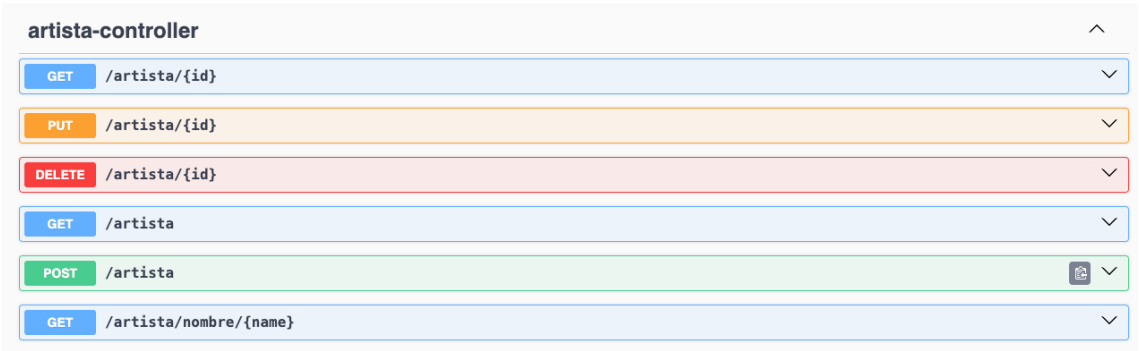
Proyecto 3 — API REST con Spring Boot

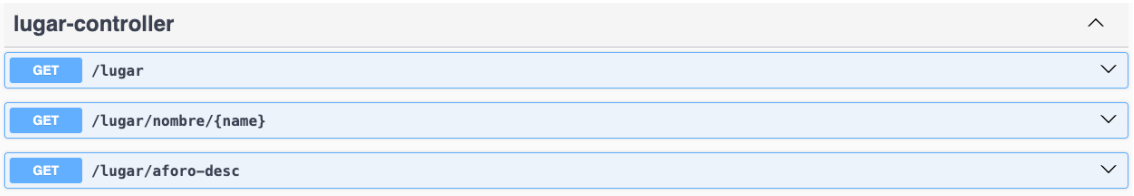
- Justificación de la nueva tabla.
  Creo la tabla Entrada para añadir más sentido a la base de datos, ya que es necesario para este proyecto.
  Así realizaba el filtro con los precios.

