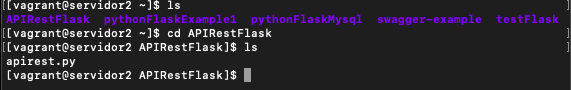
**Clonamos el repo en:**

/home/vagrant

git clone <https://github.com/omondragon/APIRestFlask>

cd APIRestFlask

ls



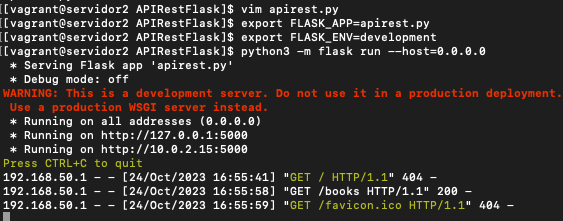
vim apirest.py

**EJECUTAR EL CODIGO**

export FLASK\_APP=apirest.py

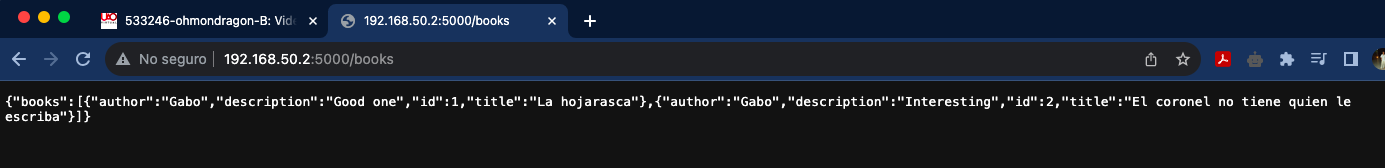
export FLASK\_ENV=development

python3 -m flask run --host=0.0.0.0



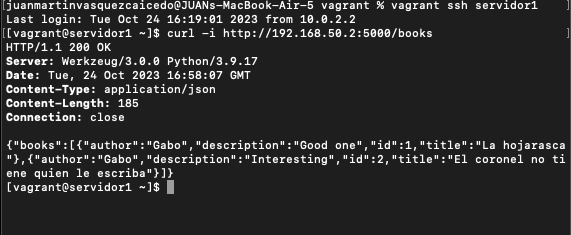
**Hacemos prueba con:**

[**http://192.168.50.2:5000/books**](http://192.168.50.2:5000/books)

****

**Hacemos prueba con:**

curl -i http://192.168.50.2:5000/books

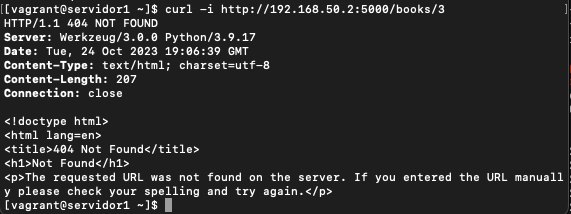
****

curl -i <http://192.168.50.2:5000/books/2>

****

**PROBAR LIBRO NO EXISTENTE**

curl -i <http://192.168.50.2:5000/books/3>

****

**POSTMAN**

**Instalamos postman**

**Creamos una coleccion:**

API REST Flask

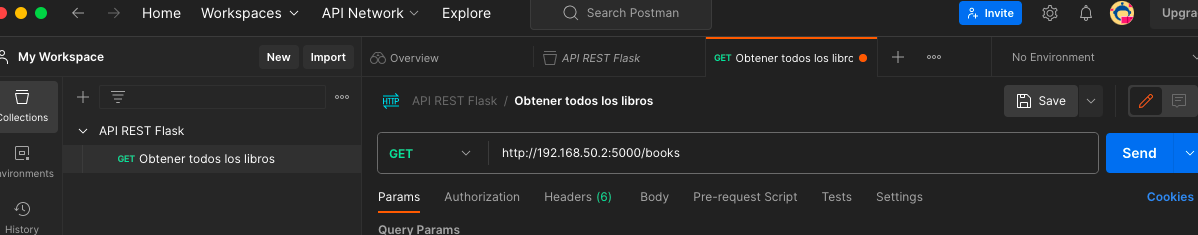
**Agregamos solicitudes a la coleccion:**

