COLOMBIA POTENCIA DE LA VIDA

TIC

Lección 5

Ampliación de las arquitecturas sin servidor con Amazon API Gateway y Organización de microservicios con AWS Step Functions

TALENTO TECH

AZ | PROYECTOS EDUCATIVOS

UTP Universidad Tecnológica de Pereira

Amazon API Gateway

Amazon API Gateway

- Permite crear, publicar, mantener, supervisar y proteger API que funcionan como puntos de entrada a recursos backend para sus aplicaciones
- Gestiona hasta cientos de miles de llamadas API simultáneas
- Puede gestionar cargas de trabajo que se ejecutan en:
  - Amazon EC2
  - Lambda
  - Cualquier aplicación web
  - Aplicaciones de comunicación en tiempo real
- Puede alojar y utilizar varias versiones y fases de API

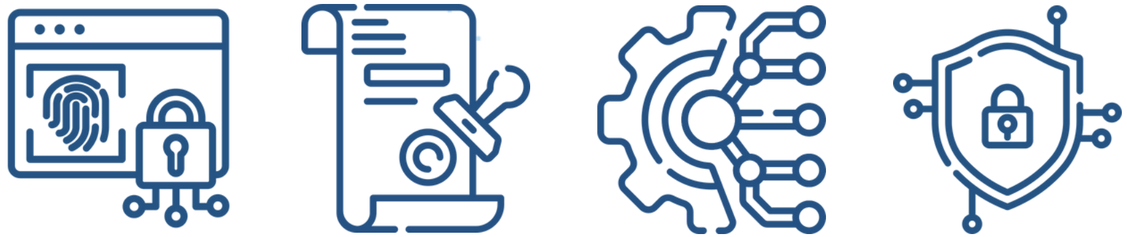
Amazon API Gateway es un servicio completamente administrado que le permite crear, publicar, mantener, supervisar y proteger API a cualquier escala.

Puede utilizarlo para crear API de transferencia de estado representacional (RESTful) y WebSocket que funcionen como un punto de entrada para que las aplicaciones puedan acceder a los recursos de backend. De este modo, las aplicaciones pueden acceder a los datos, la lógica empresarial o la funcionalidad de sus servicios de backend. Estos servicios incluyen aplicaciones que se ejecutan en Amazon EC2, código que se ejecuta en Lambda, cualquier aplicación web o aplicaciones de comunicación en tiempo real.

API Gateway gestiona todas las tareas relacionadas con la aceptación y el procesamiento de hasta cientos de miles de llamadas API simultáneas. Estas llamadas pueden incluir la administración del tráfico, el control de la autorización y el acceso, la supervisión y la administración de versiones de la API. API Gateway no requiere pagos mínimos ni tiene costos iniciales. Solo deberá pagar por las llamadas a la API que reciba y la cantidad de datos transferidos. Con el modelo de precios por niveles de API Gateway, puede reducir sus costos a medida que aumenta el uso de la API.

Puede utilizar API Gateway para alojar varias versiones y fases de sus API.

Seguridad Amazon API Gateway



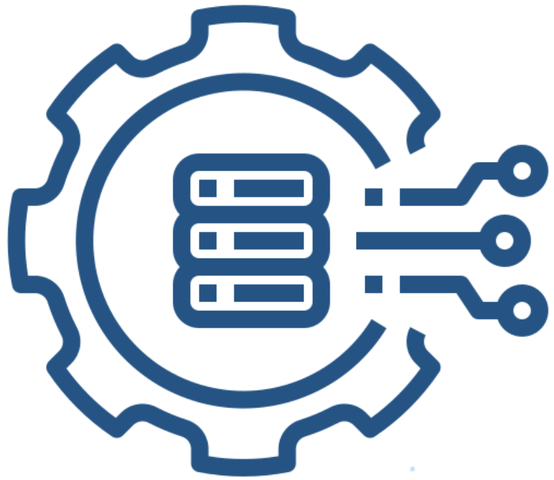
- Necesita autorización
- Aplica políticas de recursos
- Configuración de limitación controlada
- Protección contra ataques de denegación de servicio distribuido (DDoS) e inyección

Cuando publica sus API, se expone a posibles ataques que intenten explotar sus servicios. Con Amazon API Gateway, puede proteger sus API de varias formas.

Con Amazon API Gateway, puede configurar sus métodos API de manera opcional para que necesiten autorización. Cuando configure un método de autorización, puede utilizar los autorizadores Lambda o AWS Signature Version 4 para que sean compatibles con su propia estrategia de autenticación de token portador. AWS Signature Version 4 es el proceso para agregar información de autenticación a las solicitudes de AWS que se envían a través de HTTP.

Por seguridad, la mayoría de las solicitudes a AWS se deben firmar con una clave de acceso, que se compone de un ID de clave de acceso y una clave de acceso secreta. Utilice estas credenciales de AWS para firmar solicitudes a su servicio y autorizar el acceso, al igual que otros servicios de AWS. Puede recuperar credenciales temporales asociadas a un rol en su cuenta de AWS mediante Amazon Cognito. Un autorizador Lambda es una función de Lambda que autoriza el acceso a las API mediante una estrategia de autenticación de token portador como OAuth.

También puede aplicar una política de recursos a una API para restringir el acceso a una VPC de Amazon o a un punto de enlace de VPC específicos. Puede conceder acceso a una VPC de Amazon o un punto de enlace de VPC de una cuenta diferente a la API privada mediante una política de recursos.



