Mejores prácticas para usar la información para crear AWS CDK proyectos TypeScript de IaC

AWS Guía prescriptiva

AWS Guía prescriptiva: Mejores prácticas para usar la información para crear AWS CDK proyectos TypeScript de IaC

Table of Contents

Prácticas recomendadas para usar la CDK de AWS TypeScript para crear proyectos de IaC

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

Febrero de 2024 (historial de documentos)

En esta guía se proporcionan recomendaciones y prácticas recomendadas para utilizarla AWS Cloud Development Kit (AWS CDK)en TypeScript la creación e implementación de proyectos de infraestructura como código (IaC) a gran escala. AWS CDK Se trata de un marco para definir la infraestructura de nube en el código y aprovisionarla mediante ella. AWS CloudFormation Si no tiene una estructura de proyecto bien definida, crear y administrar una AWS CDK base de código para proyectos a gran escala puede ser un desafío. A fin de hacer frente a estos desafíos, algunas organizaciones utilizan antipatrones para proyectos a gran escala, pero estos patrones pueden retrasar el proyecto y crear otros problemas que afecten de forma negativa a la organización. Por ejemplo, los antipatrones pueden complicar y retrasar la incorporación de los desarrolladores, la corrección de errores y la adopción de características nuevas.

En esta guía, se ofrece una alternativa al uso de antipatrones y se muestra cómo organizar el código para garantizar la escalabilidad, las pruebas y la alineación con las prácticas recomendadas de seguridad. Puede utilizar esta guía para mejorar la calidad del código de sus proyectos de IaC y maximizar la agilidad de su empresa. Esta guía está dirigida a arquitectos, directores técnicos, ingenieros de infraestructura y cualquier otra persona que busque crear un proyecto bien diseñado para proyectos a gran escala. AWS CDK

Objetivos

Esta guía puede ayudarlo a lograr los siguientes resultados empresariales específicos:

- Costes reducidos: puede utilizarlos AWS CDK para diseñar sus propios componentes reutilizables que cumplan con los requisitos de seguridad, cumplimiento y gobierno de su organización. También puede compartir con facilidad los componentes de su organización, de modo que pueda iniciar con rapidez proyectos nuevos que se ajusten a las prácticas recomendadas de forma predeterminada.
- Lanzamiento al mercado más rápido: aproveche las funciones conocidas AWS CDK para acelerar su proceso de desarrollo. Esto aumenta la capacidad de reutilización para la implementación y reduce los esfuerzos de desarrollo.
- Mayor productividad de los desarrolladores: los desarrolladores pueden usar lenguajes de programación conocidos para definir la infraestructura. Esto ayuda a los desarrolladores a expresar y mantener AWS los recursos. Esto puede llevar a un aumento de la eficiencia y la colaboración de los desarrolladores.

Prácticas recomendadas

En esta sección, se brinda un resumen de las siguientes prácticas recomendadas:

- Organizar el código para proyectos a gran escala
- Desarrollar patrones reutilizables
- Crear o ampliar constructos
- Siga las TypeScript mejores prácticas
- Buscar vulnerabilidades de seguridad y errores de formato
- Desarrollar y perfeccionar la documentación
- Adoptar un enfoque de desarrollo basado en pruebas
- Utilizar el control de versiones y lanzamiento para los constructos
- Aplicar la administración de versiones de bibliotecas

Organizar el código para proyectos a gran escala

Por qué es importante la organización del código

Es fundamental que los AWS CDK proyectos a gran escala tengan una estructura bien definida y de alta calidad. A medida que un proyecto se hace más grande y aumenta la cantidad de características y constructos compatibles, la navegación por el código se hace más difícil. Esta dificultad puede afectar a la productividad y retrasar la incorporación de los desarrolladores.

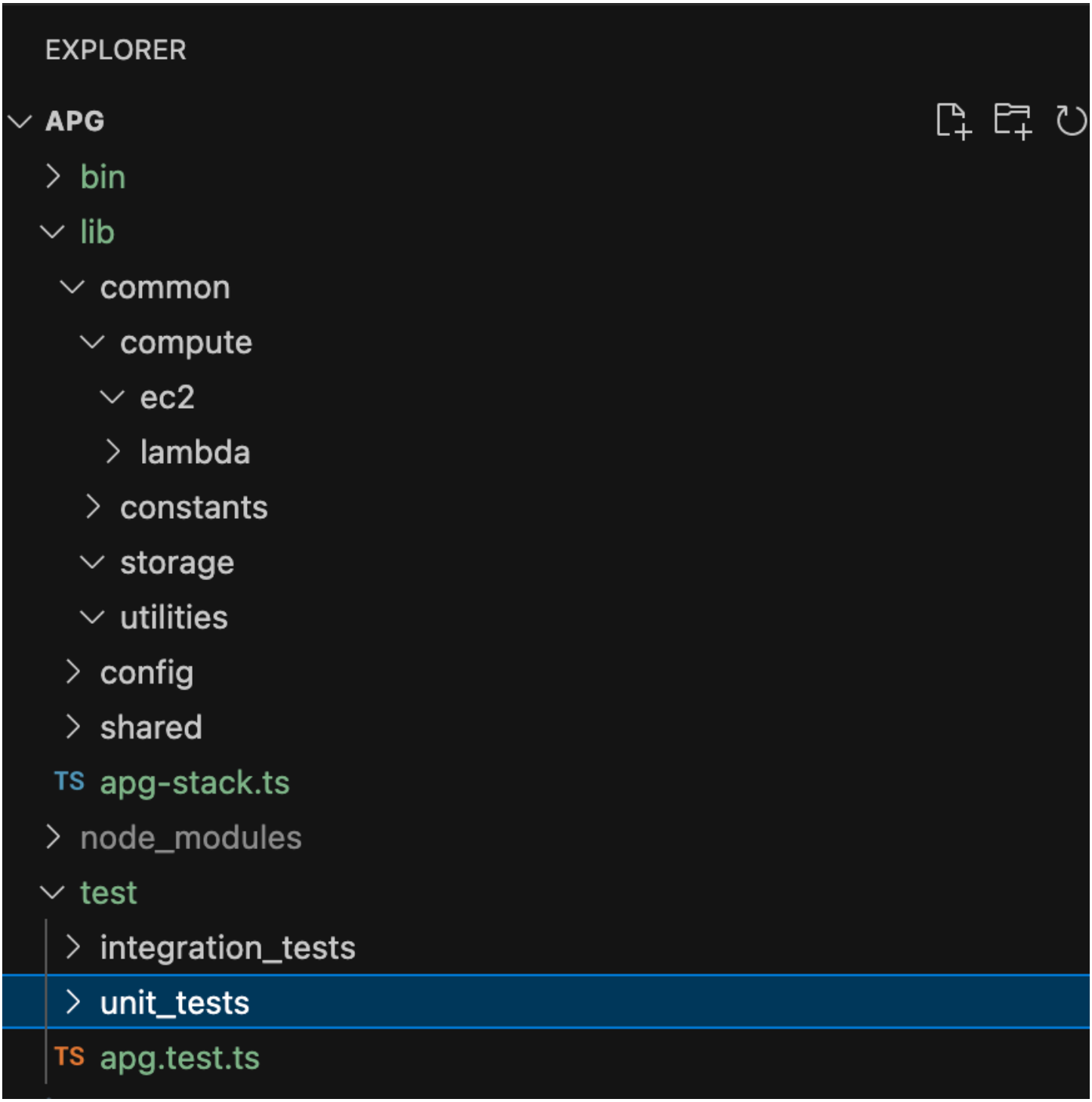
Cómo organizar el código para el escalado

Para lograr un alto nivel de flexibilidad y legibilidad del código, le recomendamos que lo divida en partes lógicas de acuerdo con la funcionalidad. Esta división refleja el hecho de que la mayoría de sus constructos se utilizan en diferentes dominios empresariales. Por ejemplo, tanto las aplicaciones de frontend como las de backend pueden requerir una AWS Lambda función y consumir el mismo código fuente. Las fábricas pueden crear objetos sin exponer la lógica de creación al cliente y utilizar una interfaz común para hacer referencia a los objetos que se crearon recientemente. Puede utilizar una fábrica como patrón eficaz para crear un comportamiento coherente en la base del código. Además, una fábrica puede servir como fuente única de información fiable para ayudarlo a evitar la repetición de códigos y facilitar la solución de problemas.

Para comprender mejor cómo funcionan las fábricas, consideremos el ejemplo de un fabricante de automóviles. Un fabricante de automóviles no necesita tener los conocimientos y la infraestructura necesarios para fabricar neumáticos. El fabricante de automóviles subcontrata a un fabricante de neumáticos especializado en esa materia y, luego, simplemente solicita los neumáticos a ese fabricante según sea necesario. El mismo principio se aplica al código. Por ejemplo, puede crear una fábrica de Lambda que sea capaz de crear funciones de Lambda de alta calidad y, a continuación, llamar a la fábrica de Lambda en el código siempre que necesite crear una función de Lambda. De igual modo, puede utilizar este mismo proceso de subcontratación para desvincular la aplicación y crear componentes modulares.

Organización del código de ejemplo

El siguiente proyecto de ejemplo, tal y como se TypeScript muestra en la imagen siguiente, incluye una carpeta común en la que puede guardar todas sus construcciones o funcionalidades comunes.



Por ejemplo, la carpeta computación (que se encuentra en la carpeta común) contiene toda la lógica de los diferentes constructos de computación. Los desarrolladores nuevos pueden agregar con facilidad constructos de computación nuevos sin afectar al resto de los recursos. Todas las demás construcciones no necesitarán crear nuevos recursos internamente. En su lugar, estos constructos

simplemente llaman a la fábrica de constructos comunes. Puede organizar otros constructos, como el almacenamiento, de la misma manera.

Las configuraciones contienen datos basados en el entorno que debe desacoplar de la carpeta común donde guarda la lógica. Le recomendamos que coloque sus datos de configuración comunes en una carpeta compartida. También recomendamos que utilice la carpeta de utilidades para ejecutar todas las funciones de ayuda y limpiar los scripts.

Desarrollar patrones reutilizables

Los patrones de diseño de software son soluciones reutilizables para problemas comunes en el desarrollo de software. Actúan como una guía o un paradigma para ayudar a los ingenieros de software a crear productos que sigan las prácticas recomendadas. Esta sección proporciona una descripción general de dos patrones reutilizables que puedes usar en tu AWS CDK base de código: el patrón Abstract Factory y el patrón Chain of Responsibility. Puede utilizar cada patrón como esquema y personalizarlo para el problema de diseño concreto de su código. Para obtener más información sobre los patrones de diseño, consulte Patrones de diseño en la documentación de Refactoring.Guru.

Fábrica abstracta

El patrón Fábrica abstracta brinda interfaces para crear familias de objetos relacionados o dependientes sin especificar sus clases concretas. Este patrón se aplica a los siguientes casos de uso:

- Cuando el cliente es independiente de la forma en que se crean y componen los objetos del sistema
- Cuando el sistema consta de varias familias de objetos y estas familias se han diseñado para utilizarse juntas
- Cuando debe tener un valor de tiempo de ejecución para crear una dependencia determinada

Para obtener más información sobre el patrón Abstract Factory, consulta Abstract Factory TypeScript en la documentación de Refactoring.Guru.

En el siguiente ejemplo de código, se muestra cómo utilizar el patrón Fábrica abstracta para crear una fábrica de almacenamiento de 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();
```

Cadena de responsabilidades

La Cadena de responsabilidades es un patrón de diseño de comportamientos que permite transmitir una solicitud a lo largo de la cadena de posibles controladores hasta que uno de ellos gestione la solicitud. El patrón Cadena de responsabilidades se aplica a los siguientes casos de uso:

- Cuando varios objetos, determinados en el tiempo de ejecución, son candidatos para gestionar una solicitud
- Cuando no desee especificar controladores de forma explícita en su código
- Cuando quiere enviar una solicitud a uno de varios objetos sin especificar el receptor de forma explícita

Para obtener más información sobre el patrón de cadena de responsabilidad, consulte Cadena de responsabilidad TypeScript en la documentación de Refactoring.Guru.

En el siguiente código, se muestra un ejemplo de cómo se utiliza el patrón Cadena de responsabilidades a fin de crear una serie de acciones necesarias para completar la tarea.

```
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);
    }
}
```

Crear o ampliar constructos

Qué es un constructo

Una construcción es el componente básico de una aplicación. AWS CDK Una construcción puede representar un único AWS recurso, como un bucket de Amazon Simple Storage Service (Amazon S3), o puede ser una abstracción de nivel superior que consta de varios recursos relacionados. AWS Los componentes de un constructo pueden incluir una cola de trabajadores con su capacidad de computación asociada o un trabajo programado con recursos de monitoreo y un panel. AWS CDK Incluye un conjunto de construcciones denominado Construct Library. AWS La biblioteca contiene

componentes fijos para cada. Servicio de AWS Puedes usar el Construct Hub para descubrir otras construcciones de terceros y de AWS la comunidad de código abierto AWS CDK .

Cuáles son los diferentes tipos de constructos

Existen tres tipos diferentes de construcciones para: AWS CDK

- Construcciones de L1: las construcciones de capa 1, o L1, son exactamente los recursos definidos por CloudFormation, ni más ni menos. Debe brindar los recursos necesarios para la configuración. Estas construcciones de L1 son muy básicas y se deben configurar manualmente. Las construcciones L1 tienen un `Cfn` prefijo y se corresponden directamente con las especificaciones. CloudFormation Servicios de AWS Los nuevos se admiten tan pronto AWS CDK como se admita este CloudFormation tipo de servicios. CfnBuketes un buen ejemplo de una construcción L1. Esta clase representa un bucket de S3 en el que se deben configurar todas las propiedades de forma explícita. Te recomendamos que solo utilices una construcción L1 si no puedes encontrar la construcción L2 o L3 para ella.
- Constructos de C2: los constructos de capa 2, o C2, tienen un código repetitivo y una lógica de enlace comunes. Estos constructos vienen con cómodos valores predeterminados y reducen la cantidad de conocimientos que necesita saber sobre ellos. Las construcciones de nivel 2 utilizan API basadas en la intención para crear sus recursos y, por lo general, encapsulan sus módulos de nivel 1 correspondientes. Un buen ejemplo de un constructo C2 es el Bucket. Esta clase crea un depósito de S3 con propiedades y métodos predeterminados, como bucket.add LifeCycle Rule (), que añade una regla de ciclo de vida al depósito.
- Constructo de C3: un constructo de capa 3, o C3, se denomina patrón. Las estructuras L3 están diseñadas para ayudarle a completar tareas comunes AWS, que suelen implicar varios tipos de recursos. Son incluso más específicos y obstinados que los constructos de C2 y sirven para un caso de uso específico. Por ejemplo, los aws-ecs-patterns. ApplicationLoadBalancedFargateLa construcción de servicio representa una arquitectura que incluye un clúster de AWS Fargate contenedores que utiliza un Application Load Balancer. Otro ejemplo es el aws-apigateway. LambdaRestConstrucción de API. Este constructo representa una API de Amazon API Gateway que se encuentra respaldada por una función de Lambda.

A medida que los niveles de constructo aumentan, se hacen más suposiciones sobre cómo se utilizarán los constructos. Esto le permite proporcionar interfaces con valores predeterminados más efectivos para casos de uso muy específicos.

Cómo crear su propio constructo

Para definir su propio constructo, debe seguir un enfoque específico. Esto se debe a que todos los constructos amplían la clase `Construct`. La clase `Construct` es el componente básico del árbol de constructo. Los constructos se implementan en clases que amplían la clase base `Construct`. Todos los constructos toman tres parámetros cuando se inicializan:

- Alcance: el padre o propietario de un constructo, ya sea una pila u otro constructo, que determina su lugar en el árbol del constructo. Por lo general, debe pasar `this` (o `self` en Python), que representa el objeto actual, para el ámbito.
- ID: un identificador que debe ser único en este alcance. El identificador sirve como espacio de nombres para todo lo que se define en la construcción actual y se utiliza para asignar identidades únicas, como nombres de recursos e identificadores CloudFormation lógicos.
- Accesorios: conjunto de propiedades que definen la configuración inicial de la construcción.

En el siguiente ejemplo, se muestra cómo definir un constructo.

