UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CS2032 - Cloud Computing (Ciclo 2024-2)
Balanceo de Carga y Alta disponibilidad
Semana 6 - Taller 2: Balanceador de Carga

ELABORADO POR: GERALDO COLCHADO

Contenido

Balanceador de Carga

1. **Objetivo del taller 2**
2. Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo
3. Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción
4. Ejercicio 3: Configurar y probar Balanceador de Carga
5. Ejercicio 4: Diagrama de Arquitectura de Solución
6. Cierre

Objetivo del taller 2:
Balanceador de Carga

- Probar Balanceo de Carga y Alta disponibilidad con Api REST con acceso a base de datos MySQL

Contenido

Balanceador de Carga

1. Objetivo del taller 2
2. **Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo**
3. Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción
4. Ejercicio 3: Configurar y probar Balanceador de Carga
5. Ejercicio 4: Diagrama de Arquitectura de Solución
6. Cierre

Ejercicio 1:
Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo

- **Paso 1:** Cree un repositorio ”api-employees” en github y suba los archivos indicados por el docente, luego ingrese a “MV desarrollo” /home/ubuntu/ y haga git clone.

- **Paso 2:** Analice el Dockerfile y main.py. Modifique el host_name en main.py

- **Paso 3:** Cree la imagen
  $ docker build -t api-employees .

Referencia del api-employees: https://techwasti.com/fastapi-mysql-simple-rest-api-example

Ejercicio 1:
Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo

- **Paso 4:** Suba la imagen a https://hub.docker.com

```
$ docker login -u gcolchado                          (Reemplace amarillo)
$ docker tag api-employees gcolchado/api-employees   (Reemplace amarillo)
$ docker push gcolchado/api-employees                (Reemplace amarillo)
$ docker logout
```

Ejercicio 1:
Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo

- **Paso 5:** Cree la BD y Tabla en MySQL
  Ingrese por ssh a la MV “MV Bases de Datos” y ejecute los 2 contenedores
  $ docker run -d --rm --name mysql_c -e MYSQL_ROOT_PASSWORD=utec -p 8005:3306 -v mysql_data:/var/lib/mysql mysql:8.0
  $ docker run -d --rm --name adminer_c -p 8080:8080 adminer
  Ingrese a adminer y ejecute el script de base de datos entregado por el docente:

```
DROP DATABASE IF EXISTS bd_api_employees;
CREATE DATABASE bd_api_employees CHARSET utf8mb4;
USE bd_api_employees;

CREATE TABLE employees (
   id INT(11) NOT NULL AUTO_INCREMENT,
   name VARCHAR(100) NOT NULL,
   age INT(11) NOT NULL,
   PRIMARY KEY (id)
);

INSERT INTO employees(name, age) VALUES('Jake', 21);
INSERT INTO employees(name, age) VALUES('Mathew', 24);
INSERT INTO employees(name, age) VALUES('Bob', 35);
commit;
```

Contenido

Balanceador de Carga

1. Objetivo del taller 2
2. Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo
3. **Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción**
4. Ejercicio 3: Configurar y probar Balanceador de Carga
5. Ejercicio 4: Diagrama de Arquitectura de Solución
6. Cierre

Ejercicio 2:
Desplegar contenedor api-employees en 2 MV de producción

- **Paso 1:** Ingrese por ssh y ejecute el contenedor en las 2 MV de producción
  $ docker run -d --rm --name api-employees_c -p 8000:8000 gcolchado/api-employees

