

# Módulo 9 Proyecto - Diplomado Java DGTIC

Ernesto Velasco Arciniega

Abril 10, 2025

## Índice

<b>1</b>	<b>Código fuente de la entrega</b>	<b>2</b>
1.1	GitHub . . . . .	2
1.2	Cuatro tablas con operaciones CRUD . . . . .	2
<b>2</b>	<b>Capturas de pantalla del funcionamiento</b>	<b>3</b>
2.1	Consumo exitoso de una API REST con Postman . . . . .	3
2.1.1	Resultados de Postman para Clasificación . . . . .	3
2.1.2	Resultados de Postman para Editorial . . . . .	5
2.1.3	Resultados de Postman para Autor . . . . .	7
2.1.4	Resultados de Postman para Nacionalidad . . . . .	9
2.1.5	Resultados de Postman para Libro . . . . .	11
2.2	La API RESTful debe procesar el manejo de excepciones . . . . .	16
2.2.1	Exepciones de Libro . . . . .	16

## Instrucciones:

1. Selecciona cuatro tablas de tu proyecto actual.
2. Implementa operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para cada una de estas tablas.
3. Al menos una de las tablas debe tener una relación con otra tabla. Por ejemplo, podrías tener una tabla de "autores" que se relaciona con la tabla de "libros" (le llamaremos \*API principal).
4. Consume la API REST ya sea usando Postman o Curl. Opcional consumirlo desde una aplicación con cualquier tecnología.

# 1 Código fuente de la entrega

## 1.1 GitHub

Enlace de GitHub donde se puede consultar el proyecto creado: Proyecto Final para M9 - Librería

## 1.2 Cuatro tablas con operaciones CRUD

Se agregó una quinta tabla (LIBRO), ya que esta tiene relación con el resto de las tablas seleccionadas. Las tablas con las que se trabajó fueron las siguientes:

- Clasificación
- Editorial
- Autor
- Libro
- Nacionalidad

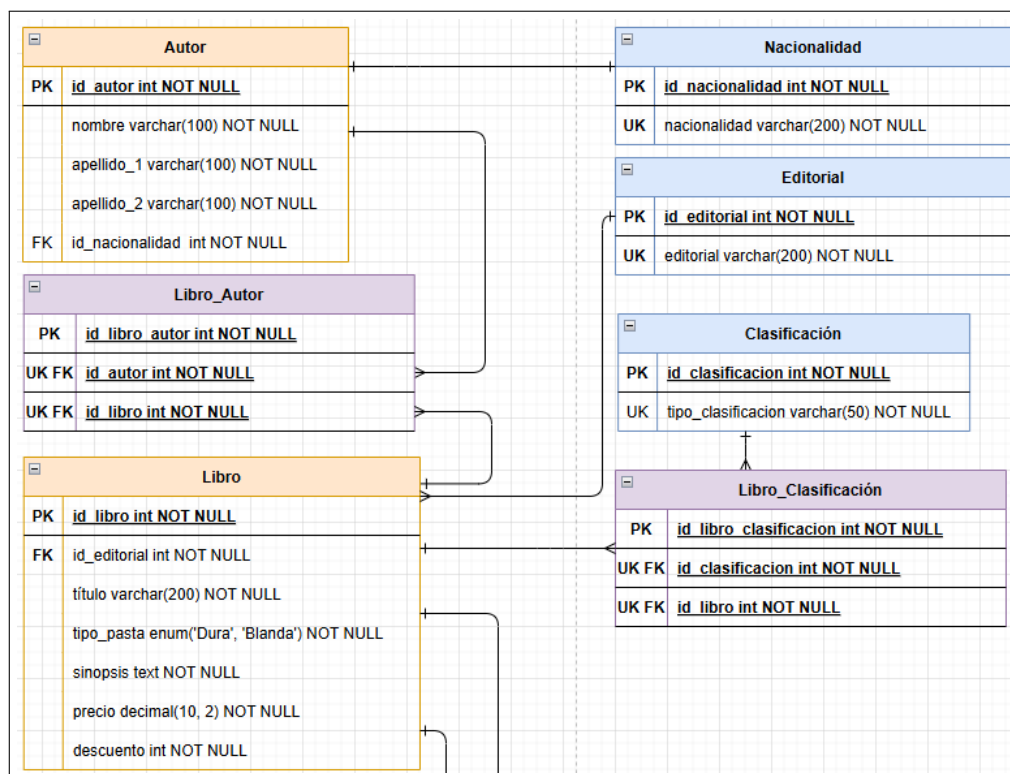


Figure 1: Cinco tablas seleccionadas del proyecto actual

## 2 Capturas de pantalla del funcionamiento

### 2.1 Consumo exitoso de una API REST con Postman

#### 2.1.1 Resultados de Postman para Clasificación

Clasificación - GET - FindAll

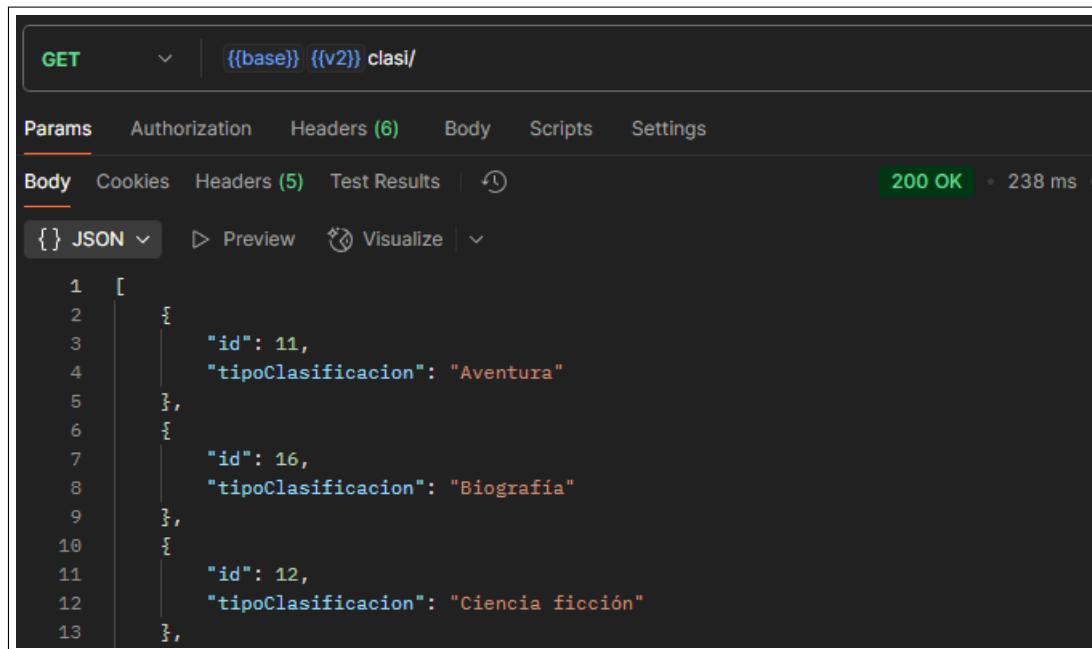


Figure 2: Clasificación FindAll - `/api/v2/clasi/`

Clasificación - GET - FindById

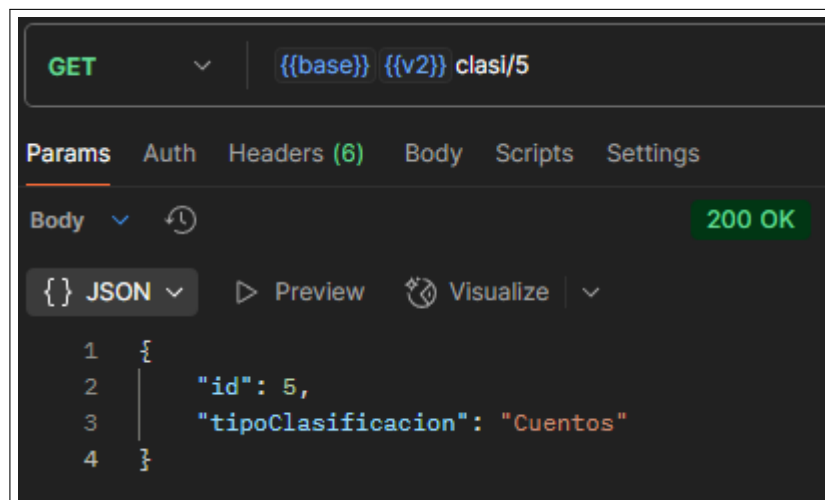


Figure 3: Clasificación FindById - `/api/v2/clasi/{id}`

## Clasificación - POST

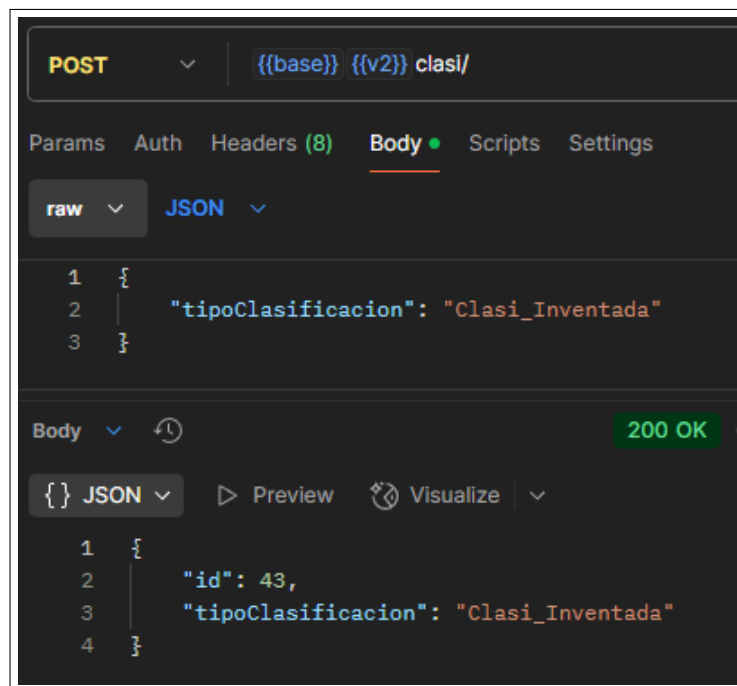


Figure 4: Clasificación Post - `/api/v2/clasi/`

## Clasificación - PUT

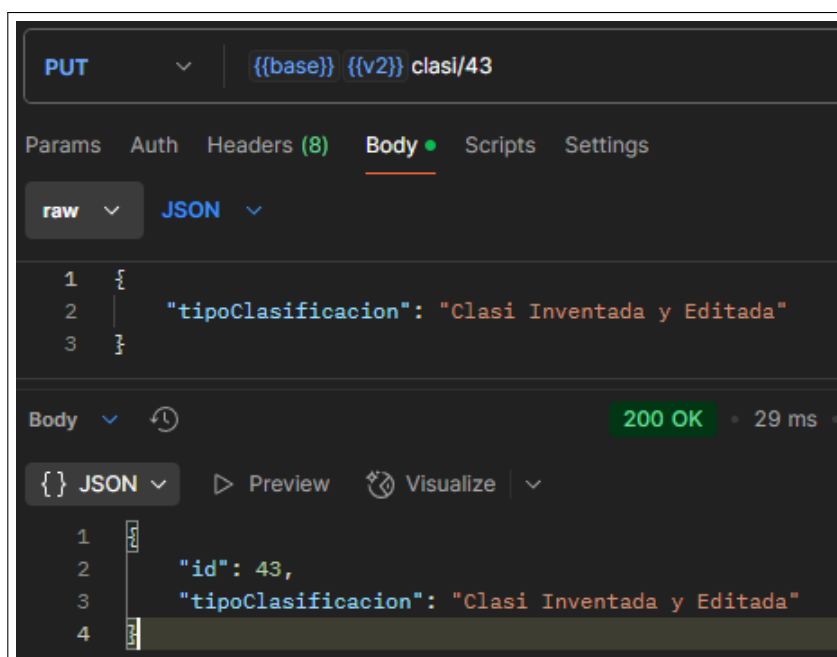


