Presentación 2: Arquitectura en la Nube

Reporte de Presentación

David Enrique Palacios García

Gabriel de Souza

Arquitectura de Software

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERIA
CARRERA DE INGENIERÍA DE SISTEMAS

BOGOTÁ, D.C.
2023

Tabla de contenidos

1. Introducción

En la actualidad, la nube se ha convertido en una herramienta fundamental para el almacenamiento, procesamiento y gestión de datos en empresas de diferentes tamaños y sectores. Amazon Web Services (AWS) es uno de los principales proveedores de servicios en la nube, ofreciendo una amplia gama de herramientas y soluciones para satisfacer las necesidades de sus clientes.

Dentro de la plataforma de AWS, se encuentran servicios como Amazon DynamoDB, Amazon API Gateway y AWS Lambda, que permiten la creación de arquitecturas altamente escalables y flexibles para el desarrollo de aplicaciones web y móviles.

En este informe se profundizará en la arquitectura de servicios en la nube de AWS y en particular, se analizará en detalle los beneficios y características de Amazon DynamoDB, Amazon API Gateway y AWS Lambda. Asimismo, se discutirán casos de uso prácticos para cada uno de estos servicios, así como las mejores prácticas para su implementación y gestión.

El objetivo de este informe es brindar una visión general de las opciones disponibles en la nube de AWS, sus beneficios y desafíos, y cómo estos servicios pueden ayudar a las empresas a optimizar su rendimiento y mejorar su eficiencia en el manejo de datos y aplicaciones.

En particular, se explorará cómo la combinación de estos servicios en una arquitectura en la nube puede mejorar la escalabilidad, el rendimiento y la seguridad de las aplicaciones.

Se comenzará presentando una descripción general de la nube de AWS y su modelo de servicio, para luego adentrarse en los detalles técnicos de Amazon DynamoDB, Amazon API Gateway y AWS Lambda. Se explicará cómo funcionan estos servicios individualmente y cómo pueden combinarse para crear una arquitectura en la nube escalable y flexible.

Además, se discutirán los casos de uso más comunes de estos servicios y se presentará un caso práctico, decidimos construir un API REST de información de personas.

Finalmente, se concluirá con una visión general de los beneficios y desafíos de la implementación de una arquitectura en la nube en AWS y se presentarán algunas recomendaciones prácticas para garantizar el éxito en su implementación y gestión.

2. Marco teórico

a. Arquitectura en la Nube

Definición

La arquitectura en la nube se refiere a la estructura y diseño de aplicaciones y servicios de software que se ejecutan en servidores remotos, normalmente en infraestructuras de proveedores de servicios en la nube como Amazon Web Services, Google Cloud Platform o Microsoft Azure. La arquitectura en la nube está diseñada para ser escalable, flexible, y de alta disponibilidad para permitir la entrega de servicios informáticos y aplicaciones a través de Internet de manera eficiente y económica.

La arquitectura en la nube se basa en la virtualización, la automatización, el despliegue automático, la gestión de la configuración, y otros conceptos de ingeniería de software. La arquitectura en la nube también implica una gestión y orquestación de recursos en la nube, donde los recursos se pueden distribuir y escalar automáticamente según las necesidades de las aplicaciones.

Historia y evolución

El término "nube" se popularizó a mediados de la década de 2000, cuando los proveedores de servicios comenzaron a ofrecer servicios informáticos a través de Internet. En 2006, Amazon lanzó su servicio AWS, que permitía a las empresas alojar sus aplicaciones en la nube, lo que permitía una mayor escalabilidad y reducción de costos en comparación con la infraestructura de TI tradicional. En 2008, Google lanzó Google App Engine, que permitía a los desarrolladores alojar aplicaciones web en la nube, mientras que Microsoft lanzó Azure en 2010.

La arquitectura en la nube ha evolucionado desde los servicios de infraestructura básica (IaaS) hasta los servicios de plataforma (PaaS) y software como servicio (SaaS). Actualmente, la arquitectura en la nube se ha convertido en una de las principales tendencias en la industria de la tecnología, ya que permite a las empresas reducir costos, mejorar la eficiencia, y escalar sus aplicaciones de manera rápida y fácil. La arquitectura en la nube también ha permitido una mayor innovación y agilidad en la industria del software, permitiendo a las empresas centrarse en su negocio principal en lugar de preocuparse por la gestión de la infraestructura.

Situaciones prácticas

La arquitectura en la nube es una tecnología que se aplica en muchas situaciones en las que se requiere una entrega de servicios informáticos y aplicaciones a través de Internet de manera eficiente y escalable. Aquí hay algunas situaciones donde se puede aplicar la arquitectura en la nube:

- Empresas en crecimiento: Las empresas que están en crecimiento pueden utilizar la arquitectura en la nube para gestionar su infraestructura de TI de manera escalable y eficiente. La arquitectura en la nube permite a las empresas escalar rápidamente sus recursos informáticos según las necesidades del negocio, lo que permite una mayor agilidad y flexibilidad.
- Empresas que requieren alta disponibilidad: Las empresas que requieren una alta disponibilidad y una infraestructura resistente a fallos pueden utilizar la arquitectura en la nube para implementar soluciones de alta disponibilidad en su infraestructura de TI. La arquitectura en la nube permite a las empresas replicar y distribuir sus aplicaciones y datos en múltiples servidores y regiones, lo que permite una mayor redundancia y resistencia a fallos.
- Empresas que desean reducir costos: Las empresas que desean reducir los costos de infraestructura de TI pueden utilizar la arquitectura en la nube para alojar sus servidores y aplicaciones. La arquitectura en la nube permite a las empresas reducir los costos de infraestructura, ya que no necesitan comprar y mantener su propia infraestructura de hardware.
- Desarrollo y prueba de aplicaciones: Los equipos de desarrollo y pruebas de aplicaciones pueden utilizar la arquitectura en la nube para crear entornos de desarrollo y pruebas escalables y eficientes. La arquitectura en la nube permite a los equipos de desarrollo y pruebas escalar rápidamente sus recursos según las necesidades de las aplicaciones en desarrollo y pruebas, lo que permite una mayor agilidad y flexibilidad.

Ventajas y Desventajas

Ventajas:

- Escalabilidad: La arquitectura en la nube permite escalar los recursos de TI según las necesidades del negocio de manera rápida y eficiente. Esto permite a las empresas aumentar o disminuir sus recursos según las necesidades del negocio, lo que les permite ahorrar costos y aumentar la eficiencia.
- Disponibilidad: La arquitectura en la nube ofrece a las empresas una gran flexibilidad para trabajar desde cualquier lugar y en cualquier momento. Los empleados pueden acceder a los recursos en la nube desde cualquier dispositivo y en cualquier ubicación, lo que aumenta la eficiencia y la colaboración.

