UNIVERSIDADE DO VALE DO ITAJAÍ

CIÊNCIAS DA COMPUTAÇÃO

LAURA ERIKA KLUGE SCHETINGER

TRABALHO M3 DE BANCO DE DADOS

SISTEMA DE STREAMING: UNIFY

PROFESSOR: MAURÍCIO PASETTO DE FREITAS

ITAJAÍ / SC

2025

SUMÁRIO

1. DEFINIÇÃO

1.1. FORMULAÇÃO DO PROBLEMA

Atualmente, os ouvintes de músicas enfrentam dificuldades em organizar e centralizar suas bibliotecas de áudio, no qual com a transição das mídias físicas para o formato digital, muitos usuários acabaram com coleções fragmentadas, dispersas entre arquivos locais desorganizados, dispositivos com armazenamento limitado e mídias obsoletas. Essa descentralização impede o acesso rápido e onipresente às faixas desejadas, criando uma experiência de usuário frustrante onde a busca por um álbum ou artista específico torna-se uma tarefa manual e ineficiente, sem a possibilidade de criar listas de reprodução personalizadas ou obter informações detalhadas sobre as obras.

Além disso, existe outro problema no qual desvincula o artista de sua obra pela falta de integridade de dados. Neste caso, sem um sistema relacional estruturado, o cadastro de artistas, lançamentos de álbuns e faixas individuais sofre com redundância de dados e inconsistências, dificultando a catalogação correta de discografias. Esse cenário prejudica tanto o usuário que consome informações erradas, quanto os artistas, que necessitam de uma plataforma confiável para que suas produções sejam armazenadas, catalogadas e recuperadas de forma lógica e hierárquica.

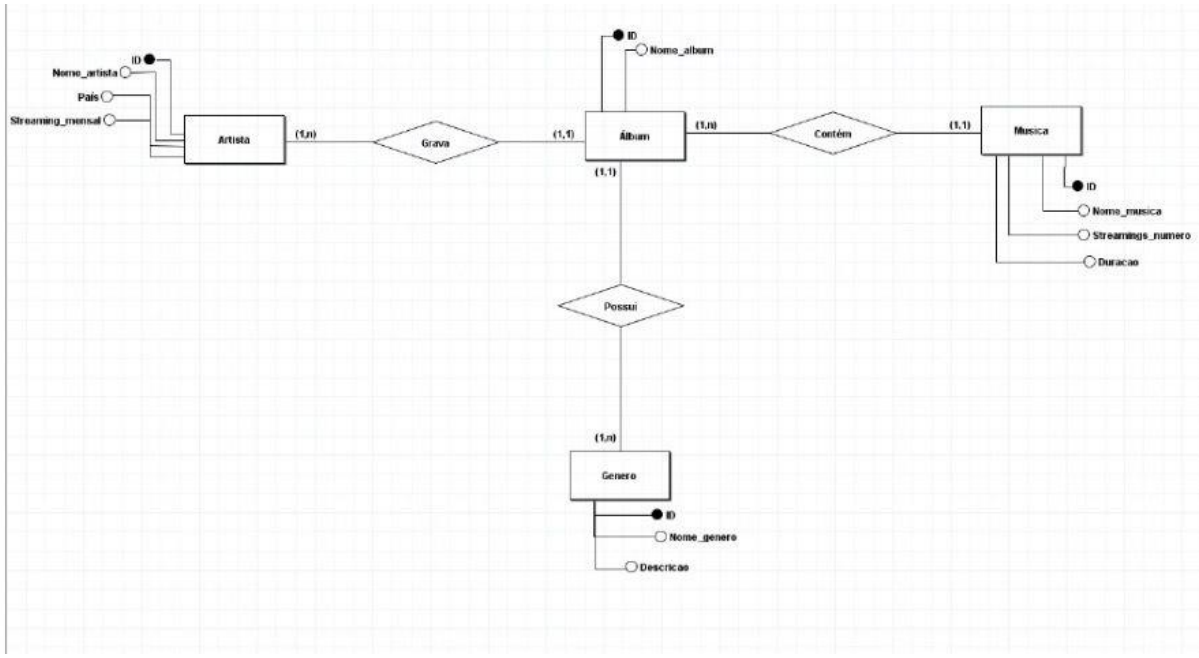
1.2. SOLUÇÃO PROPOSTA

Para mitigar os problemas de gestão e acesso ao conteúdo musical, propõe-se o desenvolvimento do **Unify**, um sistema de streaming de áudio suportado por um banco de dados relacional. A solução consiste numa plataforma centralizada que permite o registo estruturado e hierárquico das três entidades fundamentais do domínio: Artistas, Álbuns, Músicas e Gênero. Através desta arquitetura, o sistema garante a integridade referencial dos dados, assegurando que cada faixa musical esteja inequivocamente ligada ao seu álbum de origem e, consequentemente, ao seu artista criador, eliminando redundâncias e inconsistências comuns em armazenamentos amadores.

Além da organização lógica, o Unify implementará funcionalidades de manipulação de dados (CRUD) que permitirão aos administradores e usuários gerir o catálogo de forma dinâmica. A aplicação oferecerá interfaces para cadastrar

novos lançamentos, consultar discografias completas, atualizar metadados de faixas e remover conteúdos obsoletos. Desta forma, a solução não apenas resolve o problema do armazenamento disperso, mas também agrega valor ao permitir a recuperação rápida e precisa de informações, servindo como base sólida para futuras expansões, como a criação de playlists personalizadas e sistemas de recomendação.

