**User Management API Setup and Deployment Documentation**

**Introduction**

The User Management API is a simple RESTful API built using Python and Flask framework. It provides endpoints for managing user data, including creating new users, retrieving user information, and listing all users. This document outlines the structure pattern chosen for storing user data and explains how the API was implemented.

**Structure Pattern**

User data is stored in a CSV file named **data.csv**. Each user is represented as an object with **id** and **name** fields. The CSV file is loaded into memory when the API starts up, and user data is manipulated in memory during API operations. This structure pattern was chosen for its simplicity and ease of implementation, suitable for a small-scale user management system.

**API Implementation**

**Dependencies:**

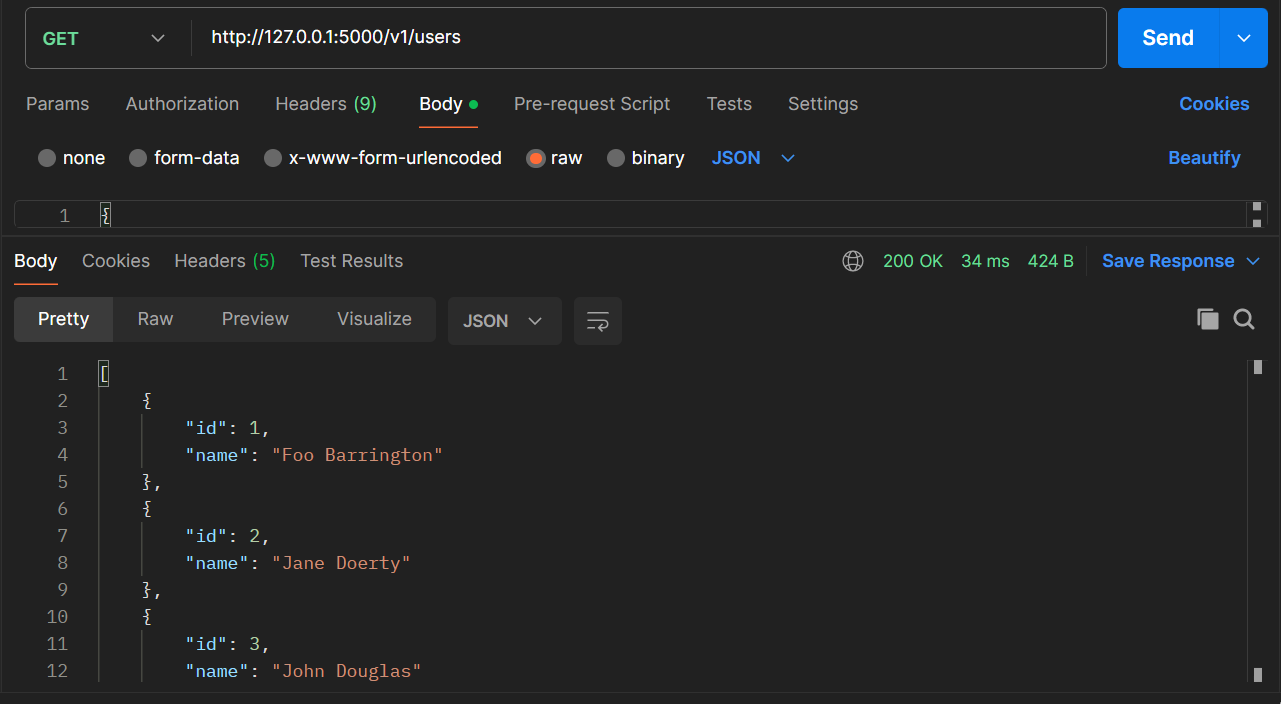
* Python 3
* Flask 2.0.2
* requests (for testing)

