Project Overview

This project implements **role-based authentication** with `JWT` in **Spring Boot**.
It supports **Admin** and **User** roles with **CRUD operations** and **profile picture upload**.

🛠 Tech Stack

- **Backend:** Spring Boot (Spring Security, JWT, Lombok, JPA, MySQL)
- **Database:** MySQL
- **Security:** JWT Authentication
- **API Testing:** Postman

🚀 API Endpoints with Postman Collection Structure

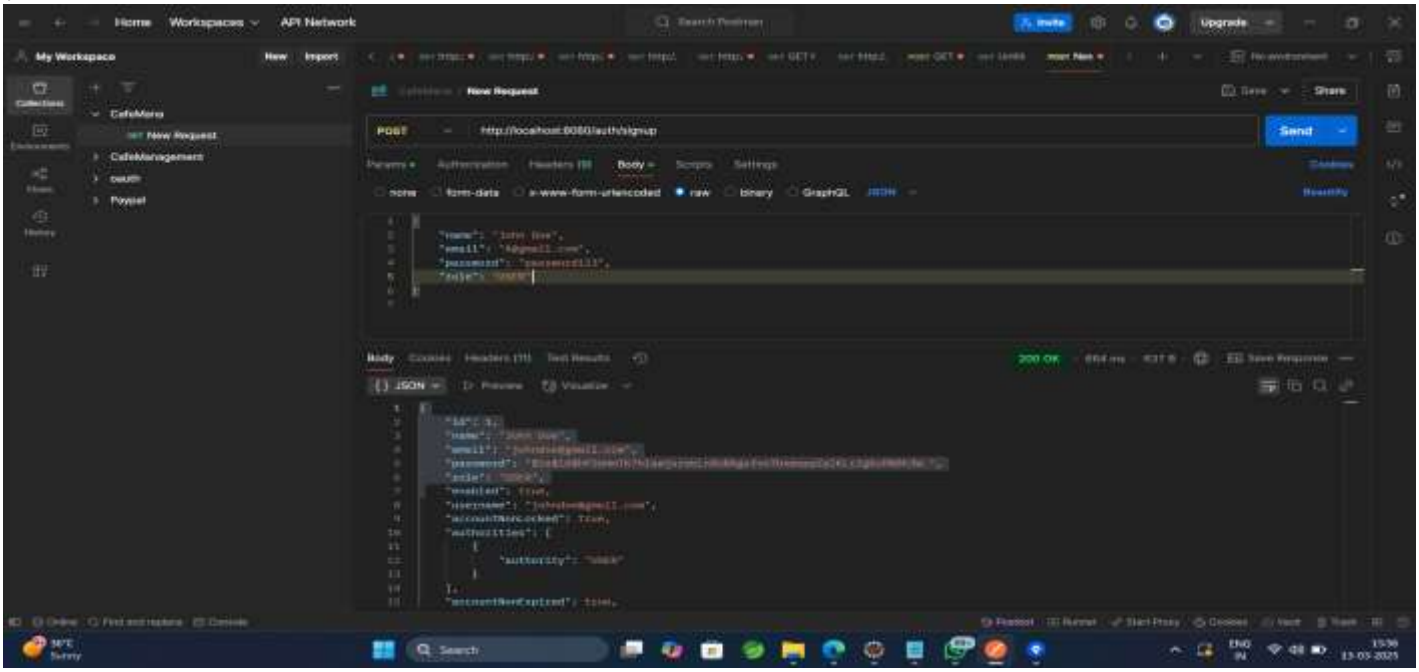
You can import this **Postman collection structure** and run the APIs in order.

