Project Report - Mahesh Patil

Introduction:

This project involves developing a RESTful API using Node.js and Express.js to manage a collection of tasks. The API supports CRUD operations, and tasks are stored in memory. The application uses EJS for rendering views.

Approach:

1. Environment Setup
    - Installed necessary dependencies: express, ejs, method-override, and uuid.
2. Routes and Handlers
    - Defined routes for each CRUD operation.
    - Implemented handlers to process incoming requests and generate appropriate responses.
3. Views
    - Created EJS templates for displaying tasks, creating new tasks, editing tasks, and showing error messages.
4. Validation and Error Handling
    - Added validation to ensure that tasks have both a title and description.
    - Implemented error handling to provide user-friendly error messages.
5. Testing
    - Tested the API using curl commands and a web browser to ensure all endpoints function correctly.
    - Verified that appropriate status codes and error messages are returned.

Implementation Details:

1. Task Storage
    - Tasks are stored in an in-memory in the form of an array of objects. Each task has a unique ID generated using uuid.
2. Endpoints
    - Implemented endpoints for retrieving all tasks, retrieving a specific task, creating a new task, updating a task, and deleting a task.

3. Validation

    - Ensured that the title and description fields are present in the request body for creating and updating tasks.

4. Error Handling

    - Added error handling middleware to catch unexpected errors and respond with a 500 Internal Server Error.

Challenges and Solutions:

- **Validation**

    - Challenge: Ensuring that all required fields are present for POST and PUT request.
    - Solution: Implemented validation logic in the POST and PUT request handlers.

- **In-Memory Storage**

    - Challenge: Maintaining unique IDs for tasks.
    - Solution: Used the uuid package to generate unique IDs for each task.

Conclusion:

The RESTful API was successfully developed and tested. It supports all required CRUD operations and includes validation and error handling. The use of EJS for rendering views enhances the user experience by providing a clear and organized way to interact with the tasks.