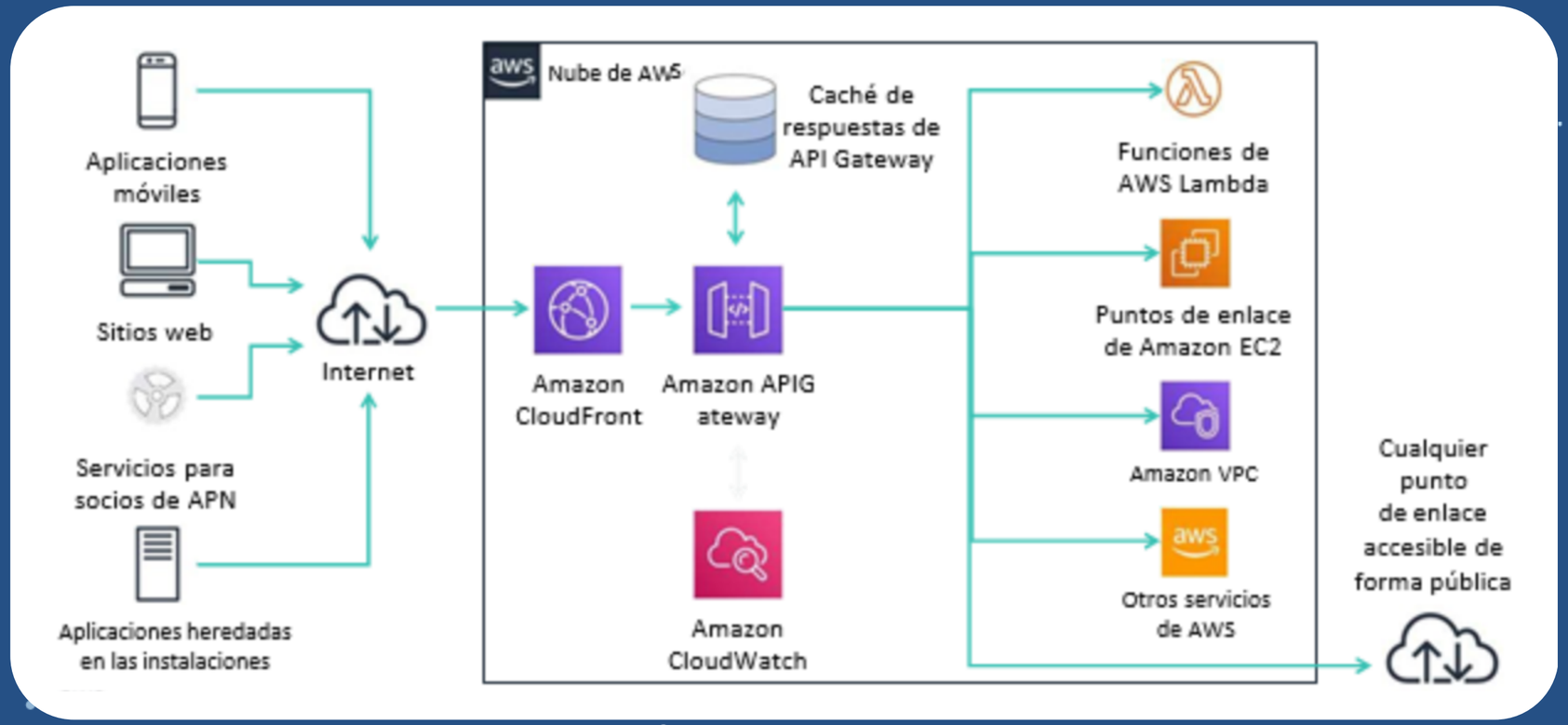
Amazon API Gateway admite la configuración de limitación controlada para cada método o ruta de sus API. Puede establecer un límite de velocidad estándar y otro de ráfaga por segundo para cada método en sus API REST y cada ruta de las API WebSocket.

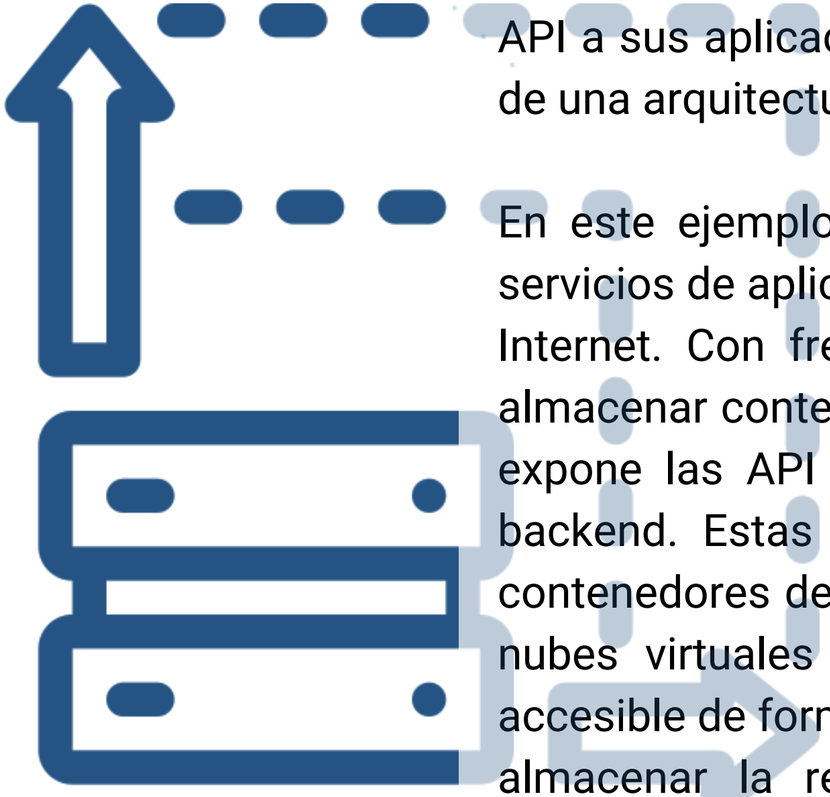
Además, puede utilizar AWS WAF para proteger sus API de API Gateway. AWS WAF es un firewall de aplicaciones web que protege sus aplicaciones web de las vulnerabilidades de seguridad más habituales que pueden afectar la disponibilidad, poner en riesgo la seguridad o consumir demasiados recursos.

