` et veiller à ce que les appelants fournissent le paramètre attendu lors de l'utilisation de cette classe.

```
import { Stack, App } from "aws-cdk-lib";
import { Construct } from "constructs";

interface BucketProps {
    name: string;
    region: string;
    encryption: boolean;

}

class S3Bucket extends Stack {
    constructor(scope: Construct, props: BucketProps) {
        super(scope);
        console.log(props.name);

    }
}
const app = App();
const myS3Bucket = new S3Bucket(app, {
    name: "my-bucket",
    region: "us-east-1",
    encryption: false
})
```

Certaines propriétés ne peuvent être modifiées que lors de la création initiale d'un objet. Vous pouvez le spécifier en plaçant `readonly` avant le nom de la propriété, comme le montre l'exemple suivant.

```
interface Position {
    readonly latitude: number;
    readonly longitute: number;
}
```

Extension d'interfaces

L'extension des interfaces réduit les doublons, car il n'est pas nécessaire de copier les propriétés entre les interfaces. En outre, le lecteur de votre code peut facilement comprendre les relations au sein de votre application.

```
interface BaseInterface{
   name: string;
 }
 interface EncryptedVolume extends BaseInterface{
     keyName: string;
 }
 interface UnencryptedVolume extends BaseInterface {
     tags: string[];
 }
```

Évitement des interfaces vides

Nous vous recommandons d'éviter les interfaces vides en raison des risques potentiels qu'elles présentent. Dans l'exemple suivant, il existe une interface vide appelée`BucketProps`. Les objets `myS3Bucket1` et `myS3Bucket2` sont tous deux valides, mais ils suivent des normes différentes, car l'interface n'applique aucun contrat. Le code suivant compilera et imprimera les propriétés, mais cela introduit des incohérences dans votre application.

```
interface BucketProps {}

class S3Bucket implements BucketProps {
    constructor(props: BucketProps){
        console.log(props);
    }
}

const myS3Bucket1 = new S3Bucket({
    name: "my-bucket",
    region: "us-east-1",
    encryption: false,
});

const myS3Bucket2 = new S3Bucket({
    name: "my-bucket",
```

```
});
```

Utilisation d'usines

Dans un modèle Abstract Factory, une interface est chargée de créer une usine d'objets connexes sans spécifier explicitement leurs classes. Par exemple, vous pouvez créer une usine Lambda pour créer des fonctions Lambda. Au lieu de créer une nouvelle fonction Lambda dans votre construction, vous déléguez le processus de création à l'usine. Pour plus d'informations sur ce modèle de conception, consultez Abstract Factory TypeScript dans la documentation de Refactoring.Guru.

Utilisation de la déstructuration sur des propriétés

La déstructuration, introduite dans ECMAScript 6 (ES6), est une JavaScript fonctionnalité qui vous permet d'extraire plusieurs données d'un tableau ou d'un objet et de les attribuer à leurs propres variables.

```
const object = {
    objname: "obj",
    scope: "this",
};

const oName = object.objname;
const oScop = object.scope;

const { objname, scope } = object;
```

Définition des conventions de dénomination standard

L'application d'une convention de dénomination permet de maintenir la cohérence de la base de code et de réduire la surcharge lorsque vous réfléchissez à la manière de nommer une variable. Nous vous recommandons la procédure suivante :

- Utilisez CamelCase pour les noms de variables et de fonctions.
- PascalCase À utiliser pour les noms de classe et les noms d'interface.
- Utilisez CamelCase pour les membres de l'interface.
- PascalCase À utiliser pour les noms de type et les noms d'énumération.
- Nommez les fichiers avec CamelCase (par exemple, `ebsVolumes.tsx` ou `storage.tsb`).

N'utilisez pas le mot clé var

L'`let`instruction est utilisée pour déclarer une variable locale dans TypeScript. Il est similaire au `var` mot clé, mais sa portée est limitée par rapport au `var` mot clé. Une variable déclarée dans un bloc avec `let` est uniquement disponible en vue d'utilisation dans ce bloc. Le `var` mot clé ne peut pas être limité à un bloc, ce qui signifie qu'il est accessible en dehors d'un bloc particulier (représenté par`{}`) mais pas en dehors de la fonction dans laquelle il est défini. Vous pouvez redéclarer et mettre à jour les `var` variables. Il est recommandé d'éviter d'utiliser le `var` mot clé.

Envisager d'utiliser ESLint et Prettier

ESLint analyse statiquement votre code pour détecter rapidement les problèmes. Vous pouvez utiliser ESLint pour créer une série d'affirmations (appelées règles de validation) qui définissent l'apparence ou le comportement de votre code. ESLint propose également des suggestions de correction automatique pour vous aider à améliorer votre code. Enfin, vous pouvez utiliser ESLint pour charger des règles de validation à partir de plug-ins partagés.

Prettier est un formateur de code reconnu qui prend en charge une variété de langages de programmation différents. Vous pouvez utiliser Prettier pour définir votre style de code afin d'éviter de le formater manuellement. Après l'installation, vous pouvez mettre à jour votre fichier `package.json` et exécuter les commandes `npm run format` et `npm run lint`.

L'exemple suivant vous montre comment activer ESLint et le formateur Prettier pour votre projet. AWS CDK

```
"scripts": {
    "build": "tsc",
    "watch": "tsc -w",
    "test": "jest",
    "cdk": "cdk",
    "lint": "eslint --ext .js,.ts .",
    "format": "prettier --ignore-path .gitignore --write '**/*.+(js|ts|json)'"
}
```

Utilisation des modificateurs d'accès

Le modificateur privé dans TypeScript limite la visibilité à la même classe uniquement. Lorsque vous ajoutez le modificateur privé à une propriété ou à une méthode, vous pouvez accéder à cette propriété ou méthode au sein de la même classe.

Le modificateur public permet aux propriétés et aux méthodes des classes d'être accessibles depuis n'importe quel emplacement. Si vous ne spécifiez aucun modificateur d'accès pour les propriétés et les méthodes, ils utiliseront le modificateur public par défaut.

Le modificateur protégé permet aux propriétés et aux méthodes d'une classe d'être accessibles au sein de la même classe et des mêmes sous-classes. Utilisez le modificateur protégé lorsque vous prévoyez de créer des sous-classes dans votre AWS CDK application.

Utiliser des types d'utilitaires

Les types d'utilitaires TypeScript sont des fonctions de type prédéfinies qui effectuent des transformations et des opérations sur des types existants. Cela vous permet de créer de nouveaux types basés sur des types existants. Par exemple, vous pouvez modifier ou extraire des propriétés, les rendre facultatives ou obligatoires, ou créer des versions immuables de types. En utilisant des types d'utilitaires, vous pouvez définir des types plus précis et détecter les erreurs potentielles au moment de la compilation.

Partiel <Type>

`Partial`marque tous les membres d'un type d'entrée `Type` comme facultatifs. Cet utilitaire renvoie un type qui représente tous les sous-ensembles d'un type donné. Voici un exemple de `Partial`.

```
interface Dog {
  name: string;
  age: number;
  breed: string;
  weight: number;
}

let partialDog: Partial<Dog> = {};
```

Nécessaire <Type>

`Required`fait le contraire de`Partial`. Cela rend tous les membres d'un type d'entrée `Type` non facultatifs (en d'autres termes, obligatoires). Voici un exemple de `Required`.

```
interface Dog {
  name: string;
  age: number;
```

```
  breed: string;
  weight?: number;
}

let dog: Required<Dog> = {
  name: "scruffy",
  age: 5,
  breed: "labrador",
  weight 55 // "Required" forces weight to be defined
};
```

Rechercher les vulnérabilités de sécurité et les erreurs de formatage

L'infrastructure en tant que code (IaC) et l'automatisation sont devenues essentielles pour les entreprises. Compte tenu de la solidité d'IaC, vous avez une grande responsabilité dans la gestion des risques de sécurité. Les risques de sécurité courants d'IaC peuvent inclure les suivants :

- Privilèges trop permissifs AWS Identity and Access Management (IAM)
- Groupes de sécurité ouverts
- Ressources non chiffrées
- Journaux d'accès non activés

Approches et outils de sécurité

Nous vous recommandons d'implémenter les approches de sécurité suivantes :

- Détection des vulnérabilités dans le développement : la correction des vulnérabilités en production est coûteuse et prend du temps en raison de la complexité du développement et de la distribution de correctifs logiciels. En outre, les vulnérabilités de la production comportent un risque d'exploitation. Nous vous recommandons d'analyser le code de vos ressources IaC afin de détecter et de corriger les vulnérabilités avant leur mise en production.
- Conformité et correction automatique : AWS Config propose des règles AWS gérées. Ces règles vous aident à renforcer la conformité et vous permettent de tenter une correction automatique en utilisant l'AWS Systems Manager automatisation. Vous pouvez également créer et associer des documents d'automatisation personnalisés à l'aide de AWS Config règles.

Outils de développement courants

Les outils présentés dans cette section vous aident à étendre leurs fonctionnalités intégrées avec vos propres règles personnalisées. Nous vous recommandons d'aligner vos règles personnalisées sur les normes de votre organisation. Voici quelques outils de développement courants à prendre en compte :

- Utilisez cfn-nag pour identifier les problèmes de sécurité de l'infrastructure, tels que les règles IAM permissives ou les littéraux de mots de passe, dans les modèles. CloudFormation Pour plus d'informations, consultez le référentiel GitHub cfn-nag de Stelligent.
- Utilisez cdk-nag, inspiré de cfn-nag, pour vérifier que les constructions d'une portée donnée sont conformes à un ensemble défini de règles. Vous pouvez également utiliser cdk-nag pour la suppression de règles et les rapports de conformité. L'outil cdk-nag valide les constructions en étendant les aspects du. AWS CDK Pour plus d'informations, voir Gérer la sécurité et la conformité des applications avec le AWS Cloud Development Kit (AWS CDK) et cdk-nag dans le blog. AWS DevOps
- Utilisez l'outil open source Checkov pour effectuer une analyse statique sur votre environnement IaC. Checkov aide à identifier les erreurs de configuration du cloud en scannant le code de votre infrastructure dans Kubernetes, Terraform ou. CloudFormation Vous pouvez utiliser Checkov pour obtenir des sorties dans différents formats, notamment JSON, JUnit XML ou CLI. Checkov peut gérer efficacement les variables en créant un graphique qui montre la dépendance dynamique du code. Pour plus d'informations, consultez le référentiel GitHub Checkov de Bridgecrew.
- Utilisez TFLint pour vérifier les erreurs et les syntaxes obsolètes et pour vous aider à appliquer les bonnes pratiques. Notez qu'il se peut que TFLint ne valide pas les problèmes propres au fournisseur. Pour plus d'informations sur TFlint, consultez le référentiel GitHub TFlint de Terraform Linters.
- Utilisez Amazon Q Developer pour effectuer des analyses de sécurité. Lorsqu'il est utilisé dans un environnement de développement intégré (IDE), Amazon Q Developer fournit une assistance au développement logiciel basée sur l'IA. Il peut discuter du code, fournir des compléments de code en ligne, générer du nouveau code, scanner votre code pour détecter les failles de sécurité et apporter des mises à niveau et des améliorations au code.

Développer et affiner la documentation

La documentation est essentielle à la réussite de votre projet. Non seulement la documentation explique le fonctionnement de votre code, mais elle aide également les développeurs à mieux comprendre les caractéristiques et les fonctionnalités de vos applications. Le développement et le perfectionnement d'une documentation de haute qualité peuvent renforcer le processus de développement logiciel, maintenir des logiciels de haute qualité et faciliter le transfert de connaissances entre les développeurs.

Il existe deux catégories de documentation : la documentation contenue dans le code et la documentation associée relative au code. La documentation contenue dans le code se présente sous forme de commentaires. La documentation associée relative au code peut se composer des fichiers README et des documents externes. Il n'est pas rare que les développeurs considèrent la documentation comme une surcharge, car le code lui-même est facile à comprendre. Cela peut être vrai pour les petits projets, mais la documentation est cruciale pour les projets de grande envergure impliquant plusieurs équipes.

Il est recommandé que l'auteur du code rédige la documentation, car il en comprend bien les fonctionnalités. Les développeurs peuvent avoir du mal à supporter la surcharge liée à la maintenance de la documentation associée distincte. Pour relever ce défi, les développeurs peuvent ajouter les commentaires dans le code et ces commentaires peuvent être extraits automatiquement afin que chaque version du code et de la documentation soit synchronisée.

Il existe différents outils pour aider les développeurs à extraire les commentaires du code et à générer la documentation correspondante. Ce guide se concentre sur le fait TypeDoc qu'il s'agit de l'outil préféré pour AWS CDK les constructions.

Pourquoi la documentation du code est requise pour les AWS CDK constructions

AWS CDK les concepts communs sont créés par plusieurs équipes au sein d'une organisation et partagés entre différentes équipes à des fins d'utilisation. Une bonne documentation aide les utilisateurs de la bibliothèque de constructions à intégrer facilement les constructions et à créer leur infrastructure avec un minimum d'effort. La synchronisation de tous les documents est une tâche conséquente. Nous vous recommandons de conserver le document dans le code, qui sera extrait à l'aide de la TypeDoc bibliothèque.

Utilisation TypeDoc avec la bibliothèque AWS Construct

TypeDoc est un générateur de documents pour TypeScript. Vous pouvez l'utiliser TypeDoc pour lire vos fichiers TypeScript source, analyser les commentaires contenus dans ces fichiers, puis générer un site statique contenant la documentation de votre code.

Le code suivant vous montre comment intégrer TypeDoc la bibliothèque AWS Construct, puis ajouter les packages suivants dans votre `package.json` fichier`devDependencies`.

