



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:

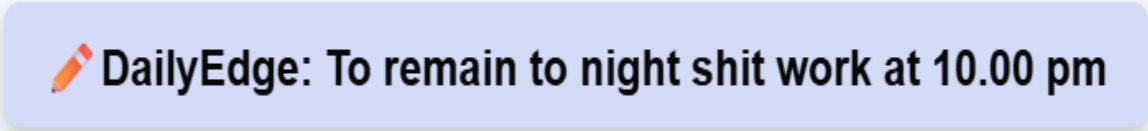
- Uses `Date` object to compare current time with task due time.
- custom-alert CSS animations provide **slide-in and fade-out effects**.
- Audio playback ensures user notices the reminder.


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



 **DailyEdge: To remain to night shit work at 10.00 pm**

CSS

```
#messageContainer {
  position: fixed;
  top: 100px;
  right: 20px;
  z-index: 9999;
}

.custom-alert {
  background-color: #d3dbf7;
  color: #070707;
  padding: 12px 20px;
  margin-top: 10px;
  border-radius: 8px;
  font-size: 20px;
  box-shadow: 0 4px 8px rgba(0,0,0,0.2);
  font-weight: bold;
  font-family: Arial, sans-serif;
  animation: slideIn 1.5s ease, fadeOut 2.5s 6.5s ease forwards;
}

@keyframes slideIn {
  from { opacity: 0; transform: translateX(100px); }
  to { opacity: 1; transform: translateX(0); }
}

@keyframes fadeOut {
  to { opacity: 0; transform: translateX(100px); }
}
```

Audio

```
<audio id="reminderSound" src="boss-sms-tone-4167-51949-0-67944.mp3" preload="auto"></audio>
```

Inline Editing:

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

UI



JS

```
span.addEventListener('dblclick', () => {
  const input = document.createElement('input');
  input.type = 'text';
  input.value = task.text;
  input.className = 'task-text';
  input.setAttribute('aria-label', 'Edit task text');

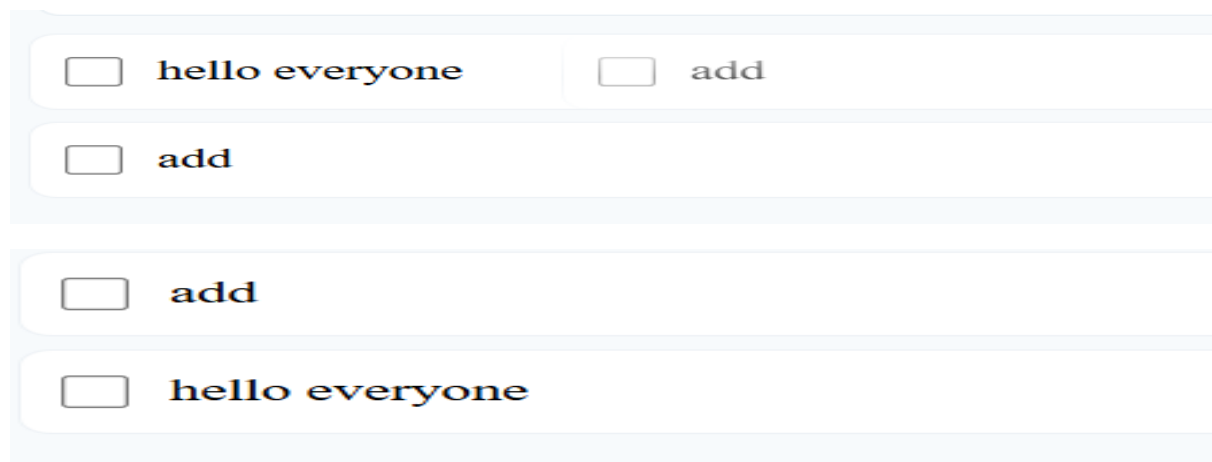
  input.addEventListener('keydown', e => {
    if (e.key === 'Enter') {
      if (input.value.trim()) editTask(task.id, input.value);
    } else if (e.key === 'Escape') {
      renderTasks();
    }
  });

  input.addEventListener('blur', () => renderTasks());
  li.replaceChild(input, span);
  input.focus();
});
```

Drag & Drop:

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

UI



JS

```
let dragId = null;

taskListEl.addEventListener('dragstart', e => {
  dragId = e.target.dataset.id;
});

taskListEl.addEventListener('dragover', e => {
  e.preventDefault();
});

taskListEl.addEventListener('drop', e => {
  e.preventDefault();
  const targetId = e.target.closest('li')?.dataset.id;

  if (dragId && targetId && dragId !== targetId) {
    const from = tasks.findIndex(t => t.id === dragId);
    const to = tasks.findIndex(t => t.id === targetId);
    const moved = tasks.splice(from, 1)[0];
    tasks.splice(to, 0, moved);
    saveTasks();
    renderTasks();
  }
});
```