- Reducción de costos: La arquitectura en la nube permite a las empresas reducir los costos de infraestructura de TI, ya que no necesitan comprar y mantener su propia infraestructura de hardware. Las empresas pueden optar por modelos de pago por uso, lo que les permite pagar solo por los recursos que utilizan.
- Alta disponibilidad y redundancia: La arquitectura en la nube permite a las empresas replicar y distribuir sus aplicaciones y datos en múltiples servidores y regiones, lo que permite una mayor redundancia y resistencia a fallos. Esto significa que, si un servidor falla, los recursos y datos todavía están disponibles en otros servidores.

Desventajas:

- Dependencia de proveedores de servicios en la nube: Las empresas que dependen de proveedores de servicios en la nube pueden verse afectadas si el proveedor experimenta una interrupción del servicio. También pueden estar sujetos a cambios en los términos y condiciones de los servicios en la nube.
- Problemas de latencia: Las aplicaciones en la nube pueden experimentar problemas de latencia si los servidores están ubicados en regiones geográficas lejanas. Esto puede afectar el rendimiento y la experiencia del usuario.
- Complejidad de gestión: La gestión de una infraestructura en la nube puede ser más compleja que la gestión de una infraestructura de TI tradicional. Las empresas necesitan tener habilidades y herramientas adicionales para administrar y monitorear su infraestructura en la nube.

Casos de estudio

- Netflix: es una de las empresas más grandes que utiliza la arquitectura en la nube para entregar su servicio de transmisión de vídeo en línea. Netflix utiliza Amazon Web Services (AWS) para alojar sus servidores y entregar contenido a millones de clientes en todo el mundo.
- Spotify: es otro ejemplo de una empresa que utiliza la arquitectura en la nube para entregar su servicio de transmisión de música en línea. Spotify utiliza Google Cloud Platform para alojar sus servidores y gestionar su infraestructura de TI.
- Airbnb: utiliza la arquitectura en la nube para alojar su sitio web y sus aplicaciones móviles. Airbnb utiliza AWS y otras tecnologías en la nube para alojar sus servidores y proporcionar una experiencia de usuario sin interrupciones.

b. Amazon Web Services

Definición

Amazon Web Services (AWS) es un servicio de computación en la nube ofrecido por Amazon. AWS es una plataforma que proporciona una amplia gama de servicios de infraestructura de TI, como almacenamiento, bases de datos, análisis, computación y redes. AWS se basa en una arquitectura de múltiples capas que permite a los usuarios acceder a los servicios de AWS de manera escalable y segura.

En la actualidad, AWS ofrece más de 200 servicios en la nube diferentes. Estos servicios cubren una amplia variedad de áreas, como la computación, el almacenamiento, la base de datos, la analítica, la inteligencia artificial, la seguridad, el Internet de las cosas (IoT) y mucho más. AWS sigue agregando nuevos servicios y mejorando los existentes para satisfacer las necesidades de sus clientes en constante evolución.

AWS ofrece una capa gratuita que permite a los usuarios acceder a una variedad de servicios de AWS de forma gratuita. Esta capa gratuita es ideal para los usuarios que desean explorar y experimentar con AWS antes de comprometerse a pagar por los servicios. La capa gratuita de AWS incluye un conjunto de servicios con límites de uso gratuitos durante un período de 12 meses. Algunos de los servicios que están disponibles en la capa gratuita incluyen Amazon EC2, Amazon S3, HTTP API Gateway, Amazon DynamoDB y AWS Lambda, entre otros. La capa gratuita también incluye acceso gratuito a la capacitación y los recursos de soporte de AWS, lo que permite a los usuarios aprender a usar AWS y resolver cualquier

problema que puedan encontrar. Cabe mencionar que, aunque la capa gratuita ofrece una excelente oportunidad para explorar AWS, es importante tener en cuenta que ciertos servicios pueden estar sujetos a tarifas después de que se superan los límites de uso gratuitos.

Historia y evolución

AWS se lanzó en el año 2006, cuando Amazon lanzó su primer servicio de infraestructura en la nube, Amazon S3. A medida que las empresas comenzaron a darse cuenta de las ventajas de la informática en la nube, AWS se convirtió en uno de los principales proveedores de servicios en la nube, gracias a su amplia gama de servicios y su innovación constante.

En los primeros años, AWS se centró en proporcionar servicios básicos de infraestructura, como almacenamiento, cómputo y redes. Con el tiempo, AWS agregó nuevos servicios y herramientas, como bases de datos, inteligencia artificial, análisis, aprendizaje automático, Internet de las cosas (IoT) y seguridad, entre otros. AWS ha seguido expandiéndose para satisfacer las necesidades en constante evolución de sus clientes, incluidas las empresas más grandes y complejas.

Además, AWS ha lanzado una serie de iniciativas para facilitar el uso de sus servicios en la nube, como AWS Marketplace, que permite a los clientes buscar, comprar y vender software en la nube; y AWS Educate, que proporciona recursos y capacitación en la nube para estudiantes y educadores.

En la actualidad, AWS es uno de los mayores proveedores de servicios en la nube, y su plataforma se utiliza en una amplia variedad de industrias y aplicaciones. La evolución continua de AWS ha sido fundamental para permitir a las empresas y organizaciones utilizar la tecnología en la nube de manera efectiva para innovar, crecer y competir en la economía digital.

Situaciones prácticas

AWS se puede aplicar en una amplia variedad de situaciones, algunas de las cuales incluyen:

- Hospedaje de sitios web y aplicaciones: AWS proporciona una infraestructura en la nube escalable y confiable para alojar sitios web y aplicaciones, lo que permite a las empresas ofrecer una experiencia de usuario de alta calidad y satisfacer la demanda del tráfico.
- Almacenamiento y recuperación de datos: AWS proporciona servicios de almacenamiento en la nube como Amazon S3 y Amazon EBS, lo que permite a las empresas almacenar grandes cantidades de datos de manera segura y acceder a ellos de manera rápida y eficiente.
- Análisis de datos y big data: AWS proporciona una amplia gama de herramientas y servicios para analizar y procesar grandes cantidades de datos, como Amazon Redshift, Amazon Athena, Amazon EMR y Amazon Kinesis, lo que permite a las empresas obtener información valiosa y tomar decisiones informadas.
- Internet de las cosas (IoT): AWS proporciona servicios y herramientas para construir y administrar aplicaciones de IoT, lo que permite a las empresas recolectar y procesar datos de dispositivos conectados y aplicar análisis avanzados para obtener información útil.
- Desarrollo y pruebas: AWS proporciona una variedad de herramientas y servicios para el desarrollo y prueba de aplicaciones, como Amazon EC2, AWS CodeCommit, AWS CodeBuild y AWS CodePipeline, lo que permite a las empresas desarrollar y probar aplicaciones de manera rápida y eficiente.
- Machine learning e inteligencia artificial: AWS proporciona servicios y herramientas para construir, entrenar y desplegar modelos de machine learning e inteligencia artificial, como Amazon SageMaker, Amazon Rekognition y Amazon Polly, lo que permite a las empresas aplicar técnicas de machine learning e inteligencia artificial para mejorar sus productos y servicios.

