Manual de Uso de la API

Este manual describe el funcionamiento de una API desarrollada con Flask y RethinkDB, que permite realizar operaciones CRUD sobre clientes, usuarios, documentos y casos legales (documentos). Además, la API almacena logs de las operaciones realizadas. A continuación, se detallan las rutas disponibles, los métodos HTTP correspondientes y la estructura de los datos esperados.

Configuración Inicial

Antes de utilizar la API, se debe asegurar que los siguientes paquetes estén instalados:

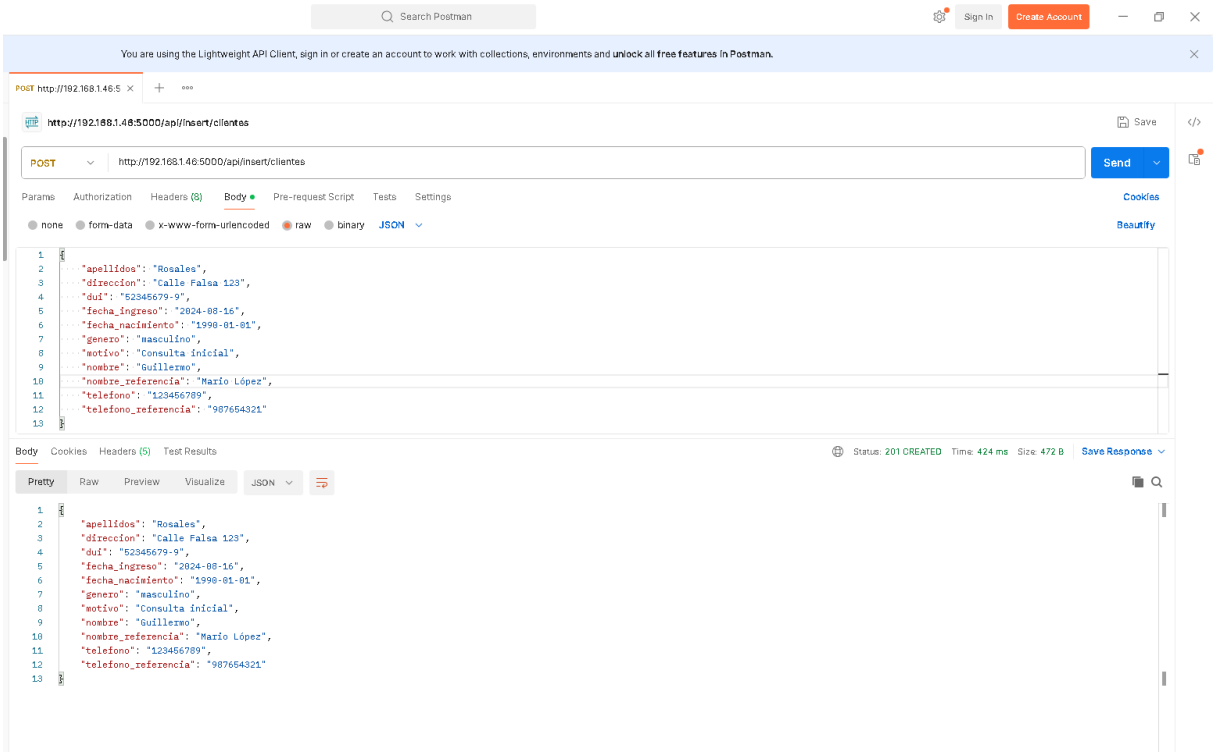
- pip install flask rethinkdb werkzeug

El archivo principal de la aplicación es “api.py”, que debe ser ejecutado para iniciar el servidor.

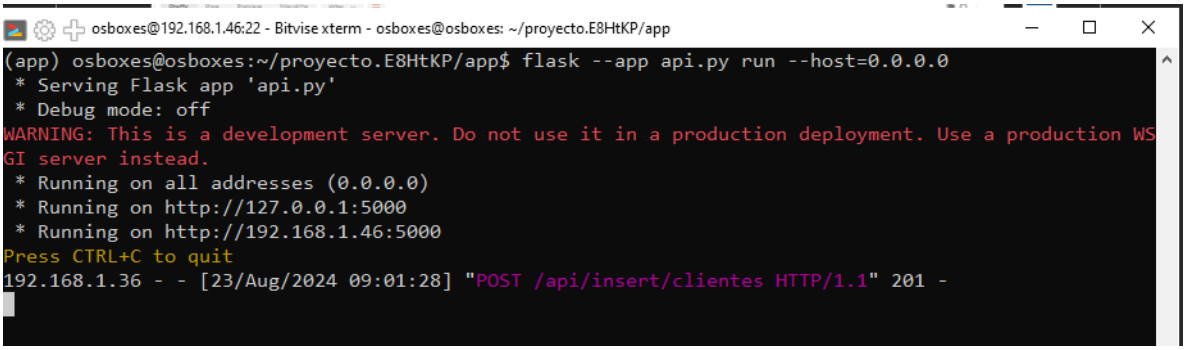
Rutas Disponibles

1. Insertar Cliente

- **URL:** /api/insert/clientes
- **Método:** POST
- **Descripción:** Inserta un nuevo cliente en la base de datos.
- **Datos Esperados (JSON)**