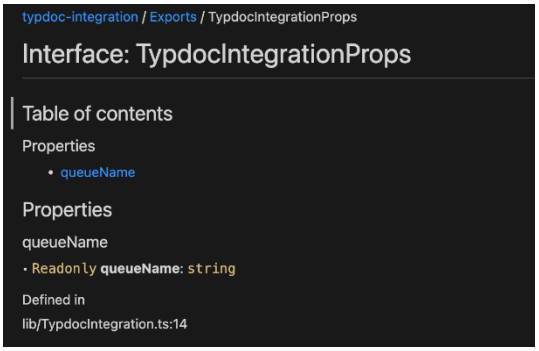
```
{

  "devDependencies": {

    "typedoc-plugin-markdown": "^3.11.7",
    "typescript": "~3.9.7"
  },

}
```

Pour ajouter `typedoc.json` dans le dossier de la bibliothèque CDK, utilisez le code suivant.

```
{
    "$schema": "https://typedoc.org/schema.json",
    "entryPoints": ["./lib"],
}
```

Pour générer les fichiers README, exécutez la `npx typedoc` commande dans le répertoire racine du projet de bibliothèque de AWS CDK construction.

L'exemple de document suivant est généré par TypeDoc.



Pour plus d'informations sur les options TypeDoc d'intégration, consultez les commentaires relatifs aux documents dans la TypeDoc documentation.

Adopter une approche de développement piloté par les tests

Nous vous recommandons de suivre une approche de développement piloté par les tests (TDD) avec le. AWS CDK Le TDD est une approche du développement logiciel dans laquelle vous développez des cas de test pour spécifier et valider votre code. En résumé, vous créez d'abord des cas de test pour chaque fonctionnalité et si le test échoue, vous écrivez le nouveau code pour transmettre le test, le simplifier et le rendre exempt de bogues.

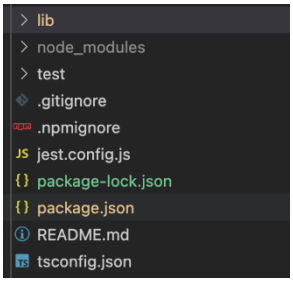
Vous pouvez d'abord utiliser l'approche de TDD pour écrire le scénario de test. Cela vous permet de valider l'infrastructure avec différentes contraintes de conception en termes d'application de la politique de sécurité pour les ressources et de respect d'une convention de dénomination unique pour le projet. L'approche standard pour tester AWS CDK les applications consiste à utiliser le module AWS CDK assertions et les frameworks de test populaires, tels que Jest pour JavaScript et/ou TypeScript pytest pour Python.

Il existe deux catégories de tests que vous pouvez rédiger pour vos AWS CDK applications :

- Utilisez des assertions détaillées pour tester un aspect spécifique du CloudFormation modèle généré, par exemple « cette ressource possède cette propriété avec cette valeur ». Ces tests peuvent détecter des régressions et sont également utiles lorsque vous développez de nouvelles fonctionnalités à l'aide de l'approche de TDD (écrivez d'abord un test, puis transmettez-le en écrivant une implémentation correcte). Les affirmations précises sont les tests que vous rédigerez le plus souvent.
- Utilisez des tests instantanés pour tester le modèle synthétisé par rapport à un CloudFormation modèle de référence précédemment stocké. Les tests d'instantanés permettent de refactoriser librement, car vous pouvez être sûr que le code refactorisé fonctionne exactement de la même manière que le code d'origine. Si les modifications étaient intentionnelles, vous pouvez accepter une nouvelle référence pour les futurs tests. Cependant, les AWS CDK mises à niveau peuvent également entraîner la modification des modèles synthétisés. Vous ne pouvez donc pas vous fier uniquement aux instantanés pour vous assurer que votre implémentation est correcte.

Test unitaire

Ce guide se concentre sur l'intégration des tests unitaires en TypeScript particulier. Pour activer les tests, assurez-vous que votre fichier `package.json` possède les bibliothèques suivantes : `@types/jest`, `jest` et `ts-jest` dans `devDependencies`. Pour ajouter ces packages, exécutez la commande `cdk init lib --language=typescript`. Une fois que vous avez exécuté la commande précédente, la structure suivante s'affiche.



Le code suivant est un exemple de `package.json` fichier activé avec la bibliothèque Jest.

```
{
  ...
  "scripts": {
    "build": "npm run lint && tsc",
    "watch": "tsc -w",
    "test": "jest",
  },
  "devDependencies": {
    ...
    "@types/jest": "27.5.2",
    "jest": "27.5.1",
    "ts-jest": "27.1.5",
    ...
  }
}
```

Sous le dossier Test, vous pouvez écrire le cas de test. L'exemple suivant montre un cas de test pour une AWS CodePipeline construction.

```
import {App,Stack} from 'aws-cdk-lib';
import { Template } from 'aws-cdk-lib/assertions';
import * as CodepipelineModule from '../lib/index';
import { Role, ServicePrincipal } from 'aws-cdk-lib/aws-iam';
import { Repository } from 'aws-cdk-lib/aws-codecommit';
```

```
import { PipelineProject } from 'aws-cdk-lib/aws-codebuild';

const testData:CodepipelineModule.CodepipelineModuleProps = {

    pipelineName: "validate-test-pipeline",
    serviceRoleARN: "",
    codeCommitRepositoryARN: "",
    branchName: "master",
    buildStages:[]
}


test('Code Pipeline Created', () => {
    const app = new App();
    const stack = new Stack(app, "TestStack");
    // WHEN
    const serviceRole = new Role(stack, "testRole", { assumedBy: new
 ServicePrincipal('codepipeline.amazonaws.com') })
    const codeCommit = new Repository(stack, "testRepo", {
        repositoryName: "validate-codeCommit-repo"
    });
    const codeBuildProject=new PipelineProject(stack,"TestCodeBuildProject",{});
    testData.serviceRoleARN = serviceRole.roleArn;
    testData.codeCommitRepositoryARN = codeCommit.repositoryArn;
    testData["buildStages"].push({
        stageName:"Deploy",
        codeBuildProject:codeBuildProject
    })
    new CodepipelineModule.CodepipelineModule(stack, 'MyTestConstruct', testData);
    // THEN
    const template = Template.fromStack(stack);

    template.hasResourceProperties('AWS::CodePipeline::Pipeline', {
        Name:testData.pipelineName
    });
});
```

Pour exécuter un test, exécutez la commande `npm run test` dans votre projet. Le test renvoie les résultats suivants.

```
PASS  test/codepipeline-module.test.ts (5.972 s)
  # Code Pipeline Created (97 ms)
Test Suites: 1 passed, 1 total
```

```
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        6.142 s, estimated 9 s
```

Pour plus d'informations sur les scénarios de test, consultez la section Tester des constructions dans le Guide du AWS Cloud Development Kit (AWS CDK) développeur.

Test d'intégration

Les tests d'intégration pour AWS CDK les constructions peuvent également être inclus à l'aide d'un `integ-tests` module. Un test d'intégration doit être défini comme une AWS CDK application. Il doit y avoir une one-to-one relation entre un test d'intégration et une AWS CDK application. Pour plus d'informations, consultez le integ-tests-alpha module dans la référence de AWS CDK l'API.

Utiliser le contrôle des éditions et des versions pour les constructions

Contrôle de version pour AWS CDK

AWS CDK des concepts communs peuvent être créés par plusieurs équipes et partagés au sein d'une organisation à des fins de consommation. Généralement, les développeurs publient de nouvelles fonctionnalités ou corrigent des bogues dans leurs AWS CDK constructions communes. Ces constructions sont utilisées par les AWS CDK applications ou toute autre AWS CDK construction existante dans le cadre d'une dépendance. Pour cette raison, il est crucial que les développeurs mettent à jour et publient indépendamment leur construction avec des versions sémantiques appropriées. AWS CDK Les applications en aval ou d'autres AWS CDK constructions peuvent mettre à jour leur dépendance pour utiliser la version de AWS CDK construction récemment publiée.

La gestion des versions sémantique (Semver) est un ensemble de règles, ou de méthodes, permettant de fournir des numéros de logiciel uniques au logiciel informatique. Les versions sont définies comme suit :

- Une version MAJEURE consiste en des modifications d'API incompatibles ou en une modification radicale.
- Une version MINEURE comprend des fonctionnalités ajoutées de manière rétrocompatible.
- Une version de CORRECTIF consiste en des corrections de bogues rétrocompatibles.

Pour plus d'informations sur le versionnement sémantique, consultez la section Spécification de version sémantique (SemVer) dans la documentation sur le versionnage sémantique.

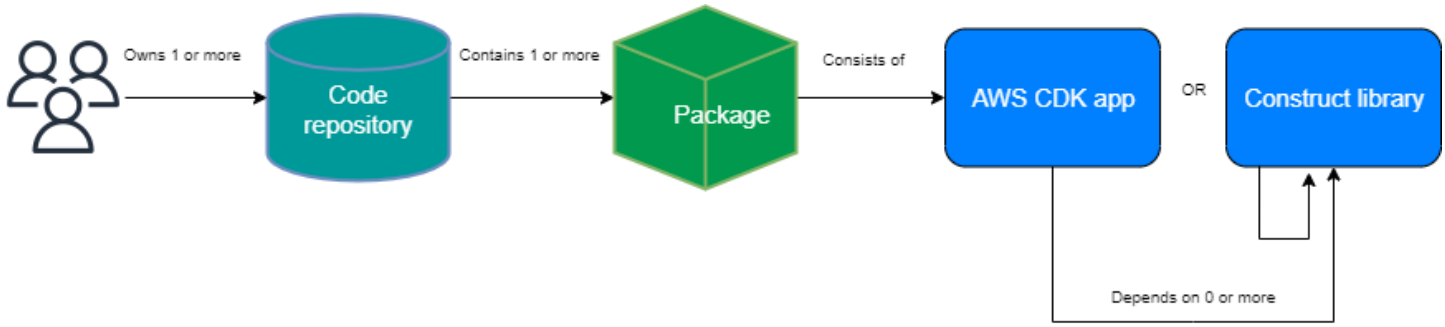
Référentiel et empaquetage pour les AWS CDK constructions

Comme AWS CDK les constructions sont développées par différentes équipes et utilisées par plusieurs AWS CDK applications, vous pouvez utiliser un référentiel distinct pour chaque AWS CDK construction. Cela peut également vous aider à appliquer le contrôle d'accès. Chaque dépôt peut contenir tout le code source lié à la même AWS CDK construction ainsi que toutes ses dépendances. En conservant une seule application (c'est-à-dire une AWS CDK construction) dans un référentiel unique, vous pouvez réduire l'impact des modifications lors du déploiement.

AWS CDK Non seulement il génère des CloudFormation modèles pour le déploiement de l'infrastructure, mais il regroupe également les actifs d'exécution tels que les fonctions Lambda et les images Docker et les déploie parallèlement à votre infrastructure. Il est non seulement possible de combiner le code qui définit votre infrastructure et le code qui implémente votre logique d'exécution en une seule construction, mais c'est également une bonne pratique. Ces deux types de code n'ont pas besoin de se trouver dans des référentiels séparés ni même dans des packages distincts.

Pour consommer des packages au-delà des limites du référentiel, vous devez disposer d'un référentiel de packages privé, similaire à npm ou à Maven Central PyPi, mais interne à votre organisation. Vous devez également disposer d'un processus de publication qui génère, teste et publie le package dans le référentiel de packages privé. Vous pouvez créer des référentiels privés, tels que des PyPi serveurs, à l'aide d'une machine virtuelle (VM) locale ou d'Amazon S3. Lorsque vous concevez ou créez un registre de packages privé, il est essentiel de prendre en compte le risque d'interruption de service en raison de la haute disponibilité et de la capacité de mise à l'échelle. Un service géré sans serveur hébergé dans le cloud pour stocker les packages peut considérablement réduire les frais de maintenance. Par exemple, vous pouvez l'utiliser AWS CodeArtifactpour héberger des packages pour les langages de programmation les plus courants. Vous pouvez également l'utiliser CodeArtifact pour définir des connexions au référentiel externe et les répliquer au sein CodeArtifact de celui-ci.

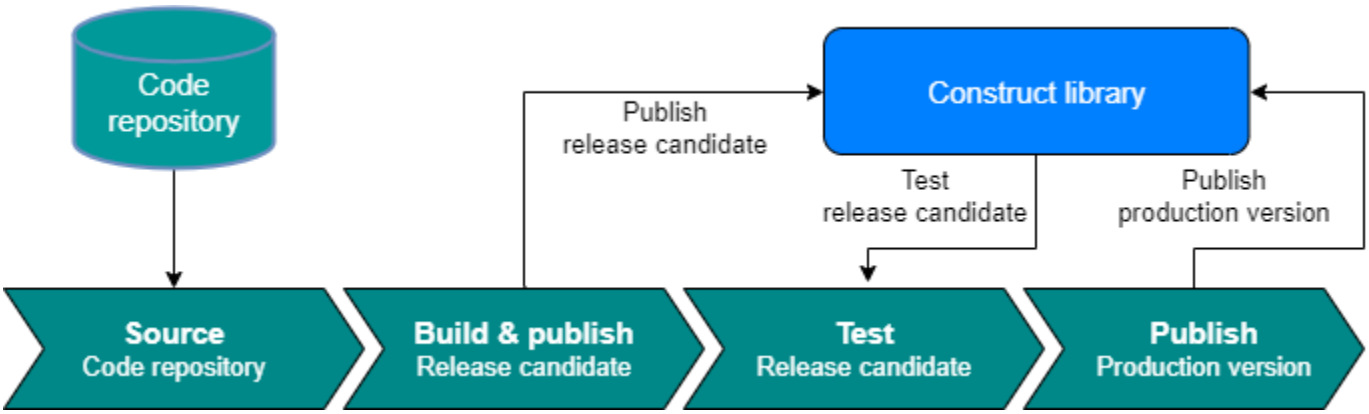
Les dépendances des packages du référentiel de packages sont gérées par le gestionnaire de packages de votre langue (par exemple, npm for TypeScript ou JavaScript applications). Votre gestionnaire de packages s'assure que les générations sont reproductibles en enregistrant les versions spécifiques de chaque package dont dépend votre application, puis vous permet de mettre à niveau ces dépendances de manière contrôlée, comme le montre le schéma suivant.



