

**To-Do List Application**

**Phase 4**

**COLLEGE CODE :** 9506

**COLLEGE NAME :** Einstein College of Engineering

**DEPARTMENT :** B.E COMPUTER SCIENCE

**STUDENT NM-ID :** 569499d7d464f30a31f49877e45750c8

**ROLL NUMBER :** 950623104104

**DATE :** 26-09-2025

**Submitted By,**

**NAME :** M.THIRUMALAIKUMAR

**MOBILE NUMBER :** 9361350749

**Phase 4 — Enhancements & Deployment**

**Additional Features**

**Inline Editing:**

**Description :**

Users can directly modify the task text by double- clicking on it.

**How it Works:**

* + On double-click, the task text turns into an input field.
  + User can press **Enter** to save changes or **Escape** to cancel.
  + On losing focus (blur event), the task reverts to normal display.

**Benefits:**

* + Faster task updates without navigating to a separate edit page.
  + Improves user experience and efficiency.

**Implementation:**

Uses JavaScript event listeners (dblclick, keydown, blur) to manage inline editing dynamically.

**Drag & Drop:**

**Description:**

Tasks can be rearranged by dragging and dropping to a new position.

**How it Works:**

* Each task is draggable using HTML draggable attribute.
* dragstart captures the dragged task ID.
* dragover allows dropping.
* drop swaps tasks in the array and updates localStorage.

**Benefits:**

* Users can prioritize tasks visually.
* Provides intuitive task management.
* Maintains order across page reloads (stored in localStorage).

**Implementation:**

Uses JavaScript DOM events (dragstart, dragover, drop) and updates task array in real-time.

**Reminders:**

**Description:**

Alerts the user when a task’s due date and time is reached.

**How it Works:**

* Periodically checks task due times using setInterval.
* If current date/time matches task date/time, shows a **custom alert** (div) and plays a notification sound.
* Tasks already reminded or completed are ignored.

**Benefits:**

* Keeps users on track with important tasks.
* Enhances interactivity and usability of the app.
* Works even if the user is multitasking in the browser.

**Implementation:**

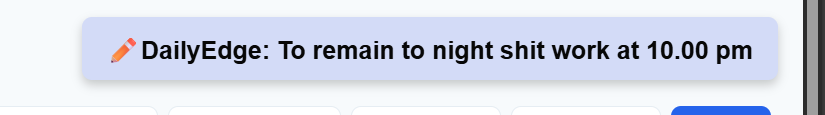
* Uses Date object to compare current time with task due time.
* custom-alert CSS animations provide **slide-in and fade-out effects**.
* Audio playback ensures user notices the reminder.

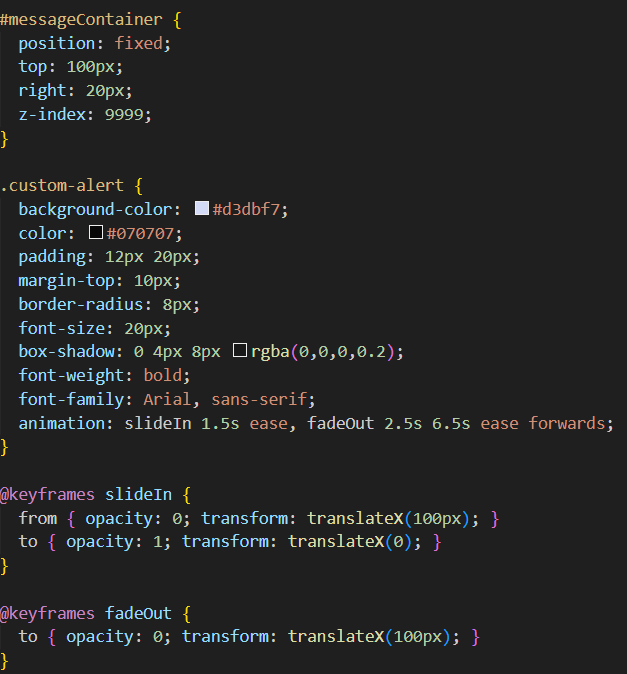
**UI/UX Improvement**

**Custom Alerts & Reminders:**

* Smooth **slide-in/fade-out** animations for reminder notifications.
* Sound alerts for due tasks, ensuring users don’t miss important deadlines.

