9530

St. MOTHER THERESA ENGINEERING COLLEGE

COMPUTER SCIENCE ENGINEERING

NM-ID: 9D1F84DA855FD45936B4653F42567E9C

REG NO: 953023104033

DATE:07-10-2025

Completed The Project Named as

Phase 5

TO-DO LIST APPLICATION

SUBMITTED BY,

IFFAT FATHIMA I

9677874707

Phase 5 - Project Demonstration & Documentation

Final Demo Walkthrough

Overview:

The To-Do List Application is a simple yet powerful task management tool designed to help users organize daily activities efficiently. It allows users to create, update, mark, and delete tasks — all in a clean and responsive interface.

Step-by-Step Demo Walkthrough

1. Application Launch

Open the application via the deployed link (e.g., on Netlify or Vercel).

The home screen displays the app title “To-Do List” and an input box to add new tasks.

2. Adding a Task

In the input field, type a new task (e.g., “Complete documentation”).

Click the “Add” button.

The task appears instantly in the task list below with options to mark complete, edit, or delete.

3. Marking a Task as Complete

Click the checkbox or “ ✔ ” icon beside a task.

The task text becomes strikethrough or moves to a “Completed Tasks” section.

This helps users track finished activities easily.

4. Editing a Task

Click the “Edit” icon beside a task.

The task text becomes editable — modify the content and press “Save” to update it.

5. Deleting a Task

Click the “ Delete” icon beside any task.

The selected task is removed from the list immediately.

6. Data Storage (Persistence)

All tasks are saved in Local Storage (or Firebase, if implemented).

Even after refreshing or closing the browser, all tasks remain intact.

7. Responsive Design

The app layout adjusts smoothly across desktop, tablet, and mobile views.

UI elements (buttons, text boxes, task cards) resize automatically.

8. Optional Enhancements (if implemented)

Dark Mode Toggle — Switch between light and dark themes.

Task Filtering — View All / Active / Completed tasks.

Due Dates / Priority Levels — Organize tasks more effectively.

Project Report

Objective:

To design and develop a web-based application that allows users to efficiently manage their daily tasks, set priorities, and track task completion in a user-friendly interface.

Problem Statement:

Managing daily tasks manually or on paper often leads to missed deadlines and disorganization. This application provides a digital solution for organizing tasks effectively, improving productivity and time management.

Key Features:

1. Add Tasks: Users can create new tasks with a title, description, and optional deadline.

2. Edit & Delete Tasks: Modify or remove tasks as needed.

3. Mark as Completed: Users can mark tasks as completed and view completed tasks separately.

4. Categorization/Prioritization: Assign priority levels or categories to tasks for better organization.

5. Search & Filter: Quickly find tasks by keyword or filter by status (pending/completed).

6. Responsive UI: Works seamlessly on desktops, tablets, and mobile devices.

7. Data Persistence: Stores tasks locally (or via backend/database) to retain data across sessions.

Technology Stack:

Frontend: HTML, CSS, JavaScript

Backend (optional): Node.js / Firebase / Express.js

Database (optional): Local Storage / Firebase Realtime Database / MongoDB

Deployment: Netlify / Vercel / Cloud Platform

Implementation:

UI Design: Clean and minimal interface with intuitive navigation.

Task Management Logic: JavaScript functions handle adding, editing, deleting, and marking tasks as completed.

Data Storage: Tasks are saved in local storage or a database to ensure persistence.

Testing: Verified functionality across browsers and devices to ensure smooth performance.

Enhancements (Optional):

Notifications/reminders for deadlines

Drag-and-drop functionality for task reordering

User authentication for multiple user accounts

Outcome:

Simplifies task management for users

Improves productivity by tracking tasks efficiently

Provides a scalable foundation for future enhancements

Conclusion:

The To-Do List Application demonstrates effective project planning, design, and implementation. It offers a practical tool for daily task management while showcasing skills in web development, UI/UX design, and data handling.

