

DOCUMENTACION DE USO

PROCESO DE RECLUTAMIENTO: 14042021-2215

Q2 Desarrollador Back Senior

Postulante: David Culebra López

INTRODUCCION

En esta documentación, aprenderá a crear una API REST siguiendo el enfoque sin servidor utilizando AWS Lambda , API Gateway, DynamoDB y Serverless Framework. AWS Lambda es el tercer servicio informático de Amazon. Es muy diferente de los dos servicios informáticos existentes EC2 (Elastic Compute Cloud) y ECS (Elastic Container Service). AWS Lambda es una plataforma informática sin servidor impulsada por eventos que ejecuta su código en respuesta a eventos. Gestiona la infraestructura subyacente escalando hacia arriba o hacia abajo para cumplir con la tasa de eventos. Solo se le cobra por el tiempo que se ejecuta su código. AWS Lambda actualmente admite tiempos de ejecución de lenguajes Java, Python y Node.js.

Requisitos previos

Para seguir este tutorial, necesitará lo siguiente:

- Cuenta de AWS
- Node.js
- AWS CLI y configurarlo

En el siguiente apartado explicare como ejecutarlo para un nivel de usuario, y explicaremos los requisitos:

Cuenta de AWS

Para crearnos una cuenta de amazon (AWS), debemos ingresar al siguiente enlace:

<https://portal.aws.amazon.com/billing/signup#/start>

El cual nos pedirá una cuenta de correo, un celular, y una tarjeta de bancaria (Nota Importante: Debe aceptar pagos en línea). Una vez registrados, ya tenemos la cuenta AWS lista para usar.

Nota: Con estos pasos a seguir, usamos la capa gratuita que nos proporciona AWS

Node.js

Para instalarnos el software de Node.js debemos de dirigirnos a la siguiente pagina web: <https://nodejs.org/es/download/> y elegir el instalador e instalarlo (en nuestro caso elegimos: **Instalador Windows (.msi)** para **x64**, ya que estamos usando Windows 10, con un sistema Operativo de 64 bits.

Una vez instalado el software, debemos abrir una terminal (Tecla Windows + R => y escribir cmd y ejecutar)

Una vez instalado, verificamos las versiones instaladas con los siguientes comandos:

- node -v
- npm -v

Una vez ejecutados dichos comandos, si todo va bien, nos indicara que versión de Node.js y de npm tenemos instaladas.

A continuación, lo que nos toca es instalar el **framework serverless**, con el siguiente comando:

- npm install serverless -g

Una vez instalado, ya tenemos la parte del framework y del Nodejs Listo.

AWS CLI y configurarlo

Es el momento de configurar nuestro usuario de AWS, tendremos que ir al apartado AMI de AWS en la siguiente url:

<https://console.aws.amazon.com/iam/home?region=us-east-1#/home>

Y seguimos estos pasos:

- Nos dirigimos a la parte “Usuarios”, y le damos a “Crear nuevo usuario” o “Añadir nuevo usuario”
- Le ponemos el nombre que deseemos al usuario
- Seleccionamos “Acceso mediante programación”
- “Siguiente Permisos”
- “Asociar directamente las políticas existentes”
- Filtramos las políticas que vamos a usar, y las seleccionamos:
 - AmazonDynamoDBFullAccess
 - AWSLambda_FullAccess
 - AmazonS3FullAccess
 - AmazonAPIGatewayAdministrator
 - AWSLambdaDynamoDBExecutionRole
- Siguiente Etiquetas
- Siguiente Revisar
- Crear Usuario
- **IMPORTANTE:** Copiar en un lugar seguro el **ID de acceso y clave secreta**.

Con esto, ya tenemos configurado el usuario de AWS.

Luego, con la terminal abierta y descargado el “**prueba_culebra_indra.zip**”, la movemos al disco C, y la extraemos en ese directorio (esto ultimo lo podemos hacer con el explorador de Windows, no es necesario hacerlo por el terminal)

Ahora nos movemos al terminal con el siguiente comando

“cd C:\prueba_culebra_indra”

Una vez allí y con el zip descomprimido, ejecutamos el siguiente comando para decirle a serverless que vamos a utilizar las credenciales del usuario creado con anterioridad:

- **serverless config credentials --provider aws --key AKIAIOSFODNN7EXAMPLE --secret wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY**

Nota: debemos cambiar el ID y la clave secreta por la que creamos en el punto anterior.