1️⃣ Authentication APIs (`/auth`)

🔹 User Signup

Endpoint: `POST /auth/signup`
Request Body:

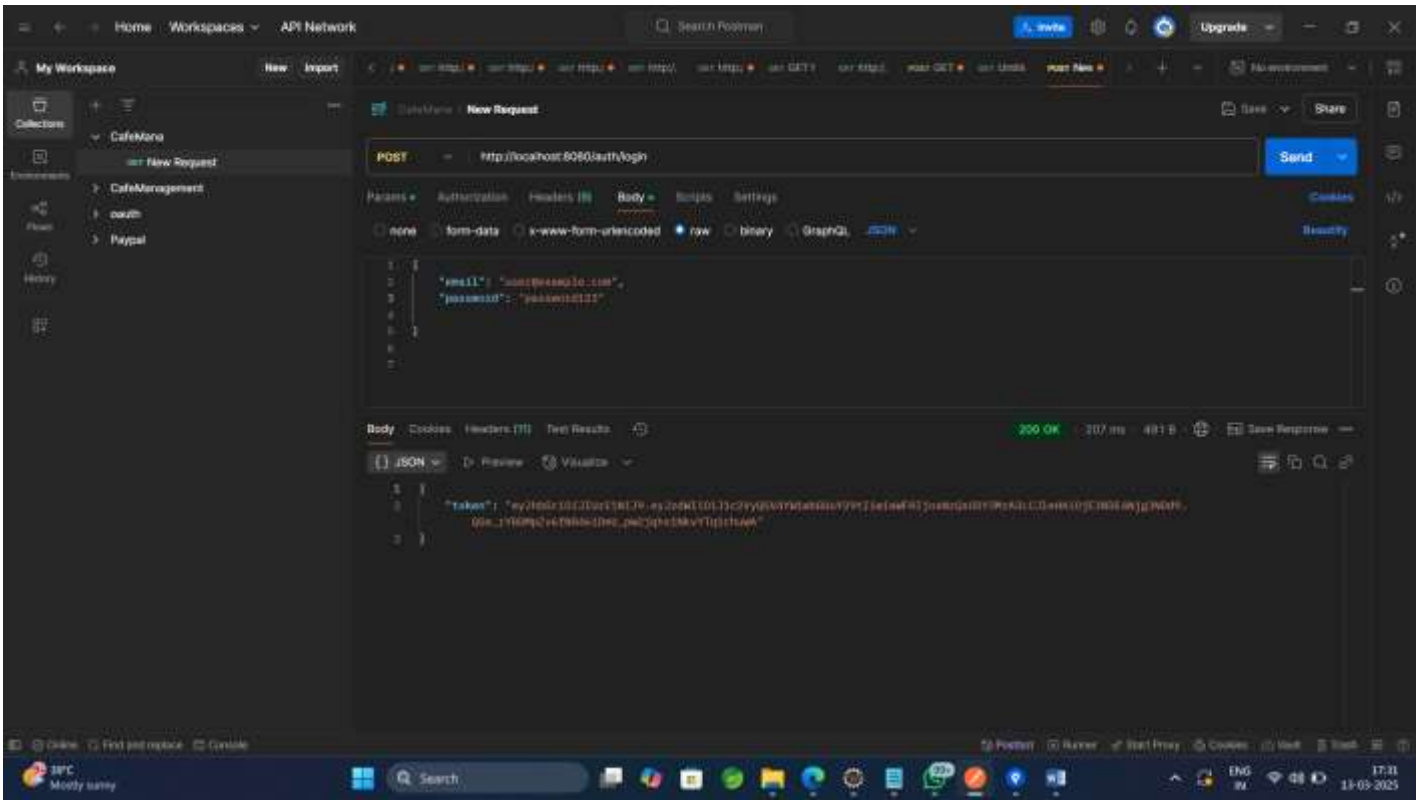
```
{
    "email": "johndoe@gmail.com",
    "password": "password123",
    "role": "USER"
}
```