Para obtener más información sobre cómo proteger sus API con Amazon API Gateway, consulte la Lección Seguridad y autorización de las preguntas frecuentes de Amazon API Gateway.

Amazon API Gateway:

Ejemplo de una arquitectura común





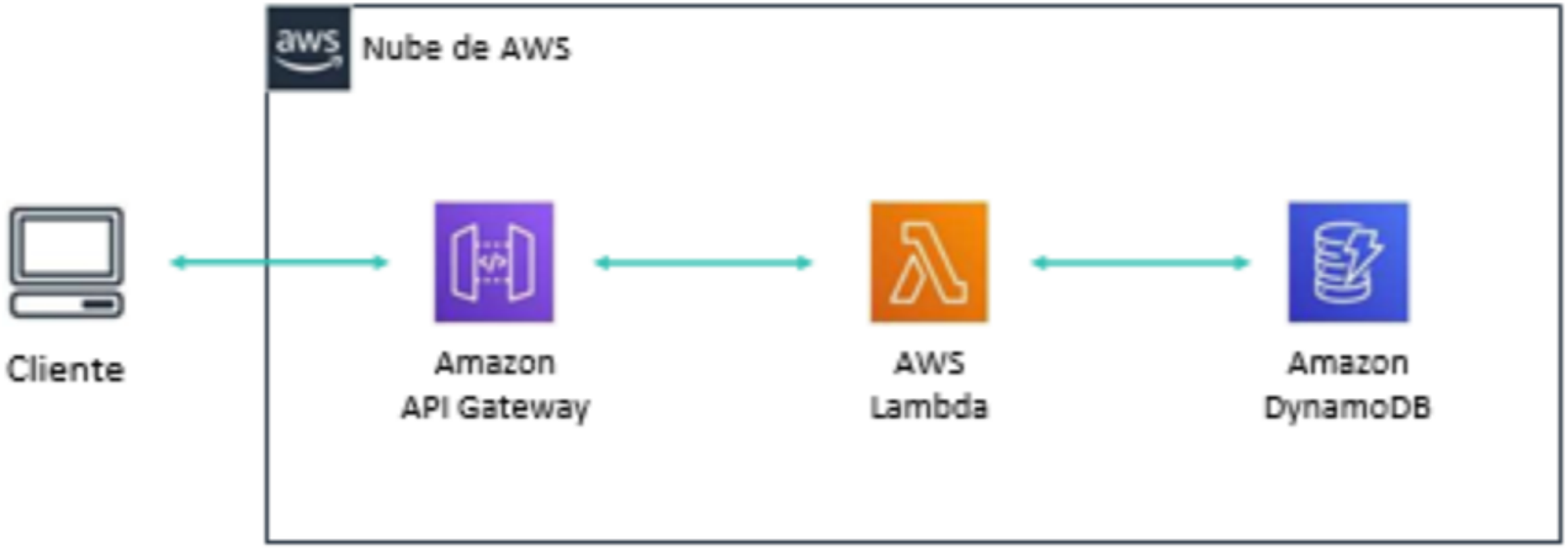
Puede utilizar Amazon API Gateway para suministrar la capa API a sus aplicaciones. A continuación, se muestra un ejemplo de una arquitectura común con Amazon API Gateway.

En este ejemplo, las aplicaciones de clientes frontend y los servicios de aplicación envían tráfico a API Gateway a través de Internet. Con frecuencia, Amazon CloudFront se utiliza para almacenar contenido estático en caché. API Gateway resume y expone las API que pueden llamar a varias aplicaciones de backend. Estas aplicaciones incluyen funciones de Lambda, contenedores de Docker que se ejecutan en instancias de EC2, nubes virtuales privada (VPC) o cualquier punto de enlace accesible de forma pública. Si es necesario, API Gateway puede almacenar la respuesta en caché. Por último, se pueden supervisar todas las llamadas API con Amazon CloudWatch.

Puede utilizar API Gateway con otros servicios administrados de AWS para crear backends sin servidor para sus aplicaciones. Por ejemplo, API Gateway puede enviar solicitudes a funciones de Lambda que ejecutan su código y generan respuestas. Incluso puede crear API que envíen solicitudes de proxy a otros servicios de AWS, como Amazon Simple Storage Service (Amazon S3), sin tener que escribir código. Puede poner en funcionamiento backends a escala de producción más rápidamente porque tiene menos código que escribir, y además delegó la carga operativa a los servicios de AWS.

Para ver un ejemplo de cómo utilizar API Gateway con AWS Lambda, consulte esta publicación del blog de AWS.

