Christian Ariel Palacios Hernández

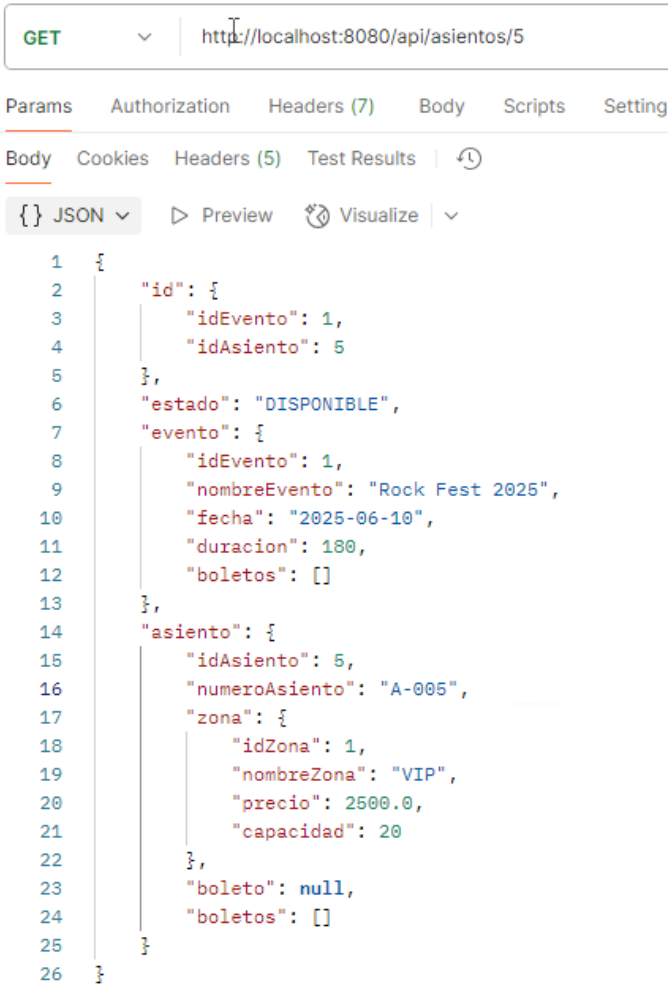
Modulo 9 Proyecto Final

Capturas de pantalla del funcionamiento.

AsientoRestController

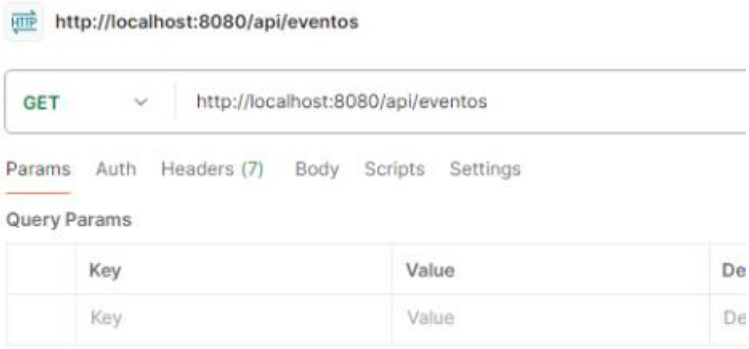
- Obtener Asientos Disponibles por Zona y Evento

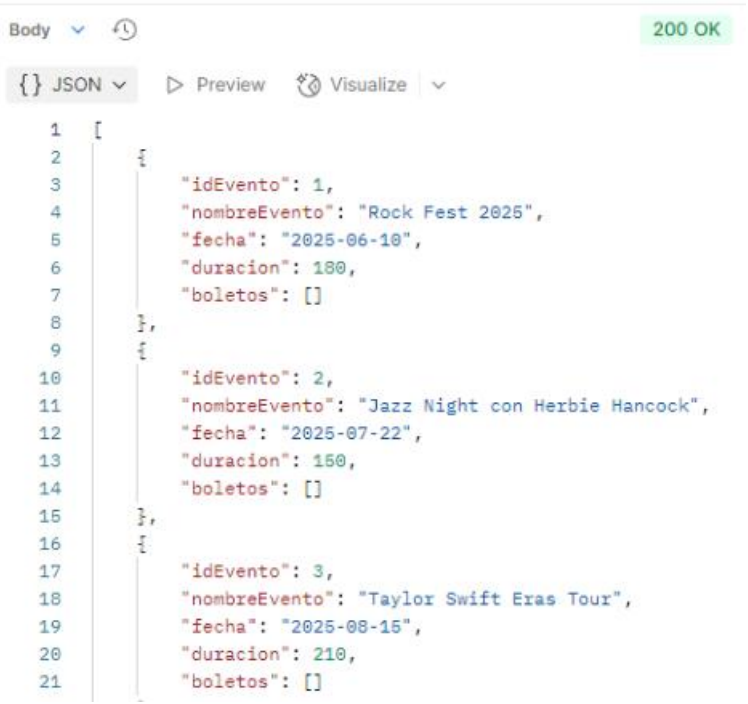
- Obtener un AsientoEvento por idAsiento



