Aplicación Serverless en AWS

Pablo Bernabeu

Índice

1. Introducción

Actualmente cualquier empresa necesita la tecnología informática para su funcionamiento diario, es tan importante, que puede ser causa de su fracaso o éxito. Por ello, es necesario que exista una infraestructura que de soporte a todo su sistema informático.
Con infraestructura, definimos toda la plataforma tecnológica que utiliza el sistema para brindar servicios al usuario final, convirtiéndose en la base de un sistema informático de cualquier organización.

Las primeras empresas en publicar una aplicación web hosteaba su código de forma 'on-premise', es decir, ellos mismos compraban sus ordenadores, los configuraban y desplegaban de código en el. Aunque esto sigue siendo normal, en los últimos años el concepto de *'Cloud'* ha ido tomando más relevancia, de forma qué muchas grandes empresas han migrado su infraestructura a plataformas Cloud, cómo es el caso de Netflix.

¿Qué es Cloud? Básicamente este término hace referencia a usar redes de ordenadores almacenadas y administradas por otras empresas, cómo Amazon, Google, etc. para hostear nosotros nuestras aplicaciones. Es decir, en vez de comprar nosotros un ordenador, configurarlo y mantenerlo, contratamos una empresa Cloud la cual nos ofrece sus ordenadores para usar con el fin qué queramos. Esto tiene ventajas cómo ahorrarnos el gasto de comprar un ordenador, pagar sus gastos de mantenimiento, despreocuparnos de si el ordenador se rompe, deja de funcionar o su hardware es antiguo, etc.

Entonces, ¿no nos tenemos qué preocupar de administrar nuestros servidores? Estas plataformas Cloud ofrecen servidores directamente cómo servicio, el cual podemos usar de forma idéntica cómo si tuviéramos uno físico. Pero, ¿y si vamos un paso más adelante? Un concepto nuevo donde las empresas puedan ejecutar aplicaciones y servicios sin tener que administrar servidores. De manera que, los desarrolladores se puedan enfocar en el producto principal, en lugar de preocuparse por el funcionamiento de los servidores, o los tiempos de ejecución, tanto en la nube como en las instalaciones. Aportando mucho más valor y solucionando los requisitos funcionales. Este concepto existe y se llama 'Serverless'.