**UI**

****

**CSS**

**Audio**



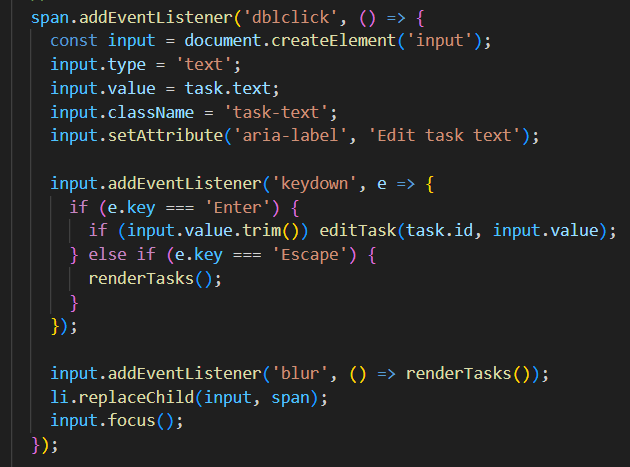
**Inline Editing:**

* + Faster task updates without navigating to a separate edit page.
  + Improves user experience and efficiency.

**UI**

****

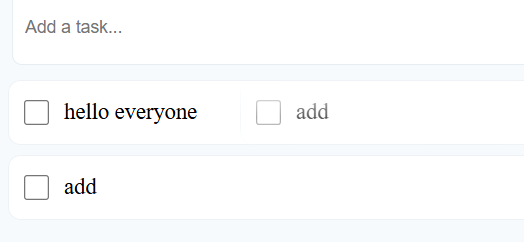
**JS**

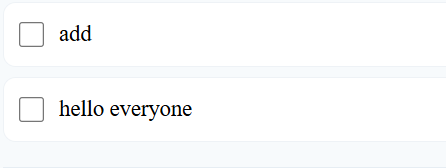
****

**Drag & Drop:**

* Users can prioritize tasks visually.
* Provides intuitive task management.
* Maintains order across page reloads (stored in localStorage).

**UI**

****

****

**JS**

****

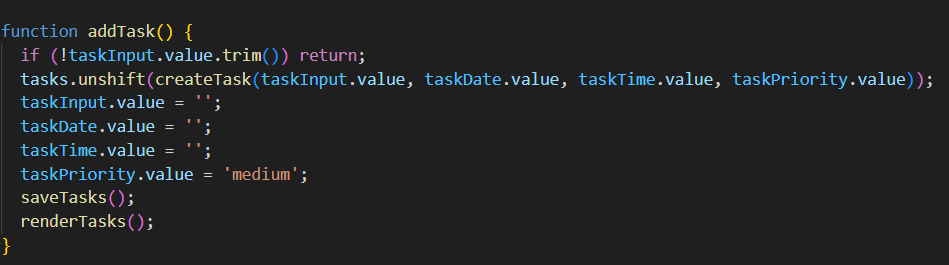
**API Enhancements**

**(Simulated locally via JavaScript)**

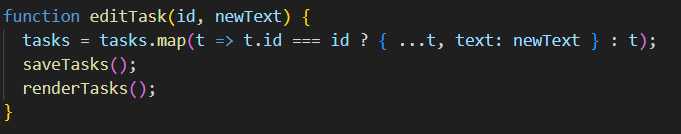
Tasks stored in **localStorage** for persistence across browser reloads.

* **Functions optimized for CRUD operations:**
  + addTask()
  + editTask()
  + deleteTask()
  + toggleTaskComplete()
  + clearCompleted().
* Real-time search and filter with minimal DOM updates.

