Task 1: Design and development of a web application (Conception Phase)

Task Manager Web Application

10246379
Hriday Dewan

Table of Contents

1. Conception Phase

The conception phase mainly develops to show the major idea of developing the project as well as define objectives, scope, target group, project plan, methodology, functional and non-functional requirements, as well as system design and uses technologies. In the section below, the detailed description is outlined.

1.1 Project Profile

Project Name

Task Manager Web Application.

1.1.1 Objectives, Scope, and Target Group

Objectives

The major agenda of this project is to develop an intuitive Task Manager that assists the user in managing their day-to-day activities effectively. The application will enable users to add, delete, prioritise, search, and track tasks, as well as deliver a visually attractive interface with dark mode support and persistent storage.

Scope

The application will incorporate necessary functionalities like adding tasks, deleting tasks, priority assignment, search functionality, and theme customisation (dark mode). The data will be saved in the Local Storage of the browser to remain persistent without having to be stored on the server.

Target Group

Here, our main target audience will be the students, professionals, and other people who require task management for productivity. Also, the interface is created to be user-friendly and simple, with limited web literacy.

1.1.2 Risks, Project Plan, and Project Organisation

Risk

- It is possible that the user accidentally deletes tasks without confirmation.
- Also, another risk is that the local Storage has limited capacity and may not work in all devices.
- Browser compatibility issues could arise if outdated browsers are used.

Risks and Mitigation

Risk	Description	Mitigation
Accidental deletion of tasks	Users may delete tasks by mistake	Add a confirmation prompt before deletion

Local Storage capacity	Browser Local Storage is limited (~5 MB)	Optimize storage usage; notify users when storage is near full
Browser compatibility	Some older browsers may not support ES6/Local Storage	Test across modern browsers; provide fallback messages

Table 1: Risks and Mitigation

Project Plan

Sprint	**Weeks**	**Plan**
Sprint 1	Week 1	Define functional and non-functional requirements Set up GitHub repository and project structure
Sprint 2	Week 2	Implement core task CRUD (Create, Read, Update, Delete) functions Build and test priority and due date features Initial UI styling (light theme)
Sprint 3	Week 3	Add search and dark mode functionality Implement Local Storage persistence Test across Chrome, Firefox
Sprint 4	Week 4	Conduct user testing with sample users Fix bugs and UI inconsistencies Prepare documentation and project report
Sprint 5	Week 5	Prepare GitHub README and final PDF submission Add screenshots & minor UI polish

Table 2: Project Plan

Project Organisation

It is an individual project, as well as developer (student) does all development, testing and documentation. Here, we use GitHub as an external tool for version control.

1.2 Software Development Methodology

For this project, the developer has chosen an Agile-inspired iterative development methodology. The reasoning, functional and non-functional requirements are underscored in the section below,

1.2.1 Reasoning

It is underscored from the scope that it is a user-friendly and simple application, and it is beneficial to do continuous improvement as per the testing and feedback, that reason agile approach is appropriate. The project is subdivided into small tasks, which can be developed, tested and refined in a short time. Feedback can be counted in every iteration, and it confirms that the final product meets user needs.

1.2.2 Functional and Non-Functional Requirements

Functional Requirements

ID	Requirement	Description
FR1	Add Task	User can add a new task with description, priority, and due date.
FR2	Delete Task	User can delete existing tasks with confirmation prompt.
FR3	Edit Task	User can modify existing task text by double-clicking it.
FR4	Complete Task	User can mark tasks as completed by clicking on them.
FR5	Search Task	User can filter tasks by keyword in under 500ms.
FR6	Theme Toggle	User can switch between light and dark modes instantly
FR7	Data Persistence	Tasks are stored in browser’s Local Storage and persist after refresh.

Table 3: Functional Requirements

Non-Functional Requirements

ID	Category	Requirement
NFR1	Performance	App should load in ≤ 2 seconds on standard broadband.
NFR2	Usability	Interface must be intuitive and accessible with minimal guidance.
NFR3	Responsiveness	App layout must adapt to screen sizes from 360px to 1920px.
NFR4	Reliability	Data must persist across sessions without corruption.
NFR5	Compatibility	Must run on latest versions of Chrome, Edge, and Firefox..
NFR6	Maintainability	Code must be modular (HTML, CSS, JS separated).
NFR7	Security	Input fields must be sanitized to prevent script injection.

Table 4: Non-Functional Requirements

1.2.3 Glossary

Term	Definition
Dark Mode	A theme with darker background and light text to reduce eye strain
Due Date	(Future enhancement) A date associated with a task’s deadline
Local Storage	A browser-based storage mechanism for persisting small amounts of data
Local Storage	Browser-based storage for persisting data locally.

Priority	The importance level of a task (High, Medium, Low)
Task	A unit of work or activity entered by the user

Table 2: Glossary (sorted alphabetically)

1.3 System Design

The Task Manager application is developed based on a single-page layout rendered in HTML and CSS. JavaScript contains all logic, task management, interactions and data storage. The Local Storage of the browser is used to save tasks so that data can be used across sessions. Here, the developer uses a mAodular approach, such as separating user interface elements from data handling to boost maintainability and accelerate user experience.