Obtener todos los libros

Nos aseguramos que el método sea GET

Colocamos la URL de la API

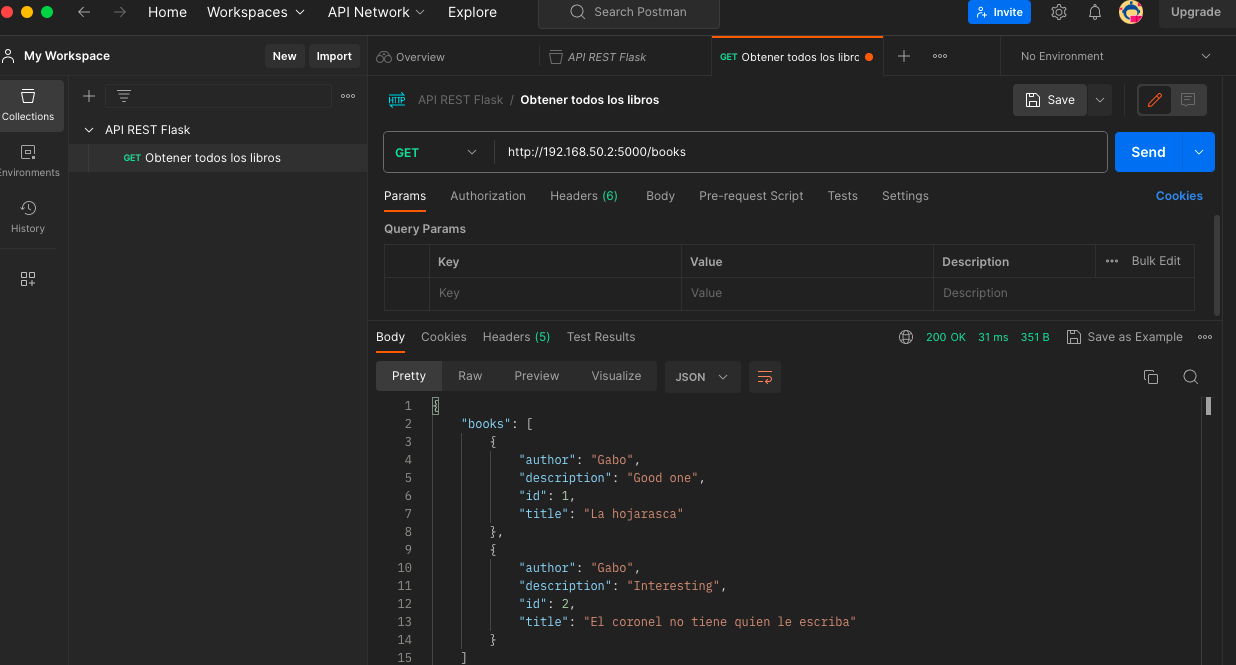
<http://192.168.50.2:5000/books>



**Enviar la Solicitud**

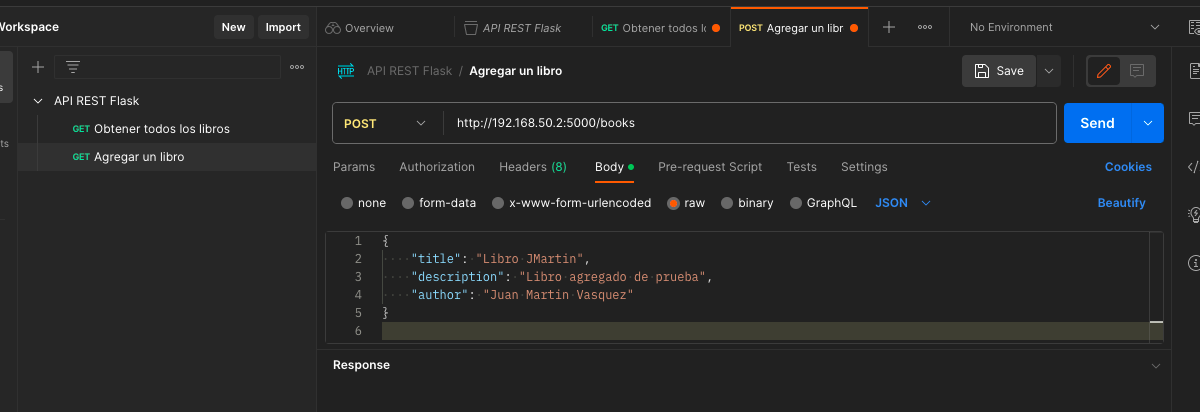
Hacemos clic en el botón "Send" (Enviar) para realizar la solicitud GET.

Vemos la respuesta de la API en la parte inferior de la ventana de Postman. Esta respuesta mostrará los libros disponibles en la API.

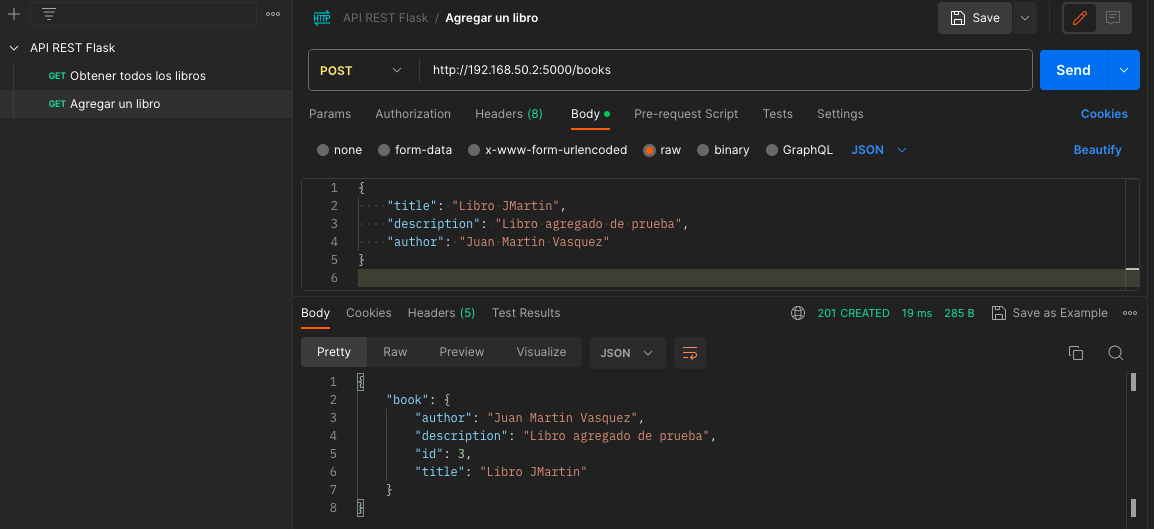


**Crear una solicitud POST para agregar un libro:**

* Nos ubicamos en la colección "API REST Flask" en Postman.
* Hacemos clic en "Add a request" y asignamos un nombre a la nueva solicitud, por ejemplo, "Agregar un libro".
* En el panel izquierdo de la solicitud, seleccionamos el método "POST".
* En la barra de URL, ingresa la URL de la API para agregar un libro: <http://192.168.50.2:5000/books>
* En la sección "Body" (Cuerpo), seleccionamos "raw" (en crudo) y elegimos "JSON (application/json)" en el menú desplegable.
* Ingresamos los datos del libro en formato JSON. Por ejemplo:

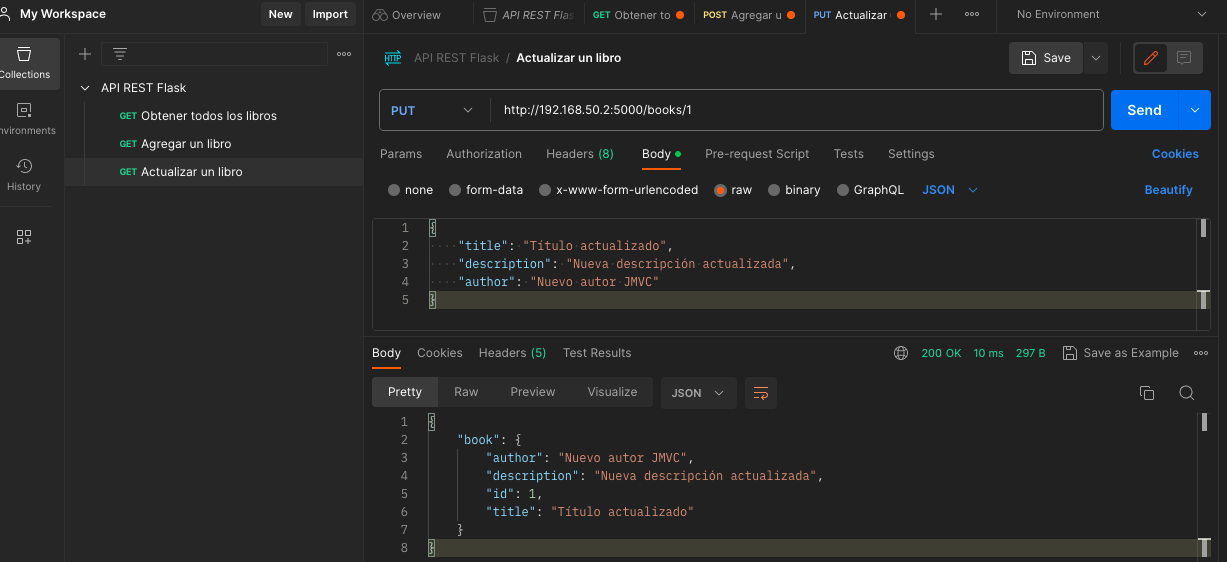


* Luego, hacemos clic en "Send" (Enviar) para agregar el libro. Deberíamos recibir una respuesta que indica que el libro ha sido creado.



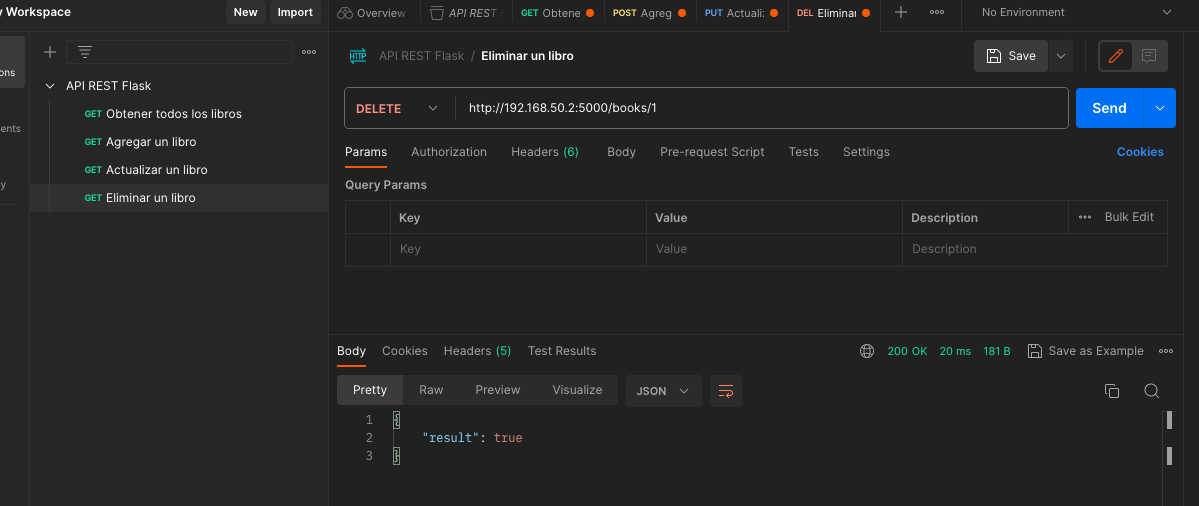
**Crear una solicitud PUT para actualizar un libro:**