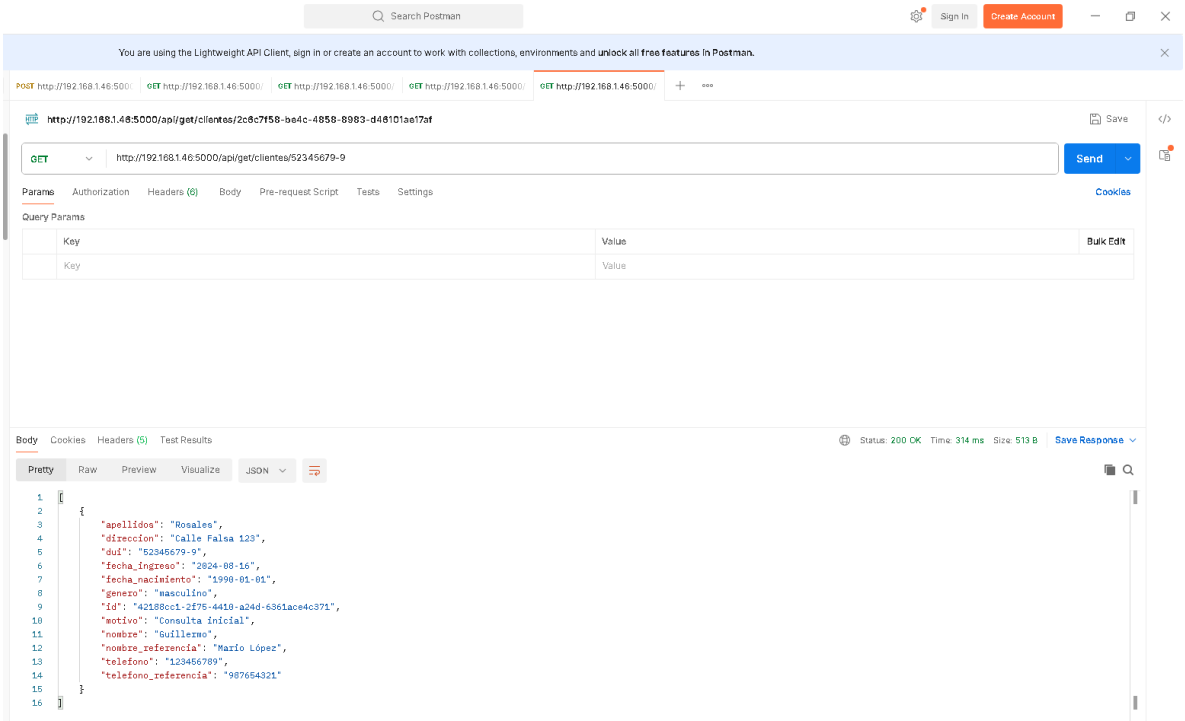


- **Respuesta:** Retorna un objeto JSON con los datos del cliente insertado y un código de estado 201.



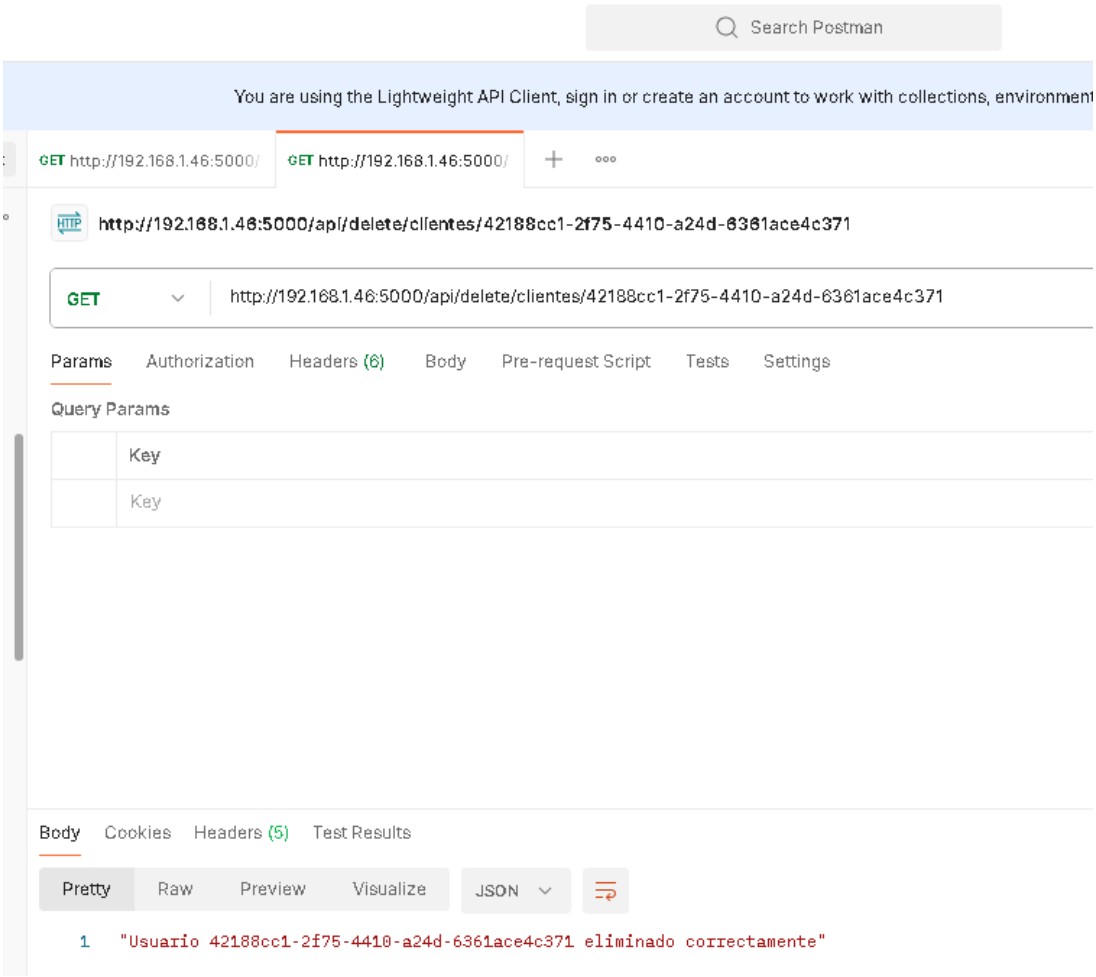
2. Obtener Cliente por ID (DUI)

- **URL:** /api/get/clientes/<id_cliente>
- **Método:** GET
- **Descripción:** Retorna los datos de un cliente específico según su DUI.
- **Parámetro URL:**
  - id_cliente: DUI del cliente.
- **Respuesta:** Un JSON con los datos del cliente.