- Tabla de EndPoints

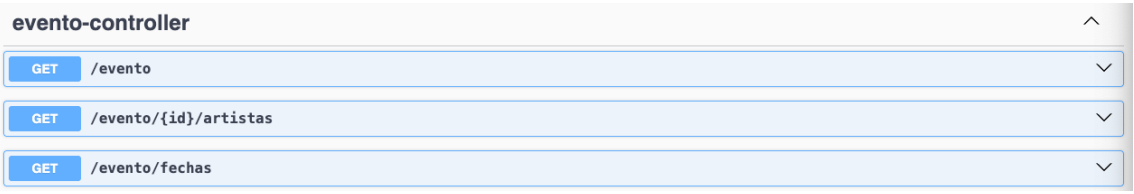
  ArtistaController



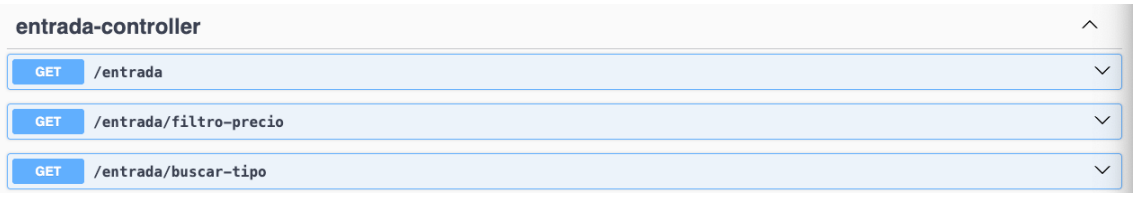
  LugarController



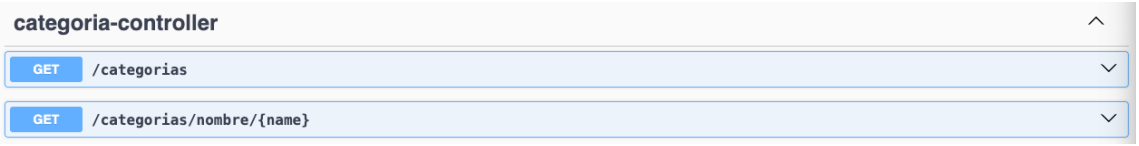
  EventoController



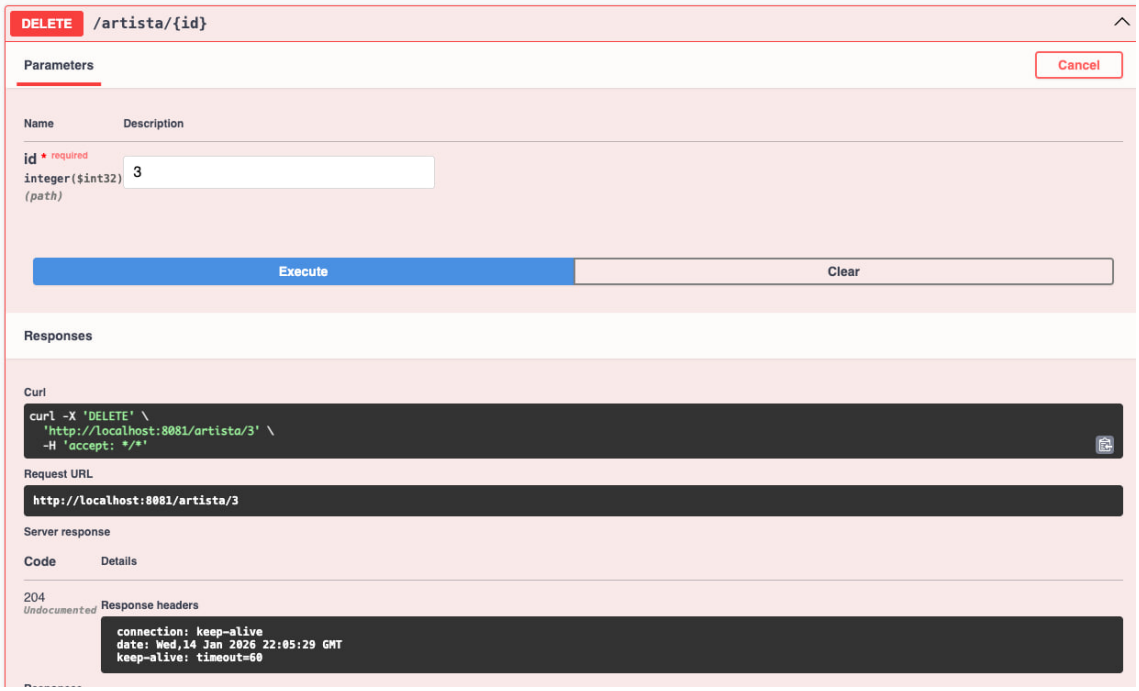
  EntradaController



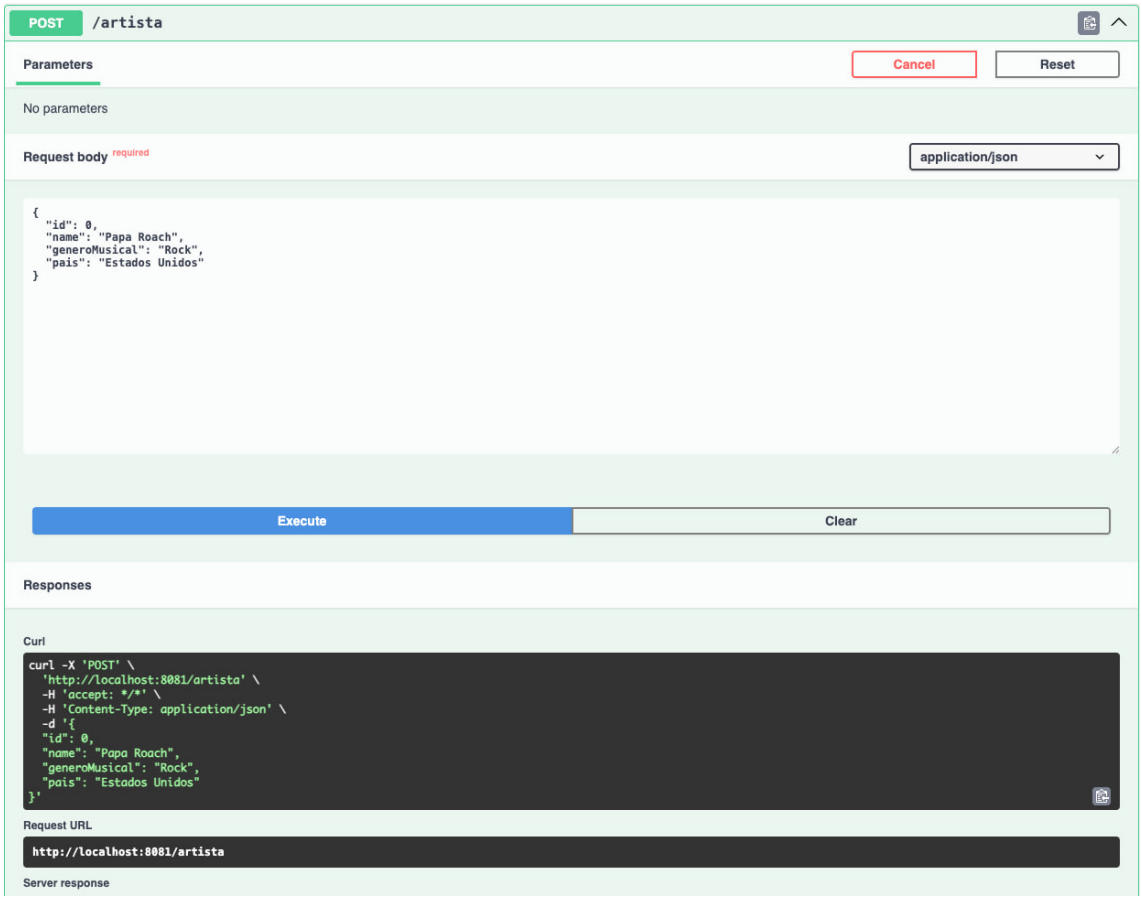
CategoriaController

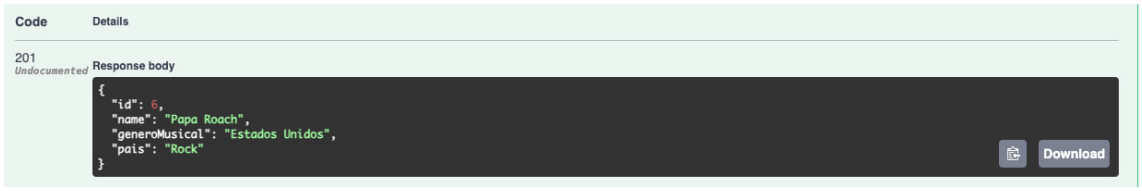


- Ejemplos de peticiones/respuestas.

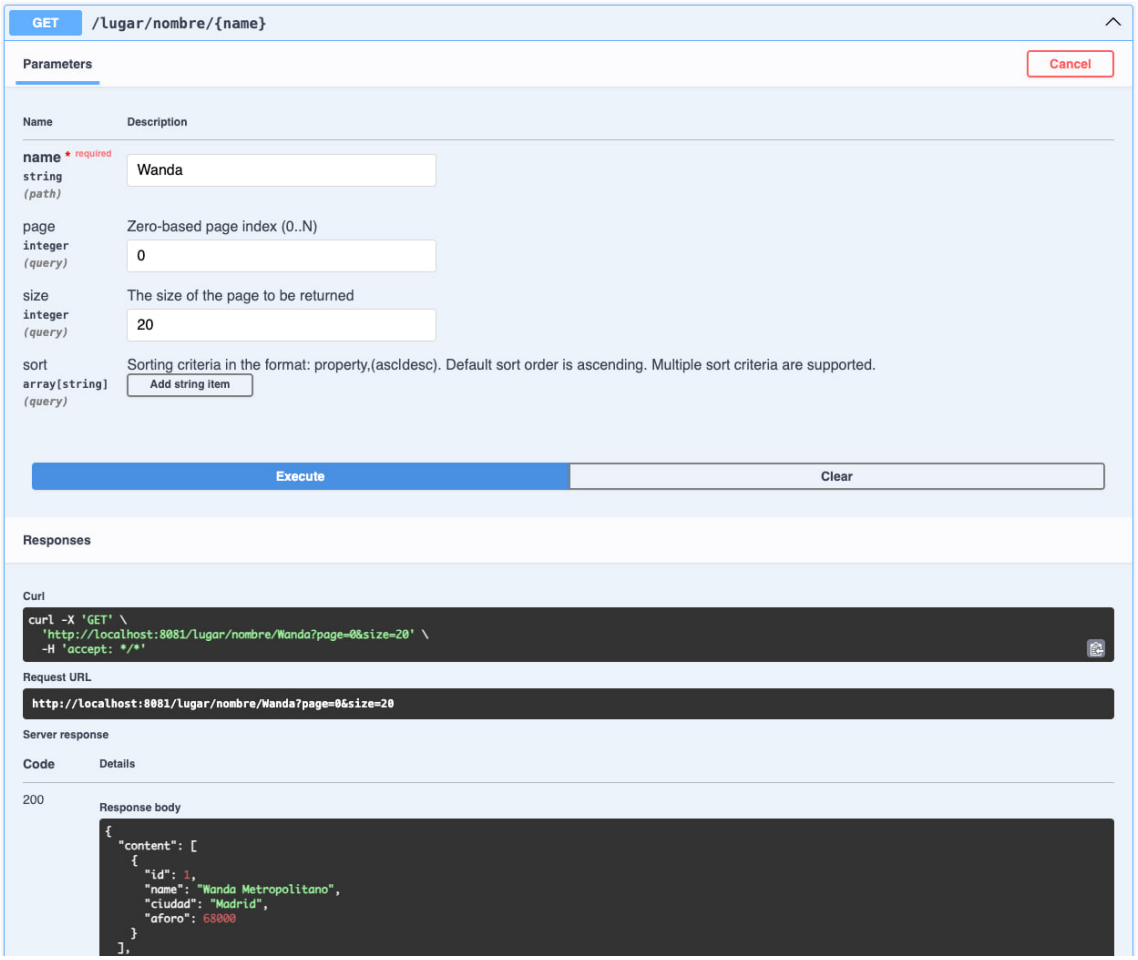


id_artista	nombre	genero_musical	pais
1	C. Tangana	Urbano/Flamenco	España
2	Dua Lipa	Pop	Reino Unido
4	The Grizzy	Indie Rock	Canadá
5	Bizarrap	EDM/Trap	Argentina
NULL	NULL	NULL	NULL

