Project Id: SCR845NGFF

Blog Application

## Project Overview:

This project is a RESTful Blog Application API built using Django and Django REST Framework (DRF). The API allows users to register, log in, create, read, update, and delete blog posts and comments. It utilizes JWT (JSON Web Tokens) for authentication, ensuring that only authenticated users can perform certain operations.

**Key Features:**

* User registration and authentication
* CRUD operations for blog posts
* CRUD operations for comments on blog posts
* JWT-based authentication for secure API access
* Pagination and search functionality for blogs

Tech Stack:

Programming Language : Python  
Framework : Django   
Database : MySQL  
Authentication : JWT

## Database Design:

## ****Database Schema****

### ****Users:****

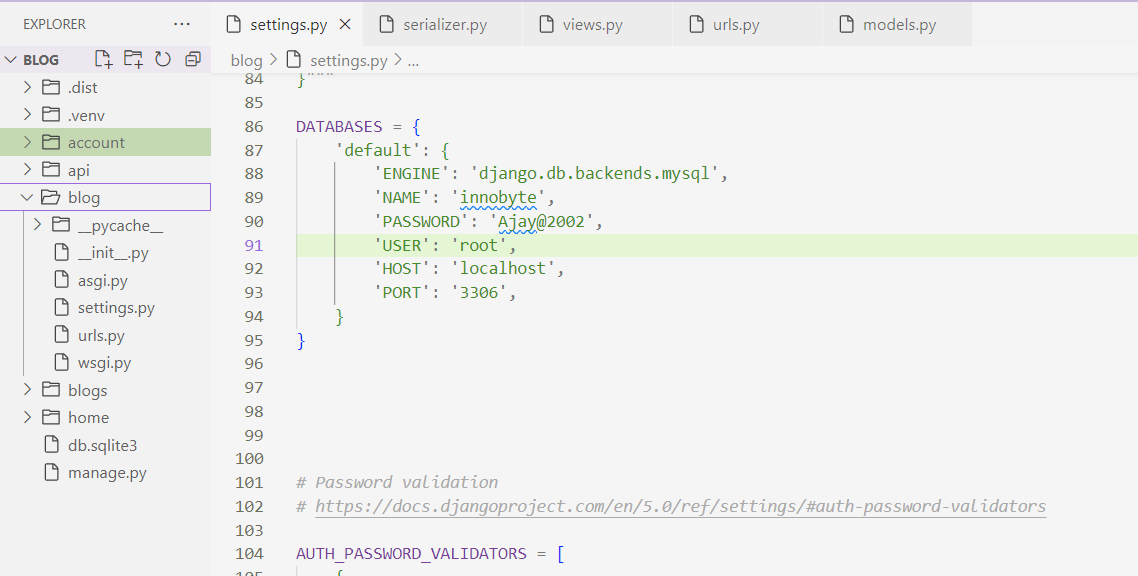
* id: - Primary key.
* username: - Unique username.
* password: - Hashed password.
* email: - User email.

### ****posts****

* + id: - Primary key.
  + title: - Blog post title.
  + content: - Blog post content.
  + author\_id: - Foreign key referencing users.
  + created\_at: - Timestamp of creation.
  + updated\_at: - Timestamp of last update.

### ****comments****

* + id: - Primary key.
  + post\_id: - Foreign key referencing posts.
  + content: - Comment content.
  + author\_id: - Foreign key referencing users.
  + created\_at: - Timestamp of creation.

Database Setup:  
  


Project Structure:

BlogApplication/

├── api /

│ ├── \_init\_.py

│ ├── urls.py

├── account/

│ ├── \_init\_.py

│ ├── admin.py

│ ├── apps.py

│ ├── migrations/

│ ├── serializers.py

│ ├── tests.py

│ ├── views.py

│ ├── urls.py

├── home/

│ ├── \_init\_.py

│ ├── admin.py

│ ├── apps.py

│ ├── migrations/

│ ├── serializers.py

│ ├── tests.py

│ ├── views.py

│ ├── urls.py

├── Blog/

│ ├── \_init\_.py

│ ├── asgi.py

│ ├── settings.py

│ ├── urls.py

│ ├── wsgi.py

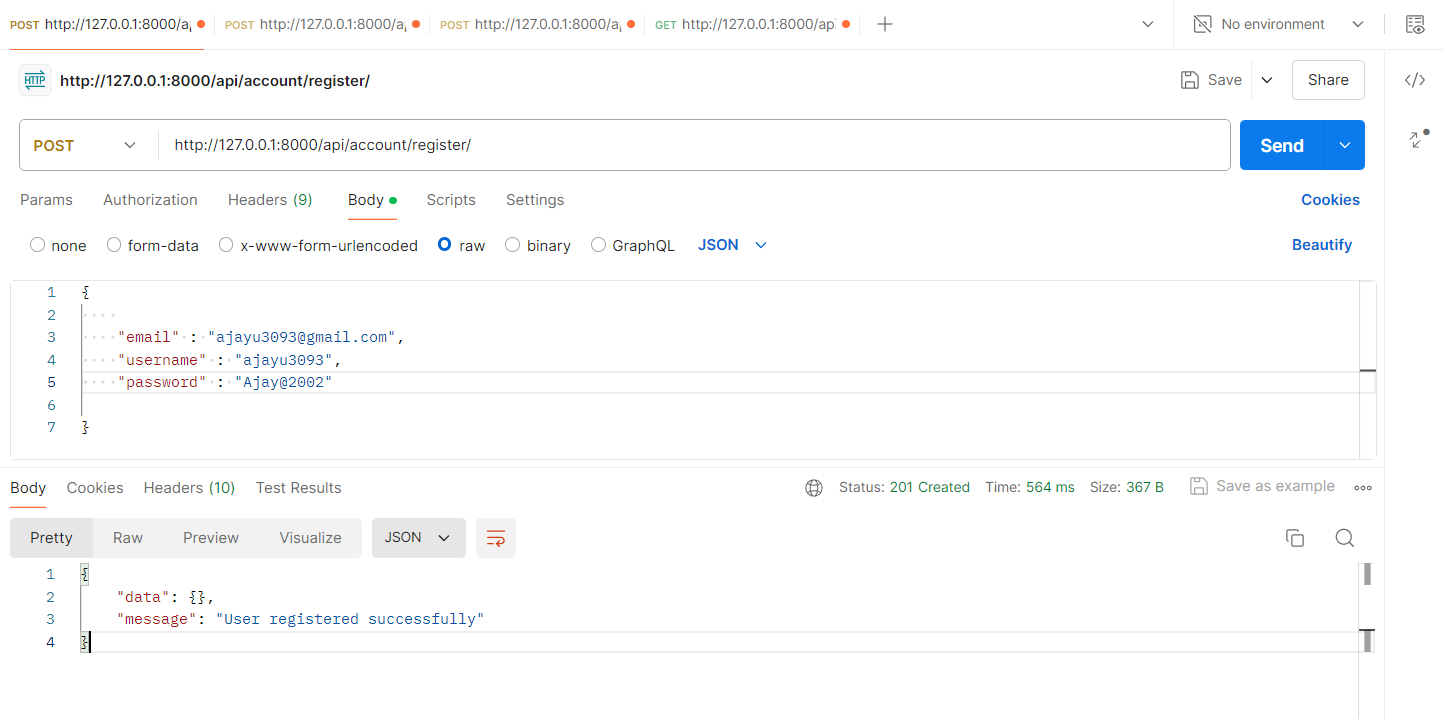
├── manage.py

## 

## ****API Endpoints** (**Testing Using Postman)

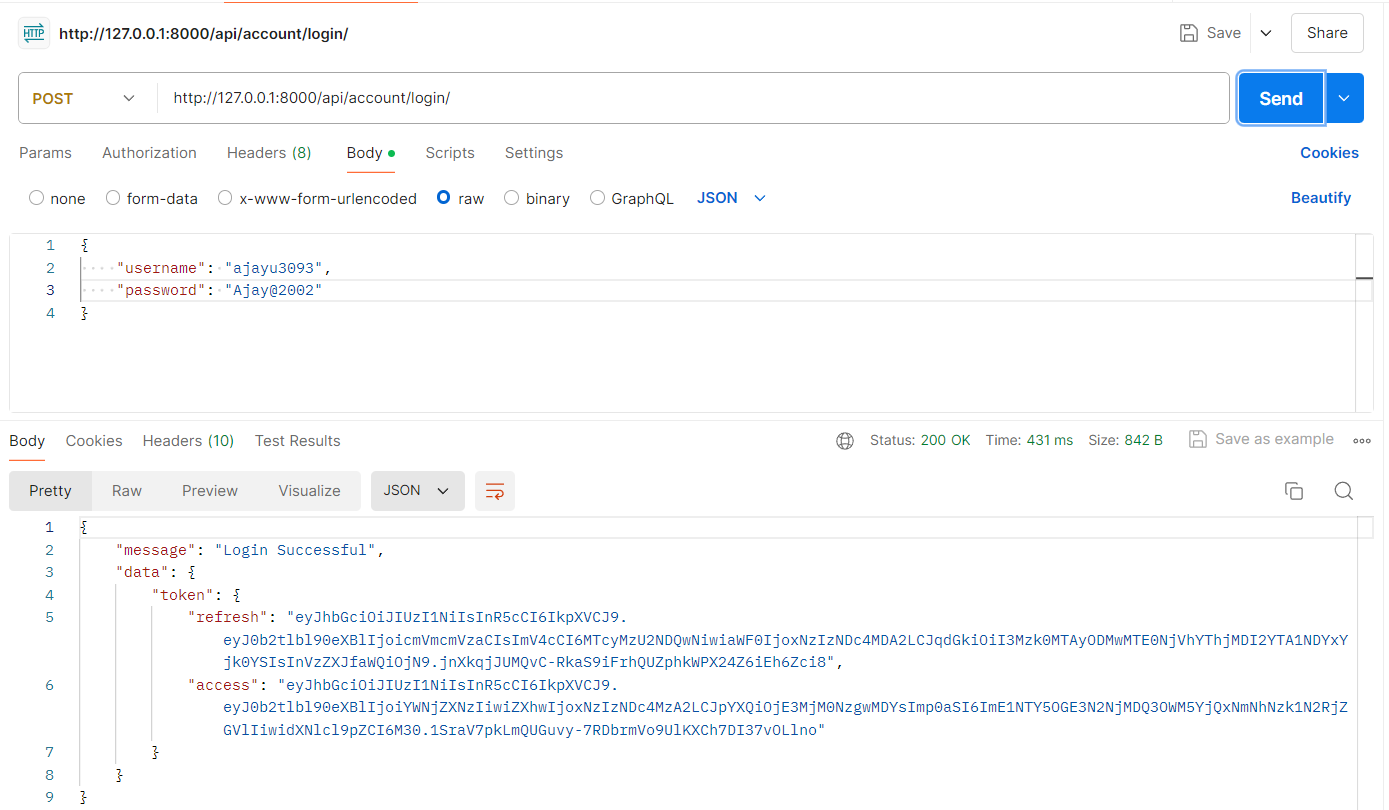
User Authentication:

**POST** /register

* Registers a new user.

