COLLEGE CODE : 8203

COLLEGE NAME : AVC COLLEGE OF ENGINEERING

DEPARTMENT : INFORMATION TECHNOLOGY

STUDENT NM-ID : C9C912D4BD1EB53B5A04BF0E7A44DD4E

ROLL NO : 23IT84

DATE :24.10.2025

Completed the project named as Phase5

TECHNOLOGY PROJECTNAME: To-Do App with React Hooks

SUBMITTED BY,

NAME: Ragulkumar A

MOBILE NO: 9360457452

Final Demo Walkthrough

Introduction & Setup (The Hook)

- **Welcome Screen:** Briefly introduce the app's core benefit (e.g., "The simplest way to manage your day and achieve your goals.").
- **Login/Sign Up:** Show the seamless entry process (e.g., "We'll sign in quickly with our existing account").
- **The Main View:** Land on the main dashboard (e.g., "My Day" or "Inbox") and point out the key navigation elements.

Core Functionality: Task Management

- **Creating a Task:**
  - Demonstrate the fastest way to add a new task (e.g., a prominent "+" button).
  - Show how to use natural language input (if supported, e.g., "Buy groceries tomorrow at 5pm").
  - *Example:* "Let's add 'Finish Project Proposal' and see how easy it is."
- **Viewing and Editing a Task:**
  - Click on the new task to open the details pane.
  - Add details: Set a due date and reminder (show the notification prompt).
  - Sub-tasks/Steps: Break down the main task (e.g., "Research," "Draft," "Review").
  - Notes/Description: Add a longer description or important context.
  - Priority/Importance: Mark the task as "High Priority" or "Star" it.
- **Completing a Task:**
  - Check off the completed task.
  - Show the visual feedback (e.g., a satisfying animation, task moves to "Completed" list).
  - *Demonstrate the core loop is complete: Add -> Detail -> Complete.*

Project Report :

Problem Statement

Individuals and small teams often struggle with **task overload** and fragmented planning across various tools (notes, calendars, emails). Existing solutions are often too complex or too basic. This project provides a **centralized, user-authenticated task manager** where users can efficiently record, prioritize, track, and complete their daily tasks and projects, moving their productivity beyond scattered lists. **Objectives**

- To create a **simple, responsive web application** for personal and small-group task management.
- To store task data (title, project/list, due date, status, priority, user ID) in **MongoDB**.
- To enable **full CRUD (Create, Read, Update, Delete)** operations for seamless task management.
- To add **filter and sort** options by due date and task priority.
- To ensure **data privacy** and separation between users through a secure authentication system.

Architecture & Design

Architecture Layers:

- Frontend: React (User Interface)
- Backend: Node.js with Express (API and Business Logic)
- Database: MongoDB Atlas (Scalable Task Storage)

Workflow:

- **User Request:** The authenticated user sends an action request (add, update, delete task, or fetch a list).
- **Express API:** The Express route validates the request (using the JWT token) and executes the necessary database logic.
- **MongoDB:** The database stores and retrieves structured task objects.

- **Response:** The API returns the updated data or status to the **React frontend** for immediate display and UI update.

Tech Stack :

Component	Technology Used
Frontend	React
Backend	Node.js, Express.js
Database	MongoDB
Authentication	JWT
API Testing	Postman
Deployment	Render / Vercel / Netlify
Version Control	Git & GitHub

API Schema & UI Structure API
Endpoints :

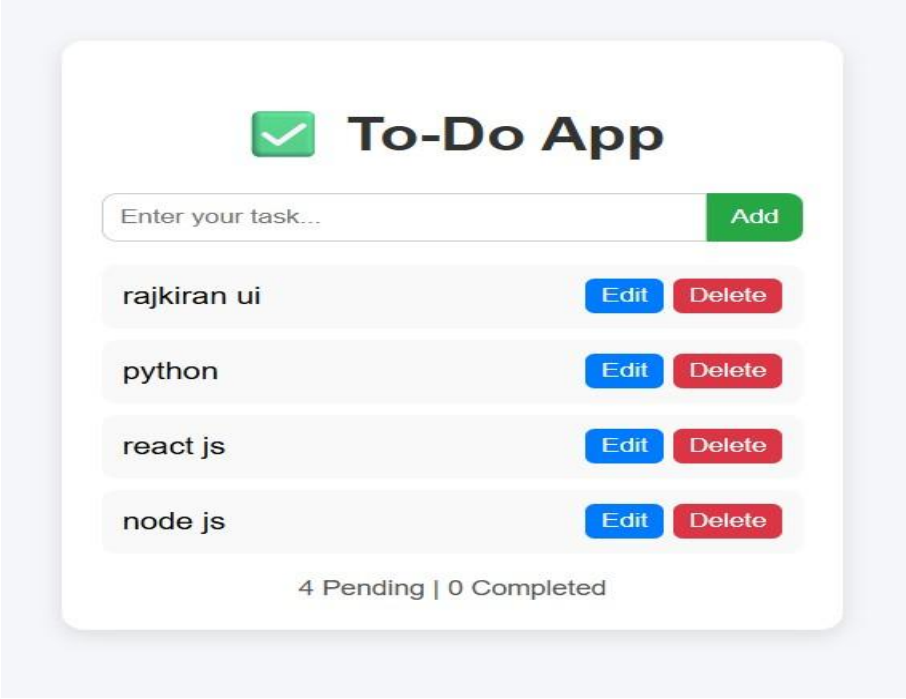
Endpoint	Method	Description
/api/tasks	GET	Fetch all tasks for the authenticated user.
/api/tasks	POST	Add a new task entry.
/api/tasks/:id	PUT	Update details (title, status, due date) of a specific task.
/api/tasks/:id	DELETE	Delete a task entry.
/api/tasks?status=completed	GET	**Filter** tasks by status or priority.
/api/auth/register	POST	Register a new user account.