Ventajas y Desventajas

Ventajas:

- Escalabilidad: AWS ofrece una infraestructura en la nube altamente escalable, lo que permite a las empresas aumentar o disminuir sus recursos de acuerdo con sus necesidades en tiempo real.
- Confiabilidad: AWS es conocido por su alta disponibilidad y durabilidad. La plataforma ofrece una arquitectura en la nube altamente redundante y resiliente que minimiza el riesgo de tiempo de inactividad.
- Flexibilidad: AWS ofrece una amplia variedad de servicios y herramientas, lo que permite a las empresas adaptarse a diferentes situaciones y casos de uso.
- Pago por uso: AWS ofrece un modelo de pago por uso, lo que significa que las empresas solo pagan por los recursos que utilizan.
- Seguridad: AWS ofrece una amplia variedad de herramientas y servicios de seguridad para proteger los datos y aplicaciones de sus clientes.

Desventajas:

- Complejidad: AWS puede ser complicado de entender y configurar para las personas sin experiencia en la nube.
- Costo: Aunque el modelo de pago por uso puede ser beneficioso para algunas empresas, AWS puede ser costoso para aquellas que tienen altos requisitos de recursos y uso.
- Dependencia de proveedores: Al usar AWS, las empresas pueden volverse dependientes de los servicios de la plataforma, lo que puede ser un problema si hay problemas o limitaciones en el futuro.
- Problemas de cumplimiento: AWS tiene una amplia variedad de certificaciones de cumplimiento, pero las empresas aún pueden encontrar problemas de cumplimiento al usar la plataforma.

Casos de estudio

- Unilever: la gigante de bienes de consumo utiliza AWS para alojar sus aplicaciones empresariales y analizar datos para la toma de decisiones.
- Capital One: la empresa de servicios financieros utiliza AWS para alojar sus aplicaciones empresariales y almacenar datos confidenciales de sus clientes.
- Pinterest: la popular plataforma de redes sociales utiliza AWS para alojar su sitio web y aplicaciones móviles, así como para almacenar y procesar datos de sus usuarios.

Porcentajes de uso

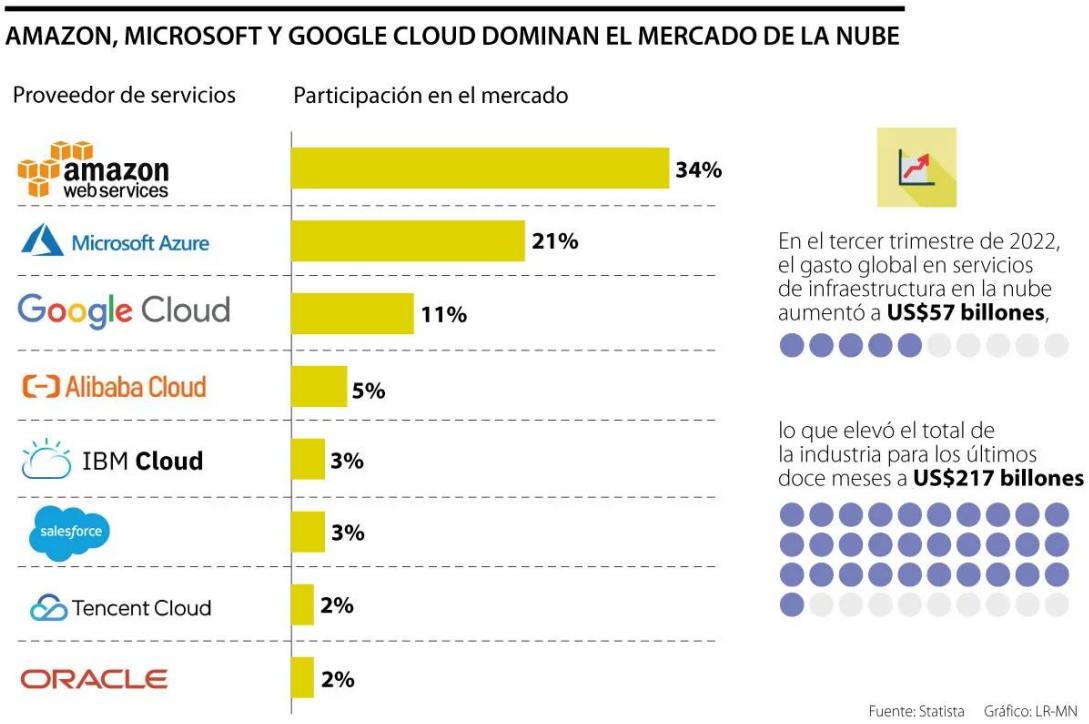

Ilustración 1: Competencia en el mercado de la Nube

Según La República (La República, 2023), AWS es líder del mercado Cloud con un 34% de la cuota del mercado. Lo siguen Microsoft Azure y Google Cloud con un 21% y 11% respectivamente. A pesar de que los 3 grandes tienen una parte importante del mercado, existen competidores que no hay que dejar a un lado, como Alibaba e IBM.

c. Function as a Service (AWS Lambda)

Definición

Function as a Service (FaaS) es un modelo de computación en la nube que permite a los desarrolladores escribir y cargar funciones de software en una plataforma en la nube, donde se ejecutan de forma escalable y bajo demanda en respuesta a eventos o solicitudes específicas. En otras palabras, FaaS proporciona un entorno en el que los desarrolladores pueden centrarse en escribir pequeñas piezas de código (funciones) que resuelven problemas específicos y luego desplegar y ejecutar esas funciones en la nube. La infraestructura subyacente, incluyendo la asignación de recursos, la gestión de la carga y el equilibrio de la carga es manejada automáticamente por el proveedor de la nube.