2. PROJETO CONCEITUAL

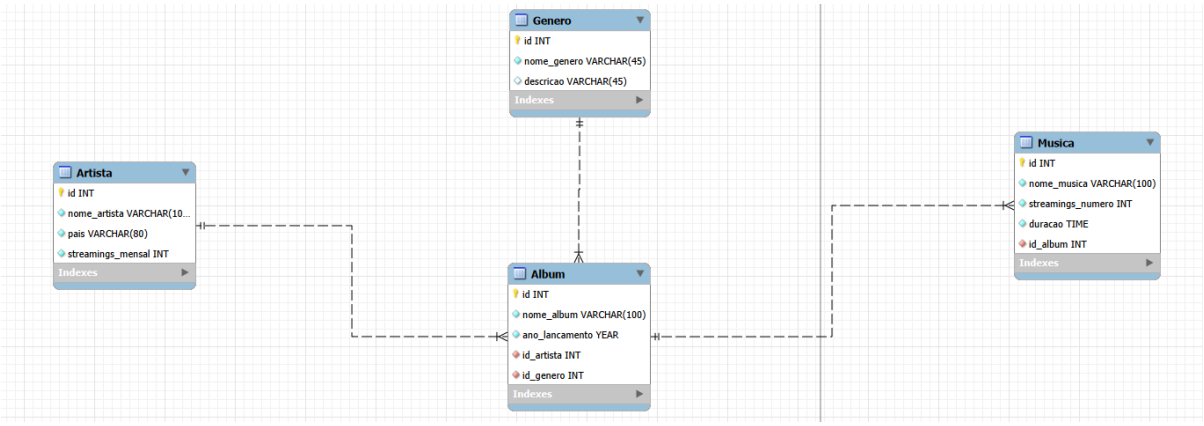


3. PROJETO LÓGICO

3.1. PROJETO LÓGICO TEXTUAL NORMALIZADO

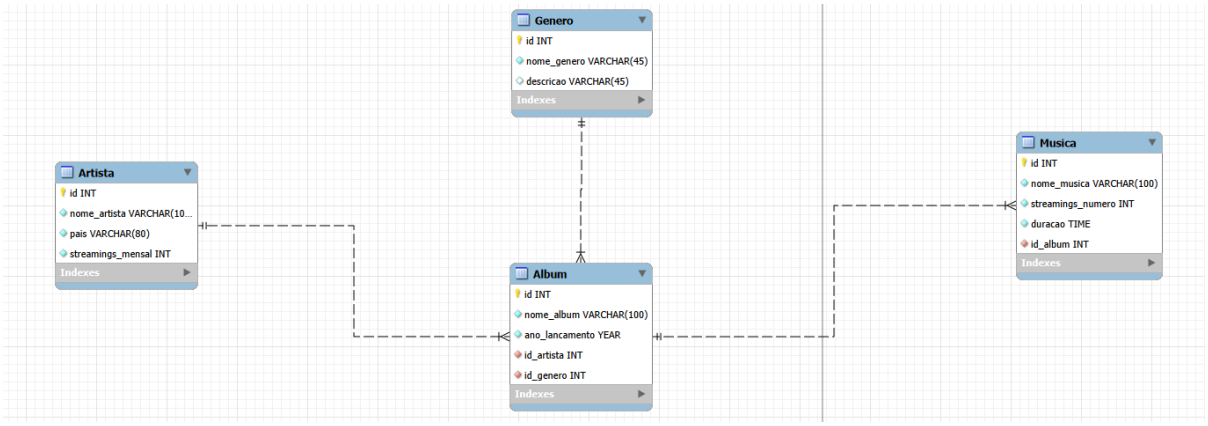
GENERO (id, nome_genero, descricao)
ARTISTA (id, nome_artista, pais, streaming_mensal)
ALBUM (id, nome_album, ano_lancamento, id_artista, id_genero)
MUSICA (id, nome_musica, streamings_numero, duracao, id_album)

3.2. DIAGRAMA RELACIONAL NORMALIZADO



4. PROJETO FÍSICO

4.1. DIAGRAMA RELACIONAL



4.2. CÓDIGO SQL

```sql
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_SUBSTITUTION';

CREATE SCHEMA IF NOT EXISTS `unify` DEFAULT CHARACTER SET utf8 ;
USE `unify` ;

CREATE TABLE IF NOT EXISTS `unify`.`Artista` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `nome_artista` VARCHAR(100) NOT NULL,
  `pais` VARCHAR(80) NOT NULL,
  `streamings_mensal` INT NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `unify`.`Genero` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `nome_genero` VARCHAR(45) NOT NULL,
  `descricao` VARCHAR(45) NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `unify`.`Album` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `nome_album` VARCHAR(100) NOT NULL,
  `ano_lancamento` YEAR NOT NULL,
  `id_artista` INT NOT NULL,
  `id_genero` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `album_artista_idx` (`id_artista` ASC) VISIBLE,
  INDEX `album_genero_idx` (`id_genero` ASC) VISIBLE,
  CONSTRAINT `album_artista`
    FOREIGN KEY (`id_artista`)
    REFERENCES `unify`.`Artista` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `album_genero`
    FOREIGN KEY (`id_genero`)
    REFERENCES `unify`.`Genero` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `unify`.`Musica` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `nome_musica` VARCHAR(100) NOT NULL,
  `streamings_numero` INT NOT NULL,
  `duracao` TIME NOT NULL,
  `id_album` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `musica_album_idx` (`id_album` ASC) VISIBLE,
  CONSTRAINT `musica_album`
    FOREIGN KEY (`id_album`)
    REFERENCES `unify`.`Album` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

ALTER TABLE `unify`.`Musica`
MODIFY COLUMN `streamings_numero` BIGINT;

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

4.3. INSERÇÃO DE DADOS

```sql
USE `unify`;

INSERT INTO `unify`.`Artista` (nome_artista, pais, streamings_mensal) VALUES
('Pink Floyd', 'Inglaterra', 245000000),
('Charli XCX', 'Inglaterra', 327000000),
('Raul Seixas', 'Brasil', 32000000),
('Linkin Park', 'Estados Unidos', 534000000);

INSERT INTO `unify`.`Genero` (nome_genero, descricao) VALUES
('Rock', 'Rock clássico, alternativo e pop rock'),
('Pop', 'Música popular comercial e hits'),
('Heavy Metal', 'Numetal e Death Metal'),
('MPB', 'Música Popular Brasileira clássica');

INSERT INTO `unify`.`Album` (nome_album, ano_lancamento, id_artista, id_genero) VALUES
('The Wall', 1979, 1, 1),
('BRAT', 2024, 2, 2),
('"Kring-Ha, Bandolo"', 1973, 3, 4),
('Meteora', 2003, 4, 3);