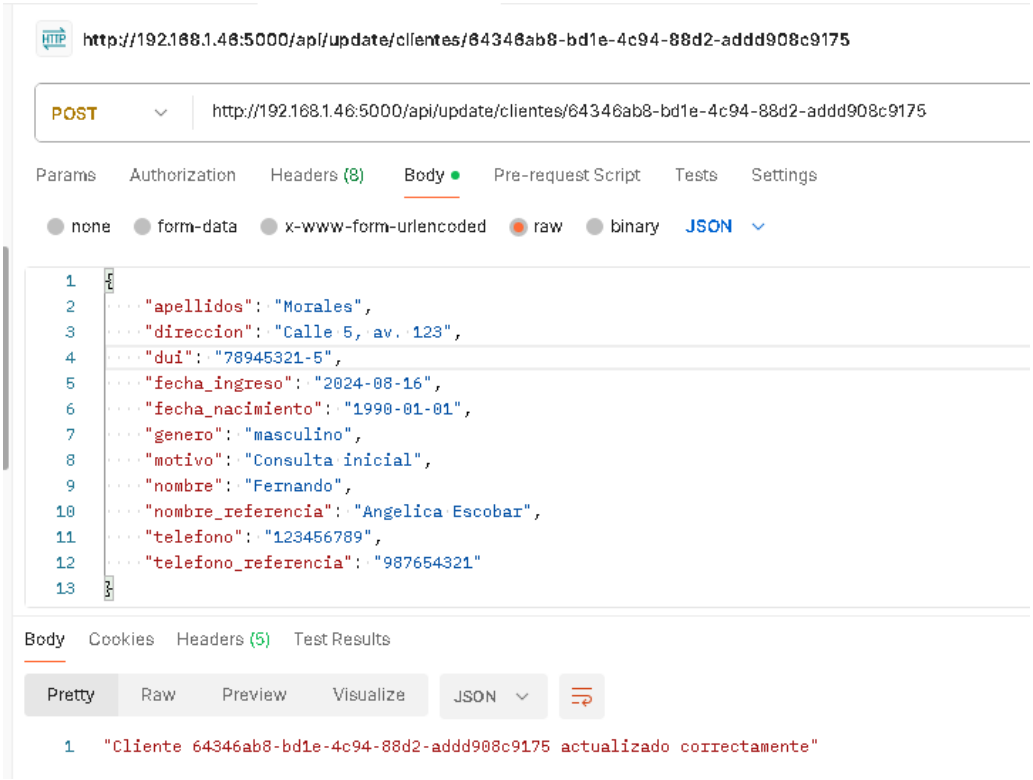
3. Eliminar Cliente por ID

- **URL:** /api/delete/clientes/<id_eliminar>
- **Método:** GET
- **Descripción:** Elimina un cliente de la base de datos según su número de ID.
- **Parámetro URL:**
  - id_eliminar: número de ID.
- **Respuesta:** Un mensaje personalizado de confirmación.



4. Actualizar Cliente por ID

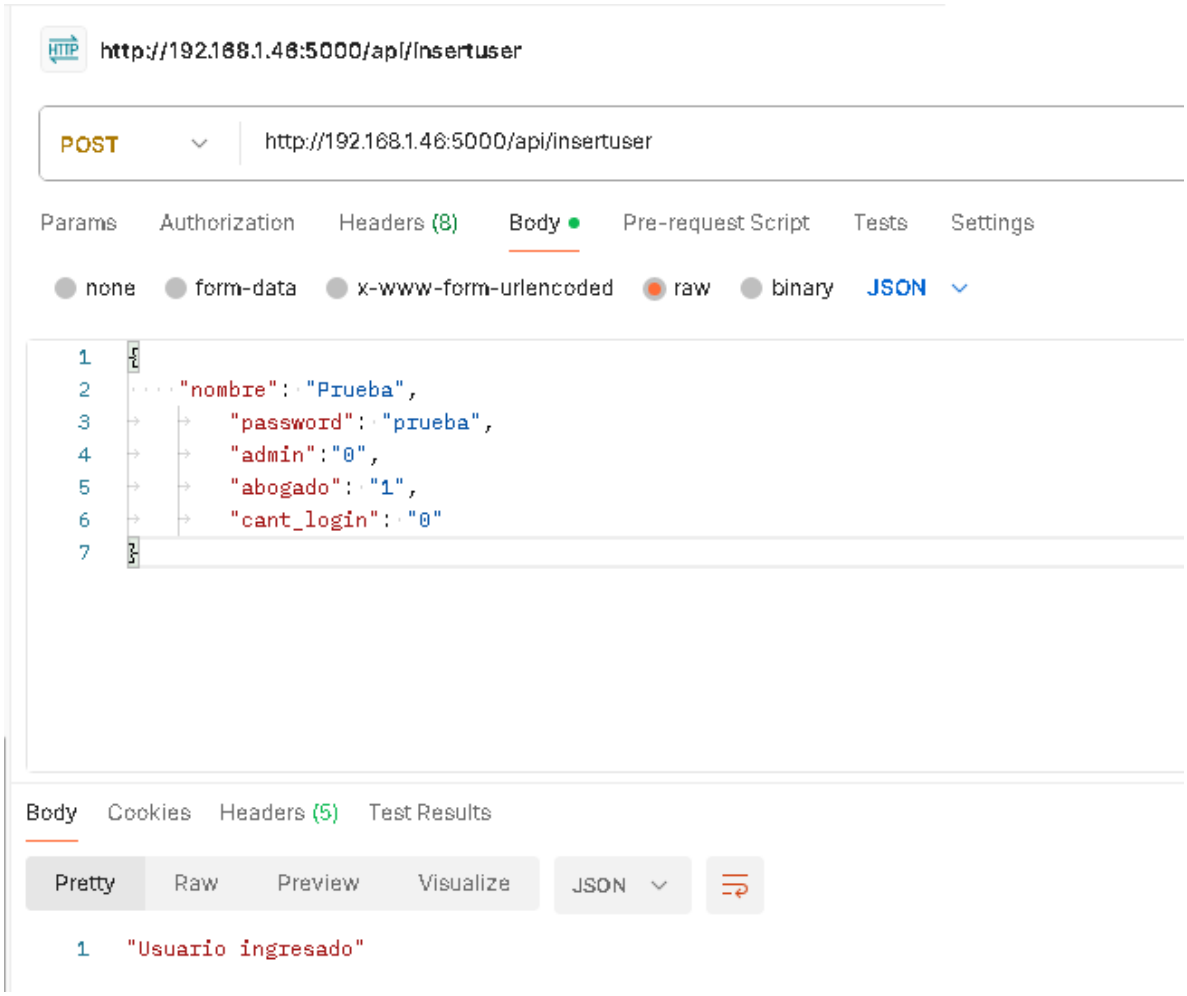
- **URL:** /api/update/clientes/<id_update>
- **Método:** POST
- **Descripción:** Actualiza los datos de un cliente existente.
- **Datos Esperados (JSON):**



- **Respuesta:** Se genera un mensaje personalizado de confirmación.

