**Module 4 Lab 1: Protecting Resources**

Naga Sumanth Vema

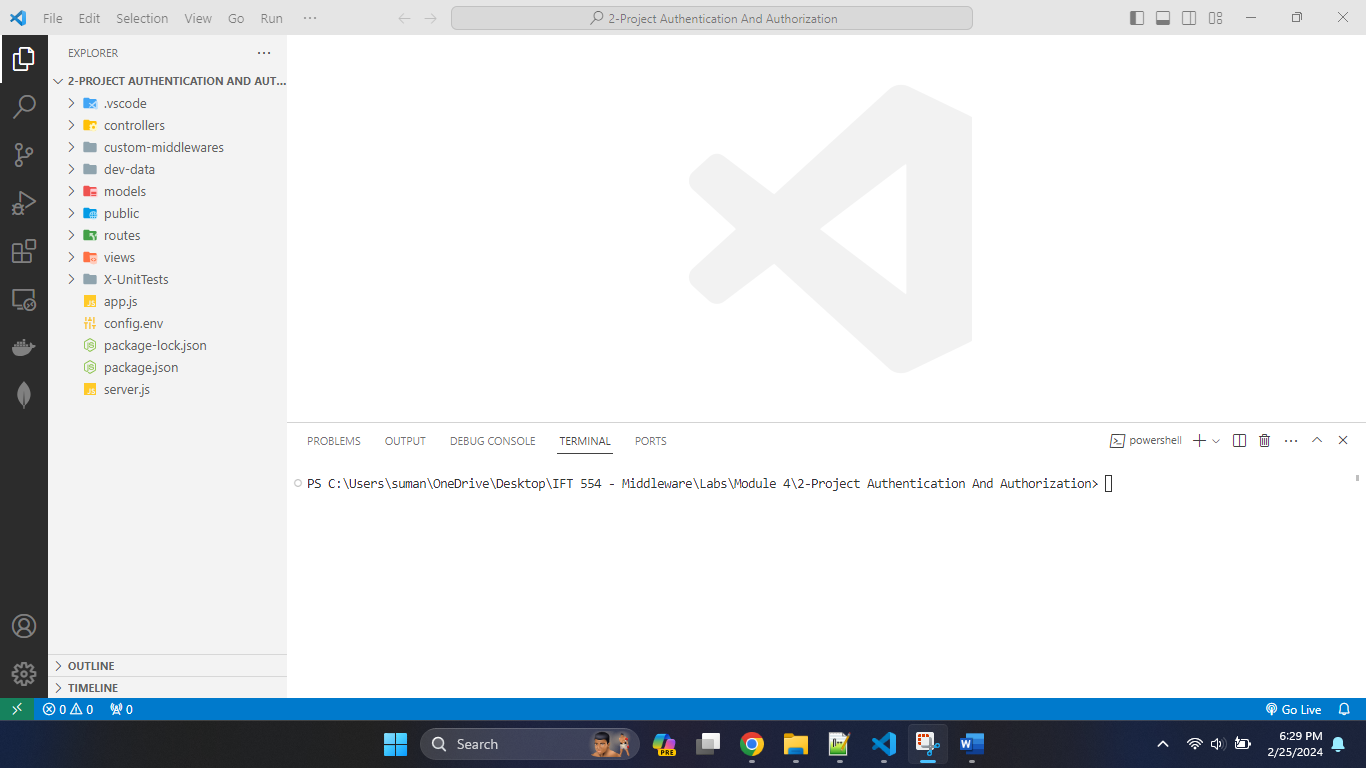
IFT 554: Middleware Programming & Database Security

Dinesh Sthapit

March 03rd, 2024

**Module 4 Lab 1: Protecting Resources**

Screenshot of terminal or command prompt showing that the application has been stopped.