Constructez la publication pour le AWS CDK

Nous vous recommandons de créer votre propre pipeline automatisé pour créer et publier de nouvelles versions de AWS CDK construction. Si vous mettez en place un processus d'approbation des demandes d'extraction approprié, une fois que vous avez validé et transféré votre code source dans la branche principale du référentiel, le pipeline peut générer et créer une version possible. Cette version peut être transférée CodeArtifact et testée avant de publier la version prête pour la production. Vous pouvez éventuellement tester votre nouvelle version de AWS CDK construction localement avant de fusionner le code avec la branche principale. Ainsi, le pipeline va publier la version prête pour la production. Tenez compte du fait que les constructions et les packages partagés doivent être testés indépendamment de l'application utilisatrice, comme s'ils étaient mis à la disposition du public.

Le schéma suivant montre un exemple de pipeline de publication de AWS CDK versions.



Vous pouvez utiliser les exemples de commandes suivants pour générer, tester et publier des packages npm. Tout d'abord, connectez-vous au référentiel d'artefacts en exécutant la commande suivante.

```
aws codeartifact login --tool npm --domain <Domain Name> --domain-owner $(aws sts get-
caller-identity --output text --query 'Account')  \
```

```
--repository <Repository Name> --region <AWS Region Name>
```

Ensuite, procédez comme suit :

1. Installez les packages requis sur la base du fichier `package.json` : `npm install`
2. Créez la version possible pour la publication : `npm version prerelease --preid rc`
3. Générez le package npm : `npm run build`
4. Testez le package npm : `npm run test`
5. Publiez le package npm : `npm publish`

Appliquer la gestion des versions de la bibliothèque

La gestion du cycle de vie représente un défi majeur lorsque vous gérez des bases de AWS CDK code. Supposons, par exemple, que vous démarriez un AWS CDK projet avec la version 1.97, puis que la version 1.169 soit disponible ultérieurement. La version 1.169 propose de nouvelles fonctionnalités et des corrections de bogues, mais vous avez déployé votre infrastructure à l'aide de l'ancienne version. Aujourd'hui, la mise à jour des constructions devient complexe, car cet écart augmente en raison des modifications majeures qui pourraient être introduites dans les nouvelles versions. Cela peut s'avérer compliqué si vous disposez de nombreuses ressources dans votre environnement. Le modèle présenté dans cette section peut vous aider à gérer la version de votre AWS CDK bibliothèque à l'aide de l'automatisation. Voici le flux de travail de ce modèle :

1. Lorsque vous lancez un nouveau produit CodeArtifact Service Catalog, les versions de la AWS CDK bibliothèque et ses dépendances sont stockées dans le `package.json` fichier.
2. Vous déployez un pipeline commun qui assure le suivi de tous les référentiels afin de pouvoir leur appliquer des mises à niveau automatiques en l'absence de modifications importantes.
3. Une AWS CodeBuild étape vérifie la présence de l'arbre de dépendances et recherche les modifications les plus importantes.
4. Le pipeline crée une branche de fonctionnalités, puis exécute `cdk synth` avec la nouvelle version pour confirmer qu'il n'y a aucune erreur.
5. La nouvelle version est déployée dans l'environnement de test et exécute enfin un test d'intégration pour s'assurer que le déploiement est sain.
6. Vous pouvez utiliser deux files Amazon Simple Queue Service (Amazon SQS) pour suivre les piles. Les utilisateurs peuvent examiner les piles manuellement dans la file des exceptions et

corriger les modifications importantes. Les éléments qui réussissent le test d'intégration peuvent être fusionnés et publiés.

FAQ

Quels problèmes peuvent être TypeScript résolus ?

Généralement, vous pouvez éliminer les bogues dans votre code en écrivant des tests automatisés, en vérifiant manuellement que le code fonctionne comme prévu, puis en demandant à une autre personne de valider votre code. La validation des connexions entre chaque partie de votre projet est une tâche chronophage. Pour accélérer le processus de validation, vous pouvez utiliser un langage de type vérifié, par exemple TypeScript pour automatiser la validation du code et fournir un feedback instantané pendant le développement.

Pourquoi devrais-je l'utiliser TypeScript ?

TypeScript est un langage open source qui simplifie le JavaScript code, le rendant plus facile à lire et à déboguer. TypeScript fournit également des outils de développement hautement productifs pour les JavaScript IDE et les pratiques, telles que la vérification statique. TypeScriptIl offre également les avantages d'ECMAScript 6 (ES6) et peut augmenter votre productivité. Enfin, cela TypeScript peut vous aider à éviter les bogues pénibles que les développeurs rencontrent fréquemment lors de l'écriture JavaScript en vérifiant le type du code.

Dois-je utiliser le AWS CDK ou CloudFormation ?

Nous vous recommandons d'utiliser le AWS Cloud Development Kit (AWS CDK) plutôt queAWS CloudFormation, si votre organisation possède l'expertise en développement nécessaire pour tirer parti duAWS CDK. C'est parce que AWS CDK c'est plus flexible que CloudFormation, puisque vous pouvez utiliser un langage de programmation et des concepts OOP. N'oubliez pas que vous pouvez utiliser CloudFormation pour créer AWS des ressources de manière ordonnée et prévisible. Dans CloudFormation, les ressources sont écrites dans des fichiers texte au format JSON ou YAML.

Et si le AWS CDK ne prend pas en charge un nouveau lancement AWS service ?

Vous pouvez utiliser une dérogation brute ou une ressource CloudFormation personnalisée.

Quels sont les différents langages de programmation pris en charge par le AWS CDK ?

AWS CDKest généralement disponible en Python JavaScript TypeScript, Java, C# et Go (dans Developer Preview).

Combien cela AWS CDK coûte-t-il ?

Il n'y a aucun frais supplémentaire pour leAWS CDK. Vous payez pour les AWS ressources (telles que les instances Amazon EC2 ou les équilibreurs de charge Elastic Load Balancing) créées lorsque vous les utilisez AWS CDK de la même manière que si vous les créiez manuellement. Vous ne payez qu'à la demande et à l'utilisation. Aucun frais minimum ni aucun engagement initial requis ne s'appliquent.

Étapes suivantes

Nous vous recommandons de commencer à construire avec AWS Cloud Development Kit (AWS CDK) TypeScript. Pour plus d'informations, consultez l'atelier AWS CDK d'une journée d'immersion.

Ressources

Références

- AWS Constructions de AWS solutions (solutions)
- AWS Cloud Development Kit (AWS CDK) (GitHub)
- AWS Référence de l'API Construct Library (documentation de AWS CDK référence)
- AWS CDK Documentation de référence (documentation AWS CDK de référence)
- AWS CDK Atelier d'une journée d'immersion (AWS Workshop Studio)

Outils

- sac cdk () GitHub
- TypeScript ESLint (documentation TypeScript ESLint)

Guides et modèles

- AWS Solutions Construit des modèles (AWS documentation)

Historique du document

Le tableau suivant décrit les modifications importantes apportées à ce guide. Pour être averti des mises à jour à venir, abonnez-vous à un fil RSS.

Modification	Description	Date
Code de mise à jour	Nous avons mis à jour les exemples de code dans la section Suivre les TypeScript meilleures pratiques.	16 février 2024
Ajouter des sections	Nous avons ajouté les sections Utiliser les types d'utilitaires et Test d'intégration.	10 janvier 2024
Mise à jour mineure	Exemple de code mis à jour pour créer une construction L3.	16 juin 2023
Publication initiale	—	8 décembre 2022

AWS Glossaire des directives prescriptives

Les termes suivants sont couramment utilisés dans les stratégies, les guides et les modèles fournis par les directives AWS prescriptives. Pour suggérer des entrées, veuillez utiliser le lien Faire un commentaire à la fin du glossaire.

Nombres

7 R

Sept politiques de migration courantes pour transférer des applications vers le cloud. Ces politiques s'appuient sur les 5 R identifiés par Gartner en 2011 et sont les suivantes :

- Refactorisation/réarchitecture : transférez une application et modifiez son architecture en tirant pleinement parti des fonctionnalités natives cloud pour améliorer l'agilité, les performances et la capacité de mise à l'échelle. Cela implique généralement le transfert du système d'exploitation et de la base de données. Exemple : migrez votre base de données Oracle sur site vers l'édition compatible avec Amazon Aurora PostgreSQL.
- Replateformer (déplacer et remodeler) : transférez une application vers le cloud et introduisez un certain niveau d'optimisation pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Amazon Relational Database Service (Amazon RDS) pour Oracle dans le. AWS Cloud
- Racheter (rachat) : optez pour un autre produit, généralement en passant d'une licence traditionnelle à un modèle SaaS. Exemple : migrez votre système de gestion de la relation client (CRM) vers Salesforce.com.
- Réhéberger (lift and shift) : transférez une application vers le cloud sans apporter de modifications pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Oracle sur une instance EC2 dans le. AWS Cloud
- Relocaliser (lift and shift au niveau de l'hyperviseur) : transférez l'infrastructure vers le cloud sans acheter de nouveau matériel, réécrire des applications ou modifier vos opérations existantes. Vous migrez des serveurs d'une plateforme sur site vers un service cloud pour la même plateforme. Exemple : migrer une Microsoft Hyper-V application vers AWS.
- Retenir : conservez les applications dans votre environnement source. Il peut s'agir d'applications nécessitant une refactorisation majeure, que vous souhaitez retarder, et d'applications existantes que vous souhaitez retenir, car rien ne justifie leur migration sur le plan commercial.

- Retirer : mettez hors service ou supprimez les applications dont vous n'avez plus besoin dans votre environnement source.

A

ABAC

Voir contrôle d'accès basé sur les attributs.

services abstraits

Consultez la section Services gérés.

ACIDE

Voir atomicité, consistance, isolation, durabilité.

migration active-active

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue (à l'aide d'un outil de réplication bidirectionnelle ou d'opérations d'écriture double), tandis que les deux bases de données gèrent les transactions provenant de la connexion d'applications pendant la migration. Cette méthode prend en charge la migration par petits lots contrôlés au lieu d'exiger un basculement ponctuel. Elle est plus flexible mais demande plus de travail qu'une migration active-passive.

migration active-passive

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue, mais seule la base de données source gère les transactions provenant de la connexion d'applications pendant que les données sont répliquées vers la base de données cible. La base de données cible n'accepte aucune transaction pendant la migration.

fonction d'agrégation

Fonction SQL qui agit sur un groupe de lignes et calcule une valeur de retour unique pour le groupe. Des exemples de fonctions d'agrégation incluent `SUM` et`MAX`.

AI

Voir intelligence artificielle.

AIOps

Voir les opérations d'intelligence artificielle.

anonymisation

Processus de suppression définitive d'informations personnelles dans un ensemble de données. L'anonymisation peut contribuer à protéger la vie privée. Les données anonymisées ne sont plus considérées comme des données personnelles.

anti-motif

Solution fréquemment utilisée pour un problème récurrent lorsque la solution est contre-productive, inefficace ou moins efficace qu'une solution alternative.

contrôle des applications

Une approche de sécurité qui permet d'utiliser uniquement des applications approuvées afin de protéger un système contre les logiciels malveillants.

portefeuille d'applications

Ensemble d'informations détaillées sur chaque application utilisée par une organisation, y compris le coût de génération et de maintenance de l'application, ainsi que sa valeur métier. Ces informations sont essentielles pour le processus de découverte et d'analyse du portefeuille et permettent d'identifier et de prioriser les applications à migrer, à moderniser et à optimiser.

intelligence artificielle (IA)

Domaine de l'informatique consacré à l'utilisation des technologies de calcul pour exécuter des fonctions cognitives généralement associées aux humains, telles que l'apprentissage, la résolution de problèmes et la reconnaissance de modèles. Pour plus d'informations, veuillez consulter Qu'est-ce que l'intelligence artificielle ?

opérations d'intelligence artificielle (AIOps)

Processus consistant à utiliser des techniques de machine learning pour résoudre les problèmes opérationnels, réduire les incidents opérationnels et les interventions humaines, mais aussi améliorer la qualité du service. Pour plus d'informations sur la façon dont les AIOps sont utilisées dans la stratégie de migration AWS , veuillez consulter le guide d'intégration des opérations.

chiffrement asymétrique

Algorithme de chiffrement qui utilise une paire de clés, une clé publique pour le chiffrement et une clé privée pour le déchiffrement. Vous pouvez partager la clé publique, car elle n'est pas utilisée pour le déchiffrement, mais l'accès à la clé privée doit être très restreint.

atomicité, cohérence, isolement, durabilité (ACID)

Ensemble de propriétés logicielles garantissant la validité des données et la fiabilité opérationnelle d'une base de données, même en cas d'erreur, de panne de courant ou d'autres problèmes.

contrôle d'accès par attributs (ABAC)

Pratique qui consiste à créer des autorisations détaillées en fonction des attributs de l'utilisateur, tels que le service, le poste et le nom de l'équipe. Pour plus d'informations, consultez ABAC pour AWS dans la documentation AWS Identity and Access Management (IAM).

source de données faisant autorité

Emplacement où vous stockez la version principale des données, considérée comme la source d'information la plus fiable. Vous pouvez copier les données de la source de données officielle vers d'autres emplacements à des fins de traitement ou de modification des données, par exemple en les anonymisant, en les expurgant ou en les pseudonymisant.