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():

```
function addTask() {  
  if (!taskInput.value.trim()) return;  
  tasks.unshift(createTask(taskInput.value, taskDate.value, taskTime.value, taskPriority.value));  
  taskInput.value = '';  
  taskDate.value = '';  
  taskTime.value = '';  
  taskPriority.value = 'medium';  
  saveTasks();  
  renderTasks();  
}
```


edittask():

```
function editTask(id, newText) {  
  tasks = tasks.map(t => t.id === id ? { ...t, text: newText } : t);  
  saveTasks();  
  renderTasks();  
}
```

deletetask():

```
function deleteTask(id) {  
  tasks = tasks.filter(t => t.id !== id);  
  saveTasks();  
  renderTasks();  
}
```

toggleTaskComplete():

```
function toggleTaskComplete(id) {  
  tasks = tasks.map(i => i.id === id ? { ...i, completed: !i.completed } : i);  
  saveTasks();  
  renderTasks();  
}
```

clearCompleted():

```
function clearCompleted() {  
  tasks = tasks.filter(t => !t.completed);  
  saveTasks();  
  renderTasks();  
}
```

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.

dd-mm-yyyy

--:-- --

Medium

Add

☐ To remain to night shit work at 10.00 pm

🔔 2025-10-09 ⌚ 21:45 • Medium 🗑️

✓ Tasks persist across page reloads.

☐ To remain to night shit work at 10.00 pm

🔔 2025-10-09 ⌚ 21:44 • Medium 🗑️

☐ add

🔔 2025-10-08 ⌚ 22:39 • Medium 🗑️

☐ hello everyone

🔔 2025-10-09 ⌚ 19:53 • Medium 🗑️

✓ Filter and search work correctly.

🌙

Welcome Buddy !!