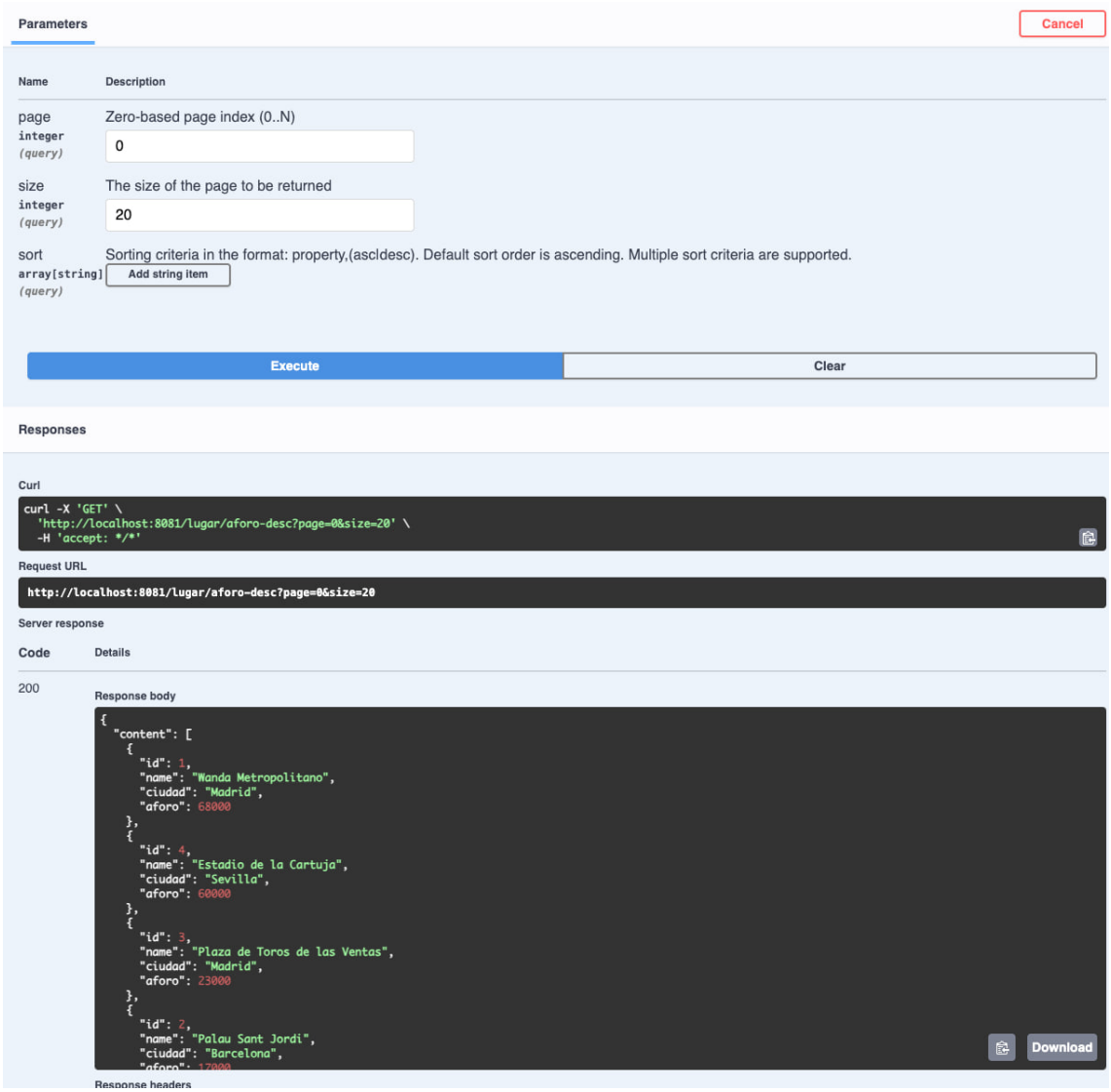




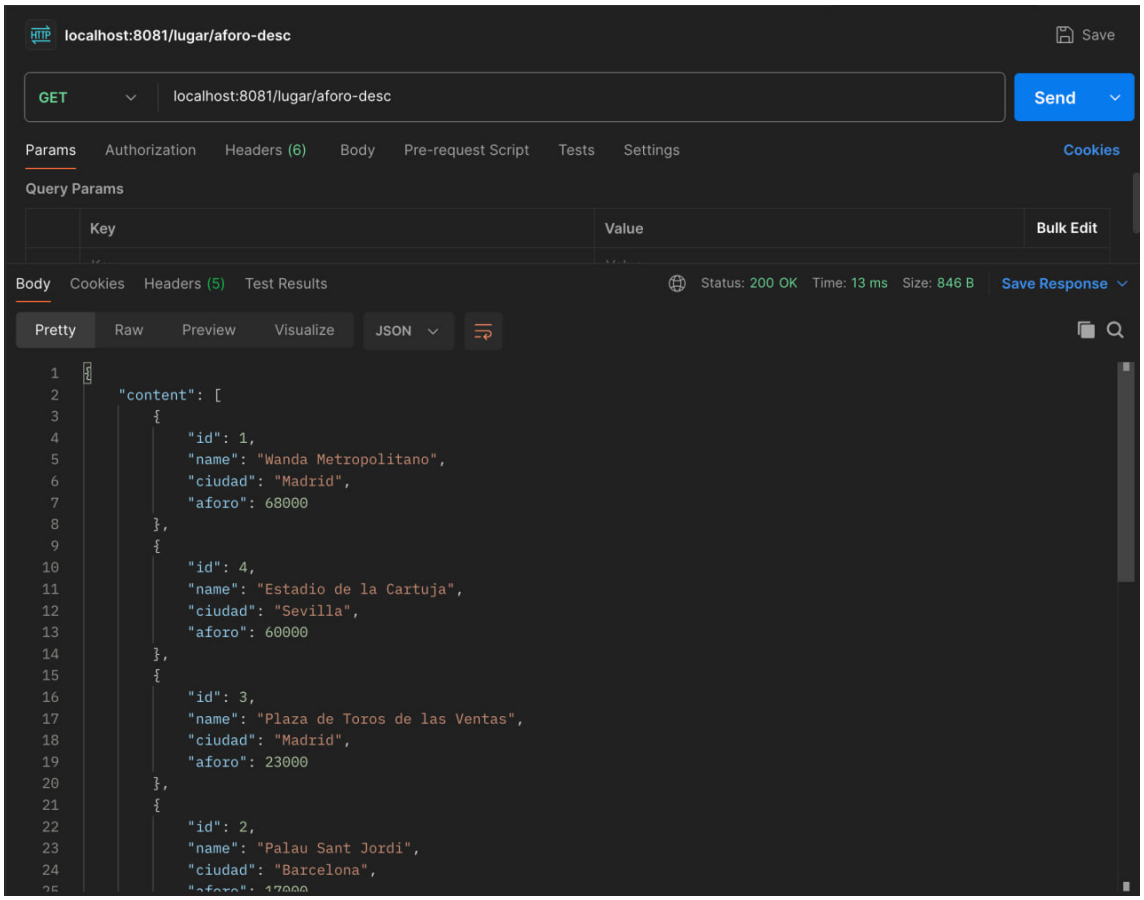
id_artista	nombre	genero_musical	pais
1	C. Tangana	Urbano/Flamenco	España
2	Dua Lipa	Pop	Reino Unido
4	The Grizzy	Indie Rock	Canadá
5	Bizarrap	EDM/Trap	Argentina
6	Papa Roach	Rock	Estados Unidos
NULL	NULL	NULL	NULL


GET
/lugar/nombre/{name}
Parameters
Cancel
Name
Description
name * required
string
(path)
Wanda
page
integer
(query)
Zero-based page index (0..N)
0
size
integer
(query)
The size of the page to be returned
20
sort
array[string]
(query)
Sorting criteria in the format: property,(asc|desc). Default sort order is ascending. Multiple sort criteria are supported.
Add string item
Execute
Clear
Responses
Curl
curl -X 'GET' \
'http://localhost:8081/lugar/nombre/Wanda?page=0&size=20' \
-H 'accept: */*'
Request URL
http://localhost:8081/lugar/nombre/Wanda?page=0&size=20
Server response
Code
Details
200
Response body
{
"content": [
{
"id": 1,
"name": "Wanda Metropolitano",
"ciudad": "Madrid",
"aforo": 68000
}
],