Ejemplo: microservicios RESTful

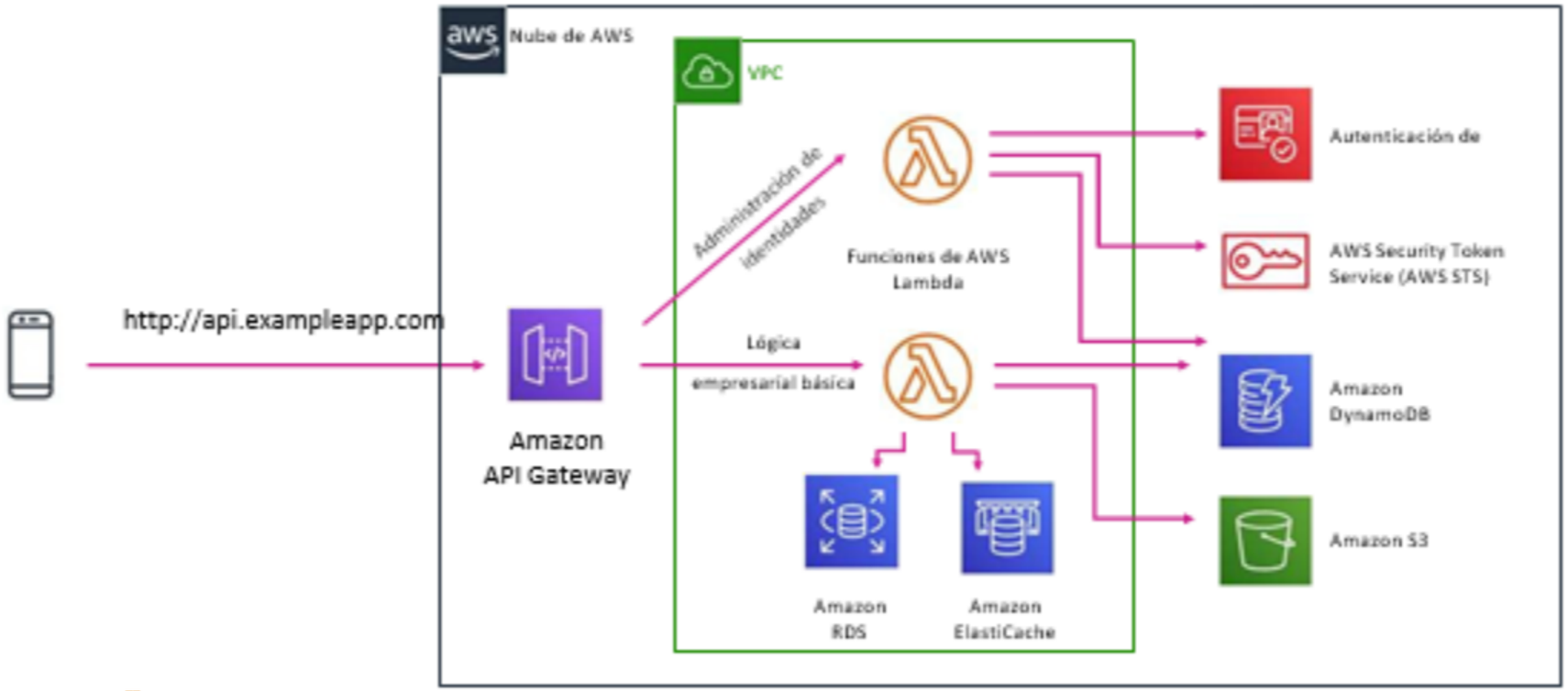


Amazon API Gateway está profundamente integrado con AWS Lambda. Este ejemplo muestra cómo los dos servicios pueden utilizarse juntos en una arquitectura para una aplicación de microservicios RESTful.

En este ejemplo, los clientes pueden utilizar su microservicio realizando llamadas HTTP API. Amazon API Gateway aloja solicitudes RESTful HTTP de sus clientes y las respuestas a estas. En este escenario, API Gateway proporciona autorización integrada, limitación controlada, seguridad, tolerancia a errores, mapeo de solicitud-respuesta y optimización del rendimiento.

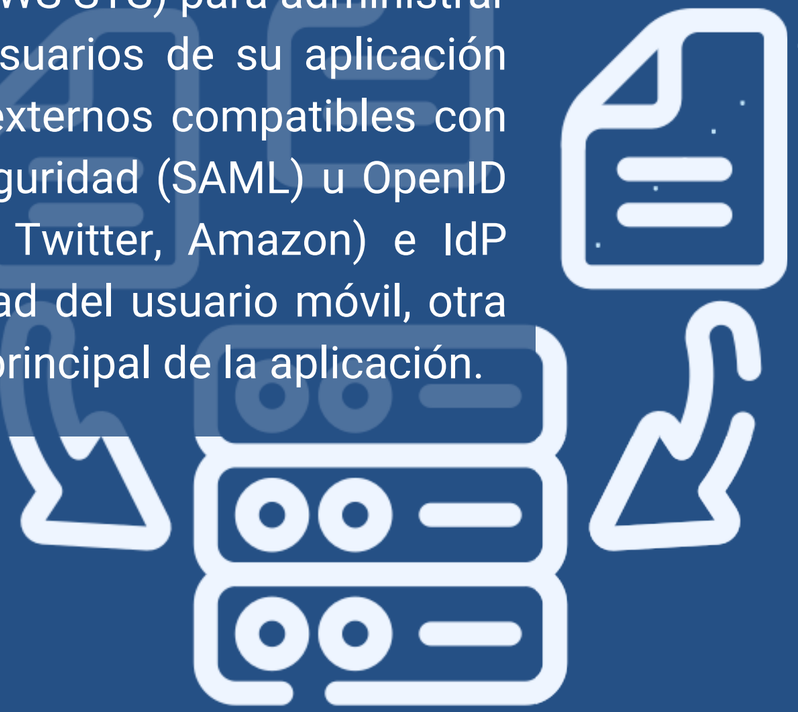
AWS Lambda contiene la lógica empresarial para procesar las llamadas entrantes a la API y utiliza DynamoDB como almacenamiento persistente. Amazon DynamoDB almacena de forma persistente los datos de los microservicios y escala en función de la demanda. Dado que los microservicios están diseñados para hacer una cosa de forma eficiente, se suele incorporar un almacén de datos NoSQL sin esquema.