* Creamos una nueva solicitud en la colección "API REST Flask" y la nómbramos, por ejemplo, "Actualizar un libro".
* Seleccionamos el método "PUT" en el panel izquierdo de la solicitud.
* En la barra de URL, ingresa la URL de la API para actualizar un libro, por ejemplo: <http://192.168.50.2:5000/books/1> (donde "1" es el ID del libro que deseamos actualizar).
* En la sección "Body" (Cuerpo), seleccionamos "raw" y elegimos "JSON (application/json)".
* Ingresa los datos actualizados del libro en formato JSON. Por ejemplo:

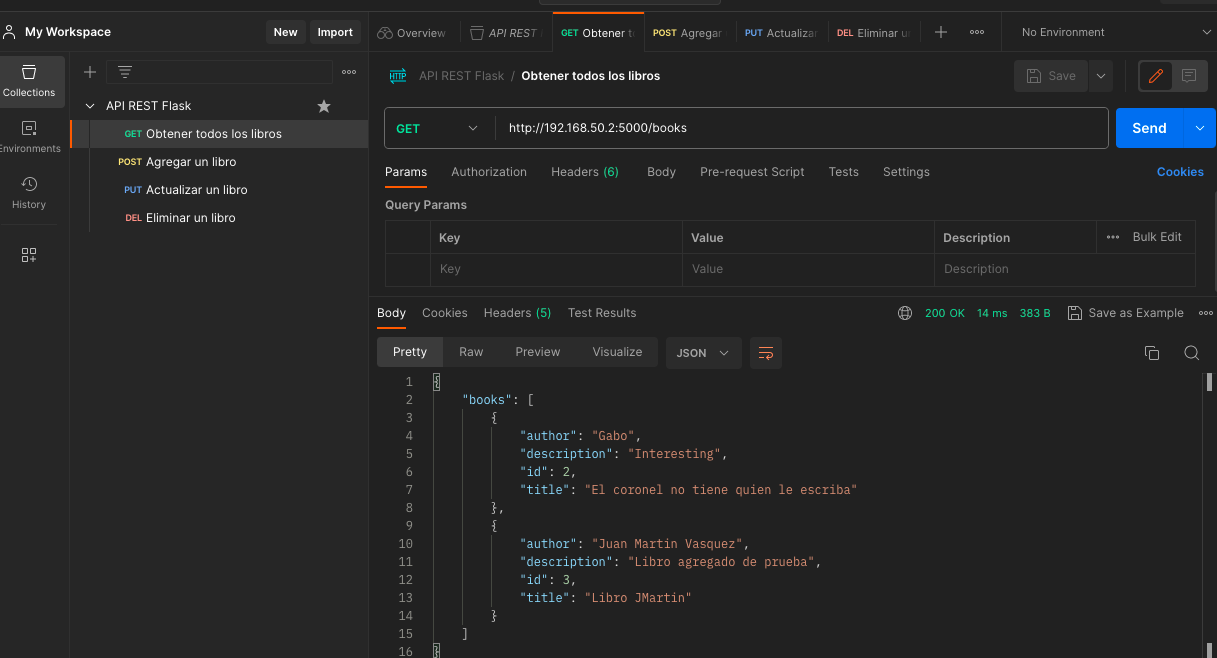


**Crear una solicitud DELETE para eliminar un libro:**

* Creamos una nueva solicitud en la colección "API REST Flask" y la nómbramos, por ejemplo, "Eliminar un libro".
* Seleccionamos el método "DELETE" en el panel izquierdo de la solicitud.
* En la barra de URL, ingresamos la URL de la API para eliminar un libro, por ejemplo: <http://192.168.50.2:5000/books/1> (donde "1" es el ID del libro que deseamos eliminar).
* Luego, hacemos clic en "Send" (Enviar) para eliminar el libro. Deberíamos recibir una respuesta que indica que el libro ha sido eliminado.