```
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
  }
}
```

Crear o ampliar un constructo de C2

Una construcción L2 representa un «componente de nube» y encapsula todo lo CloudFormation necesario para crear el componente. Una construcción L2 puede contener uno o más AWS recursos, y puedes personalizar la construcción tú mismo. La ventaja de crear o ampliar una construcción de L2 es que se pueden reutilizar los componentes de las CloudFormation pilas sin tener que redefinir el código. Simplemente puede importar el constructo como una clase.

Cuando existe una relación «es una» con una construcción existente, puede ampliarla para añadir características predeterminadas adicionales. Se recomienda reutilizar las propiedades de la construcción L2 existente. Puede sobrescribir las propiedades al modificarlas directamente en el constructo.

En el siguiente ejemplo, se muestra cómo alinearse con las prácticas recomendadas y ampliar un constructo de C2 existente denominado `s3.Bucket`. La extensión establece las propiedades predeterminadas, como `versioned`, `publicReadAccess`, `blockPublicAccess`, para garantizar que todos los objetos (en este ejemplo, los buckets de S3) creados a partir de este constructo nuevo tengan siempre establecidos estos valores predeterminados.

```
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
    });
  }
}
```

Crear un constructo de C3

La composición es la mejor opción cuando existe una relación «tiene una» con una composición de construcción existente. La composición significa que crea su propio constructo sobre otros constructos existentes. Puede crear su propio patrón para incorporar todos los recursos y sus valores predeterminados dentro de un único constructo de C3 de nivel superior que se puede compartir. El beneficio de crear sus propios constructos de C3 (patrones) es que puede reutilizar los componentes en pilas sin tener que redefinir el código. Simplemente puede importar el constructo como una clase. Estos patrones se han diseñado para ayudar a los consumidores a ofrecer diferentes recursos basados en patrones comunes con un conocimiento limitado y conciso.

En el siguiente ejemplo de código se crea una AWS CDK construcción llamada`ExampleConstruct`. Puede utilizar este constructo como plantilla para definir los componentes en la nube.

```
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
  }
}
```

En el siguiente ejemplo, se muestra cómo importar la construcción recién creada en AWS CDK la aplicación o pila.

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

En el siguiente ejemplo, se muestra cómo puede instanciar una instancia del constructo que ha ampliado desde la clase base.

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

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

Para obtener más información, consulte AWS CDK Workshop en la documentación del AWS CDK Workshop.

Escotilla de escape

Puede utilizar una trampilla de escape AWS CDK para subir un nivel de abstracción y acceder al nivel inferior de los constructos. Las trampillas de escape se utilizan para extender el componente fijo a elementos que no están expuestos en la versión actual, AWS pero que están disponibles en. CloudFormation

Le recomendamos utilizar una escotilla de escape en los siguientes escenarios:

- Hay una Servicio de AWS función disponible a través de CloudFormation ella, pero no hay `Construct` componentes fijos para ella.

- Una Servicio de AWS función está disponible a través del servicio CloudFormation y hay `Construct` componentes fijos para él, pero estos componentes aún no exponen la función. Como `Construct` las construcciones se desarrollan «a mano», a veces pueden estar a la zaga de las construcciones de CloudFormation recursos.

En el siguiente código de ejemplo, se muestra un caso de uso común para el uso de una escotilla de escape. En este ejemplo, la funcionalidad que aún no se ha implementado en el constructo de nivel superior se establece a fin de agregar `httpPutResponseHopLimit` para el escalado automático de `LaunchConfiguration`.

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

En el ejemplo de código anterior, se muestra el siguiente flujo de trabajo:

1. Define su `AutoScalingGroup` mediante el uso de un constructo de C2. La construcción L2 no permite actualizarla`httpPutResponseHopLimit`, por lo que debes usar una trampilla de escape.
2. Accede a la propiedad `node.defaultChild` en el constructo `AutoScalingGroup` de C2 y lo convierte en el recurso `CfnLaunchConfiguration`.
3. Ahora puede establecer la propiedad `launchConfig.metadataOptions` en la C1 `CfnLaunchConfiguration`.

Recursos personalizados

Puedes usar recursos personalizados para escribir una lógica de aprovisionamiento personalizada en las plantillas que CloudFormation se ejecute siempre que crees, actualices (si cambiaste el recurso personalizado) o elimines pilas. Por ejemplo, puedes usar un recurso personalizado si quieres incluir recursos que no estén disponibles en. AWS CDK De esta forma, puede seguir administrando todos los recursos relacionados en una sola pila.

La creación de un recurso personalizado implica escribir una función de Lambda que responda a los eventos del ciclo de vida CREATE, UPDATE y DELETE de un recurso. Si tu recurso personalizado

debe realizar solo una llamada a la API, considera la posibilidad de utilizar la construcción AwsCustomResource. Esto permite realizar llamadas arbitrarias al SDK durante una CloudFormation implementación. De lo contrario, le sugerimos que escriba su propia función de Lambda para llevar a cabo el trabajo que debe realizar.

Para obtener más información sobre los recursos personalizados, consulta los recursos personalizados en la CloudFormation documentación. Para ver un ejemplo de cómo utilizar un recurso personalizado, consulte el repositorio de recursos personalizados en GitHub.

El siguiente ejemplo muestra cómo crear una clase de recurso personalizada para iniciar una función Lambda y enviar CloudFormation una señal de éxito o error.

```
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();
    }
}
```

En el siguiente ejemplo, se muestra la lógica principal del recurso personalizado.

```
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)
```

En el siguiente ejemplo, se muestra cómo la AWS CDK pila llama al recurso personalizado.

```
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();
```

Siga las TypeScript prácticas recomendadas

TypeScript es un lenguaje que amplía las capacidades de JavaScript. Es un lenguaje fuertemente tipificado y orientado a objetos. Se puede utilizar TypeScript para especificar los tipos de datos que se transmiten en el código y tiene la capacidad de informar de errores cuando los tipos no coinciden. En esta sección se proporciona una descripción general de las prácticas TypeScript recomendadas.

Describir los datos

Puede utilizarla TypeScript para describir la forma de los objetos y las funciones de su código. El uso del tipo `any` equivale a dejar de comprobar el tipo de una variable. Recomendamos que evite el uso de `any` en su código. A continuación se muestra un ejemplo.

```
type Result = "success" | "failure"
```

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

Utilizar enumeraciones

Puede utilizar enumeraciones para definir un conjunto de constantes con nombre y definir estándares que se puedan reutilizar en su base de código. Le recomendamos que exporte las enumeraciones una vez a nivel global y, a continuación, deje que otras clases importen y utilicen las enumeraciones. Suponga que desea crear un conjunto de posibles acciones para registrar los eventos en su base de código. TypeScript proporciona enumeraciones numéricas y basadas en cadenas. En el siguiente ejemplo, se utiliza una enumeración.

```
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)
```

Utilizar interfaces

Una interfaz es un contrato para la clase. Si crea un contrato, sus usuarios deberán cumplirlo. En el siguiente ejemplo, se utiliza una interfaz para estandarizar las `props` y garantizar que las personas que llaman ingresen el parámetro esperado al utilizar esta clase.

```
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
})
```

Algunas propiedades solo se pueden modificar cuando se crea un objeto por primera vez. Puede especificar esto al poner `readonly` antes del nombre de la propiedad, como se muestra en el siguiente ejemplo.

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

Ampliar interfaces

La amplificación de las interfaces reduce la duplicación, ya que no es necesario copiar las propiedades entre las interfaces. Además, el lector del código puede entender con facilidad las relaciones de la aplicación.

```
 interface BaseInterface{
```

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

Evitar interfaces vacías

Le recomendamos que evite las interfaces vacías debido a los posibles riesgos que conllevan. En el siguiente ejemplo, hay una interfaz vacía llamada. `BucketProps` Los objetos `myS3Bucket1` y `myS3Bucket2` son válidos, pero siguen estándares diferentes porque la interfaz no exige ningún contrato. El siguiente código compilará e imprimirá las propiedades, pero esto generará incoherencias en la aplicación.

```
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",
});
```

Utilizar fábricas

En un patrón Fábrica abstracta, una interfaz es responsable de crear una fábrica de objetos relacionados sin especificar sus clases de forma explícita. Por ejemplo, puede crear una fábrica de Lambda para crear funciones de Lambda. En lugar de crear una nueva función Lambda dentro de su

construcción, delega el proceso de creación en la fábrica. Para obtener más información sobre este patrón de diseño, consulte Abstract Factory TypeScript en la documentación de Refactoring.Guru.

Utilizar la desestructuración en las propiedades

La desestructuración, introducida en ECMAScript 6 (ES6), es una JavaScript función que permite extraer varios datos de una matriz u objeto y asignarlos a sus propias variables.

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

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

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

Definir las convenciones de nomenclatura estándar

La aplicación de una convención de nomenclatura mantiene la coherencia de la base de código y reduce la sobrecarga a la hora de pensar en cómo nombrar una variable. Le recomendamos lo siguiente:

- Utilice camelCase para los nombres de variables y funciones.
- Se utiliza PascalCase para los nombres de las clases y los nombres de las interfaces.
- Utilice camelCase para los miembros de la interfaz.
- Se utiliza PascalCase para nombres de tipos y nombres de enumeración.
- Nombre los archivos con camelCase (por ejemplo, `ebsVolumes.tsx` o `storage.tsb`)

No utilices la palabra clave var

La `let` sentencia se utiliza para declarar una variable local en TypeScript. Es similar a la `var` palabra clave, pero tiene algunas restricciones de alcance en comparación con la `var` palabra clave. Una variable declarada en un bloque con `let` solo se puede utilizar en ese bloque. La `var` palabra clave no puede tener un ámbito de bloques, lo que significa que se puede acceder a ella desde fuera de un bloque concreto (representado por`{}`), pero no fuera de la función en la que está definida.

Puede volver a declarar y actualizar las variables. `var` Se recomienda evitar el uso de la palabra clave. `var`

Considerar el uso de ESLint y Prettier

ESLint analiza de forma estática su código para encontrar problemas con rapidez. Puede utilizar ESLint para crear una serie de afirmaciones (denominadas reglas lint) que definen cómo debe verse o comportarse su código. ESLint también tiene sugerencias de reparación automática para ayudarlo a mejorar su código. Por último, puede utilizar ESLint para cargar reglas lint desde complementos compartidos.

Prettier es un formateador de código conocido que admite una variedad de lenguajes de programación diferentes. Puede utilizar Prettier para establecer el estilo de su código y evitar formatear el código de forma manual. Tras la instalación, puede actualizar su archivo `package.json` y ejecutar los comandos `npm run format` y `npm run lint`.

El siguiente ejemplo le muestra cómo habilitar ESLint y el formateador Prettier para su proyecto. 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)'"
}
```

Utilizar modificadores de acceso

El modificador privado TypeScript limita la visibilidad solo a la misma clase. Cuando agrega el modificador privado a una propiedad o método, puede acceder a esa propiedad o método dentro de la misma clase.

El modificador público permite acceder a las propiedades y los métodos de las clases desde todas las ubicaciones. Si no especificas ningún modificador de acceso para las propiedades y los métodos, usarán el modificador público de forma predeterminada.

El modificador protegido permite acceder a las propiedades y los métodos de una clase dentro de la misma clase y dentro de las subclases. Usa el modificador protegido cuando pienses crear subclases en tu aplicación. AWS CDK

Usa tipos de utilidades

Los tipos de utilidades TypeScript son funciones de tipos predefinidas que realizan transformaciones y operaciones en los tipos existentes. Esto le ayuda a crear tipos nuevos basados en los tipos existentes. Por ejemplo, puede cambiar o extraer propiedades, hacer que las propiedades sean opcionales u obligatorias, o crear versiones inmutables de los tipos. Al usar tipos de utilidades, puede definir tipos más precisos y detectar posibles errores en tiempo de compilación.

Parcial <Type>

`Partial`marca todos los miembros de un tipo de entrada `Type` como opcionales. Esta utilidad devuelve un tipo que representa todos los subconjuntos de un tipo determinado. A continuación se muestra un ejemplo de `Partial`.

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

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

Necesario <Type>

`Required`hace lo contrario de`Partial`. Hace que todos los miembros de un tipo de entrada `Type` no sean opcionales (en otras palabras, obligatorios). A continuación se muestra un ejemplo 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
};
```

Buscar vulnerabilidades de seguridad y errores de formato

La infraestructura como código (IaC) y la automatización se han vuelto fundamentales para las empresas. Debido a que la IaC es sumamente sólida, tiene la gran responsabilidad de administrar los riesgos de seguridad. Los riesgos de seguridad más comunes de la IaC pueden incluir los siguientes:

- Privilegios sobrepermisivos AWS Identity and Access Management (IAM)
- Grupos de seguridad abiertos
- Recursos no cifrados
- Registros de acceso no activados

Enfoques y herramientas de seguridad

Recomendamos que implemente los siguientes enfoques de seguridad:

- Detección de vulnerabilidades en el desarrollo: corregir las vulnerabilidades en la producción es caro y conlleva mucho tiempo debido a la complejidad del desarrollo y la distribución de las revisiones de software. Además, las vulnerabilidades en la producción implican el riesgo de ser explotadas. Le recomendamos que emplee el escaneo de código en sus recursos de IaC para poder detectar y corregir las vulnerabilidades antes de pasar a la producción.
- Cumplimiento y corrección automática: AWS Config ofrece reglas AWS administradas. Estas reglas le ayudan a garantizar el cumplimiento y le permiten intentar la corrección automática mediante AWS Systems Manager la automatización. También puede crear y asociar documentos de automatización personalizados mediante AWS Config reglas.

Herramientas de desarrollo comunes

Las herramientas que se describen en esta sección lo ayudarán a ampliar su funcionalidad integrada con sus propias reglas personalizadas. Le recomendamos que alinee las reglas personalizadas con

los estándares de su organización. Estas son algunas herramientas de desarrollo comunes que se deben tener en cuenta:

- Utilice cfn-nag para identificar los problemas de seguridad de la infraestructura, como las reglas de IAM permisivas o los literales de contraseña, en las plantillas. CloudFormation Para obtener más información, consulte el repositorio cfn-nag de Stelligent. GitHub
- Utilice cdk-nag, inspirado en cfn-nag, para validar que los constructos dentro de un alcance determinado cumplan con un conjunto definido de reglas. También puede utilizar cdk-nag para la supresión de reglas y los informes de conformidad. La herramienta cdk-nag valida los constructos ampliando aspectos del. AWS CDK Para obtener más información, consulte Administrar la seguridad de las aplicaciones y el cumplimiento de las normas y cdk-nag en el AWS Cloud Development Kit (AWS CDK) blog. AWS DevOps
- Utilice la herramienta de código abierto Checkov para realizar análisis estáticos en su entorno de IaC. Checkov ayuda a identificar los errores de configuración en la nube escaneando el código de su infraestructura en Kubernetes, Terraform o. CloudFormation Puede utilizar Checkov para obtener resultados en diferentes formatos, incluidos JSON, JUnit XML o CLI. Checkov puede gestionar las variables de forma eficaz mediante la creación de un gráfico que muestre la dependencia dinámica del código. Para obtener más información, consulte el repositorio de Checkov de Bridgecrew. GitHub
- Utilice TFLint para comprobar si hay errores y sintaxis obsoleta y para obtener ayuda en la aplicación de las prácticas recomendadas. Tenga en cuenta que es posible que TFLint no valide los problemas específicos del proveedor. Para obtener más información sobre TFlint, consulte el repositorio TFlint de Terraform Linters. GitHub
- Usa Amazon CodeWhisperer para realizar escaneos de seguridad. CodeWhisperer es una herramienta de productividad basada en inteligencia artificial que puede generar sugerencias de código en tiempo real. Puede ayudarlo a identificar problemas de seguridad, seguir las mejores prácticas y aumentar la productividad a la hora de implementar el código.

Desarrollar y perfeccionar la documentación

La documentación es fundamental para el éxito de su proyecto. La documentación no solo explica cómo funciona su código, sino que también ayuda a los desarrolladores a comprender mejor las características y la funcionalidad de sus aplicaciones. Desarrollar y perfeccionar la documentación de alta calidad puede fortalecer el proceso de desarrollo de software, mantener un software de alta calidad y ayudar a la transferencia de conocimientos entre los desarrolladores.

Existen dos categorías de documentación: la documentación incluida en el código y la documentación de respaldo sobre el código. La documentación incluida en el código se presenta en forma de comentarios. La documentación de respaldo sobre el código puede consistir en archivos README y documentos externos. No es raro que los desarrolladores piensen en la documentación como una sobrecarga, ya que el código en sí es fácil de entender. Esto podría ser cierto para proyectos pequeños, pero la documentación es fundamental para proyectos a gran escala en los que participan varios equipos.

Se recomienda que el autor del código escriba la documentación, ya que conoce bien sus funcionalidades. Los desarrolladores pueden tener dificultades con la sobrecarga adicional que supone mantener una documentación de respaldo independiente. Para superar este desafío, los desarrolladores pueden agregar los comentarios al código y extraerlos de forma automática a fin de que todas las versiones del código y la documentación se encuentren sincronizadas.

Existe una variedad de herramientas diferentes que ayudan a los desarrolladores a extraer los comentarios del código y generar la documentación correspondiente. Esta guía se centra en TypeDoc la herramienta preferida para AWS CDK las construcciones.

Por qué se requiere documentación de código para las construcciones AWS CDK

AWS CDK Varios equipos de una organización crean construcciones comunes y las comparten entre diferentes equipos para su consumo. Una buena documentación ayuda a los usuarios de la biblioteca de constructos a integrar con facilidad los constructos y a crear su infraestructura con el mínimo esfuerzo. Mantener todos los documentos sincronizados es una tarea ardua. Le recomendamos que mantenga el documento dentro del código, que se extraerá mediante la TypeDoc biblioteca.

TypeDoc Utilizándolo con la biblioteca AWS Construct

TypeDoc es un generador de documentos para TypeScript. Puede utilizarlos TypeDoc para leer los archivos TypeScript fuente, analizar los comentarios de esos archivos y, a continuación, generar un sitio estático que contenga la documentación del código.

El siguiente código muestra cómo realizar la integración TypeDoc con la biblioteca AWS Construct y, a continuación, añadir los siguientes paquetes a su `package.json` archivo`devDependencies`.