dd-mm-yyyy

--:-- --

Medium

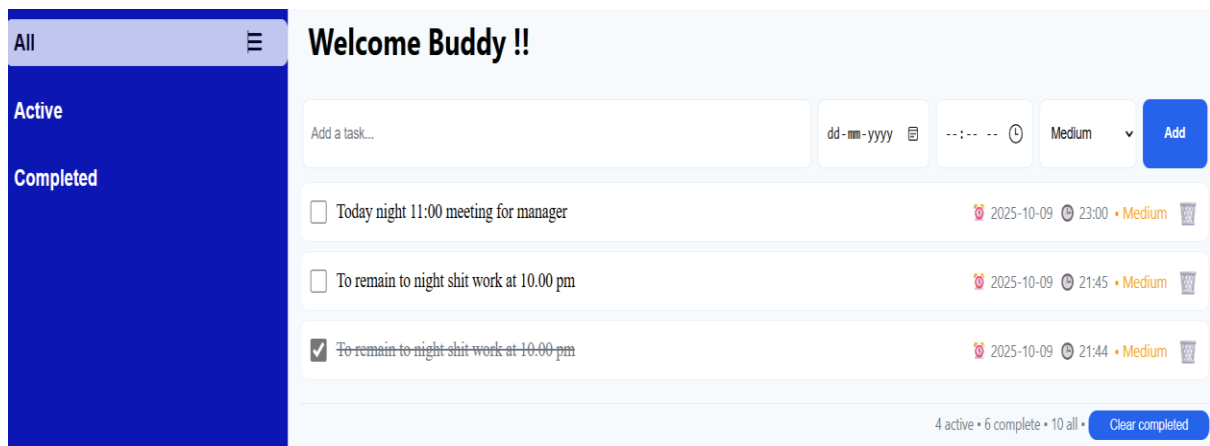
Add

☐ Today night 11:00 meeting for manager

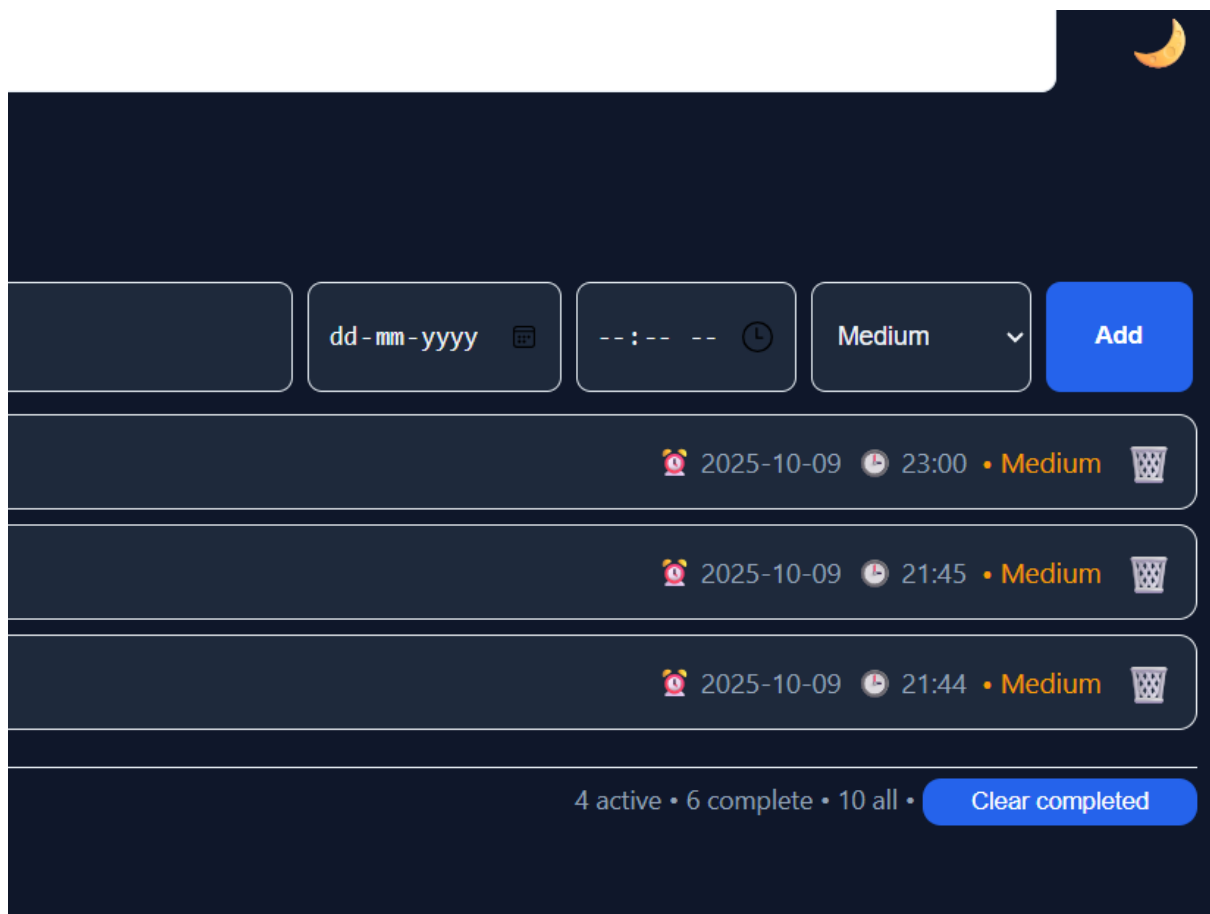
🔔 2025-10-09 ⌚ 23:00 • Medium 🗑️

☐ To remain to night shit work at 10.00 pm

🔔 2025-10-09 ⌚ 21:45 • Medium 🗑️



✓ Dark mode toggle functions as expected.



✓ Drag & drop reorders tasks accurately.

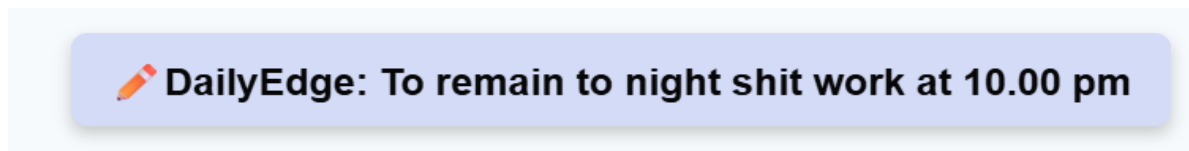
Add a task...

☐ hello everyone

☐ add

☐ add

✓ Reminder alerts trigger on correct date/time.



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

Add a task...

dd-mm-yyyy

--:--:--

Medium

Add

☐

Today night 11:00 meeting for manager

2025-10-09

23:00

• Medium

☐

To remain to night shit work at 10.00 pm

2025-10-09

21:45

• Medium

☒

To remain to night shit work at 10.00 pm

2025-10-09

21:44

• Medium

4 active • 6 complete • 10 all •

Clear completed