GUARDAMOS LA COLECCIÓN



**AHORA CON Node.js**

Instalamos Node.js y npm:

sudo dnf install nodejs

sudo dnf install npm

Instalamos Express y otras dependencias necesarias:

npm install express --save

npm install body-parser --save

Creamos el directorio del proyecto:

mkdir my-rest-api

cd my-rest-api

Creamos el archivo:

vim app.js

DENTRO DEL ARCHIVO:

const express = require('express');

const bodyParser = require('body-parser');

const app = express();

const port = 3000; // Puedes cambiar el puerto si lo deseas

app.use(bodyParser.json());

// Datos de ejemplo (una colección de libros)

let books = [

{

id: 1,

title: 'Libro 1',

author: 'Autor 1',

},

{

id: 2,

title: 'Libro 2',

author: 'Autor 2',

},

];

// Ruta para obtener todos los libros

app.get('/books', (req, res) => {

res.json(books);

});

// Ruta para obtener un libro por su ID

app.get('/books/:id', (req, res) => {

const bookId = parseInt(req.params.id);

const book = books.find((book) => book.id === bookId);

if (book) {

res.json(book);

} else {

res.status(404).send('Libro no encontrado');

}

});

// Ruta para agregar un nuevo libro

app.post('/books', (req, res) => {

const newBook = req.body;

newBook.id = books.length + 1;

books.push(newBook);

res.status(201).json(newBook);

});

// Ruta para actualizar un libro por su ID

app.put('/books/:id', (req, res) => {

const bookId = parseInt(req.params.id);

const book = books.find((book) => book.id === bookId);

if (book) {

Object.assign(book, req.body);

res.json(book);

} else {

res.status(404).send('Libro no encontrado');

}

});

// Ruta para eliminar un libro por su ID

app.delete('/books/:id', (req, res) => {

const bookId = parseInt(req.params.id);

const index = books.findIndex((book) => book.id === bookId);

if (index !== -1) {

books.splice(index, 1);

res.status(204).send('Libro eliminado');

} else {

res.status(404).send('Libro no encontrado');

}

});

app.listen(port, () => {

console.log(`Servidor escuchando en http://localhost:${port}`);

});