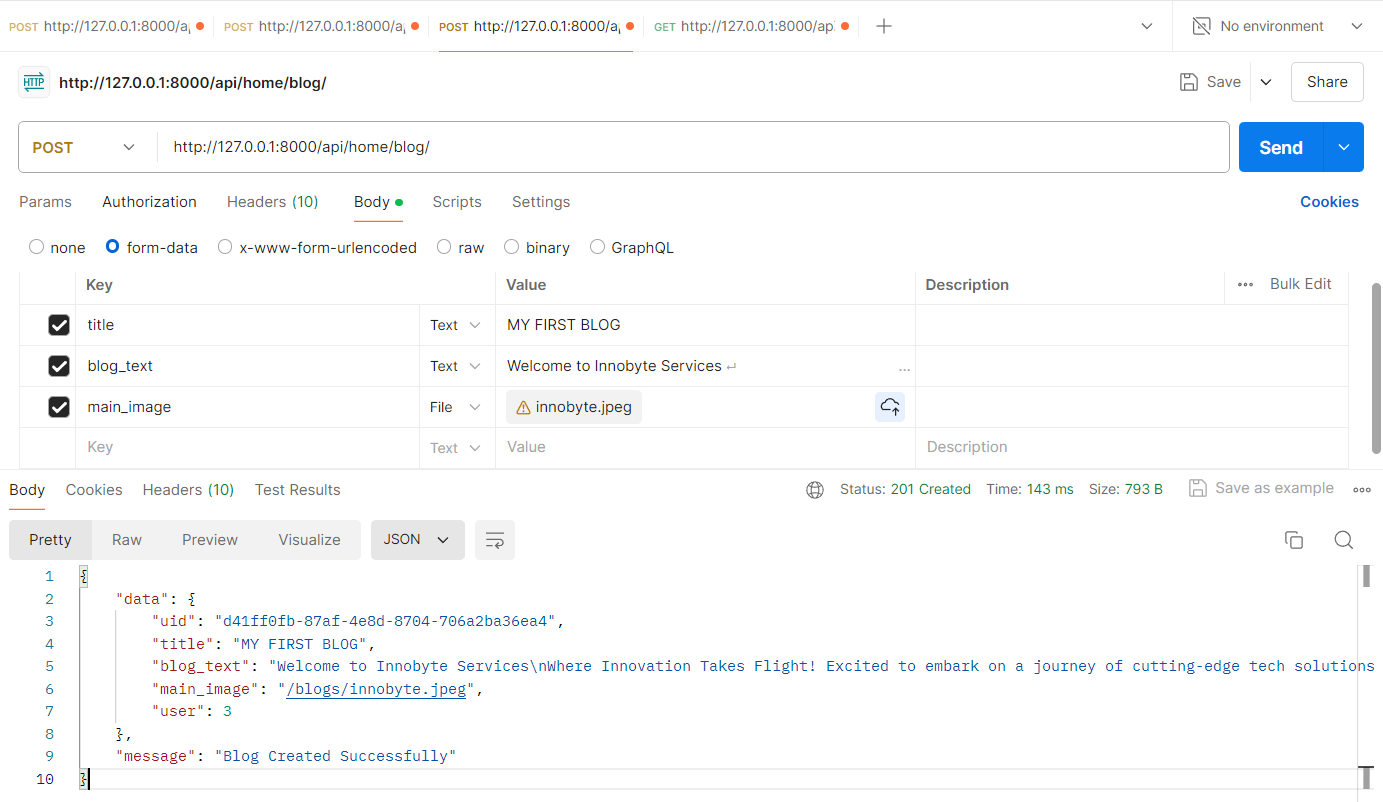
POST /login

* Authenticates a user and returns a JWT token.



Blog Posts

* POST /posts
  + Creates a new blog post.
  + Request Header:
    - Authorization: Bearer <JWT\_TOKEN>