Ejemplo:
backend móvil sin servidor



Analicemos otro ejemplo de cómo se puede utilizar Amazon API Gateway con Lambda en una arquitectura sin servidor como puerta de enlace al backend para una aplicación móvil. Se espera que las aplicaciones móviles ofrezcan al usuario una experiencia rápida, consistente y rica en funciones, además de que suelen tener una presencia mundial. Además, los patrones de los usuarios móviles son dinámicos, con picos de uso impredecibles. Las aplicaciones móviles requieren un amplio conjunto de servicios móviles que funcionen juntos a la perfección sin sacrificar el control ni la flexibilidad de la infraestructura de backend.

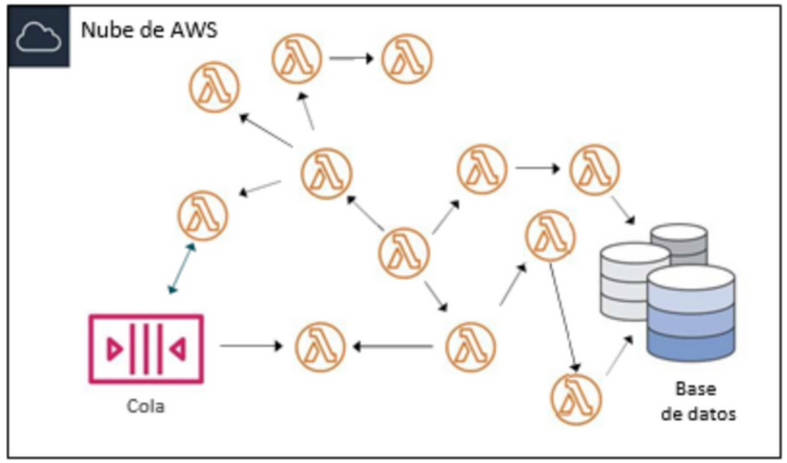
En este ejemplo, una aplicación móvil envía una solicitud a Amazon API Gateway, que reenvía la solicitud a una función de Lambda que llama a Amazon Cognito y AWS Security Token Service (AWS STS) para administrar la identidad. Amazon Cognito autentica a los usuarios de su aplicación móvil mediante proveedores de identidad (IdP) externos compatibles con lenguaje de marcado para confirmaciones de seguridad (SAML) u OpenID Connect, IdP sociales (por ejemplo, Facebook, Twitter, Amazon) e IdP personalizados. Después de confirmar la identidad del usuario móvil, otra función de Lambda ejecuta la lógica empresarial principal de la aplicación.



Entre los puntos clave de esta Lección, se incluyen los siguientes:

- Amazon API Gateway es un servicio completamente administrado que le permite crear, publicar, mantener, supervisar y proteger API a cualquier escala.
- Amazon API Gateway funciona como punto de entrada a los recursos de backend para sus aplicaciones. Resume y expone las API que pueden llamar a varias aplicaciones de backend. Estas aplicaciones incluyen funciones de Lambda, contenedores de Docker que se ejecutan en instancias de EC2, VPC o cualquier punto de enlace accesible de forma pública.
- Amazon API Gateway está profundamente integrado con Lambda.

Desafíos de las aplicaciones de microservicios

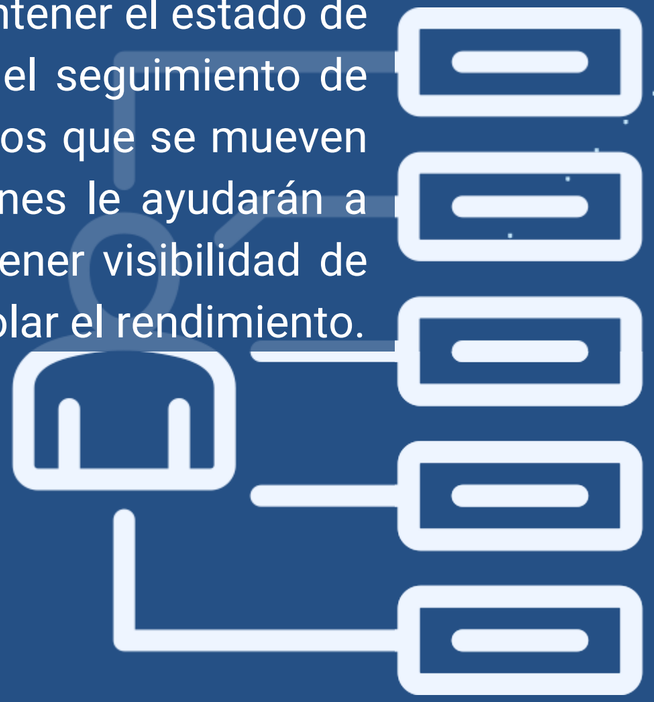


A medida que la aplicación crece en tamaño, el número de componentes aumenta y surgen muchos patrones posibles de qué tareas ejecutar y en qué orden. Por ejemplo, consideremos una aplicación de microservicio que se crea utilizando funciones de Lambda.

Es posible que desee invocar una función de Lambda inmediatamente después de otra función, y solo si la primera función se ejecuta correctamente. Es posible que desee invocar dos funciones en paralelo y, a continuación, enviar los resultados combinados a una tercera función. O puede que desee elegir cuál de las dos funciones se invoca basándose en la salida de otra función.