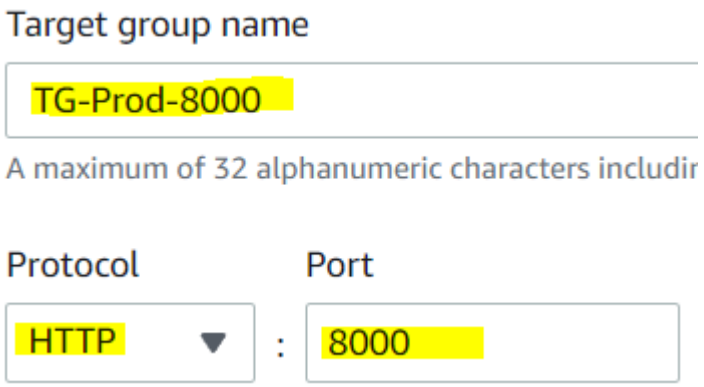
Contenido

Balanceador de Carga

1. Objetivo del taller 2
2. Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo
3. Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción
4. **Ejercicio 3: Configurar y probar Balanceador de Carga**
5. Ejercicio 4: Diagrama de Arquitectura de Solución
6. Cierre

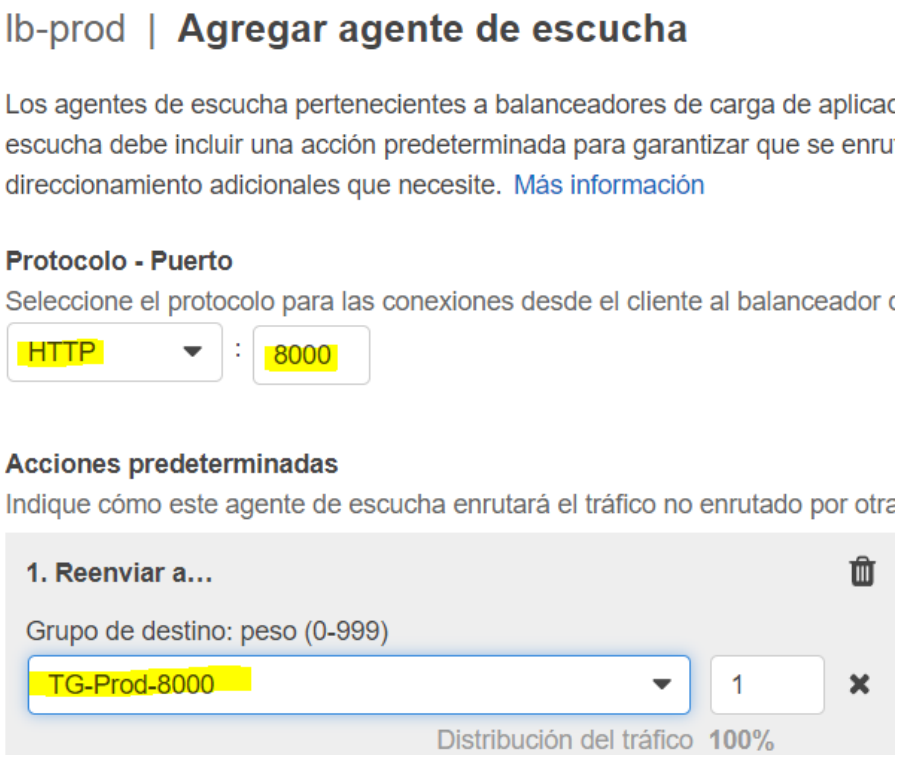
Ejercicio 3:
Configurar y probar Balanceador de Carga

- **Paso 1:** En grupo de seguridad “GS-Prod”, que usan las 2 MV de producción, abra puerto 8000
- **Paso 2:** Crear un Target Group con las 2 MV de producción para el puerto 8000