AWS Lambda es un servicio de Function as a Service (FaaS) que permite a los desarrolladores cargar y ejecutar código sin tener que preocuparse por la infraestructura subyacente. Con AWS Lambda, los desarrolladores pueden escribir pequeñas piezas de código (funciones) y desplegarlas en la nube. Las funciones se ejecutan automáticamente en respuesta a eventos específicos, como la carga de datos, una solicitud de API o una interacción de usuario.

AWS Lambda es un servicio completamente administrado que escala automáticamente el rendimiento para manejar la carga de trabajo entrante. Los desarrolladores solo pagan por el tiempo de ejecución de su función y no por el tiempo que está en espera, lo que permite un uso eficiente de los recursos y un ahorro de costos significativo. AWS Lambda también se integra con otros servicios de AWS, lo que permite a los desarrolladores crear aplicaciones complejas que aprovechan la funcionalidad de varios servicios de AWS.

Ilustración 2: Capa gratuita: AWS Lambda

Historia y evolución

AWS Lambda se lanzó por primera vez en noviembre de 2014 en la conferencia anual de AWS, re:Invent. En ese momento, era uno de los primeros servicios en ofrecer un modelo de programación sin servidor. Desde entonces, AWS Lambda se ha convertido en uno de los servicios más populares de AWS, y se ha utilizado para construir aplicaciones y servicios escalables y altamente disponibles en una amplia variedad de sectores y aplicaciones.

Desde su lanzamiento inicial, AWS Lambda ha evolucionado significativamente en términos de características y funcionalidad. AWS ha lanzado numerosas actualizaciones y mejoras a lo largo de los años, incluyendo:

- Adición de soporte para más lenguajes de programación, incluyendo Python, Java, C# y Go.
- Mejoras en la integración con otros servicios de AWS, como Amazon S3, Amazon DynamoDB y Amazon API Gateway.
- Mejoras en el rendimiento y la escalabilidad, lo que permite a los desarrolladores manejar cargas de trabajo aún mayores y más complejas.
- Lanzamiento de herramientas y características adicionales, como AWS Step Functions y AWS SAM (Serverless Application Model), para facilitar la creación y el despliegue de aplicaciones sin servidor.

En la actualidad, AWS Lambda es uno de los servicios más utilizados en la plataforma AWS, y ha sido fundamental para la popularización del modelo de programación sin servidor.

Situaciones prácticas

AWS Lambda se puede aplicar en una amplia variedad de situaciones, algunas de las cuales son:

- Procesamiento de datos en tiempo real: AWS Lambda puede utilizarse para procesar y analizar grandes cantidades de datos en tiempo real. Por ejemplo, puede utilizarse para realizar un seguimiento en tiempo real de los usuarios que acceden a un sitio web o para procesar datos de sensores en una fábrica.
- Automatización de tareas: AWS Lambda puede automatizar tareas como el procesamiento de correos electrónicos, la actualización de bases de datos, la generación de informes, entre otros.
- Procesamiento de imágenes y videos: AWS Lambda puede utilizarse para procesar y analizar imágenes y videos de manera eficiente y escalable. Por ejemplo, puede utilizarse para crear miniaturas de imágenes o para procesar videos para su transmisión.
- Procesamiento de solicitudes de API: AWS Lambda puede utilizarse para procesar solicitudes de API y devolver respuestas en tiempo real. Puede integrarse con otros servicios de AWS, como Amazon API Gateway, para crear aplicaciones escalables y de alta disponibilidad.

Ventajas y Desventajas

Ventajas:

- Escalabilidad automática: AWS Lambda escala automáticamente para manejar una gran cantidad de solicitudes simultáneas, lo que permite a los desarrolladores centrarse en escribir código de alta calidad en lugar de preocuparse por la infraestructura subyacente.
- Pago por uso: AWS Lambda permite a los desarrolladores pagar solo por el tiempo de ejecución de su código, lo que lo convierte en una solución rentable para aplicaciones con cargas de trabajo variables.
- Integración con otros servicios de AWS: AWS Lambda se integra sin problemas con otros servicios de AWS, como Amazon S3, Amazon DynamoDB, Amazon API Gateway y Amazon CloudFront, lo que facilita la creación de aplicaciones escalables y de alta disponibilidad.
- Configuración flexible: AWS Lambda permite a los desarrolladores personalizar la configuración de su función en función de sus necesidades, lo que les permite optimizar el rendimiento de su aplicación.

Desventajas:

- Tiempo de inicio: AWS Lambda puede tardar unos segundos en iniciar la ejecución de una función, lo que puede retrasar la respuesta de la aplicación para solicitudes de tiempo real.
- Límites de tiempo de ejecución: AWS Lambda limita la duración de la ejecución de una función a un máximo de 15 minutos, lo que puede limitar su capacidad para procesar grandes conjuntos de datos o realizar tareas que requieren mucho tiempo.
- Limitaciones de memoria: AWS Lambda limita la cantidad de memoria que puede asignarse a una función, lo que puede limitar su capacidad para procesar grandes conjuntos de datos o realizar tareas que requieren mucha memoria.

Casos de estudio

AWS Lambda es una tecnología utilizada por una gran cantidad de empresas, algunas de ellas son:

- Netflix: utiliza AWS Lambda para procesar grandes cantidades de datos y analizar los patrones de visualización de sus usuarios. También lo utiliza para implementar funciones de notificación y automatización de tareas.
- Airbnb: utiliza AWS Lambda para gestionar la escalabilidad de su plataforma y realizar tareas de procesamiento en tiempo real, como la detección de fraude y la clasificación de datos.
- Thomson Reuters: utiliza AWS Lambda para procesar grandes conjuntos de datos y generar informes personalizados en tiempo real para sus clientes.

d. API Gateway (Amazon API Gateway)

Definición

Amazon API Gateway es un servicio de AWS que permite a los desarrolladores crear, publicar, mantener, monitorear y proteger APIs (Interfaces de Programación de Aplicaciones) de manera fácil y segura.

Las APIs son una parte fundamental en el desarrollo de aplicaciones modernas, permitiendo a diferentes aplicaciones y sistemas comunicarse entre sí y compartir datos y funcionalidades. Amazon API Gateway facilita la creación y el despliegue de APIs en AWS, lo que permite a las empresas desarrollar y desplegar sus aplicaciones de manera más rápida y eficiente.

API Gateway proporciona una capa de abstracción sobre la infraestructura de la nube, lo que permite a los desarrolladores enfocarse en el diseño y la implementación de sus APIs sin preocuparse por los detalles de la infraestructura subyacente. Además, API Gateway puede integrarse con otros servicios de AWS, como AWS Lambda y Amazon DynamoDB, para construir aplicaciones escalables y altamente disponibles.