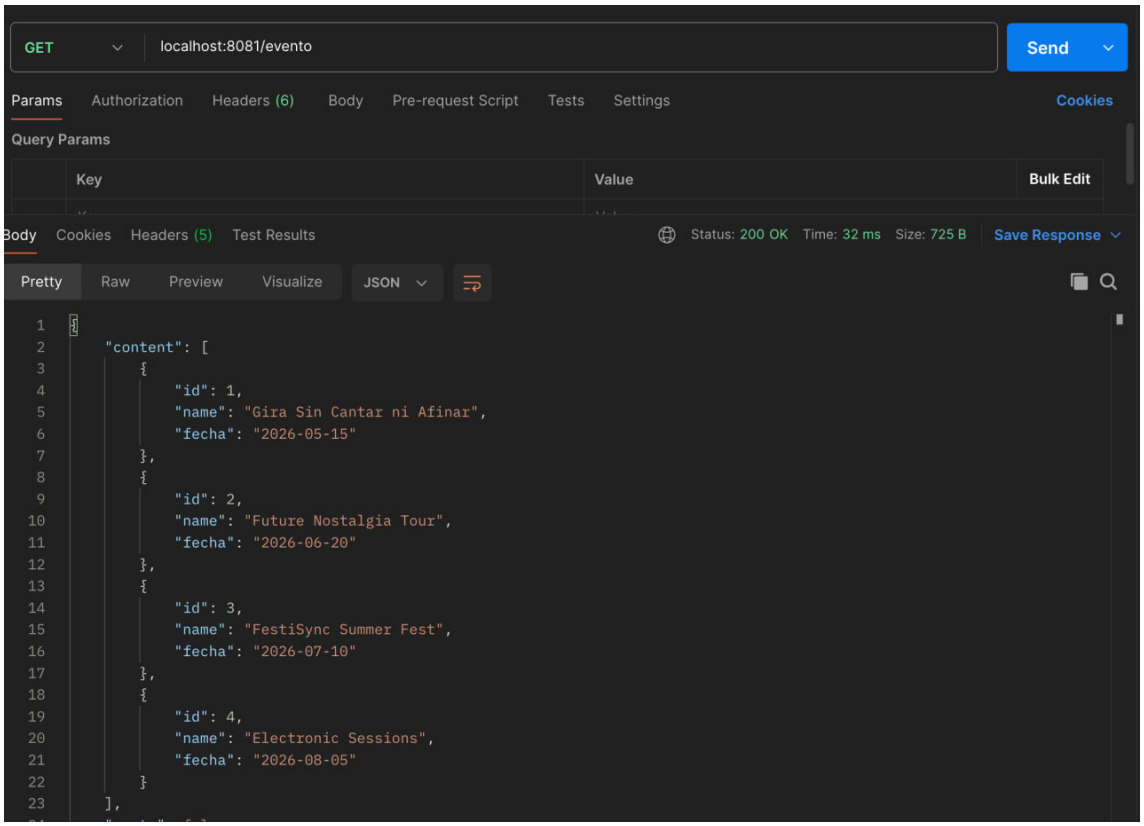
Aforo descendente



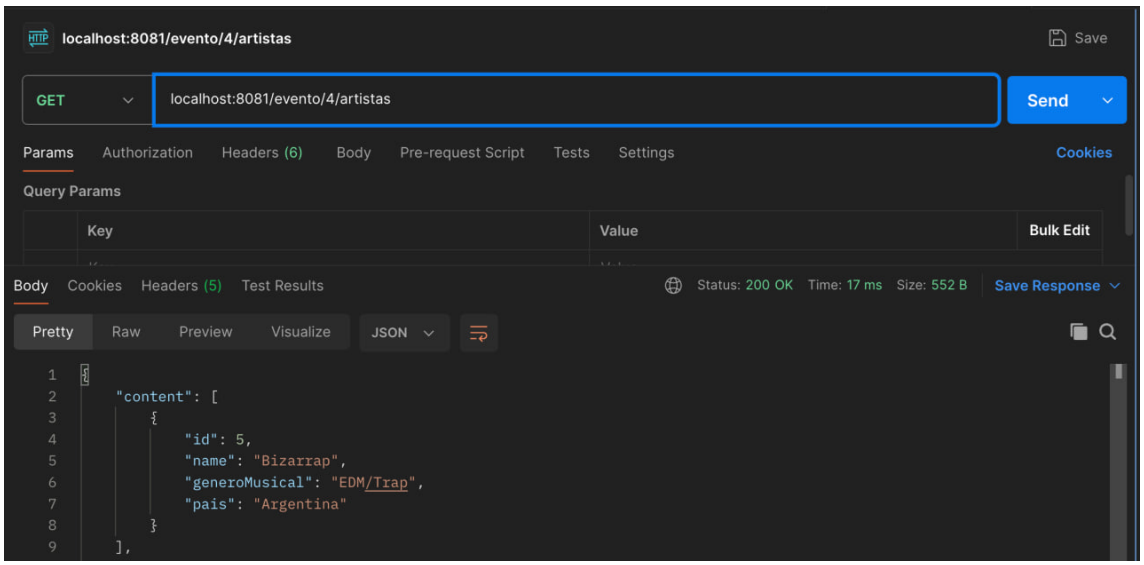
Comprobación en Postman.