```
{
```

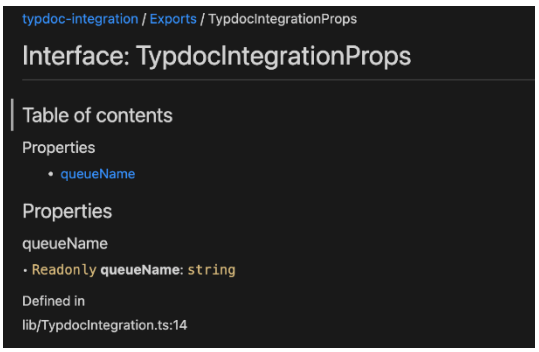
```
  "devDependencies": {

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

}
```

Para agregar `typedoc.json` en la carpeta de la biblioteca de CDK, utilice el siguiente código.

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

Para generar los archivos README, ejecute el `npx typedoc` comando en el directorio raíz del proyecto de la biblioteca de AWS CDK construcciones.

El siguiente documento de ejemplo ha sido generado por TypeDoc.



Para obtener más información sobre las opciones de TypeDoc integración, consulte los comentarios del documento en la TypeDoc documentación.

Adoptar un enfoque de desarrollo basado en pruebas

Le recomendamos que siga un enfoque de desarrollo basado en pruebas (TDD) con el. AWS CDK El TDD es un enfoque de desarrollo de software en el que se desarrollan casos de prueba para especificar y validar el código. En pocas palabras, primero se crean casos de prueba para cada funcionalidad y, si la prueba falla, se escribe el código nuevo a fin de pasar la prueba y hacer que el código sea simple y libre de errores.

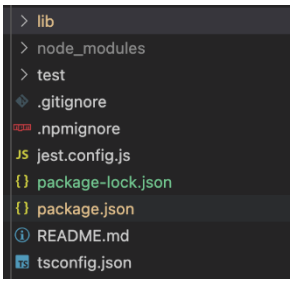
Puede utilizar el TDD para escribir primero el caso de prueba. Esto lo ayuda a validar la infraestructura con diferentes restricciones de diseño en términos de aplicar la política de seguridad para los recursos y seguir una convención de nomenclatura única para el proyecto. El enfoque estándar para probar AWS CDK aplicaciones es utilizar el módulo de AWS CDK aserciones y los marcos de prueba más populares, como Jest para JavaScript y/o TypeScript pytest para Python.

Hay dos categorías de pruebas que puedes escribir para tus aplicaciones: AWS CDK

- Usa afirmaciones detalladas para probar un aspecto específico de la CloudFormation plantilla generada, como «este recurso tiene esta propiedad con este valor». Estas pruebas pueden detectar regresiones y también son útiles cuando se desarrollan características nuevas con el TDD (primero hay que escribir una prueba y, después, hacer que se apruebe al escribir una implementación correcta). Las afirmaciones detalladas son las pruebas que escribirá con más frecuencia.
- Utilice pruebas instantáneas para comparar la CloudFormation plantilla sintetizada con una plantilla de referencia previamente almacenada. Las pruebas de instantáneas permiten refactorizar con libertad, ya que puede estar seguro de que el código refactorizado funciona exactamente de la misma manera que el original. Si los cambios eran intencionales, puede aceptar un punto de referencia nuevo para pruebas futuras. Sin embargo, AWS CDK las actualizaciones también pueden provocar cambios en las plantillas sintetizadas, por lo que no puede confiar únicamente en las instantáneas para asegurarse de que la implementación es correcta.

Prueba unitaria

Esta guía se centra TypeScript específicamente en la integración de las pruebas unitarias.
Para habilitar las pruebas, asegúrese de que el archivo `package.json` tenga las siguientes bibliotecas: `@types/jest`, `jest` y `ts-jest` en `devDependencies`. Para agregar estos paquetes, ejecute el comando `cdk init lib --language=typescript`. Tras ejecutar el comando anterior, verá la siguiente estructura.



El siguiente código es un ejemplo de un `package.json` archivo que está habilitado con la biblioteca 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",
    ...
  }
}
```

En la carpeta Prueba, puede escribir el caso de prueba. El siguiente ejemplo muestra un caso de prueba para una AWS CodePipeline construcción.

```
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
    });
});
```

Para realizar una prueba, ejecute el comando `npm run test` en su proyecto. La prueba devuelve los siguientes resultados.

```
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
```

Para obtener más información sobre los casos de prueba, consulte Probar construcciones en la Guía para AWS Cloud Development Kit (AWS CDK) desarrolladores.

Prueba de integración

Las pruebas de integración para AWS CDK construcciones también se pueden incluir mediante un `integ-tests` módulo. Una prueba de integración debe definirse como una AWS CDK aplicación. Debe haber una one-to-one relación entre una prueba de integración y una AWS CDK aplicación.

Para obtener más información, visita el integ-tests-alpha módulo de la referencia AWS CDK de la API.

Utilizar el control de versiones y lanzamiento para los constructos

Control de versiones para AWS CDK

AWS CDK Varios equipos pueden crear estructuras comunes y compartirlas en una organización para su consumo. Por lo general, los desarrolladores lanzan nuevas funciones o corrigen errores en sus AWS CDK construcciones comunes. Estas construcciones las utilizan las AWS CDK aplicaciones o cualquier otra AWS CDK construcción existente como parte de una dependencia. Por este motivo, es fundamental que los desarrolladores actualicen y publiquen su constructo con las versiones semánticas adecuadas de forma independiente. AWS CDK Las aplicaciones posteriores u otras AWS CDK construcciones pueden actualizar su dependencia para usar la versión de construcción publicada recientemente. AWS CDK

El control de versiones semántico (Semver) es un conjunto de reglas, o un método, para proporcionar números de software únicos al software del equipo. Las versiones se definen de la siguiente manera:

- Una versión IMPORTANTE consiste en cambios de API incompatibles o cambios importantes.
- Una versión MENOR consiste en una funcionalidad que se agrega de forma compatible con versiones anteriores.
- Una versión REVISIÓN consiste en correcciones de errores compatibles con versiones anteriores.

Para obtener más información sobre el control de versiones semántico, consulte la Especificación del control de versiones semántico (SemVer) en la documentación sobre el control de versiones semántico.

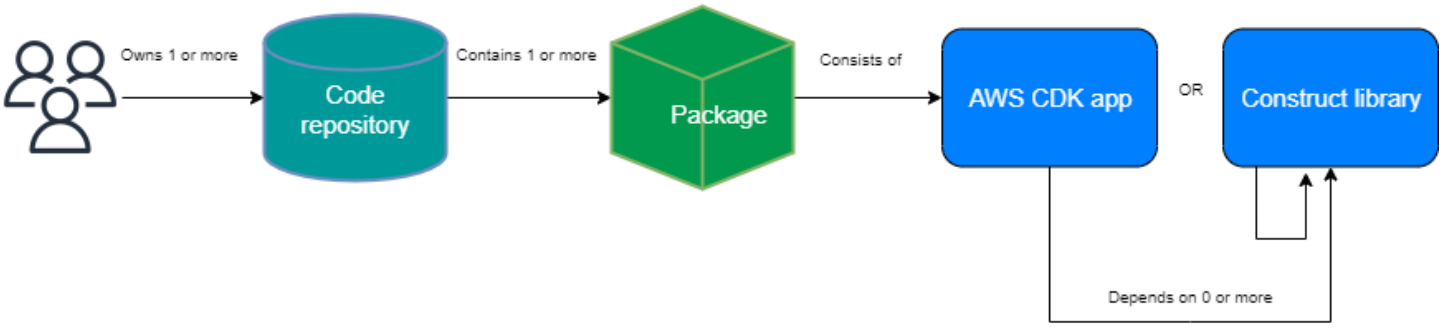
AWS CDK Repositorio y empaquetado para construcciones

Como AWS CDK las construcciones las desarrollan equipos distintos y son utilizadas por varias AWS CDK aplicaciones, puede utilizar un repositorio independiente para cada AWS CDK construcción. Esto también puede ayudarlo a aplicar el control de acceso. Cada repositorio puede contener todo el código fuente relacionado con la misma AWS CDK construcción junto con todas sus dependencias. Al mantener una sola aplicación (es decir, una AWS CDK construcción) en un único repositorio, se puede reducir el alcance del impacto de los cambios durante la implementación.

AWS CDK No solo genera CloudFormation plantillas para implementar la infraestructura, sino que también agrupa activos en tiempo de ejecución, como funciones Lambda e imágenes de Docker, y los implementa junto con su infraestructura. No solo es posible combinar el código que define la infraestructura y el código que implementa la lógica de tiempo de ejecución en una sola construcción, sino que es una práctica recomendada. No es necesario que estos dos tipos de código se encuentren en repositorios separados o incluso en paquetes separados.

Para consumir paquetes más allá de los límites del repositorio, debes tener un repositorio de paquetes privado, similar a npm o Maven Central PyPi, pero interno en tu organización. También debe tener un proceso de publicación que compile, pruebe y publique el paquete en el repositorio de paquetes privado. Puede crear repositorios privados, como un PyPi servidor, mediante una máquina virtual (VM) local o Amazon S3. Al diseñar o crear un registro de paquetes privado, es fundamental tener en cuenta el riesgo de interrupción del servicio debido a la alta disponibilidad y escalabilidad. Un servicio gestionado sin servidor alojado en la nube para almacenar paquetes puede reducir considerablemente la sobrecarga de mantenimiento. Por ejemplo, se puede utilizar AWS CodeArtifactpara alojar paquetes para los lenguajes de programación más populares. También se puede utilizar CodeArtifact para establecer conexiones de repositorios externos y replicarlas en ellas CodeArtifact.

El administrador de paquetes de tu idioma (por ejemplo, npm for TypeScript o JavaScript applications) gestiona las dependencias de los paquetes del repositorio de paquetes. El administrador de paquetes se asegura de que las compilaciones sean repetibles al registrar las versiones específicas de cada paquete del que depende su aplicación y, a continuación, le permite actualizar esas dependencias de forma controlada, como se muestra en el siguiente diagrama.

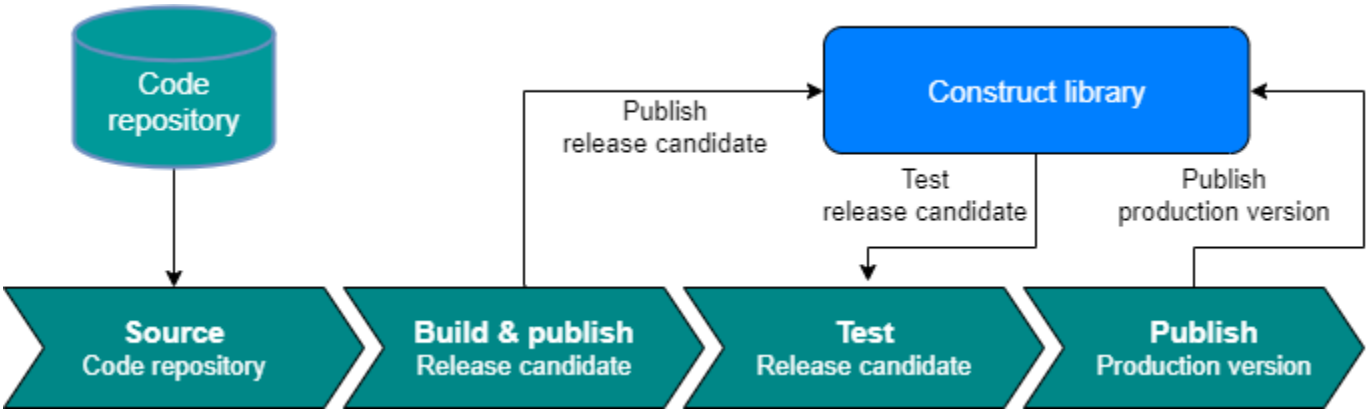


Construya una versión para AWS CDK

Le recomendamos que cree su propia canalización automatizada para crear y lanzar nuevas versiones de AWS CDK construcción. Si implementa un proceso de aprobación de solicitudes de extracción adecuado, una vez que confirme e inserte su código fuente en la rama principal del

repositorio, la canalización podrá compilar y crear una versión candidata a ser lanzada. Puedes actualizar esa versión CodeArtifact y probarla antes de lanzar la versión lista para producción. Si lo desea, puede probar la nueva versión de AWS CDK construcción localmente antes de fusionar el código con la rama principal. Esto hace que la canalización publique la versión lista para la producción. Tenga en cuenta que los constructos y los paquetes compartidos deben probarse por separado de la aplicación que los utilice, como si se estuvieran lanzando al público.

En el siguiente diagrama se muestra un ejemplo del proceso de publicación de una AWS CDK versión.