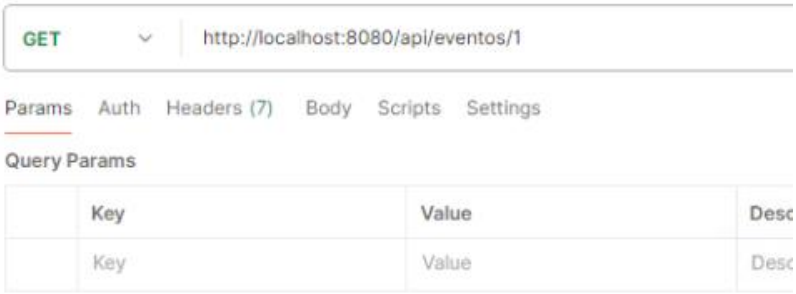
EventoRestController

- Obtener todos los eventos

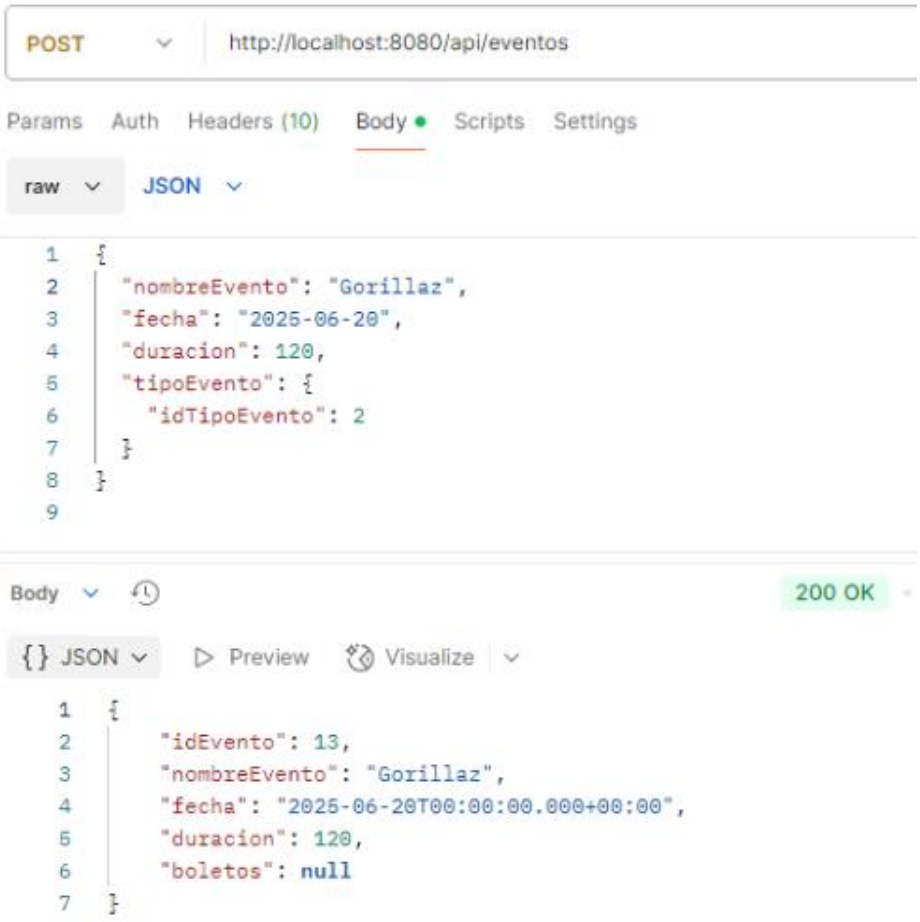




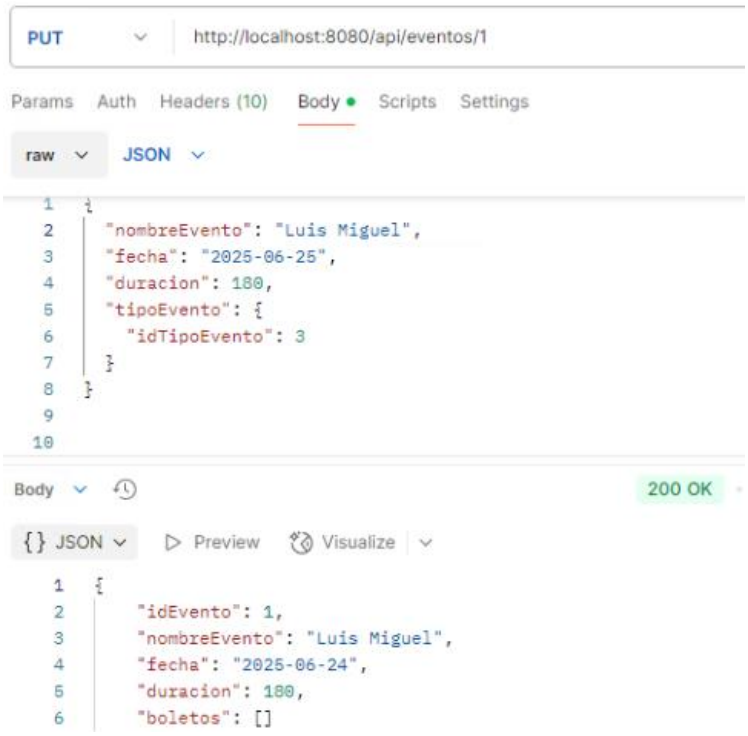
- Obtener un evento por ID



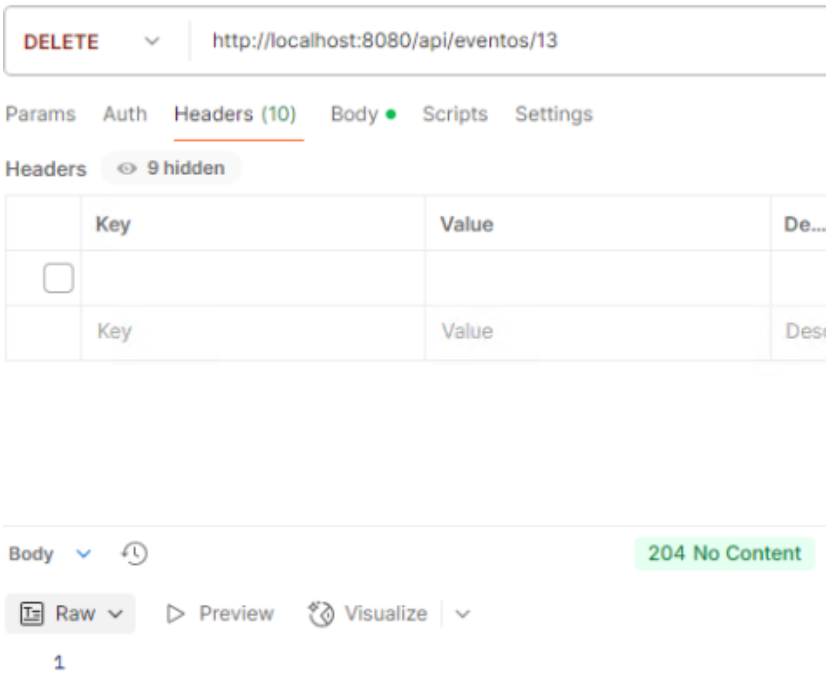
- Crear un evento



- Actualizar un evento

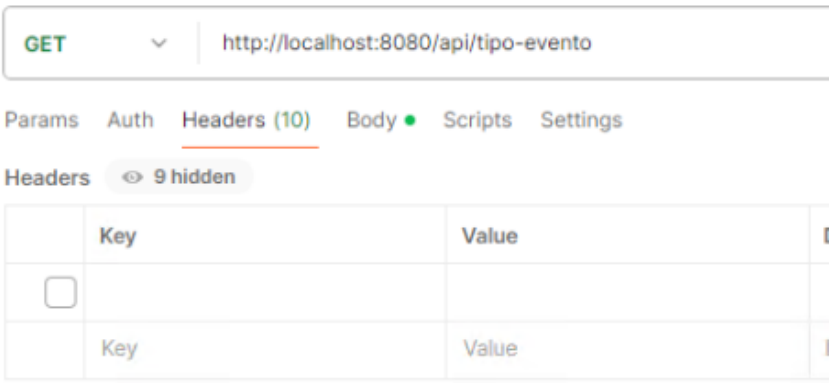