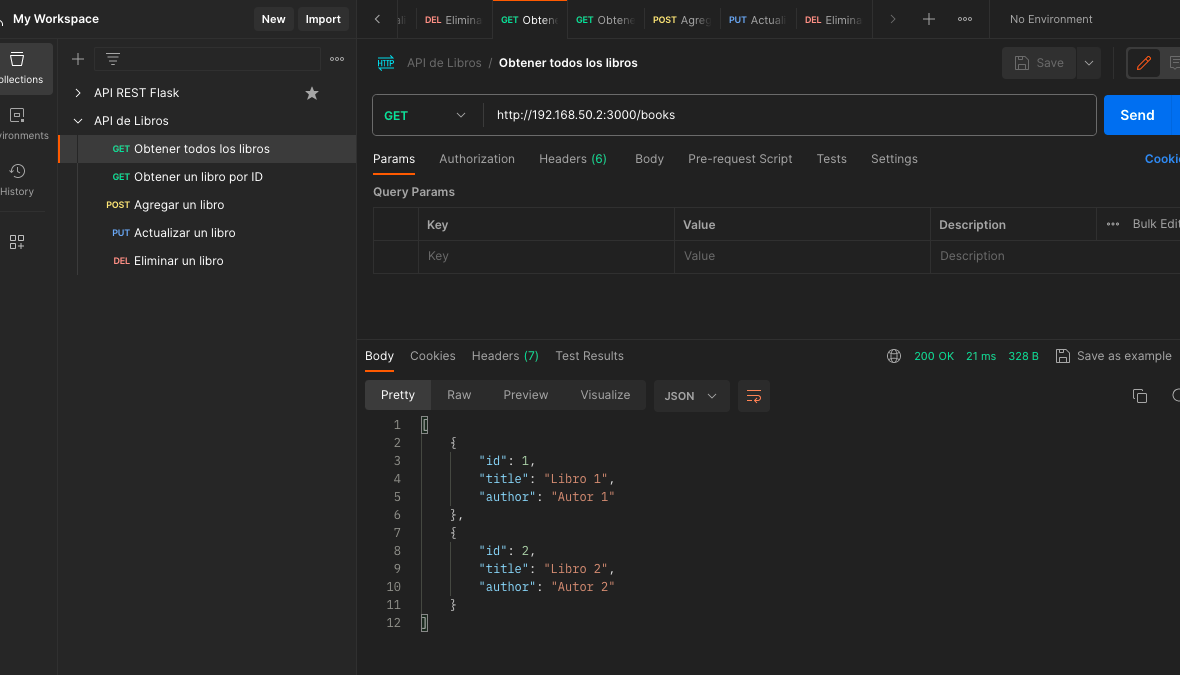
**PARA INCIAR EL SERVICIO:**

node app.js

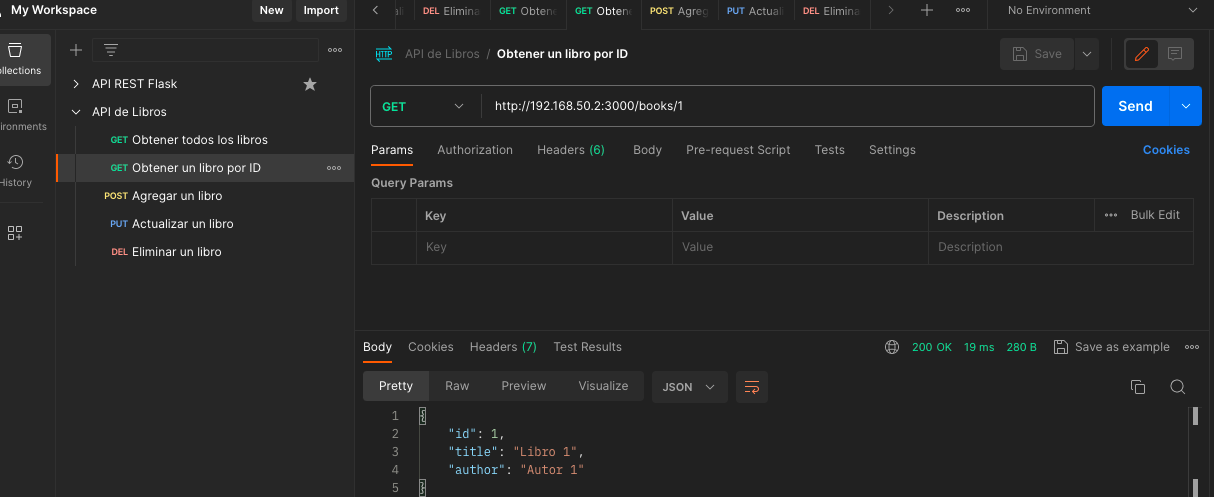
<http://192.168.50.2:3000/books>