**Endpoints:**

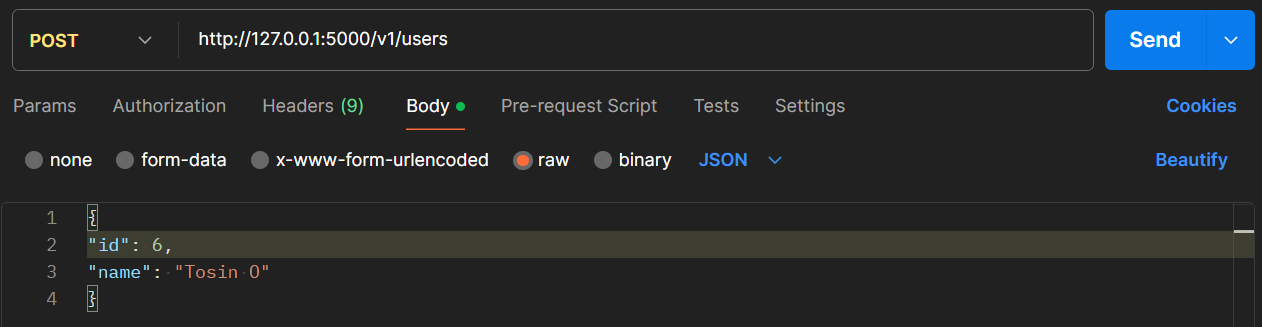
* **GET /v1/users**: Returns a list of all users.
* **POST /v1/users**: Creates a new user.
* **GET /v1/users/{userId}**: Returns user information by ID.

**/v1/users**

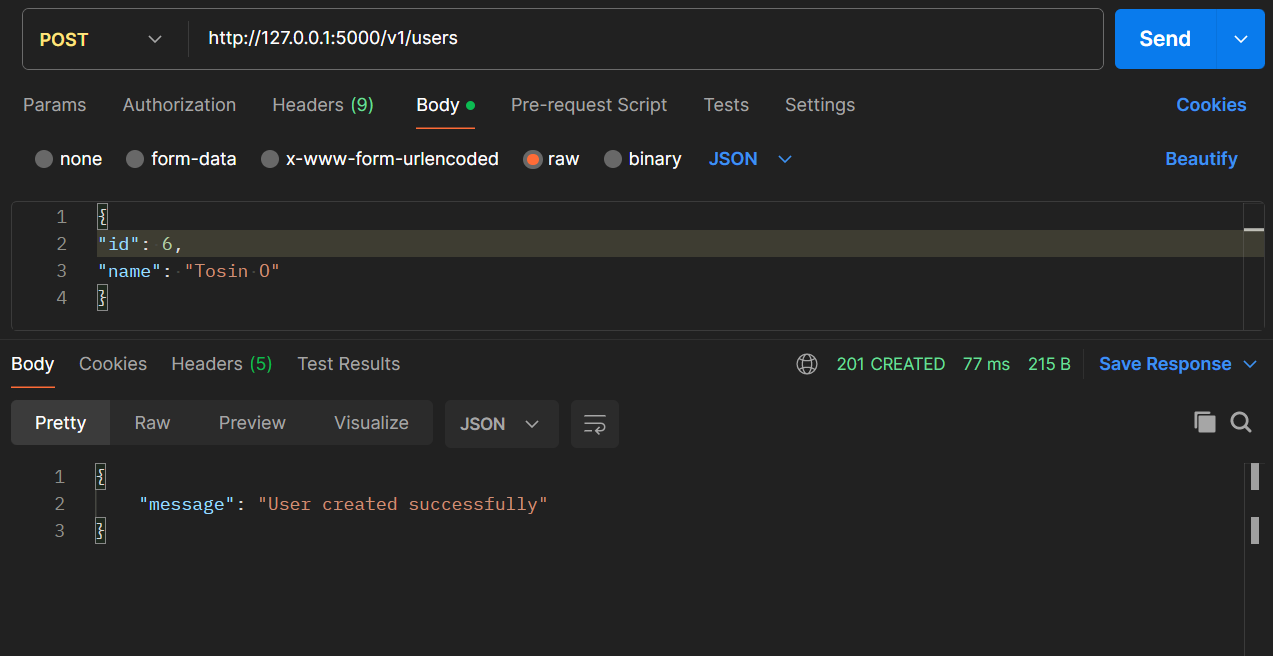
* **GET**: Returns a list of users.
  + Parameters: None
  + Response:
    - Status Code: 200
    - Content Type: application/json
    - Body: Array of user objects