Figure 5: Clasificación Put - `/api/v2/clasi/{id}`

## Clasificación - DELETE

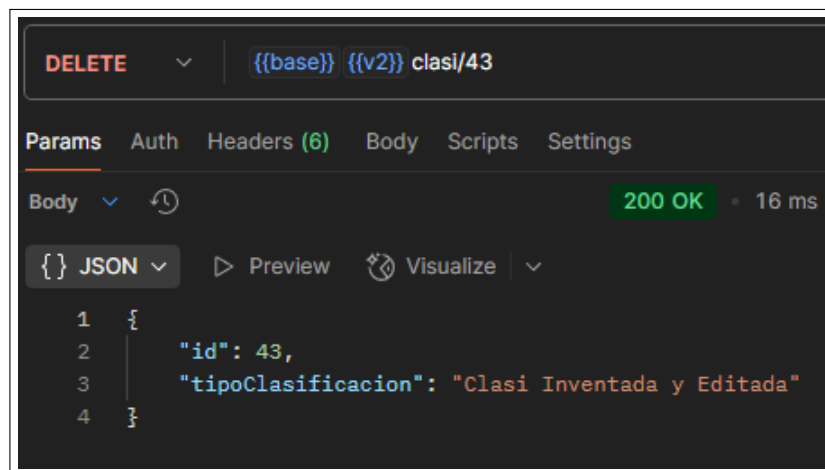


Figure 6: Clasificación Delete - /api/v2/clasi/{id}

### 2.1.2 Resultados de Postman para Editorial

#### Editorial - GET - FindAll

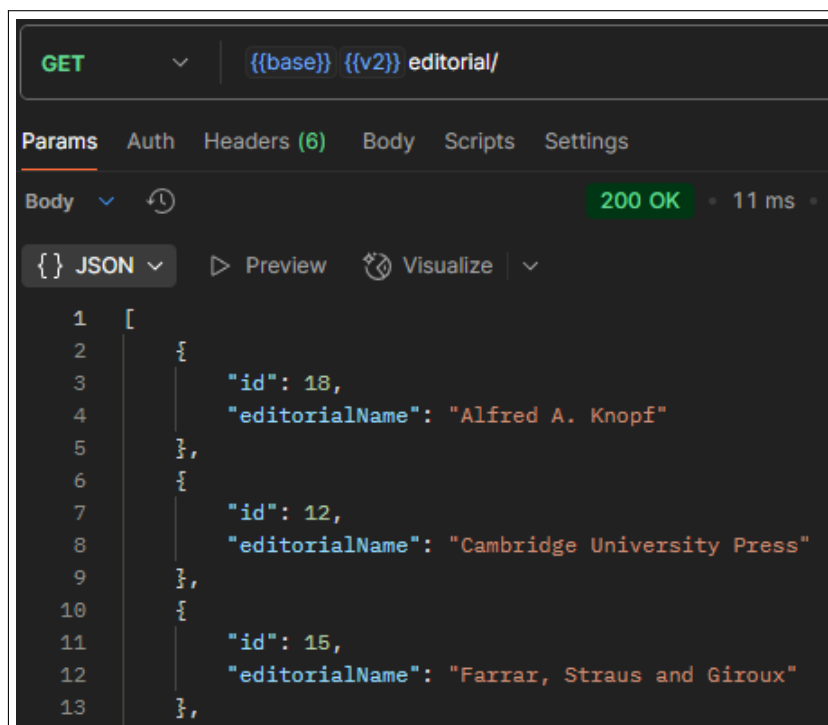


Figure 7: Editorial FindAll - /api/v2/editorial/

## Editorial - GET - FindById

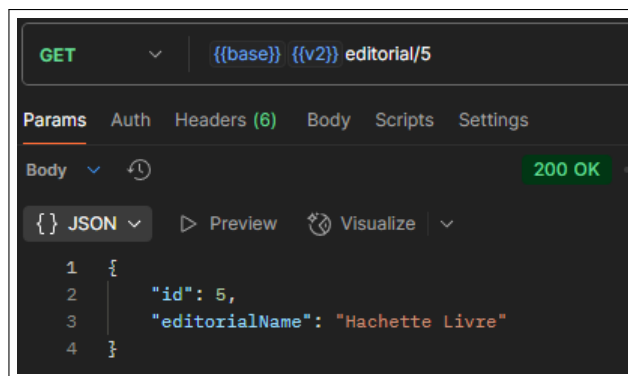


Figure 8: Editorial FindById - `/api/v2/editorial/{id}`

## Editorial - POST

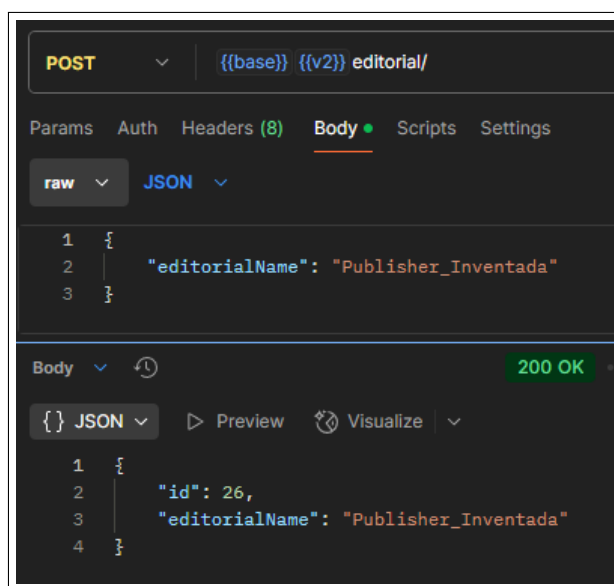


Figure 9: Editorial Post - `/api/v2/editorial/`

## Editorial - PUT

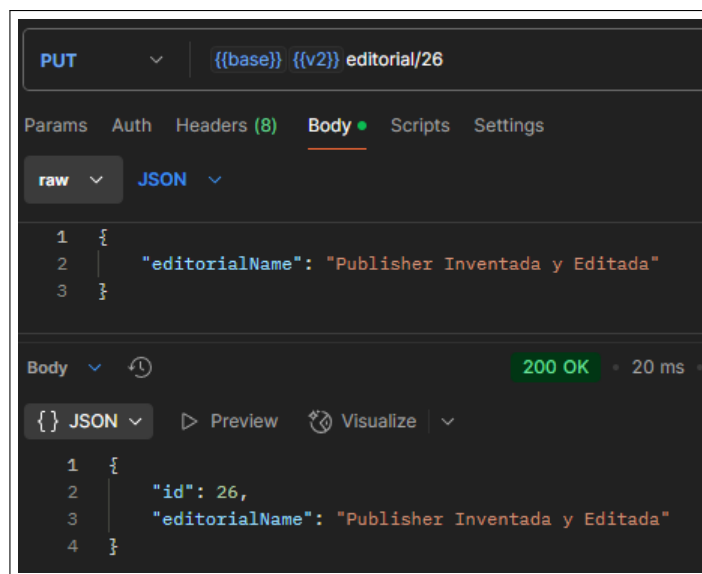


Figure 10: Editorial Put - `/api/v2/editorial/{id}`

## Editorial - DELETE

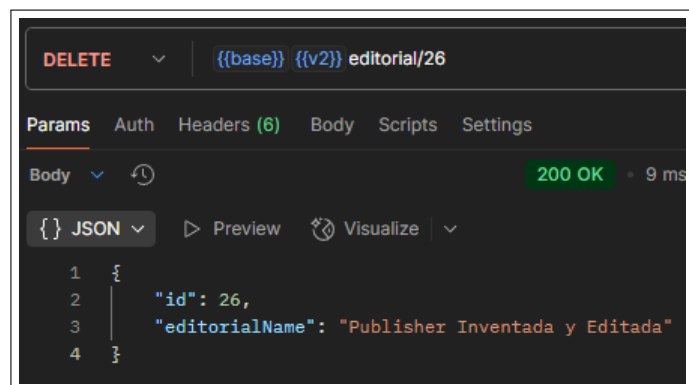