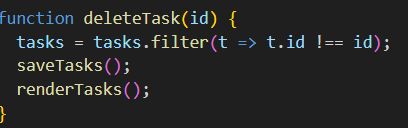
**addtask():**

****

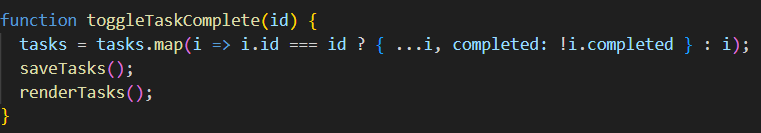
**edittask():**

****

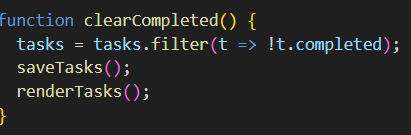
**deletetask():**

****

**toggleTaskComplete():**

****

**clearCompleted():**

****

**Performance & Security Checks**

**Performance Enhancements**

1. **Optimized Rendering:**
   * The application minimizes DOM updates by re-rendering only changed elements instead of refreshing the entire task list.
   * This approach significantly improves UI responsiveness and reduces browser workload.
2. **Efficient Task Management:**
   * Tasks are stored and accessed from localStorage, ensuring faster load times and smooth transitions between actions.
   * Filtering (All, Active, Completed) and searching tasks are handled efficiently using in-memory arrays, avoiding unnecessary loops or reloads.
3. **Lazy Updates:**
   * Only modified tasks (added, deleted, edited, or completed) trigger UI updates.
   * Reduces unnecessary reflows and repaints in the browser, leading to better runtime performance.
4. **Lightweight CSS and JS:**
   * The design uses clean, modular CSS and minimal JavaScript dependencies, ensuring quick load time and optimal performance on low-end devices as well.

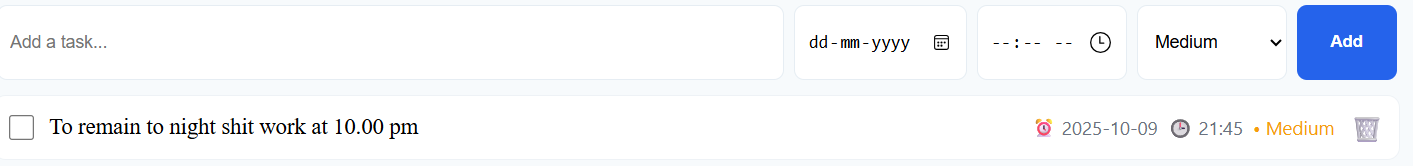
**Security Measures**

1. **Input Validation:**
   * Users cannot add empty tasks or invalid date/time inputs.
   * Ensures data integrity within the application and prevents accidental blank entries.
2. **Data Sanitization:**
   * User-entered text is properly escaped before being displayed to prevent Cross-Site Scripting (XSS) or script injection vulnerabilities.
3. **Local Data Protection:**
   * Task data stored in localStorage is structured and managed securely, preventing corruption or misuse.
   * Each task is assigned a unique ID, ensuring no accidental overwriting.
4. **Safe Editing Mechanism:**
   * Inline editing uses controlled input fields with event listeners restricted to known actions (Enter, Escape, or blur).