La invocación de una función puede derivar en un error por varias razones. El código puede plantear una excepción, agotar el tiempo de espera o quedarse sin memoria. El tiempo de ejecución que ejecuta el código puede encontrar un error y detenerse. Cuando se produce un error, es posible que su código se haya ejecutado completamente, parcialmente o no se haya ejecutado. En la mayoría de los casos, el cliente o servicio que invoca la función reintentará la ejecución si encuentra un error, por lo que el código debe tener la capacidad de procesar el mismo evento repetidamente sin que surjan efectos no deseados. Si su función administra recursos o escribe en una base de datos, debe manejar los casos en los que la misma solicitud se realiza varias veces.

Necesita una forma de coordinar los componentes de su aplicación. Esta capa de coordinación se debe poder escalar automáticamente en función de los cambios en las cargas de trabajo y administrar los errores y los tiempos de espera. También debe mantener el estado de su aplicación mientras se está ejecutando, como el seguimiento de los pasos actuales y el almacenamiento de los datos que se mueven entre los pasos del flujo de trabajo. Estas funciones le ayudarán a crear y utilizar sus aplicaciones. También desea tener visibilidad de su aplicación para poder solucionar errores y controlar el rendimiento.



COLOMBIA POTENCIA DE LA VIDA

TIC

AWS Step Functions

- Coordina los microservicios mediante flujos de trabajo visuales
- Permite hacer un recorrido por las funciones de su aplicación
- Desencadena automáticamente y realiza un seguimiento de cada paso
- Proporciona captura y registro de errores simples si falla un paso

AWS Step Functions es un servicio web que facilita la coordinación de componentes de aplicaciones y microservicios distribuidos mediante flujos de trabajo visuales. Step Functions proporciona una manera fiable de coordinar los componentes y procesar las funciones de su aplicación.

Step Functions ofrece una consola gráfica para que pueda visualizar los componentes de su aplicación en varios pasos. Activa cada paso y le da seguimiento de manera automática, y realiza reintentos cuando se producen errores, por lo que su aplicación se ejecuta en orden y según lo previsto. Step Functions registra el estado de cada paso, de manera puede diagnosticar y depurar los problemas con rapidez.

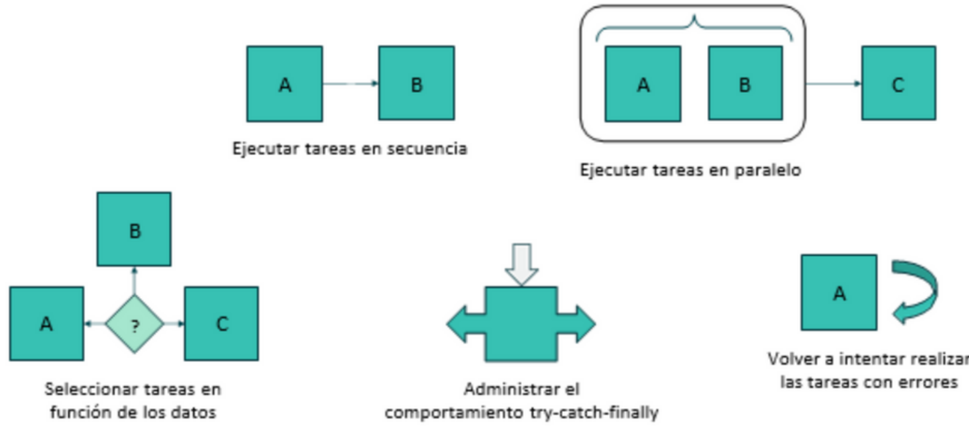
TALENTO TECH

AZ | PROYECTOS EDUCATIVOS

UTP Universidad Tecnológica de Pereira

Coordinación del flujo de trabajo

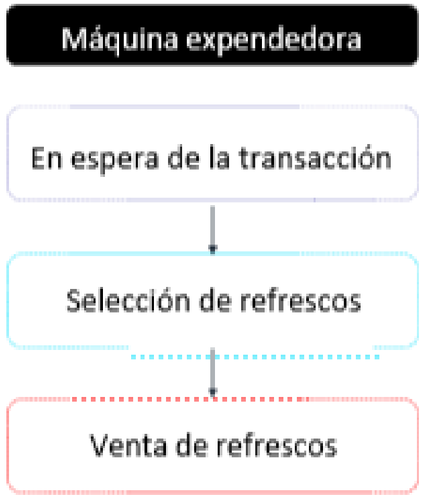
AWS Step Functions administra la lógica de su aplicación por usted. Implementa primitivas básicas, como ejecutar tareas de forma secuencial o en paralelo, ramificar y establecer tiempos de espera. Esta técnica elimina el código adicional que puede repetirse en los microservicios y funciones.



AWS Step Functions administra automáticamente los errores y las excepciones gracias a la función integrada “intentar, capturar y reintentar”, ya sea que la tarea tarde segundo o meses en completarse. Puede reintentar ejecutar automáticamente las tareas que presentaron errores o interrupciones. Puede responder de otra forma a los distintos tipos de errores y recuperarse adecuadamente recurriendo a códigos de limpieza y recuperación designados.

Máquinas de estados

Una máquina de estados es una colección de estados que pueden realizar un trabajo.