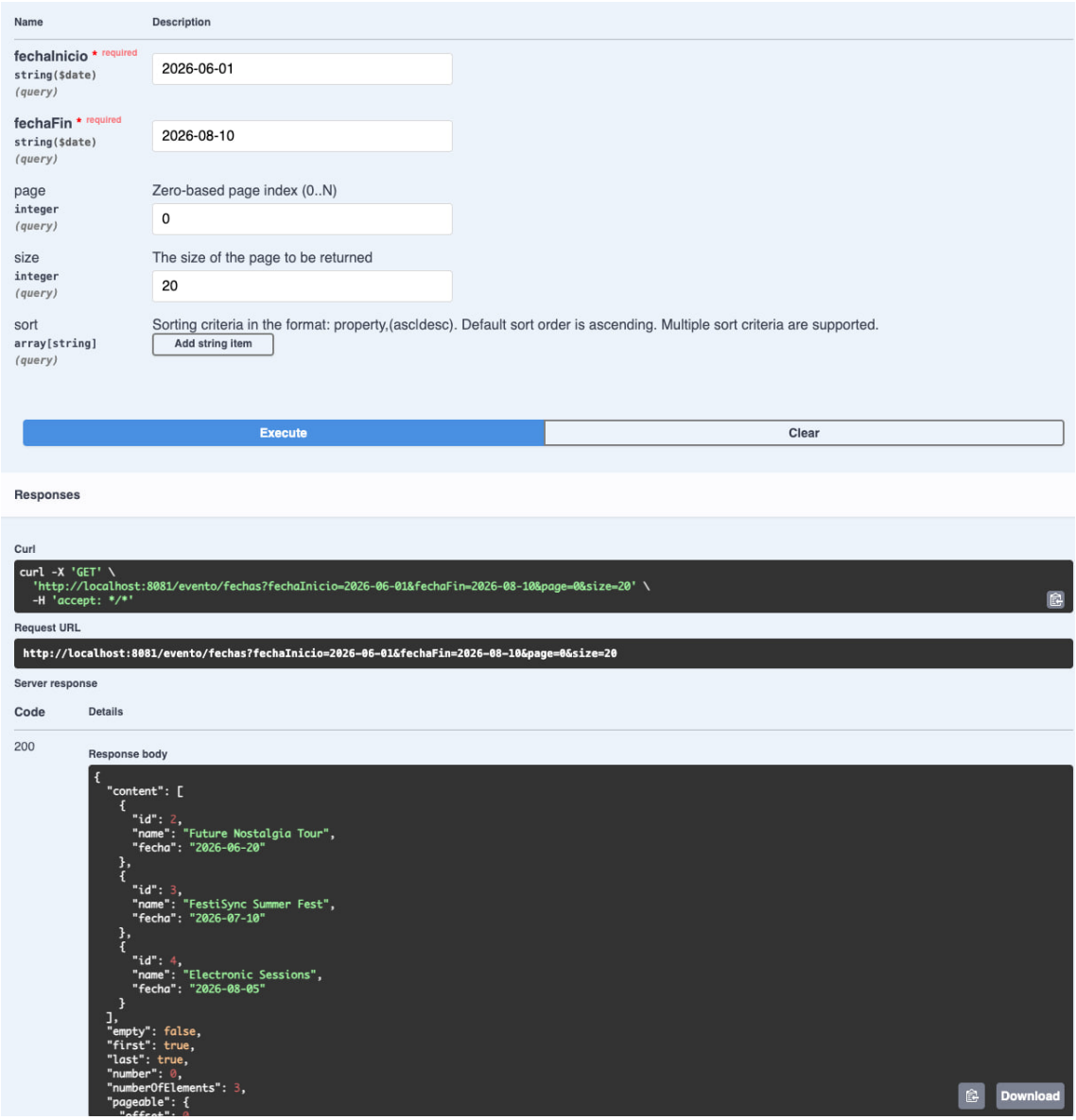
EVENTO Postman



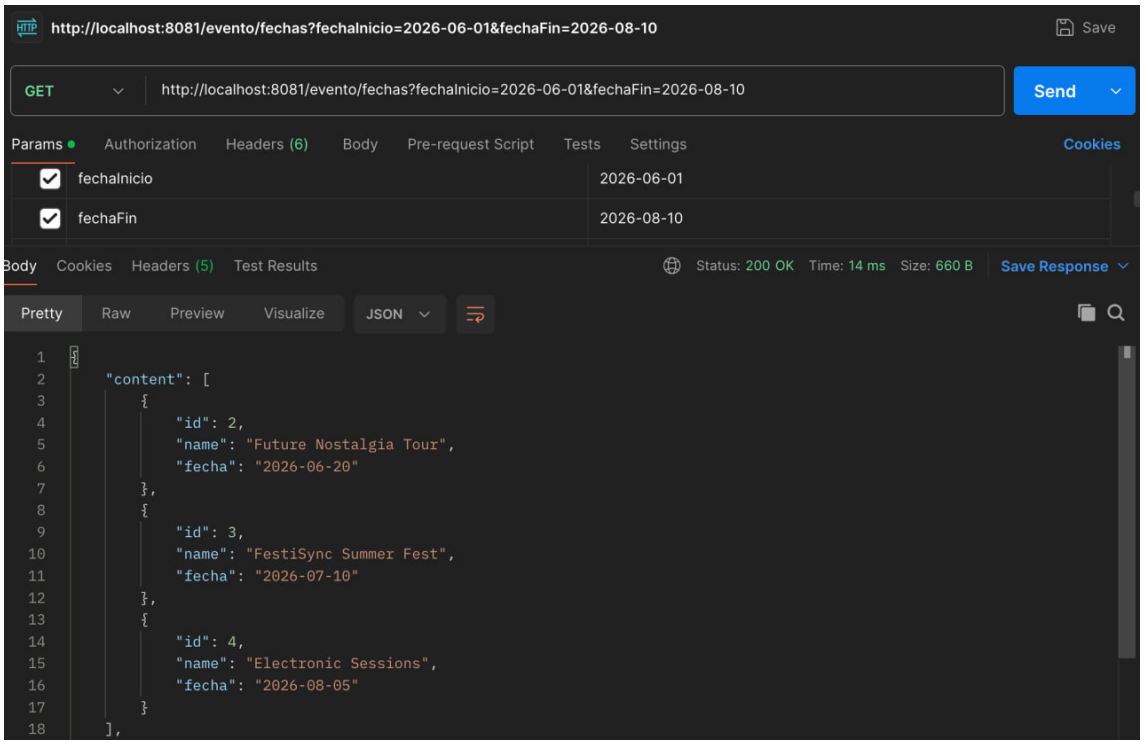
“Artistas que estén en ese IDEvento”