INSERT INTO `unify`.`Musica` (nome_musica, streamings_numero, duracao, id_album) VALUES
('Comfortably Numb', 788019458, '00:06:22', 1),
('365', 245568747, '00:03:22', 2),
('Metamorfose Ambulante', 132144934, '00:03:50', 3),
('Numb', 2514936885, '00:03:06', 4);
```

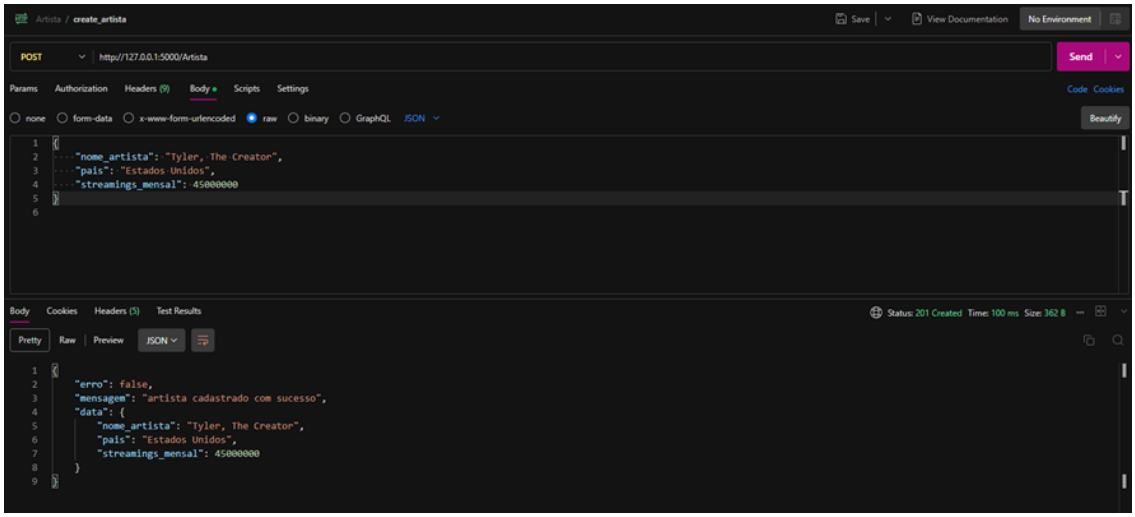
6. API EM PYTHON

Na segunda parte que foi solicitado, foi utilizado a linguagem Python para produzir uma API utilizando as operações CRUD (Create, Read, Update, Delete) com o serviço de streaming produzido no mySQL, foi utilizado o Postman para ser realizado as requisições.

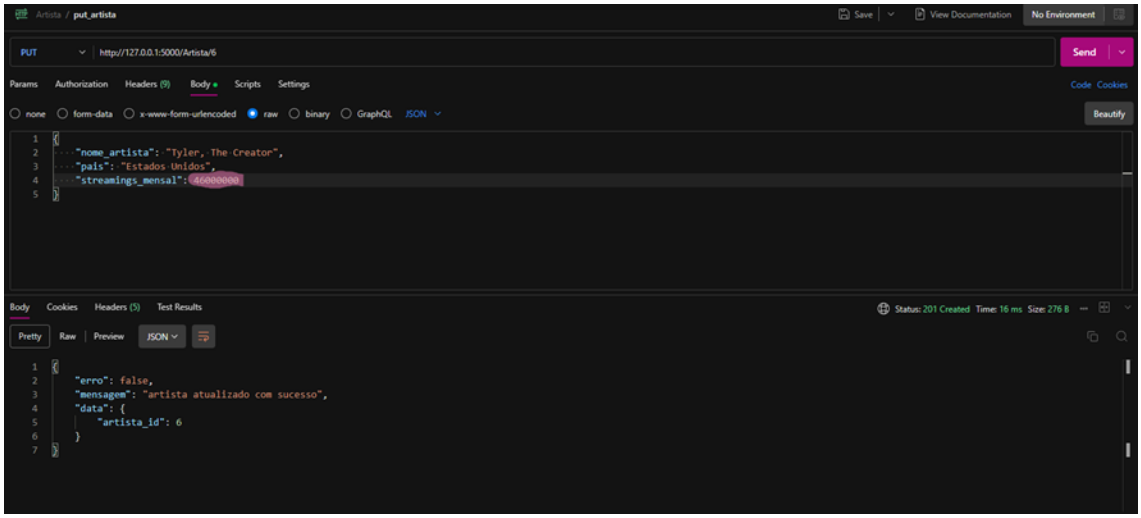
A seguir, prints tirados para visualizar os comandos feitos juntamente com as tabelas utilizadas.