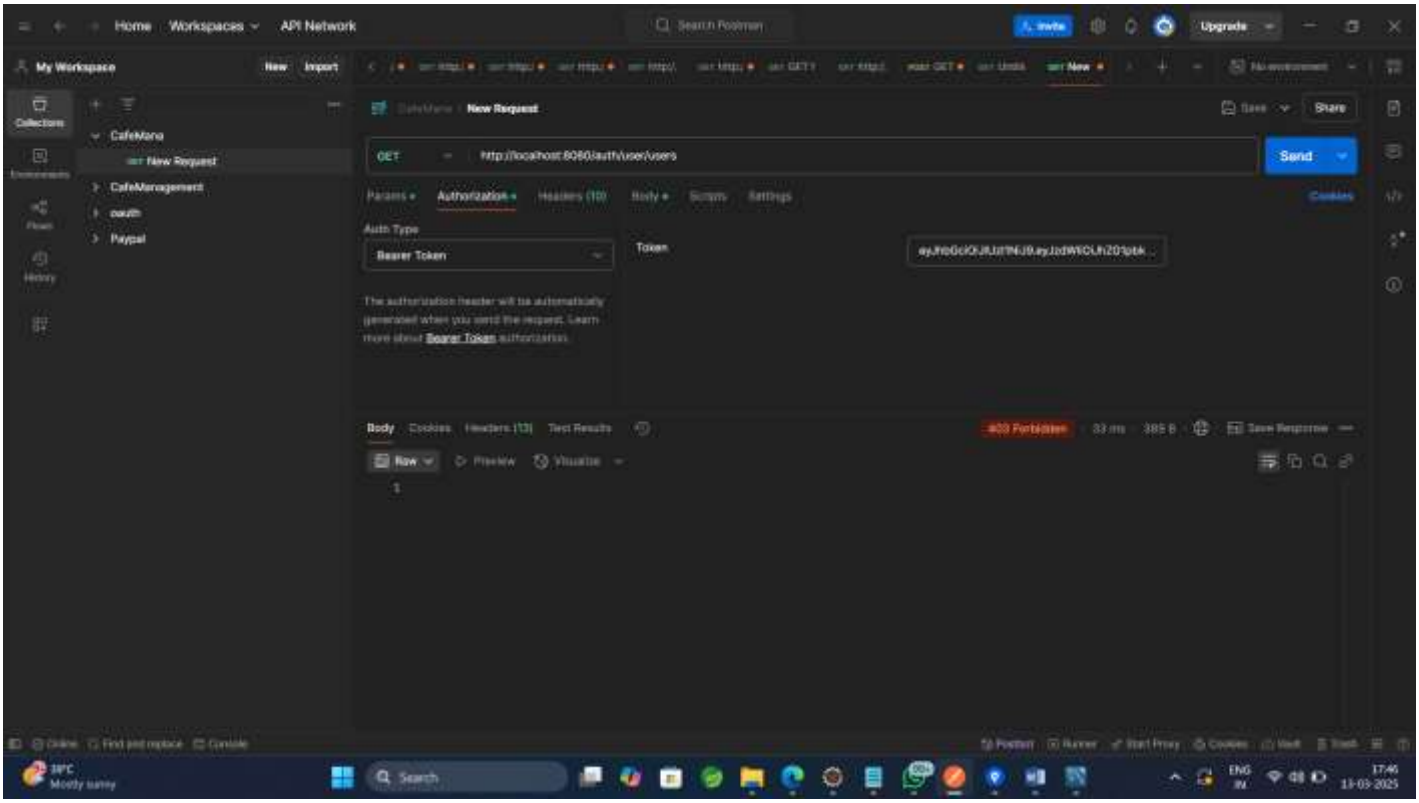
User Login

Endpoint: `POST /auth/login`
Request Body:

```
{
    "email": "user@example.com",
    "password": "password123"
}
```