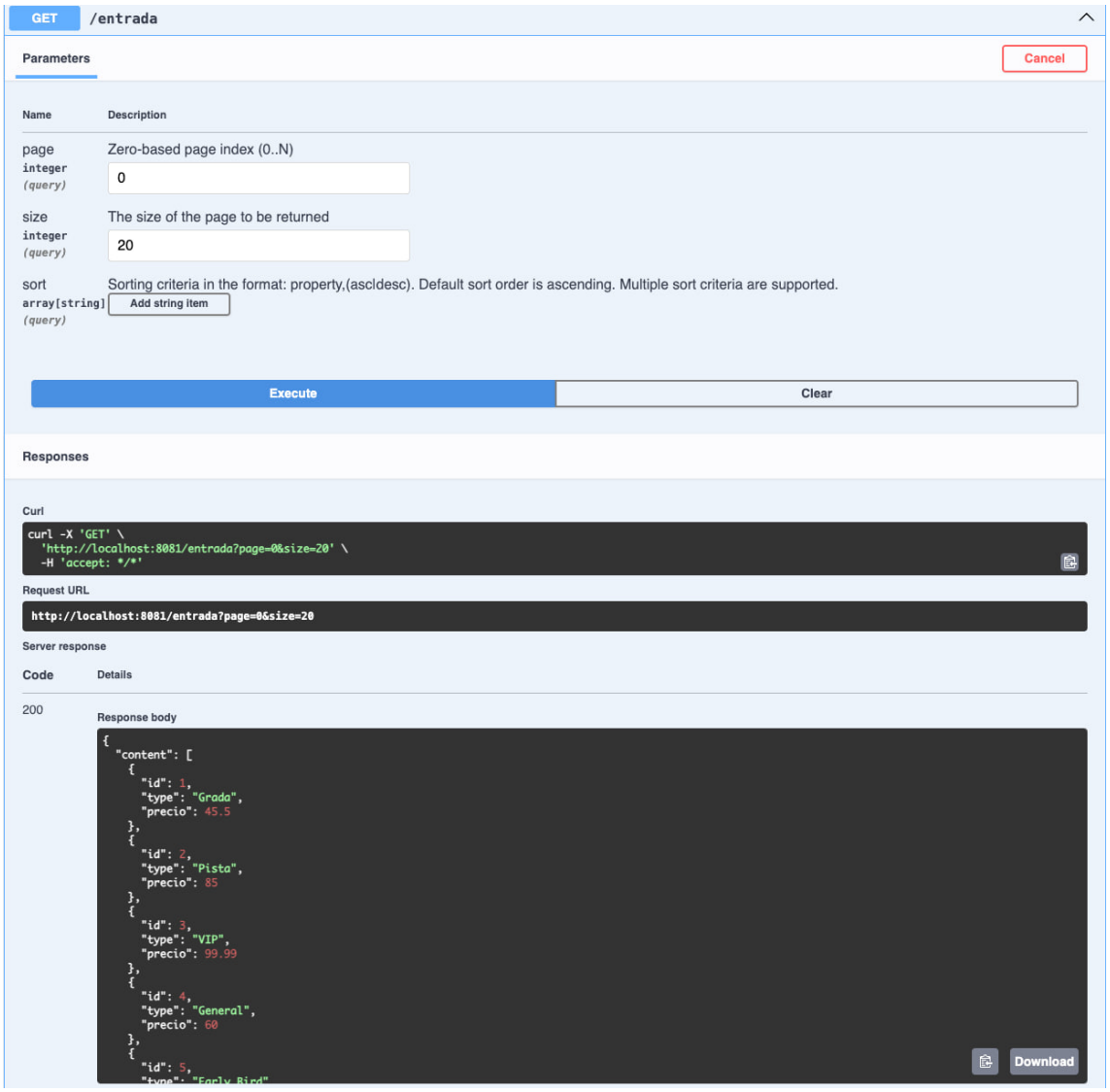
Filtrado entre fechas



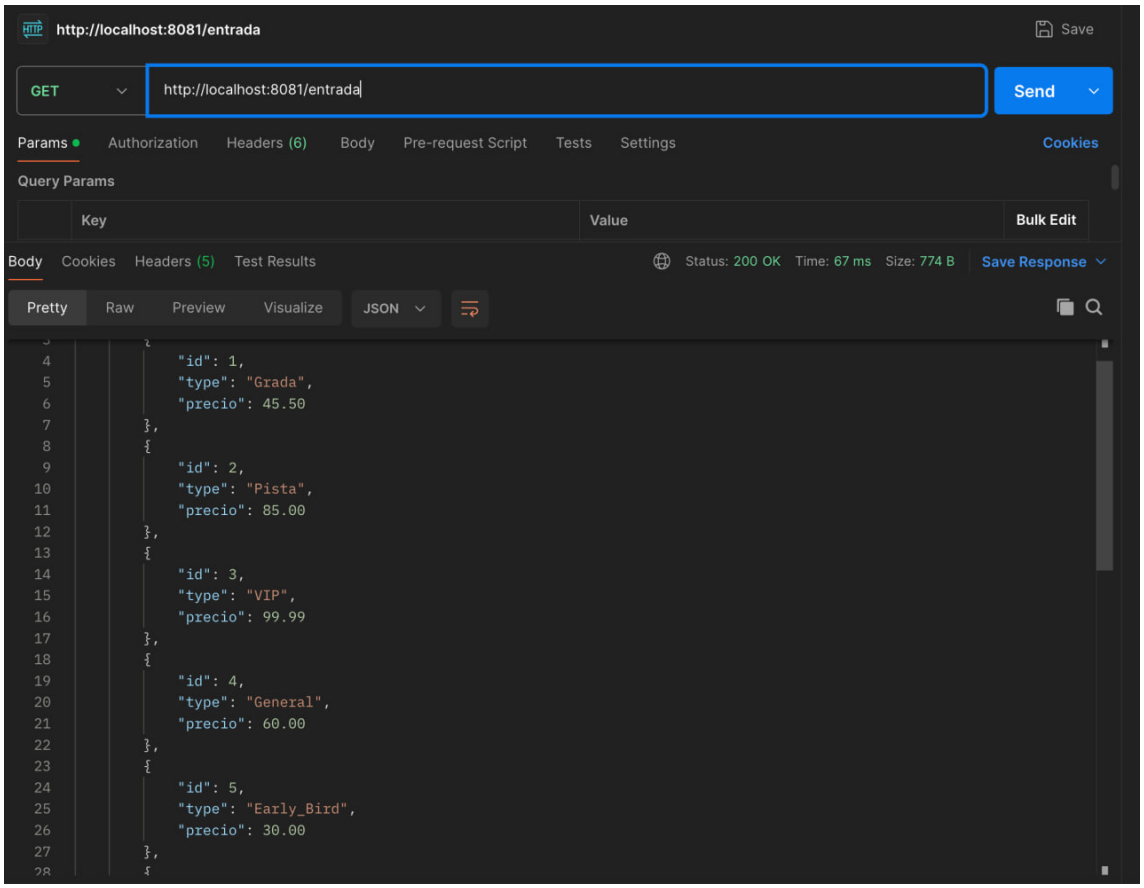
Filtrado en Postman



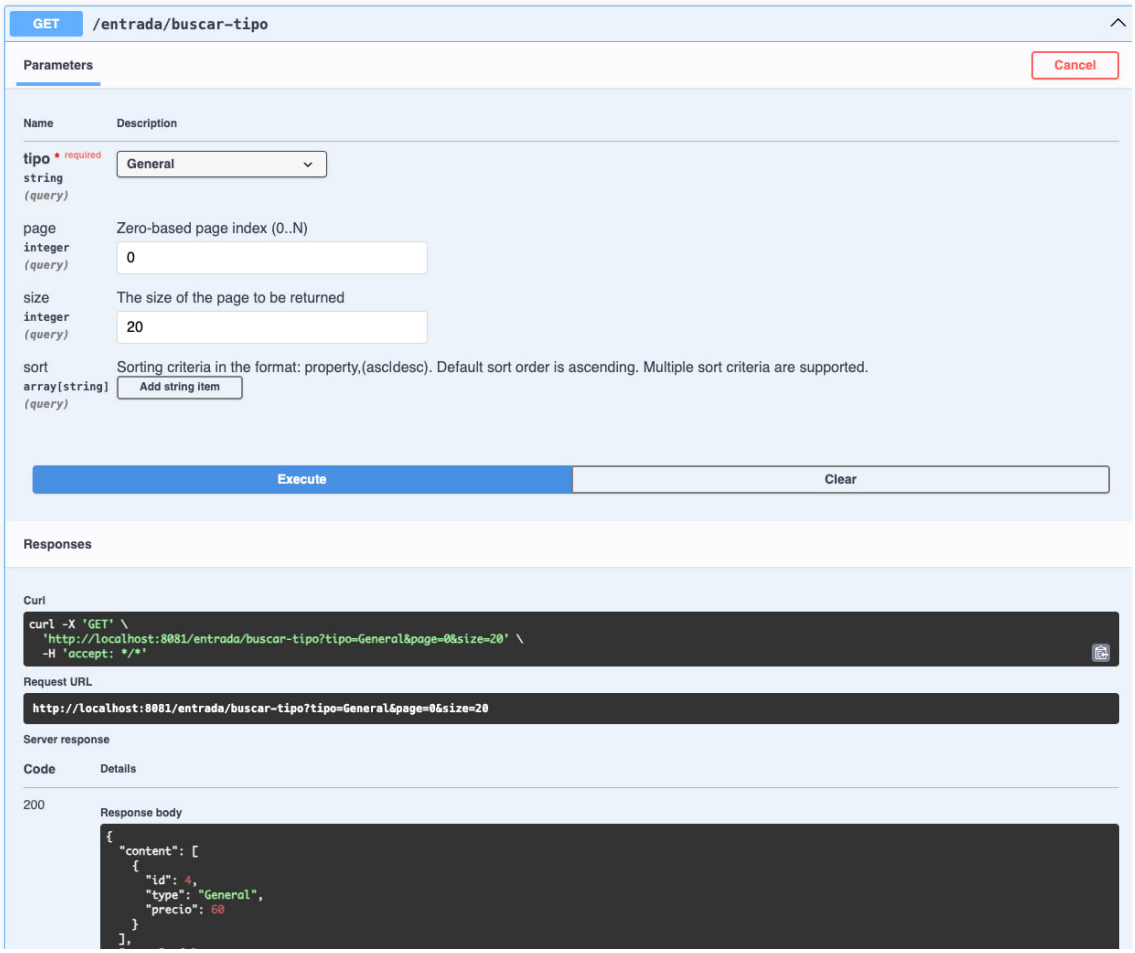
GET Entradas



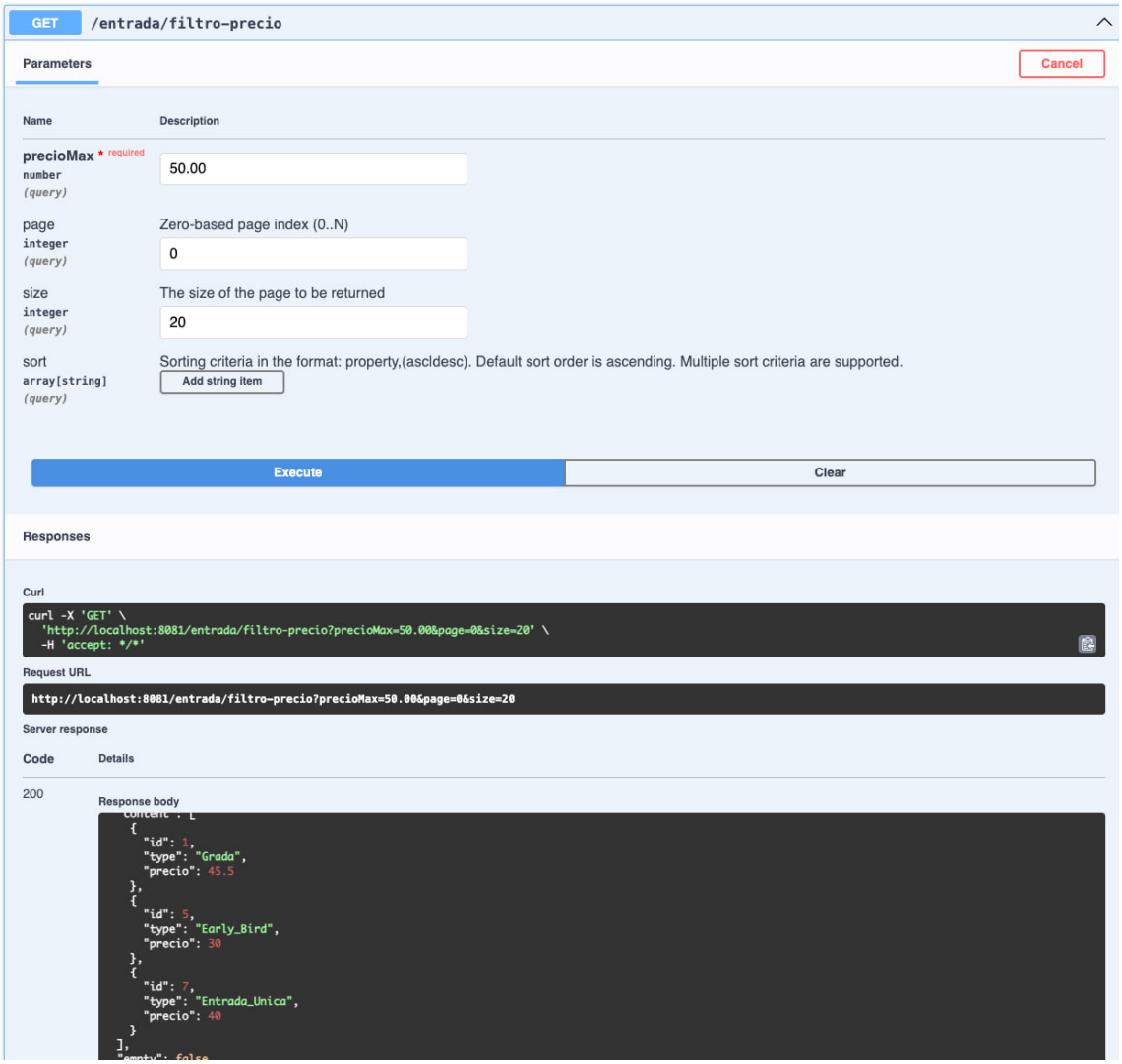
GET Entradas en Postman



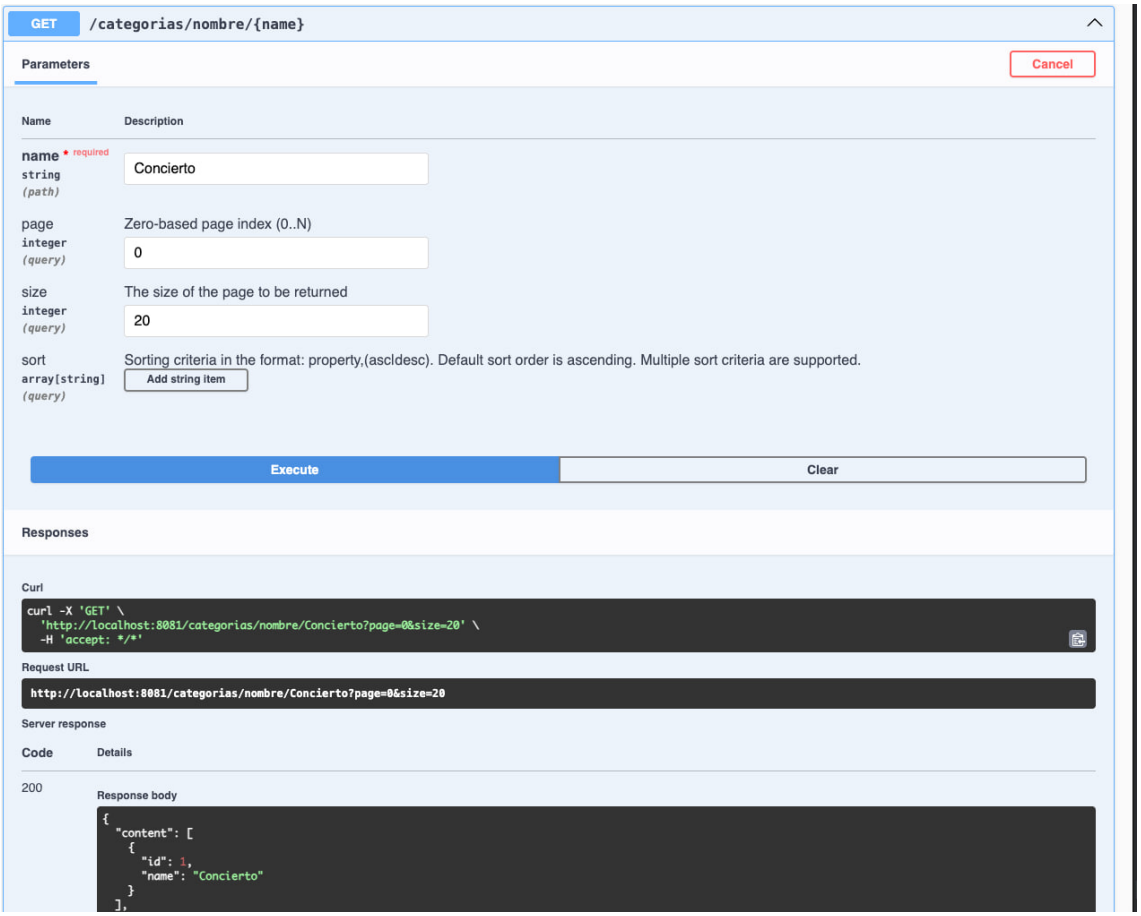
GET Entrada/buscartipo



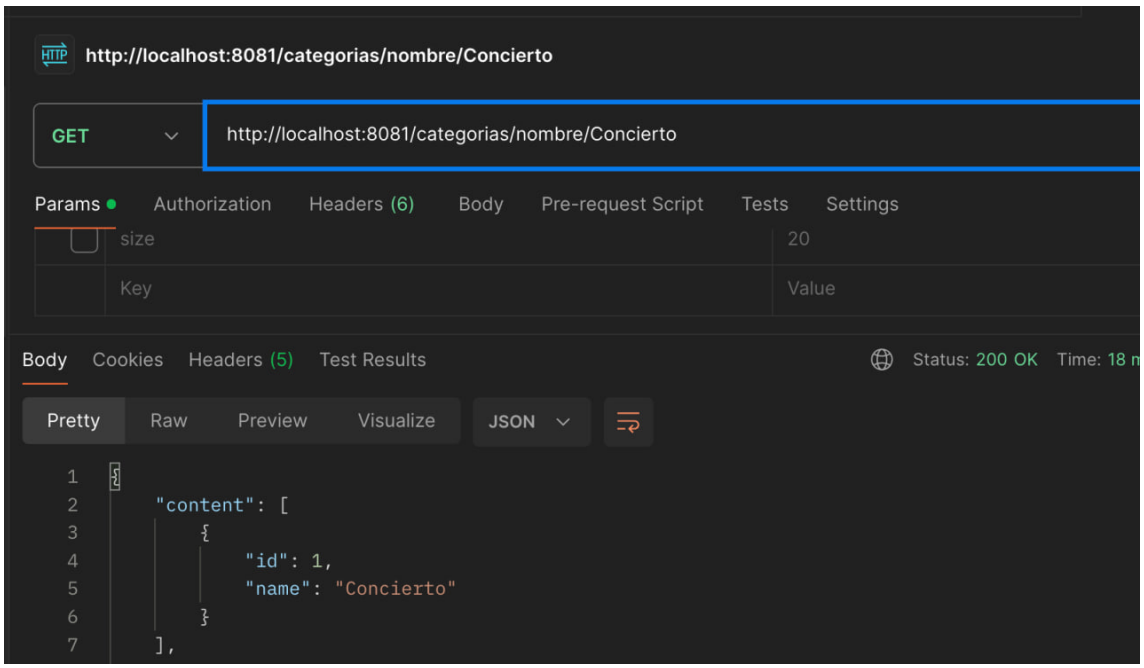
GET entrada/filtroprecio



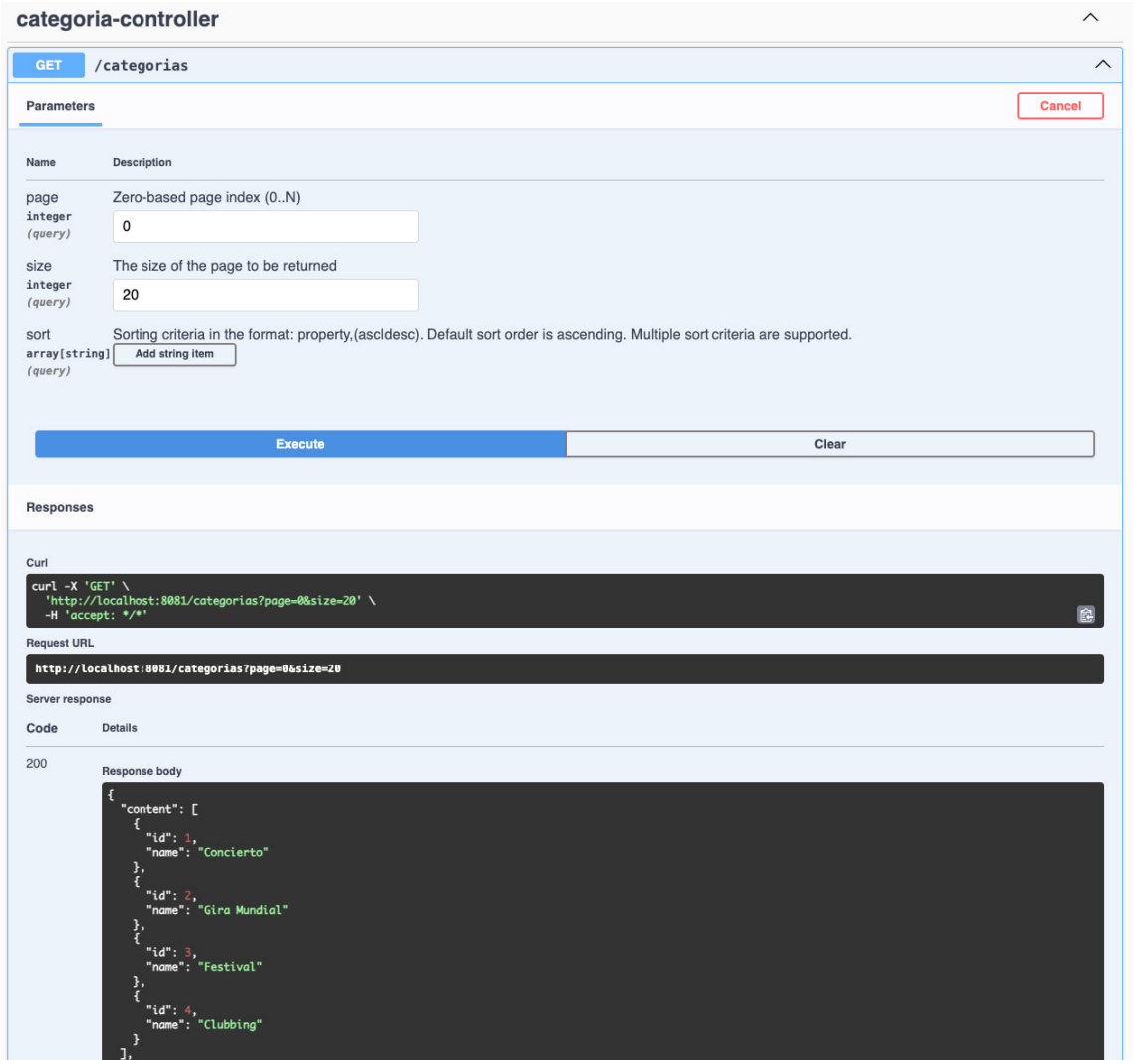
GET categorias/nombre



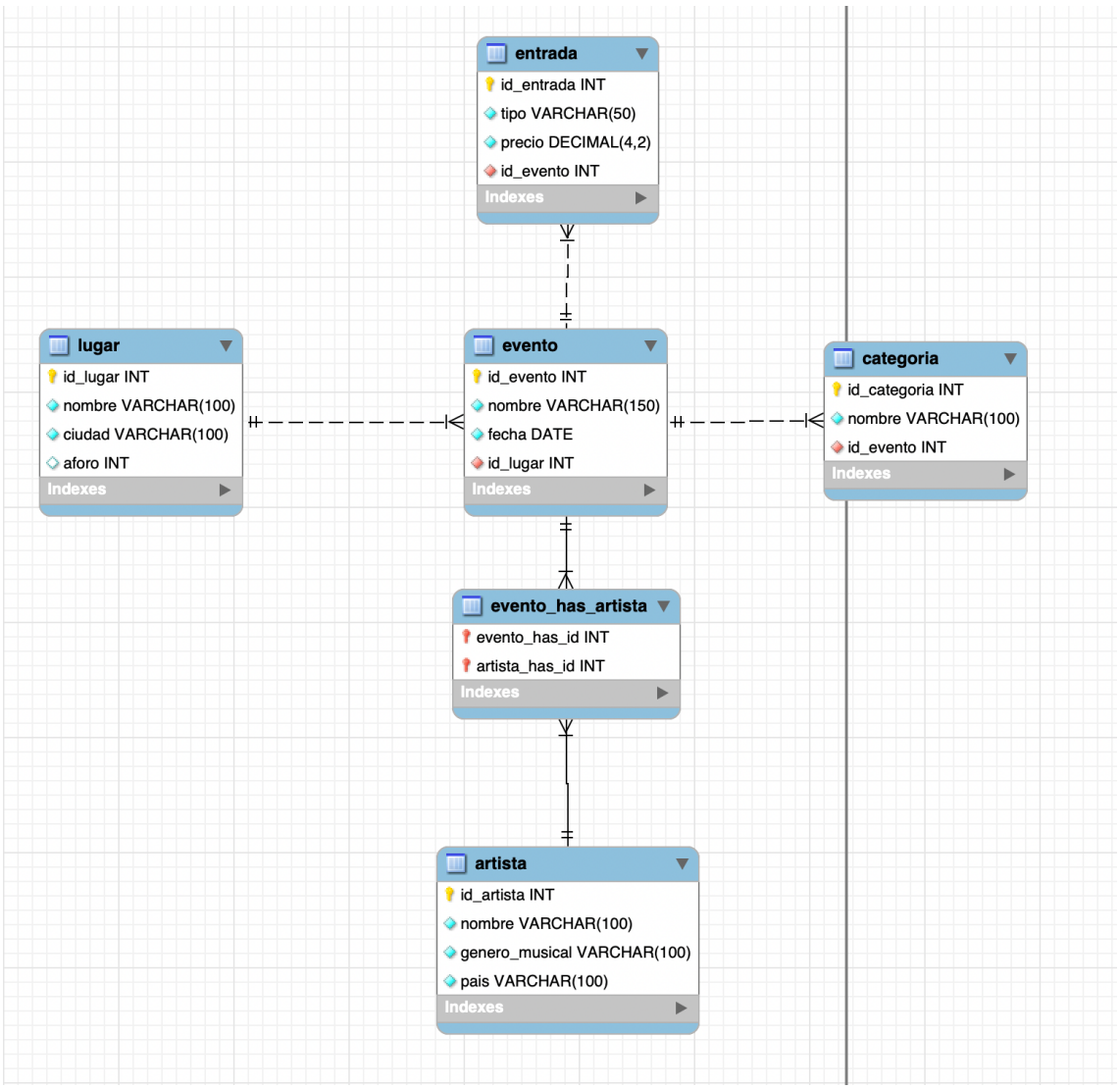
En Postman.



GET Categorias



- Diagrama E/R final.



```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN
_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SU
BSTITUTION';

-- -----------------------------------------------------
-- Schema FestiSync
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema FestiSync
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS FestiSync ;
USE FestiSync ;

-- -----------------------------------------------------
-- Table FestiSync.lugar
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.lugar ;

CREATE TABLE IF NOT EXISTS FestiSync.lugar (
  id_lugar INT NOT NULL AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL,
  ciudad VARCHAR(100) NOT NULL,
  aforo INT NULL,
  PRIMARY KEY (id_lugar))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table FestiSync.evento