Zone de disponibilité

Un emplacement distinct au sein d'un Région AWS réseau isolé des défaillances dans d'autres zones de disponibilité et fournissant une connectivité réseau peu coûteuse et à faible latence aux autres zones de disponibilité de la même région.

AWS Cadre d'adoption du cloud (AWS CAF)

Un cadre de directives et de meilleures pratiques visant AWS à aider les entreprises à élaborer un plan efficace pour réussir leur migration vers le cloud. AWS La CAF organise ses conseils en six domaines prioritaires appelés perspectives : les affaires, les personnes, la gouvernance, les plateformes, la sécurité et les opérations. Les perspectives d'entreprise, de personnes et de gouvernance mettent l'accent sur les compétences et les processus métier, tandis que les perspectives relatives à la plateforme, à la sécurité et aux opérations se concentrent sur les compétences et les processus techniques. Par exemple, la perspective liée aux personnes cible les parties prenantes qui s'occupent des ressources humaines (RH), des fonctions de dotation en personnel et de la gestion des personnes. Dans cette perspective, la AWS CAF fournit des conseils pour le développement du personnel, la formation et les communications afin de préparer

l'organisation à une adoption réussie du cloud. Pour plus d'informations, veuillez consulter le site Web AWS CAF et le livre blanc AWS CAF.

AWS Cadre de qualification de la charge de travail (AWS WQF)

Outil qui évalue les charges de travail liées à la migration des bases de données, recommande des stratégies de migration et fournit des estimations de travail. AWS Le WQF est inclus avec AWS Schema Conversion Tool (AWS SCT). Il analyse les schémas de base de données et les objets de code, le code d'application, les dépendances et les caractéristiques de performance, et fournit des rapports d'évaluation.

B

mauvais bot

Un bot destiné à perturber ou à nuire à des individus ou à des organisations.

BCP

Consultez la section Planification de la continuité des activités.

graphique de comportement

Vue unifiée et interactive des comportements des ressources et des interactions au fil du temps. Vous pouvez utiliser un graphique de comportement avec Amazon Detective pour examiner les tentatives de connexion infructueuses, les appels d'API suspects et les actions similaires. Pour plus d'informations, veuillez consulter Data in a behavior graph dans la documentation Detective.

système de poids fort

Système qui stocke d'abord l'octet le plus significatif. Voir aussi endianité.

classification binaire

Processus qui prédit un résultat binaire (l'une des deux classes possibles). Par exemple, votre modèle de machine learning peut avoir besoin de prévoir des problèmes tels que « Cet e-mail est-il du spam ou non ? » ou « Ce produit est-il un livre ou une voiture ? ».

filtre de Bloom

Structure de données probabiliste et efficace en termes de mémoire qui est utilisée pour tester si un élément fait partie d'un ensemble.

déploiement bleu/vert

Stratégie de déploiement dans laquelle vous créez deux environnements distincts mais identiques. Vous exécutez la version actuelle de l'application dans un environnement (bleu) et la nouvelle version de l'application dans l'autre environnement (vert). Cette stratégie vous permet de revenir rapidement en arrière avec un impact minimal.

bot

Application logicielle qui exécute des tâches automatisées sur Internet et simule l'activité ou l'interaction humaine. Certains robots sont utiles ou bénéfiques, comme les robots d'exploration Web qui indexent des informations sur Internet. D'autres robots, connus sous le nom de mauvais robots, sont destinés à perturber ou à nuire à des individus ou à des organisations.

botnet

Réseaux de robots infectés par des logiciels malveillants et contrôlés par une seule entité, connue sous le nom d'herder ou d'opérateur de bots. Les botnets sont le mécanisme le plus connu pour faire évoluer les bots et leur impact.

branche

Zone contenue d'un référentiel de code. La première branche créée dans un référentiel est la branche principale. Vous pouvez créer une branche à partir d'une branche existante, puis développer des fonctionnalités ou corriger des bogues dans la nouvelle branche. Une branche que vous créez pour générer une fonctionnalité est communément appelée branche de fonctionnalités. Lorsque la fonctionnalité est prête à être publiée, vous fusionnez à nouveau la branche de fonctionnalités dans la branche principale. Pour plus d'informations, consultez À propos des branches (GitHub documentation).

accès par brise-vitre

Dans des circonstances exceptionnelles et par le biais d'un processus approuvé, c'est un moyen rapide pour un utilisateur d'accéder à un accès auquel Compte AWS il n'est généralement pas autorisé. Pour plus d'informations, consultez l'indicateur Implementation break-glass procedures dans le guide Well-Architected AWS .

stratégie existante (brownfield)

L'infrastructure existante de votre environnement. Lorsque vous adoptez une stratégie existante pour une architecture système, vous concevez l'architecture en fonction des contraintes des systèmes et de l'infrastructure actuels. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

cache de tampon

Zone de mémoire dans laquelle sont stockées les données les plus fréquemment consultées.

capacité métier

Ce que fait une entreprise pour générer de la valeur (par exemple, les ventes, le service client ou le marketing). Les architectures de microservices et les décisions de développement peuvent être dictées par les capacités métier. Pour plus d'informations, veuillez consulter la section Organisation en fonction des capacités métier du livre blanc Exécution de microservices conteneurisés sur AWS.

planification de la continuité des activités (BCP)

Plan qui tient compte de l'impact potentiel d'un événement perturbateur, tel qu'une migration à grande échelle, sur les opérations, et qui permet à une entreprise de reprendre ses activités rapidement.

C

CAF

Voir le cadre d'adoption du AWS cloud.

déploiement de Canary

Diffusion lente et progressive d'une version pour les utilisateurs finaux. Lorsque vous êtes sûr, vous déployez la nouvelle version et remplacez la version actuelle dans son intégralité.

CCoE

Voir le Centre d'excellence du cloud.

CDC

Consultez la section Capture des données de modification.

capture des données de modification (CDC)

Processus de suivi des modifications apportées à une source de données, telle qu'une table de base de données, et d'enregistrement des métadonnées relatives à ces modifications. Vous pouvez utiliser la CDC à diverses fins, telles que l'audit ou la réplication des modifications dans un système cible afin de maintenir la synchronisation.

ingénierie du chaos

Introduire intentionnellement des défaillances ou des événements perturbateurs pour tester la résilience d'un système. Vous pouvez utiliser AWS Fault Injection Service (AWS FIS) pour effectuer des expériences qui stressent vos AWS charges de travail et évaluer leur réponse.

CI/CD

Découvrez l'intégration continue et la livraison continue.

classification

Processus de catégorisation qui permet de générer des prédictions. Les modèles de ML pour les problèmes de classification prédisent une valeur discrète. Les valeurs discrètes se distinguent toujours les unes des autres. Par exemple, un modèle peut avoir besoin d'évaluer la présence ou non d'une voiture sur une image.

chiffrement côté client

Chiffrement des données localement, avant que la cible ne les AWS service reçoive.

Centre d'excellence cloud (CCoE)

Une équipe multidisciplinaire qui dirige les efforts d'adoption du cloud au sein d'une organisation, notamment en développant les bonnes pratiques en matière de cloud, en mobilisant des ressources, en établissant des délais de migration et en guidant l'organisation dans le cadre de transformations à grande échelle. Pour plus d'informations, consultez les articles du CCoE sur le blog de stratégie AWS Cloud d'entreprise.

cloud computing

Technologie cloud généralement utilisée pour le stockage de données à distance et la gestion des appareils IoT. Le cloud computing est généralement associé à la technologie informatique de pointe.

modèle d'exploitation du cloud

Dans une organisation informatique, modèle d'exploitation utilisé pour créer, faire évoluer et optimiser un ou plusieurs environnements cloud. Pour plus d'informations, consultez la section Création de votre modèle d'exploitation cloud.

étapes d'adoption du cloud

Les quatre phases que les entreprises traversent généralement lorsqu'elles migrent vers AWS Cloud :

- Projet : exécution de quelques projets liés au cloud à des fins de preuve de concept et d'apprentissage
- Base : réaliser des investissements fondamentaux pour mettre à l'échelle l'adoption du cloud (par exemple, en créant une zone de destination, en définissant un CCoE ou en établissant un modèle opérationnel)
- Migration : migration d'applications individuelles
- Réinvention : optimisation des produits et services et innovation dans le cloud

Ces étapes ont été définies par Stephen Orban dans le billet de blog The Journey Toward Cloud-First & the Stages of Adoption publié sur le blog AWS Cloud Enterprise Strategy. Pour plus d'informations sur leur lien avec la stratégie de AWS migration, consultez le guide de préparation à la migration.

CMDB

Voir base de données de gestion de configuration.

référentiel de code

Emplacement où le code source et d'autres ressources, comme la documentation, les exemples et les scripts, sont stockés et mis à jour par le biais de processus de contrôle de version. Les référentiels cloud courants incluent GitHub ou AWS CodeCommit. Chaque version du code est appelée branche. Dans une structure de microservice, chaque référentiel est consacré à une seule fonctionnalité. Un seul pipeline CI/CD peut utiliser plusieurs référentiels.

cache passif

Cache tampon vide, mal rempli ou contenant des données obsolètes ou non pertinentes. Cela affecte les performances, car l'instance de base de données doit lire à partir de la mémoire principale ou du disque, ce qui est plus lent que la lecture à partir du cache tampon.

données gelées

Données rarement consultées et généralement historiques. Lorsque vous interrogez ce type de données, les requêtes lentes sont généralement acceptables. Le transfert de ces données vers des niveaux ou classes de stockage moins performants et moins coûteux peut réduire les coûts.

vision par ordinateur (CV)

Domaine de l'IA qui utilise l'apprentissage automatique pour analyser et extraire des informations à partir de formats visuels tels que des images numériques et des vidéos. Par exemple, AWS

Panorama propose des appareils qui ajoutent des CV aux réseaux de caméras locaux, et Amazon SageMaker fournit des algorithmes de traitement d'image pour les CV.

dérive de configuration

Pour une charge de travail, une modification de configuration par rapport à l'état attendu. Cela peut entraîner une non-conformité de la charge de travail, et cela est généralement progressif et involontaire.

base de données de gestion des configurations (CMDB)

Référentiel qui stocke et gère les informations relatives à une base de données et à son environnement informatique, y compris les composants matériels et logiciels ainsi que leurs configurations. Vous utilisez généralement les données d'une CMDB lors de la phase de découverte et d'analyse du portefeuille de la migration.

pack de conformité

Ensemble de AWS Config règles et d'actions correctives que vous pouvez assembler pour personnaliser vos contrôles de conformité et de sécurité. Vous pouvez déployer un pack de conformité en tant qu'entité unique dans une région Compte AWS et, ou au sein d'une organisation, à l'aide d'un modèle YAML. Pour plus d'informations, consultez la section Packs de conformité dans la AWS Config documentation.

intégration continue et livraison continue (CI/CD)

Processus d'automatisation des étapes source, de génération, de test, intermédiaire et de production du processus de publication du logiciel. CI/CD est communément décrit comme un pipeline. CI/CD peut vous aider à automatiser les processus, à améliorer la productivité, à améliorer la qualité du code et à accélérer les livraisons. Pour plus d'informations, veuillez consulter Avantages de la livraison continue. CD peut également signifier déploiement continu. Pour plus d'informations, veuillez consulter Livraison continue et déploiement continu.

CV

Voir vision par ordinateur.

D

données au repos

Données stationnaires dans votre réseau, telles que les données stockées.

classification des données

Processus permettant d'identifier et de catégoriser les données de votre réseau en fonction de leur sévérité et de leur sensibilité. Il s'agit d'un élément essentiel de toute stratégie de gestion des risques de cybersécurité, car il vous aide à déterminer les contrôles de protection et de conservation appropriés pour les données. La classification des données est une composante du pilier de sécurité du AWS Well-Architected Framework. Pour plus d'informations, veuillez consulter Classification des données.

dérive des données

Une variation significative entre les données de production et les données utilisées pour entraîner un modèle ML, ou une modification significative des données d'entrée au fil du temps. La dérive des données peut réduire la qualité, la précision et l'équité globales des prédictions des modèles ML.

données en transit

Données qui circulent activement sur votre réseau, par exemple entre les ressources du réseau.

maillage de données

Un cadre architectural qui fournit une propriété des données distribuée et décentralisée avec une gestion et une gouvernance centralisées.

minimisation des données

Le principe de collecte et de traitement des seules données strictement nécessaires. La pratique de la minimisation des données AWS Cloud peut réduire les risques liés à la confidentialité, les coûts et l'empreinte carbone de vos analyses.

périmètre de données

Ensemble de garde-fous préventifs dans votre AWS environnement qui permettent de garantir que seules les identités fiables accèdent aux ressources fiables des réseaux attendus. Pour plus d'informations, voir Création d'un périmètre de données sur AWS.

prétraitement des données

Pour transformer les données brutes en un format facile à analyser par votre modèle de ML. Le prétraitement des données peut impliquer la suppression de certaines colonnes ou lignes et le traitement des valeurs manquantes, incohérentes ou en double.

provenance des données

Le processus de suivi de l'origine et de l'historique des données tout au long de leur cycle de vie, par exemple la manière dont les données ont été générées, transmises et stockées.

sujet des données

Personne dont les données sont collectées et traitées.

entrepôt des données

Un système de gestion des données qui prend en charge les informations commerciales, telles que les analyses. Les entrepôts de données contiennent généralement de grandes quantités de données historiques et sont généralement utilisés pour les requêtes et les analyses.

langage de définition de base de données (DDL)

Instructions ou commandes permettant de créer ou de modifier la structure des tables et des objets dans une base de données.

langage de manipulation de base de données (DML)

Instructions ou commandes permettant de modifier (insérer, mettre à jour et supprimer) des informations dans une base de données.