5. Insertar Usuario

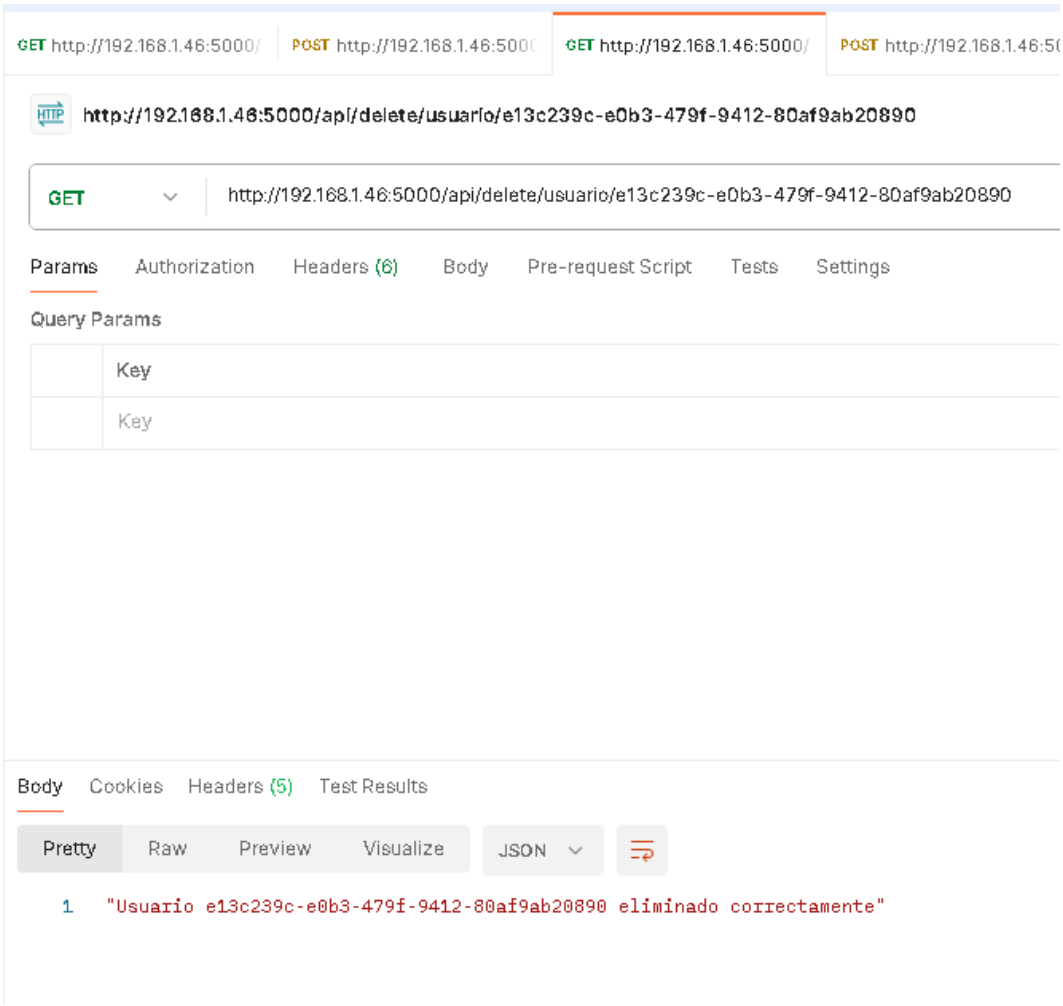
- **URL:** /api/insertuser
- **Método:** POST
- **Descripción:** Inserta un nuevo usuario en la base de datos.
- **Datos Esperados (JSON):**



- **Respuesta:** Un mensaje personalizado de confirmación.

6. Eliminar Usuario por ID

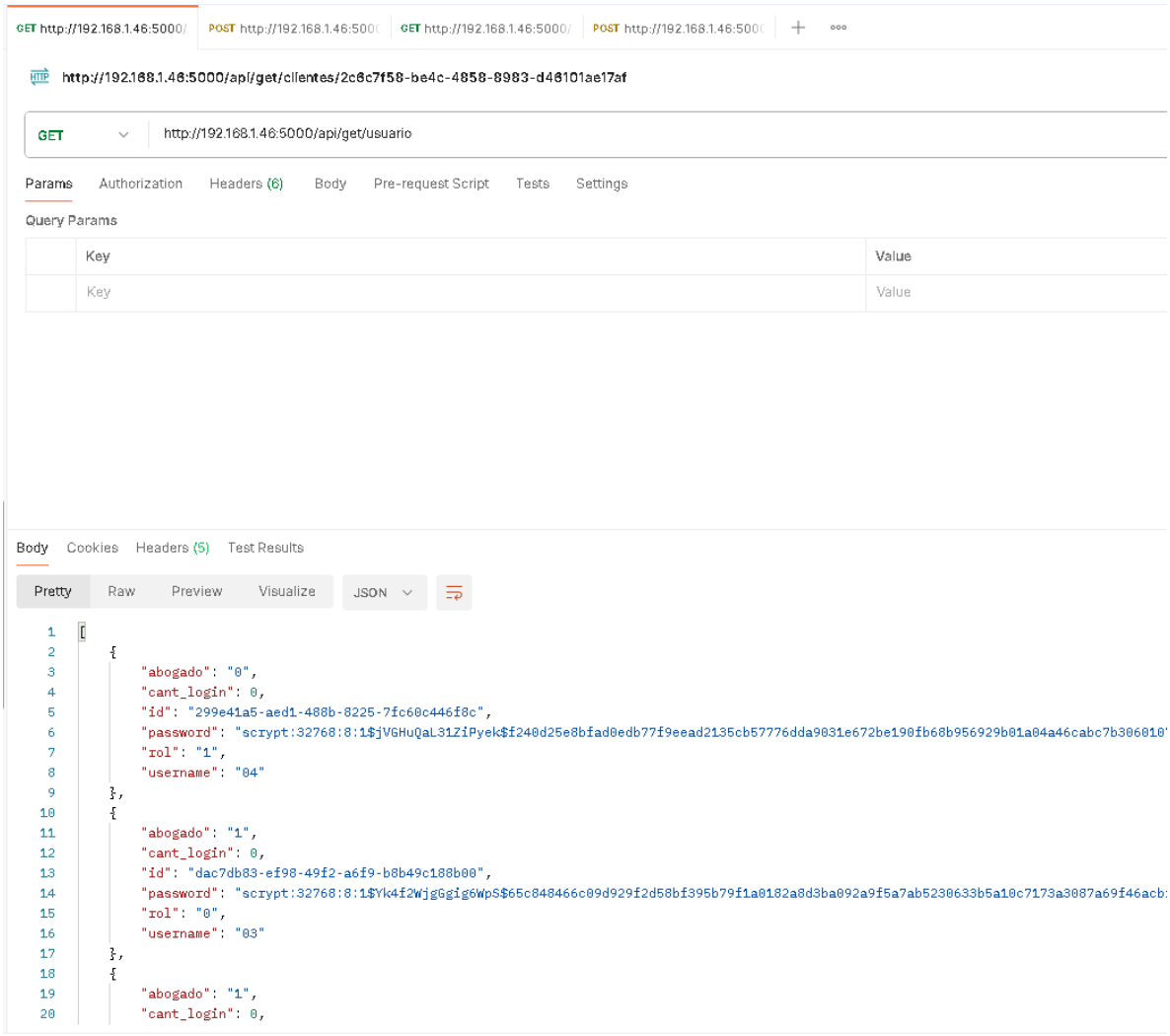
- **URL:** /api/delete/usuario/<id_deleteuser>
- **Método:** GET
- **Descripción:** Elimina un usuario de la base de datos según su número de ID.
- **Parámetro URL:**
    - id_deleteuser: numero de ID del usuario.