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.evento ;

CREATE TABLE IF NOT EXISTS FestiSync.evento (
  id_evento INT NOT NULL AUTO_INCREMENT,
  nombre VARCHAR(150) NOT NULL,
  fecha DATE NOT NULL,
  id_lugar INT NOT NULL,
  PRIMARY KEY (id_evento),
  INDEX fk_evento_lugar_idx (id_lugar ASC) VISIBLE,
```

```
  CONSTRAINT fk_evento_lugar
    FOREIGN KEY (id_lugar)
    REFERENCES FestiSync.lugar (id_lugar)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table FestiSync.categoria
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.categoria ;

CREATE TABLE IF NOT EXISTS FestiSync.categoria (
  id_categoria INT NOT NULL AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL,
  id_evento INT NOT NULL,
  PRIMARY KEY (id_categoria),
  INDEX fk_categoria_evento1_idx (id_evento ASC) VISIBLE,
  CONSTRAINT fk_categoria_evento1
    FOREIGN KEY (id_evento)
    REFERENCES FestiSync.evento (id_evento)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table FestiSync.artista
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.artista ;

CREATE TABLE IF NOT EXISTS FestiSync.artista (
  id_artista INT NOT NULL AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL,
  genero_musical VARCHAR(100) NOT NULL,
  pais VARCHAR(100) NOT NULL,
  PRIMARY KEY (id_artista))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table FestiSync.evento_has_artista