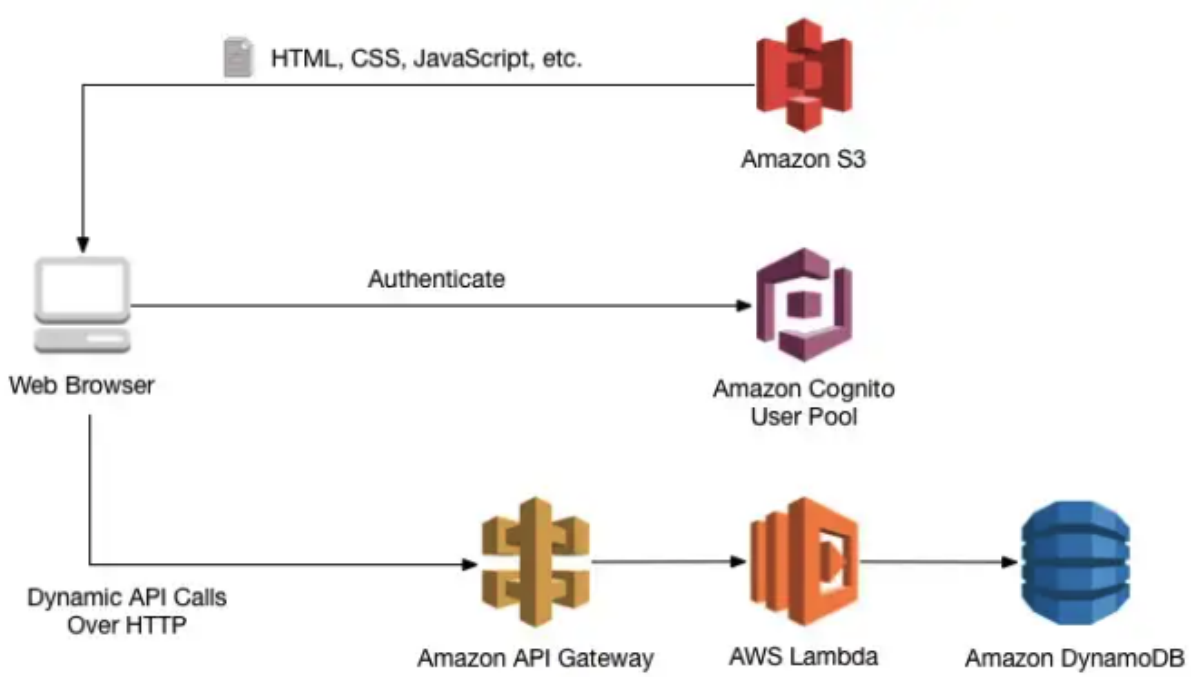
2. Objetivo

En este punto, nuestro objetivo es construir una aplicación totalmente Serveless, es decir, sin tener que preocuparnos de servidores para publicar nuestra aplicación web. Para ello vamos a utilizar Amazon Web Services. Vamos a publicar el proyecto

Vue realizado en la última práctica de la asignatura, sin necesidad de administrar el servidor qué lo hostea, e implementando la API desde 0 en AWS, y de la misma forma, sin necesidad de preocuparnos por los servidores.

3. Tecnologías de la aplicación

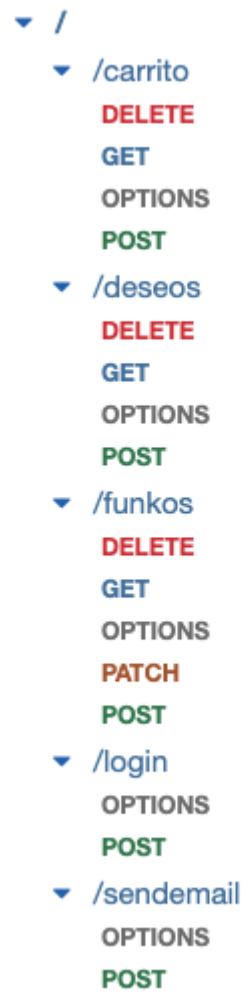
Cómo ya hemos comentado, vamos a utilizar AWS, sin embargo, esta plataforma cuenta con más de 200 servicios, por lo qué tenemos qué diseñar una infraestructura qué sea coherente y sencilla para nuestro objetivo.
Basándonos en las propias recomendaciones de AWS, vamos a utilizar la siguiente arquitectura



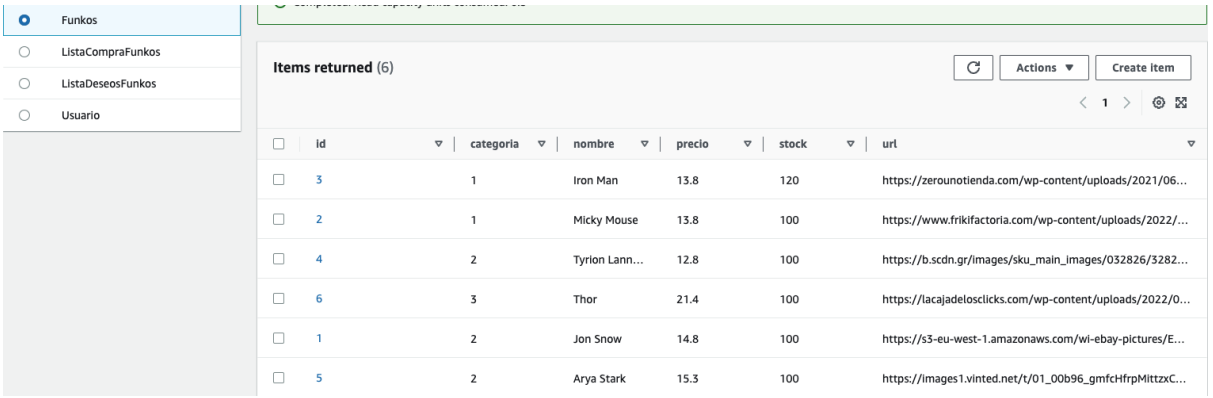
De este diagrama, vamos a obviar la parte de autenticación, ya qué la vamos a realizar de igual forma cómo se pidió en la asignatura, utilizando un token JWT.