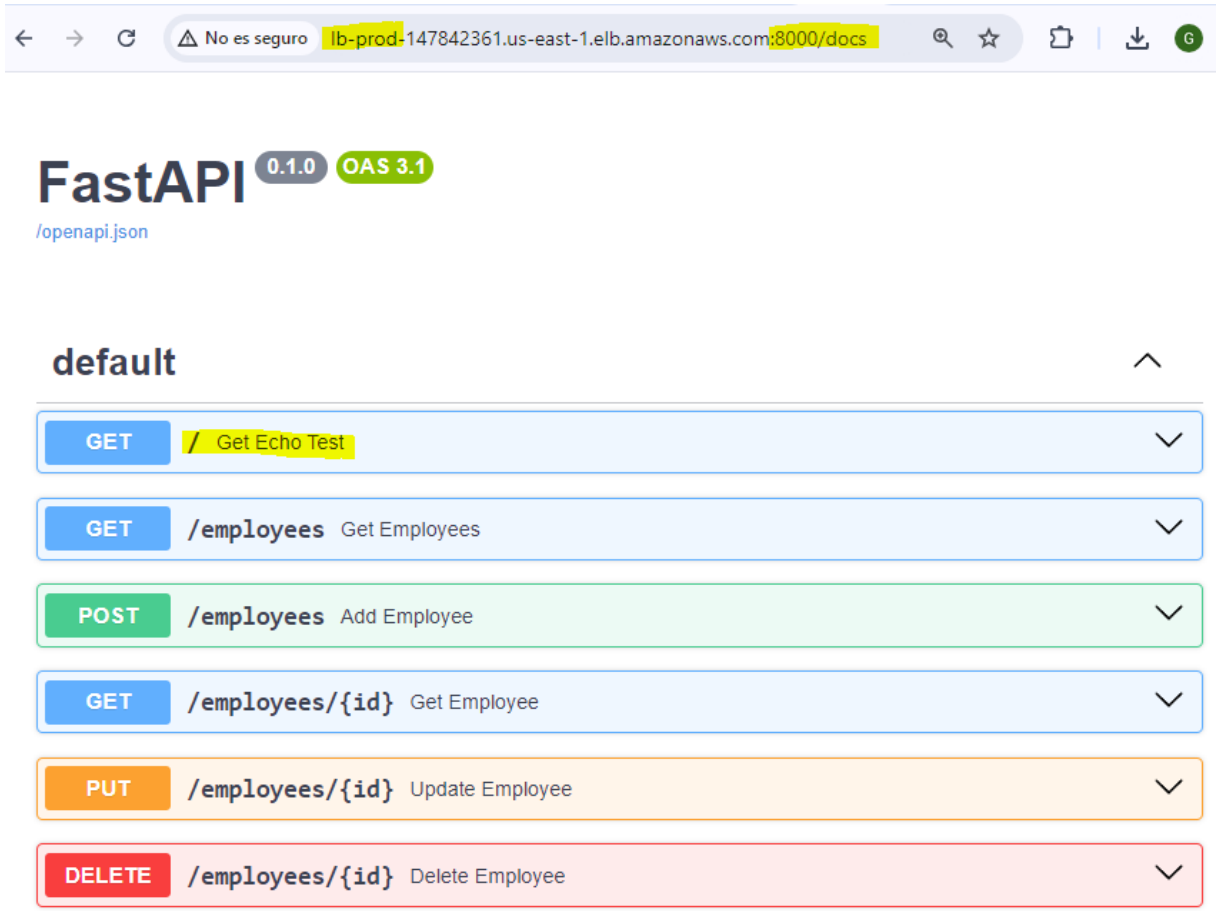
Ejercicio 3:
Configurar y probar Balanceador de Carga