DDL

Voir langage de définition de base de données.

ensemble profond

Sert à combiner plusieurs modèles de deep learning à des fins de prédiction. Vous pouvez utiliser des ensembles profonds pour obtenir une prévision plus précise ou pour estimer l'incertitude des prédictions.

deep learning

Un sous-champ de ML qui utilise plusieurs couches de réseaux neuronaux artificiels pour identifier le mappage entre les données d'entrée et les variables cibles d'intérêt.

defense-in-depth

Approche de la sécurité de l'information dans laquelle une série de mécanismes et de contrôles de sécurité sont judicieusement répartis sur l'ensemble d'un réseau informatique afin de protéger la confidentialité, l'intégrité et la disponibilité du réseau et des données qu'il contient. Lorsque vous adoptez cette stratégie AWS, vous ajoutez plusieurs contrôles à différentes couches de

la AWS Organizations structure afin de sécuriser les ressources. Par exemple, une defense-in-depth approche peut combiner l'authentification multifactorielle, la segmentation du réseau et le chiffrement.

administrateur délégué

Dans AWS Organizations, un service compatible peut enregistrer un compte AWS membre pour administrer les comptes de l'organisation et gérer les autorisations pour ce service. Ce compte est appelé administrateur délégué pour ce service. Pour plus d'informations et une liste des services compatibles, veuillez consulter la rubrique Services qui fonctionnent avec AWS Organizations dans la documentation AWS Organizations .

déploiement

Processus de mise à disposition d'une application, de nouvelles fonctionnalités ou de corrections de code dans l'environnement cible. Le déploiement implique la mise en œuvre de modifications dans une base de code, puis la génération et l'exécution de cette base de code dans les environnements de l'application.

environnement de développement

Voir environnement.

contrôle de détection

Contrôle de sécurité conçu pour détecter, journaliser et alerter après la survenue d'un événement. Ces contrôles constituent une deuxième ligne de défense et vous alertent en cas d'événements de sécurité qui ont contourné les contrôles préventifs en place. Pour plus d'informations, veuillez consulter la rubrique Contrôles de détection dans Implementing security controls on AWS.

cartographie de la chaîne de valeur du développement (DVSM)

Processus utilisé pour identifier et hiérarchiser les contraintes qui nuisent à la rapidité et à la qualité du cycle de vie du développement logiciel. DVSM étend le processus de cartographie de la chaîne de valeur initialement conçu pour les pratiques de production allégée. Il met l'accent sur les étapes et les équipes nécessaires pour créer et transférer de la valeur tout au long du processus de développement logiciel.

jumeau numérique

Représentation virtuelle d'un système réel, tel qu'un bâtiment, une usine, un équipement industriel ou une ligne de production. Les jumeaux numériques prennent en charge la maintenance prédictive, la surveillance à distance et l'optimisation de la production.

tableau des dimensions

Dans un schéma en étoile, table plus petite contenant les attributs de données relatifs aux données quantitatives d'une table de faits. Les attributs des tables de dimensions sont généralement des champs de texte ou des nombres discrets qui se comportent comme du texte. Ces attributs sont couramment utilisés pour la contrainte des requêtes, le filtrage et l'étiquetage des ensembles de résultats.

catastrophe

Un événement qui empêche une charge de travail ou un système d'atteindre ses objectifs commerciaux sur son site de déploiement principal. Ces événements peuvent être des catastrophes naturelles, des défaillances techniques ou le résultat d'actions humaines, telles qu'une mauvaise configuration involontaire ou une attaque de logiciel malveillant.

reprise après sinistre (DR)

La stratégie et le processus que vous utilisez pour minimiser les temps d'arrêt et les pertes de données causés par un sinistre. Pour plus d'informations, consultez Disaster Recovery of Workloads on AWS : Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Voir langage de manipulation de base de données.

conception axée sur le domaine

Approche visant à développer un système logiciel complexe en connectant ses composants à des domaines évolutifs, ou objectifs métier essentiels, que sert chaque composant. Ce concept a été introduit par Eric Evans dans son ouvrage Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston : Addison-Wesley Professional, 2003). Pour plus d'informations sur l'utilisation du design piloté par domaine avec le modèle de figuier étrangleur, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

Consultez la section Reprise après sinistre.

détection de dérive

Suivi des écarts par rapport à une configuration de référence. Par exemple, vous pouvez l'utiliser AWS CloudFormation pour détecter la dérive des ressources du système ou AWS Control Tower

pour détecter les modifications de votre zone d'atterrissage susceptibles d'affecter le respect des exigences de gouvernance.

DVSM

Voir la cartographie de la chaîne de valeur du développement.

E

EDA

Voir analyse exploratoire des données.

informatique de périphérie

Technologie qui augmente la puissance de calcul des appareils intelligents en périphérie d'un réseau IoT. Comparé au cloud computing, l'informatique de pointe peut réduire la latence des communications et améliorer le temps de réponse.

chiffrement

Processus informatique qui transforme des données en texte clair, lisibles par l'homme, en texte chiffré.

clé de chiffrement

Chaîne cryptographique de bits aléatoires générée par un algorithme cryptographique. La longueur des clés peut varier, et chaque clé est conçue pour être imprévisible et unique.

endianisme

Ordre selon lequel les octets sont stockés dans la mémoire de l'ordinateur. Les systèmes de poids fort stockent d'abord l'octet le plus significatif. Les systèmes de poids faible stockent d'abord l'octet le moins significatif.

point de terminaison

Voir point de terminaison de service.

service de point de terminaison

Service que vous pouvez héberger sur un cloud privé virtuel (VPC) pour le partager avec d'autres utilisateurs. Vous pouvez créer un service de point de terminaison avec AWS PrivateLink et accorder des autorisations à d'autres principaux Comptes AWS ou à AWS Identity and Access Management (IAM) principaux. Ces comptes ou principaux peuvent se connecter à votre

service de point de terminaison de manière privée en créant des points de terminaison d'un VPC d'interface. Pour plus d'informations, veuillez consulter Création d'un service de point de terminaison dans la documentation Amazon Virtual Private Cloud (Amazon VPC).

planification des ressources d'entreprise (ERP)

Système qui automatise et gère les principaux processus métier (tels que la comptabilité, le MES et la gestion de projet) pour une entreprise.

chiffrement d'enveloppe

Processus de chiffrement d'une clé de chiffrement à l'aide d'une autre clé de chiffrement. Pour plus d'informations, consultez la section Chiffrement des enveloppes dans la documentation AWS Key Management Service (AWS KMS).

environment

Instance d'une application en cours d'exécution. Les types d'environnement les plus courants dans le cloud computing sont les suivants :

- Environnement de développement : instance d'une application en cours d'exécution à laquelle seule l'équipe principale chargée de la maintenance de l'application peut accéder. Les environnements de développement sont utilisés pour tester les modifications avant de les promouvoir dans les environnements supérieurs. Ce type d'environnement est parfois appelé environnement de test.
- Environnements inférieurs : tous les environnements de développement d'une application, tels que ceux utilisés pour les générations et les tests initiaux.
- Environnement de production : instance d'une application en cours d'exécution à laquelle les utilisateurs finaux peuvent accéder. Dans un pipeline CI/CD, l'environnement de production est le dernier environnement de déploiement.
- Environnements supérieurs : tous les environnements accessibles aux utilisateurs autres que l'équipe de développement principale. Ils peuvent inclure un environnement de production, des environnements de préproduction et des environnements pour les tests d'acceptation par les utilisateurs.

épopée

Dans les méthodologies agiles, catégories fonctionnelles qui aident à organiser et à prioriser votre travail. Les épopées fournissent une description détaillée des exigences et des tâches d'implémentation. Par exemple, les points forts de la AWS CAF en matière de sécurité incluent la gestion des identités et des accès, les contrôles de détection, la sécurité des infrastructures,

la protection des données et la réponse aux incidents. Pour plus d'informations sur les épopées dans la stratégie de migration AWS , veuillez consulter le guide d'implémentation du programme.

ERP

Voir Planification des ressources d'entreprise.

analyse exploratoire des données (EDA)

Processus d'analyse d'un jeu de données pour comprendre ses principales caractéristiques. Vous collectez ou agrégez des données, puis vous effectuez des enquêtes initiales pour trouver des modèles, détecter des anomalies et vérifier les hypothèses. L'EDA est réalisée en calculant des statistiques récapitulatives et en créant des visualisations de données.

F

tableau des faits

La table centrale dans un schéma en étoile. Il stocke des données quantitatives sur les opérations commerciales. Généralement, une table de faits contient deux types de colonnes : celles qui contiennent des mesures et celles qui contiennent une clé étrangère pour une table de dimensions.

échouer rapidement

Une philosophie qui utilise des tests fréquents et progressifs pour réduire le cycle de vie du développement. C'est un élément essentiel d'une approche agile.

limite d'isolation des défauts

Dans le AWS Cloud, une limite telle qu'une zone de disponibilité Région AWS, un plan de contrôle ou un plan de données qui limite l'effet d'une panne et contribue à améliorer la résilience des charges de travail. Pour plus d'informations, consultez la section Limites d'isolation des AWS pannes.

branche de fonctionnalités

Voir la succursale.

fonctionnalités

Les données d'entrée que vous utilisez pour faire une prédiction. Par exemple, dans un contexte de fabrication, les fonctionnalités peuvent être des images capturées périodiquement à partir de la ligne de fabrication.

importance des fonctionnalités

Le niveau d'importance d'une fonctionnalité pour les prédictions d'un modèle. Il s'exprime généralement sous la forme d'un score numérique qui peut être calculé à l'aide de différentes techniques, telles que la méthode Shapley Additive Explanations (SHAP) et les gradients intégrés. Pour plus d'informations, voir Interprétabilité du modèle d'apprentissage automatique avec :AWS.

transformation de fonctionnalité

Optimiser les données pour le processus de ML, notamment en enrichissant les données avec des sources supplémentaires, en mettant à l'échelle les valeurs ou en extrayant plusieurs ensembles d'informations à partir d'un seul champ de données. Cela permet au modèle de ML de tirer parti des données. Par exemple, si vous décomposez la date « 2021-05-27 00:15:37 » en « 2021 », « mai », « jeudi » et « 15 », vous pouvez aider l'algorithme d'apprentissage à apprendre des modèles nuancés associés à différents composants de données.

FGAC

Découvrez le contrôle d'accès détaillé.

contrôle d'accès détaillé (FGAC)

Utilisation de plusieurs conditions pour autoriser ou refuser une demande d'accès.

migration instantanée (flash-cut)

Méthode de migration de base de données qui utilise la réplication continue des données via la capture des données de modification afin de migrer les données dans les plus brefs délais, au lieu d'utiliser une approche progressive. L'objectif est de réduire au maximum les temps d'arrêt.

G

blocage géographique

Voir les restrictions géographiques.

restrictions géographiques (blocage géographique)

Sur Amazon CloudFront, option permettant d'empêcher les utilisateurs de certains pays d'accéder aux distributions de contenu. Vous pouvez utiliser une liste d'autorisation ou une liste de blocage pour spécifier les pays approuvés et interdits. Pour plus d'informations, consultez la section Restreindre la distribution géographique de votre contenu dans la CloudFront documentation.

Flux de travail Gitflow

Approche dans laquelle les environnements inférieurs et supérieurs utilisent différentes branches dans un référentiel de code source. Le flux de travail Gitflow est considéré comme existant, et le flux de travail basé sur les troncs est l'approche moderne préférée.

stratégie inédite

L'absence d'infrastructures existantes dans un nouvel environnement. Lorsque vous adoptez une stratégie inédite pour une architecture système, vous pouvez sélectionner toutes les nouvelles technologies sans restriction de compatibilité avec l'infrastructure existante, également appelée brownfield. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

barrière de protection

Règle de haut niveau qui permet de régir les ressources, les politiques et la conformité au sein des unités d'organisation (UO). Les barrières de protection préventives appliquent des politiques pour garantir l'alignement sur les normes de conformité. Elles sont mises en œuvre à l'aide de politiques de contrôle des services et de limites des autorisations IAM. Les barrières de protection de détection détectent les violations des politiques et les problèmes de conformité, et génèrent des alertes pour y remédier. Ils sont implémentés à l'aide d'Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, d'Amazon Inspector et de AWS Lambda contrôles personnalisés.

H

HA

Découvrez la haute disponibilité.

migration de base de données hétérogène

Migration de votre base de données source vers une base de données cible qui utilise un moteur de base de données différent (par exemple, Oracle vers Amazon Aurora). La migration hétérogène fait généralement partie d'un effort de réarchitecture, et la conversion du schéma peut s'avérer une tâche complexe. AWS propose AWS SCT qui facilite les conversions de schémas.

haute disponibilité (HA)

Capacité d'une charge de travail à fonctionner en continu, sans intervention, en cas de difficultés ou de catastrophes. Les systèmes HA sont conçus pour basculer automatiquement, fournir

constamment des performances de haute qualité et gérer différentes charges et défaillances avec un impact minimal sur les performances.

modernisation de l'historien

Approche utilisée pour moderniser et mettre à niveau les systèmes de technologie opérationnelle (OT) afin de mieux répondre aux besoins de l'industrie manufacturière. Un historien est un type de base de données utilisé pour collecter et stocker des données provenant de diverses sources dans une usine.

migration de base de données homogène

Migration de votre base de données source vers une base de données cible qui partage le même moteur de base de données (par exemple, Microsoft SQL Server vers Amazon RDS for SQL Server). La migration homogène s'inscrit généralement dans le cadre d'un effort de réhébergement ou de replateforme. Vous pouvez utiliser les utilitaires de base de données natifs pour migrer le schéma.

données chaudes

Données fréquemment consultées, telles que les données en temps réel ou les données translationnelles récentes. Ces données nécessitent généralement un niveau ou une classe de stockage à hautes performances pour fournir des réponses rapides aux requêtes.

correctif

Solution d'urgence à un problème critique dans un environnement de production. En raison de son urgence, un correctif est généralement créé en dehors du flux de travail de DevOps publication habituel.

période de soins intensifs

Immédiatement après le basculement, période pendant laquelle une équipe de migration gère et surveille les applications migrées dans le cloud afin de résoudre les problèmes éventuels. En règle générale, cette période dure de 1 à 4 jours. À la fin de la période de soins intensifs, l'équipe de migration transfère généralement la responsabilité des applications à l'équipe des opérations cloud.