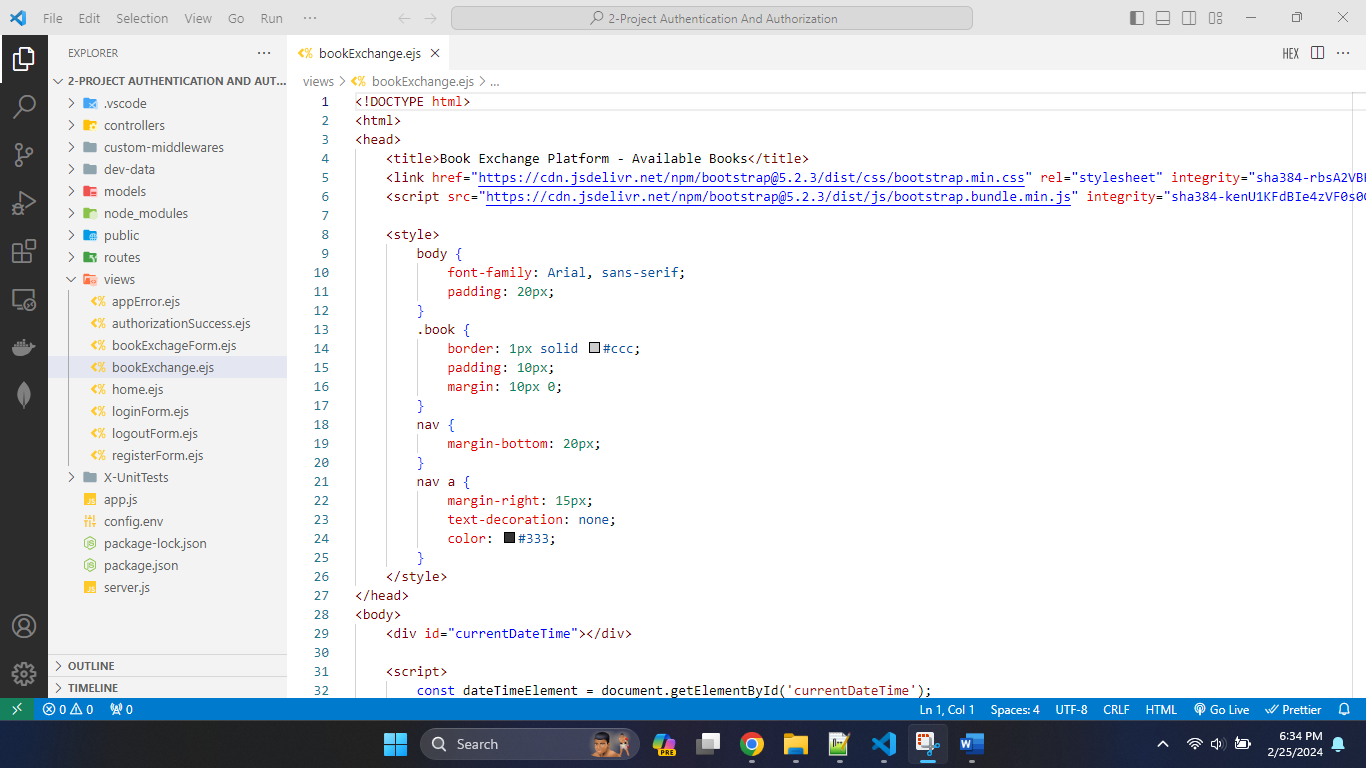


The above screenshot shows that code has not been started running.

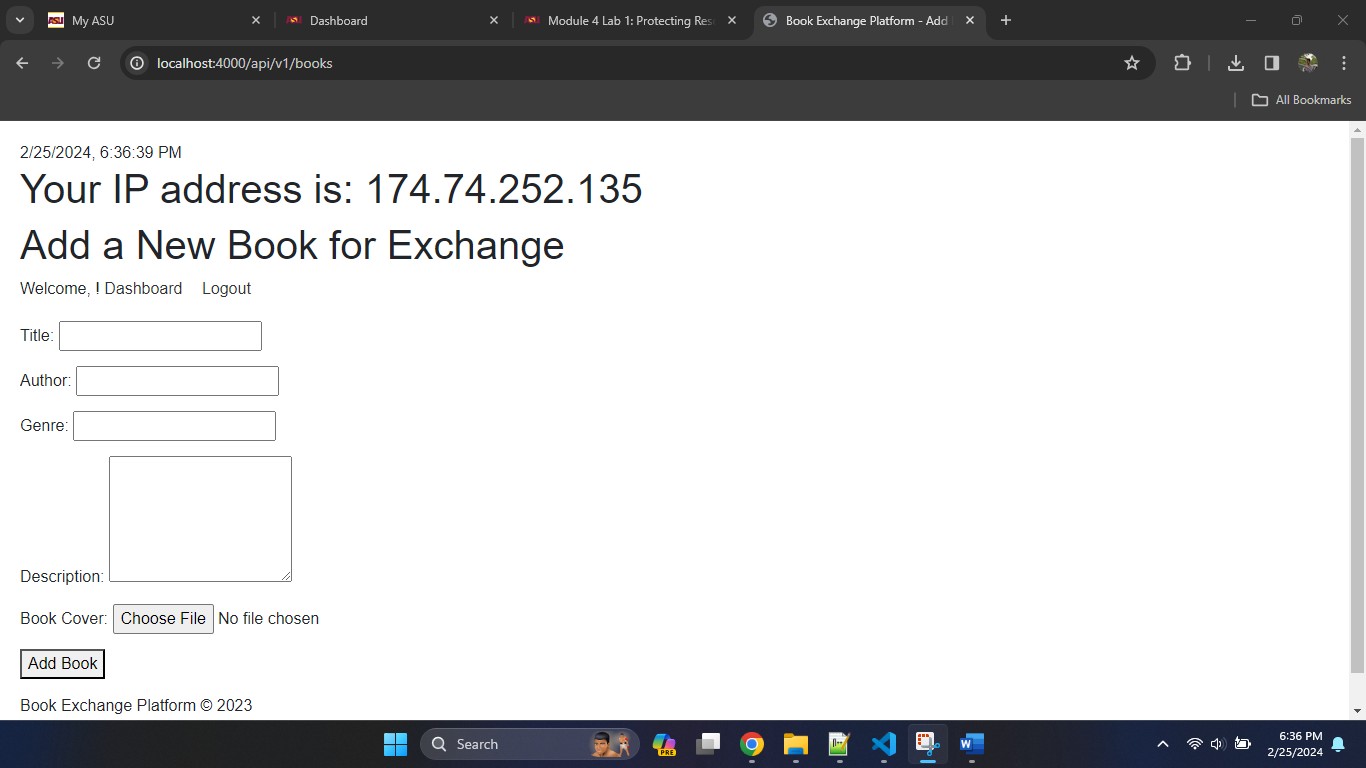
Screenshot of terminal showing successful installation of dependencies.