Puede utilizar los siguientes comandos de ejemplo para crear, probar y publicar paquetes npm. En primer lugar, inicie sesión en el repositorio de artefactos al ejecutar el siguiente comando.

```
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>
```

A continuación, complete los pasos siguientes:

1. Instale los paquetes necesarios en función del archivo `package.json`: `npm install`
2. Cree la versión candidata a ser lanzada: `npm version prerelease --preid rc`
3. Cree el paquete npm: `npm run build`
4. Pruebe el paquete npm: `npm run test`
5. Publique el paquete npm: `npm publish`

Aplicar la administración de versiones de bibliotecas

La administración del ciclo de vida es un desafío importante cuando se mantienen bases AWS CDK de código. Por ejemplo, supongamos que comienza un AWS CDK proyecto con la versión 1.97 y, después, la versión 1.169 estará disponible más adelante. La versión 1.169 ofrece características nuevas y correcciones de errores, pero ha implementado su infraestructura con la versión anterior. Ahora, actualizar los constructos se convierte en un desafío, ya que esta brecha aumenta debido a los cambios importantes que podrían introducirse en las versiones nuevas. Esto puede ser un desafío si tiene muchos recursos en su entorno. El patrón presentado en esta sección puede ayudarle a administrar la versión de la AWS CDK biblioteca mediante la automatización. Este es el flujo de trabajo de este patrón:

1. Al lanzar un nuevo producto de CodeArtifact Service Catalog, las versiones de la AWS CDK biblioteca y sus dependencias se almacenan en el `package.json` archivo.
2. Implementa una canalización común que realiza un seguimiento de todos los repositorios para que pueda aplicarles actualizaciones automáticas si no se producen cambios importantes.
3. Una AWS CodeBuild etapa comprueba el árbol de dependencias y busca los cambios más importantes.
4. La canalización crea una rama de característica y, a continuación, ejecuta `cdk synth` con la versión nueva para confirmar que no hay errores.
5. La versión nueva se implementa en el entorno de prueba y, finalmente, ejecuta una prueba de integración para asegurarse de que la implementación es correcta.
6. Puede utilizar dos colas de Amazon Simple Queue Service (Amazon SQS) para realizar un seguimiento de las pilas. Los usuarios pueden revisar las pilas de forma manual en la cola de excepciones y corregir los cambios importantes. Los elementos que superen la prueba de integración pueden fusionarse y publicarse.

Preguntas frecuentes

¿Qué problemas se pueden TypeScript resolver?

Por lo general, puede eliminar los errores del código al escribir pruebas automatizadas, verificar de forma manual que el código funciona según lo esperado y, por último, al hacer que otra persona lo valide. Validar las conexiones entre todas las partes del proyecto lleva mucho tiempo. Para acelerar el proceso de validación, puedes usar un lenguaje de tipografía comprobada, por ejemplo, TypeScript para automatizar la validación del código y proporcionar comentarios instantáneos durante el desarrollo.

¿Por qué debo usarlo? TypeScript

TypeScript es un lenguaje de código abierto que simplifica el JavaScript código, lo que facilita su lectura y depuración. TypeScript también proporciona herramientas de desarrollo altamente productivas para JavaScript IDE y prácticas, como la comprobación estática. Además, TypeScript ofrece las ventajas de ECMAScript 6 (ES6) y puede aumentar su productividad. Por último, TypeScript puede ayudarte a evitar los molestos errores que suelen encontrar los desarrolladores al escribir comprobando JavaScript el código.

¿Debo usar el AWS CDK o CloudFormation?

Le recomendamos que utilice el AWS Cloud Development Kit (AWS CDK) en lugar deAWS CloudFormation, si su organización tiene la experiencia en desarrollo para aprovechar elAWS CDK. Esto se debe a que AWS CDK es más flexible que CloudFormation, ya que puede utilizar un lenguaje de programación y conceptos de programación orientada a objetos. Tenga en cuenta que puede utilizarlos CloudFormation para crear AWS recursos de forma ordenada y predecible. En CloudFormation, los recursos se escriben en archivos de texto con el formato JSON o YAML.

¿Qué pasa si AWS CDK no es compatible con un lanzamiento reciente? Servicio de AWS

Puedes usar una anulación sin procesar o un recurso CloudFormation personalizado.

¿Cuáles son los diferentes lenguajes de programación compatibles con? AWS CDK

AWS CDKestá disponible generalmente en Python JavaScript TypeScript, Java, C# y Go (en la versión preliminar para desarrolladores).

¿Cuánto AWS CDK cuesta?

No hay ningún cargo adicional por elAWS CDK. Usted paga por los AWS recursos (como las instancias de Amazon EC2 o los balanceadores de carga de Elastic Load Balancing) que se crean cuando los usa AWS CDK de la misma manera que si los creara manualmente. Solo paga por lo que utiliza, cuando lo utiliza. No se requieren pagos mínimos ni compromisos iniciales.

Siguientes pasos

Le recomendamos que comience a construir desde AWS Cloud Development Kit (AWS CDK) dentro TypeScript. Para obtener más información, consulte el taller del día de AWS CDK inmersión.

Recursos

Referencias

- AWS Construcciones de AWS soluciones (soluciones)
- AWS Kits de desarrollo en la nube (AWS CDK) (GitHub)
- AWS Referencia de la API de Construct Library (documentación de AWS CDK referencia)
- AWS CDK Documentación de referencia (documentación AWS CDK de referencia)
- AWS CDK Taller de un día de inmersión (AWS taller de estudio)

Herramientas

- cdk-nag () GitHub
- TypeScript ESLint (documentación de ESLint) TypeScript

Guías y patrones

- AWS Soluciones: construye patrones (documentación)AWS

Historial del documento

En la siguiente tabla, se describen cambios significativos de esta guía. Si quiere recibir notificaciones de futuras actualizaciones, puede suscribirse a las notificaciones RSS.

Cambio	Descripción	Fecha
Actualice el código	Hemos actualizado los ejemplos de código en la sección Siga las TypeScript mejores prácticas.	16 de febrero de 2024
Añadir secciones	Hemos añadido las secciones Uso de tipos de utilidades y Pruebas de integración.	10 de enero de 2024
Actualización menor	Ejemplo de código actualizado para crear un constructo de C3.	16 de junio de 2023
Publicación inicial	—	8 de diciembre de 2022

AWS Glosario de orientación prescriptiva

Los siguientes son términos de uso común en las estrategias, guías y patrones proporcionados por AWS Prescriptive Guidance. Para sugerir entradas, utilice el enlace Enviar comentarios al final del glosario.

Números

Las 7 R

Siete estrategias de migración comunes para trasladar aplicaciones a la nube. Estas estrategias se basan en las 5 R que Gartner identificó en 2011 y consisten en lo siguiente:

- Refactorizar/rediseñar: traslade una aplicación y modifique su arquitectura mediante el máximo aprovechamiento de las características nativas en la nube para mejorar la agilidad, el rendimiento y la escalabilidad. Por lo general, esto implica trasladar el sistema operativo y la base de datos. Ejemplo: Migre la base de datos de Oracle en las instalaciones a Amazon Aurora PostgreSQL-Compatible Edition.
- Redefinir la plataforma (transportar y redefinir): traslade una aplicación a la nube e introduzca algún nivel de optimización para aprovechar las capacidades de la nube. Ejemplo: migre su base de datos Oracle local a Amazon Relational Database Service (Amazon RDS) para Oracle in the Cloud. AWS
- Recomprar (readquirir): cambie a un producto diferente, lo cual se suele llevar a cabo al pasar de una licencia tradicional a un modelo SaaS. Ejemplo: Migre el sistema de administración de las relaciones con los clientes (CRM) a Salesforce.com.
- Volver a alojar (migrar mediante lift-and-shift): traslade una aplicación a la nube sin realizar cambios para aprovechar las capacidades de la nube. Ejemplo: migre su base de datos Oracle local a Oracle en una instancia EC2 en la nube. AWS
- Reubicar: (migrar el hipervisor mediante lift and shift): traslade la infraestructura a la nube sin comprar equipo nuevo, reescribir aplicaciones o modificar las operaciones actuales. Este escenario de migración es específico de VMware Cloud on AWS, que admite la compatibilidad de máquinas virtuales (VM) y la portabilidad de las cargas de trabajo entre su entorno local y. AWS Puede utilizar las tecnologías de VMware Cloud Foundation desde los centros de datos en las instalaciones al migrar una infraestructura a VMware Cloud en AWS. Ejemplo: traslade el hipervisor que aloja su base de datos de Oracle a VMware Cloud on. AWS

- Retener (revisitar): conserve las aplicaciones en el entorno de origen. Estas pueden incluir las aplicaciones que requieren una refactorización importante, que desee posponer para más adelante, y las aplicaciones heredadas que desee retener, ya que no hay ninguna justificación empresarial para migrarlas.
- Retirar: retire o elimine las aplicaciones que ya no sean necesarias en un entorno de origen.

A

ABAC

Consulte el control de acceso basado en atributos.

servicios abstractos

Consulte servicios gestionados.

ACID

Consulte atomicidad, consistencia, aislamiento y durabilidad.

migración activa-activa

Método de migración de bases de datos en el que las bases de datos de origen y destino se mantienen sincronizadas (mediante una herramienta de replicación bidireccional o mediante operaciones de escritura doble) y ambas bases de datos gestionan las transacciones de las aplicaciones conectadas durante la migración. Este método permite la migración en lotes pequeños y controlados, en lugar de requerir una transición única. Es más flexible, pero requiere más trabajo que la migración activa-pasiva.

migración activa-pasiva

Método de migración de bases de datos en el que las bases de datos de origen y destino se mantienen sincronizadas, pero solo la base de datos de origen gestiona las transacciones de las aplicaciones conectadas, mientras los datos se replican en la base de datos de destino. La base de datos de destino no acepta ninguna transacción durante la migración.

función agregada

Función SQL que opera en un grupo de filas y calcula un único valor de retorno para el grupo. Entre los ejemplos de funciones agregadas se incluyen `SUM` y `MAX`.

IA

Véase inteligencia artificial.

AIOps

Consulte las operaciones de inteligencia artificial.

anonimización

El proceso de eliminar permanentemente la información personal de un conjunto de datos. La anonimización puede ayudar a proteger la privacidad personal. Los datos anonimizados ya no se consideran datos personales.

antipatrones

Una solución que se utiliza con frecuencia para un problema recurrente en el que la solución es contraproducente, ineficaz o menos eficaz que una alternativa.

control de aplicaciones

Un enfoque de seguridad que permite el uso únicamente de aplicaciones aprobadas para ayudar a proteger un sistema contra el malware.

cartera de aplicaciones

Recopilación de información detallada sobre cada aplicación que utiliza una organización, incluido el costo de creación y mantenimiento de la aplicación y su valor empresarial. Esta información es clave para el proceso de detección y análisis de la cartera y ayuda a identificar y priorizar las aplicaciones que se van a migrar, modernizar y optimizar.

inteligencia artificial (IA)

El campo de la informática que se dedica al uso de tecnologías informáticas para realizar funciones cognitivas que suelen estar asociadas a los seres humanos, como el aprendizaje, la resolución de problemas y el reconocimiento de patrones. Para más información, consulte ¿Qué es la inteligencia artificial?

operaciones de inteligencia artificial (AIOps)

El proceso de utilizar técnicas de machine learning para resolver problemas operativos, reducir los incidentes operativos y la intervención humana, y mejorar la calidad del servicio. Para obtener más información sobre cómo se utiliza AIOps en la estrategia de migración de AWS , consulte la Guía de integración de operaciones.

cifrado asimétrico

Algoritmo de cifrado que utiliza un par de claves, una clave pública para el cifrado y una clave privada para el descifrado. Puede compartir la clave pública porque no se utiliza para el descifrado, pero el acceso a la clave privada debe estar sumamente restringido.

atomicidad, consistencia, aislamiento, durabilidad (ACID)

Conjunto de propiedades de software que garantizan la validez de los datos y la fiabilidad operativa de una base de datos, incluso en caso de errores, cortes de energía u otros problemas.

control de acceso basado en atributos (ABAC)

La práctica de crear permisos detallados basados en los atributos del usuario, como el departamento, el puesto de trabajo y el nombre del equipo. Para obtener más información, consulte ABAC AWS en la documentación AWS Identity and Access Management (IAM).

origen de datos fidedigno

Ubicación en la que se almacena la versión principal de los datos, que se considera la fuente de información más fiable. Puede copiar los datos del origen de datos autorizado a otras ubicaciones con el fin de procesarlos o modificarlos, por ejemplo, anonimizarlos, redactarlos o seudonimizarlos.

Zona de disponibilidad

Una ubicación distinta dentro de una Región de AWS que está aislada de los fallos en otras zonas de disponibilidad y que proporciona una conectividad de red económica y de baja latencia a otras zonas de disponibilidad de la misma región.

AWS Marco de adopción de la nube (AWS CAF)

Un marco de directrices y mejores prácticas AWS para ayudar a las organizaciones a desarrollar un plan eficiente y eficaz para migrar con éxito a la nube. AWS CAF organiza la orientación en seis áreas de enfoque denominadas perspectivas: negocios, personas, gobierno, plataforma, seguridad y operaciones. Las perspectivas empresariales, humanas y de gobernanza se centran en las habilidades y los procesos empresariales; las perspectivas de plataforma, seguridad y operaciones se centran en las habilidades y los procesos técnicos. Por ejemplo, la perspectiva humana se dirige a las partes interesadas que se ocupan de los Recursos Humanos (RR. HH.), las funciones del personal y la administración de las personas. Desde esta perspectiva, AWS CAF proporciona orientación para el desarrollo, la formación y la comunicación de las personas a fin de preparar a la organización para una adopción exitosa de la nube. Para obtener más información, consulte la Página web de AWS CAF y el Documento técnico de AWS CAF.

AWS Marco de calificación de la carga de trabajo (AWS WQF)

Herramienta que evalúa las cargas de trabajo de migración de bases de datos, recomienda estrategias de migración y proporciona estimaciones de trabajo. AWS WQF se incluye con AWS Schema Conversion Tool ().AWS SCT Analiza los esquemas de bases de datos y los objetos de código, el código de las aplicaciones, las dependencias y las características de rendimiento y proporciona informes de evaluación.

B

Un bot malo

Un bot destinado a interrumpir o causar daño a personas u organizaciones.

BCP

Consulte la planificación de la continuidad del negocio.

gráfico de comportamiento

Una vista unificada e interactiva del comportamiento de los recursos y de las interacciones a lo largo del tiempo. Puede utilizar un gráfico de comportamiento con Amazon Detective para examinar los intentos de inicio de sesión fallidos, las llamadas sospechosas a la API y acciones similares. Para obtener más información, consulte Datos en un gráfico de comportamiento en la documentación de Detective.

sistema big-endian

Un sistema que almacena primero el byte más significativo. Véase también endianness.

clasificación binaria

Un proceso que predice un resultado binario (una de las dos clases posibles). Por ejemplo, es posible que su modelo de ML necesite predecir problemas como “¿Este correo electrónico es spam o no es spam?” o “¿Este producto es un libro o un automóvil?”.

filtro de floración

Estructura de datos probabilística y eficiente en términos de memoria que se utiliza para comprobar si un elemento es miembro de un conjunto.

implementación azul/verde

Una estrategia de despliegue en la que se crean dos entornos separados pero idénticos. La versión actual de la aplicación se ejecuta en un entorno (azul) y la nueva versión de la aplicación en el otro entorno (verde). Esta estrategia le ayuda a revertirla rápidamente con un impacto mínimo.

bot

Una aplicación de software que ejecuta tareas automatizadas a través de Internet y simula la actividad o interacción humana. Algunos bots son útiles o beneficiosos, como los rastreadores web que indexan información en Internet. Algunos otros bots, conocidos como bots malos, tienen como objetivo interrumpir o causar daños a personas u organizaciones.

botnet

Redes de bots que están infectadas por malware y que están bajo el control de una sola parte, conocida como pastor u operador de bots. Las botnets son el mecanismo más conocido para escalar los bots y su impacto.

rama

Área contenida de un repositorio de código. La primera rama que se crea en un repositorio es la rama principal. Puede crear una rama nueva a partir de una rama existente y, a continuación, desarrollar características o corregir errores en la rama nueva. Una rama que se genera para crear una característica se denomina comúnmente rama de característica. Cuando la característica se encuentra lista para su lanzamiento, se vuelve a combinar la rama de característica con la rama principal. Para obtener más información, consulte Acerca de las sucursales (GitHub documentación).

acceso con cristales rotos

En circunstancias excepcionales y mediante un proceso aprobado, un usuario puede acceder rápidamente a un sitio para el Cuenta de AWS que normalmente no tiene permisos de acceso. Para obtener más información, consulte el indicador Implemente procedimientos de rotura de cristales en la guía Well-Architected AWS .

estrategia de implementación sobre infraestructura existente

La infraestructura existente en su entorno. Al adoptar una estrategia de implementación sobre infraestructura existente para una arquitectura de sistemas, se diseña la arquitectura en función de las limitaciones de los sistemas y la infraestructura actuales. Si está ampliando

la infraestructura existente, puede combinar las estrategias de implementación sobre infraestructuras existentes y de implementación desde cero.

caché de búfer

El área de memoria donde se almacenan los datos a los que se accede con más frecuencia.

capacidad empresarial

Lo que hace una empresa para generar valor (por ejemplo, ventas, servicio al cliente o marketing). Las arquitecturas de microservicios y las decisiones de desarrollo pueden estar impulsadas por las capacidades empresariales. Para obtener más información, consulte la sección Organizado en torno a las capacidades empresariales del documento técnico Ejecutar microservicios en contenedores en AWS.

planificación de la continuidad del negocio (BCP)

Plan que aborda el posible impacto de un evento disruptivo, como una migración a gran escala en las operaciones y permite a la empresa reanudar las operaciones rápidamente.

