Title : **Task Management System**

Introduction :

- **Project Name:** Task Management System
- **Developed By: Gaurav Gupta**
- **University Roll No: 2200290120070**
- **Branch: Computer Science**
- **Technology Stack:** Node.js, Express.js, MongoDB, JavaScript
- **Purpose:** The project is aimed at creating a task management system that allows users to manage tasks efficiently with features like user authentication, real-time updates, and CRUD operations.

Project Objectives :

- Implement a secure authentication system using JWT.
- Provide task management features (Create, Read, Update, Delete).
- Enable real-time updates using WebSockets.
- Ensure efficient performance using dependency management and modular code structure.

Project Requirements :

Backend:

- Node.js and Express.js for the server.
- MongoDB for data persistence.

Dependencies:

- bcrypt.js: for password hashing.
- JWT (json-web-token): for authentication.
- mongoose: for MongoDB integration.
- socket.io: for real-time updates.

System Design :

- **Authentication:** JWT-based token system for user login and session management.
- **Database:**
  - MongoDB for storing user and task data.
  - Task schema with attributes like title, description, status, and due date.
- **Real-time Functionality:** WebSocket-based updates to provide live task updates.
- **Controllers:**
  - task.controller.js: Manages task CRUD operations.
  - user.controllers.js: Handles authentication (login, registration).
- **Routes:**
  - User and task-related routes to handle API requests.

Implementation :

- **Database Setup:** Connection to MongoDB using mongoose.
- **Task Management API:** RESTful APIs for task creation, viewing, updating, and deletion.
- **Authentication API:** APIs to handle user registration and login.
- **Real-time Update Feature:** Utilizes socket.io for real-time task updates**.**
- **Testing:** The system is tested for performance, security, and real-time interaction.

Features :

- **User Authentication:** Registration, login, and password hashing using bcrypt.js and JWT.
- **Task Management:** CRUD operations for tasks.
- **Real-time Updates:** Live updates for tasks using WebSockets.

Challenges and Solutions :

- **Real-time Updates:** Integrated socket.io for real-time communication.
- **User Authentication:** JWT and password hashing ensured secure user authentication.

Future Scope :

- **Task Categorization:** Adding categories to tasks.
- **Task Prioritization:** Implementing task priorities for better management.
- **User Interface:** Improving UI for a better user experience.
- **Mobile Application:** Extending the functionality to a mobile platform.

Conclusion :

The Task Management System successfully implemented core features like task management, real-time updates, and authentication. It provides a foundation for task handling and user management, with room for future improvements.