6.1. TABELA ‘ARTISTA’

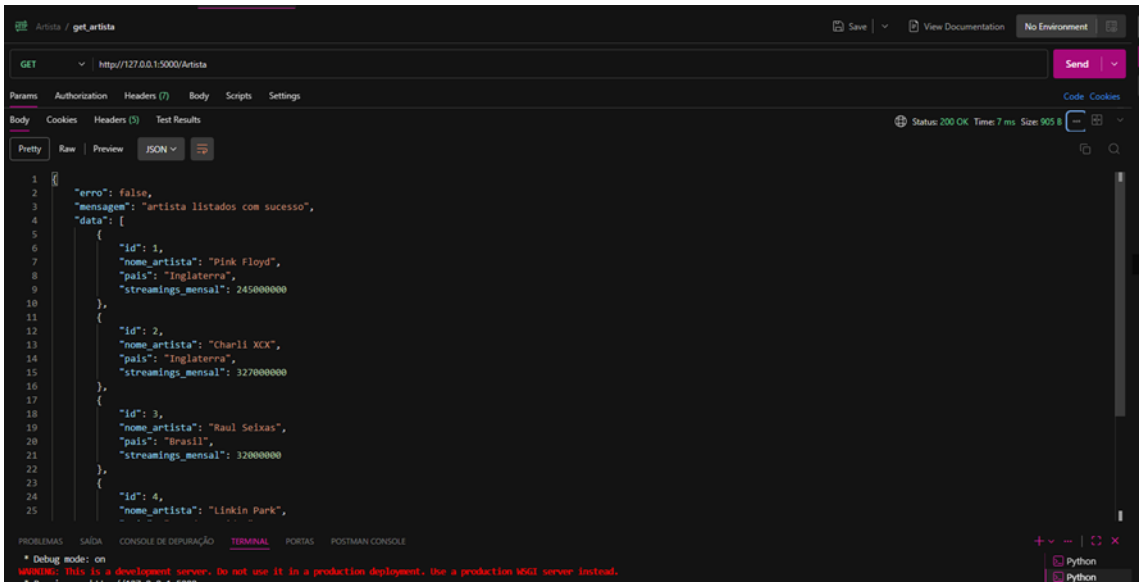
6.1.1. Create

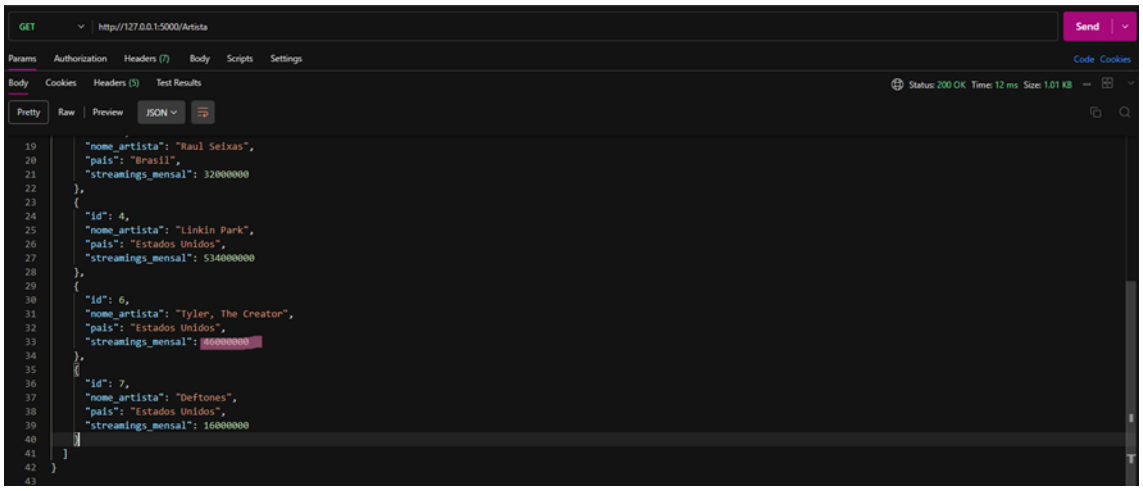


6.1.2. Update

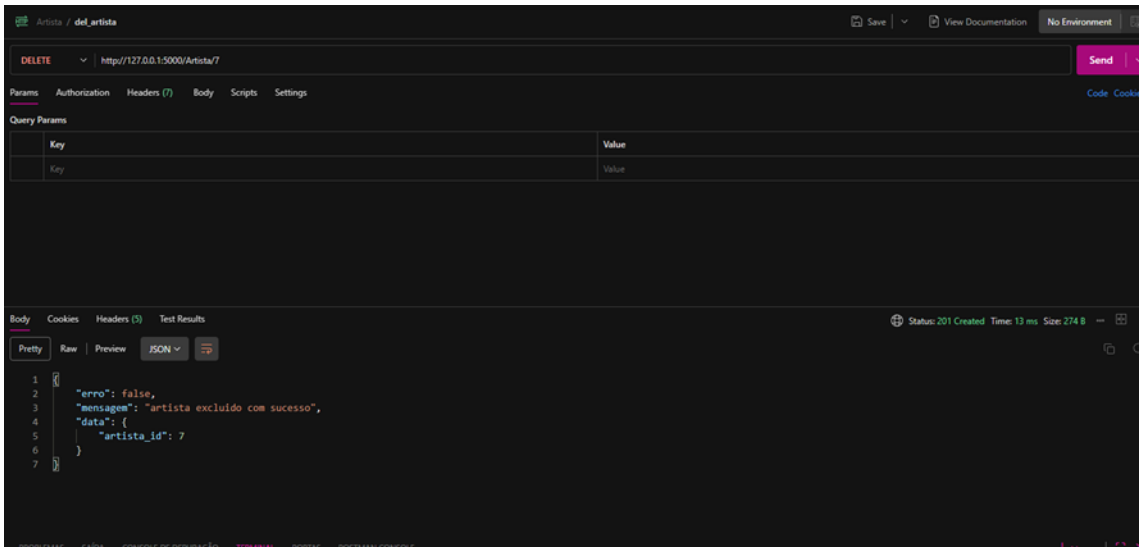


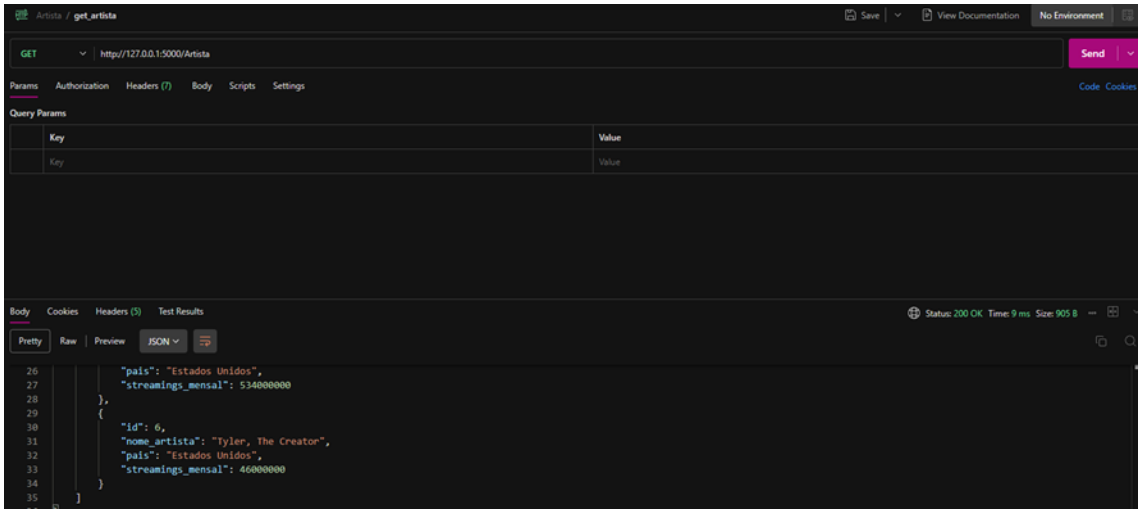
6.1.3. Read





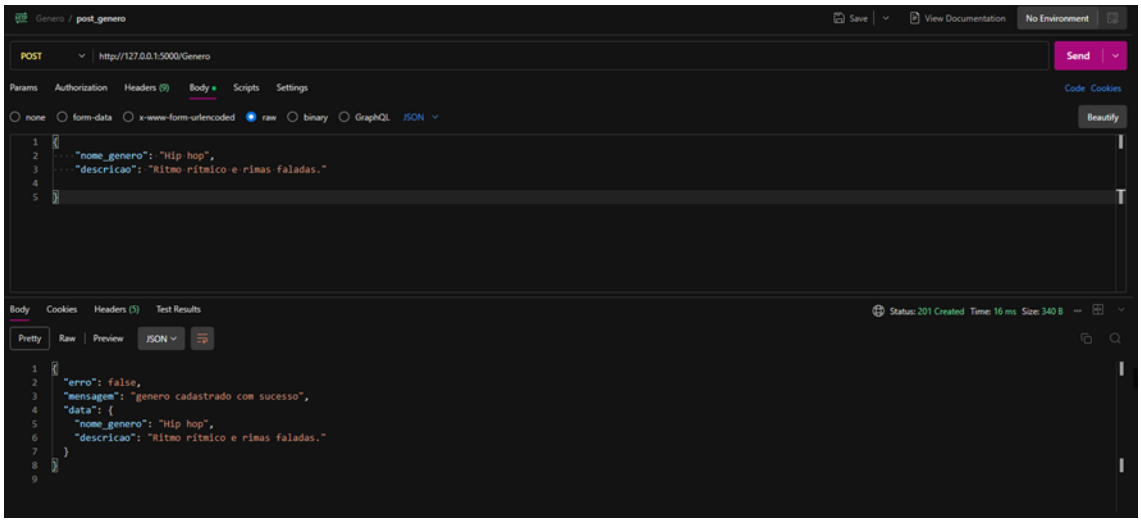
6.1.4. Delete



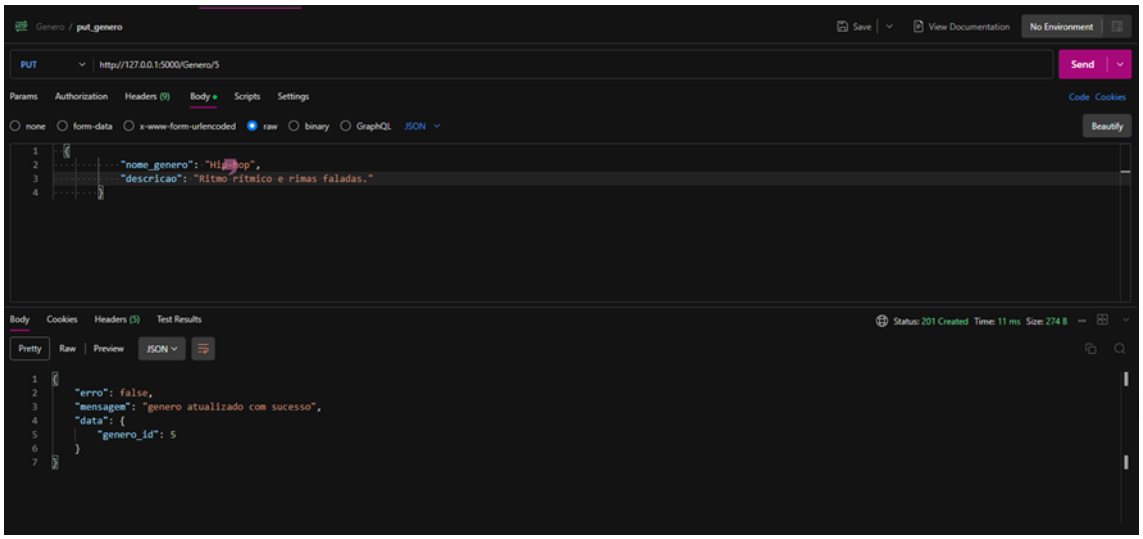


6.2. TABELA ‘GÊNERO’

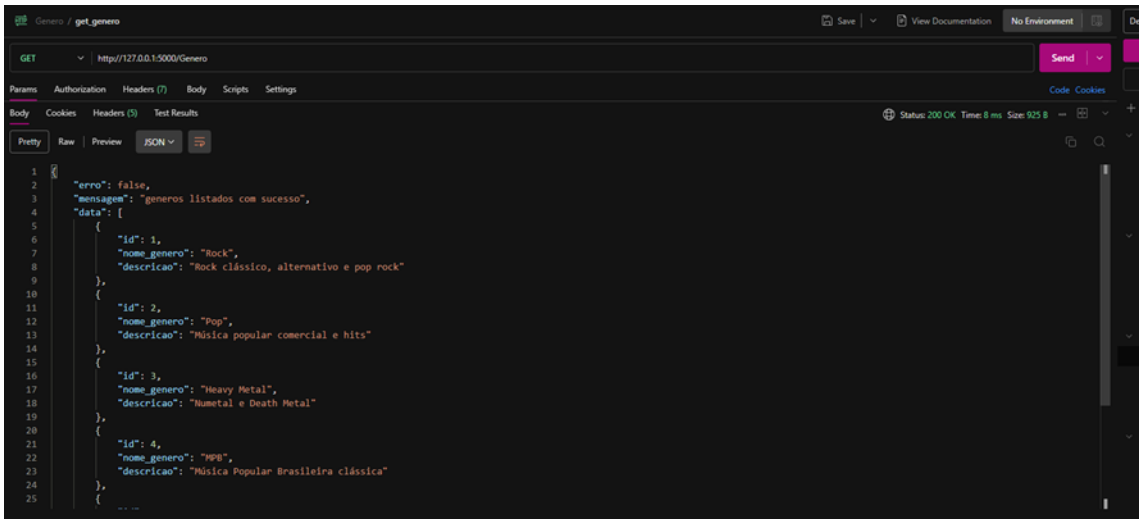
6.2.1. Create



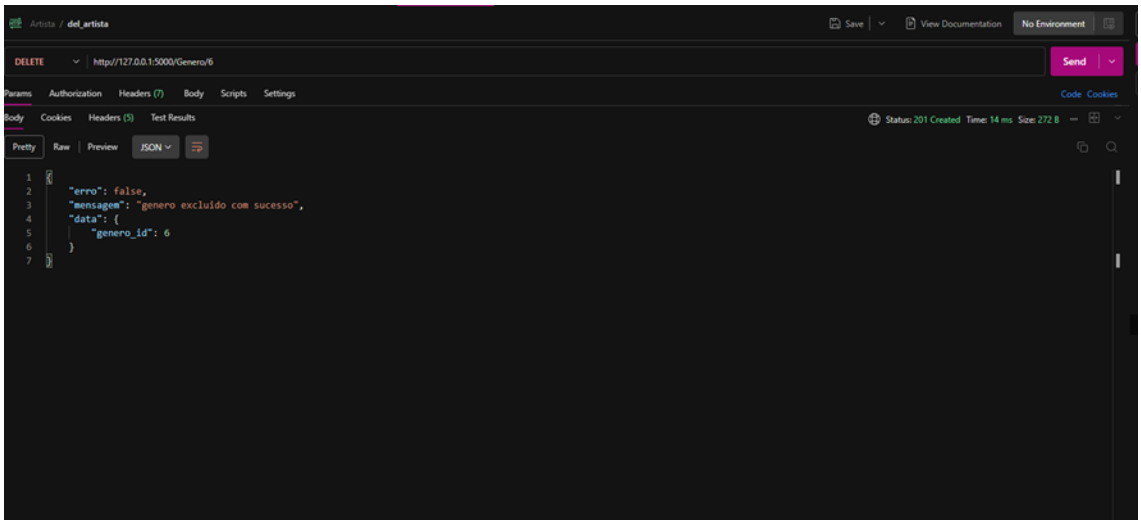
6.2.2. Update



6.2.3. Read

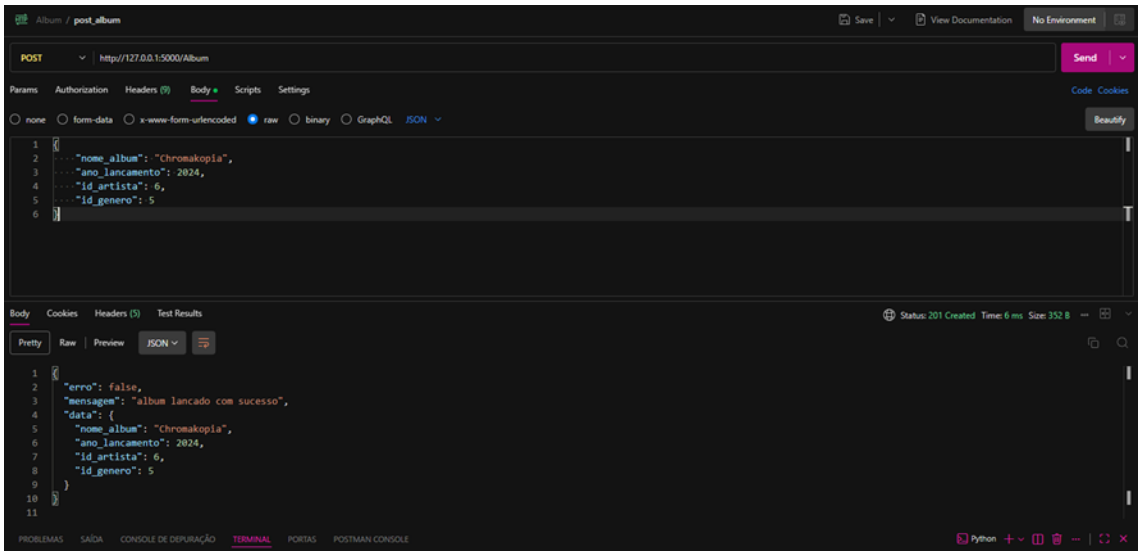


6.2.4. Delete

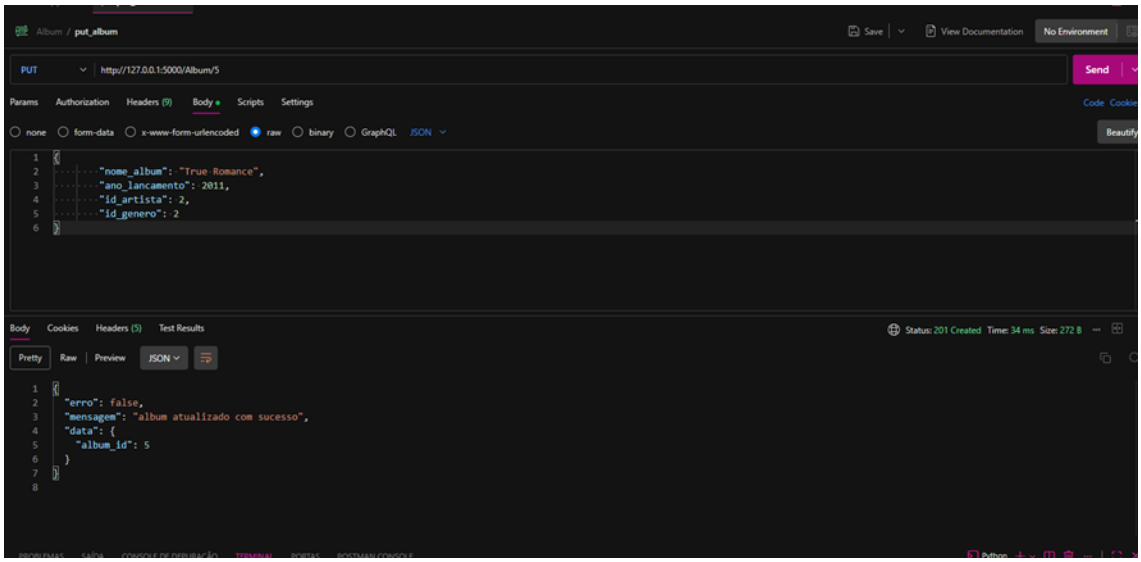


6.3. TABELA ‘ÁLBUM’

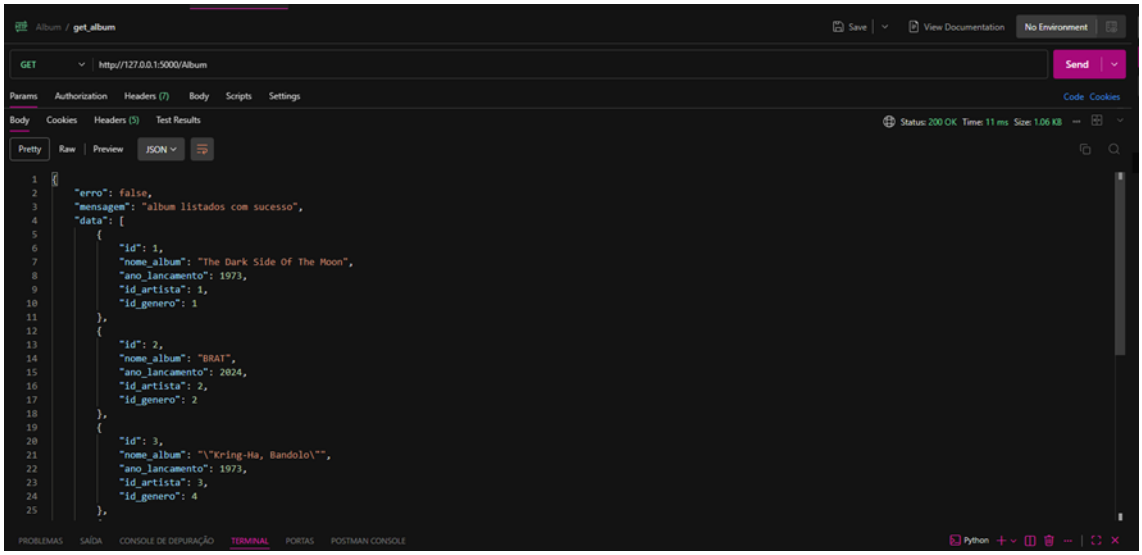