* **POST**: Creates a new user.
  + Parameters:
    - Body: User object (JSON)

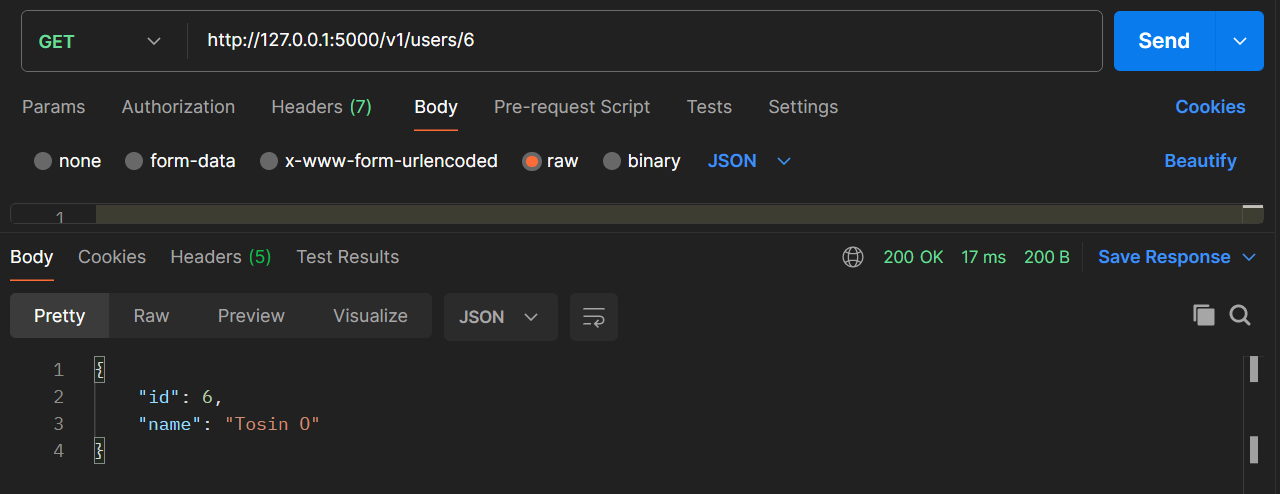


* + Response:
    - Status Code: 200
    - Content Type: application/json
    - Body: User object of the newly created user



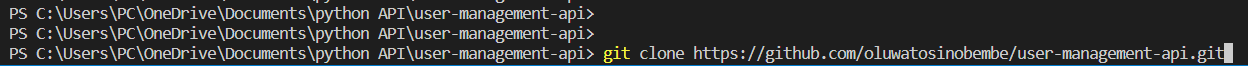
**/v1/users/{userId}**

* **GET**: Returns user details by ID.
  + Parameters:
    - Path Parameter: **userId** (integer) - ID of the user to fetch
  + Response:
    - Status Code: 200
    - Content Type: application/json
    - Body: User object