- **Respuesta:** Un mensaje personalizado de confirmación.

7. Obtener Todos los Usuarios

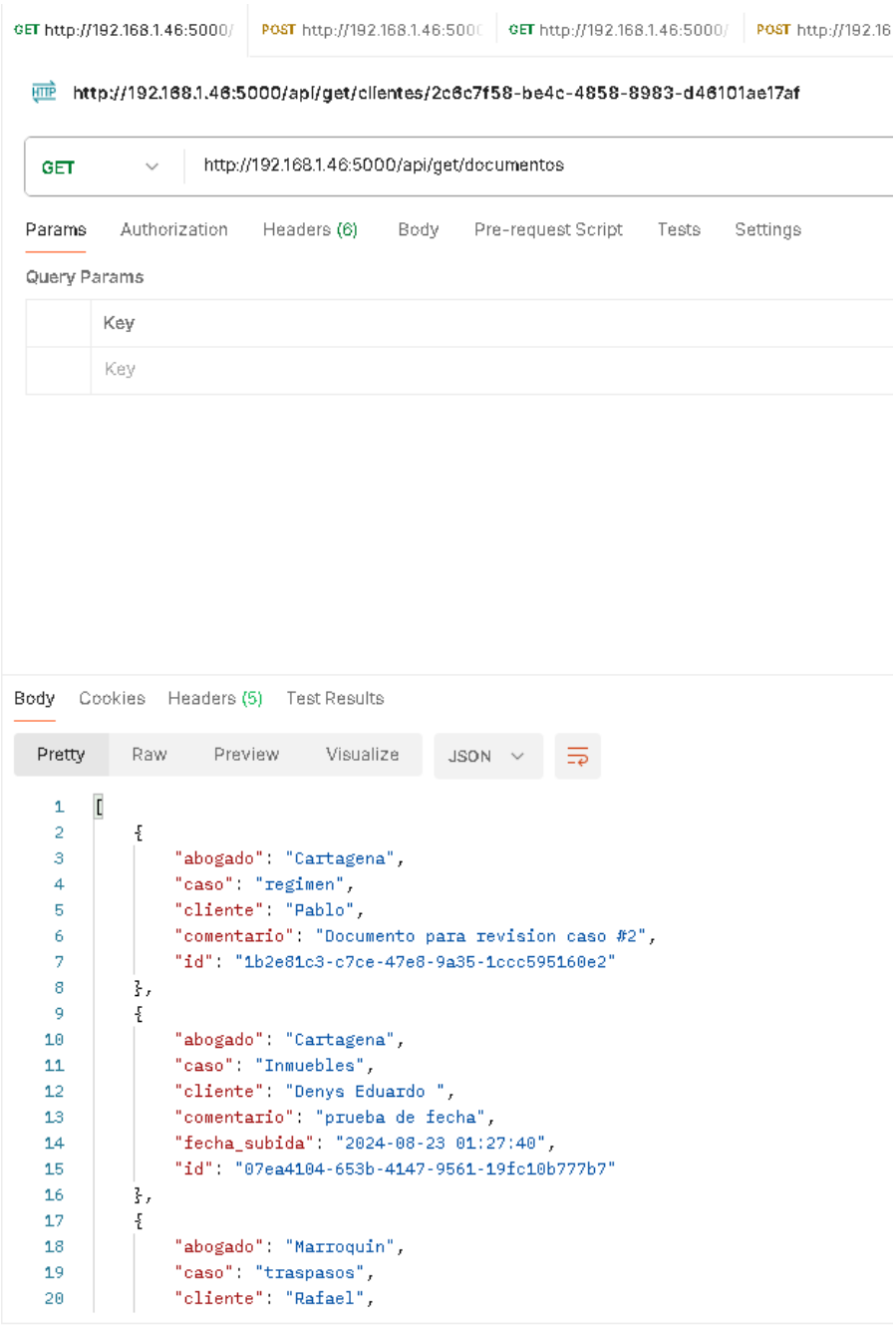
- **URL:** /api/get/usuario
- **Método:** GET
- **Descripción:** Retorna una lista de todos los usuarios en la base de datos.



- **Respuesta:** Un JSON con los datos de todos los usuarios.