- Eliminar un evento

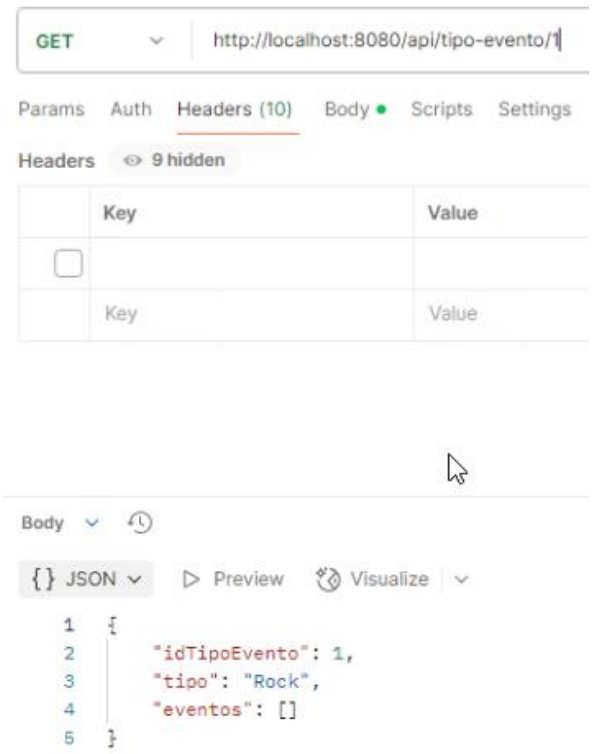


TipoEventoRestController

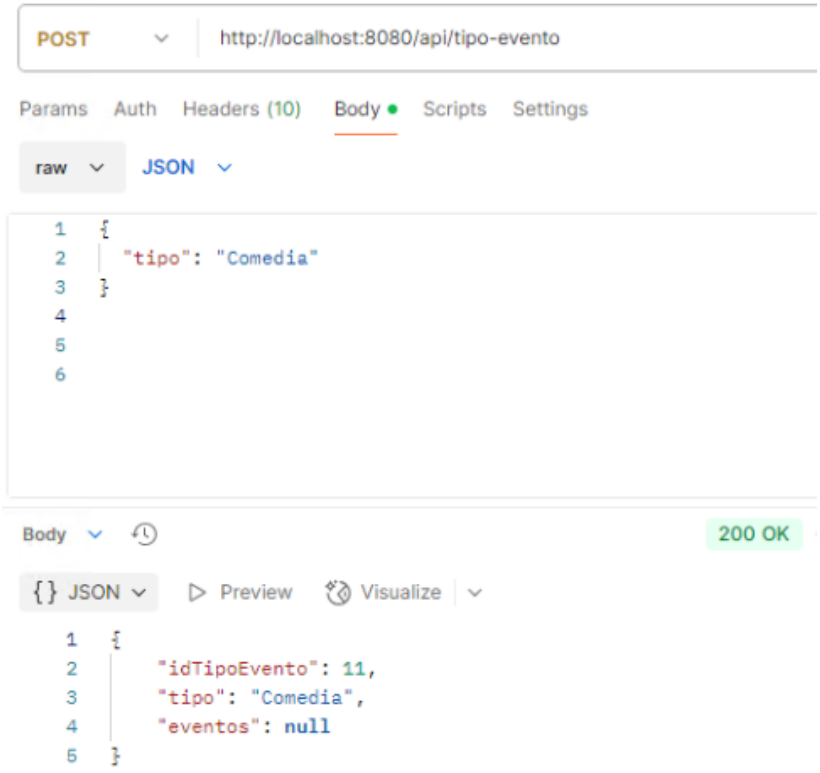
- Obtener todos



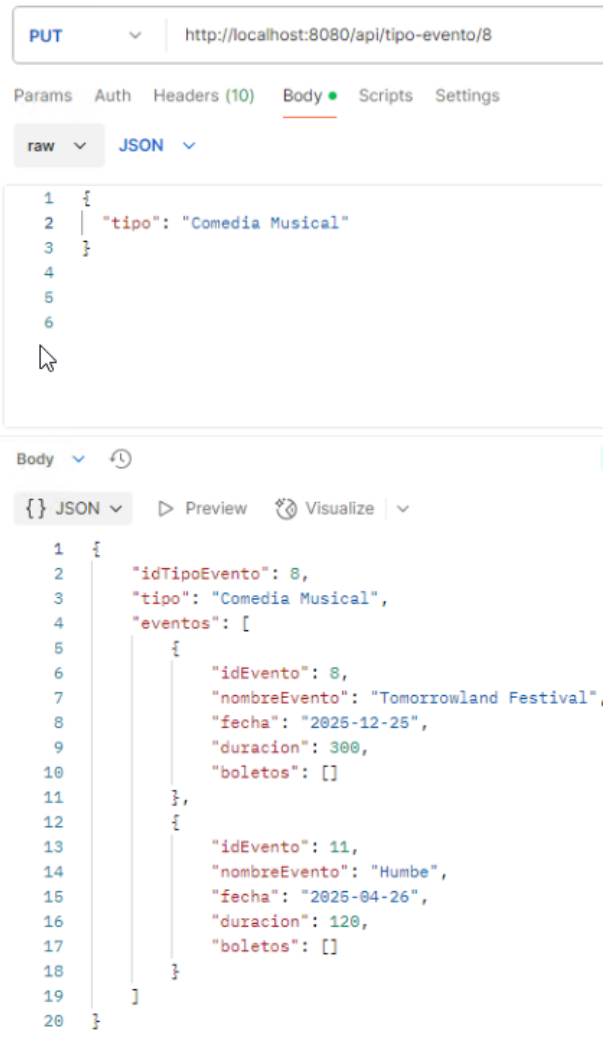
- Obtener por ID



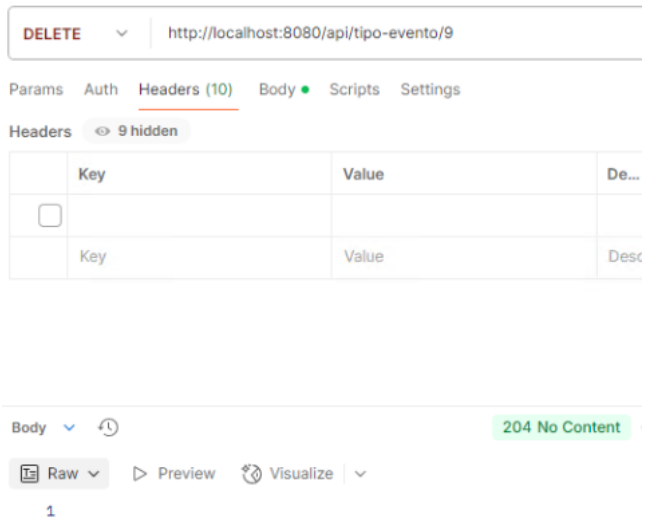
- Crear nuevo



- Actualizar

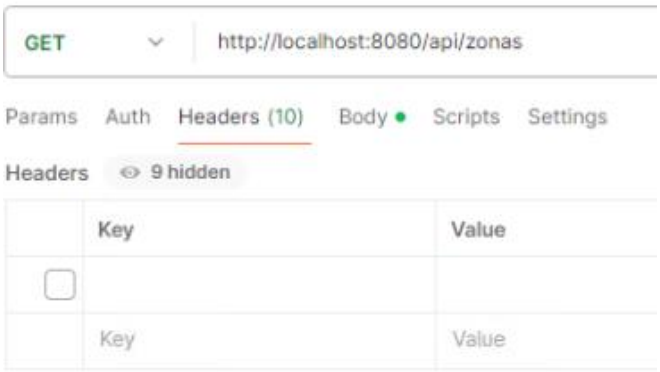


- Eliminar