Figure 11: Editorial Delete - `/api/v2/editorial/{id}`

### 2.1.3 Resultados de Postman para Autor

#### Autor - GET - FindAll

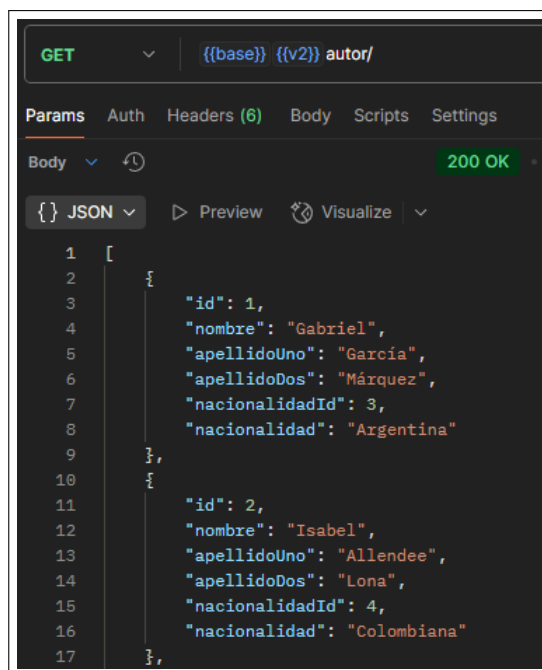


Figure 12: Autor FindAll - `/api/v2/autor/`

#### Autor - GET - FindById

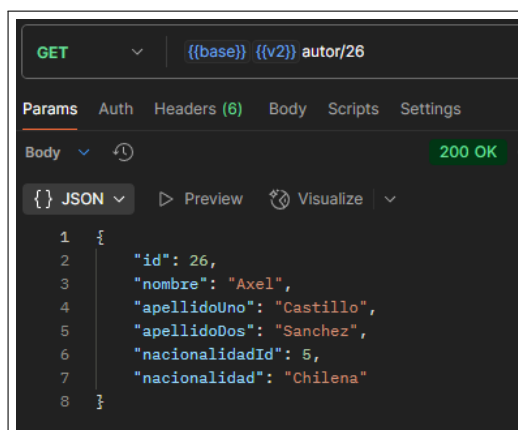


Figure 13: Autor FindById - `/api/v2/autor/{id}`

## Autor - POST

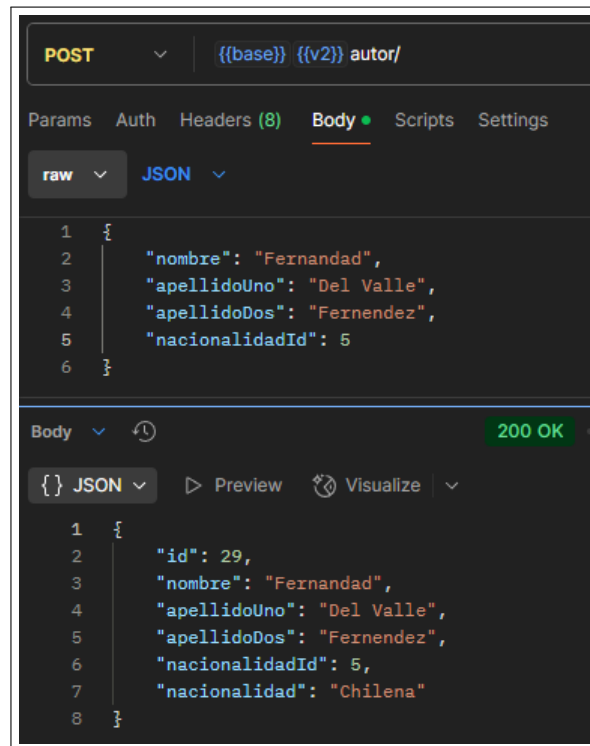


Figure 14: Autor Post - /api/v2/autor/

## Autor - PUT

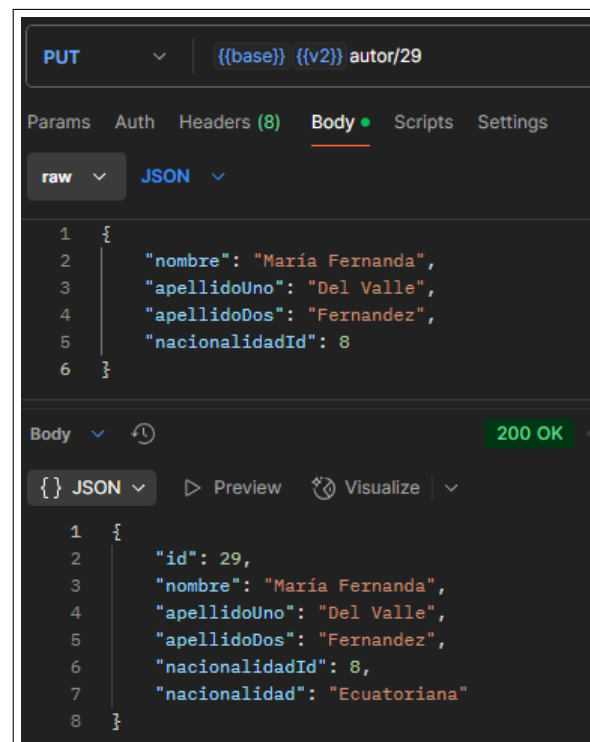


Figure 15: Autor Put - /api/v2/autor/{id}



## Autor - DELETE

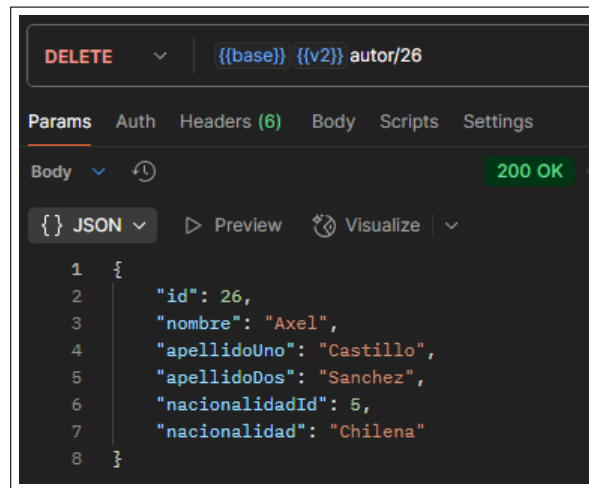


Figure 16: Autor Delete - `/api/v2/autor/{id}`

### 2.1.4 Resultados de Postman para Nacionalidad

#### Nacionalidad - GET - FindAll

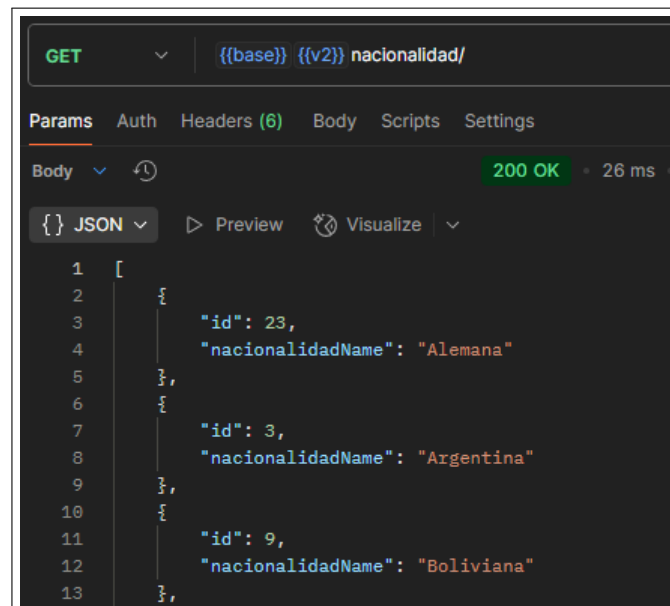