****

**Local Setup Instructions**

1. A new repository was crated on GitHub and cloned:



1. Dependencies were installed using pip:



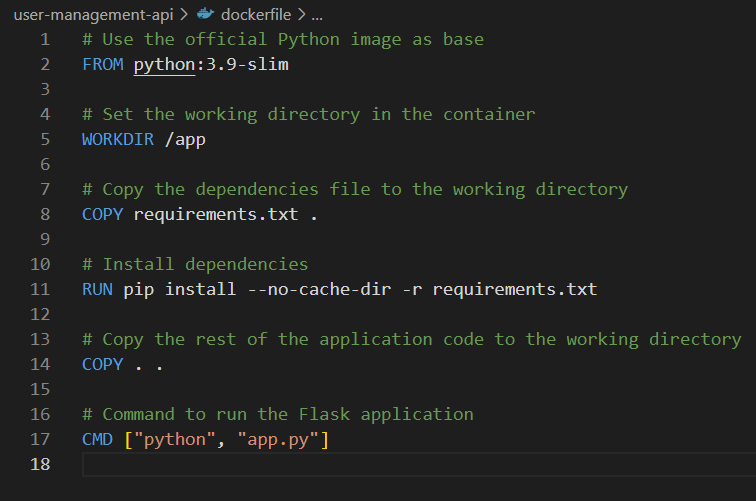
1. Start the API server:



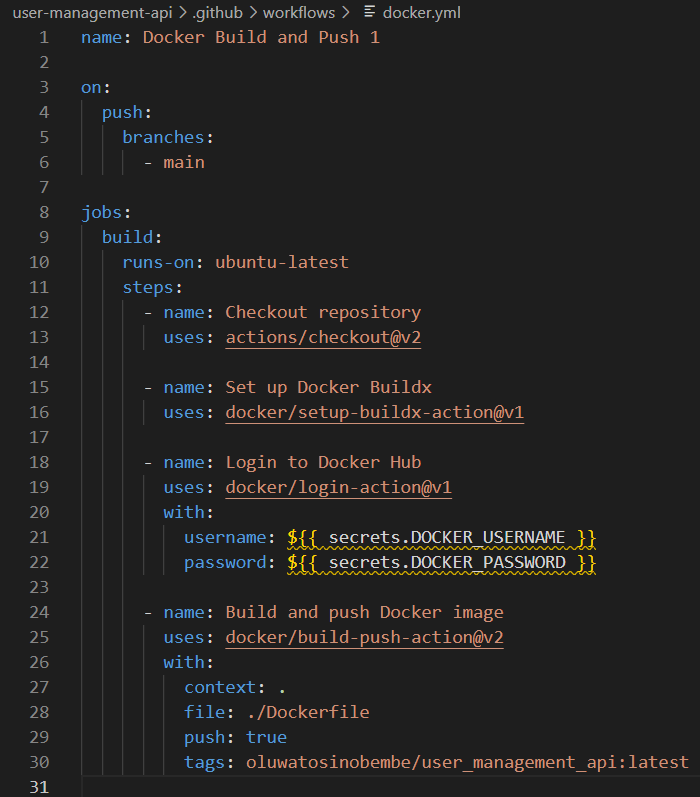
1. The API server will be running at [**http://127.0.0.1:5000**](http://127.0.0.1:5000) (localhost).

**Deployment Instructions**

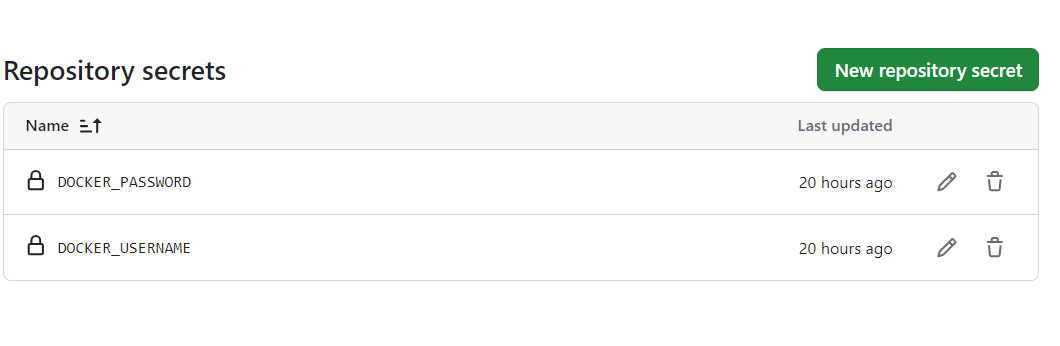
1. The application was containerized using Docker. A dockerfile was created which contained the following configuration:



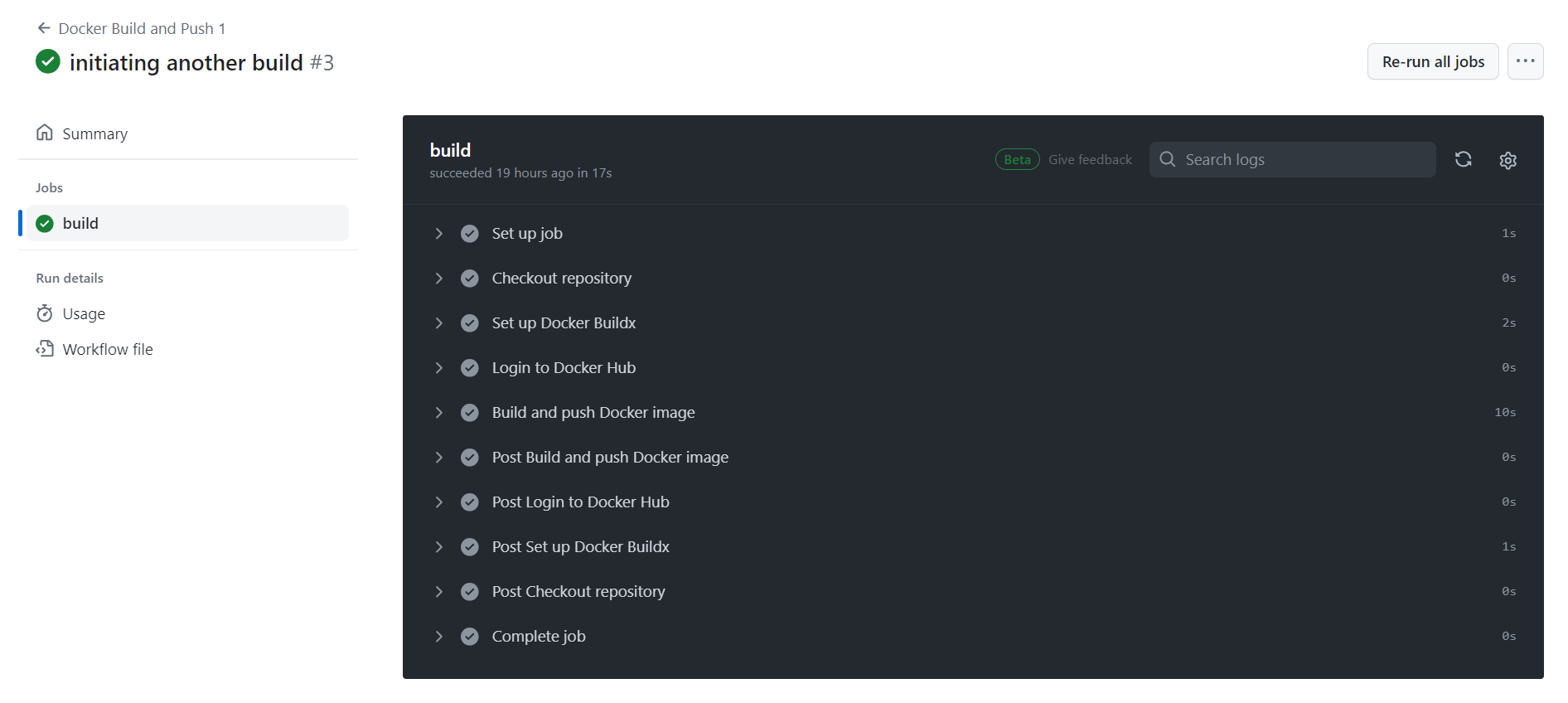
1. Github actions was used to implement the automation of building the Docker image and pushing it to a public Docker repository. A file with the path: *.github\workflows\docker.yml* was created with the following Github actions steps:

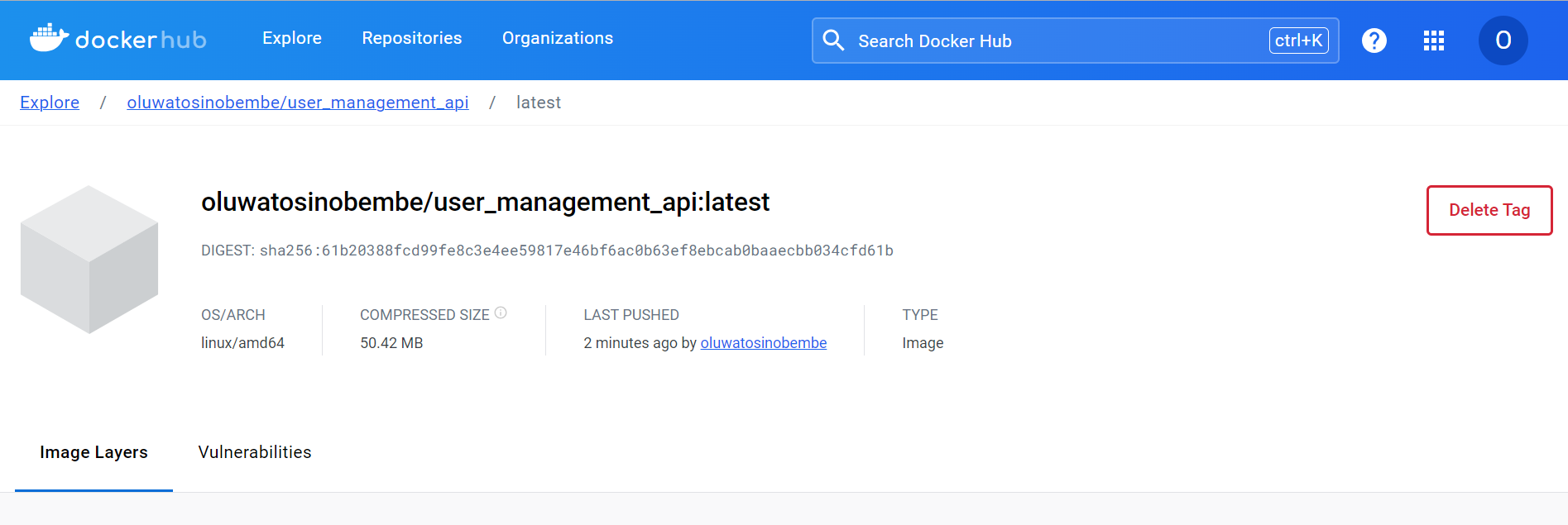


ii. In lines 21 and 22 of the above screenshot, the username and password were stored in the repository secrets in Github.



1. The build process was triggered in Github actions after the configuration files were committed to the public Github repository of this project.

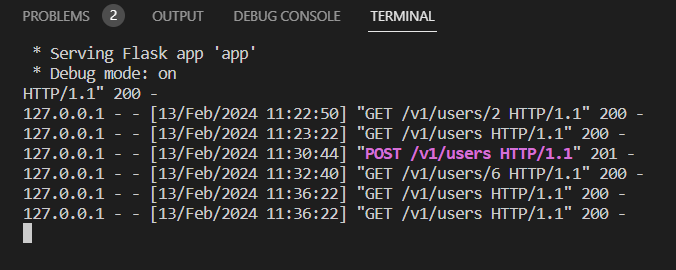




The container image can be viewed at this link: https://hub.docker.com/repository/docker/oluwatosinobembe/user\_management\_api/

**Testing**

1. The test environment was setup using Postman.
2. Tests were run by sending requests to the API endpoints and verifying the responses.



*Log of API calls on the terminal*

**Conclusion**

The User Management API provides a simple and efficient solution for managing user data. It follows a straightforward structure pattern for storing user data and offers a RESTful interface for interacting with the data. By storing user data in a CSV file, the API achieves simplicity and ease of use, making it suitable for small-scale projects and prototyping.