- **Paso 3:** Agregue un agente de escucha en el Balanceador de Carga



Ejercicio 3:
Configurar y probar Balanceador de Carga

- **Paso 4:** Consulte la documentación y **pruebe** el api

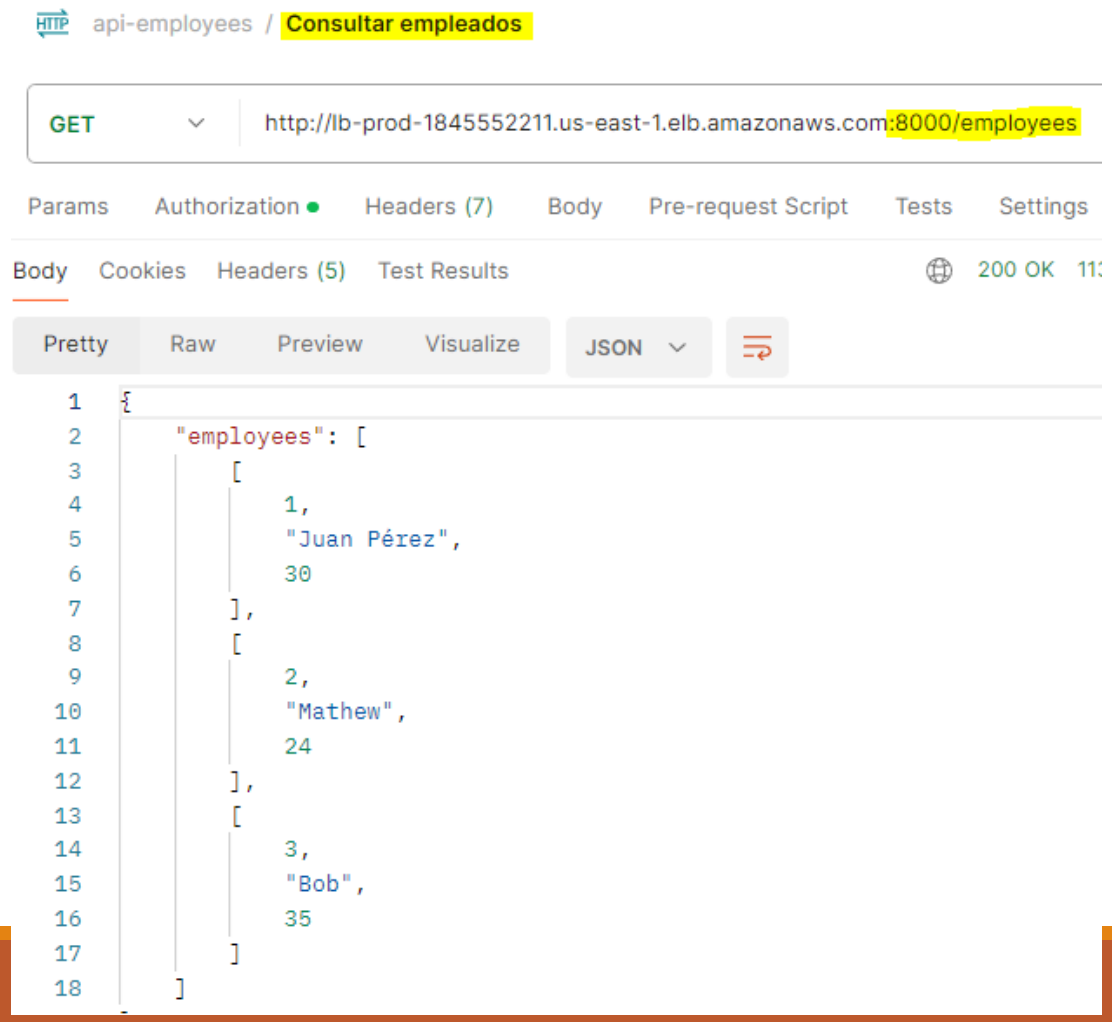


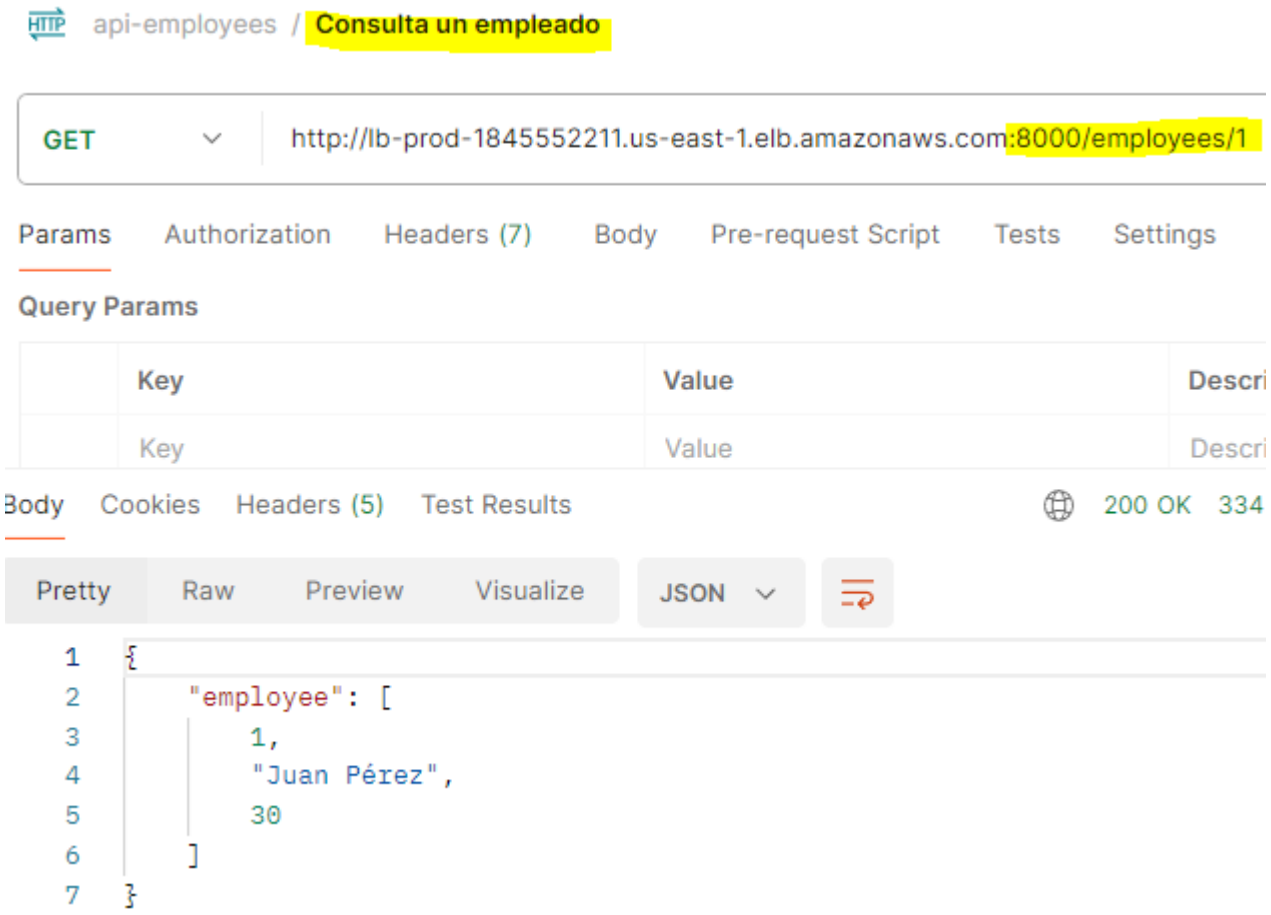
Nota: The **Echo Test** is used to detect whether the service is available