Figure 17: Nacionalidad FindAll - `/api/v2/nacionalidad/`

#### Nacionalidad - GET - FindById

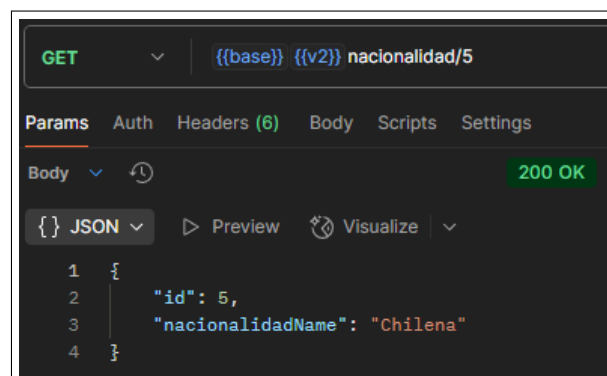


Figure 18: Nacionalidad FindById - `/api/v2/nacionalidad/{id}`

## Nacionalidad - POST

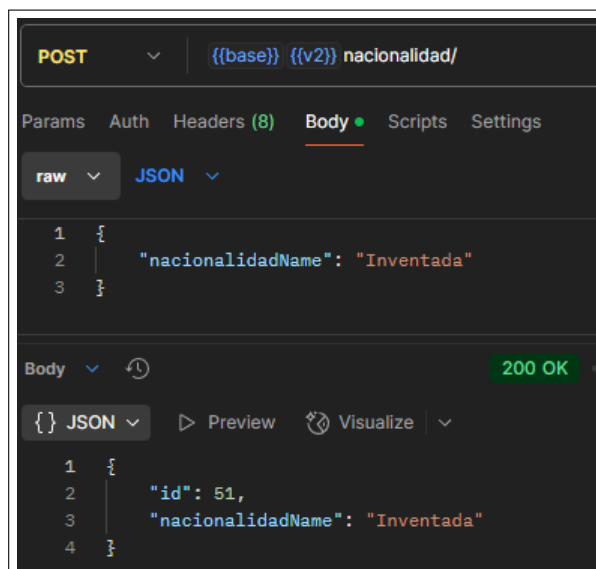


Figure 19: Nacionalidad Post - /api/v2/nacionalidad/

## Nacionalidad - PUT

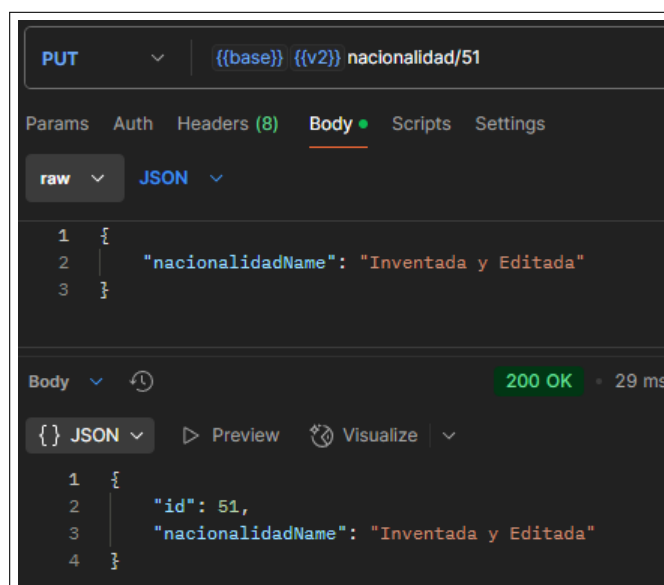


Figure 20: Nacionalidad Put - /api/v2/nacionalidad/

## Nacionalidad - DELETE

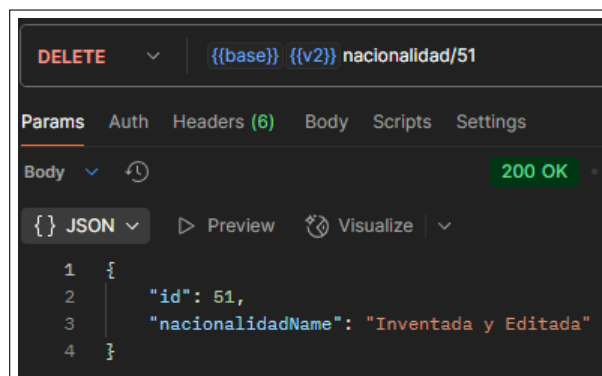


Figure 21: Nacionalidad Delete - /api/v2/nacionalidad/

### 2.1.5 Resultados de Postman para Libro

#### Libro - GET - FindAll

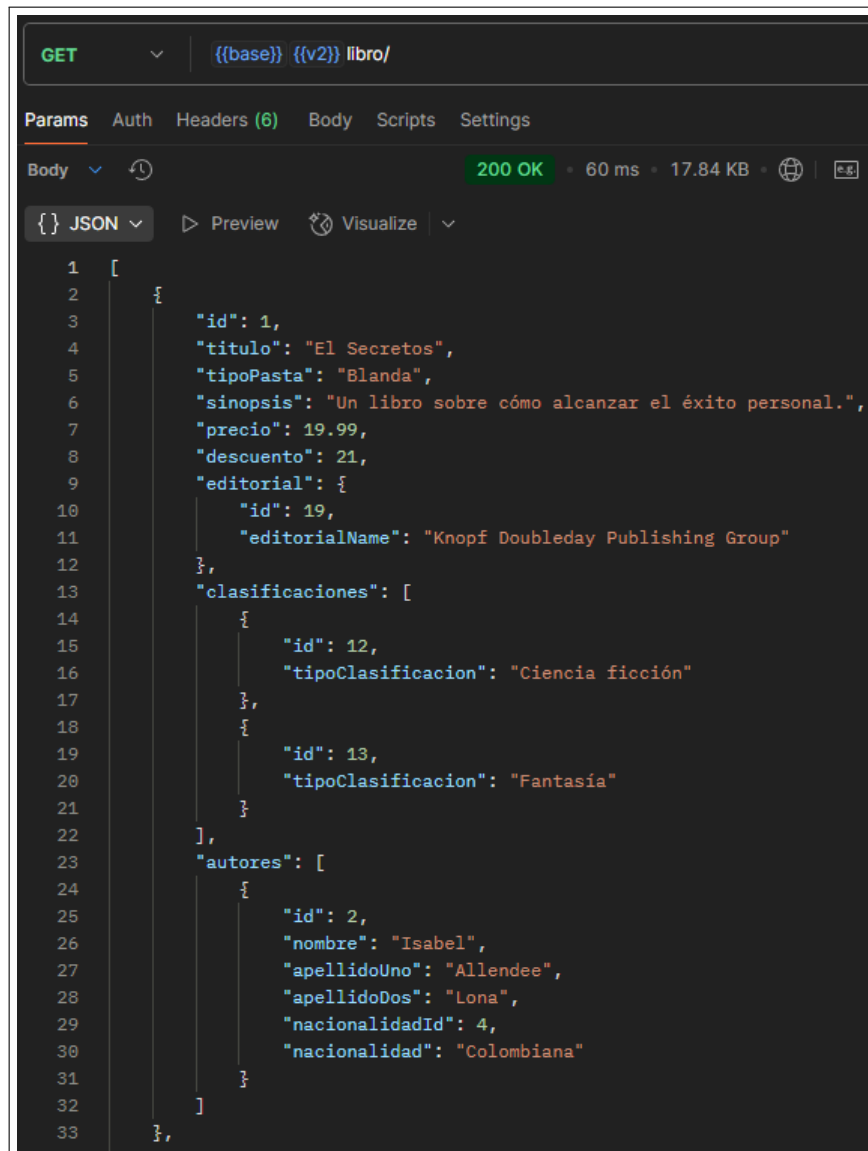
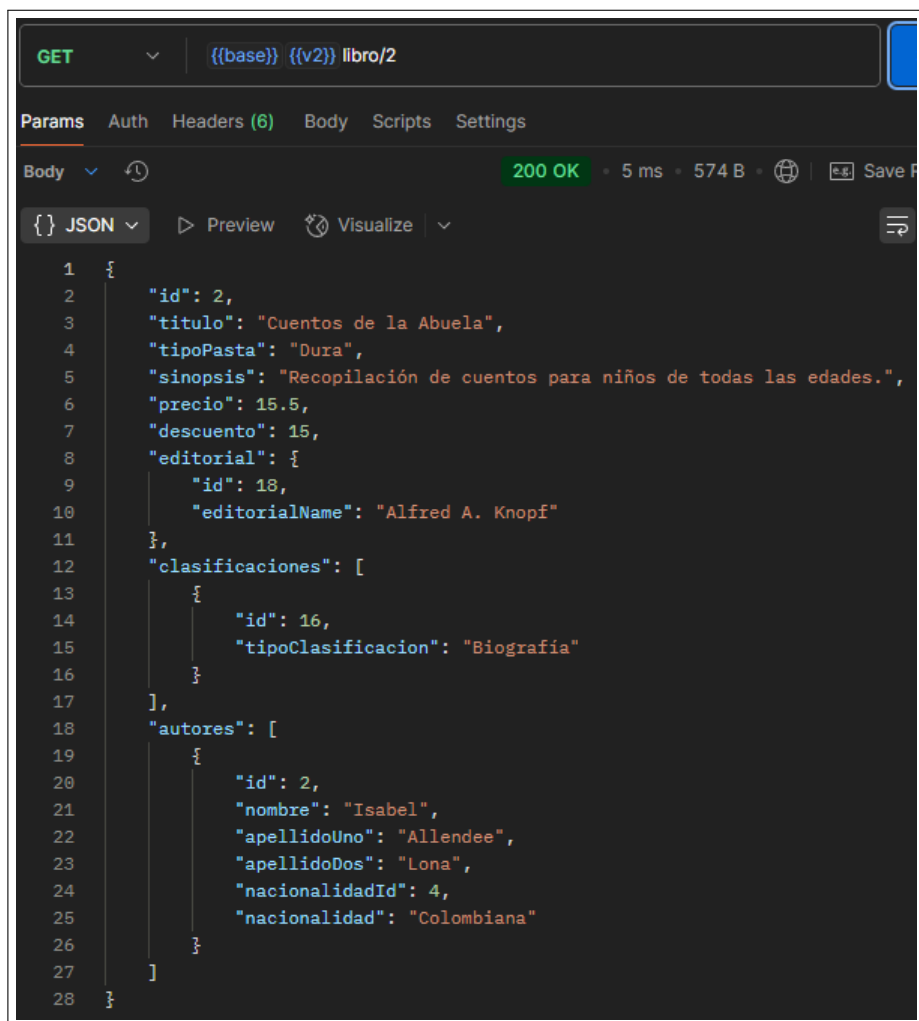