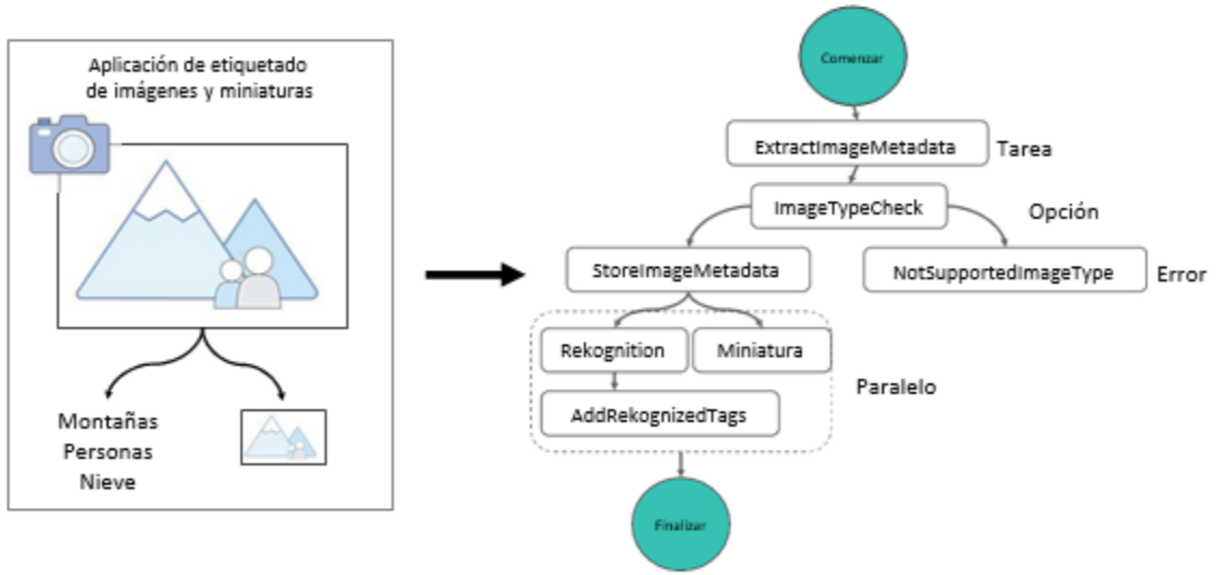
AWS Step Functions le permite crear y automatizar sus propias máquinas de estado dentro del entorno de AWS mediante el uso de Amazon States Language basado en JSON. Este lenguaje contiene una estructura compuesta por varios estados, tareas, elecciones, gestión de errores y más.

Una máquina de estados es una colección de estados que pueden realizar un trabajo.

Un ejemplo común de una máquina de estados es una máquina expendedora de refrescos. La máquina se inicia en estado operacional (a la espera de una transacción) y, luego, pasa a la selección de refrescos cuando se le agrega dinero. A continuación, pasa al estado Venta, en el que el refresco se entrega al cliente. Después de terminar la transacción, el estado vuelve al estado operativo.

Estados

Una máquina de estados finitos puede expresar un algoritmo como un número de estados, sus relaciones y su entrada y salida.



Los estados son elementos de la máquina de estados. Los estados individuales pueden tomar decisiones en función de la entrada, realizar acciones y transferir la salida a otros estados.

Tipos de estado

Tarea	Una sola unidad de trabajo realizada por una máquina de estados
Opción	Incorpora lógica de ramificación a una máquina de estados
Error	Detiene una máquina de estados en ejecución y la marca como error
Correcto	Detiene correctamente una máquina de estados en ejecución
Aprobado	Paso de entrada a la salida, sin realizar ningún trabajo
Esperar	Retrasa la continuación durante un tiempo determinado
Paralelo	Crea ramificaciones de ejecución paralelas en su máquina de estados
Mapa	Itera los pasos de forma dinámica

Los estados pueden realizar varias funciones en una máquina de estado:

- Realizar algunas tareas en la máquina de estado (un estado Task [Tarea])
- Elegir entre las ramificaciones de las máquinas de estado que se van a ejecutar (un estado Choice [Elección])
- Detener una máquina de estados en ejecución con un error o que se realizó correctamente (un estado Fail [Error] o Succeed [Éxito])
- Pasar los datos de entrada a la salida o insertar ciertos datos fijos (un estado Pass [Aprobado])
- Otorgar un plazo que dure cierto tiempo o hasta una fecha y hora determinada (un estado Wait [Espera])
- Comenzar ramificaciones de ejecución paralelas en su máquina de estados (un estado Parallel [Paralelo])
- Iterar pasos de forma dinámica (un estado Map [Mapeo])

Para obtener más información sobre los tipos de estado, consulte Estados en la documentación de AWS.

Amazon States Language

Define el flujo de trabajo en JSON mediante Amazon States Language

```
{
  "Comment": "An example of the ASL.",
  "StartAt": "StartState",
  "States": {
  "StartState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east...,
  "Next": "FinalState"
  }
  "FinalState": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east...,
  "End": true
  }
  }
}
```

Visualiza el flujo de trabajo en la consola Step Functions

Las máquinas de estado se definen utilizando Amazon States Language. Amazon States Language es un lenguaje estructurado basado en JSON. A continuación, AWS Step Functions representa la estructura JSON en una vista gráfica en tiempo real.

