COLLEGE CODE : 9111

COLLEGE NAME : SRM Madurai College for Engineering and Technology

DEPARTMENT : Artificial Intelligence and Data Science

STUDENT NM-ID :

Arun Sethupathy S	460E3E83FE52DF4AB724E9388531E34D	911123243009
Jeyasuriya R S	70a76be95faf771bad04bd859e630b22	911123243023
Kirthick Kumar N J	0E48EC068D3E28885A943CE27AE87724	911123243025
Kishore S	F1F444CC987BC16854346CEC685C9902	911123243301
Ram Visshal G A	7BE0F4D2FB3FA33956045D734A1C0D72	911123243340

DATE : 6.10.2025

Completed the project named as Phase 5

TECHNOLOGY PROJECT NAME : To Do List Application

SUBMITTED BY,

Arun Sethupathy S	6383627990
Jeyasuriya R S	6381271778
Kirthick Kumar N J	6381388587
Kishore S	7010834912
Ram Visshal G A	9245849024

Phase 5 — Project Demonstration and Documentation

1. Final Demo Walkthrough

In this final demo, I present the fully functional To-Do List application that allows users to create, prioritize, categorize, and manage tasks efficiently. The app features:

- Task prioritization with clear color-coded indicators (High - red, Medium - yellow, Low - green).
- Due dates and reminder notifications via browser alerts.
- Task filtering by status, priority, and categories.
- Responsive UI design optimized for desktop and mobile devices.
- Secure user authentication with JWT for personalized task management.
- Real-time sync across devices through WebSocket integration.
- Smooth UI animations for task addition and deletion.

The live demonstration highlights these features working seamlessly, showing the app's usability and robustness.

---

2. Project Report

Project Title: Advanced To-Do List Application
Objective: Develop a user-friendly and scalable task management app with enhanced features and secure deployment.

Technology Stack: React, Node.js, Express, MongoDB, Socket.IO, JWT, Tailwind CSS, Netlify

Architecture:

- Frontend built in React with Tailwind CSS for styling and Framer Motion for animations.
- Backend REST API supporting full CRUD operations, authentication, and real-time updates.
- MongoDB for persistent task storage.

Key Features Implemented:

- Task prioritization, due dates, categories, filtering/sorting
- User authentication and multi-device synchronization
- Responsive design and accessibility improvements
- API rate limiting and input validation for security
- Deployment on Netlify with continuous integration via GitHub Actions

Challenges and Solutions:

- Ensuring real-time synchronization without data conflicts was challenging; implemented optimistic UI updates and conflict resolution strategies.
- Managing authentication tokens securely across devices required careful storage and refresh token logic.
- Achieving responsive UI consistency involved extensive testing and media queries refinement.

Testing:

- Comprehensive unit and integration tests with Jest and React Testing Library.
- Manual cross-browser and device testing to ensure reliability.

---

3. Screenshots and API Documentation

Screenshots:

- Task creation form with priority selection
- Task list filtered by category and sorted by due date
- User login and registration screens
- Responsive layouts on desktop and mobile

API Documentation (Excerpt):

Endpoint	Method	Description	Auth Required	Request Body	Response
/api/tasks	GET	Retrieve all tasks for user	Yes	None	Array of task objects
/api/tasks	POST	Create a new task	Yes	`{ title, description, priority, dueDate }`	Created task object
/api/tasks/:id	PUT	Update a specific task	Yes	Partial or full task update fields	Updated task object
/api/tasks/:id	DELETE	Delete a specific task	Yes	None	Success message
/api/auth/login	POST	User login	No	`{ email, password }`	JWT token

---

4. Challenges & Solutions

- **Challenge:** Handling real-time multi-device sync without race conditions.
  Solution: Implemented WebSocket events with optimistic updates and versioning to resolve conflicts.

- **Challenge:** Securing API endpoints and preventing token misuse.
  Solution: Adopted JWT with refresh tokens and HTTP-only cookies to safeguard authentication flow.
- **Challenge:** Maintaining responsive design across many screen sizes.
  Solution: Used Tailwind CSS utilities and media queries extensively; tested on real devices.

---

5. GitHub README & Setup Guide

README Highlights:

To-Do List Application

An advanced task management app with features including task prioritization, due dates, categories, real-time sync, and secure user authentication.

Features

- Create, edit, and delete tasks with priorities and due dates
- Filter and sort tasks by various parameters
- Responsive design for desktop and mobile
- User registration and login with JWT authentication
- Real-time task synchronization across devices

Setup

1. Clone repository
2. Install dependencies: `npm install`
3. Configure environment variables (`.env`) for backend API URL and JWT secrets
4. Run backend server: `npm run server`
5. Run frontend app: `npm start`
6. Visit `http://localhost:3000` in your browser

Testing

Run tests with `npm test`.

---

6. Final Submission

GitHub Repository: https://github.com/ramvisshal/NM-Project.git