8. Obtener Todos los Documentos

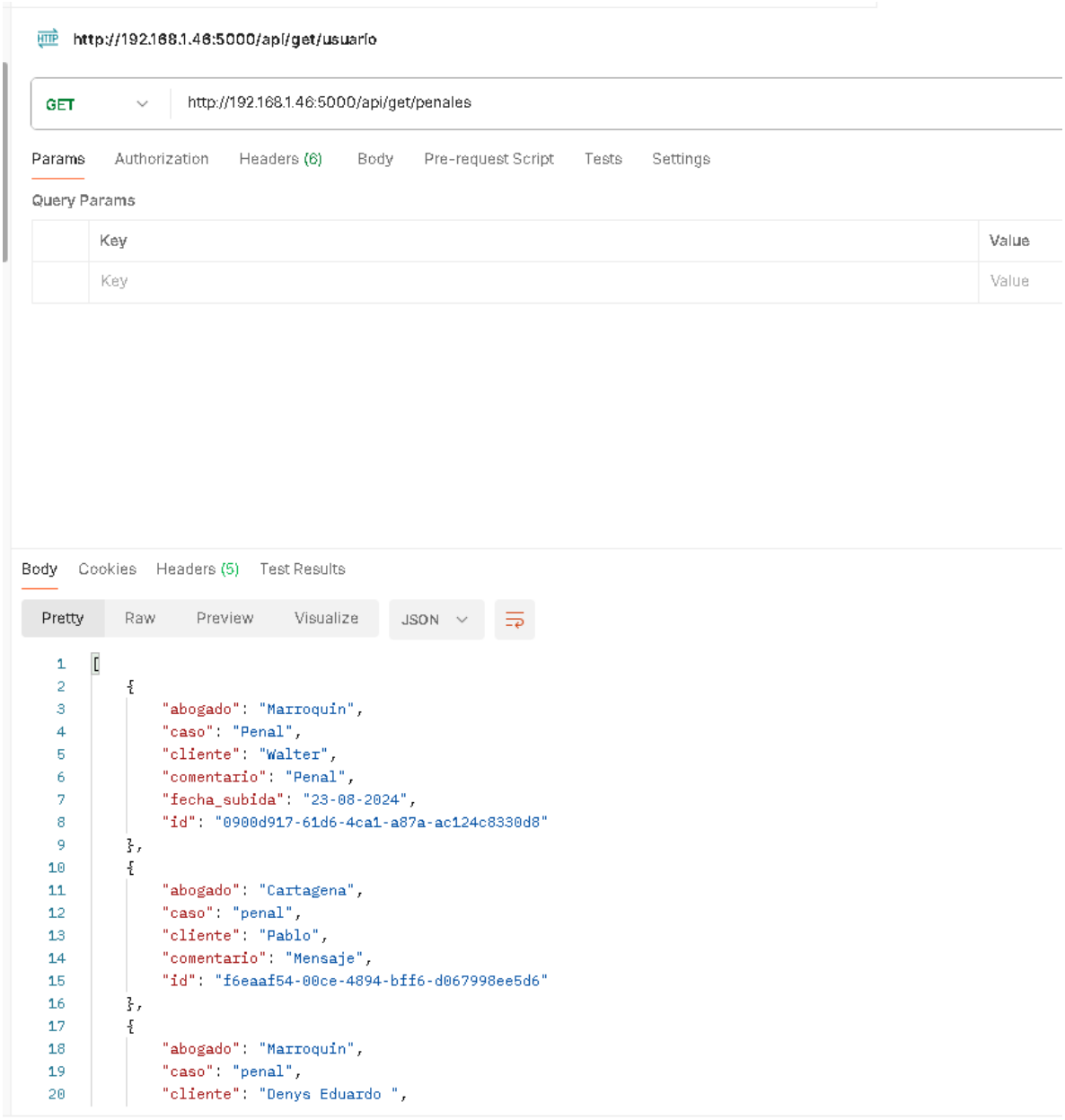
- **URL:** /api/get/documentos
- **Método:** GET
- **Descripción:** Retorna una lista de todos los documentos almacenados.



- **Respuesta:** Un JSON con los datos de todos los documentos.

9. Obtener Casos Penales

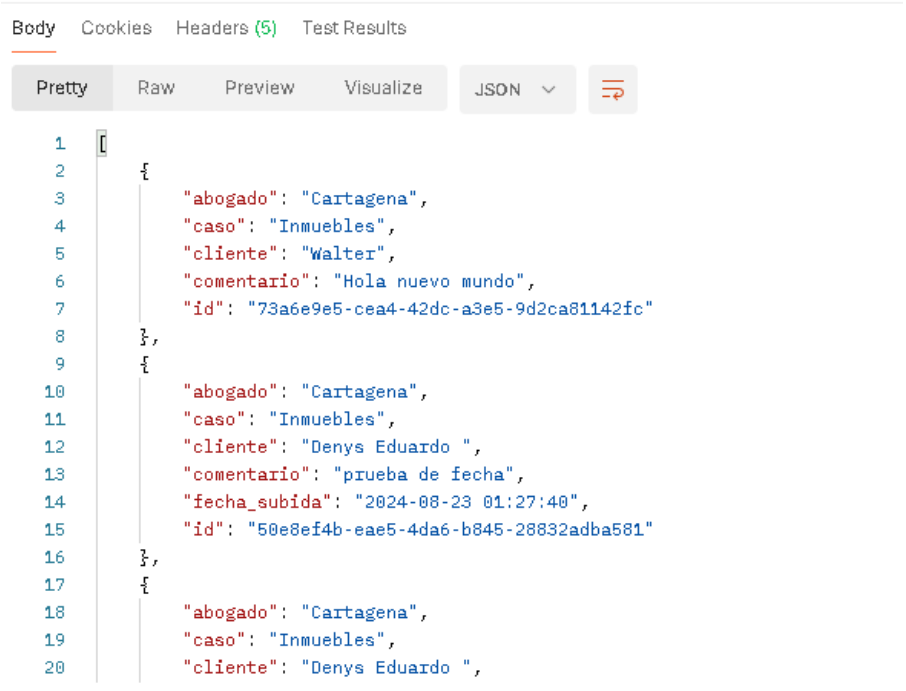
- **URL:** /api/get/penales
- **Método:** GET
- **Descripción:** Retorna una lista de casos penales.