6.3.1. Create

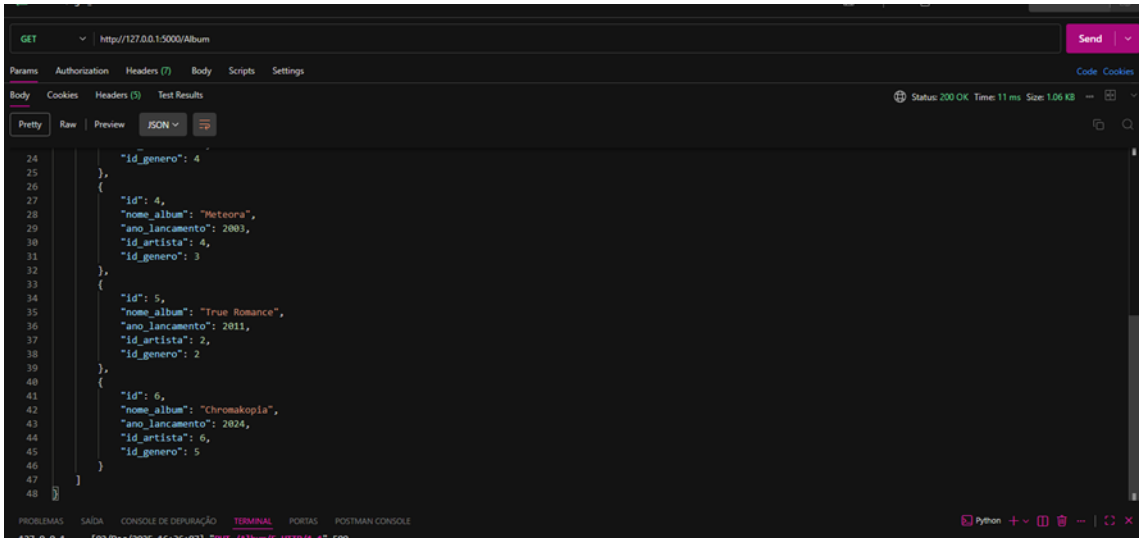


6.3.2. Update

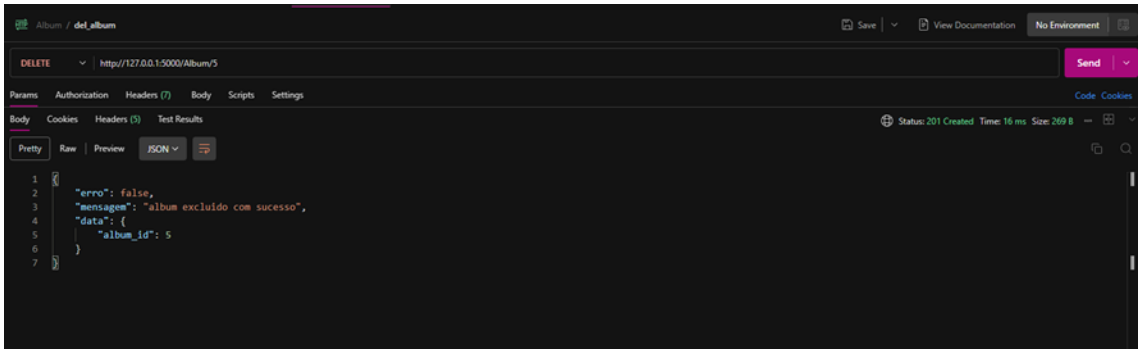

Album / put_album
Save
View Documentation
No Environment
PUT
http://127.0.0.1:5000/Album/5
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Code
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"nome_album": "True Romance",
"ano_lancamento": 2011,
"id_artista": 2,
"id_genero": 2
}
Body
Cookies
Headers (5)
Test Results
Status: 201 Created
Time: 34 ms
Size: 272 B
Pretty
Raw
Preview
JSON
{
"erro": false,
"mensagem": "album atualizado com sucesso",
"data": {
"album_id": 5
}
}

6.3.3. Read


Album / get_album
Save
View Documentation
No Environment
GET