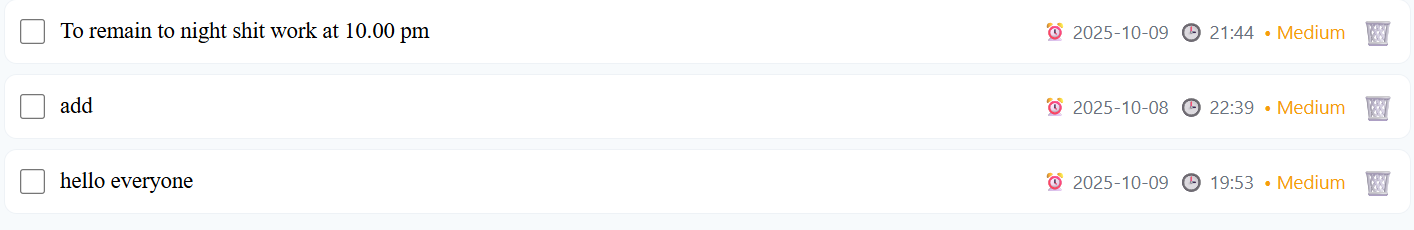
**Testing of Enhancements**

**Manual Testing:**

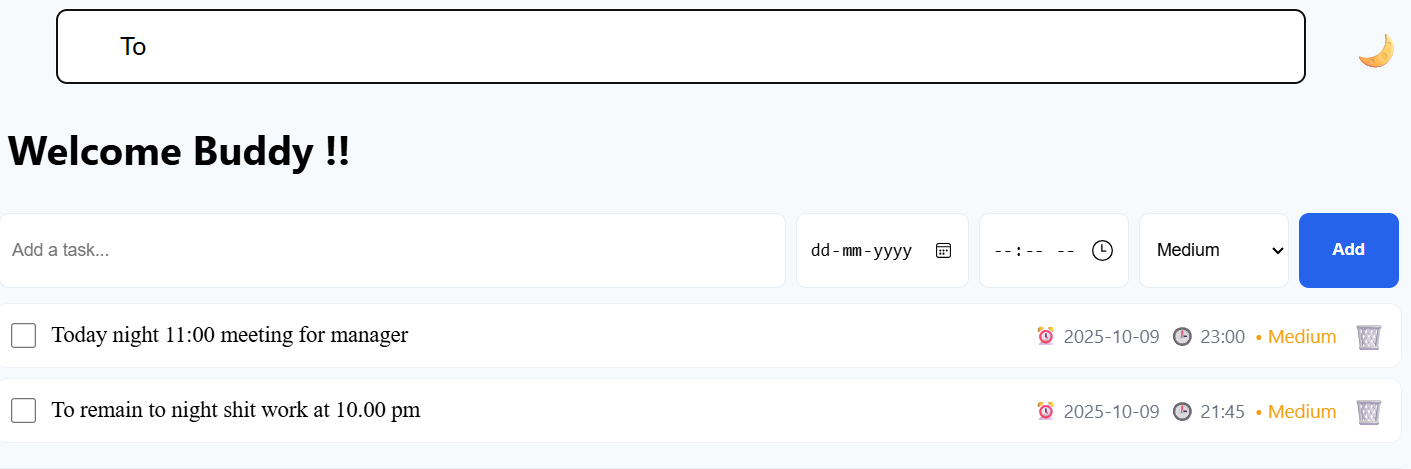
* **Adding, editing, deleting tasks works.**

