**Project Setup and Documentation**

**Overview:**

This project is a backend application built using Node.js, Express, and MongoDB (via Mongoose). It handles user and admin authentication, assignment management, and task status updates.

**System Requirements:**

Node.js(v16.x or higher)

MongoDB(either locally or a cloud MongoDB service like MongoDB Atlas)

Postman or any similar API testing tool for making API requests

**1. Clone the Repository**

First, clone the repository to your local machine.

bash

git clone <repository\_url>

cd <project\_directory>

**2. Install Dependencies**

Ensure that you have Node.js and npm (Node Package Manager) installed.

To install the project dependencies, run:

bash

npm install

This command will install all required packages listed in `package.json`.

**3. Set Up Environment Variables**

Create a `.env` file in the root directory of the project to store environment variables. This file will include sensitive information such as your MongoDB URI and JWT secret. Here's an example `.env` file:

bash

MONGO\_URI="your\_mongodb\_connection\_string"

JWT\_SECRET="supersecurejwtsecret"

JWT\_EXPIRY="1d"

PORT=5000

- MONGO\_URI: The connection string for your MongoDB database (e.g., from MongoDB Atlas).

- JWT\_SECRET: A secret key used for signing and verifying JWT tokens.

- JWT\_EXPIRY: The expiration time of the JWT token (e.g., "1d" for 1 day).

- PORT: The port on which the server will run (default is 5000).

> Note: Make sure to replace `your\_mongodb\_connection\_string` with your actual MongoDB connection URL. For example, if you are using MongoDB Atlas, you can find this connection string in your MongoDB Atlas dashboard.

**4. Database Setup**

Ensure that you have a MongoDB instance running (locally or via MongoDB Atlas).

1**. For local MongoDB:**

- Install MongoDB locally on your machine if it's not installed.

- Start the MongoDB server using the following command:

bash

mongod

**2. For MongoDB Atlas:**

- Go to [MongoDB Atlas](https://www.mongodb.com/cloud/atlas) and create a new project.

- Set up a new database cluster and get the connection string (you'll need this for the `MONGO\_URI` in your `.env` file).

**5. Run the Project**

To start the backend server, use the following command:

bash

npm start

This will start the server on the port specified in the `.env` file (default: 5000).

bash

Server running on port 5000

**6. Testing the API with Postman**

# API Endpoints

1. Create User:

- Endpoint: `POST /users`

- Request Body:

json

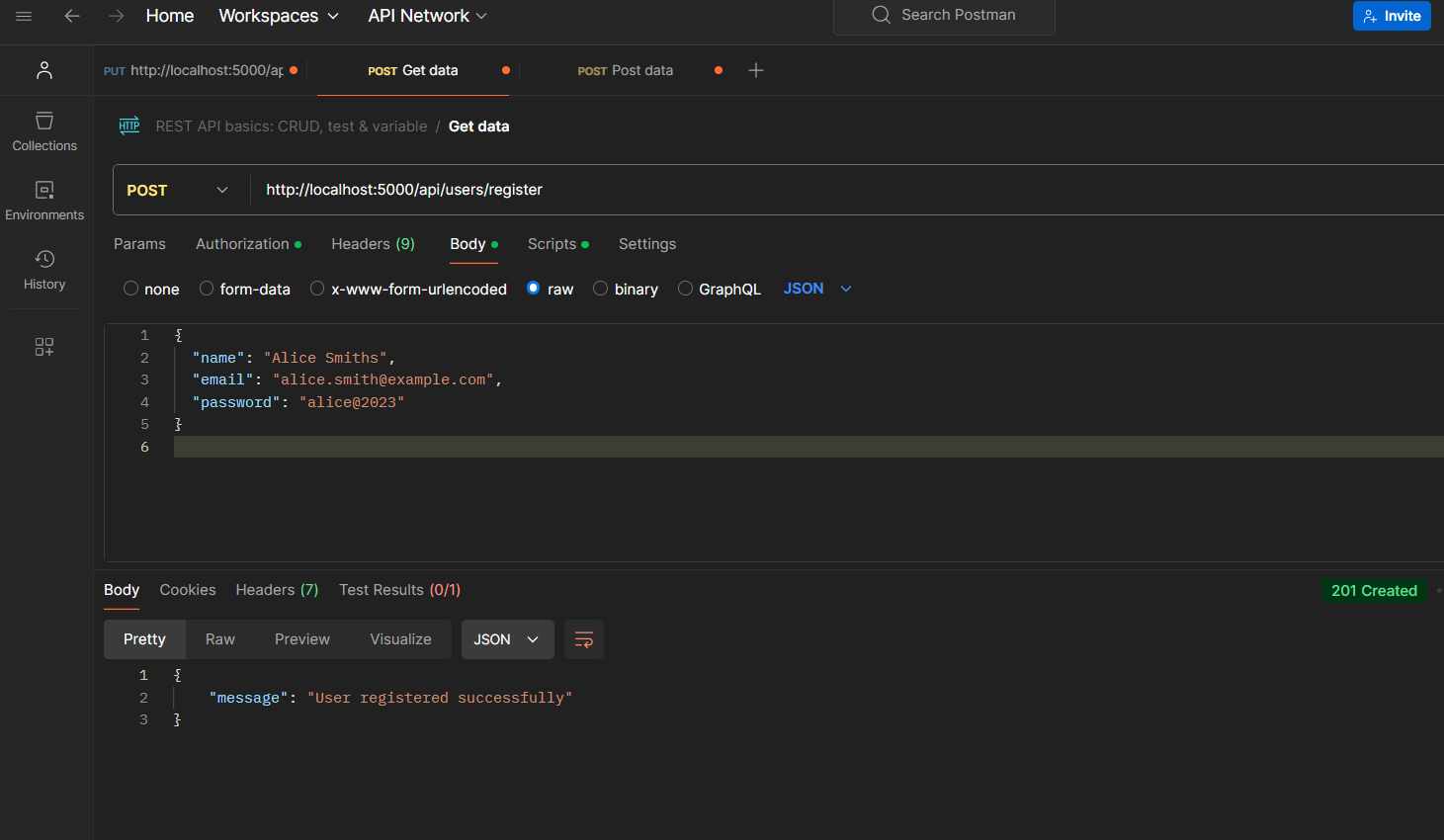
{

"name": "John Doe",

"email": "john@example.com",

"password": "password123"

}



2. Create Admin:

- Endpoint: `POST /admins`

- Request Body:

json

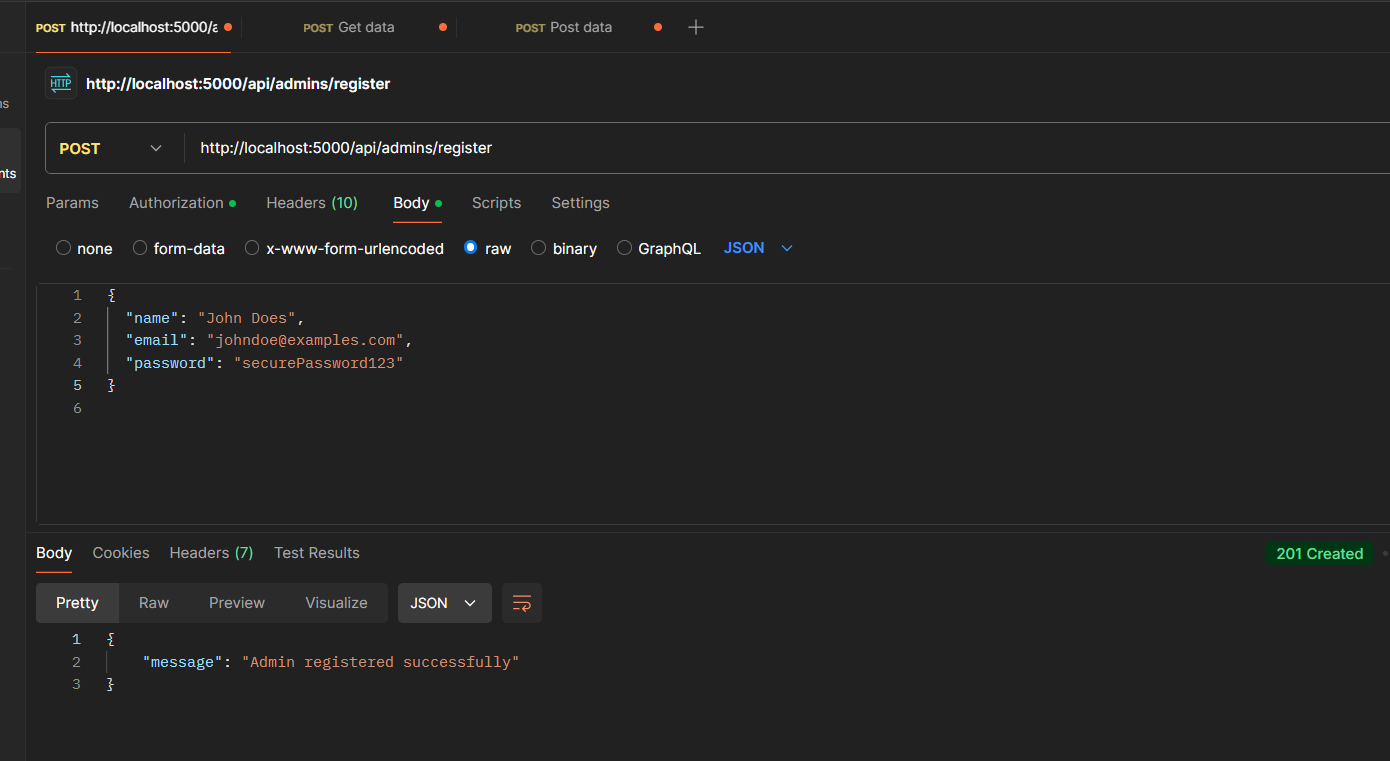
{

"name": "Admin User",

"email": "admin@example.com",

"password": "adminpassword"

}



3. Create Assignment:

- Endpoint: `POST /assignments`

- Request Body:

json

{

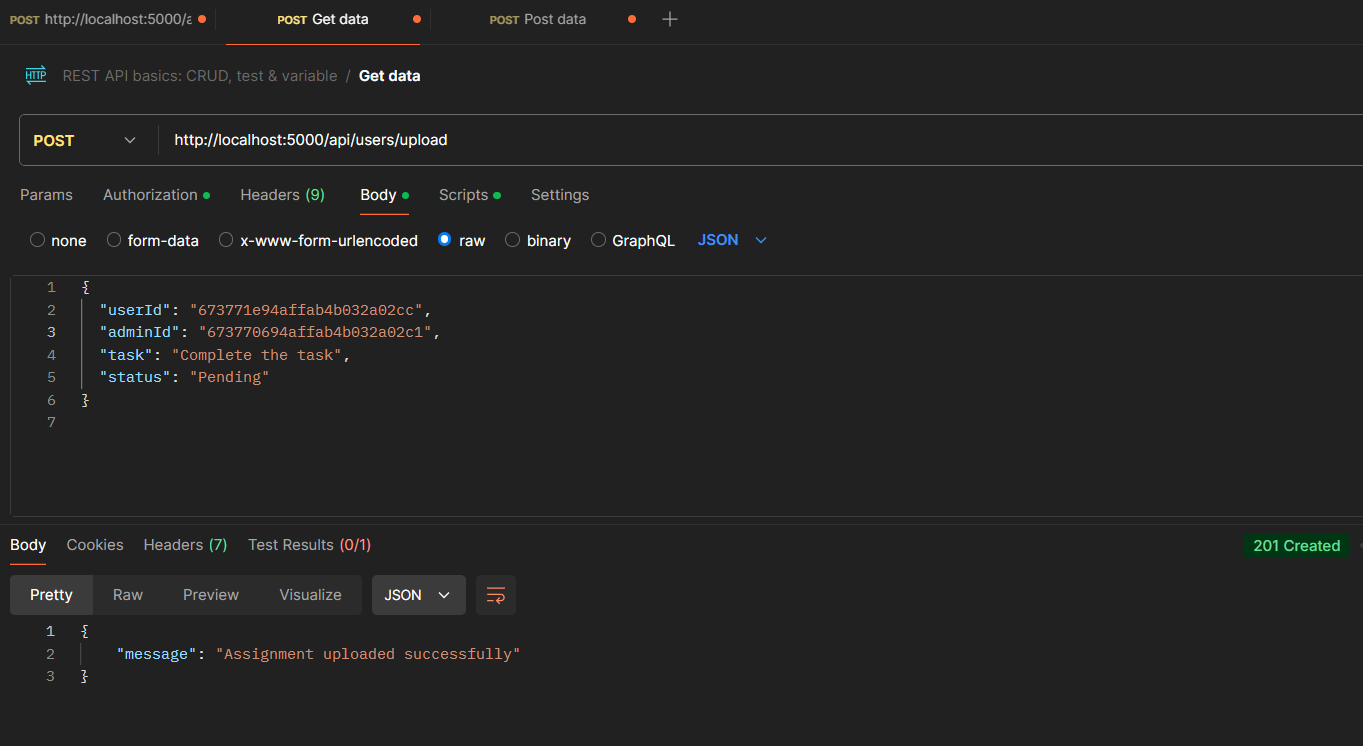
"userId": "user\_id\_from\_db",

"adminId": "admin\_id\_from\_db",

"task": "Complete the task",

"status": "Pending"

}



4. Update Assignment Status:

- Endpoint: `PUT /assignments/:id/status`

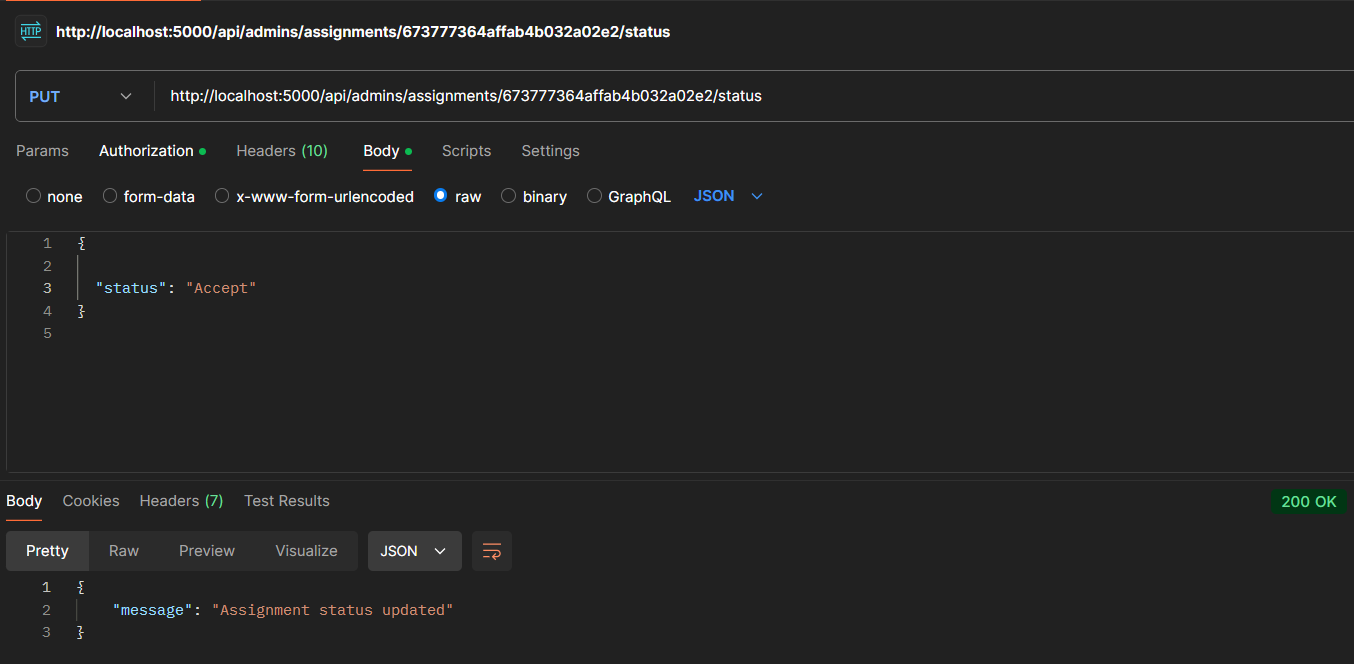
- Request Body:

json

{

"status": "Accepted"

