Secure Task Management System (TMS) Marks: 15

Objective:

Build a web application for managing tasks with user authentication using Node.js, Express.js, React, MongoDB, and JWT.

Features:

i. User Authentication
   a. Implement user registration with email verification.
   b. Implement different user roles (e.g., admin, regular user).
   c. Implement user login using JWT (JSON Web Tokens).
   d. Ensure password hashing for security.
ii. Task: Users can create tasks with details: title, description, due date, and priority.
iii. Task Management: Users can perform CRUD operations.
iv. Task List: Display a list of tasks with the ability to sort and filter tasks based on criteria such as priority or due date.
v. Task Details: Users can view detailed information about a task, edit task details, mark tasks as completed, and delete tasks.
vi. Task Categories: Implement the ability to categorize tasks into different categories.
vii. Search and Filters: Provide a search feature and filters to easily find specific tasks.

Technology:

i. Backend: Utilize Express.js to create the server, handle API endpoints for task management, and interact with the MongoDB database.
ii. Database: Use MongoDB to store task information, including task details, due dates, priorities, and categories.
iii. Front-End: Develop a user interface using HTML, CSS, and JavaScript. You can employ popular front-end libraries and frameworks like React, Angular, or anything that you like.

Submission:

i. Submit the codebase via a version control system (e.g., GitHub). [Help 1, 2]
ii. Include a README file with instructions on how to run the application locally.
iii. Submit the GitHub link in the Google Classroom.

Evaluation criteria

i. Correct implementation of user authentication.
ii. Proper usage of JWT for secure routes.
iii. CRUD operations for tasks.
iv. Efficient and secure communication between the front end and back end.
v. Proper error handling and validation.
vi. Clear and organized code structure.