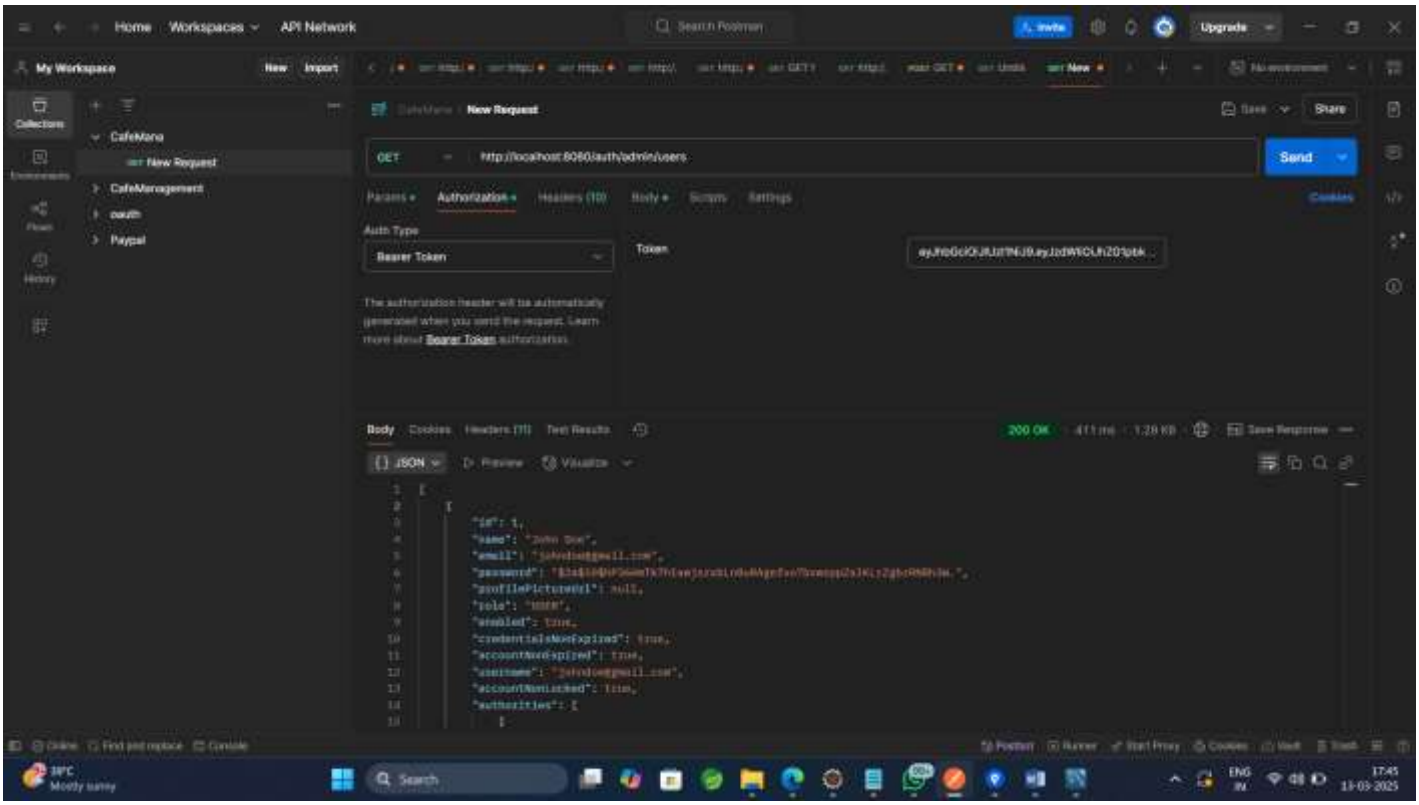

```
[
  {
    "id": 1,
    "email": "user@example.com",
    "role": "USER"
  },
```



{ "id": 2, "email": "admin@gmail.com",

"role": "ADMIN"

}



Here is a **tabular representation** of all the APIs in your **Role-Based User Authentication System**:

📌 API Endpoints Table

#	Method	Endpoint	Description	Role	Authorization Required?
◆ Authentication APIs (`/auth`)					
1	`POST`	`/auth/signup`	Register a new user	Public	❌ No
2	`POST`	`/auth/login`	Authenticate user and get JWT token	Public	❌ No
◆ Admin APIs (`/auth/admin`)					
3	`GET`	`/auth/admin/users`	Get all users	Admin	✅ Yes (Admin)
4	`DELETE`	`/auth/admin/users/{id}`	Delete a user by ID	Admin	✅ Yes (Admin)
◆ User APIs (`/auth/user`)					
5	`GET`	`/auth/user/{id}`	Get user details by ID	User (Self)	✅ Yes (User)
6	`PUT`	`/auth/user/{id}`	Update user details (self)	User (Self)	✅ Yes (User)
◆ Profile Picture APIs (`/auth/user/{id}/upload`)					
7	`POST`	`/auth/user/{id}/upload`	Upload profile picture	User (Self)	✅ Yes (User)