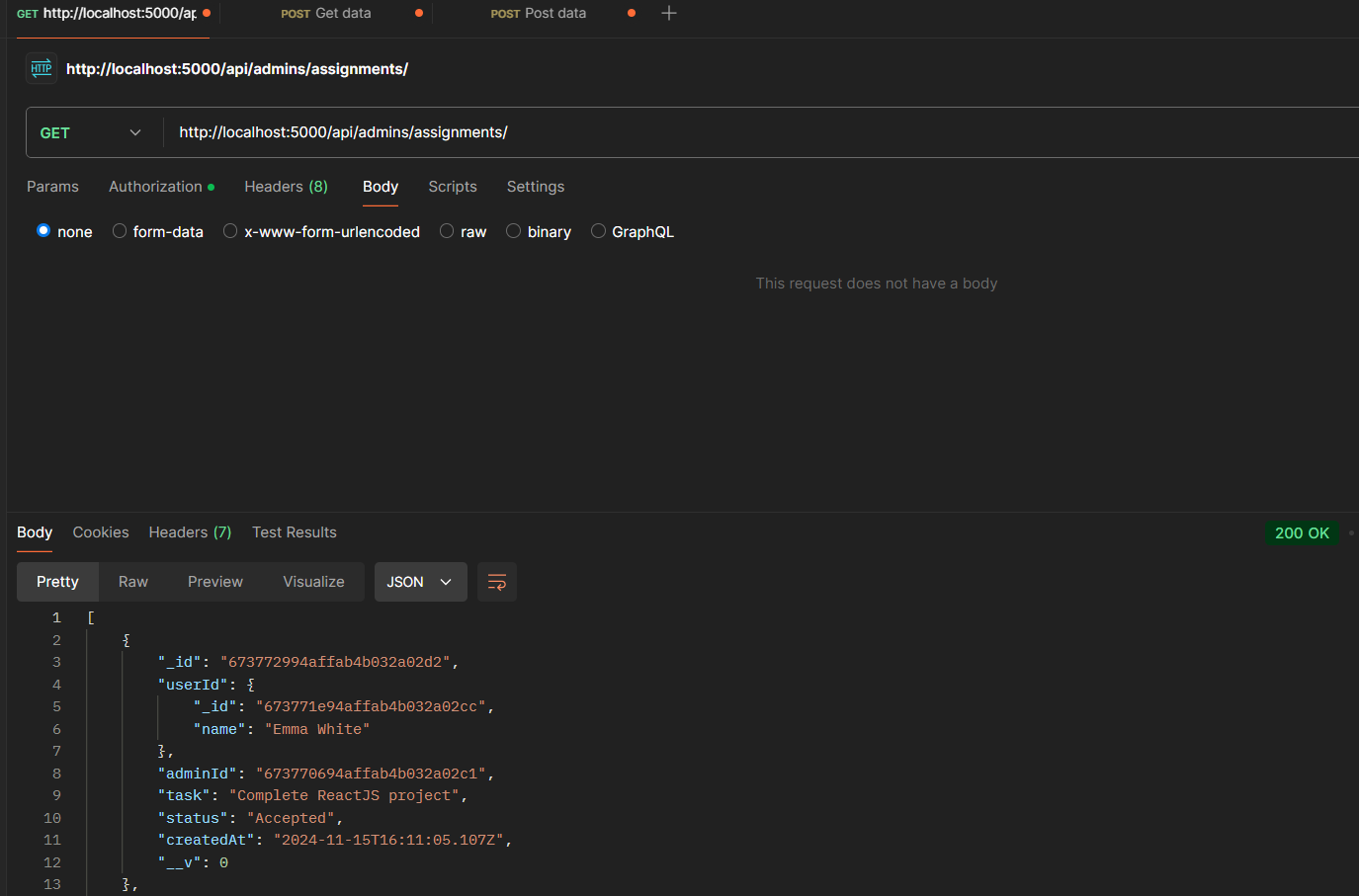
}



- Replace `:id` with the actual assignment ID from the database.

5. Get All Assignments:

- Endpoint: `GET /assignments`



**7. Troubleshooting**

- MongoDB Connection Issues:

- Ensure your MongoDB URI is correct in the `.env` file.

- If using MongoDB Atlas, ensure your cluster is online and your IP address is whitelisted.

- JWT Secret Issues:

- Ensure that the JWT\_SECRET value is consistent and not empty.

- Port Issues:

- If the application is not starting on the default port, check if another application is using the same port or change the port in the `.env` file.