Figure 22: Libro FindAll - /api/v2/libro/

## Libro - GET - FindById



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** {{base}} {{v2}} libro/2
- Params:** Auth, Headers (6), Body, Scripts, Settings
- Status:** 200 OK, 5 ms, 574 B
- Body:** JSON
- Response:**

```
1 {
2   "id": 2,
3   "titulo": "Cuentos de la Abuela",
4   "tipoPasta": "Dura",
5   "sinopsis": "Recopilación de cuentos para niños de todas las edades.",
6   "precio": 15.5,
7   "descuento": 15,
8   "editorial": {
9     "id": 18,
10    "editorialName": "Alfred A. Knopf"
11  },
12  "clasificaciones": [
13    {
14      "id": 16,
15      "tipoClasificacion": "Biografía"
16    }
17  ],
18  "autores": [
19    {
20      "id": 2,
21      "nombre": "Isabel",
22      "apellidoUno": "Allendee",
23      "apellidoDos": "Lona",
24      "nacionalidadId": 4,
25      "nacionalidad": "Colombiana"
26    }
27  ]
28 }
```

Figure 23: Libro FindById - /api/v2/libro/{id}

## Libro - POST

The image shows a REST client interface with a POST request to the endpoint `{{base}} {{v2}} libro/`. The request body is a JSON object. A red arrow points to the response, which is a 200 OK status with a JSON body.

**Request Body:**

```
1 {
2   "titulo": "Nuevo Libro",
3   "tipoPasta": "Blanda",
4   "sinopsis": "Libro de prueba",
5   "precio": 19.99,
6   "descuento": 21,
7   "editorial": {
8     "id": 19
9   },
10  "clasificaciones": [
11    {
12      "id": 12
13    },
14    {
15      "id": 13
16    }
17  ],
18  "autores": [
19    {
20      "id": 2
21    }
22  ]
23 }
```

**Response Body (200 OK):**

```
1 {
2   "id": 45,
3   "titulo": "Nuevo Libro",
4   "tipoPasta": "Blanda",
5   "sinopsis": "Libro de prueba",
6   "precio": 19.99,
7   "descuento": 21,
8   "editorial": {
9     "id": 19,
10    "editorialName": null
11  },
12  "clasificaciones": [
13    {
14      "id": 12,
15      "tipoClasificacion": null
16    },
17    {
18      "id": 13,
19      "tipoClasificacion": null
20    }
21  ],
22  "autores": [
23    {
24      "id": 2,
25      "nombre": null,
26      "apellidoUno": null,
27      "apellidoDos": null,
28      "nacionalidadId": null,
29      "nacionalidad": null
30    }
31  ]
32 }
```