- **Respuesta:** Un JSON con los datos de los casos penales.

10. Obtener Casos de Inmuebles

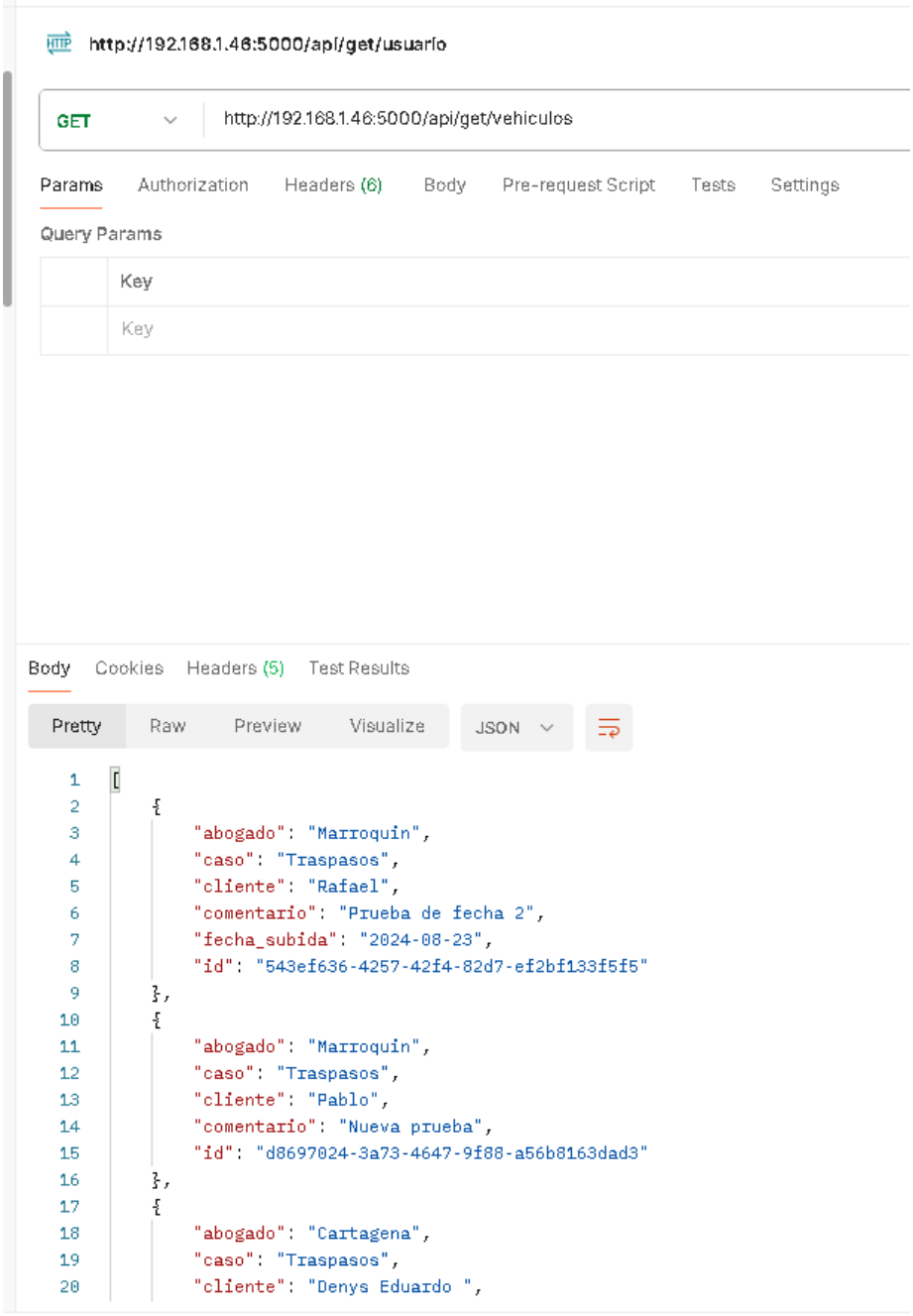
- **URL:** /api/get/inmueble
- **Método:** GET
- **Descripción:** Retorna una lista de casos de inmuebles.



- **Respuesta:** Un JSON con los datos de los casos de inmuebles.

11. Obtener Casos de Vehículos

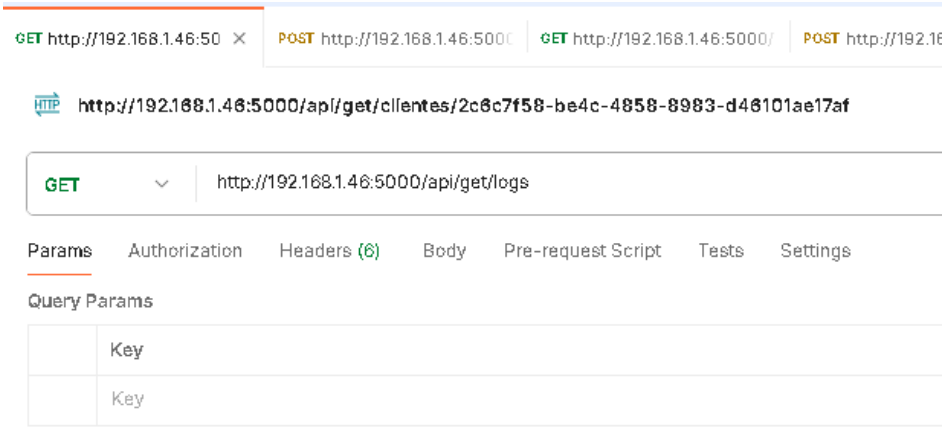
- **URL:** /api/get/vehiculos
- **Método:** GET
- **Descripción:** Retorna una lista de casos de vehículos asociados a un usuario específico.