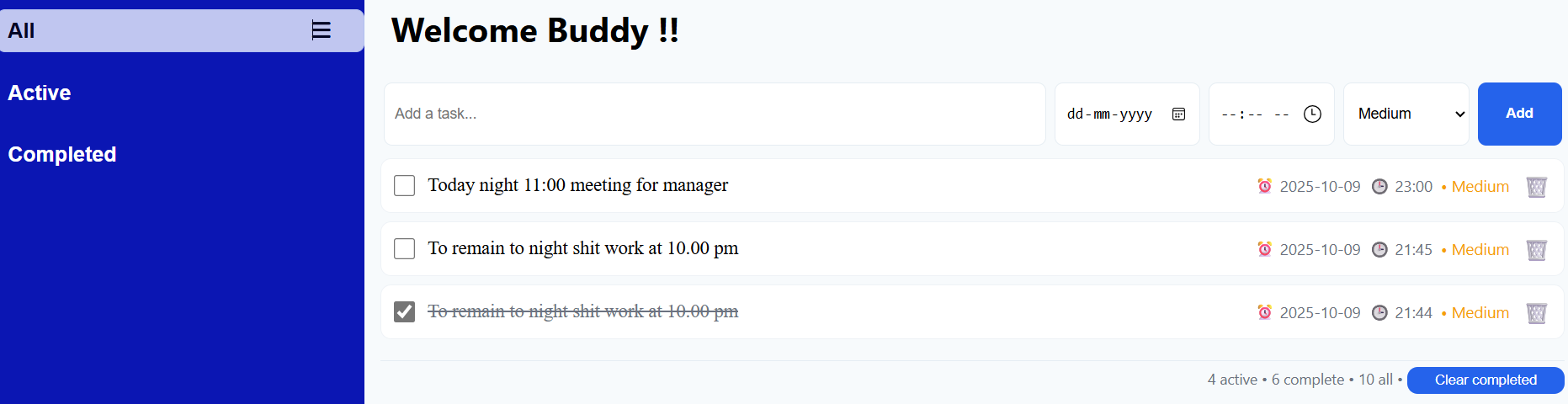
****

* **Tasks persist across page reloads.**

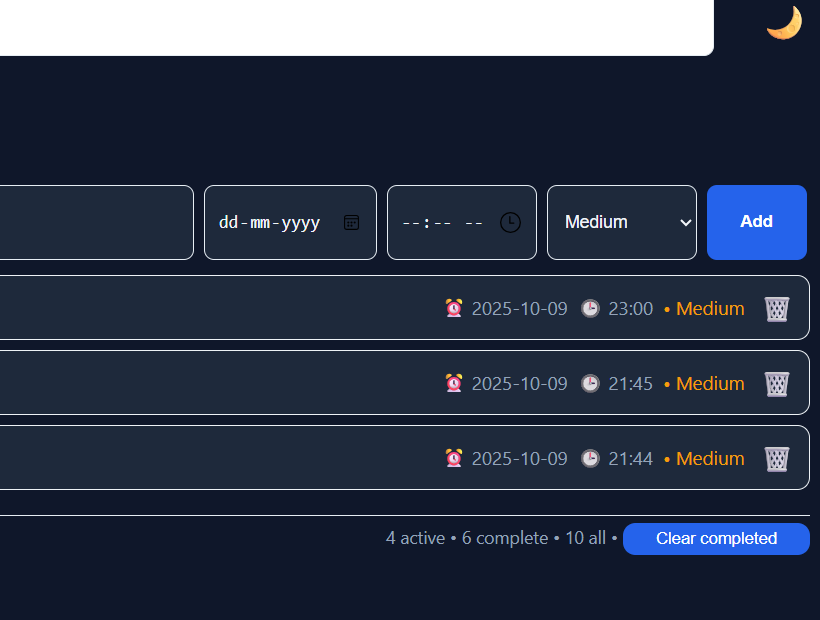
****

* **Filter and search work correctly.**

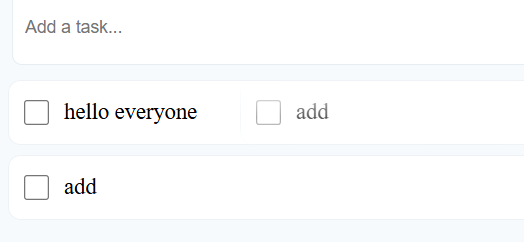
****

****

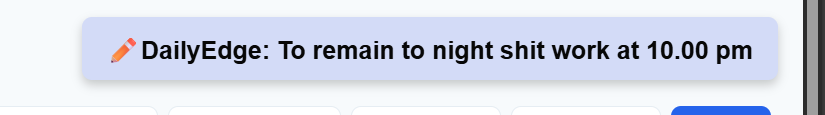
* **Dark mode toggle functions as expected.**

****

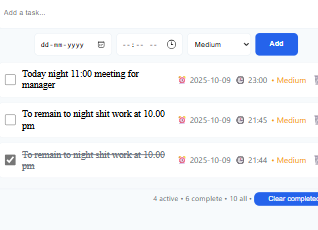
* **Drag & drop reorders tasks accurately.**

****

* **Reminder alerts trigger on correct date/time.**



* **Responsive Testing: Checked on desktop, tablet, and mobile screens.**

****