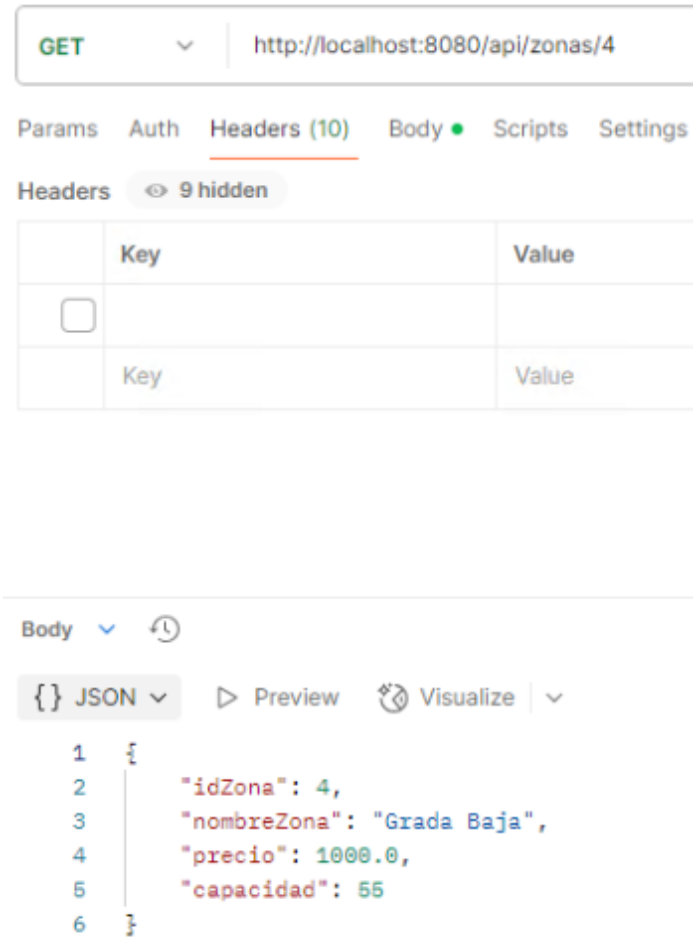


ZonaRestController

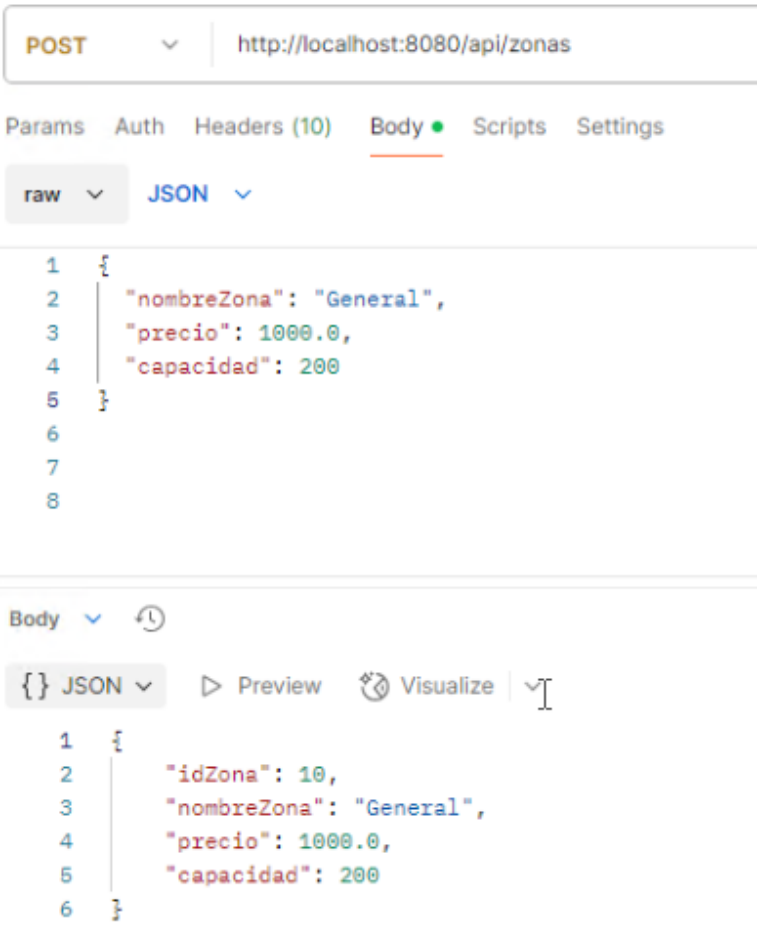
- Obtener todas las zonas



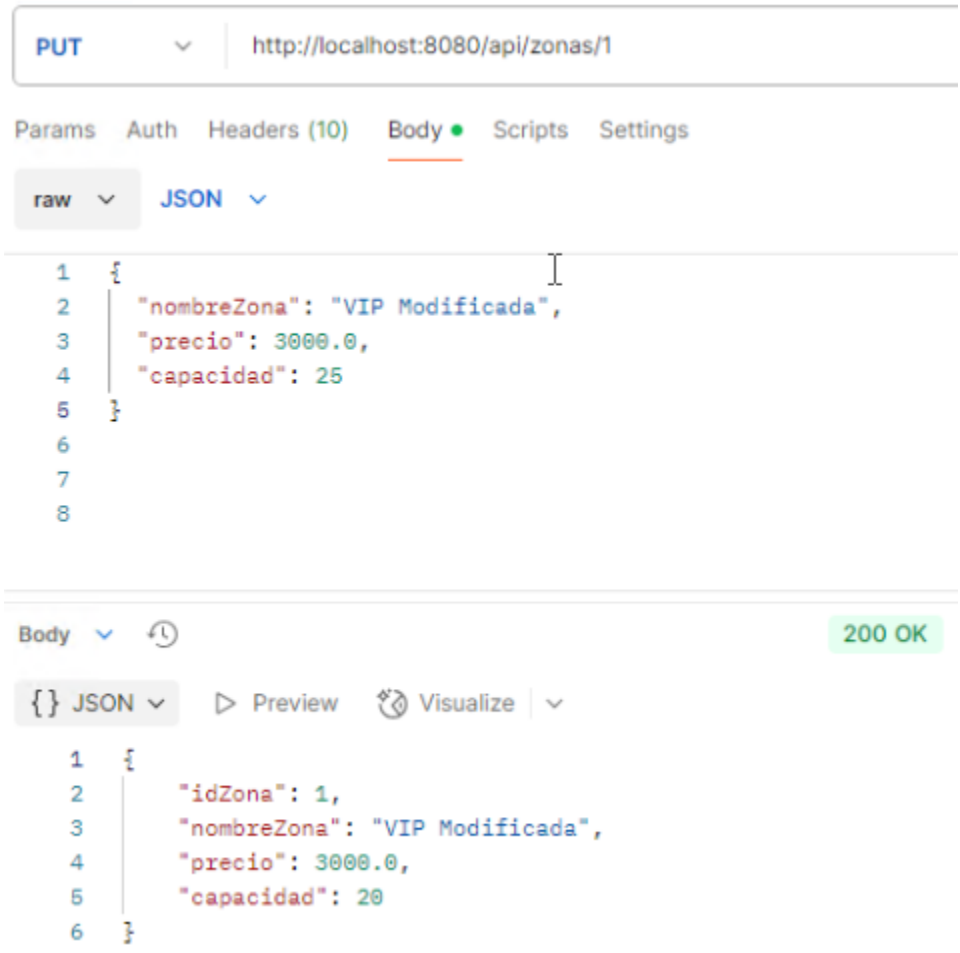
- Obtener zona por ID



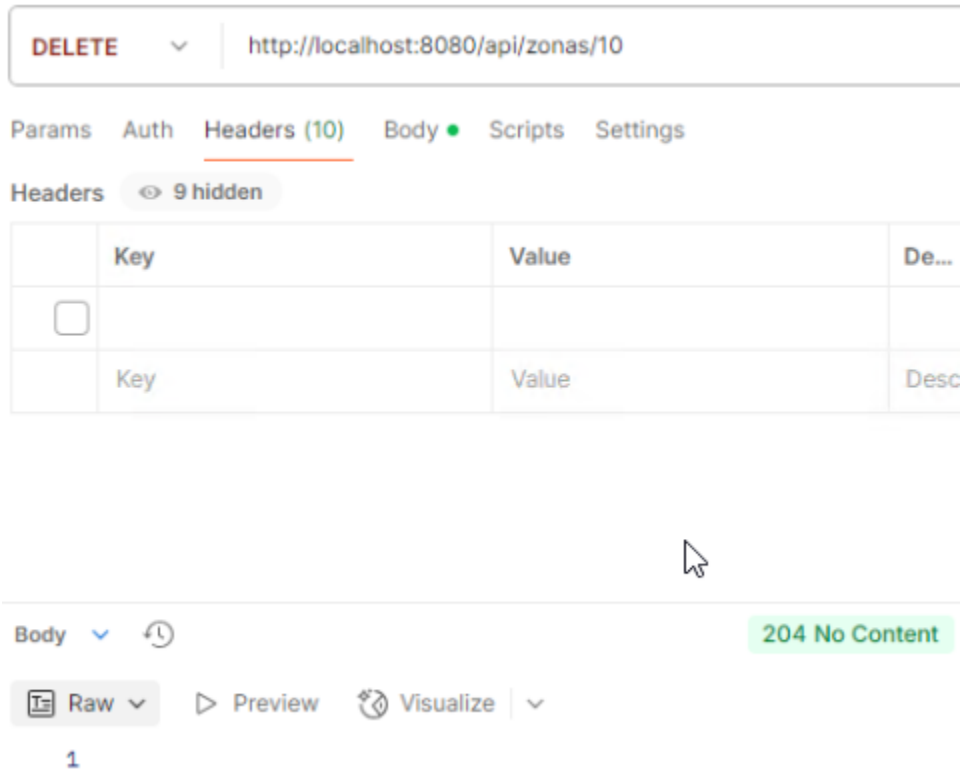
- Crear nueva zona



- Actualizar zona



- Eliminar zona



Configuraciones necesarias para la aplicación de consumo.

Para que la aplicación RESTful funcione correctamente y se pueda consumir desde herramientas como Postman, o un cliente front-end, se realizaron las siguientes configuraciones:

- Dependencias en pom.xml
  Se añadieron las dependencias necesarias para construir una aplicación con Spring Boot

```
<!-- Spring Web para crear controladores REST -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<!-- Spring Data JPA para el acceso a base de datos -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

```
<!-- Driver de base de datos (MariaDB en este caso) -->
<dependency>
  <groupId>org.mariadb.jdbc</groupId>
  <artifactId>mariadb-java-client</artifactId>
  <version>3.4.1</version>
</dependency>
```

- Configuración en application.properties

```
# Puerto del servidor
server.port=8080

# Conexión a la base de datos MariaDB
spring.datasource.url=jdbc:mariadb://localhost:3306/prototype3
spring.datasource.username=tu_usuario
spring.datasource.password=tu_contraseña