Screenshot of the EJS file code and its rendered UI in the browser.

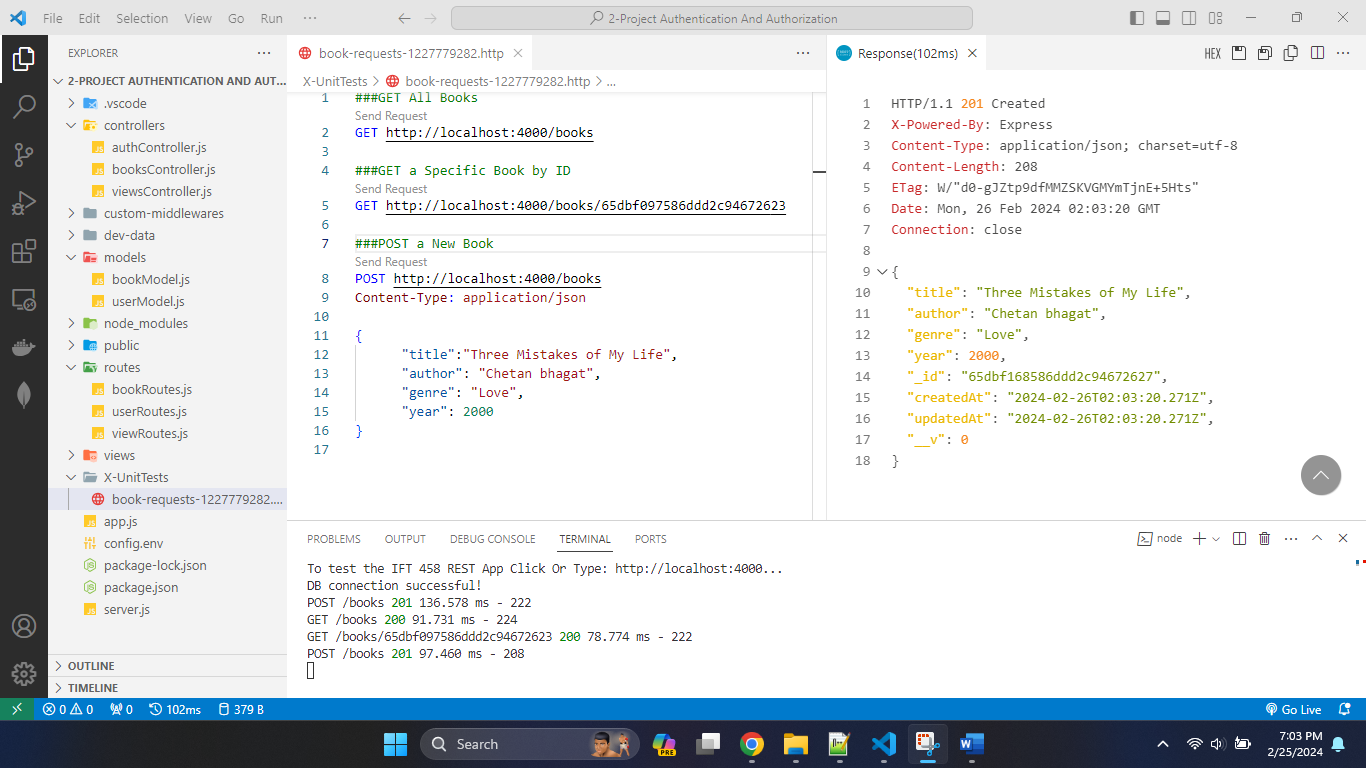


Rendered UI in browser.