http://127.0.0.1:5000/Album
Send
Params
Authorization
Headers (7)
Body
Scripts
Settings
Code
Cookies
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 11 ms
Size: 1.06 KB
Pretty
Raw
Preview
JSON
{
"erro": false,
"mensagem": "album listados com sucesso",
"data": [
{
"id": 1,
"nome_album": "The Dark Side Of The Moon",
"ano_lancamento": 1973,
"id_artista": 1,
"id_genero": 1
},
{
"id": 2,
"nome_album": "BRAT",
"ano_lancamento": 2024,
"id_artista": 2,
"id_genero": 2
},
{
"id": 3,
"nome_album": "\"Kring-Ha, Bandolo\"",
"ano_lancamento": 1973,
"id_artista": 3,
"id_genero": 4
},
PROBLEMAS
SAÍDA
CONSOLE DE DEPURAÇÃO
TERMINAL
PORTAS
POSTMAN CONSOLE
Python

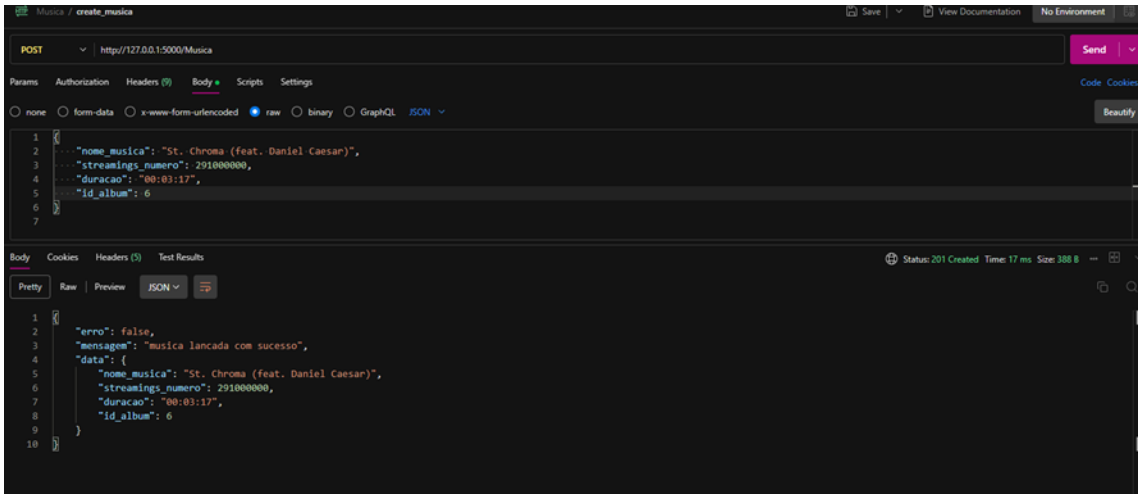