 GET /posts

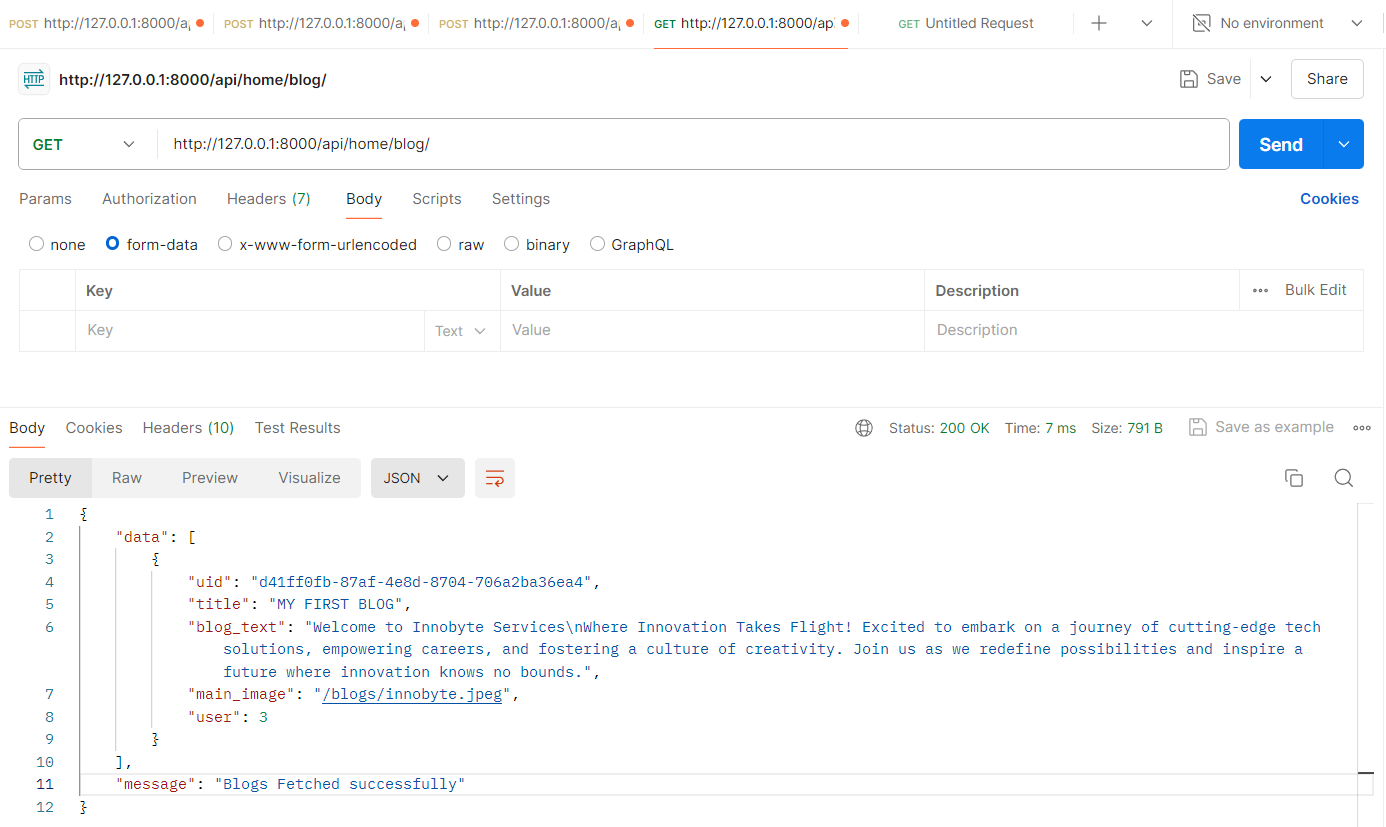
* Retrieves all blog posts.
* Response:
  + Status: 200 OK