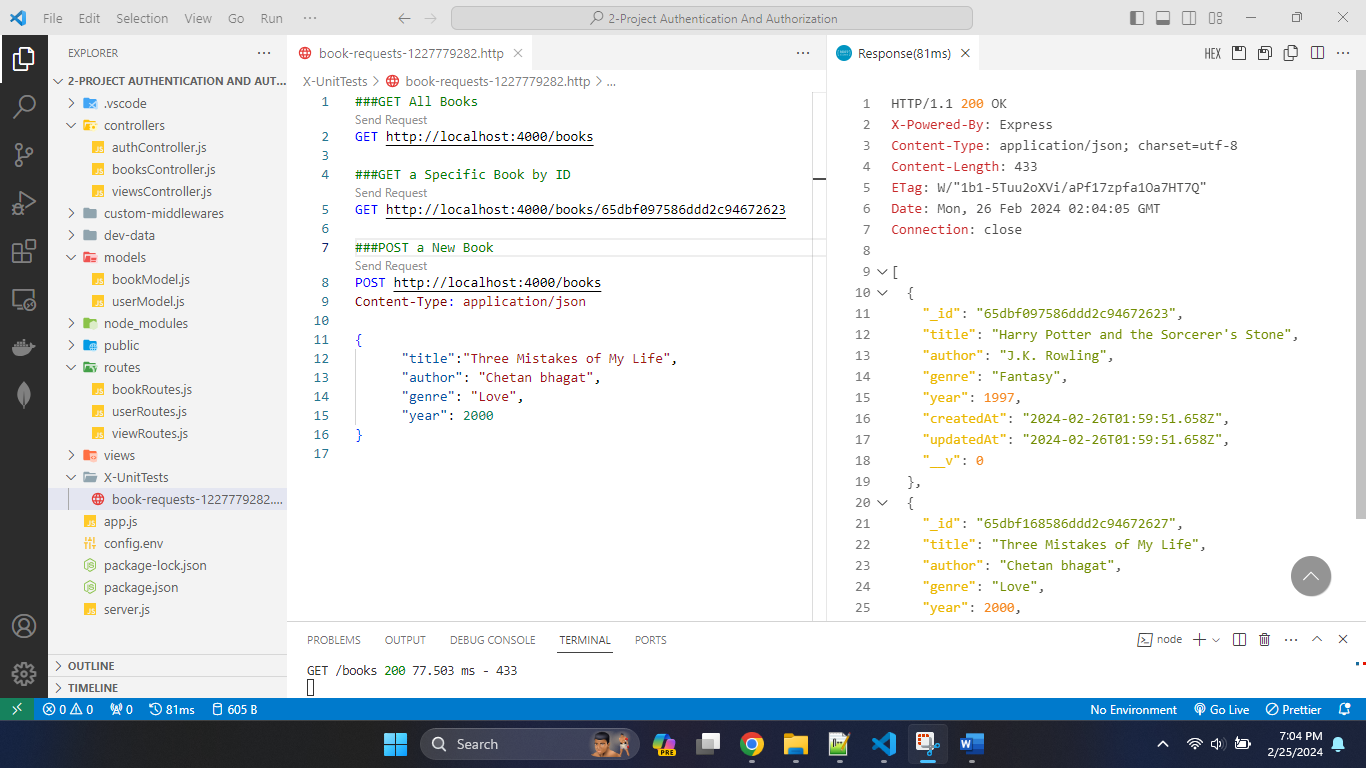


Screenshots of all requests and corresponding responses in the REST Client.