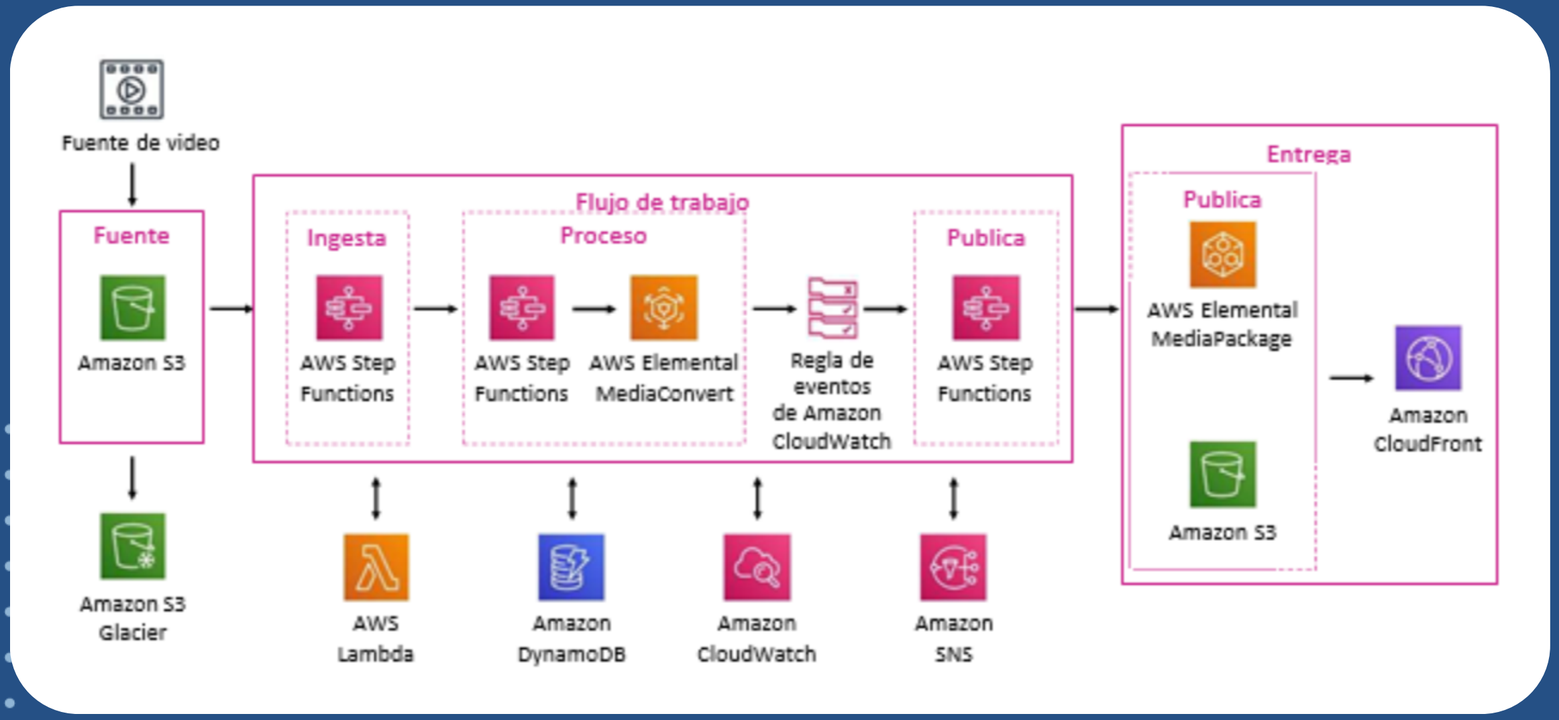
De este modo, puede visualizar su máquina de estados directamente en la consola de Step Functions.

Este es un ejemplo de una máquina de estados con dos tipos de estado de tarea.

Para obtener más información sobre Amazon States Language, consulte la documentación de AWS.

Ejemplo de AWS Step Functions:

Arquitectura de video bajo demanda (VOD)



Este diagrama de arquitectura muestra un ejemplo de caso práctico de AWS Step Functions en una solución de video bajo demanda (VOD). Otro componente clave de esta arquitectura es AWS Elemental MediaConvert, un servicio de transcodificación de video basado en archivos con funciones de transmisión. Permite crear contenido VOD para la transmisión y distribución en pantallas múltiples a gran escala.

En esta solución, los videos de origen y los archivos de metadatos se ingestan y procesan para su reproducción en una amplia gama de dispositivos.

- AWS Step Functions crea Step Functions de Ingest (Ingerir), Process (Procesar) y Publish (Publicar).
- Las funciones de AWS Lambda realizan el trabajo de cada paso y procesan los mensajes de error.
- Los buckets de Amazon S3 almacenan archivos multimedia de origen y destino.
- Amazon CloudWatch se utiliza para el registro.
- Reglas de eventos de Amazon CloudWatch para notificaciones de AWS Elemental MediaConvert.
- Una tabla de Amazon DynamoDB almacena los datos capturados mediante el flujo de trabajo.
- Los temas de Amazon SNS envían notificaciones de codificación, publicación y error.
- Los archivos multimedia transcodificados se almacenan para entregarse a los usuarios bajo demanda a través de Amazon CloudFront.

Para más información sobre esta arquitectura, consulte Información general sobre la arquitectura.

Entre los puntos clave de esta Lección, se incluyen los siguientes:

- AWS Step Functions es un servicio web que facilita la coordinación de componentes de aplicaciones y microservicios distribuidos mediante flujos de trabajo visuales
- AWS Step Functions le permite crear y automatizar sus propias máquinas de estado dentro del entorno de AWS
- AWS Step Functions administra la lógica de su aplicación e implementa primitivas básicas, como las ramificaciones de ejecución paralelas o secuenciales, y los tiempos de espera
- Las máquinas de estado se definen utilizando Amazon States Language

Ahora es el momento de revisar la unidad y concluir con una evaluación de conocimientos.

En resumen, en esta unidad aprendieron a hacer lo siguiente:

- Indicar las características de los microservicios
- Refactorizar una aplicación monolítica en microservicios y utilizar Amazon ECS para implementar los microservicios en contenedores
- Explicar la arquitectura sin servidor
- Implementar una arquitectura sin servidor con AWS Lambda
- Describir una arquitectura común para Amazon API Gateway
- Describir los tipos de flujos de trabajo que admite AWS Step Functions

INICIO