En este sentido, Amazon API Gateway se convierte en una herramienta importante para permitir la exposición y gestión de los servicios en la nube, así como para el diseño y desarrollo de arquitecturas en la nube altamente escalables y flexibles.

Ilustración 3: Capa gratuita AWS API Gateway

Ventajas y Desventajas

Ventajas:

- Facilidad de uso: Amazon API Gateway ofrece una interfaz gráfica intuitiva y sencilla para crear y configurar APIs, lo que permite a los desarrolladores crear y publicar rápidamente sus servicios en la nube.
- Escalabilidad: Amazon API Gateway es altamente escalable, lo que permite a las empresas manejar grandes volúmenes de tráfico sin preocuparse por la sobrecarga del sistema.
- Integración con otros servicios de AWS: API Gateway puede integrarse con otros servicios de AWS, como AWS Lambda y Amazon DynamoDB, para construir aplicaciones escalables y altamente disponibles.

- Seguridad: API Gateway ofrece herramientas de seguridad y protección de datos para proteger las APIs contra ataques y asegurar la privacidad de los usuarios.
- Monitorización y análisis: API Gateway ofrece herramientas para monitorear y analizar el rendimiento de las APIs, lo que permite a los desarrolladores detectar y solucionar problemas rápidamente.

Desventajas:

- Costos: El uso de Amazon API Gateway puede generar costos adicionales, especialmente si se utiliza para manejar grandes volúmenes de tráfico.
- Limitaciones de personalización: Aunque Amazon API Gateway es altamente escalable y fácil de usar, la personalización de las APIs puede ser limitada en algunos casos, especialmente en comparación con soluciones personalizadas de infraestructura de API.
- Dependencia de AWS: El uso de Amazon API Gateway implica una dependencia en la infraestructura de AWS, lo que puede ser un inconveniente para algunas empresas que prefieren utilizar infraestructura propia o de otros proveedores.
- Limitaciones en el tamaño del payload: Amazon API Gateway tiene un límite en el tamaño del payload que se puede enviar y recibir en una API. Este límite puede ser insuficiente para algunas aplicaciones que manejan grandes cantidades de datos.

Casos de estudio

A continuación, presentamos algunos casos de estudio de empresas que han utilizado Amazon API Gateway para desarrollar y desplegar sus APIs en la nube:

- Airbnb: Airbnb, una plataforma de alojamiento en línea, utiliza Amazon API Gateway para exponer su API de reservas y búsquedas de alojamiento a través de la web y dispositivos móviles. API Gateway se integra con AWS Lambda para manejar la lógica de la aplicación y Amazon DynamoDB para almacenar los datos.
- The Washington Post: The Washington Post, un periódico de los Estados Unidos, utiliza Amazon API Gateway para desarrollar una API para sus lectores que permite buscar, leer y compartir noticias. API Gateway se integra con AWS Lambda para manejar la lógica de la aplicación y Amazon RDS para almacenar los datos.
- Philips Hue: Philips Hue, una marca de iluminación inteligente, utiliza Amazon API Gateway para exponer su API a través de la web y dispositivos móviles. API Gateway se integra con AWS Lambda para manejar la lógica de la aplicación y Amazon DynamoDB para almacenar los datos.

e. Dynamo DB

Definición

DynamoDB es un servicio de base de datos NoSQL totalmente administrado por Amazon Web Services (AWS). Este servicio permite a las empresas almacenar y recuperar grandes volúmenes de datos, con alta disponibilidad y escalabilidad.

A diferencia de las bases de datos relacionales tradicionales, DynamoDB no requiere la definición de un esquema fijo para los datos, lo que permite una mayor flexibilidad en la estructura de los datos. Además, DynamoDB se integra bien con otros servicios de AWS, como Amazon API Gateway y AWS Lambda, para permitir la creación de aplicaciones escalables y altamente disponibles en la nube.

Ilustración 4: Capa gratuita DynamoDB

Ventajas y Desventajas

Ventajas:

- Escalabilidad: DynamoDB está diseñado para manejar grandes volúmenes de datos y altas tasas de transacciones, lo que lo convierte en una opción ideal para aplicaciones con una gran cantidad de usuarios y datos.
- Alta disponibilidad: DynamoDB está diseñado para ser altamente disponible, con redundancia de datos y servicios en múltiples zonas de disponibilidad en todo el mundo, lo que garantiza una alta disponibilidad incluso en caso de fallos de hardware o software.
- Flexibilidad en la estructura de los datos: DynamoDB no requiere un esquema fijo para los datos, lo que permite una mayor flexibilidad en la estructura de los datos y facilita la adición de nuevos datos y campos a medida que cambian las necesidades de la aplicación.
- Integración con otros servicios de AWS: DynamoDB se integra bien con otros servicios de AWS, como Amazon API Gateway y AWS Lambda, para permitir la creación de aplicaciones escalables y altamente disponibles en la nube.
- Fácil de usar: DynamoDB ofrece una interfaz de usuario fácil de usar y una API simple para la creación y gestión de bases de datos NoSQL.

Desventajas:

- Costos: El uso de DynamoDB puede generar costos adicionales, especialmente si se utiliza para manejar grandes volúmenes de datos y transacciones.

- Limitaciones en la consulta de datos: DynamoDB no ofrece soporte para consultas complejas como JOINs o consultas de tipo OR. Las consultas deben ser diseñadas específicamente para utilizar las claves de partición y ordenamiento de la base de datos.
- Dependencia de AWS: El uso de DynamoDB implica una dependencia en la infraestructura de AWS, lo que puede ser un inconveniente para algunas empresas que prefieren utilizar infraestructura propia o de otros proveedores.

A pesar de estas limitaciones, DynamoDB sigue siendo una opción popular para muchas empresas debido a su alta escalabilidad y disponibilidad, su facilidad de uso y su integración con otros servicios de AWS.

Casos de estudio

A continuación, presentamos algunos casos de estudio de empresas que han utilizado DynamoDB para gestionar sus datos en la nube:

- Samsung: Samsung, una empresa de tecnología, utiliza DynamoDB para almacenar y gestionar los datos de su aplicación de televisión inteligente. La alta escalabilidad de DynamoDB ha permitido a Samsung manejar grandes volúmenes de datos y proporcionar una experiencia de usuario rápida y sin interrupciones.
- Lyft: Lyft, una plataforma de transporte en línea, utiliza DynamoDB para gestionar los datos de sus conductores y pasajeros, como la información de la cuenta y la ubicación en tiempo real. La alta disponibilidad de DynamoDB ha permitido a Lyft garantizar que la plataforma esté siempre disponible para los conductores y los pasajeros.
- Dow Jones: Dow Jones, una empresa de noticias y medios de comunicación, utiliza DynamoDB para almacenar y gestionar los datos de su plataforma de noticias en tiempo real. La alta escalabilidad de DynamoDB ha permitido a Dow Jones manejar grandes volúmenes de datos y proporcionar una experiencia de usuario rápida y sin interrupciones.