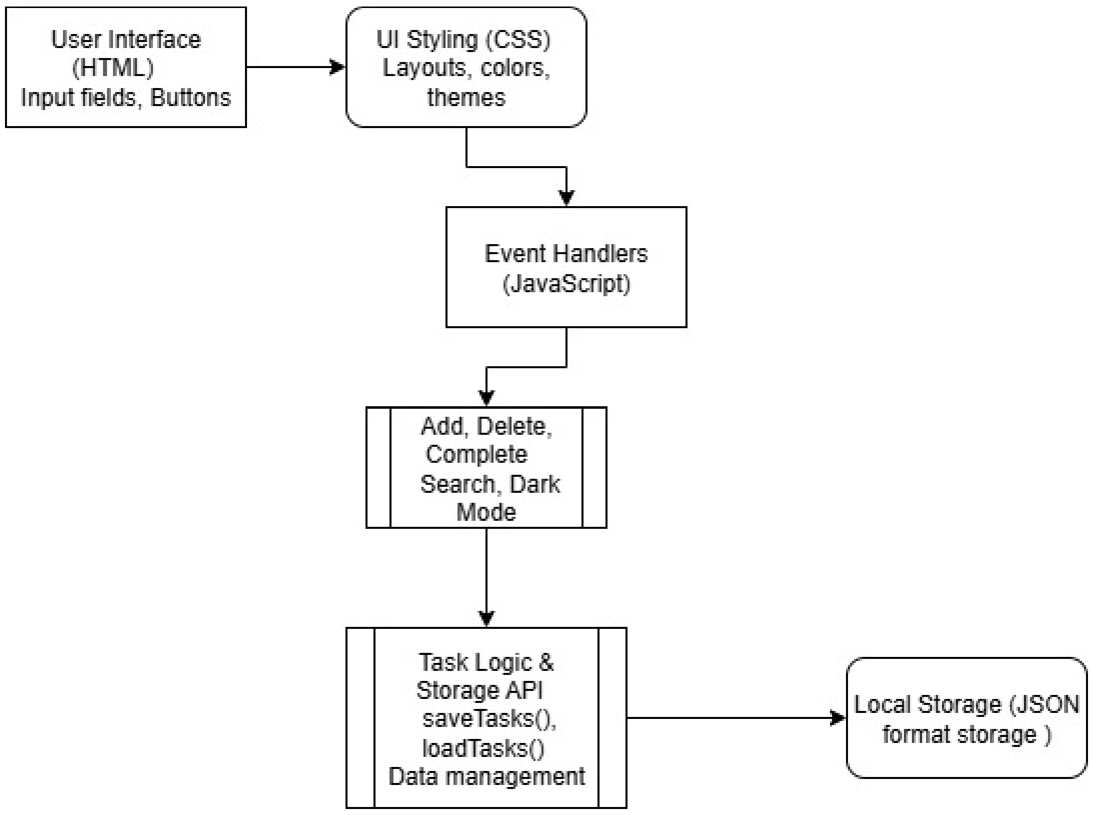
User Interface (HTML) Input fields, Buttons
UI Styling (CSS) Layouts, colors, themes
Event Handlers (JavaScript)
Add, Delete, Complete Search, Dark Mode
Task Logic & Storage API saveTasks(), loadTasks() Data management
Local Storage (JSON format storage)

Figure 1: System Architecture Diagram

(Source: Developed by draw.io)

It has been underscore from the above architecture diagram that researcher shows the three-layer layout of Task Manager web application. Here, we underscore how user interface (HTML/CSS) can be used to deal with layout and design, and JavaScript logic can be used to deal with user interaction, such as adding, deleting, or updating tasks. The Local Storage of the Web browser is a persistent storage of data. Arrows are used to show the data flow between the components and indicate how the user interacts with the storage and dynamically displays the changes in the user interface.

1.4 Technologies and Tools

HTML5

- Provides the base structure of the application.

CSS3

- Enables responsive design, styling, and dark mode.

JavaScript (ES6)

- Implements logic, DOM manipulation, and task management.

Local Storage API

- Provides lightweight persistent storage. While limited (~5MB), it avoids backend complexity and is suitable for small-scale applications.

GitHub

- Used for version control and documentation tracking.

1.4.1 Technology Selection Rationale

React, Vue, and Angular were frameworks that were thought about in developing the Task Manager. But in the simplicity and small scale of this application, these frameworks would have added complexity and accelerated resource load.

The use of Vanilla JavaScript confirms,

- Lightweight performance and faster initial load.
- No dependency installation or build process required.
- Complete browser compatibility without additional frameworks.

This practice is in line with the objective of simplicity, accessibility, and performance of the project.

1.4.2 Motivation

The aforementioned tools have been delivering a lightweight, accessible framework that does not need any server configuration and enables all the necessary functionalities. CSS animations boost user experiences, and Local Storage offers persistence without any complications.