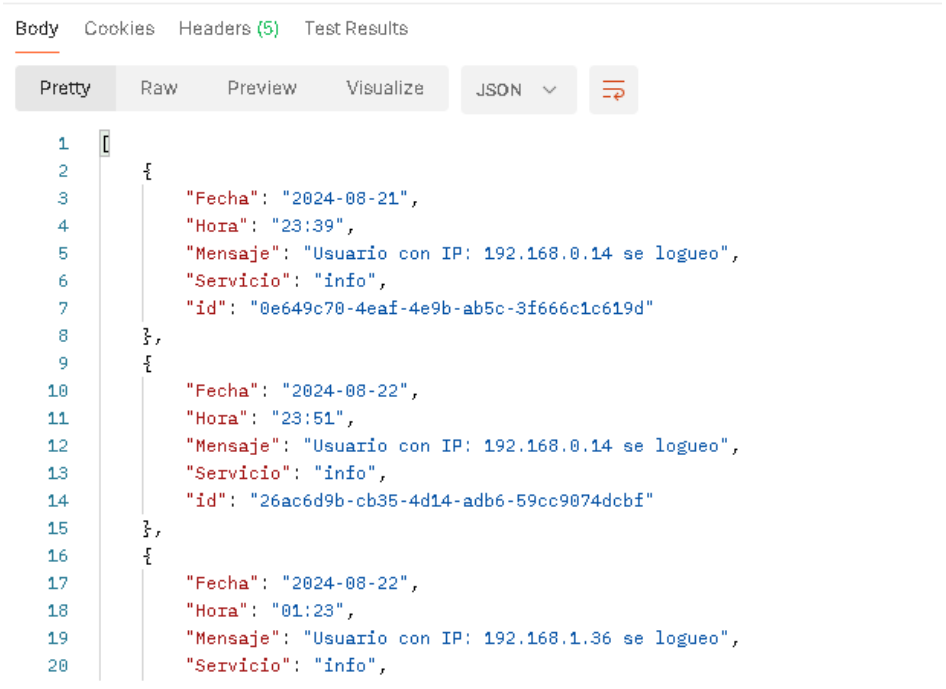


- **Respuesta:** Un JSON con los datos de los casos de vehículos.

12. Obtener Logs

- **URL:** /api/get/logs
- **Método:** GET
- **Descripción:** Retorna una lista de todos los logs almacenados en la base de datos.

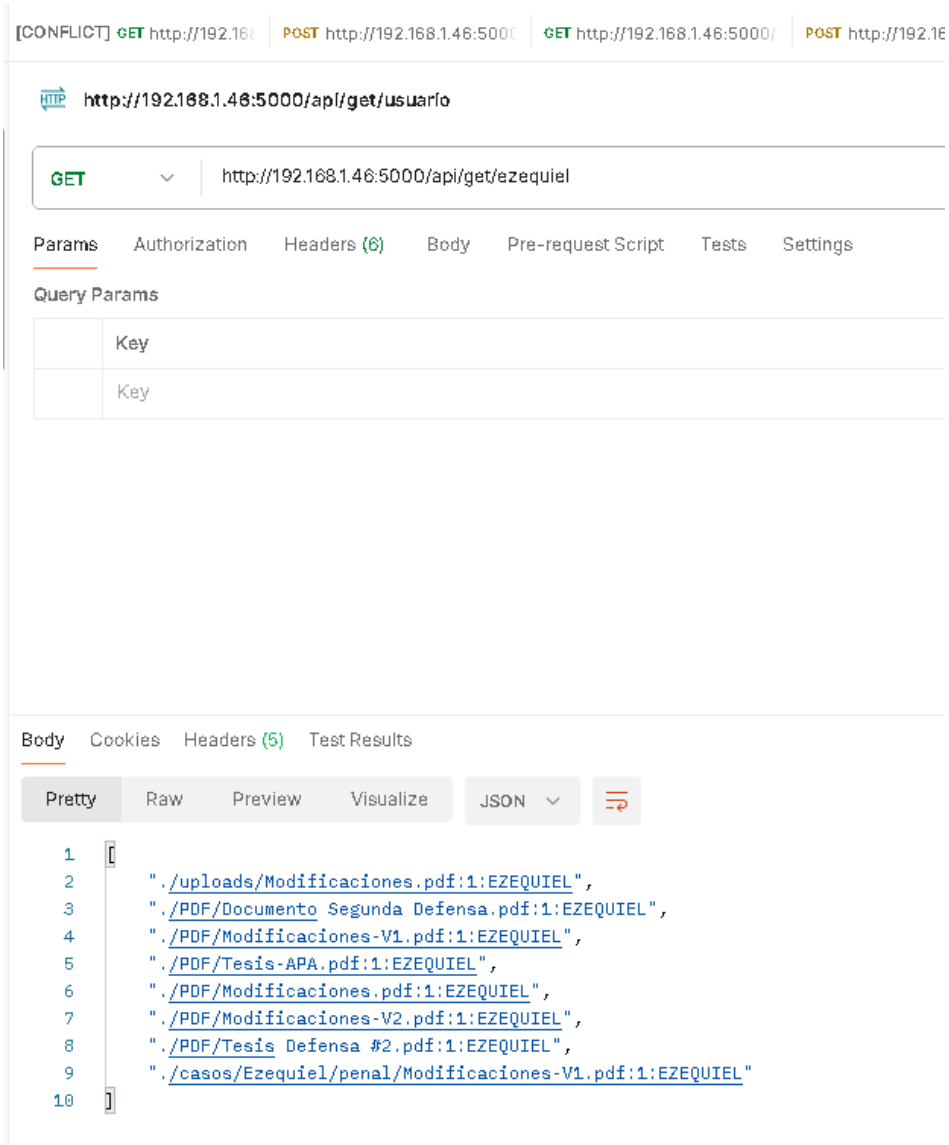




- **Respuesta:** Un JSON con los datos de todos los logs.

13. Realizar busqueda

- **URL:** /api/get/<texto>
- **Método:** GET
- **Descripción:** Retorna una lista de todas las rutas donde se encontró el texto.



Ejecución de la API

Para ejecutar la API, asegurar que todos los paquetes necesarios estén instalados y que el archivo api.py esté configurado correctamente. Luego, se debe iniciar el servidor utilizando el siguiente comando:

flask --app api.py run --host=0.0.0.0

Notas Finales

- La API utiliza RethinkDB para la gestión de la base de datos.
- Los logs de todas las operaciones realizadas se almacenan en la base de datos en la colección logs.
- Asegurar que las rutas de los directorios para los casos legales estén configuradas correctamente y que los permisos de escritura sean los adecuados para evitar errores al crear directorios.