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.evento_has_artista ;

CREATE TABLE IF NOT EXISTS FestiSync.evento_has_artista (
```

```
  evento_has_id INT NOT NULL,
  artista_has_id INT NOT NULL,
  PRIMARY KEY (evento_has_id, artista_has_id),
  INDEX fk_evento_has_artista_artista1_idx (artista_has_id ASC) VISIBLE,
  INDEX fk_evento_has_artista_evento1_idx (evento_has_id ASC) VISIBLE,
  CONSTRAINT fk_evento_has_artista_evento1
    FOREIGN KEY (evento_has_id)
    REFERENCES FestiSync.evento (id_evento)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT fk_evento_has_artista_artista1
    FOREIGN KEY (artista_has_id)
    REFERENCES FestiSync.artista (id_artista)
    ON DELETE CASCADE
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table FestiSync.entrada
-- -----------------------------------------------------
DROP TABLE IF EXISTS FestiSync.entrada ;

CREATE TABLE IF NOT EXISTS FestiSync.entrada (
  id_entrada INT NOT NULL AUTO_INCREMENT,
  tipo VARCHAR(50) NOT NULL,
  precio DECIMAL(4,2) NOT NULL,
  id_evento INT NOT NULL,
  PRIMARY KEY (id_entrada),
  INDEX fk_entrada_evento1_idx (id_evento ASC) VISIBLE,
  CONSTRAINT fk_entrada_evento1
    FOREIGN KEY (id_evento)
    REFERENCES FestiSync.evento (id_evento)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```