C

CAF

Consulte el marco AWS de adopción de la nube.

despliegue canario

El lanzamiento lento e incremental de una versión para los usuarios finales. Cuando se tiene confianza, se despliega la nueva versión y se reemplaza la versión actual en su totalidad.

CCoE

Consulte el Centro de excelencia en la nube.

CDC

Consulte la captura de datos de cambios.

captura de datos de cambio (CDC)

Proceso de seguimiento de los cambios en un origen de datos, como una tabla de base de datos, y registro de los metadatos relacionados con el cambio. Puede utilizar los CDC para

diversos fines, como auditar o replicar los cambios en un sistema de destino para mantener la sincronización.

ingeniería del caos

Introducir intencionalmente fallos o eventos disruptivos para poner a prueba la resiliencia de un sistema. Puedes usar AWS Fault Injection Service (AWS FIS) para realizar experimentos que estresen tus AWS cargas de trabajo y evalúen su respuesta.

CI/CD

Consulte la integración continua y la entrega continua.

clasificación

Un proceso de categorización que permite generar predicciones. Los modelos de ML para problemas de clasificación predicen un valor discreto. Los valores discretos siempre son distintos entre sí. Por ejemplo, es posible que un modelo necesite evaluar si hay o no un automóvil en una imagen.

cifrado del cliente

Cifrado de datos localmente, antes de que el objetivo los Servicio de AWS reciba.

Centro de excelencia en la nube (CCoE)

Equipo multidisciplinario que impulsa los esfuerzos de adopción de la nube en toda la organización, incluido el desarrollo de las prácticas recomendadas en la nube, la movilización de recursos, el establecimiento de plazos de migración y la dirección de la organización durante las transformaciones a gran escala. Para obtener más información, consulte las publicaciones de CCoE en el blog de estrategia empresarial en la AWS nube.

computación en la nube

La tecnología en la nube que se utiliza normalmente para la administración de dispositivos de IoT y el almacenamiento de datos de forma remota. La computación en la nube suele estar conectada a la tecnología de computación perimetral.

modelo operativo en la nube

En una organización de TI, el modelo operativo que se utiliza para crear, madurar y optimizar uno o más entornos de nube. Para obtener más información, consulte Creación de su modelo operativo de nube.

etapas de adopción de la nube

Las cuatro fases por las que suelen pasar las organizaciones cuando migran a la AWS nube:

- Proyecto: ejecución de algunos proyectos relacionados con la nube con fines de prueba de concepto y aprendizaje
- Fundamento: realización de inversiones fundamentales para escalar la adopción de la nube (p. ej., crear una zona de aterrizaje, definir un CCoE, establecer un modelo de operaciones)
- Migración: migración de aplicaciones individuales
- Reinvención: optimización de productos y servicios e innovación en la nube

Stephen Orban definió estas etapas en la entrada del blog The Journey Toward Cloud-First & the Stages of Adoption, del blog AWS Cloud Enterprise Strategy. Para obtener información sobre su relación con la estrategia de AWS migración, consulte la guía de preparación para la migración.

CMDB

Consulte la base de datos de administración de la configuración.

repositorio de código

Una ubicación donde el código fuente y otros activos, como documentación, muestras y scripts, se almacenan y actualizan mediante procesos de control de versiones. Los repositorios en la nube más comunes incluyen GitHub o AWS CodeCommit. Cada versión del código se denomina rama. En una estructura de microservicios, cada repositorio se encuentra dedicado a una única funcionalidad. Una sola canalización de CI/CD puede utilizar varios repositorios.

caché en frío

Una caché de búfer que está vacía no está bien poblada o contiene datos obsoletos o irrelevantes. Esto afecta al rendimiento, ya que la instancia de la base de datos debe leer desde la memoria principal o el disco, lo que es más lento que leer desde la memoria caché del búfer.

datos fríos

Datos a los que se accede con poca frecuencia y que suelen ser históricos. Al consultar este tipo de datos, normalmente se aceptan consultas lentas. Trasladar estos datos a niveles o clases de almacenamiento de menor rendimiento y menos costosos puede reducir los costos.

visión artificial (CV)

Campo de la IA que utiliza el aprendizaje automático para analizar y extraer información de formatos visuales, como imágenes y vídeos digitales. Por ejemplo, AWS Panorama ofrece

dispositivos que añaden CV a las redes de cámaras locales, y Amazon SageMaker proporciona algoritmos de procesamiento de imágenes para CV.

desviación de configuración

En el caso de una carga de trabajo, un cambio de configuración con respecto al estado esperado. Puede provocar que la carga de trabajo deje de cumplir las normas y, por lo general, es gradual e involuntario.

base de datos de administración de configuración (CMDB)

Repositorio que almacena y administra información sobre una base de datos y su entorno de TI, incluidos los componentes de hardware y software y sus configuraciones. Por lo general, los datos de una CMDB se utilizan en la etapa de detección y análisis de la cartera de productos durante la migración.

paquete de conformidad

Conjunto de AWS Config reglas y medidas correctivas que puede reunir para personalizar sus comprobaciones de conformidad y seguridad. Puede implementar un paquete de conformidad como una entidad única en una región Cuenta de AWS y, o en una organización, mediante una plantilla YAML. Para obtener más información, consulta los paquetes de conformidad en la documentación. AWS Config

integración y entrega continuas (CI/CD)

El proceso de automatización de las etapas de origen, compilación, prueba, presentación y producción del proceso de lanzamiento del software. La CI/CD se describe comúnmente como una canalización. La CI/CD puede ayudarlo a automatizar los procesos, mejorar la productividad, mejorar la calidad del código y entregar con mayor rapidez. Para obtener más información, consulte Beneficios de la entrega continua. CD también puede significar implementación continua. Para obtener más información, consulte Entrega continua frente a implementación continua.

CV

Consulte visión artificial.

D

datos en reposo

Datos que están estacionarios en la red, como los datos que se encuentran almacenados.

clasificación de datos

Un proceso para identificar y clasificar los datos de su red en función de su importancia y sensibilidad. Es un componente fundamental de cualquier estrategia de administración de riesgos de ciberseguridad porque lo ayuda a determinar los controles de protección y retención adecuados para los datos. La clasificación de datos es un componente del pilar de seguridad del AWS Well-Architected Framework. Para obtener más información, consulte Clasificación de datos.

desviación de datos

Una variación significativa entre los datos de producción y los datos que se utilizaron para entrenar un modelo de machine learning, o un cambio significativo en los datos de entrada a lo largo del tiempo. La desviación de los datos puede reducir la calidad, la precisión y la imparcialidad generales de las predicciones de los modelos de machine learning.

datos en tránsito

Datos que se mueven de forma activa por la red, por ejemplo, entre los recursos de la red.

malla de datos

Un marco arquitectónico que proporciona una propiedad de datos distribuida y descentralizada con una administración y un gobierno centralizados.

minimización de datos

El principio de recopilar y procesar solo los datos estrictamente necesarios. Practicar la minimización de los datos Nube de AWS puede reducir los riesgos de privacidad, los costos y la huella de carbono de la analítica.

perímetro de datos

Un conjunto de barreras preventivas en su AWS entorno que ayudan a garantizar que solo las identidades confiables accedan a los recursos confiables desde las redes esperadas. Para obtener más información, consulte Crear un perímetro de datos sobre. AWS

preprocesamiento de datos

Transformar los datos sin procesar en un formato que su modelo de ML pueda analizar fácilmente. El preprocesamiento de datos puede implicar eliminar determinadas columnas o filas y corregir los valores faltantes, incoherentes o duplicados.

procedencia de los datos

El proceso de rastrear el origen y el historial de los datos a lo largo de su ciclo de vida, por ejemplo, la forma en que se generaron, transmitieron y almacenaron los datos.

titular de los datos

Persona cuyos datos se recopilan y procesan.

almacenamiento de datos

Un sistema de administración de datos que respalde la inteligencia empresarial, como el análisis. Los almacenes de datos suelen contener grandes cantidades de datos históricos y, por lo general, se utilizan para consultas y análisis.

lenguaje de definición de datos (DDL)

Instrucciones o comandos para crear o modificar la estructura de tablas y objetos de una base de datos.

lenguaje de manipulación de datos (DML)

Instrucciones o comandos para modificar (insertar, actualizar y eliminar) la información de una base de datos.

DDL

Consulte el lenguaje de definición de bases de datos.

conjunto profundo

Combinar varios modelos de aprendizaje profundo para la predicción. Puede utilizar conjuntos profundos para obtener una predicción más precisa o para estimar la incertidumbre de las predicciones.

aprendizaje profundo

Un subcampo del ML que utiliza múltiples capas de redes neuronales artificiales para identificar el mapeo entre los datos de entrada y las variables objetivo de interés.

defense-in-depth

Un enfoque de seguridad de la información en el que se distribuyen cuidadosamente una serie de mecanismos y controles de seguridad en una red informática para proteger la confidencialidad, la integridad y la disponibilidad de la red y de los datos que contiene. Al adoptar esta estrategia AWS, se añaden varios controles en diferentes capas de la AWS Organizations estructura para

ayudar a proteger los recursos. Por ejemplo, un defense-in-depth enfoque podría combinar la autenticación multifactorial, la segmentación de la red y el cifrado.

administrador delegado

En AWS Organizations, un servicio compatible puede registrar una cuenta de AWS miembro para administrar las cuentas de la organización y gestionar los permisos de ese servicio. Esta cuenta se denomina administrador delegado para ese servicio. Para obtener más información y una lista de servicios compatibles, consulte Servicios que funcionan con AWS Organizations en la documentación de AWS Organizations .

Implementación

El proceso de hacer que una aplicación, características nuevas o correcciones de código se encuentren disponibles en el entorno de destino. La implementación abarca implementar cambios en una base de código y, a continuación, crear y ejecutar esa base en los entornos de la aplicación.

entorno de desarrollo

Consulte entorno.

control de detección

Un control de seguridad que se ha diseñado para detectar, registrar y alertar después de que se produzca un evento. Estos controles son una segunda línea de defensa, ya que lo advierten sobre los eventos de seguridad que han eludido los controles preventivos establecidos. Para obtener más información, consulte Controles de detección en Implementación de controles de seguridad en AWS.

asignación de flujos de valor para el desarrollo (DVSM)

Proceso que se utiliza para identificar y priorizar las restricciones que afectan negativamente a la velocidad y la calidad en el ciclo de vida del desarrollo de software. DVSM amplía el proceso de asignación del flujo de valor diseñado originalmente para las prácticas de fabricación ajustada. Se centra en los pasos y los equipos necesarios para crear y transferir valor a través del proceso de desarrollo de software.

gemelo digital

Representación virtual de un sistema del mundo real, como un edificio, una fábrica, un equipo industrial o una línea de producción. Los gemelos digitales son compatibles con el mantenimiento predictivo, la supervisión remota y la optimización de la producción.

tabla de dimensiones

En un esquema en estrella, tabla más pequeña que contiene los atributos de datos sobre los datos cuantitativos de una tabla de hechos. Los atributos de la tabla de dimensiones suelen ser campos de texto o números discretos que se comportan como texto. Estos atributos se utilizan habitualmente para restringir consultas, filtrar y etiquetar conjuntos de resultados.

desastre

Un evento que impide que una carga de trabajo o un sistema cumplan sus objetivos empresariales en su ubicación principal de implementación. Estos eventos pueden ser desastres naturales, fallos técnicos o el resultado de acciones humanas, como una configuración incorrecta involuntaria o un ataque de malware.

recuperación de desastres (DR)

La estrategia y el proceso que se utilizan para minimizar el tiempo de inactividad y la pérdida de datos ocasionados por un desastre. Para obtener más información, consulte Recuperación ante desastres de cargas de trabajo en AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Consulte el lenguaje de manipulación de bases de datos.

diseño basado en el dominio

Un enfoque para desarrollar un sistema de software complejo mediante la conexión de sus componentes a dominios en evolución, o a los objetivos empresariales principales, a los que sirve cada componente. Este concepto lo introdujo Eric Evans en su libro, Diseño impulsado por el dominio: abordando la complejidad en el corazón del software (Boston: Addison-Wesley Professional, 2003). Para obtener información sobre cómo utilizar el diseño basado en dominios con el patrón de higos estranguladores, consulte Modernización gradual de los servicios web antiguos de Microsoft ASP.NET (ASMX) mediante contenedores y Amazon API Gateway.

DR

Consulte recuperación ante desastres.

detección de deriva

Seguimiento de las desviaciones con respecto a una configuración de referencia. Por ejemplo, puedes usarlo AWS CloudFormation para detectar desviaciones en los recursos del sistema o

puedes usarlo AWS Control Tower para detectar cambios en tu landing zone que puedan afectar al cumplimiento de los requisitos de gobierno.

DVSM

Consulte el mapeo del flujo de valor del desarrollo.

E

EDA

Consulte el análisis exploratorio de datos.

computación en la periferia

La tecnología que aumenta la potencia de cálculo de los dispositivos inteligentes en la periferia de una red de IoT. En comparación con la computación en nube, la computación perimetral puede reducir la latencia de la comunicación y mejorar el tiempo de respuesta.

cifrado

Proceso informático que transforma datos de texto plano, legibles por humanos, en texto cifrado.

clave de cifrado

Cadena criptográfica de bits aleatorios que se genera mediante un algoritmo de cifrado. Las claves pueden variar en longitud y cada una se ha diseñado para ser impredecible y única.

endianidad

El orden en el que se almacenan los bytes en la memoria del ordenador. Los sistemas big-endianos almacenan primero el byte más significativo. Los sistemas Little-Endian almacenan primero el byte menos significativo.

punto de conexión

Consulte el punto final del servicio.

servicio de punto de conexión

Servicio que puede alojar en una nube privada virtual (VPC) para compartir con otros usuarios. Puede crear un servicio de punto final AWS PrivateLink y conceder permisos a otros directores Cuentas de AWS o a AWS Identity and Access Management (IAM). Estas cuentas o entidades

principales pueden conectarse a su servicio de punto de conexión de forma privada mediante la creación de puntos de conexión de VPC de interfaz. Para obtener más información, consulte Creación de un servicio de punto de conexión en la documentación de Amazon Virtual Private Cloud (Amazon VPC).

planificación de recursos empresariales (ERP)

Un sistema que automatiza y gestiona los procesos empresariales clave (como la contabilidad, el MES y la gestión de proyectos) de una empresa.

cifrado de sobre

El proceso de cifrar una clave de cifrado con otra clave de cifrado. Para obtener más información, consulte el cifrado de sobres en la documentación de AWS Key Management Service (AWS KMS).

environment

Una instancia de una aplicación en ejecución. Los siguientes son los tipos de entornos más comunes en la computación en la nube:

- entorno de desarrollo: instancia de una aplicación en ejecución que solo se encuentra disponible para el equipo principal responsable del mantenimiento de la aplicación. Los entornos de desarrollo se utilizan para probar los cambios antes de promocionarlos a los entornos superiores. Este tipo de entorno a veces se denomina entorno de prueba.
- entornos inferiores: todos los entornos de desarrollo de una aplicación, como los que se utilizan para las compilaciones y pruebas iniciales.
- entorno de producción: instancia de una aplicación en ejecución a la que pueden acceder los usuarios finales. En una canalización de CI/CD, el entorno de producción es el último entorno de implementación.
- entornos superiores: todos los entornos a los que pueden acceder usuarios que no sean del equipo de desarrollo principal. Esto puede incluir un entorno de producción, entornos de preproducción y entornos para las pruebas de aceptación por parte de los usuarios.

epopeya

En las metodologías ágiles, son categorías funcionales que ayudan a organizar y priorizar el trabajo. Las epopeyas brindan una descripción detallada de los requisitos y las tareas de implementación. Por ejemplo, las epopeyas AWS de seguridad de CAF incluyen la gestión de identidades y accesos, los controles de detección, la seguridad de la infraestructura, la protección

de datos y la respuesta a incidentes. Para obtener más información sobre las epopeyas en la estrategia de migración de AWS , consulte la Guía de implementación del programa.