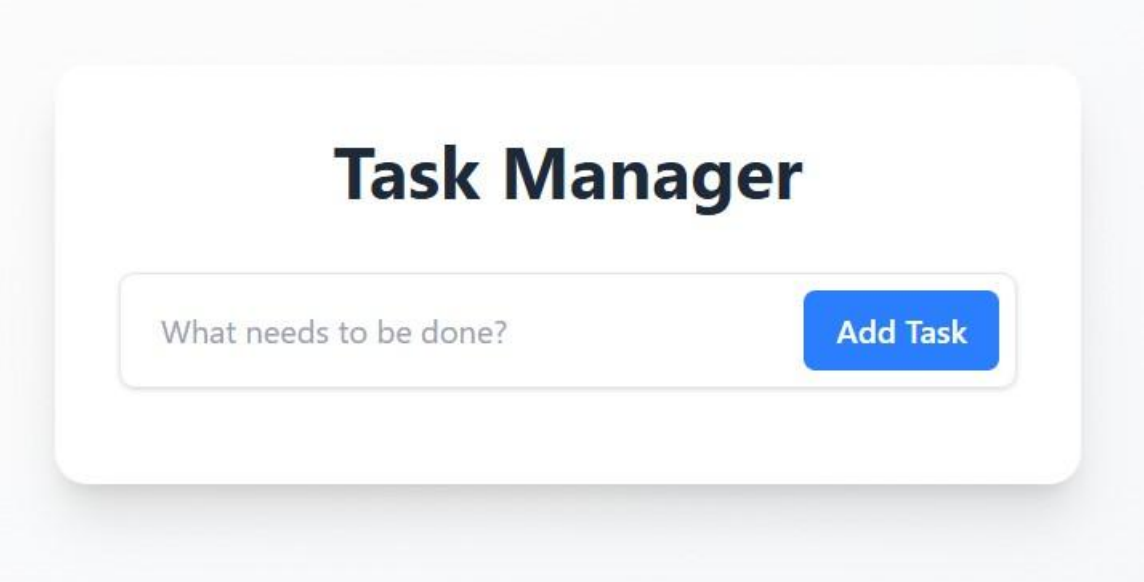
/api/auth/login	POST	Authenticate user and issue JWT.

UI Components :

- Task Form Section: A dedicated area for quickly adding new tasks and a modal/side panel for editing all task details (due date, priority, notes).
- Task List / Grid: The main display, presenting tasks in a clean list or card format, sortable by criteria (e.g., Due Date, Priority).
- Filter & Sort Bar: Interface elements to quickly filter tasks by Completed/Incomplete status and sort by Due Date or Creation Date.
- Login / Register Pages: Secure user sign-up and sign-in interface.
- "My Day" Panel: A simple dashboard showing overdue tasks, tasks due today, and a count of total tasks remaining.

Screenshots (to include):





Challenges & Solutions :

Challenges	Solutions
API routes returning undefined task data	Added **Mongoose schema validation** and robust **error handling** in Express routes to ensure data existence and integrity before sending a response.
Duplicate task entries (e.g., creating the same task twice)	Implemented validation checks in the **Mongoose model** to prevent task titles from being identical within the same user's context.
Authentication session issues (e.g., losing login status)	Utilized **JWT tokens** for stateless authentication and stored the token securely on the client-side; employed **bcrypt** for secure password hashing.
Cross-origin (CORS) errors during development/testing	Explicitly configured and **enabled CORS middleware** in the Express server to allow requests from the React frontend's origin URL.

Deployment failure (MongoDB connection)	Configured **environment variables** securely on the hosting platform (Render) and used the correct, firewallwhitelisted **MongoDB Atlas URI**.

GitHub README & Setup Guide

Project Title: To-Do App with React Hooks

Overview: A full-stack MERN (MongoDB, Express, React, Node.js) web application that enables authenticated users to efficiently create, organize, prioritize, and track their personal tasks in a centralized and secure platform.

Setup Instructions:

1. Clone repository:

- git clone https://github.com/Rajkiran1408/nm-todo-app.git

2. Install dependencies: npm install
3. Create .env file and add your credentials:
4. MONGO_URI= mongodb+srv://kingstar151617_db_user:qkxJHIqLF7ZfGO2U@cluster0.18wyflt.mongodb.net
5. PORT=5000
6. Run the server: npm start
7. Visit: http://localhost:5000 (or the port defined in your .env file).

Final Submission

Deployment:

- Backend (API): Hosted on Render
- Frontend (UI): Hosted on Render
- Database: MongoDB Atlas
- Deployed Link: https://todo-app-hfgn.onrender.com/

- GitHub Repository: https://github.com/ragulkumar839/NM-IBM-AVCCE-Ragulkumar.git
- Future Enhancements
    - Project/List Sharing: Implement secure access control to allow users to share specific task lists (e.g., a "Groceries" list) with other authenticated users.
    - Email Reminders: Set up a cron job or scheduled function to send email notifications for tasks due in the next 24 hours.
    - Calendar View: Integrate a simple calendar component (e.g., FullCalendar) on the frontend to visualize tasks by due date.
    - Drag-and-Drop Prioritization: Enable interactive ordering of tasks within lists.
    - Mobile App Version: Develop a complementary mobile application using React Native to ensure seamless productivity on the go.

End Result

The Minimalist To-Do Task Manager successfully provides a centralized, efficient, and secure platform for organizing tasks. It leverages the power of the MERN stack to deliver fast, reliable CRUD operations, ensuring users can achieve mental clarity and stay productive without juggling multiple, scattered lists.