**POSTMAN**

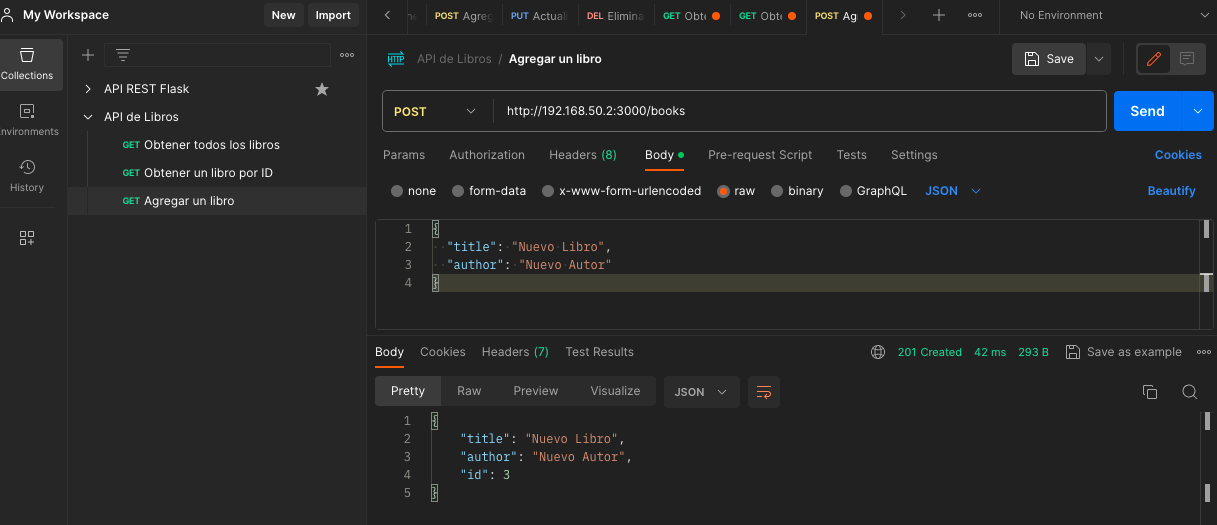
Obtener todos los libros



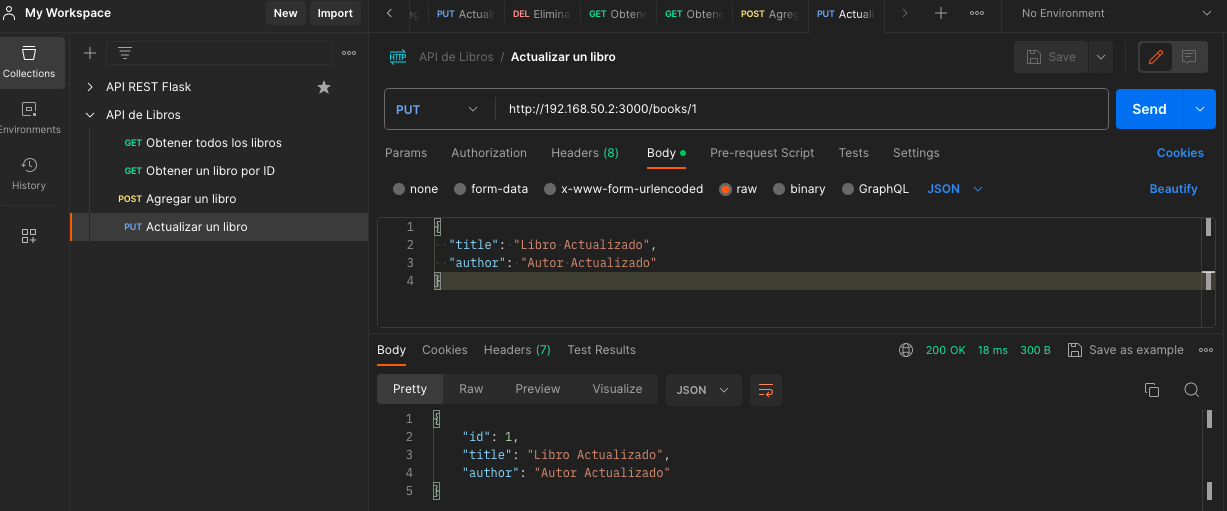
Obtener un libro por ID



Agregar un libro



Actualizar libro



Eliminar un libro