I

IaC

Considérez l'infrastructure comme un code.

politique basée sur l'identité

Politique attachée à un ou plusieurs principaux IAM qui définit leurs autorisations au sein de l' AWS Cloud environnement.

application inactive

Application dont l'utilisation moyenne du processeur et de la mémoire se situe entre 5 et 20 % sur une période de 90 jours. Dans un projet de migration, il est courant de retirer ces applications ou de les retenir sur site.

IIoT

Voir Internet industriel des objets.

infrastructure immuable

Modèle qui déploie une nouvelle infrastructure pour les charges de travail de production au lieu de mettre à jour, d'appliquer des correctifs ou de modifier l'infrastructure existante. Les infrastructures immuables sont intrinsèquement plus cohérentes, fiables et prévisibles que les infrastructures mutables. Pour plus d'informations, consultez les meilleures pratiques de déploiement à l'aide d'une infrastructure immuable dans le AWS Well-Architected Framework.

VPC entrant (d'entrée)

Dans une architecture AWS multi-comptes, un VPC qui accepte, inspecte et achemine les connexions réseau depuis l'extérieur d'une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

migration incrémentielle

Stratégie de basculement dans le cadre de laquelle vous migrez votre application par petites parties au lieu d'effectuer un basculement complet unique. Par exemple, il se peut que vous ne transfériez que quelques microservices ou utilisateurs vers le nouveau système dans un

premier temps. Après avoir vérifié que tout fonctionne correctement, vous pouvez transférer progressivement des microservices ou des utilisateurs supplémentaires jusqu'à ce que vous puissiez mettre hors service votre système hérité. Cette stratégie réduit les risques associés aux migrations de grande ampleur.

Industry 4.0

Terme introduit par Klaus Schwab en 2016 pour désigner la modernisation des processus de fabrication grâce aux avancées en matière de connectivité, de données en temps réel, d'automatisation, d'analyse et d'IA/ML.

infrastructure

Ensemble des ressources et des actifs contenus dans l'environnement d'une application.

infrastructure en tant que code (IaC)

Processus de mise en service et de gestion de l'infrastructure d'une application via un ensemble de fichiers de configuration. IaC est conçue pour vous aider à centraliser la gestion de l'infrastructure, à normaliser les ressources et à mettre à l'échelle rapidement afin que les nouveaux environnements soient reproductibles, fiables et cohérents.

internet industriel des objets (IIoT)

L'utilisation de capteurs et d'appareils connectés à Internet dans les secteurs industriels tels que la fabrication, l'énergie, l'automobile, les soins de santé, les sciences de la vie et l'agriculture. Pour plus d'informations, veuillez consulter Building an industrial Internet of Things (IIoT) digital transformation strategy.

VPC d'inspection

Dans une architecture AWS multi-comptes, un VPC centralisé qui gère les inspections du trafic réseau entre les VPC (identiques ou Régions AWS différents), Internet et les réseaux sur site. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

Internet des objets (IoT)

Réseau d'objets physiques connectés dotés de capteurs ou de processeurs intégrés qui communiquent avec d'autres appareils et systèmes via Internet ou via un réseau de communication local. Pour plus d'informations, veuillez consulter la section Qu'est-ce que l'IoT ?.

interprétabilité

Caractéristique d'un modèle de machine learning qui décrit dans quelle mesure un être humain peut comprendre comment les prédictions du modèle dépendent de ses entrées. Pour plus d'informations, veuillez consulter Machine learning model interpretability with AWS.

IoT

Voir Internet des objets.

Bibliothèque d'informations informatiques (ITIL)

Ensemble de bonnes pratiques pour proposer des services informatiques et les aligner sur les exigences métier. L'ITIL constitue la base de l'ITSM.

gestion des services informatiques (ITSM)

Activités associées à la conception, à la mise en œuvre, à la gestion et à la prise en charge de services informatiques d'une organisation. Pour plus d'informations sur l'intégration des opérations cloud aux outils ITSM, veuillez consulter le guide d'intégration des opérations.

ITIL

Consultez la bibliothèque d'informations informatiques.

ITSM

Consultez la section Gestion des services informatiques.

L

contrôle d'accès basé sur des étiquettes (LBAC)

Une implémentation du contrôle d'accès obligatoire (MAC) dans laquelle une valeur d'étiquette de sécurité est explicitement attribuée aux utilisateurs et aux données elles-mêmes. L'intersection entre l'étiquette de sécurité utilisateur et l'étiquette de sécurité des données détermine les lignes et les colonnes visibles par l'utilisateur.

zone de destination

Une zone d'atterrissage est un AWS environnement multi-comptes bien conçu, évolutif et sécurisé. Il s'agit d'un point de départ à partir duquel vos entreprises peuvent rapidement lancer et déployer des charges de travail et des applications en toute confiance dans leur environnement

de sécurité et d'infrastructure. Pour plus d'informations sur les zones de destination, veuillez consulter Setting up a secure and scalable multi-account AWS environment.

migration de grande envergure

Migration de 300 serveurs ou plus.

LBAC

Voir contrôle d'accès basé sur des étiquettes.

principe de moindre privilège

Bonne pratique de sécurité qui consiste à accorder les autorisations minimales nécessaires à l'exécution d'une tâche. Pour plus d'informations, veuillez consulter la rubrique Accorder les autorisations de moindre privilège dans la documentation IAM.

lift and shift

Voir 7 Rs.

système de poids faible

Système qui stocke d'abord l'octet le moins significatif. Voir aussi endianité.

environnements inférieurs

Voir environnement.

M

machine learning (ML)

Type d'intelligence artificielle qui utilise des algorithmes et des techniques pour la reconnaissance et l'apprentissage de modèles. Le ML analyse et apprend à partir de données enregistrées, telles que les données de l'Internet des objets (IoT), pour générer un modèle statistique basé sur des modèles. Pour plus d'informations, veuillez consulter Machine Learning.

branche principale

Voir la succursale.

malware

Logiciel conçu pour compromettre la sécurité ou la confidentialité de l'ordinateur. Les logiciels malveillants peuvent perturber les systèmes informatiques, divulguer des informations sensibles

ou obtenir un accès non autorisé. Parmi les malwares, on peut citer les virus, les vers, les rançongiciels, les chevaux de Troie, les logiciels espions et les enregistreurs de frappe.

services gérés

AWS services qui AWS gère la couche d'infrastructure, le système d'exploitation et les plateformes, et vous accédez aux points de terminaison pour stocker et récupérer des données. Amazon Simple Storage Service (Amazon S3) et Amazon DynamoDB sont des exemples de services gérés. Ils sont également connus sous le nom de services abstraits.

système d'exécution de la fabrication (MES)

Un système logiciel pour le suivi, la surveillance, la documentation et le contrôle des processus de production qui convertissent les matières premières en produits finis dans l'atelier.

MAP

Voir Migration Acceleration Program.

mécanisme

Processus complet au cours duquel vous créez un outil, favorisez son adoption, puis inspectez les résultats afin de procéder aux ajustements nécessaires. Un mécanisme est un cycle qui se renforce et s'améliore lorsqu'il fonctionne. Pour plus d'informations, voir Création de mécanismes dans le cadre AWS Well-Architected.

compte membre

Tous, à l' Comptes AWS exception du compte de gestion, qui font partie d'une organisation dans AWS Organizations. Un compte ne peut être membre que d'une seule organisation à la fois.

MAILLES

Voir le système d'exécution de la fabrication.

Transport télémétrique en file d'attente de messages (MQTT)

Protocole de communication léger machine-to-machine (M2M), basé sur le modèle de publication/d'abonnement, pour les appareils IoT aux ressources limitées.

microservice

Petit service indépendant qui communique via des API bien définies et qui est généralement détenu par de petites équipes autonomes. Par exemple, un système d'assurance peut inclure des microservices qui mappent à des capacités métier, telles que les ventes ou le marketing, ou

à des sous-domaines, tels que les achats, les réclamations ou l'analytique. Les avantages des microservices incluent l'agilité, la flexibilité de la mise à l'échelle, la facilité de déploiement, la réutilisation du code et la résilience. Pour plus d'informations, consultez la section Intégration de microservices à l'aide de services AWS sans serveur.

architecture de microservices

Approche de création d'une application avec des composants indépendants qui exécutent chaque processus d'application en tant que microservice. Ces microservices communiquent via une interface bien définie à l'aide d'API légères. Chaque microservice de cette architecture peut être mis à jour, déployé et mis à l'échelle pour répondre à la demande de fonctions spécifiques d'une application. Pour plus d'informations, consultez la section Implémentation de microservices sur AWS.

Programme d'accélération des migrations (MAP)

Un AWS programme qui fournit un support de conseil, des formations et des services pour aider les entreprises à établir une base opérationnelle solide pour passer au cloud, et pour aider à compenser le coût initial des migrations. MAP inclut une méthodologie de migration pour exécuter les migrations héritées de manière méthodique, ainsi qu'un ensemble d'outils pour automatiser et accélérer les scénarios de migration courants.

migration à grande échelle

Processus consistant à transférer la majeure partie du portefeuille d'applications vers le cloud par vagues, un plus grand nombre d'applications étant déplacées plus rapidement à chaque vague. Cette phase utilise les bonnes pratiques et les enseignements tirés des phases précédentes pour implémenter une usine de migration d'équipes, d'outils et de processus en vue de rationaliser la migration des charges de travail grâce à l'automatisation et à la livraison agile. Il s'agit de la troisième phase de la stratégie de migration AWS.

usine de migration

Équipes interfonctionnelles qui rationalisent la migration des charges de travail grâce à des approches automatisées et agiles. Les équipes de Migration Factory comprennent généralement les opérations, les analystes commerciaux et les propriétaires, les ingénieurs de migration, les développeurs et les DevOps professionnels travaillant dans le cadre de sprints. Entre 20 et 50 % du portefeuille d'applications d'entreprise est constitué de modèles répétés qui peuvent être optimisés par une approche d'usine. Pour plus d'informations, veuillez consulter la rubrique discussion of migration factories et le guide Cloud Migration Factory dans cet ensemble de contenus.

métadonnées de migration

Informations relatives à l'application et au serveur nécessaires pour finaliser la migration. Chaque modèle de migration nécessite un ensemble de métadonnées de migration différent. Les exemples de métadonnées de migration incluent le sous-réseau cible, le groupe de sécurité et le AWS compte.

modèle de migration

Tâche de migration reproductible qui détaille la stratégie de migration, la destination de la migration et l'application ou le service de migration utilisé. Exemple : réorganisez la migration vers Amazon EC2 AWS avec le service de migration d'applications.

Évaluation du portefeuille de migration (MPA)

Outil en ligne qui fournit des informations pour valider l'analyse de rentabilisation en faveur de la migration vers le. AWS Cloud La MPA propose une évaluation détaillée du portefeuille (dimensionnement approprié des serveurs, tarification, comparaison du coût total de possession, analyse des coûts de migration), ainsi que la planification de la migration (analyse et collecte des données d'applications, regroupement des applications, priorisation des migrations et planification des vagues). L'outil MPA (connexion requise) est disponible gratuitement pour tous les AWS consultants et consultants APN Partner.

Évaluation de la préparation à la migration (MRA)

Processus qui consiste à obtenir des informations sur l'état de préparation d'une organisation au cloud, à identifier les forces et les faiblesses et à élaborer un plan d'action pour combler les lacunes identifiées, à l'aide du AWS CAF. Pour plus d'informations, veuillez consulter le guide de préparation à la migration. La MRA est la première phase de la stratégie de migration AWS.

stratégie de migration

L'approche utilisée pour migrer une charge de travail vers le AWS Cloud. Pour plus d'informations, reportez-vous aux 7 R de ce glossaire et à Mobiliser votre organisation pour accélérer les migrations à grande échelle.

ML

Voir apprentissage automatique.

modernisation

Transformation d'une application obsolète (héritée ou monolithique) et de son infrastructure en un système agile, élastique et hautement disponible dans le cloud afin de réduire les coûts, de

gagner en efficacité et de tirer parti des innovations. Pour plus d'informations, consultez la section Stratégie de modernisation des applications dans le AWS Cloud.

évaluation de la préparation à la modernisation

Évaluation qui permet de déterminer si les applications d'une organisation sont prêtes à être modernisées, d'identifier les avantages, les risques et les dépendances, et qui détermine dans quelle mesure l'organisation peut prendre en charge l'état futur de ces applications. Le résultat de l'évaluation est un plan de l'architecture cible, une feuille de route détaillant les phases de développement et les étapes du processus de modernisation, ainsi qu'un plan d'action pour combler les lacunes identifiées. Pour plus d'informations, consultez la section Évaluation de l'état de préparation à la modernisation des applications dans le AWS Cloud.

applications monolithiques (monolithes)

Applications qui s'exécutent en tant que service unique avec des processus étroitement couplés. Les applications monolithiques ont plusieurs inconvénients. Si une fonctionnalité de l'application connaît un pic de demande, l'architecture entière doit être mise à l'échelle. L'ajout ou l'amélioration des fonctionnalités d'une application monolithique devient également plus complexe lorsque la base de code s'élargit. Pour résoudre ces problèmes, vous pouvez utiliser une architecture de microservices. Pour plus d'informations, veuillez consulter Decomposing monoliths into microservices.