Ejercicio 3:
Configurar y probar Balanceador de Carga

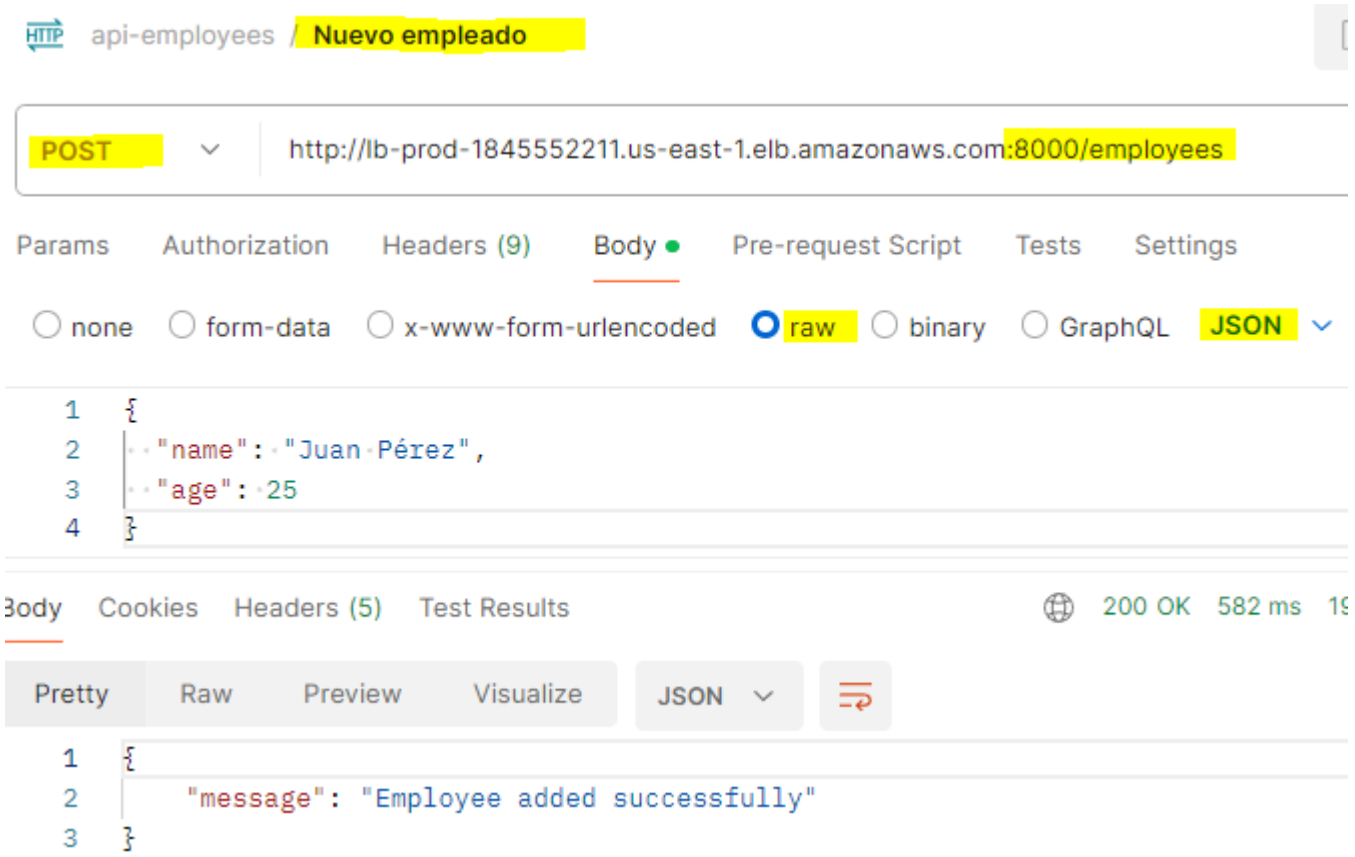
- **Paso 5:** Pruebe también en postman el api-employees con el enlace del balanceador usando la colección postman.