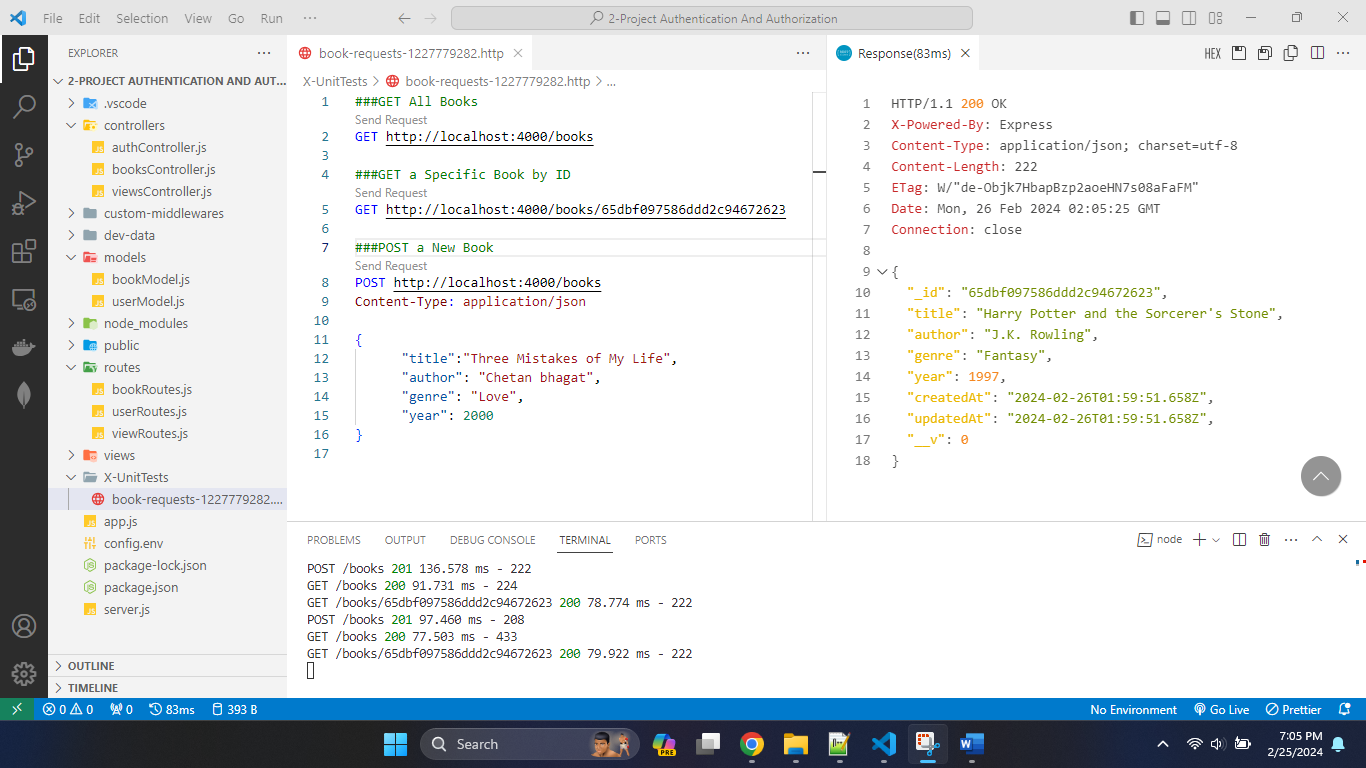
REST Client POST request screenshot:



REST Client GET request screenshot:



REST Client GET By ID request screenshot:



**Process Explanation**

The first step involved setting up the development environment, ensuring the presence of Node.js and npm, and initializing our project. With the environment ready, we proceeded to install necessary dependencies such as Express.js for server-side framework, EJS for templating, and MongoDB driver for database interaction. Moving forward, we developed the backend functionality by creating a server.js file to configure our Express server. This file handled routing, middleware, and database connections. We designed RESTful APIs to facilitate Create, Read for managing books within our application. This backend setup provided the foundation for the functionality of our book exchange platform.

Next, we turned our attention to implementing the frontend aspect of the project. Utilizing EJS as our templating engine, we crafted dynamic views to render user interfaces. Screenshots of the EJS file code alongside its rendered UI in the browser were included to provide a visual representation of the frontend design. This step ensured that users could interact with our platform seamlessly. To ensure the robustness of our application, we rigorously tested our APIs using a REST Client. Screenshots of all requests and corresponding responses, including POST, GET, and GET by ID requests, were captured and documented. These screenshots served as evidence of successful interaction between the frontend and backend components, demonstrating the functionality and reliability of our book exchange website.

In conclusion, through meticulous planning and execution, we successfully developed a book exchange website that allows users to exchange books effortlessly. By leveraging JavaScript, Express, and MongoDB, we created a platform that fulfills the needs of book enthusiasts while showcasing our proficiency in web development.