 GET /posts/{id}

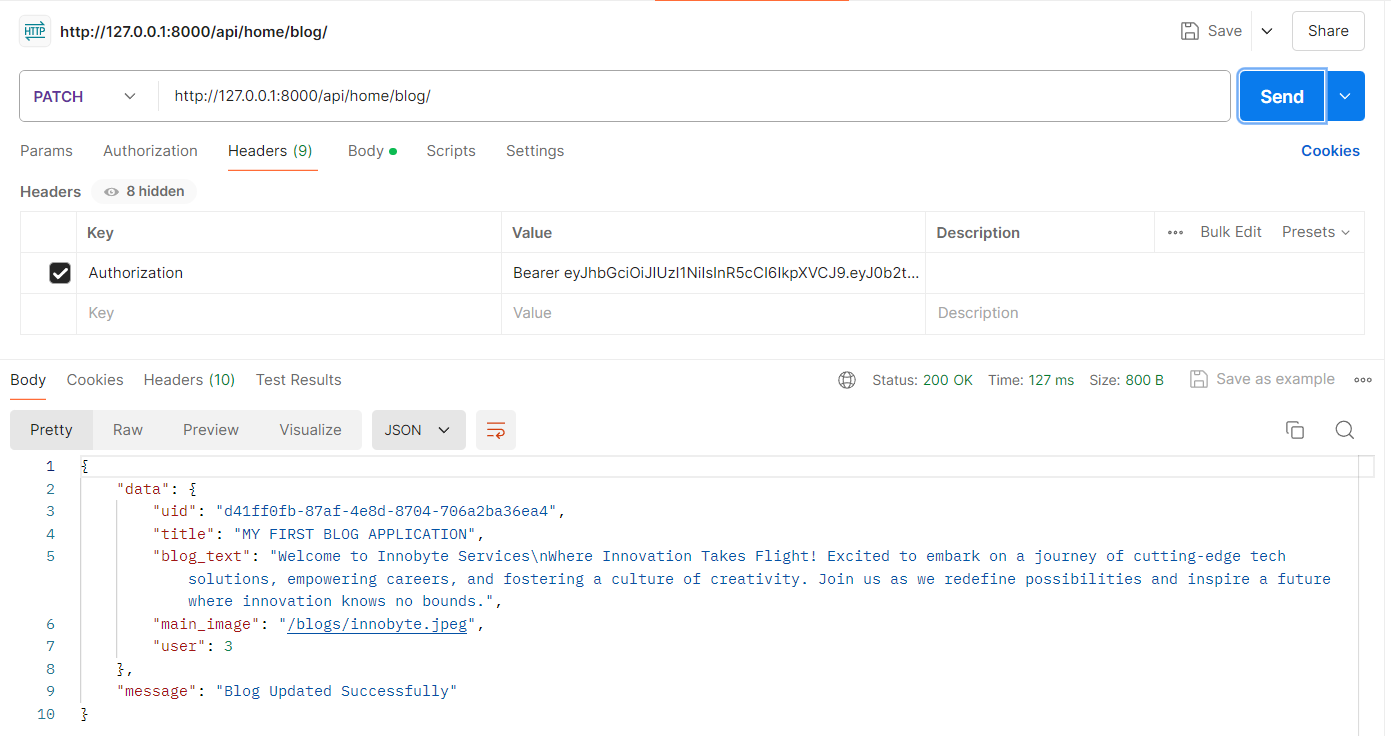
* Retrieves a single blog post by ID.
* Response:
  + Status: 200 OK

 PUT /posts/{id}

* Updates an existing blog post.
* Request Header:
  + Authorization: Bearer <JWT\_TOKEN>

