UNIVERSIDAD NACIONAL DE COLOMBIA

Laboratory #3

Logic, Part 2

Camilo Andres Dajer Piñerez
Software Architecture
2023-II

Actividades

i. Requisitos

- Laboratorio #2 finalizado en su totalidad.
- Ver el siguiente video: GraphQL - Big Picture (Architecture).

ii. API Gateway

Componentes

1. Usar los siguientes componentes de software:

- Base de Datos **swarch2023ii_db** (MySQL): puerto *3306*.
- Microservicio **swarch2023ii_ms** (Python + Django + API-REST): puerto *4000*.
- API Gateway **swarch2023ii_ag** (JavaScript + Node.js + API-GraphQL): puerto *5000* (Descargar) (https://drive.google.com/file/d/1436rIEEGMqI4IJ1jnWf1E_exYEPu1l8t/view?usp=sharing).

2. Desplegar el microservicio y la base de datos, de la misma forma como se realizó en los Laboratorios #1 y #2.

3. Desplegar el **API Gateway**. Ubicarse en la raíz del proyecto respectivo y ejecutar los siguientes comandos:

```
docker build -t swarch2023i_ag .
docker run -p 5000:5000 swarch2023i_ag
```

API-GraphQL

Abrir el editor gráfico GraphiQL: **host:5000/graphiql**.

a. Mutations (i)

1. Crear una categoria y retornar su *nombre*:

```
mutation {
  createCategory(category: {
    name: "Categoría Swarch 2023ii",
    description: "Descripción - Categoría Swarch 2023ii"
  }) {
    name
  }
}
```

2. Actualizar la categoría con **id: 1**, y retornar su *id*, *nombre* y *descripción*:

```
mutation {
  updateCategory(id: 1, category: {
    name: "Categoría Swarch 2023ii"
    description: "Nueva Descripción - Categoría Swarch 2023ii"
  }) {
    id
    name
    description
  }
}
```

b. Queries

1. Consultar **únicamente** el *nombre* de todos las categorías:

(Crear categorías adicionales antes de realizar la consulta)

```
query {
  allCategories {
    name
  }
}
```

2. Consultar el *id* y el *nombre* de la categoría con **id: 1**:

```
query {
  categoryById(id: 1) {
    id
```

```
    name
  }
}
```

c. Mutations (ii)

1. Eliminar el usuario con **id: 1**:

```
mutation {
  deleteCategory(id: 1)
}
```

iii. Cola de Mensajes

Desarrollar el siguiente ejercicio: RabbitMQ Tutorial.
(https://www.rabbitmq.com/tutorials/tutorial-one-python.html)

Entrega

Entregable: archivo (en formato *.pdf*) con nombre **l3.pdf**, el cual debe contener:

1. Nombre completo de los integrantes del grupo.
2. Soporte visual del despliegue de los siguientes componentes de software: *swarch2023ii_db*, *swarch2023ii_ms* y *swarch2023ii_ag*.
3. Soporte visual de la ejecución de las peticiones HTTP sobre la API-GraphQL del API Gateway.
4. Soporte visual de las acciones evidenciadas en la base de datos, tras la ejecución de las peticiones sobre el API Gateway.
5. Soporte visual del desarrollo del ejercicio de RabbitMQ.

Forma de Entrega: por medio de la plataforma virtual Moodle.

Fecha de Entrega: Jueves, 14 de septiembre de 2023, antes de las 23:59.

Nota: se debe realizar una única entrega por cada grupo, es decir, solo uno de los integrantes del grupo debe realizar el envío del archivo.