MPA

Voir Évaluation du portefeuille de migration.

MQTT

Voir Message Queuing Telemetry Transport.

classification multi-classes

Processus qui permet de générer des prédictions pour plusieurs classes (prédiction d'un résultat parmi plus de deux). Par exemple, un modèle de ML peut demander « Ce produit est-il un livre, une voiture ou un téléphone ? » ou « Quelle catégorie de produits intéresse le plus ce client ? ».

infrastructure mutable

Modèle qui met à jour et modifie l'infrastructure existante pour les charges de travail de production. Pour améliorer la cohérence, la fiabilité et la prévisibilité, le AWS Well-Architected Framework recommande l'utilisation d'une infrastructure immuable comme meilleure pratique.

O

OAC

Voir Contrôle d'accès à l'origine.

OAI

Voir l'identité d'accès à l'origine.

OCM

Voir gestion du changement organisationnel.

migration hors ligne

Méthode de migration dans laquelle la charge de travail source est supprimée au cours du processus de migration. Cette méthode implique un temps d'arrêt prolongé et est généralement utilisée pour de petites charges de travail non critiques.

OI

Consultez la section Intégration des opérations.

OLA

Voir l'accord au niveau opérationnel.

migration en ligne

Méthode de migration dans laquelle la charge de travail source est copiée sur le système cible sans être mise hors ligne. Les applications connectées à la charge de travail peuvent continuer à fonctionner pendant la migration. Cette méthode implique un temps d'arrêt nul ou minimal et est généralement utilisée pour les charges de travail de production critiques.

OPC-UA

Voir Open Process Communications - Architecture unifiée.

Communications par processus ouvert - Architecture unifiée (OPC-UA)

Un protocole de communication machine-to-machine (M2M) pour l'automatisation industrielle. L'OPC-UA fournit une norme d'interopérabilité avec des schémas de cryptage, d'authentification et d'autorisation des données.

accord au niveau opérationnel (OLA)

Accord qui précise ce que les groupes informatiques fonctionnels s'engagent à fournir les uns aux autres, afin de prendre en charge un contrat de niveau de service (SLA).

examen de l'état de préparation opérationnelle (ORR)

Une liste de questions et de bonnes pratiques associées qui vous aident à comprendre, évaluer, prévenir ou réduire l'ampleur des incidents et des défaillances possibles. Pour plus d'informations, voir Operational Readiness Reviews (ORR) dans le AWS Well-Architected Framework.

technologie opérationnelle (OT)

Systèmes matériels et logiciels qui fonctionnent avec l'environnement physique pour contrôler les opérations, les équipements et les infrastructures industriels. Dans le secteur manufacturier, l'intégration des systèmes OT et des technologies de l'information (IT) est au cœur des transformations de l'industrie 4.0.

intégration des opérations (OI)

Processus de modernisation des opérations dans le cloud, qui implique la planification de la préparation, l'automatisation et l'intégration. Pour en savoir plus, veuillez consulter le guide d'intégration des opérations.

journal de suivi d'organisation

Un parcours créé par AWS CloudTrail qui enregistre tous les événements pour tous les membres Comptes AWS d'une organisation dans AWS Organizations. Ce journal de suivi est créé dans chaque Compte AWS qui fait partie de l'organisation et suit l'activité de chaque compte. Pour plus d'informations, consultez la section Création d'un suivi pour une organisation dans la CloudTrail documentation.

gestion du changement organisationnel (OCM)

Cadre pour gérer les transformations métier majeures et perturbatrices du point de vue des personnes, de la culture et du leadership. L'OCM aide les organisations à se préparer et à effectuer la transition vers de nouveaux systèmes et de nouvelles politiques en accélérant l'adoption des changements, en abordant les problèmes de transition et en favorisant des changements culturels et organisationnels. Dans la stratégie de AWS migration, ce cadre est appelé accélération du personnel, en raison de la rapidité du changement requise dans les projets d'adoption du cloud. Pour plus d'informations, veuillez consulter le guide OCM.

contrôle d'accès d'origine (OAC)

Dans CloudFront, une option améliorée pour restreindre l'accès afin de sécuriser votre contenu Amazon Simple Storage Service (Amazon S3). L'OAC prend en charge tous les compartiments S3 dans leur ensemble Régions AWS, le chiffrement côté serveur avec AWS KMS (SSE-KMS) et les requêtes dynamiques `PUT` adressées au compartiment S3. `DELETE`

identité d'accès d'origine (OAI)

Dans CloudFront, une option permettant de restreindre l'accès afin de sécuriser votre contenu Amazon S3. Lorsque vous utilisez OAI, il CloudFront crée un principal auprès duquel Amazon S3 peut s'authentifier. Les principaux authentifiés ne peuvent accéder au contenu d'un compartiment S3 que par le biais d'une distribution spécifique CloudFront . Voir également OAC, qui fournit un contrôle d'accès plus précis et amélioré.

OU

Voir l'examen de l'état de préparation opérationnelle.

DE

Voir technologie opérationnelle.

VPC sortant (de sortie)

Dans une architecture AWS multi-comptes, un VPC qui gère les connexions réseau initiées depuis une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

P

limite des autorisations

Politique de gestion IAM attachée aux principaux IAM pour définir les autorisations maximales que peut avoir l'utilisateur ou le rôle. Pour plus d'informations, veuillez consulter la rubrique Limites des autorisations dans la documentation IAM.

informations personnelles identifiables (PII)

Informations qui, lorsqu'elles sont consultées directement ou associées à d'autres données connexes, peuvent être utilisées pour déduire raisonnablement l'identité d'une personne. Les

exemples d'informations personnelles incluent les noms, les adresses et les informations de contact.

PII

Voir les informations personnelles identifiables.

manuel stratégique

Ensemble d'étapes prédéfinies qui capturent le travail associé aux migrations, comme la fourniture de fonctions d'opérations de base dans le cloud. Un manuel stratégique peut revêtir la forme de scripts, de runbooks automatisés ou d'un résumé des processus ou des étapes nécessaires au fonctionnement de votre environnement modernisé.

PLC

Voir contrôleur logique programmable.

PLM

Consultez la section Gestion du cycle de vie des produits.

politique

Objet capable de définir les autorisations (voir la politique basée sur l'identité), de spécifier les conditions d'accès (voir la politique basée sur les ressources) ou de définir les autorisations maximales pour tous les comptes d'une organisation dans AWS Organizations (voir la politique de contrôle des services).

persistance polyglotte

Choix indépendant de la technologie de stockage de données d'un microservice en fonction des modèles d'accès aux données et d'autres exigences. Si vos microservices utilisent la même technologie de stockage de données, ils peuvent rencontrer des difficultés d'implémentation ou présenter des performances médiocres. Les microservices sont plus faciles à mettre en œuvre, atteignent de meilleures performances, ainsi qu'une meilleure capacité de mise à l'échelle s'ils utilisent l'entrepôt de données le mieux adapté à leurs besoins. Pour plus d'informations, veuillez consulter Enabling data persistence in microservices.

évaluation du portefeuille

Processus de découverte, d'analyse et de priorisation du portefeuille d'applications afin de planifier la migration. Pour plus d'informations, veuillez consulter Evaluating migration readiness.

predicate

Une condition de requête qui renvoie `true` ou`false`, généralement située dans une `WHERE` clause.

prédicat pushdown

Technique d'optimisation des requêtes de base de données qui filtre les données de la requête avant le transfert. Cela réduit la quantité de données qui doivent être extraites et traitées à partir de la base de données relationnelle et améliore les performances des requêtes.

contrôle préventif

Contrôle de sécurité conçu pour empêcher qu'un événement ne se produise. Ces contrôles constituent une première ligne de défense pour empêcher tout accès non autorisé ou toute modification indésirable de votre réseau. Pour plus d'informations, veuillez consulter Preventative controls dans Implementing security controls on AWS.

principal

Entité AWS capable d'effectuer des actions et d'accéder aux ressources. Cette entité est généralement un utilisateur root pour un Compte AWS rôle IAM ou un utilisateur. Pour plus d'informations, veuillez consulter la rubrique Principal dans Termes et concepts relatifs aux rôles, dans la documentation IAM.

Confidentialité dès la conception

Une approche de l'ingénierie des systèmes qui prend en compte la confidentialité tout au long du processus d'ingénierie.

zones hébergées privées

Conteneur qui contient des informations concernant la façon dont vous souhaitez qu'Amazon Route 53 réponde aux requêtes DNS pour un domaine et ses sous-domaines dans un ou plusieurs VPC. Pour plus d'informations, veuillez consulter Working with private hosted zones dans la documentation Route 53.

contrôle proactif

Contrôle de sécurité conçu pour empêcher le déploiement de ressources non conformes. Ces contrôles analysent les ressources avant qu'elles ne soient provisionnées. Si la ressource n'est pas conforme au contrôle, elle n'est pas provisionnée. Pour plus d'informations, consultez le guide de référence sur les contrôles dans la AWS Control Tower documentation et consultez la section Contrôles proactifs dans Implémentation des contrôles de sécurité sur AWS.

gestion du cycle de vie des produits (PLM)

Gestion des données et des processus d'un produit tout au long de son cycle de vie, depuis la conception, le développement et le lancement, en passant par la croissance et la maturité, jusqu'au déclin et au retrait.

environnement de production

Voir environnement.

contrôleur logique programmable (PLC)

Dans le secteur manufacturier, un ordinateur hautement fiable et adaptable qui surveille les machines et automatise les processus de fabrication.

pseudonymisation

Processus de remplacement des identifiants personnels dans un ensemble de données par des valeurs fictives. La pseudonymisation peut contribuer à protéger la vie privée. Les données pseudonymisées sont toujours considérées comme des données personnelles.

publier/souscrire (pub/sub)

Modèle qui permet des communications asynchrones entre les microservices afin d'améliorer l'évolutivité et la réactivité. Par exemple, dans un MES basé sur des microservices, un microservice peut publier des messages d'événements sur un canal auquel d'autres microservices peuvent s'abonner. Le système peut ajouter de nouveaux microservices sans modifier le service de publication.

Q

plan de requête

Série d'étapes, telles que des instructions, utilisées pour accéder aux données d'un système de base de données relationnelle SQL.

régression du plan de requêtes

Le cas où un optimiseur de service de base de données choisit un plan moins optimal qu'avant une modification donnée de l'environnement de base de données. Cela peut être dû à des changements en termes de statistiques, de contraintes, de paramètres d'environnement, de liaisons de paramètres de requêtes et de mises à jour du moteur de base de données.

R

Matrice RACI

Voir responsable, responsable, consulté, informé (RACI).

rançongiciel

Logiciel malveillant conçu pour bloquer l'accès à un système informatique ou à des données jusqu'à ce qu'un paiement soit effectué.

Matrice RASCI

Voir responsable, responsable, consulté, informé (RACI).

RCAC

Voir contrôle d'accès aux lignes et aux colonnes.

réplica en lecture

Copie d'une base de données utilisée en lecture seule. Vous pouvez acheminer les requêtes vers le réplica de lecture pour réduire la charge sur votre base de données principale.

réarchitecte

Voir 7 Rs.

objectif de point de récupération (RPO)

Durée maximale acceptable depuis le dernier point de récupération des données. Cela permet de déterminer ce qui est considéré comme une perte de données acceptable entre le dernier point de restauration et l'interruption du service.

objectif de temps de récupération (RTO)

Le délai maximum acceptable entre l'interruption du service et le rétablissement du service.

refactoriser

Voir 7 Rs.

Région

Un ensemble de AWS ressources dans une zone géographique. Chacune Région AWS est isolée et indépendante des autres pour garantir tolérance aux pannes, stabilité et résilience. Pour plus d'informations, voir Spécifier ce que Régions AWS votre compte peut utiliser.

régression

Technique de ML qui prédit une valeur numérique. Par exemple, pour résoudre le problème « Quel sera le prix de vente de cette maison ? », un modèle de ML pourrait utiliser un modèle de régression linéaire pour prédire le prix de vente d'une maison sur la base de faits connus à son sujet (par exemple, la superficie en mètres carrés).

réhéberger

Voir 7 Rs.

version

Dans un processus de déploiement, action visant à promouvoir les modifications apportées à un environnement de production.

déplacer

Voir 7 Rs.

replateforme

Voir 7 Rs.

rachat

Voir 7 Rs.

résilience

La capacité d'une application à résister aux perturbations ou à s'en remettre. La haute disponibilité et la reprise après sinistre sont des considérations courantes lors de la planification de la résilience dans le AWS Cloud. Pour plus d'informations, consultez AWS Cloud Résilience.

politique basée sur les ressources

Politique attachée à une ressource, comme un compartiment Amazon S3, un point de terminaison ou une clé de chiffrement. Ce type de politique précise les principaux auxquels l'accès est autorisé, les actions prises en charge et toutes les autres conditions qui doivent être remplies.

matrice responsable, redevable, consulté et informé (RACI)

Une matrice qui définit les rôles et les responsabilités de toutes les parties impliquées dans les activités de migration et les opérations cloud. Le nom de la matrice est dérivé des types de responsabilité définis dans la matrice : responsable (R), responsable (A), consulté (C) et informé (I). Le type de support (S) est facultatif. Si vous incluez le support, la matrice est appelée matrice RASCI, et si vous l'excluez, elle est appelée matrice RACI.

contrôle réactif

Contrôle de sécurité conçu pour permettre de remédier aux événements indésirables ou aux écarts par rapport à votre référence de sécurité. Pour plus d'informations, veuillez consulter la rubrique Responsive controls dans Implementing security controls on AWS.

retain

Voir 7 Rs.

se retirer

Voir 7 Rs.

rotation

Processus de mise à jour périodique d'un secret pour empêcher un attaquant d'accéder aux informations d'identification.

contrôle d'accès aux lignes et aux colonnes (RCAC)

Utilisation d'expressions SQL simples et flexibles dotées de règles d'accès définies. Le RCAC comprend des autorisations de ligne et des masques de colonnes.