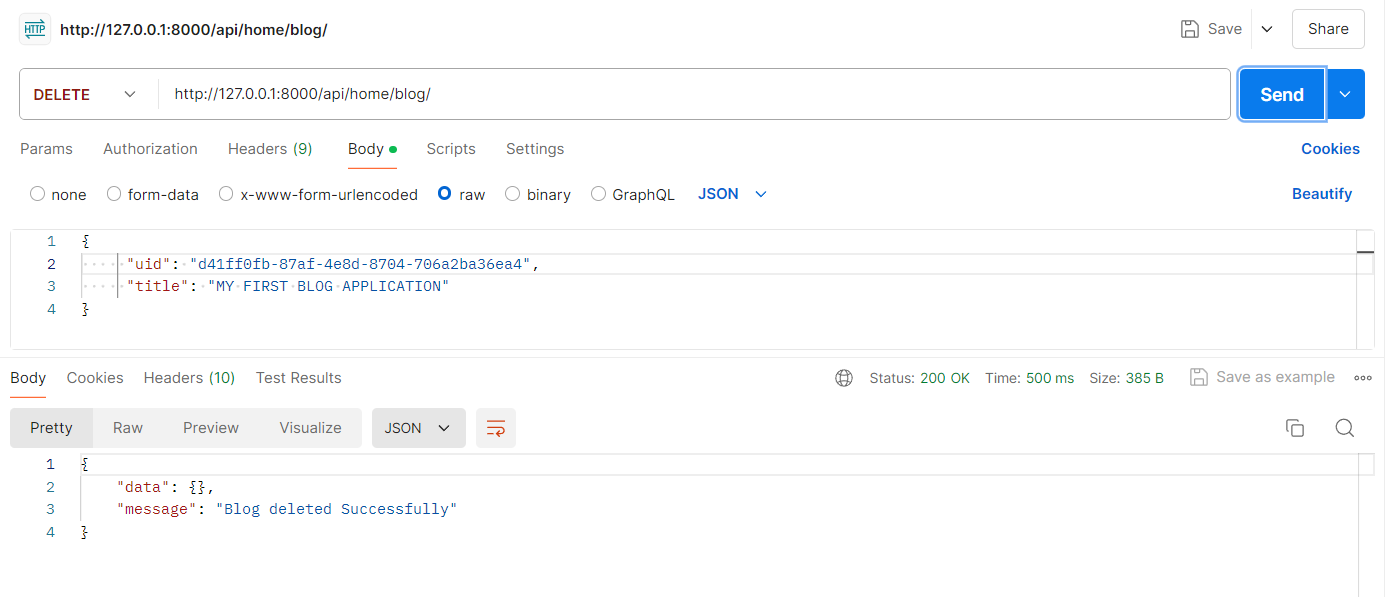


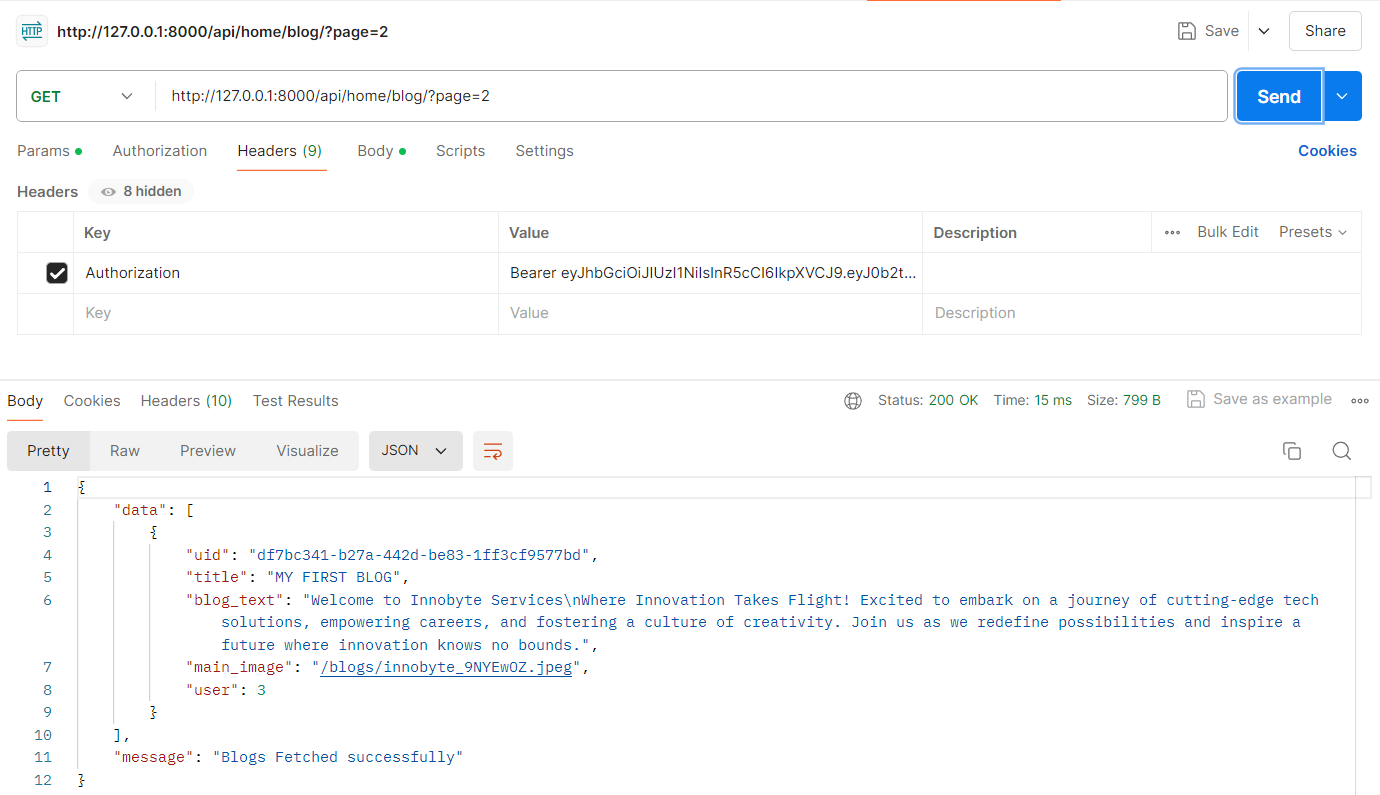
PATCH – Update Blog:



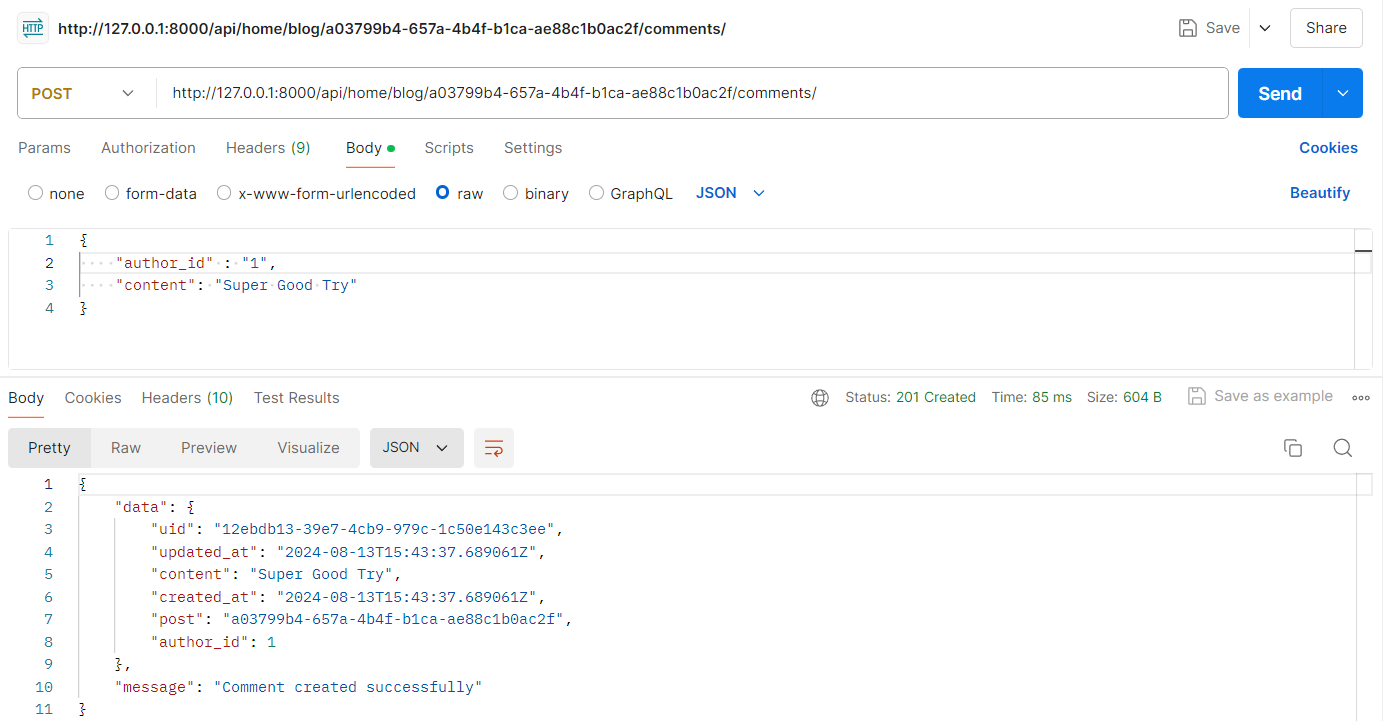
DELETE /posts/{id}

* Deletes a blog post.
* Request Header:
  + Authorization: Bearer <JWT\_TOKEN>
* Response:
  + Status: 204 No Content



Public Blog Views:

Comments/post{id}

* + id: Primary key.
  + post\_id: Foreign key referencing posts.
  + content: Comment content.
  + author\_id: Foreign key referencing users.
  + created\_at: Timestamp of creation.