PERP

Consulte planificación de recursos empresariales.

análisis de datos de tipo exploratorio (EDA)

El proceso de analizar un conjunto de datos para comprender sus características principales. Se recopilan o agregan datos y, a continuación, se realizan las investigaciones iniciales para encontrar patrones, detectar anomalías y comprobar las suposiciones. El EDA se realiza mediante el cálculo de estadísticas resumidas y la creación de visualizaciones de datos.

F

tabla de datos

La tabla central de un esquema en forma de estrella. Almacena datos cuantitativos sobre las operaciones comerciales. Normalmente, una tabla de hechos contiene dos tipos de columnas: las que contienen medidas y las que contienen una clave externa para una tabla de dimensiones.

fallan rápidamente

Una filosofía que utiliza pruebas frecuentes e incrementales para reducir el ciclo de vida del desarrollo. Es una parte fundamental de un enfoque ágil.

límite de aislamiento de fallas

En el Nube de AWS, un límite, como una zona de disponibilidad Región de AWS, un plano de control o un plano de datos, que limita el efecto de una falla y ayuda a mejorar la resiliencia de las cargas de trabajo. Para obtener más información, consulte Límites de AWS aislamiento de errores.

rama de característica

Consulte la sucursal.

características

Los datos de entrada que se utilizan para hacer una predicción. Por ejemplo, en un contexto de fabricación, las características pueden ser imágenes que se capturan periódicamente desde la línea de fabricación.

importancia de las características

La importancia que tiene una característica para las predicciones de un modelo. Por lo general, esto se expresa como una puntuación numérica que se puede calcular mediante diversas técnicas, como las explicaciones aditivas de Shapley (SHAP) y los gradientes integrados. Para obtener más información, consulte Interpretabilidad del modelo de aprendizaje automático con:AWS.

transformación de funciones

Optimizar los datos para el proceso de ML, lo que incluye enriquecer los datos con fuentes adicionales, escalar los valores o extraer varios conjuntos de información de un solo campo de datos. Esto permite que el modelo de ML se beneficie de los datos. Por ejemplo, si divide la fecha del “27 de mayo de 2021 00:15:37” en “jueves”, “mayo”, “2021” y “15”, puede ayudar al algoritmo de aprendizaje a aprender patrones matizados asociados a los diferentes componentes de los datos.

FGAC

Consulte el control de acceso detallado.

control de acceso preciso (FGAC)

El uso de varias condiciones que tienen por objetivo permitir o denegar una solicitud de acceso.

migración relámpago

Método de migración de bases de datos que utiliza la replicación continua de datos mediante la captura de datos modificados para migrar los datos en el menor tiempo posible, en lugar de utilizar un enfoque gradual. El objetivo es reducir al mínimo el tiempo de inactividad.

G

bloqueo geográfico

Consulta las restricciones geográficas.

restricciones geográficas (bloqueo geográfico)

En Amazon CloudFront, una opción para impedir que los usuarios de países específicos accedan a las distribuciones de contenido. Puede utilizar una lista de permitidos o bloqueados para especificar los países aprobados y prohibidos. Para obtener más información, consulta Restringir la distribución geográfica del contenido en la CloudFront documentación.

Flujo de trabajo de Gitflow

Un enfoque en el que los entornos inferiores y superiores utilizan diferentes ramas en un repositorio de código fuente. El flujo de trabajo de Gitflow se considera heredado, y el flujo de trabajo basado en enlaces troncales es el enfoque moderno preferido.

estrategia de implementación desde cero

La ausencia de infraestructura existente en un entorno nuevo. Al adoptar una estrategia de implementación desde cero para una arquitectura de sistemas, puede seleccionar todas las tecnologías nuevas sin que estas deban ser compatibles con una infraestructura existente, lo que también se conoce como implementación sobre infraestructura existente. Si está ampliando la infraestructura existente, puede combinar las estrategias de implementación sobre infraestructuras existentes y de implementación desde cero.

barrera de protección

Una regla de alto nivel que ayuda a regular los recursos, las políticas y la conformidad en todas las unidades organizativas (OU). Las barreras de protección preventivas aplican políticas para garantizar la alineación con los estándares de conformidad. Se implementan mediante políticas de control de servicios y límites de permisos de IAM. Las barreras de protección de detección detectan las vulneraciones de las políticas y los problemas de conformidad, y generan alertas para su corrección. Se implementan mediante Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, Amazon Inspector y AWS Lambda cheques personalizados.

H

JA

Consulte alta disponibilidad.

migración heterogénea de bases de datos

Migración de la base de datos de origen a una base de datos de destino que utilice un motor de base de datos diferente (por ejemplo, de Oracle a Amazon Aurora). La migración heterogénea suele ser parte de un esfuerzo de rediseño de la arquitectura y convertir el esquema puede ser una tarea compleja. AWS ofrece AWS SCT, lo cual ayuda con las conversiones de esquemas.

alta disponibilidad (HA)

La capacidad de una carga de trabajo para funcionar de forma continua, sin intervención, en caso de desafíos o desastres. Los sistemas de alta disponibilidad están diseñados para realizar una

conmutación por error automática, ofrecer un rendimiento de alta calidad de forma constante y gestionar diferentes cargas y fallos con un impacto mínimo en el rendimiento.

modernización histórica

Un enfoque utilizado para modernizar y actualizar los sistemas de tecnología operativa (TO) a fin de satisfacer mejor las necesidades de la industria manufacturera. Un histórico es un tipo de base de datos que se utiliza para recopilar y almacenar datos de diversas fuentes en una fábrica.

migración homogénea de bases de datos

Migración de la base de datos de origen a una base de datos de destino que comparte el mismo motor de base de datos (por ejemplo, Microsoft SQL Server a Amazon RDS para SQL Server). La migración homogénea suele formar parte de un esfuerzo para volver a alojar o redefinir la plataforma. Puede utilizar las utilidades de bases de datos nativas para migrar el esquema.

datos recientes

Datos a los que se accede con frecuencia, como datos en tiempo real o datos traslacionales recientes. Por lo general, estos datos requieren un nivel o una clase de almacenamiento de alto rendimiento para proporcionar respuestas rápidas a las consultas.

hotfix

Una solución urgente para un problema crítico en un entorno de producción. Debido a su urgencia, las revisiones suelen realizarse fuera del flujo de trabajo habitual de las DevOps versiones.

periodo de hiperatención

Periodo, inmediatamente después de la transición, durante el cual un equipo de migración administra y monitorea las aplicaciones migradas en la nube para solucionar cualquier problema. Por lo general, este periodo dura de 1 a 4 días. Al final del periodo de hiperatención, el equipo de migración suele transferir la responsabilidad de las aplicaciones al equipo de operaciones en la nube.

I

IaC

Vea la infraestructura como código.

políticas basadas en identidad

Política asociada a uno o más directores de IAM que define sus permisos en el Nube de AWS entorno.

aplicación inactiva

Aplicación que utiliza un promedio de CPU y memoria de entre 5 y 20 por ciento durante un periodo de 90 días. En un proyecto de migración, es habitual retirar estas aplicaciones o mantenerlas en las instalaciones.

IIoT

Consulte Internet de las cosas industrial.

infraestructura inmutable

Un modelo que implementa una nueva infraestructura para las cargas de trabajo de producción en lugar de actualizar, parchear o modificar la infraestructura existente. Las infraestructuras inmutables son intrínsecamente más consistentes, fiables y predecibles que las infraestructuras mutables. Para obtener más información, consulte las prácticas recomendadas para implementar con una infraestructura inmutable en Well-Architected Framework AWS .

VPC entrante (de entrada)

En una arquitectura de AWS cuentas múltiples, una VPC que acepta, inspecciona y enruta las conexiones de red desde fuera de una aplicación. La Arquitectura de referencia de seguridad de AWS recomienda configurar su cuenta de red con VPC entrantes, salientes y de inspección para proteger la interfaz bidireccional entre su aplicación e Internet en general.

migración gradual

Estrategia de transición en la que se migra la aplicación en partes pequeñas en lugar de realizar una transición única y completa. Por ejemplo, puede trasladar inicialmente solo unos pocos microservicios o usuarios al nuevo sistema. Tras comprobar que todo funciona correctamente, puede trasladar microservicios o usuarios adicionales de forma gradual hasta que pueda retirar su sistema heredado. Esta estrategia reduce los riesgos asociados a las grandes migraciones.

Industria 4.0

Un término que Klaus Schwab introdujo en 2016 para referirse a la modernización de los procesos de fabricación mediante avances en la conectividad, los datos en tiempo real, la automatización, el análisis y la inteligencia artificial/aprendizaje automático.

infraestructura

Todos los recursos y activos que se encuentran en el entorno de una aplicación.

infraestructura como código (IaC)

Proceso de aprovisionamiento y administración de la infraestructura de una aplicación mediante un conjunto de archivos de configuración. La IaC se ha diseñado para ayudarlo a centralizar la administración de la infraestructura, estandarizar los recursos y escalar con rapidez a fin de que los entornos nuevos sean repetibles, fiables y consistentes.

Internet de las cosas industrial (IIoT)

El uso de sensores y dispositivos conectados a Internet en los sectores industriales, como el productivo, el eléctrico, el automotriz, el sanitario, el de las ciencias de la vida y el de la agricultura. Para obtener más información, consulte Creación de una estrategia de transformación digital del Internet de las cosas industrial (IIoT).

VPC de inspección

En una arquitectura de AWS cuentas múltiples, una VPC centralizada que gestiona las inspecciones del tráfico de red entre las VPC (iguales o Regiones de AWS diferentes), Internet y las redes locales. La Arquitectura de referencia de seguridad de AWS recomienda configurar su cuenta de red con VPC entrantes, salientes y de inspección para proteger la interfaz bidireccional entre su aplicación e Internet en general.

Internet de las cosas (IoT)

Red de objetos físicos conectados con sensores o procesadores integrados que se comunican con otros dispositivos y sistemas a través de Internet o de una red de comunicación local. Para obtener más información, consulte ¿Qué es IoT?.

interpretabilidad

Característica de un modelo de machine learning que describe el grado en que un ser humano puede entender cómo las predicciones del modelo dependen de sus entradas. Para más información, consulte Interpretabilidad del modelo de machine learning con AWS.

IoT

Consulte Internet de las cosas.

biblioteca de información de TI (ITIL)

Conjunto de prácticas recomendadas para ofrecer servicios de TI y alinearlos con los requisitos empresariales. La ITIL proporciona la base para la ITSM.

administración de servicios de TI (ITSM)

Actividades asociadas con el diseño, la implementación, la administración y el soporte de los servicios de TI para una organización. Para obtener información sobre la integración de las operaciones en la nube con las herramientas de ITSM, consulte la Guía de integración de operaciones.

ITIL

Consulte la biblioteca de información de TI.

ITSM

Consulte Administración de servicios de TI.

L

control de acceso basado en etiquetas (LBAC)

Una implementación del control de acceso obligatorio (MAC) en la que a los usuarios y a los propios datos se les asigna explícitamente un valor de etiqueta de seguridad. La intersección entre la etiqueta de seguridad del usuario y la etiqueta de seguridad de los datos determina qué filas y columnas puede ver el usuario.

zona de aterrizaje

Una landing zone es un AWS entorno multicuenta bien diseñado, escalable y seguro. Este es un punto de partida desde el cual las empresas pueden lanzar e implementar rápidamente cargas de trabajo y aplicaciones con confianza en su entorno de seguridad e infraestructura. Para obtener más información sobre las zonas de aterrizaje, consulte Configuración de un entorno de AWS seguro y escalable con varias cuentas.

migración grande

Migración de 300 servidores o más.

LBAC

Consulte control de acceso basado en etiquetas.

privilegio mínimo

La práctica recomendada de seguridad que consiste en conceder los permisos mínimos necesarios para realizar una tarea. Para obtener más información, consulte Aplicar permisos de privilegio mínimo en la documentación de IAM.

migrar mediante lift-and-shift

Ver 7 Rs.

sistema little-endian

Un sistema que almacena primero el byte menos significativo. Véase también endianness.

entornos inferiores

Véase entorno.

M

machine learning (ML)

Un tipo de inteligencia artificial que utiliza algoritmos y técnicas para el reconocimiento y el aprendizaje de patrones. El ML analiza y aprende de los datos registrados, como los datos del Internet de las cosas (IoT), para generar un modelo estadístico basado en patrones. Para más información, consulte Machine learning.

rama principal

Ver sucursal.

malware

Software diseñado para comprometer la seguridad o la privacidad de la computadora. El malware puede interrumpir los sistemas informáticos, filtrar información confidencial u obtener acceso no autorizado. Algunos ejemplos de malware son los virus, los gusanos, el ransomware, los troyanos, el spyware y los keyloggers.

servicios gestionados

Servicios de AWS para los que AWS opera la capa de infraestructura, el sistema operativo y las plataformas, y usted accede a los puntos finales para almacenar y recuperar datos. Amazon Simple Storage Service (Amazon S3) y Amazon DynamoDB son ejemplos de servicios gestionados. También se conocen como servicios abstractos.

sistema de ejecución de fabricación (MES)

Un sistema de software para rastrear, monitorear, documentar y controlar los procesos de producción que convierten las materias primas en productos terminados en el taller.

MAP

Consulte Migration Acceleration Program.

mecanismo

Un proceso completo en el que se crea una herramienta, se impulsa su adopción y, a continuación, se inspeccionan los resultados para realizar ajustes. Un mecanismo es un ciclo que se refuerza y mejora a sí mismo a medida que funciona. Para obtener más información, consulte Creación de mecanismos en el AWS Well-Architected Framework.

cuenta de miembro

Todas las Cuentas de AWS demás cuentas, excepto la de administración, que forman parte de una organización. AWS Organizations Una cuenta no puede pertenecer a más de una organización a la vez.

MES

Consulte el sistema de ejecución de la fabricación.