# Configuración JPA / Hibernate
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect
```

- Controladores REST habilitados
  Todos los controladores REST están anotados con @RestController y manejan rutas del tipo /api/
  sto permite consumir los datos desde clientes externos o Postman accediendo a rutas como:
  - GET http://localhost:8080/api/eventos
  - POST http://localhost:8080/api/zonas

- ScriptSQL
  Se pasará el script con las sentencias SQL y todo lo necesario para la conexión que se hará con el POSTMAN y con la información necesaria

Como acceder a la sección del consumo del RESTful.

A continuación, se explica paso a paso cómo consumir la API RESTful desarrollada con Spring Boot utilizando Postman, una herramienta popular para realizar peticiones HTTP y probar servicios web:

1. Iniciar la aplicación Spring Boot

Antes de consumir cualquier endpoint REST, asegúrate de que la aplicación esté corriendo en el puerto configurado (por defecto 8080). Ejecuta el proyecto desde tu IDE (IntelliJ, Eclipse, etc.)

2.- Abrir Postman

3.- Acceder a los endpoints REST

En los ejemplos que se realizaron en la primera parte de este documento se pueden ver los diferentes endpoints para cada uno de los estos ejemplos.

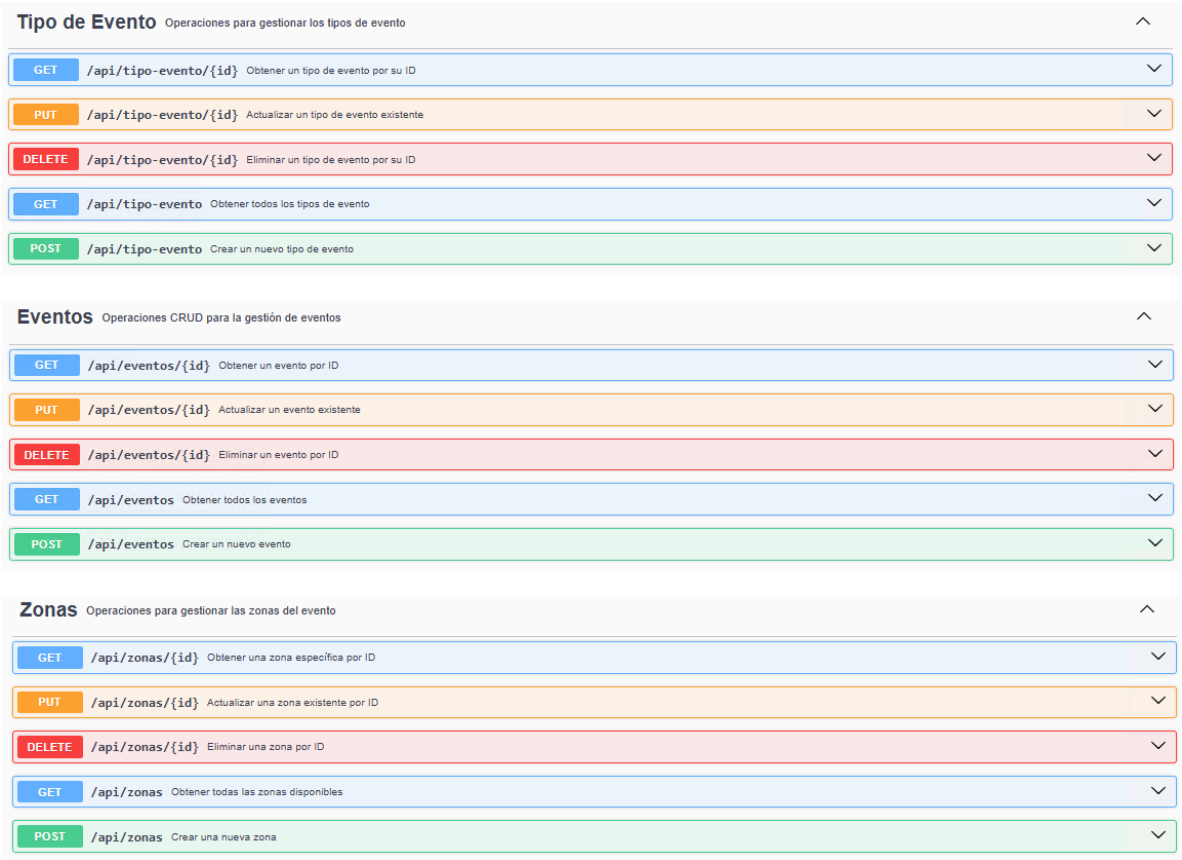
Documentación en Swagger

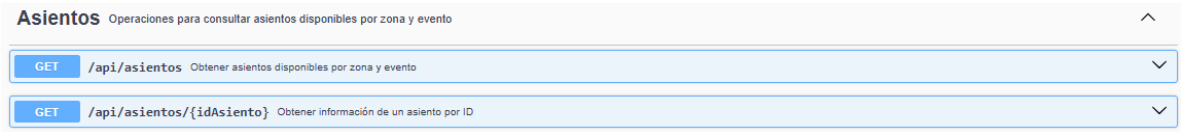
Una vez ejecutada la aplicación, la interfaz de Swagger se encuentra disponible en la siguiente URL:

http://localhost:8080/swagger-ui/index.html

Se aplicaron las siguientes anotaciones para enriquecer la documentación de Swagger:

- @RestController y @RequestMapping: delimitan los recursos y rutas.
- @Tag: describe el propósito del grupo de endpoints.
- @Operation: indica el objetivo de cada operación HTTP (GET, POST, PUT, DELETE).
- @Parameter: documenta los parámetros de entrada con descripción y ejemplos.




Asientos
Operaciones para consultar asientos disponibles por zona y evento
GET
/api/asientos
Obtener asientos disponibles por zona y evento
GET
/api/asientos/{idAsiento}
Obtener información de un asiento por ID