RPO

Voir l'objectif du point de récupération.

RTO

Voir l'objectif en matière de temps de rétablissement.

runbook

Ensemble de procédures manuelles ou automatisées nécessaires à l'exécution d'une tâche spécifique. Elles visent généralement à rationaliser les opérations ou les procédures répétitives présentant des taux d'erreur élevés.

S

SAML 2.0

Un standard ouvert utilisé par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter AWS Management Console ou appeler les opérations d' AWS API sans que vous ayez à créer

un utilisateur dans IAM pour tous les membres de votre organisation. Pour plus d'informations sur la fédération SAML 2.0, veuillez consulter À propos de la fédération SAML 2.0 dans la documentation IAM.

SCADA

Voir Contrôle de supervision et acquisition de données.

SCP

Voir la politique de contrôle des services.

secret

Dans AWS Secrets Manager des informations confidentielles ou restreintes, telles qu'un mot de passe ou des informations d'identification utilisateur, que vous stockez sous forme cryptée. Il comprend la valeur secrète et ses métadonnées. La valeur secrète peut être binaire, une chaîne unique ou plusieurs chaînes. Pour plus d'informations, voir Que contient le secret d'un Secrets Manager ? dans la documentation de Secrets Manager.

contrôle de sécurité

Barrière de protection technique ou administrative qui empêche, détecte ou réduit la capacité d'un assaillant d'exploiter une vulnérabilité de sécurité. Il existe quatre principaux types de contrôles de sécurité : préventifs, détectifs, réactifs et proactifs.

renforcement de la sécurité

Processus qui consiste à réduire la surface d'attaque pour la rendre plus résistante aux attaques. Cela peut inclure des actions telles que la suppression de ressources qui ne sont plus requises, la mise en œuvre des bonnes pratiques de sécurité consistant à accorder le moindre privilège ou la désactivation de fonctionnalités inutiles dans les fichiers de configuration.

système de gestion des informations et des événements de sécurité (SIEM)

Outils et services qui associent les systèmes de gestion des informations de sécurité (SIM) et de gestion des événements de sécurité (SEM). Un système SIEM collecte, surveille et analyse les données provenant de serveurs, de réseaux, d'appareils et d'autres sources afin de détecter les menaces et les failles de sécurité, mais aussi de générer des alertes.

automatisation des réponses de sécurité

Action prédéfinie et programmée conçue pour répondre automatiquement à un événement de sécurité ou y remédier. Ces automatisations servent de contrôles de sécurité détectifs ou réactifs

qui vous aident à mettre en œuvre les meilleures pratiques AWS de sécurité. Parmi les actions de réponse automatique, citons la modification d'un groupe de sécurité VPC, l'application de correctifs à une instance Amazon EC2 ou la rotation des informations d'identification.

chiffrement côté serveur

Chiffrement des données à destination, par celui AWS service qui les reçoit.

Politique de contrôle des services (SCP)

Politique qui propose un contrôle centralisé des autorisations pour tous les comptes d'une organisation dans AWS Organizations. Les SCP définissent des barrières de protection ou des limites aux actions qu'un administrateur peut déléguer à des utilisateurs ou à des rôles. Vous pouvez utiliser les SCP comme listes d'autorisation ou de refus, pour indiquer les services ou les actions autorisés ou interdits. Pour plus d'informations, consultez la section Politiques de contrôle des services dans la AWS Organizations documentation.

point de terminaison du service

URL du point d'entrée pour un AWS service. Pour vous connecter par programmation au service cible, vous pouvez utiliser un point de terminaison. Pour plus d'informations, veuillez consulter la rubrique AWS service endpoints dans Références générales AWS.

contrat de niveau de service (SLA)

Accord qui précise ce qu'une équipe informatique promet de fournir à ses clients, comme le temps de disponibilité et les performances des services.

indicateur de niveau de service (SLI)

Mesure d'un aspect des performances d'un service, tel que son taux d'erreur, sa disponibilité ou son débit.

objectif de niveau de service (SLO)

Mesure cible qui représente l'état d'un service, tel que mesuré par un indicateur de niveau de service.

modèle de responsabilité partagée

Un modèle décrivant la responsabilité que vous partagez en matière AWS de sécurité et de conformité dans le cloud. AWS est responsable de la sécurité du cloud, alors que vous êtes responsable de la sécurité dans le cloud. Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée.

SIEM

Consultez les informations de sécurité et le système de gestion des événements.

point de défaillance unique (SPOF)

Défaillance d'un seul composant critique d'une application susceptible de perturber le système.

SLA

Voir le contrat de niveau de service.

SLI

Voir l'indicateur de niveau de service.

SLO

Voir l'objectif de niveau de service.

split-and-seed modèle

Modèle permettant de mettre à l'échelle et d'accélérer les projets de modernisation. Au fur et à mesure que les nouvelles fonctionnalités et les nouvelles versions de produits sont définies, l'équipe principale se divise pour créer des équipes de produit. Cela permet de mettre à l'échelle les capacités et les services de votre organisation, d'améliorer la productivité des développeurs et de favoriser une innovation rapide. Pour plus d'informations, consultez la section Approche progressive de la modernisation des applications dans le. AWS Cloud

SPOF

Voir point de défaillance unique.

schéma en étoile

Structure organisationnelle de base de données qui utilise une grande table de faits pour stocker les données transactionnelles ou mesurées et utilise une ou plusieurs tables dimensionnelles plus petites pour stocker les attributs des données. Cette structure est conçue pour être utilisée dans un entrepôt de données ou à des fins de business intelligence.

modèle de figuier étrangleur

Approche de modernisation des systèmes monolithiques en réécrivant et en remplaçant progressivement les fonctionnalités du système jusqu'à ce que le système hérité puisse être mis hors service. Ce modèle utilise l'analogie d'un figuier de vigne qui se développe dans un arbre existant et qui finit par supplanter son hôte. Le schéma a été présenté par Martin Fowler comme

un moyen de gérer les risques lors de la réécriture de systèmes monolithiques. Pour obtenir un exemple d'application de ce modèle, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

sous-réseau

Plage d'adresses IP dans votre VPC. Un sous-réseau doit se trouver dans une seule zone de disponibilité.

contrôle de supervision et acquisition de données (SCADA)

Dans le secteur manufacturier, un système qui utilise du matériel et des logiciels pour surveiller les actifs physiques et les opérations de production.

chiffrement symétrique

Algorithme de chiffrement qui utilise la même clé pour chiffrer et déchiffrer les données.

tests synthétiques

Tester un système de manière à simuler les interactions des utilisateurs afin de détecter les problèmes potentiels ou de surveiller les performances. Vous pouvez utiliser Amazon CloudWatch Synthetics pour créer ces tests.

T

balises

Des paires clé-valeur qui agissent comme des métadonnées pour organiser vos AWS ressources. Les balises peuvent vous aider à gérer, identifier, organiser, rechercher et filtrer des ressources. Pour plus d'informations, veuillez consulter la rubrique Balisage de vos AWS ressources.

variable cible

La valeur que vous essayez de prédire dans le cadre du ML supervisé. Elle est également qualifiée de variable de résultat. Par exemple, dans un environnement de fabrication, la variable cible peut être un défaut du produit.

liste de tâches

Outil utilisé pour suivre les progrès dans un runbook. Liste de tâches qui contient une vue d'ensemble du runbook et une liste des tâches générales à effectuer. Pour chaque tâche générale, elle inclut le temps estimé nécessaire, le propriétaire et l'avancement.

environnement de test

Voir environnement.

entraînement

Pour fournir des données à partir desquelles votre modèle de ML peut apprendre. Les données d'entraînement doivent contenir la bonne réponse. L'algorithme d'apprentissage identifie des modèles dans les données d'entraînement, qui mettent en correspondance les attributs des données d'entrée avec la cible (la réponse que vous souhaitez prédire). Il fournit un modèle de ML qui capture ces modèles. Vous pouvez alors utiliser le modèle de ML pour obtenir des prédictions sur de nouvelles données pour lesquelles vous ne connaissez pas la cible.

passerelle de transit

Hub de transit de réseau que vous pouvez utiliser pour relier vos VPC et vos réseaux sur site. Pour plus d'informations, voir Qu'est-ce qu'une passerelle de transit dans la AWS Transit Gateway documentation.

flux de travail basé sur jonction

Approche selon laquelle les développeurs génèrent et testent des fonctionnalités localement dans une branche de fonctionnalités, puis fusionnent ces modifications dans la branche principale. La branche principale est ensuite intégrée aux environnements de développement, de préproduction et de production, de manière séquentielle.

accès sécurisé

Accorder des autorisations à un service que vous spécifiez pour effectuer des tâches au sein de votre organisation AWS Organizations et dans ses comptes en votre nom. Le service de confiance crée un rôle lié au service dans chaque compte, lorsque ce rôle est nécessaire, pour effectuer des tâches de gestion à votre place. Pour plus d'informations, consultez la section Utilisation AWS Organizations avec d'autres AWS services dans la AWS Organizations documentation.

réglage

Pour modifier certains aspects de votre processus d'entraînement afin d'améliorer la précision du modèle de ML. Par exemple, vous pouvez entraîner le modèle de ML en générant un ensemble d'étiquetage, en ajoutant des étiquettes, puis en répétant ces étapes plusieurs fois avec différents paramètres pour optimiser le modèle.

équipe de deux pizzas

Une petite DevOps équipe que vous pouvez nourrir avec deux pizzas. Une équipe de deux pizzas garantit les meilleures opportunités de collaboration possible dans le développement de logiciels.

U

incertitude

Un concept qui fait référence à des informations imprécises, incomplètes ou inconnues susceptibles de compromettre la fiabilité des modèles de ML prédictifs. Il existe deux types d'incertitude : l'incertitude épistémique est causée par des données limitées et incomplètes, alors que l'incertitude aléatoire est causée par le bruit et le caractère aléatoire inhérents aux données. Pour plus d'informations, veuillez consulter le guide Quantifying uncertainty in deep learning systems.

tâches indifférenciées

Également connu sous le nom de « levage de charges lourdes », ce travail est nécessaire pour créer et exploiter une application, mais qui n'apporte pas de valeur directe à l'utilisateur final ni d'avantage concurrentiel. Les exemples de tâches indifférenciées incluent l'approvisionnement, la maintenance et la planification des capacités.

environnements supérieurs

Voir environnement.

V

mise à vide

Opération de maintenance de base de données qui implique un nettoyage après des mises à jour incrémentielles afin de récupérer de l'espace de stockage et d'améliorer les performances.

contrôle de version

Processus et outils permettant de suivre les modifications, telles que les modifications apportées au code source dans un référentiel.

Appairage de VPC

Connexion entre deux VPC qui vous permet d'acheminer le trafic à l'aide d'adresses IP privées. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que l'appairage de VPC ? dans la documentation Amazon VPC.

vulnérabilités

Défaut logiciel ou matériel qui compromet la sécurité du système.

W

cache actif

Cache tampon qui contient les données actuelles et pertinentes fréquemment consultées. L'instance de base de données peut lire à partir du cache tampon, ce qui est plus rapide que la lecture à partir de la mémoire principale ou du disque.

données chaudes

Données rarement consultées. Lorsque vous interrogez ce type de données, des requêtes modérément lentes sont généralement acceptables.

fonction de fenêtre

Fonction SQL qui effectue un calcul sur un groupe de lignes liées d'une manière ou d'une autre à l'enregistrement en cours. Les fonctions de fenêtre sont utiles pour traiter des tâches, telles que le calcul d'une moyenne mobile ou l'accès à la valeur des lignes en fonction de la position relative de la ligne en cours.

charge de travail

Ensemble de ressources et de code qui fournit une valeur métier, par exemple une application destinée au client ou un processus de backend.

flux de travail

Groupes fonctionnels d'un projet de migration chargés d'un ensemble de tâches spécifique. Chaque flux de travail est indépendant, mais prend en charge les autres flux de travail du projet. Par exemple, le flux de travail du portefeuille est chargé de prioriser les applications, de planifier les vagues et de collecter les métadonnées de migration. Le flux de travail du portefeuille fournit ces actifs au flux de travail de migration, qui migre ensuite les serveurs et les applications.

VER

Voir écrire une fois, lire plusieurs.

WQF

Consultez le cadre de qualification des charges de travail AWS.

écrire une fois, lire plusieurs (WORM)

Modèle de stockage qui écrit les données une seule fois et empêche leur suppression ou leur modification. Les utilisateurs autorisés peuvent lire les données autant de fois que nécessaire, mais ils ne peuvent pas les modifier. Cette infrastructure de stockage de données est considérée comme immuable.

Z

exploit Zero-Day

Une attaque, généralement un logiciel malveillant, qui tire parti d'une vulnérabilité de type « jour zéro ».

vulnérabilité de type « jour zéro »

Une faille ou une vulnérabilité non atténuée dans un système de production. Les acteurs malveillants peuvent utiliser ce type de vulnérabilité pour attaquer le système. Les développeurs prennent souvent conscience de la vulnérabilité à la suite de l'attaque.

application zombie

Application dont l'utilisation moyenne du processeur et de la mémoire est inférieure à 5 %. Dans un projet de migration, il est courant de retirer ces applications.

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.