Screenshots/API Documentation

Include screenshots to visually demonstrate key features and workflow of the application. Suggested screenshots:

1. Home/Dashboard: Showing all tasks (pending and completed) with options to add, edit, delete, or mark tasks.

2. Add Task Form: Input fields for task title, description, deadline, and priority.

3. Edit Task: Editing an existing task with updated details.

4. Completed Tasks: Display of tasks marked as completed.

5. Responsive View: How the application looks on mobile and tablet screens.

6. Search & Filter Feature: Filtering tasks by keyword, category, or status.

API Enhancements:

If your To-Do List app uses APIs (either custom backend or third-party), you can highlight enhancements for better functionality and integration. Examples:

1. CRUD API Endpoints:

POST /tasks – Add a new task

GET /tasks – Retrieve all tasks

PUT /tasks/:id – Update a task

DELETE /tasks/:id – Delete a task

2. User Authentication API (Optional):

POST /register – User registration

POST /login – User login

Secure token-based authentication (JWT) for accessing user tasks

3. Search & Filter API:

GET /tasks?status=pending – Fetch pending tasks

GET /tasks?priority=high – Fetch tasks by priority

4. Real-Time Sync (Optional):

Use WebSocket or Firebase Realtime Database to update tasks in real-time across devices

5. Enhancements Made:

Input validation on APIs to avoid incorrect data

Error handling and response messages for smooth user experience

API optimized for performance and security

Challenges and Solutions

1. Data Persistence Across Sessions

Challenge: Ensuring that tasks remain saved even after the browser is closed or the page is refreshed.
Solution: Implemented local storage for storing tasks on the client-side. For multi-user or cloud-based versions, integrated Firebase Realtime Database to store and sync tasks in real-time.

2. Responsive Design

Challenge: Making the application usable across different devices (desktop, tablet, mobile).
Solution: Used CSS Flexbox/Grid and media queries to create a responsive UI that adapts to various screen sizes.

3. Task Management Functionality

Challenge: Implementing seamless add, edit, delete, and mark-complete functionality without bugs or errors.
Solution: Developed modular JavaScript functions for each operation and performed unit testing for every function to ensure reliability.

4. Search and Filter Efficiency

Challenge: Allowing users to quickly search or filter tasks, especially as the number of tasks grows.
Solution: Implemented efficient search and filter logic using JavaScript array methods (filter, find, map) to minimize lag and improve performance.

5. Error Handling and Validation

Challenge: Preventing incorrect or empty task entries from being added.
Solution: Added input validation checks for all fields and displayed user-friendly error messages to guide users.

6. API Integration (Optional for Backend Version)

Challenge: Ensuring smooth communication between frontend and backend APIs, handling asynchronous requests.
Solution: Used Fetch API / Axios with proper async-await handling, error catching, and status checks.

7. Real-Time Updates (Optional Enhancement)

Challenge: Syncing task changes instantly across multiple devices/users.
Solution: Integrated Firebase Realtime Database or WebSocket for instant updates whenever a task is added, updated, or deleted.

GitHub README & Setup Guide

Setting up the project on GitHub

1.Create a new repository:

- Log in to GitHub and click the + icon in the top-right corner to create a new repository.
- Enter a name for your repository, for example, my-todo-list.
- Add a short description.
- Select Initialize this repository with a README. This creates the README.md file automatically.
- Click Create repository.

2.Write and edit the README.md:

- On your new repository page, click the pencil icon to edit the README.md file.
- Use the Markdown template above to write the content for your project.
- The edit page offers a "Preview" tab to see how your changes will render.

3.Push your code:

- Clone your new, empty repository to your local machine using git clone.
- Copy your project files into the local repository folder.
- Add your files with git add .
- Commit your changes with git commit -m "Initial commit for todo list project"
- Push to GitHub with git push origin main (or master).

4.Add a license:

- On your GitHub repository page, click the Add file button and choose Create new file.
- Name the file LICENSE and click Choose a license template.
- Select a license like MIT from the list and fill in the details. This will automatically populate the file for you.

Final Submission (Repo + Deployed Link)

The GitHub repository link containing the full source code, documentation, and project history.

REPOSITORYLINK: **https://github.com/iffatfathimaa777-tech/Naan-muthalvan.git**

The deployed application link (using platforms like Vercel, Netlify, or Git Hub Hosting), so evaluators can interact with the live version of the Chat Application UI.

LINK: **https://iffatfathimaa777-tech.github.io/Naan-muthalvan/**

All supporting documents including the project report and screenshots will also be shared as part of the submission package.