Figure 24: Libro Post - `/api/v2/libro/`

## Libro - PUT

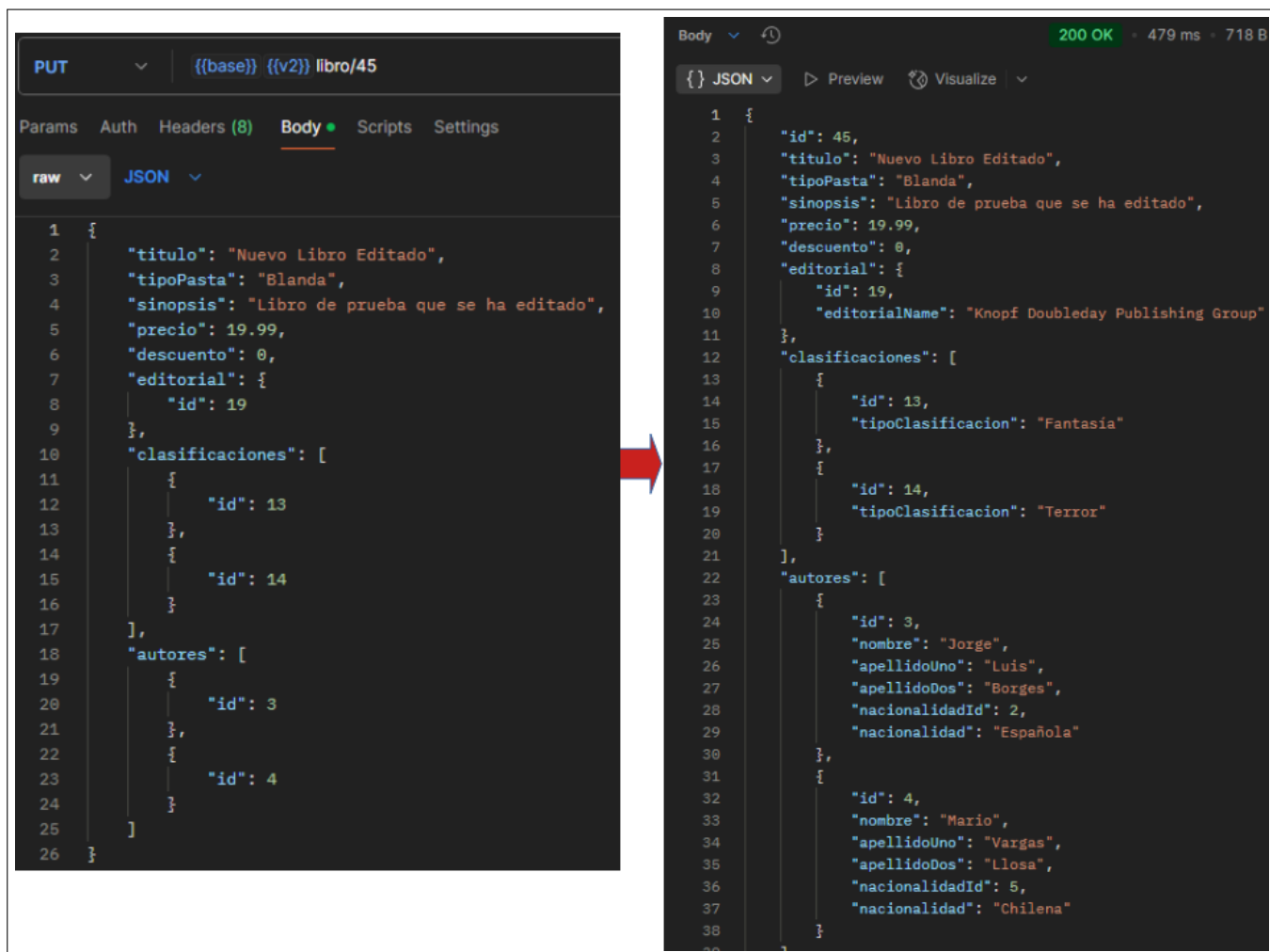


Figure 25: Libro Put - `/api/v2/libro/{id}`

## Libro - DELETE

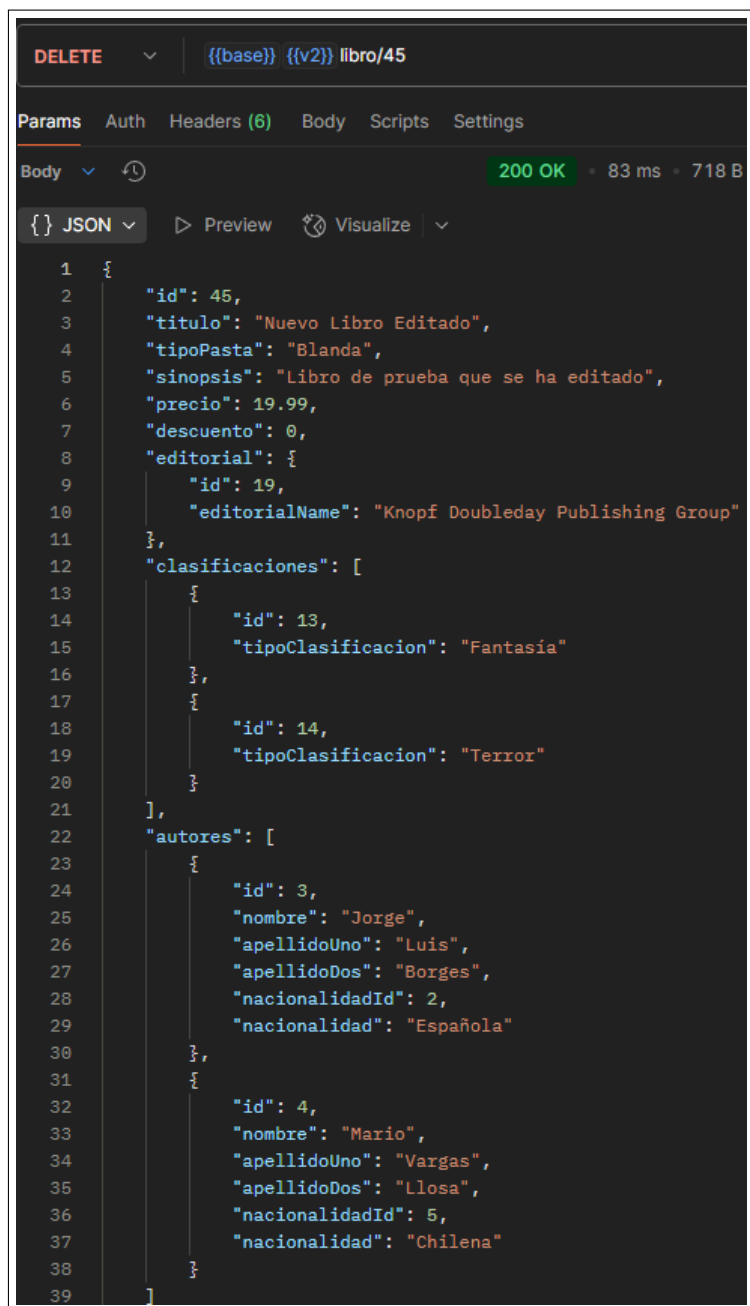


Figure 26: Libro Delete - `/api/v2/libro/{id}`

## 2.2 La API RESTful debe procesar el manejo de excepciones

Todas las tablas para este proyecto tiene el manejo de excepciones. Para demostrar que sí se implementó el manejo de excepciones, se mostrarán a continuación las capturas de pantalla para LIBRO