Vamos a utilizar S3 para hostear el Front-End. S3 es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes del sector. Este servicio permite crear *'buckets'* para guardar nuestros archivos, y también da la posibilidad de habilitar estos buckets cómo hosting de páginas web estáticas, dándonos un nombre DNS público en Internet.
En cuanto al Back-End, vamos a utilizar el servicio API Gateway, el cual nos permite crear una API de forma sencilla y rápida. Este servicio también nos proporciona un nombre DNS, el cual es al qué realizaremos las llamadas desde el Front-End.

El resultado de la API creada es el siguiente:



Cada vez qué realicemos una llamada a cada método de un recurso de la API, se invocará una función Lambda. AWS Lambda es un servicio informático que permite ejecutar código sin aprovisionar ni administrar servidores. Lambda ejecuta el código en una infraestructura de computación de alta disponibilidad y realiza todas las tareas de administración de los recursos de computación, incluido el mantenimiento del servidor y del sistema operativo, el aprovisionamiento de capacidad y el escalado automático, así como las funciones de registro. Estas funciones Lambda equivalen a los métodos qué creamos en nuestro *'index.js'* del servidor, por lo qué contienen la lógica de nuestra aplicación.
Cómo motor de base de datos vamos a utilizar AWS DynamoDB, un servicio de base de datos NoSQL totalmente administrado que ofrece un rendimiento rápido y predecible, así como una perfecta escalabilidad. DynamoDB le permite delegar las cargas administrativas que supone tener que utilizar y escalar bases de datos distribuidas, para que no tenga que preocuparse del aprovisionamiento, la instalación ni la configuración del hardware, ni tampoco de las tareas de replicación, aplicación de parches de software o escalado de clústeres.



Aquí podemos ver las tablas qué se han creado y los datos de la tabla '*Funkos*'.

Con todo esto terminamos la arquitectura qué habíamos visto en el diagrama, una aplicación totalmente Serveless, en la qué no nos tenemos qué preocupar de disponibilidad y escalabilidad, ya qué es el propio Amazon Web Services el qué garantiza todas estas características.

4. Demo

https://youtu.be/CZbEylw38Tk

5. Conclusiones

Cómo reflexión final, creo qué las aplicaciones Serveless van a ir creciendo en popularidad, hasta el punto en el qué será raro tener servidores administrados por nosotros hosteando nuestras aplicaciones. Este tipo de arquitecturas nos garantizan qué nuestra aplicación va a ser totalmente disponible, no nos vamos a tener qué preocupar de la escalabilidad, ya qué es el propio AWS el qué gestiona todo internamente. Esto dá cómo resultado qué las empresas se centren más en el desarrollo del producto final, sin preocuparse de la plataforma qué lo hostea.

6. Bibliografía

https://docs.aws.amazon.com/es_es/amazondynamodb/latest/developerguide/Introduction.html

https://medium.com/employbl/tutorial-for-building-a-web-application-with-amazon-s3-lambda-dynamodb-and-api-gateway-6d3ddf77f15a
https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html
https://www.tibco.com/es/reference-center/what-is-an-api-gateway#:~:text=Un%20API%20Gateway%20es%20el,API%20para%20proteger%20datos%20valiosos.
https://docs.aws.amazon.com/es_es/AmazonS3/latest/userguide/Welcome.html