En general, DynamoDB es una opción atractiva para muchas empresas que necesitan almacenar y gestionar grandes volúmenes de datos en la nube de manera eficiente y escalable.

3. Caso práctico

Para nuestro caso práctico, decidimos construir un API REST de información de personas (id, nombre, apellidos y edad) basados en la siguiente arquitectura:

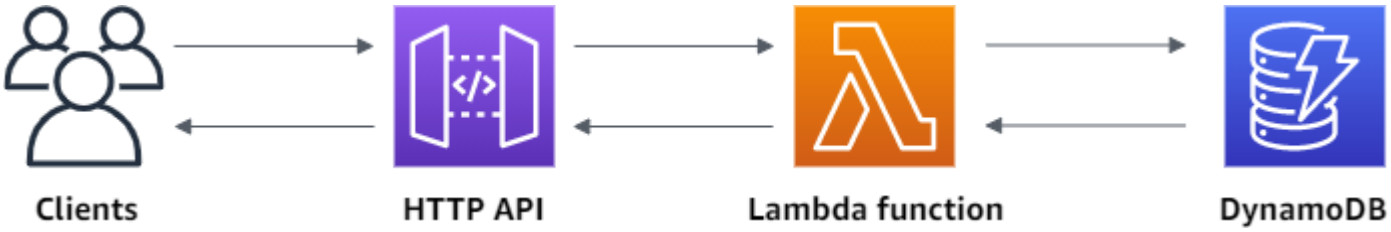

Ilustración 5: Arquitectura de Alto Nivel

En ella, tenemos una serie de clientes (App en React, Postman) que se conectan a un API Gateway configurado y provisto por AWS. Este API Gateway gatilla 3 funciones Lambda según la ruta configurada; una de escritura, una de lectura y una de eliminación. Cada una de estas funciones, interactúa con la tabla creada en DynamoDB, lo que actualiza los registros según los eventos creados por el usuario.

Cabe resaltar, que todo el proceso fue guiado por el siguiente tutorial: AWS Tutorial, en donde se explica la creación, configuración y uso de todos los servicios. El repositorio final se encuentra en GitHub, con las instrucciones para ejecutar la aplicación en React.

De esta manera, se integran los servicios provistos con AWS pudiéndolos utilizar gracias a su capa gratuita. A continuación, se muestra el uso y configuración de todos los servicios utilizados.

DynamoDB

Lo primero que se hizo fue la creación de una tabla de DynamoDB para el almacenamiento de la información de nuestro API. Cabe resaltar que, al ser NoSQL, no necesita de la especificación de un esquema ni restricciones de datos. Además, se agregó la clave de partición *id* como la llave principal de cada registro, pues este representa la llave única de cada persona.

Ilustración 6: Tabla personas DynamoDB

AWS Lambda

Teniendo lista la tabla para almacenar datos, lo siguiente fue crear las funciones que interactuarían como controladores para los endpoints de nuestra API. Aquí, es importante aclarar que se siguió paso a paso del tutorial, por lo que las funciones se construyeron de manera muy similar a como estaba allí. Es por ello que se utilizó Node JS (Javascript) y el SDK de DynamoDB para construirlas. A continuación, se enseña cada una de las funciones y su utilización:

1) Lectura de datos: En este caso, se cuenta con 2 endpoints GET, */personas* y */personas/{id}*. El primero escanea toda la base de datos y devuelve sus ítems, y el segundo busca la persona con el id brindado.

get.mjs

```mjs
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  DynamoDBDocumentClient,
  ScanCommand,
  GetCommand,
} from "@aws-sdk/lib-dynamodb";

const client = new DynamoDBClient({});

const dynamo = DynamoDBDocumentClient.from(client);

const tableName = "http-crud-personas-arquitectura";

export const handler = async (event, context) => {
  let body;
  let statusCode = 200;
  const headers = {
    "Content-Type": "application/json",
  };

  try {
    switch (event.routeKey) {
      case "GET /personas/{id}":
        body = await dynamo.send(
          new GetCommand({
            TableName: tableName,
            Key: {
              id: event.pathParameters.id,
            },
          })
        );
        body = body.Item;
        break;
      case "GET /personas":
        body = await dynamo.send(
          new ScanCommand({ TableName: tableName })
        );
        body = body.Items;
        break;
      default:
        throw new Error(`Unsupported route: "${event.routeKey}"`);
    }
  } catch (err) {
    statusCode = 400;
    body = err.message;
  } finally {
    body = JSON.stringify(body);
  }

  return {
    statusCode,
    body,
    headers,
  };
};
```

Snipped

Ilustración 7: Lectura de datos AWS Lambda

2) Escritura de datos: Para la escritura únicamente se crea un endpoint: PUT /personas, donde se busca crear o actualizar un registro. Cabe resaltar que el SDK de Dynamo DB no ofrece un método específico para POST, por lo que el tutorial directamente lo hace de esta manera.

put.mjs

```
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  DynamoDBDocumentClient,
  PutCommand,
} from "@aws-sdk/lib-dynamodb";

const client = new DynamoDBClient({});

const dynamo = DynamoDBDocumentClient.from(client);

const tableName = "http-crud-personas-arquitectura";

export const handler = async (event, context) => {
  let body;
  let statusCode = 200;
  const headers = {
    "Content-Type": "application/json",
  };

  try {
    switch (event.routeKey) {
      case "PUT /personas":
        let requestJSON = JSON.parse(event.body);
        const Item = {
            id: requestJSON.id,
            name: requestJSON.name,
            lastName: requestJSON.lastName,
            age: requestJSON.age,
        }
        await dynamo.send(
          new PutCommand({
            TableName: tableName,
            Item: Item,
          })
        );
        body = Item;
        break;
      default:
        throw new Error(`Unsupported route: "${event.routeKey}"`);
    }
  } catch (err) {
    statusCode = 400;
    body = err.message;
  } finally {
    body = JSON.stringify(body);
  }

  return {
    statusCode,
    body,
    headers,
  };
};

Snipped
```

Ilustración 8: Escritura de datos AWS Lambda

3) Eliminación de datos: Para este caso, también se cuenta únicamente con un endpoint: DELETE /personas/{id}, donde, gracias al SDK de DynamoDB es muy sencillo el eliminar registros de la base de datos.