Transporte telemétrico de Message Queue Queue (MQTT)

Un protocolo de comunicación ligero machine-to-machine (M2M), basado en el patrón de publicación/suscripción, para dispositivos de IoT con recursos limitados.

microservicio

Un servicio pequeño e independiente que se comunica a través de API bien definidas y que, por lo general, es propiedad de equipos pequeños e independientes. Por ejemplo, un sistema de seguros puede incluir microservicios que se adapten a las capacidades empresariales, como las de ventas o marketing, o a subdominios, como las de compras, reclamaciones o análisis. Los beneficios de los microservicios incluyen la agilidad, la escalabilidad flexible, la facilidad de implementación, el código reutilizable y la resiliencia. Para obtener más información, consulte Integrar microservicios mediante servicios sin servidor. AWS

arquitectura de microservicios

Un enfoque para crear una aplicación con componentes independientes que ejecutan cada proceso de la aplicación como un microservicio. Estos microservicios se comunican a través de

una interfaz bien definida mediante API ligeras. Cada microservicio de esta arquitectura se puede actualizar, implementar y escalar para satisfacer la demanda de funciones específicas de una aplicación. Para obtener más información, consulte Implementación de microservicios en. AWS

Programa de aceleración de la migración (MAP)

Un AWS programa que proporciona soporte de consultoría, formación y servicios para ayudar a las organizaciones a crear una base operativa sólida para migrar a la nube y para ayudar a compensar el costo inicial de las migraciones. El MAP incluye una metodología de migración para ejecutar las migraciones antiguas de forma metódica y un conjunto de herramientas para automatizar y acelerar los escenarios de migración más comunes.

migración a escala

Proceso de transferencia de la mayoría de la cartera de aplicaciones a la nube en oleadas, con más aplicaciones desplazadas a un ritmo más rápido en cada oleada. En esta fase, se utilizan las prácticas recomendadas y las lecciones aprendidas en las fases anteriores para implementar una fábrica de migración de equipos, herramientas y procesos con el fin de agilizar la migración de las cargas de trabajo mediante la automatización y la entrega ágil. Esta es la tercera fase de la estrategia de migración de AWS.

fábrica de migración

Equipos multifuncionales que agilizan la migración de las cargas de trabajo mediante enfoques automatizados y ágiles. Los equipos de las fábricas de migración suelen incluir a analistas y propietarios de operaciones, empresas, ingenieros de migración, desarrolladores y DevOps profesionales que trabajan a pasos agigantados. Entre el 20 y el 50 por ciento de la cartera de aplicaciones empresariales se compone de patrones repetidos que pueden optimizarse mediante un enfoque de fábrica. Para obtener más información, consulte la discusión sobre las fábricas de migración y la Guía de fábricas de migración a la nube en este contenido.

metadatos de migración

Información sobre la aplicación y el servidor que se necesita para completar la migración. Cada patrón de migración requiere un conjunto diferente de metadatos de migración. Algunos ejemplos de metadatos de migración son la subred de destino, el grupo de seguridad y AWS la cuenta.

patrón de migración

Tarea de migración repetible que detalla la estrategia de migración, el destino de la migración y la aplicación o el servicio de migración utilizados. Ejemplo: rehospede la migración a Amazon EC2 AWS con Application Migration Service.

Migration Portfolio Assessment (MPA)

Una herramienta en línea que proporciona información para validar el modelo de negocio para la migración a la nube. AWS La MPA ofrece una evaluación detallada de la cartera (adecuación del tamaño de los servidores, precios, comparaciones del costo total de propiedad, análisis de los costos de migración), así como una planificación de la migración (análisis y recopilación de datos de aplicaciones, agrupación de aplicaciones, priorización de la migración y planificación de oleadas). La herramienta MPA (requiere iniciar sesión) está disponible de forma gratuita para todos los AWS consultores y consultores asociados de APN.

Evaluación de la preparación para la migración (MRA)

Proceso que consiste en obtener información sobre el estado de preparación de una organización para la nube, identificar sus puntos fuertes y débiles y elaborar un plan de acción para cerrar las brechas identificadas mediante el AWS CAF. Para obtener más información, consulte la Guía de preparación para la migración. La MRA es la primera fase de la estrategia de migración de AWS.

estrategia de migración

El enfoque utilizado para migrar una carga de trabajo a la AWS nube. Para obtener más información, consulte la entrada de las 7 R de este glosario y consulte Movilice a su organización para acelerar las migraciones a gran escala.

ML

Consulte el aprendizaje automático.

modernización

Transformar una aplicación obsoleta (antigua o monolítica) y su infraestructura en un sistema ágil, elástico y de alta disponibilidad en la nube para reducir los gastos, aumentar la eficiencia y aprovechar las innovaciones. Para obtener más información, consulte Estrategia para modernizar las aplicaciones en el Nube de AWS.

evaluación de la preparación para la modernización

Evaluación que ayuda a determinar la preparación para la modernización de las aplicaciones de una organización; identifica los beneficios, los riesgos y las dependencias; y determina qué tan bien la organización puede soportar el estado futuro de esas aplicaciones. El resultado de la evaluación es un esquema de la arquitectura objetivo, una hoja de ruta que detalla las fases de desarrollo y los hitos del proceso de modernización y un plan de acción para abordar las brechas identificadas. Para obtener más información, consulte Evaluación de la preparación para la modernización de las aplicaciones en la nube de AWS.

aplicaciones monolíticas (monolitos)

Aplicaciones que se ejecutan como un único servicio con procesos estrechamente acoplados. Las aplicaciones monolíticas presentan varios inconvenientes. Si una característica de la aplicación experimenta un aumento en la demanda, se debe escalar toda la arquitectura. Agregar o mejorar las características de una aplicación monolítica también se vuelve más complejo a medida que crece la base de código. Para solucionar problemas con la aplicación, puede utilizar una arquitectura de microservicios. Para obtener más información, consulte Descomposición de monolitos en microservicios.

MAPA

Consulte la evaluación de la cartera de migración.

MQTT

Consulte Message Queue Queue Telemetría y Transporte.

clasificación multiclase

Un proceso que ayuda a generar predicciones para varias clases (predice uno de más de dos resultados). Por ejemplo, un modelo de ML podría preguntar "¿Este producto es un libro, un automóvil o un teléfono?" o "¿Qué categoría de productos es más interesante para este cliente?".

infraestructura mutable

Un modelo que actualiza y modifica la infraestructura existente para las cargas de trabajo de producción. Para mejorar la coherencia, la fiabilidad y la previsibilidad, el AWS Well-Architected Framework recomienda el uso de una infraestructura inmutable como práctica recomendada.

O

OAC

Consulte el control de acceso de origen.

OAI

Consulte la identidad de acceso de origen.

OCM

Consulte gestión del cambio organizacional.

migración fuera de línea

Método de migración en el que la carga de trabajo de origen se elimina durante el proceso de migración. Este método implica un tiempo de inactividad prolongado y, por lo general, se utiliza para cargas de trabajo pequeñas y no críticas.

OI

Consulte integración de operaciones.

OLA

Véase el acuerdo a nivel operativo.

migración en línea

Método de migración en el que la carga de trabajo de origen se copia al sistema de destino sin que se desconecte. Las aplicaciones que están conectadas a la carga de trabajo pueden seguir funcionando durante la migración. Este método implica un tiempo de inactividad nulo o mínimo y, por lo general, se utiliza para cargas de trabajo de producción críticas.

OPC-UA

Consulte Open Process Communications: arquitectura unificada.

Comunicaciones de proceso abierto: arquitectura unificada (OPC-UA)

Un protocolo de comunicación machine-to-machine (M2M) para la automatización industrial. El OPC-UA proporciona un estándar de interoperabilidad con esquemas de cifrado, autenticación y autorización de datos.

acuerdo de nivel operativo (OLA)

Acuerdo que aclara lo que los grupos de TI operativos se comprometen a ofrecerse entre sí, para respaldar un acuerdo de nivel de servicio (SLA).

revisión de la preparación operativa (ORR)

Una lista de preguntas y las mejores prácticas asociadas que le ayudan a comprender, evaluar, prevenir o reducir el alcance de los incidentes y posibles fallos. Para obtener más información, consulte Operational Readiness Reviews (ORR) en AWS Well-Architected Framework.

tecnología operativa (OT)

Sistemas de hardware y software que funcionan con el entorno físico para controlar las operaciones, los equipos y la infraestructura industriales. En la industria manufacturera, la

integración de los sistemas de TO y tecnología de la información (TI) es un enfoque clave para las transformaciones de la industria 4.0.

integración de operaciones (OI)

Proceso de modernización de las operaciones en la nube, que implica la planificación de la preparación, la automatización y la integración. Para obtener más información, consulte la Guía de integración de las operaciones.

registro de seguimiento organizativo

Un registro creado por el AWS CloudTrail que se registran todos los eventos para todos Cuentas de AWS los miembros de una organización AWS Organizations. Este registro de seguimiento se crea en cada Cuenta de AWS que forma parte de la organización y realiza un seguimiento de la actividad en cada cuenta. Para obtener más información, consulte Crear un registro para una organización en la CloudTrail documentación.

administración del cambio organizacional (OCM)

Marco para administrar las transformaciones empresariales importantes y disruptivas desde la perspectiva de las personas, la cultura y el liderazgo. La OCM ayuda a las empresas a prepararse para nuevos sistemas y estrategias y a realizar la transición a ellos, al acelerar la adopción de cambios, abordar los problemas de transición e impulsar cambios culturales y organizacionales. En la estrategia de AWS migración, este marco se denomina aceleración del personal, debido a la velocidad de cambio que requieren los proyectos de adopción de la nube. Para obtener más información, consulte la Guía de OCM.

control de acceso de origen (OAC)

En CloudFront, una opción mejorada para restringir el acceso y proteger el contenido del Amazon Simple Storage Service (Amazon S3). El OAC admite todos los buckets de S3 Regiones de AWS, el cifrado del lado del servidor AWS KMS (SSE-KMS) y las solicitudes dinámicas `PUT` y `DELETE` dirigidas al bucket de S3.

identidad de acceso de origen (OAI)

En CloudFront, una opción para restringir el acceso y proteger el contenido de Amazon S3. Cuando utiliza OAI, CloudFront crea un principal con el que Amazon S3 puede autenticarse. Los directores autenticados solo pueden acceder al contenido de un bucket de S3 a través de una distribución específica. CloudFront Consulte también el OAC, que proporciona un control de acceso más detallado y mejorado.

O

Consulte la revisión de la preparación operativa.

NO

Consulte tecnología operativa.

VPC saliente (de salida)

En una arquitectura de AWS cuentas múltiples, una VPC que gestiona las conexiones de red que se inician desde una aplicación. La Arquitectura de referencia de seguridad de AWS recomienda configurar su cuenta de red con VPC entrantes, salientes y de inspección para proteger la interfaz bidireccional entre su aplicación e Internet en general.

P

límite de permisos

Una política de administración de IAM que se adjunta a las entidades principales de IAM para establecer los permisos máximos que puede tener el usuario o el rol. Para obtener más información, consulte Límites de permisos en la documentación de IAM.

información de identificación personal (PII)

Información que, vista directamente o combinada con otros datos relacionados, puede utilizarse para deducir de manera razonable la identidad de una persona. Algunos ejemplos de información de identificación personal son los nombres, las direcciones y la información de contacto.

PII

Consulte la información de identificación personal.

manual de estrategias

Conjunto de pasos predefinidos que capturan el trabajo asociado a las migraciones, como la entrega de las funciones de operaciones principales en la nube. Un manual puede adoptar la forma de scripts, manuales de procedimientos automatizados o resúmenes de los procesos o pasos necesarios para operar un entorno modernizado.

PLC

Consulte controlador lógico programable.

PLM

Consulte la gestión del ciclo de vida del producto.

política

Un objeto que puede definir los permisos (consulte la política basada en la identidad), especifique las condiciones de acceso (consulte la política basada en los recursos) o defina los permisos máximos para todas las cuentas de una organización AWS Organizations (consulte la política de control de servicios).

persistencia políglota

Elegir de forma independiente la tecnología de almacenamiento de datos de un microservicio en función de los patrones de acceso a los datos y otros requisitos. Si sus microservicios tienen la misma tecnología de almacenamiento de datos, pueden enfrentarse a desafíos de implementación o experimentar un rendimiento deficiente. Los microservicios se implementan más fácilmente y logran un mejor rendimiento y escalabilidad si utilizan el almacén de datos que mejor se adapte a sus necesidades. Para obtener más información, consulte Habilitación de la persistencia de datos en los microservicios.

evaluación de cartera

Proceso de detección, análisis y priorización de la cartera de aplicaciones para planificar la migración. Para obtener más información, consulte la Evaluación de la preparación para la migración.

predicate

Una condición de consulta que devuelve `true` o`false`, por lo general, se encuentra en una cláusula. `WHERE`

pulsar un predicado

Técnica de optimización de consultas de bases de datos que filtra los datos de la consulta antes de transferirlos. Esto reduce la cantidad de datos que se deben recuperar y procesar de la base de datos relacional y mejora el rendimiento de las consultas.

control preventivo

Un control de seguridad diseñado para evitar que ocurra un evento. Estos controles son la primera línea de defensa para evitar el acceso no autorizado o los cambios no deseados en la red. Para obtener más información, consulte Controles preventivos en Implementación de controles de seguridad en AWS.

entidad principal

Una entidad AWS que puede realizar acciones y acceder a los recursos. Esta entidad suele ser un usuario raíz para un Cuenta de AWS rol de IAM o un usuario. Para obtener más información, consulte Entidad principal en Términos y conceptos de roles en la documentación de IAM.

Privacidad desde el diseño

Un enfoque de ingeniería de sistemas que tiene en cuenta la privacidad durante todo el proceso de ingeniería.

zonas alojadas privadas

Contenedor que aloja información acerca de cómo desea que responda Amazon Route 53 a las consultas de DNS de un dominio y sus subdominios en una o varias VPC. Para obtener más información, consulte Uso de zonas alojadas privadas en la documentación de Route 53.

control proactivo

Un control de seguridad diseñado para evitar el despliegue de recursos no conformes. Estos controles escanean los recursos antes de aprovisionarlos. Si el recurso no cumple con el control, significa que no está aprovisionado. Para obtener más información, consulte la guía de referencia de controles en la AWS Control Tower documentación y consulte Controles proactivos en Implementación de controles de seguridad en AWS.

gestión del ciclo de vida del producto (PLM)

La gestión de los datos y los procesos de un producto a lo largo de todo su ciclo de vida, desde el diseño, el desarrollo y el lanzamiento, pasando por el crecimiento y la madurez, hasta el rechazo y la retirada.

entorno de producción

Consulte el entorno.

controlador lógico programable (PLC)

En la fabricación, una computadora adaptable y altamente confiable que monitorea las máquinas y automatiza los procesos de fabricación.

seudonimización

El proceso de reemplazar los identificadores personales de un conjunto de datos por valores de marcadores de posición. La seudonimización puede ayudar a proteger la privacidad personal. Los datos seudonimizados siguen considerándose datos personales.

publicar/suscribirse (pub/sub)

Un patrón que permite las comunicaciones asíncronas entre microservicios para mejorar la escalabilidad y la capacidad de respuesta. Por ejemplo, en un MES basado en microservicios, un microservicio puede publicar mensajes de eventos en un canal al que se puedan suscribir otros microservicios. El sistema puede añadir nuevos microservicios sin cambiar el servicio de publicación.

Q

plan de consulta

Serie de pasos, como instrucciones, que se utilizan para acceder a los datos de un sistema de base de datos relacional SQL.

regresión del plan de consulta

El optimizador de servicios de la base de datos elige un plan menos óptimo que antes de un cambio determinado en el entorno de la base de datos. Los cambios en estadísticas, restricciones, configuración del entorno, enlaces de parámetros de consultas y actualizaciones del motor de base de datos PostgreSQL pueden provocar una regresión del plan.

R

Matriz RACI

Véase responsable, responsable, consultado, informado (RACI).

ransomware

Software malicioso que se ha diseñado para bloquear el acceso a un sistema informático o a los datos hasta que se efectúe un pago.

Matriz RASCI

Véase responsable, responsable, consultado, informado (RACI).