6.3.4. Delete

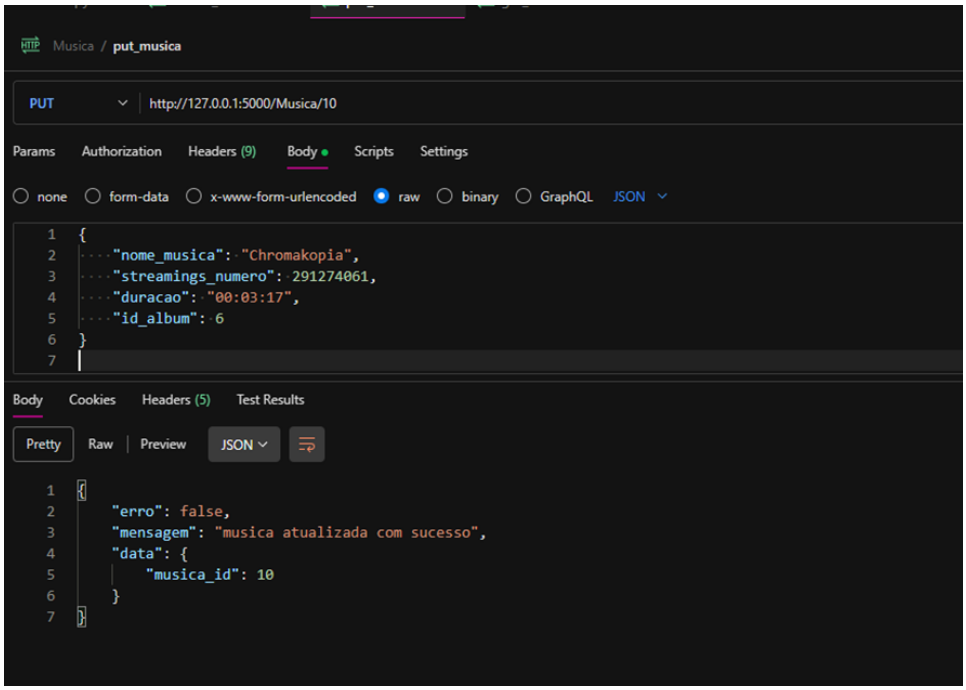


6.4. TABELA ‘MÚSICA’

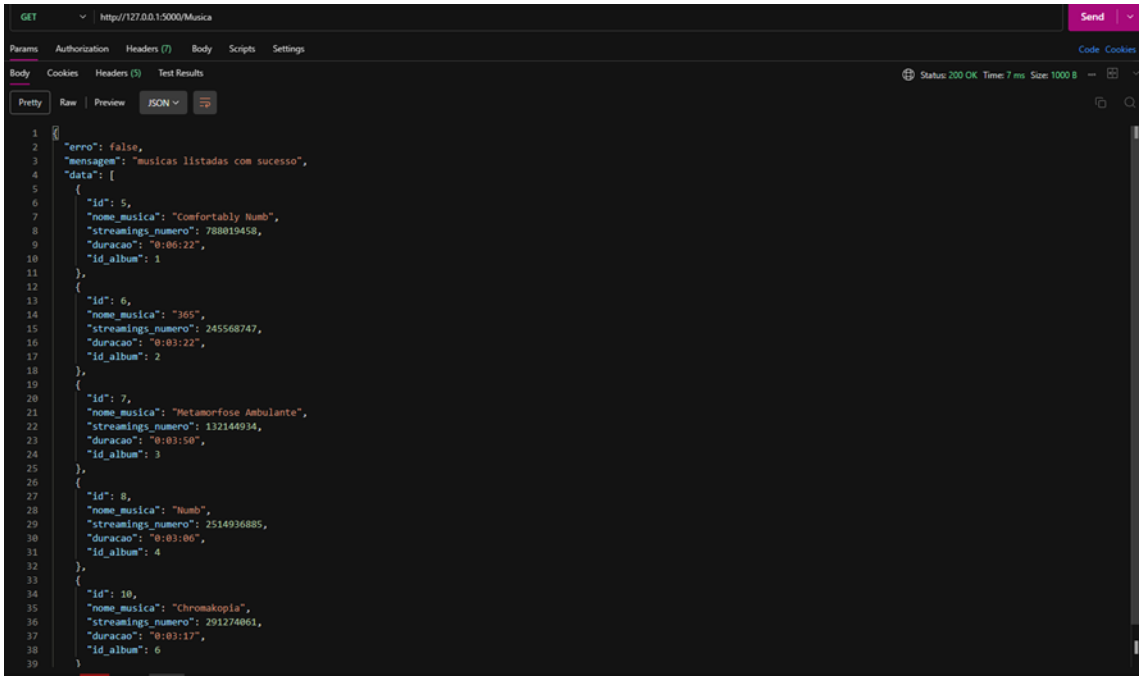
6.4.1. Create



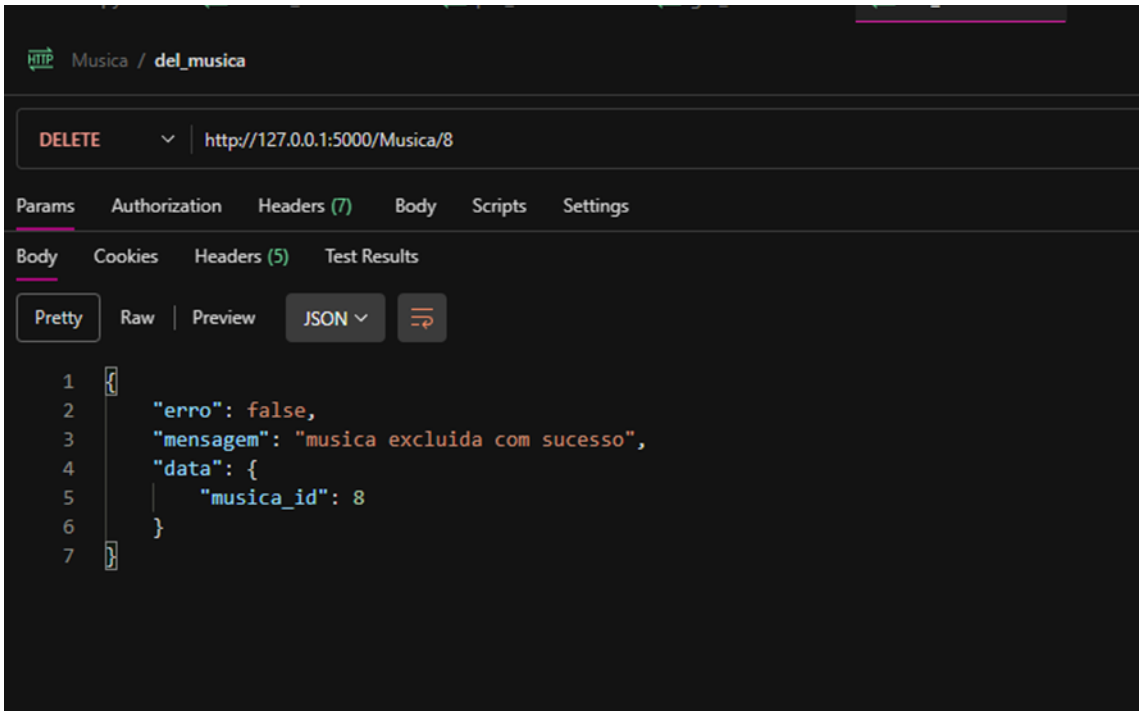
6.4.2. Update



6.4.3. Read


GET
http://127.0.0.1:5000/Musica
Send
Params
Authorization
Headers (7)
Body
Scripts
Settings
Code
Cookies
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 7 ms
Size: 1000 B
Pretty
Raw
Preview
JSON
{
"erro": false,
"mensagem": "musicas listadas com sucesso",
"data": [
{
"id": 5,
"nome_musica": "Comfortably Numb",
"streamings_numero": 788019458,
"duracao": "0:06:22",
"id_album": 1
},
{
"id": 6,
"nome_musica": "365",
"streamings_numero": 245568747,
"duracao": "0:03:22",
"id_album": 2
},
{
"id": 7,
"nome_musica": "Metamorfose Ambulante",
"streamings_numero": 132144934,
"duracao": "0:03:50",
"id_album": 3
},
{
"id": 8,
"nome_musica": "Numb",
"streamings_numero": 2514936885,
"duracao": "0:03:06",
"id_album": 4
},
{
"id": 10,
"nome_musica": "Chromakopia",
"streamings_numero": 291274061,
"duracao": "0:03:17",
"id_album": 6
}

6.4.4. Delete


HTTP
Musica / del_musica
DELETE
http://127.0.0.1:5000/Musica/8
Params
Authorization
Headers (7)
Body
Scripts
Settings
Body
Cookies
Headers (5)
Test Results
Pretty
Raw
Preview
JSON
{
"erro": false,
"mensagem": "musica excluida com sucesso",
"data": {
"musica_id": 8
}
}

7. REPOSITÓRIO GITHUB