Debemos instalar la dependencia de nodejs “node-fetch”, si no funciona la api de SWAPI para ejecutar la api de SWAPI, con el siguiente comando:

- **npm install node-fetch**

Una vez hecho esto, ya podemos desplegar el proyecto, con el siguiente comando:

- **sls deploy**

Nos aparecerá una consola de comando con una imagen similar a esta:

```
Serverless: Packaging service...
Serverless: Excluding development dependencies...
Serverless: Uploading CloudFormation file to S3...
Serverless: Uploading artifacts...
Serverless: Uploading service api-indra-culebra.zip file to S3 (55.13 KB)...
Serverless: Validating template...
Serverless: Updating Stack...
Serverless: Checking Stack update progress...
.....
Serverless: Stack update finished...
Service Information
-----
service: api-indra-culebra
stage: dev
region: us-east-1
stack: api-indra-culebra-dev
resources: 20
api keys:
  None
endpoints:
  POST - https://ogahdcdqb3.execute-api.us-east-1.amazonaws.com/dev/insertar
  GET - https://ogahdcdqb3.execute-api.us-east-1.amazonaws.com/dev/obtener
functions:
  insertar: api-indra-culebra-dev-insertar
  obtener: api-indra-culebra-dev-obtener
layers:
  None
Serverless: Removing old service artifacts from S3...
```

Indicando los Puntos de acceso, y las funciones creadas en el proyecto.

FORMAS DE USO:

En este apartado vamos a ver las formas de uso de dicha aplicación, que son las siguientes:

- **Insertar un registro** en una base de datos con el **endpoint de POST**
- **Visualizar un registro** de la base de datos con el **endpoint de GET**
- **Visualizar registros de la API de STAR WARS** de las naves estelares, mediante el **endpoint de GET**

Para este caso hemos usado el programa **Postman**, para verificar que todo esta correctamente realizado.

INSERTAR UN REGISTRO EN DYNAMODB

Con el programa abierto de **Postman**, añadimos la URL **POST**:

<https://ogahdcdqb3.execute-api.us-east-1.amazonaws.com/dev/insertar>

Y le añadimos los siguientes parámetros a enviar

- titulo (tipo Cadena de texto)
- materia_id (tipo Numero)

Procedemos a enviar con el botón **“Send”**

El resultado seria como nuestro a continuación:



VISUALIZAR UN REGISTRO DE DYNAMODB

Con el programa abierto de **Postman**, añadimos la URL **GET**:

<https://ogahdcddb3.execute-api.us-east-1.amazonaws.com/dev/obtener>

Y le añadimos el siguientes parámetro a enviar

- `materia_id` (tipo Numero)

Procedemos a enviar con el botón **“Send”**

El resultado seria como nuestro a continuación:



```
Body Cookies Headers (11) Test Results
Pretty Raw Preview Visualize JSON
1
2  "mensaje": "Datos encontrados",
3  "parametros": {
4    "materia_id": "8"
5  },
6  "busqueda": {
7    "titulo": "titulo1",
8    "materia_id": 8
9  }
10 }
```

VISUALIZAR UN REGISTRO DE SWAPI

Con el programa abierto de **Postman**, añadimos la URL **GET**:

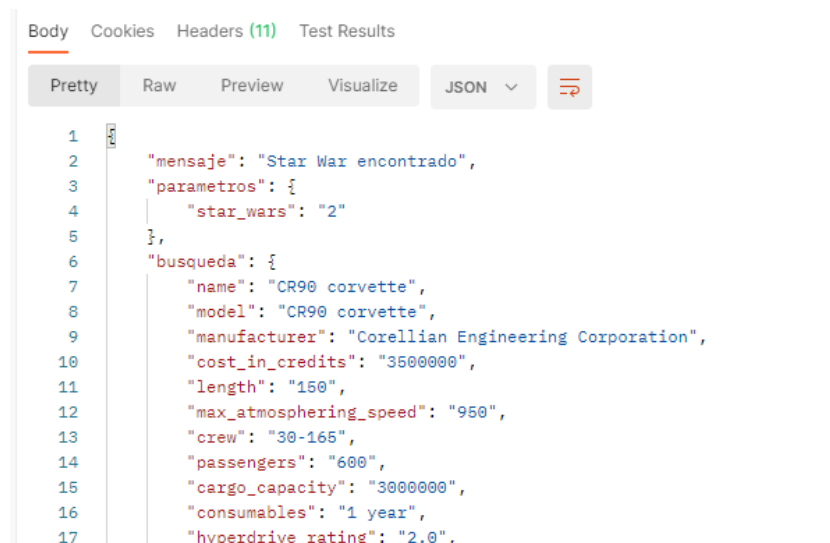
<https://ogahdcddb3.execute-api.us-east-1.amazonaws.com/dev/obtener>

Y le añadimos el siguientes parámetro a enviar

- `star_wars` (Tipo Numero)

Procedemos a enviar con el botón **“Send”**

El resultado seria como nuestro a continuación:



```
Body Cookies Headers (11) Test Results
Pretty Raw Preview Visualize JSON
1
2  "mensaje": "Star War encontrado",
3  "parametros": {
4    "star_wars": "2"
5  },
6  "busqueda": {
7    "name": "CR90 corvette",
8    "model": "CR90 corvette",
9    "manufacturer": "Corellian Engineering Corporation",
10   "cost_in_credits": "3500000",
11   "length": "150",
12   "max_atmosphering_speed": "950",
13   "crew": "30-165",
14   "passengers": "600",
15   "cargo_capacity": "3000000",
16   "consumables": "1 year",
17   "hyperdrive_rating": "2.0".
}
```