RCAC

Consulte control de acceso por filas y columnas.

read replica

Una copia de una base de datos que se utiliza con fines de solo lectura. Puede enrutar las consultas a la réplica de lectura para reducir la carga en la base de datos principal.

rediseñar

Ver 7 Rs.

objetivo de punto de recuperación (RPO)

La cantidad de tiempo máximo aceptable desde el último punto de recuperación de datos. Esto determina qué se considera una pérdida de datos aceptable entre el último punto de recuperación y la interrupción del servicio.

objetivo de tiempo de recuperación (RTO)

La demora máxima aceptable entre la interrupción del servicio y el restablecimiento del servicio.

refactorizar

Ver 7 Rs.

Región

Una colección de AWS recursos en un área geográfica. Cada uno Región de AWS está aislado y es independiente de los demás para proporcionar tolerancia a las fallas, estabilidad y resiliencia. Para obtener más información, consulte Regiones de AWS Especificar qué cuenta puede usar.

regresión

Una técnica de ML que predice un valor numérico. Por ejemplo, para resolver el problema de "¿A qué precio se venderá esta casa?", un modelo de ML podría utilizar un modelo de regresión lineal para predecir el precio de venta de una vivienda en función de datos conocidos sobre ella (por ejemplo, los metros cuadrados).

volver a alojar

Consulte 7 Rs.

versión

En un proceso de implementación, el acto de promover cambios en un entorno de producción.

trasladarse

Ver 7 Rs.

redefinir la plataforma

Ver 7 Rs.

recompra

Ver 7 Rs.

resilencia

La capacidad de una aplicación para resistir las interrupciones o recuperarse de ellas. La alta disponibilidad y la recuperación ante desastres son consideraciones comunes a la hora de planificar la resiliencia en el. Nube de AWS Para obtener más información, consulte Nube de AWS Resiliencia.

política basada en recursos

Una política asociada a un recurso, como un bucket de Amazon S3, un punto de conexión o una clave de cifrado. Este tipo de política especifica a qué entidades principales se les permite el acceso, las acciones compatibles y cualquier otra condición que deba cumplirse.

matriz responsable, confiable, consultada e informada (RACI)

Una matriz que define las funciones y responsabilidades de todas las partes involucradas en las actividades de migración y las operaciones de la nube. El nombre de la matriz se deriva de los tipos de responsabilidad definidos en la matriz: responsable (R), contable (A), consultado (C) e informado (I). El tipo de soporte (S) es opcional. Si incluye el soporte, la matriz se denomina matriz RASCI y, si la excluye, se denomina matriz RACI.

control receptivo

Un control de seguridad que se ha diseñado para corregir los eventos adversos o las desviaciones con respecto a su base de seguridad. Para obtener más información, consulte Controles receptivos en Implementación de controles de seguridad en AWS.

retain

Consulte 7 Rs.

jubilarse

Ver 7 Rs.

rotación

Proceso de actualizar periódicamente un secreto para dificultar el acceso de un atacante a las credenciales.

control de acceso por filas y columnas (RCAC)

El uso de expresiones SQL básicas y flexibles que tienen reglas de acceso definidas. El RCAC consta de permisos de fila y máscaras de columnas.

RPO

Consulte el objetivo del punto de recuperación.

RTO

Consulte el objetivo de tiempo de recuperación.

manual de procedimientos

Conjunto de procedimientos manuales o automatizados necesarios para realizar una tarea específica. Por lo general, se diseñan para agilizar las operaciones o los procedimientos repetitivos con altas tasas de error.

S

SAML 2.0

Un estándar abierto que utilizan muchos proveedores de identidad (IdPs). Esta función permite el inicio de sesión único (SSO) federado, de modo que los usuarios pueden iniciar sesión AWS Management Console o llamar a las operaciones de la AWS API sin tener que crear un usuario en IAM para todos los miembros de la organización. Para obtener más información sobre la federación basada en SAML 2.0, consulte Acerca de la federación basada en SAML 2.0 en la documentación de IAM.

SCADA

Consulte el control de supervisión y la adquisición de datos.

SCP

Consulte la política de control de servicios.

secreta

Información confidencial o restringida, como una contraseña o credenciales de usuario, que almacene de forma cifrada. AWS Secrets Manager Se compone del valor secreto y sus

metadatos. El valor secreto puede ser binario, una sola cadena o varias cadenas. Para obtener más información, consulte la documentación de Secret in the Secrets Manager.

control de seguridad

Barrera de protección técnica o administrativa que impide, detecta o reduce la capacidad de un agente de amenazas para aprovechar una vulnerabilidad de seguridad. Existen cuatro tipos principales de controles de seguridad: preventivos, de detección, con capacidad de respuesta y proactivos.

refuerzo de la seguridad

Proceso de reducir la superficie expuesta a ataques para hacerla más resistente a los ataques. Esto puede incluir acciones, como la eliminación de los recursos que ya no se necesitan, la implementación de prácticas recomendadas de seguridad consistente en conceder privilegios mínimos o la desactivación de características innecesarias en los archivos de configuración.

sistema de información sobre seguridad y administración de eventos (SIEM)

Herramientas y servicios que combinan sistemas de administración de información sobre seguridad (SIM) y de administración de eventos de seguridad (SEM). Un sistema de SIEM recopila, monitorea y analiza los datos de servidores, redes, dispositivos y otras fuentes para detectar amenazas y brechas de seguridad y generar alertas.

automatización de la respuesta de seguridad

Una acción predefinida y programada que está diseñada para responder automáticamente a un evento de seguridad o remediarlo. Estas automatizaciones sirven como controles de seguridad detectables o adaptables que le ayudan a implementar las mejores prácticas AWS de seguridad. Algunos ejemplos de acciones de respuesta automatizadas incluyen la modificación de un grupo de seguridad de VPC, la aplicación de parches a una instancia de Amazon EC2 o la rotación de credenciales.

cifrado del servidor

Cifrado de los datos en su destino, por parte de quien Servicio de AWS los recibe.

política de control de servicio (SCP)

Una política que proporciona un control centralizado de los permisos de todas las cuentas de una organización en AWS Organizations. Las SCP definen barreras de protección o establecen límites a las acciones que un administrador puede delegar en los usuarios o roles. Puede utilizar las SCP como listas de permitidos o rechazados, para especificar qué servicios o acciones se

encuentra permitidos o prohibidos. Para obtener más información, consulte las políticas de control de servicios en la AWS Organizations documentación.

punto de enlace de servicio

La URL del punto de entrada de un Servicio de AWS. Para conectarse mediante programación a un servicio de destino, puede utilizar un punto de conexión. Para obtener más información, consulte Puntos de conexión de Servicio de AWS en Referencia general de AWS.

acuerdo de nivel de servicio (SLA)

Acuerdo que aclara lo que un equipo de TI se compromete a ofrecer a los clientes, como el tiempo de actividad y el rendimiento del servicio.

indicador de nivel de servicio (SLI)

Medición de un aspecto del rendimiento de un servicio, como la tasa de errores, la disponibilidad o el rendimiento.

objetivo de nivel de servicio (SLO)

Una métrica objetivo que representa el estado de un servicio, medido mediante un indicador de nivel de servicio.

modelo de responsabilidad compartida

Un modelo que describe la responsabilidad que compartes con respecto a la seguridad y AWS el cumplimiento de la nube. AWS es responsable de la seguridad de la nube, mientras que usted es responsable de la seguridad en la nube. Para obtener más información, consulte el Modelo de responsabilidad compartida.

SIEM

Consulte la información de seguridad y el sistema de gestión de eventos.

punto único de fallo (SPOF)

Una falla en un único componente crítico de una aplicación que puede interrumpir el sistema.

SLA

Consulte el acuerdo de nivel de servicio.

SLI

Consulte el indicador de nivel de servicio.

ASÍ QUE

Consulte el objetivo de nivel de servicio.

split-and-seed modelo

Un patrón para escalar y acelerar los proyectos de modernización. A medida que se definen las nuevas funciones y los lanzamientos de los productos, el equipo principal se divide para crear nuevos equipos de productos. Esto ayuda a ampliar las capacidades y los servicios de su organización, mejora la productividad de los desarrolladores y apoya la innovación rápida. Para obtener más información, consulte Enfoque gradual para modernizar las aplicaciones en el. Nube de AWS

SPOT

Consulte el punto único de falla.

esquema en forma de estrella

Estructura organizativa de una base de datos que utiliza una tabla de datos grande para almacenar datos transaccionales o medidos y una o más tablas dimensionales más pequeñas para almacenar los atributos de los datos. Esta estructura está diseñada para usarse en un almacén de datos o con fines de inteligencia empresarial.

patrón de higo estrangulador

Un enfoque para modernizar los sistemas monolíticos mediante la reescritura y el reemplazo gradual de las funciones del sistema hasta que se pueda desmantelar el sistema heredado. Este patrón utiliza la analogía de una higuera que crece hasta convertirse en un árbol estable y, finalmente, se apodera y reemplaza a su host. El patrón fue presentado por Martin Fowler como una forma de gestionar el riesgo al reescribir sistemas monolíticos. Para ver un ejemplo con la aplicación de este patrón, consulte Modernización gradual de los servicios web antiguos de Microsoft ASP.NET (ASMX) mediante contenedores y Amazon API Gateway.

subred

Un intervalo de direcciones IP en la VPC. Una subred debe residir en una sola zona de disponibilidad.

supervisión, control y adquisición de datos (SCADA)

En la industria manufacturera, un sistema que utiliza hardware y software para monitorear los activos físicos y las operaciones de producción.

cifrado simétrico

Un algoritmo de cifrado que utiliza la misma clave para cifrar y descifrar los datos.

pruebas sintéticas

Probar un sistema de manera que simule las interacciones de los usuarios para detectar posibles problemas o monitorear el rendimiento. Puede usar Amazon CloudWatch Synthetics para crear estas pruebas.

T

etiquetas

Pares clave-valor que actúan como metadatos para organizar los recursos. AWS Las etiquetas pueden ayudarle a administrar, identificar, organizar, buscar y filtrar recursos. Para obtener más información, consulte Etiquetado de los recursos de AWS.

variable de destino

El valor que intenta predecir en el ML supervisado. Esto también se conoce como variable de resultado. Por ejemplo, en un entorno de fabricación, la variable objetivo podría ser un defecto del producto.

lista de tareas

Herramienta que se utiliza para hacer un seguimiento del progreso mediante un manual de procedimientos. La lista de tareas contiene una descripción general del manual de procedimientos y una lista de las tareas generales que deben completarse. Para cada tarea general, se incluye la cantidad estimada de tiempo necesario, el propietario y el progreso.

entorno de prueba

Consulte entorno.

entrenamiento

Proporcionar datos de los que pueda aprender su modelo de ML. Los datos de entrenamiento deben contener la respuesta correcta. El algoritmo de aprendizaje encuentra patrones en los datos de entrenamiento que asignan los atributos de los datos de entrada al destino (la respuesta que desea predecir). Genera un modelo de ML que captura estos patrones. Luego, el modelo de ML se puede utilizar para obtener predicciones sobre datos nuevos para los que no se conoce el destino.

puerta de enlace de tránsito

Centro de tránsito de red que puede utilizar para interconectar las VPC y las redes en las instalaciones. Para obtener más información, consulte Qué es una pasarela de tránsito en la AWS Transit Gateway documentación.

flujo de trabajo basado en enlaces troncales

Un enfoque en el que los desarrolladores crean y prueban características de forma local en una rama de característica y, a continuación, combinan esos cambios en la rama principal. Luego, la rama principal se adapta a los entornos de desarrollo, preproducción y producción, de forma secuencial.

acceso de confianza

Otorgar permisos a un servicio que especifique para realizar tareas en su organización AWS Organizations y en sus cuentas en su nombre. El servicio de confianza crea un rol vinculado al servicio en cada cuenta, cuando ese rol es necesario, para realizar las tareas de administración por usted. Para obtener más información, consulte AWS Organizations Utilización con otros AWS servicios en la AWS Organizations documentación.

ajuste

Cambiar aspectos de su proceso de formación a fin de mejorar la precisión del modelo de ML. Por ejemplo, puede entrenar el modelo de ML al generar un conjunto de etiquetas, incorporar etiquetas y, luego, repetir estos pasos varias veces con diferentes ajustes para optimizar el modelo.

equipo de dos pizzas

Un DevOps equipo pequeño al que puedes alimentar con dos pizzas. Un equipo formado por dos integrantes garantiza la mejor oportunidad posible de colaboración en el desarrollo de software.

U

incertidumbre

Un concepto que hace referencia a información imprecisa, incompleta o desconocida que puede socavar la fiabilidad de los modelos predictivos de ML. Hay dos tipos de incertidumbre: la incertidumbre epistémica se debe a datos limitados e incompletos, mientras que la incertidumbre aleatoria se debe al ruido y la aleatoriedad inherentes a los datos. Para más información, consulte la guía Cuantificación de la incertidumbre en los sistemas de aprendizaje profundo.

tareas indiferenciadas

También conocido como tareas arduas, es el trabajo que es necesario para crear y operar una aplicación, pero que no proporciona un valor directo al usuario final ni proporciona una ventaja competitiva. Algunos ejemplos de tareas indiferenciadas son la adquisición, el mantenimiento y la planificación de la capacidad.

entornos superiores

Ver entorno.

V

succión

Una operación de mantenimiento de bases de datos que implica limpiar después de las actualizaciones incrementales para recuperar espacio de almacenamiento y mejorar el rendimiento.

control de versión

Procesos y herramientas que realizan un seguimiento de los cambios, como los cambios en el código fuente de un repositorio.

Emparejamiento de VPC

Conexión entre dos VPC que permite enrutar el tráfico mediante direcciones IP privadas. Para obtener más información, consulte ¿Qué es una interconexión de VPC? en la documentación de Amazon VPC.

vulnerabilidad

Defecto de software o hardware que pone en peligro la seguridad del sistema.

W

caché caliente

Un búfer caché que contiene datos actuales y relevantes a los que se accede con frecuencia. La instancia de base de datos puede leer desde la caché del búfer, lo que es más rápido que leer desde la memoria principal o el disco.

datos templados

Datos a los que el acceso es infrecuente. Al consultar este tipo de datos, normalmente se aceptan consultas moderadamente lentas.

función de ventana

Función SQL que realiza un cálculo en un grupo de filas que se relacionan de alguna manera con el registro actual. Las funciones de ventana son útiles para procesar tareas, como calcular una media móvil o acceder al valor de las filas en función de la posición relativa de la fila actual.

carga de trabajo

Conjunto de recursos y código que ofrece valor comercial, como una aplicación orientada al cliente o un proceso de backend.

flujo de trabajo

Grupos funcionales de un proyecto de migración que son responsables de un conjunto específico de tareas. Cada flujo de trabajo es independiente, pero respalda a los demás flujos de trabajo del proyecto. Por ejemplo, el flujo de trabajo de la cartera es responsable de priorizar las aplicaciones, planificar las oleadas y recopilar los metadatos de migración. El flujo de trabajo de la cartera entrega estos recursos al flujo de trabajo de migración, que luego migra los servidores y las aplicaciones.

GUSANO

Mira, escribe una vez, lee muchas.

WQF

Consulte el marco de calificación de cargas de trabajo de AWS.

escribe una vez, lee muchas (WORM)

Un modelo de almacenamiento que escribe los datos una sola vez y evita que los datos se eliminen o modifiquen. Los usuarios autorizados pueden leer los datos tantas veces como sea necesario, pero no pueden cambiarlos. Esta infraestructura de almacenamiento de datos se considera inmutable.

Z

ataque de día cero

Un ataque, normalmente de malware, que aprovecha una vulnerabilidad de día cero.

vulnerabilidad de día cero

Un defecto o una vulnerabilidad sin mitigación en un sistema de producción. Los agentes de amenazas pueden usar este tipo de vulnerabilidad para atacar el sistema. Los desarrolladores suelen darse cuenta de la vulnerabilidad a raíz del ataque.

aplicación zombi

Aplicación que utiliza un promedio de CPU y memoria menor al 5 por ciento. En un proyecto de migración, es habitual retirar estas aplicaciones.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.