Ejercicio 3:
Configurar y probar Balanceador de Carga

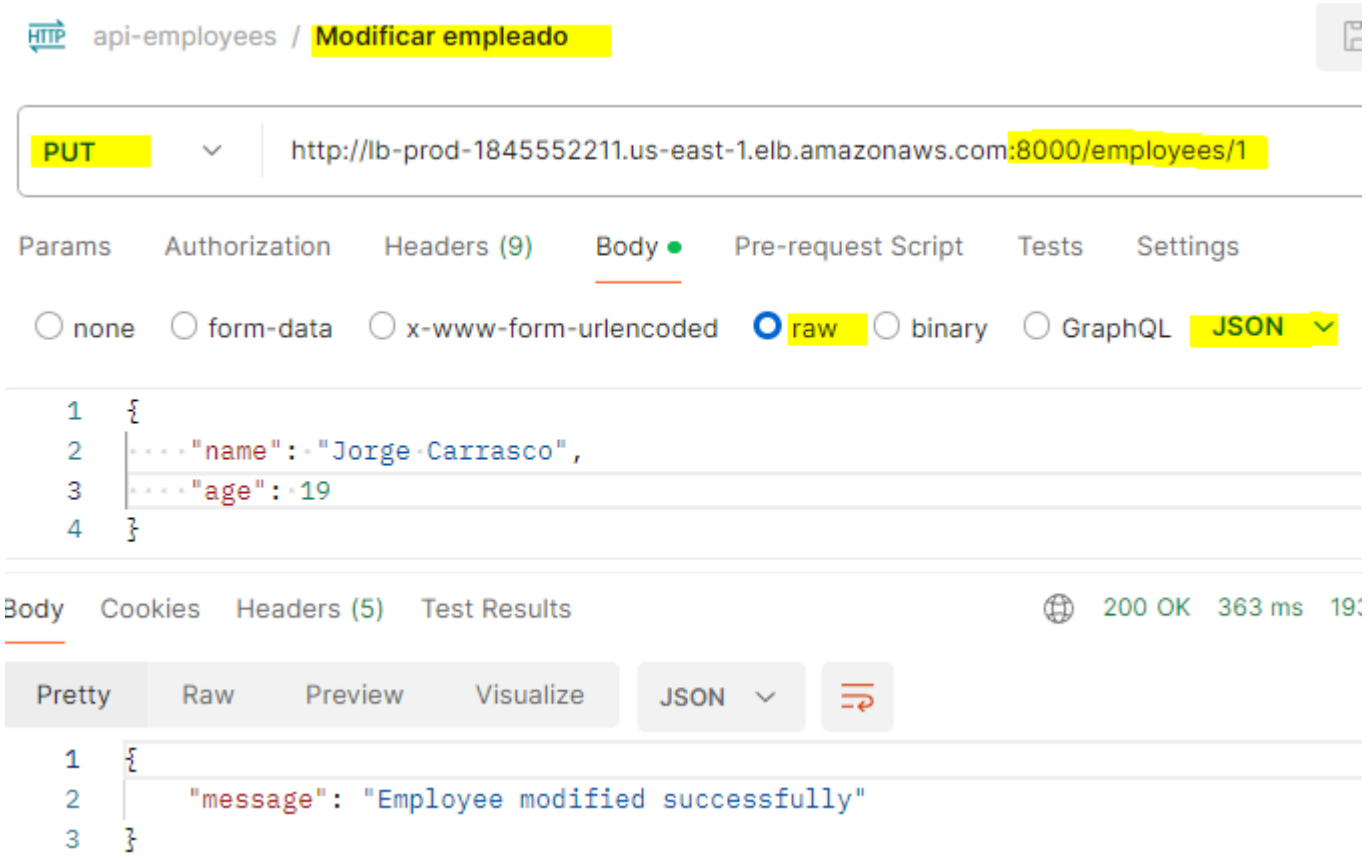




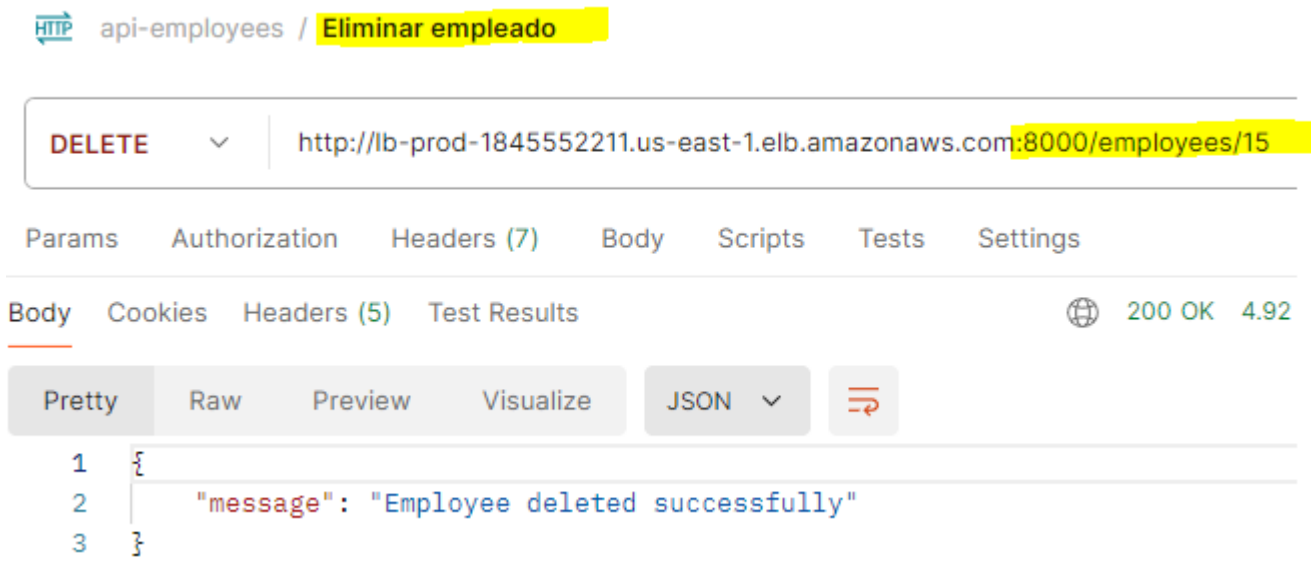
Ejercicio 3:
Configurar y probar Balanceador de Carga



Ejercicio 3:
Configurar y probar Balanceador de Carga



Ejercicio 3:
Configurar y probar Balanceador de Carga



Ejercicio 3:
Configurar y probar Balanceador de Carga

- **Paso 6:** Detener la instancia “MV Prod 1” y probar
- **Paso 7:** Detener la instancia “MV Prod 2” y probar
- **Paso 8:** Iniciar la instancia “MV Prod 1”, ejecutar el contenedor y probar
  $ docker run -d --rm --name api-employees_c -p 8000:8000 gcolchado/api-employees
- **Paso 9:** Iniciar la instancia “MV Prod 2”, ejecutar el contenedor y probar
  $ docker run -d --rm --name api-employees_c -p 8000:8000 gcolchado/api-employees

Contenido

Balanceador de Carga

1. Objetivo del taller 2
2. Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo
3. Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción
4. Ejercicio 3: Configurar y probar Balanceador de Carga
5. **Ejercicio 4: Diagrama de Arquitectura de Solución**
6. Cierre

Ejercicio 4:
Diagrama de Arquitectura de Solución

- Elabore en draw.io el Diagrama de Arquitectura de Solución del Api REST con acceso a base de datos MySQL balanceada en carga usando el puerto 8000. Publique su diagrama en el padlet. Este ejercicio es guiado.

Contenido

Balanceador de Carga

1. Objetivo del taller 2
2. Ejercicio 1: Crear imagen de api-employees con acceso a BD MySQL en MV desarrollo
3. Ejercicio 2: Desplegar contenedor api-employees en 2 MV de producción
4. Ejercicio 3: Configurar y probar Balanceador de Carga
5. Ejercicio 4: Diagrama de Arquitectura de Solución
6. **Cierre**

Cierre:
Balanceador de Carga - Qué aprendimos?

- Balanceo de Carga y Alta disponibilidad con Api REST con acceso a base de datos MySQL

Gracias

Elaborado por docente: Geraldo Colchado