### 2.2.1 Exepciones de Libro

#### Libro - GET - findById - Excepciones

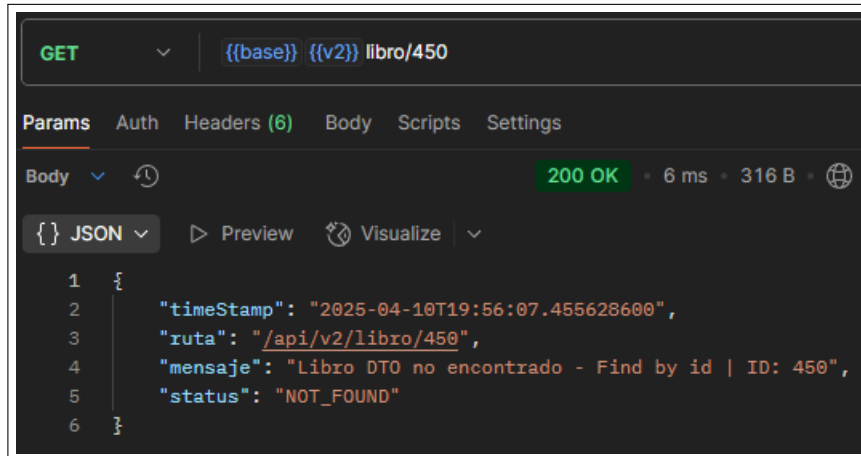


Figure 27: Libro FindAll - Excepciones - `/api/v2/libro/{id}`

#### Libro - POST - Todas las excepciones

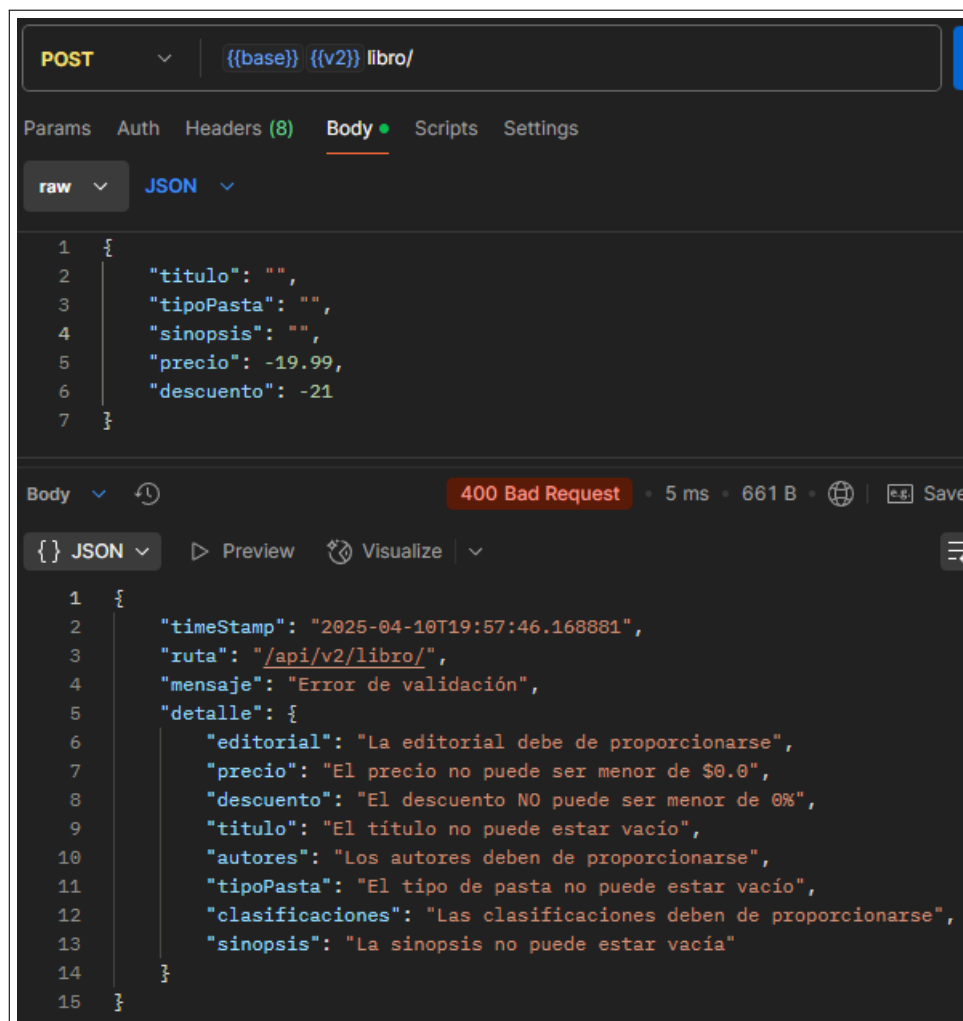


Figure 28: Libro Post - Todas las excepciones - `/api/v2/libro/`



## Libro - POST - Algunas excepciones

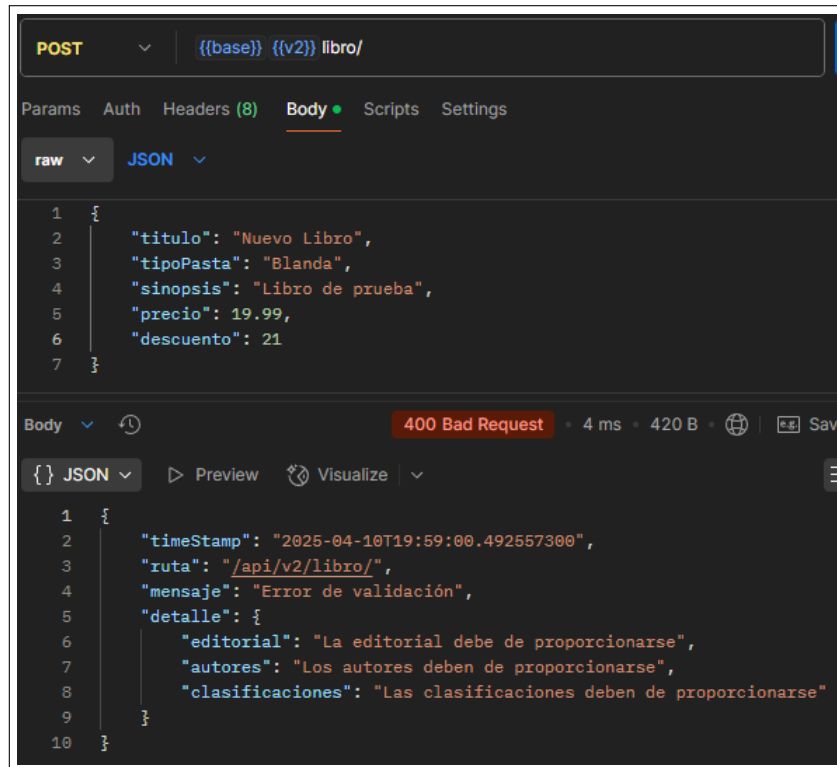


Figure 29: Libro Post - Algunas excepciones - `/api/v2/libro/`

## Libro - PUT - Excepciones

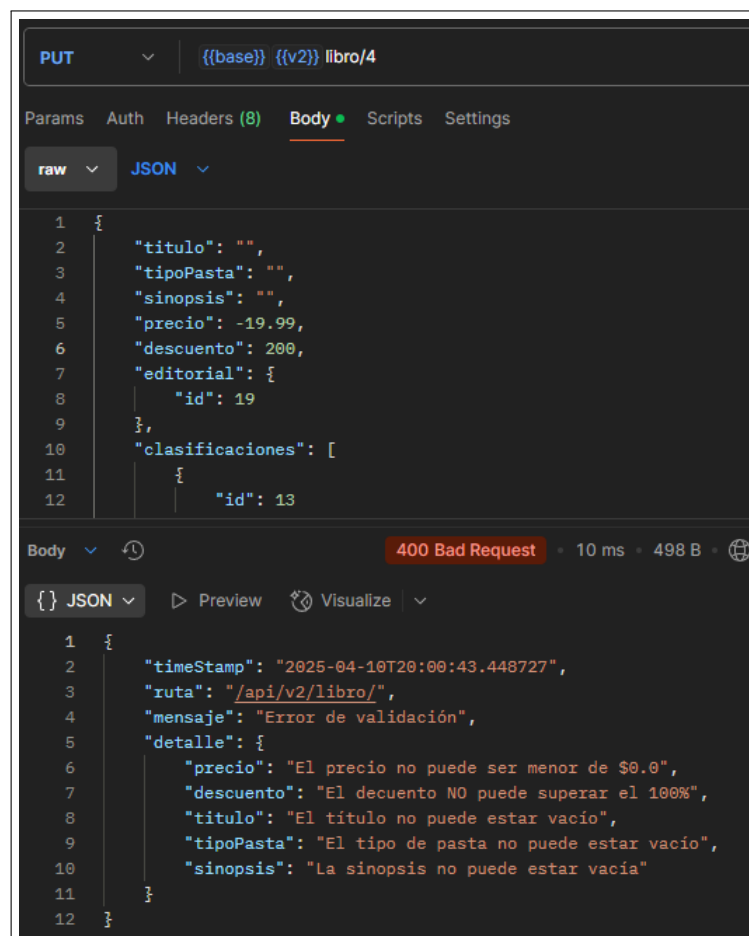


Figure 30: Libro Put - Excepciones - `/api/v2/libro/{id}`

## Libro - DELETE - Excepciones

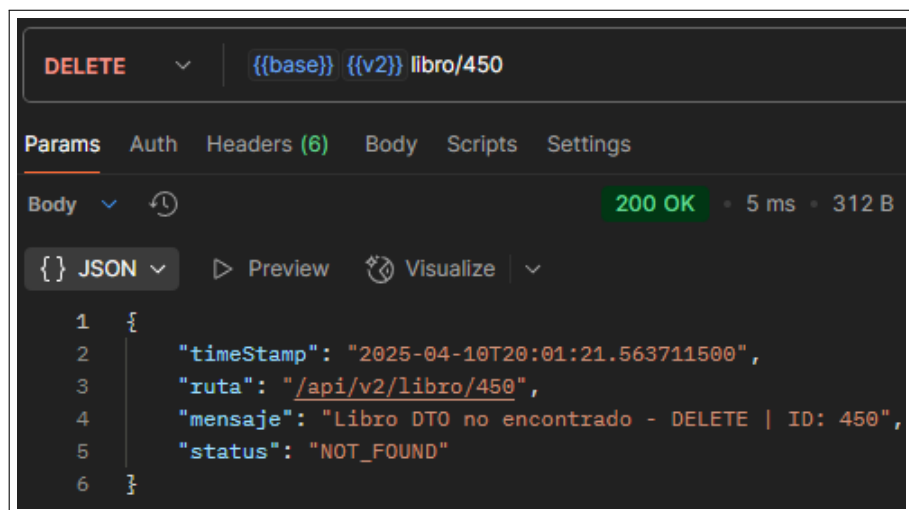


Figure 31: Libro Delete - Excepciones - /api/v2/libro/{id}