Ilustración 9: Eliminación de datos AWS Lambda

API Gateway

Para la utilización del API Gateway de Amazon, lo primero es su creación y asignación de sus rutas. El nuestro tiene este aspecto:

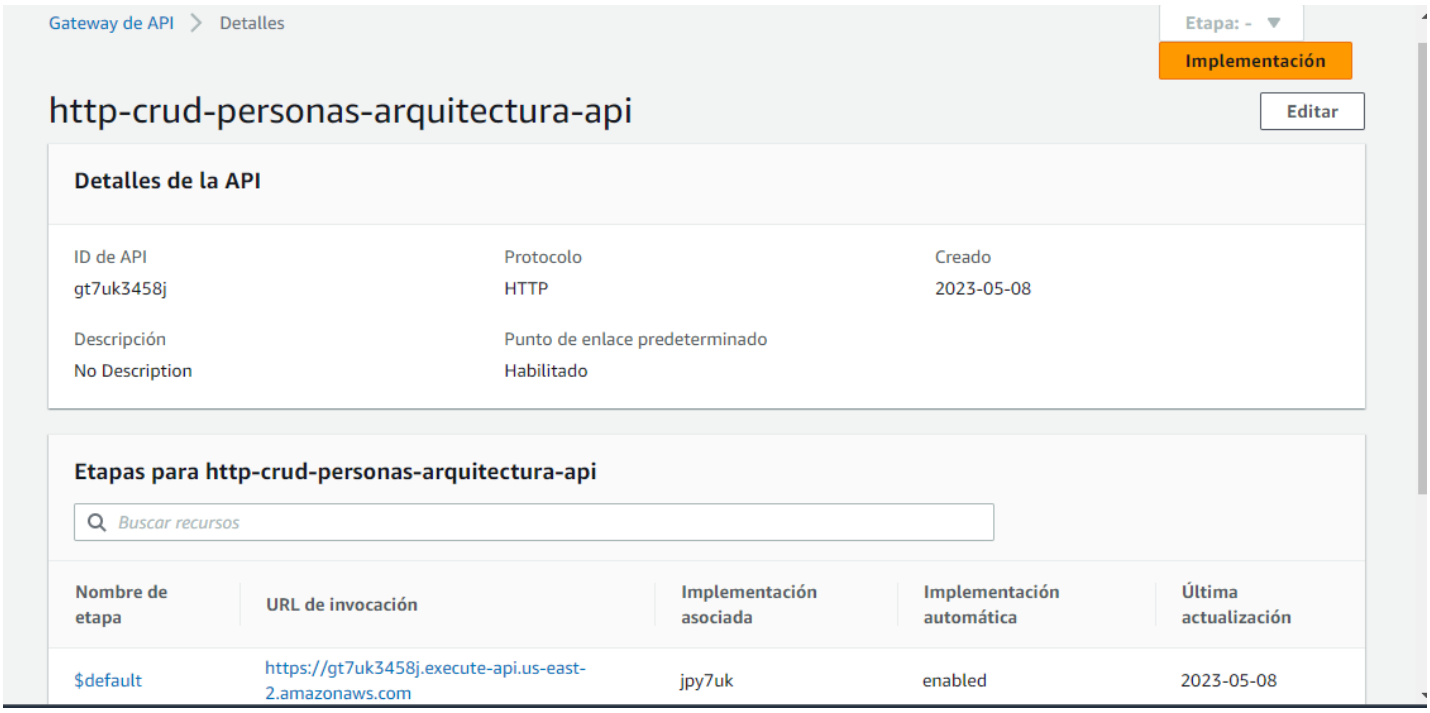

Ilustración 10: API Gateway AWS

Vale la pena nombrar el *url de invocación* pues es la dirección pública que en este caso nos brindó AWS con el despliegue del API Gateway, por lo que es el punto de acceso a nuestro API Rest. El enlace al endpoint de lectura de todos los registros es el siguiente: /personas.

Para la creación de rutas, se utilizaron los endpoints descritos anteriormente en las funciones lambda, dejando como resultado las siguientes 4 rutas:

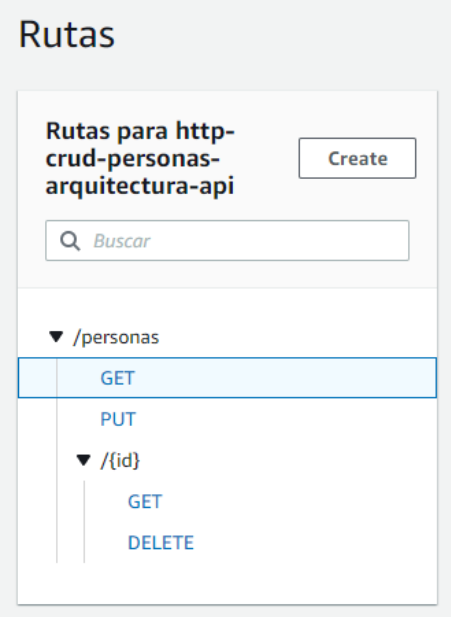

Ilustración 11: Rutas API Gateway

Lo único que haría falta es integrar el API Gateway con cada una de las funciones Lambda creadas anteriormente. En este punto, AWS permite crear roles de acceso a la aplicación, por lo que se permite la conexión entre cada uno de los endpoints con la ruta específica del API Gateway. De esta manera, nuestros enlaces se ven así:

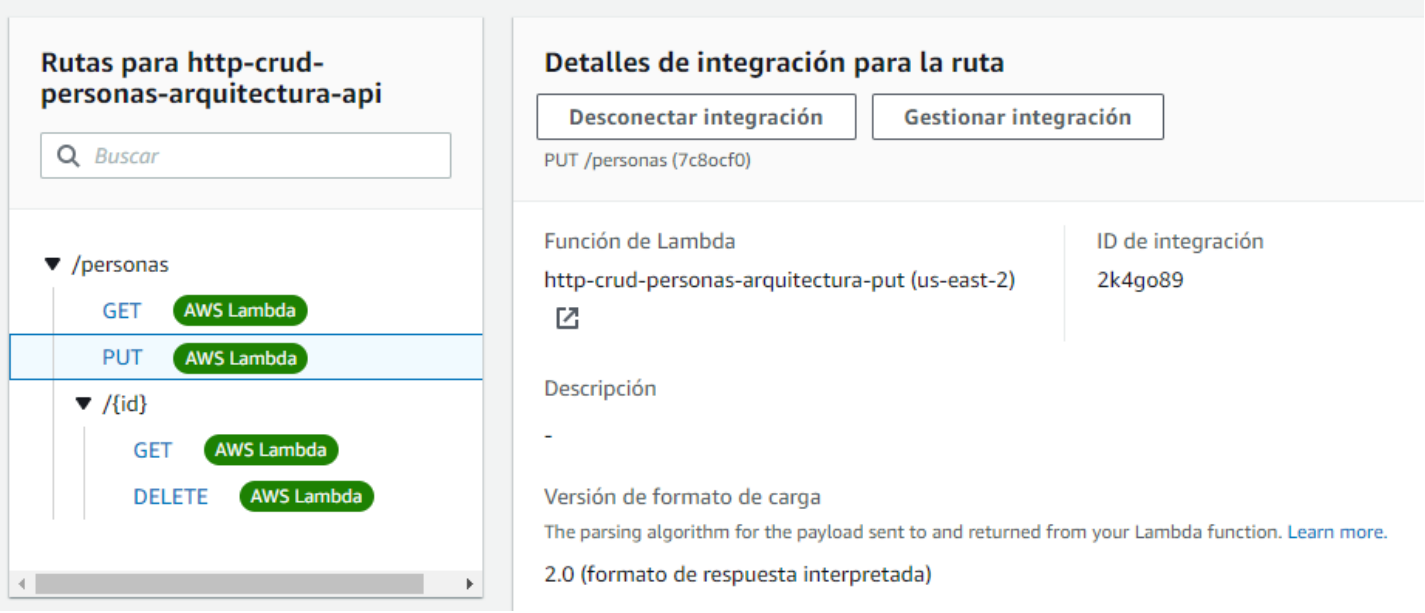

Ilustración 12: Integración del API Gateway y las funciones Lambda

Por último, vale la pena resaltar la integración predeterminada que tiene el servicio con el mecanismo CORS, por lo que su configuración es muy sencilla y permite añadir una capa de seguridad adicional en nuestra aplicación. Para esta presentación de carácter académico, se añadió el acceso de todos los orígenes disponibles y de todos los métodos. La configuración se ve de esta manera:

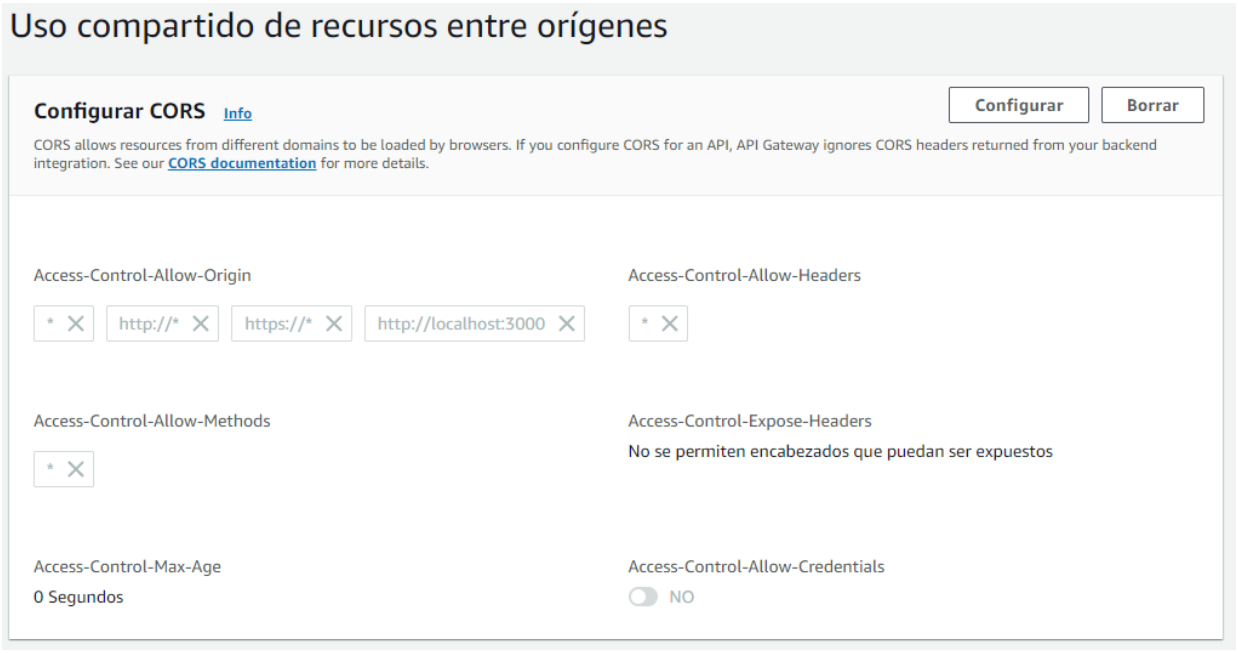

Ilustración 13: Configuración CORS API Gateway

De esta manera, queda completamente configurada la conexión entre el API Gateway, las funciones Lambda y DynamoDB, permitiéndonos usar los endpoints descritos utilizando la url expuesta anteriormente, en diversos clientes como nuestra aplicación en React o herramientas como Postman.

4. Conclusiones y Lecciones aprendidas

En conclusión, la arquitectura basada en los servicios de la nube de Amazon como DynamoDB, API Gateway y AWS Lambda ofrece una solución escalable, flexible y altamente disponible para la gestión de datos y la creación de aplicaciones en la nube.

En el caso específico de este proyecto, se pudo ver cómo la integración de API Gateway con las funciones Lambda permitió la creación de una API RESTful para la gestión de datos en DynamoDB. Esto permitió que la aplicación React y otras aplicaciones cliente pudieran interactuar fácilmente con los datos almacenados en DynamoDB a través de la API Gateway.

Además, la configuración de múltiples funciones Lambda permitió separar las responsabilidades de escritura, lectura y eliminación de los datos, lo que permitió una mejor escalabilidad y mantenimiento del sistema.

En cuanto a las lecciones aprendidas, se destaca la importancia de tener un buen diseño de la arquitectura antes de comenzar a desarrollar el sistema, para poder aprovechar al máximo las ventajas de los servicios de la nube de Amazon. Además, se debe prestar atención a las limitaciones de cada servicio y planificar en consecuencia.

En resumen, la combinación de DynamoDB, API Gateway y AWS Lambda ofrece una solución eficaz y escalable para la gestión de datos en la nube, lo que permite a las empresas desarrollar aplicaciones de alta calidad y alto rendimiento.