Front-End UI/UX Mini Project -1

TO-DO LIST APPLICATION

Submitted by

Diya Susan Binu (2463019)

[email i_d : diya.susan@btech.christuniversity.in]

Agnes Mariya Sara (2463004)

[email i_d :agnes.mariya@btech.christuniversity.in

Shivani kp (2463054)

[email i_d :shivani.kp@btech.christuniversity.in]

Course: , *UI/UX Design Fundamentals*

Instructor Name: *Dhiraj*

1nstitution: *Christ University*

2. Abstract

The To-Do List Application is a simple and efficient web-based tool designed to help users manage their daily tasks effectively. It allows users to add, edit, delete, and organize tasks according to their needs. Tasks can be marked as completed with a single click, providing an easy way to track progress. The application also includes filtering options, enabling users to view only completed or active tasks. Developed using HTML, CSS, JavaScript, Bootstrap, and jQuery, the project ensures a responsive and interactive user experience across various devices. This application aims to improve productivity and task organization through an intuitive and user-friendly interface.

3. Objectives

1. To develop a user-friendly web application for managing daily tasks efficiently.
2. To allow users to add, edit, delete, and organize tasks based on their requirements.
3. To enable marking tasks as completed and visually track progress.
4. To provide filtering options for viewing active, completed, or all tasks.
5. To ensure a responsive and interactive user interface that works well on different devices.
6. To implement dynamic functionality using JavaScript and jQuery for a seamless user experience.

4. Scope of the Project

1. To develop a web-based application that helps users manage daily tasks efficiently.
2. To provide features for adding, editing, deleting, and organizing tasks.
3. To enable users to mark tasks as completed and track their progress easily.
4. To offer filtering options to view active, completed, or all tasks.
5. To design a responsive and user-friendly interface accessible on different devices.
6. To make the application useful for students, professionals, and individuals for managing personal or work-related tasks.
7. To allow future improvements such as task categorization, reminders, and user authentication.

5. Tools & Technologies Used

Tool/Technology	Purpose
HTML5	Markup and content structure
CSS3	Styling and layout management
VS Code	Code editor
Chrome DevTools	Testing and debugging

6. HTML Structure Overview

- Used semantic tags: <header>, <nav>, <main>, <section>, <footer>
- Structured into reusable sections: About, Projects, Contact
- Navigation menu using <ul> and anchor links for smooth scrolling

7. CSS Styling Strategy

- Used external CSS file (style.css)
- Organized with comments and sections
- Techniques used:
  - Flexbox and Grid for layout
  - Media Queries for responsiveness
  - CSS Variables for theme customization
  - Hover effects and transitions
  - Mobile-first design approach

8. Key Features

Feature	Description
Task Management	Add, edit, and delete tasks to keep track of daily activities.
Mark as Completed	Mark tasks as done using a checkbox to monitor progress.
Filter Tasks	View tasks based on their status – All, Active, or Completed.
Dynamic Interactivity	Use JavaScript for real-time updates without refreshing the page.
Responsive Design	Works seamlessly across devices with different screen sizes.
User-Friendly Interface	Clean and intuitive UI for easy navigation and task tracking..

9. Challenges Faced & Solutions

Challenge	Solution
Difficulty in dynamically updating tasks	Used JavaScript event listeners and DOM manipulation for real-time updates.
Issues with filtering completed and active tasks	Implemented conditional rendering and filter functions in JavaScript.
Overlapping layout on smaller screens	Used responsive design techniques with Bootstrap and media queries.
Task data not persisting after refresh	Utilized browser localStorage to save and retrieve tasks.
Checkbox status not updating correctly	Added event listeners to sync task state with the DOM dynamically.

10. Outcome

- Successfully developed a fully functional web-based To-Do List application.
- All core features — adding, editing, deleting, marking as completed, and filtering tasks — work as intended.
- Implemented a dynamic and responsive user interface that adapts to different screen sizes.
- Enhanced understanding of DOM manipulation, event handling, and local storage using JavaScript.

- Gained practical experience in front-end development with HTML, CSS, Bootstrap, and jQuery.
- Created a user-friendly tool that helps users manage and organize their daily tasks efficiently.

11. Future Enhancements

- Add a deadline and reminder feature to notify users about upcoming tasks.
- Implement task categorization (e.g., work, personal, urgent) for better organization.
- Introduce a search bar to quickly find specific tasks.
- Enable user authentication and login to store tasks for individual users.
- Integrate cloud storage or database to save tasks online instead of local storage.
- Provide drag-and-drop functionality to reorder tasks based on priority.
- Add a dark mode/light mode toggle to enhance user experience**.**

12. Sample Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Task N Go</title>

  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;700&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Roboto+Mono:wght@400;500;700&display=swap" rel="stylesheet">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
```

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script src="https://code.jquery.com/ui/1.13.2/jquery-ui.min.js"></script>

<style>
    /* Dark Theme Color Palette:
        Background: #2A2F36
        Primary Text: #EAEAEA (Ice White)
        Accent: #00FFFF (Electric Blue)
        Highlight: #9D00FF (Neon Purple)
        Completed Task: #708090 (Slate Gray)
    */

    body {
        min-height: 100vh;
        background-color: #2A2F36;
        font-family: 'Poppins', sans-serif;
        color: #EAEAEA;
    }

    .header-bar {
        background-color: #1a1e24; /* A slightly darker shade for the header */
        padding: 10px 20px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
        margin-bottom: 20px;
        display: flex;
        align-items: center;
        justify-content: space-between;
```

```
}

.header-left {
   display: flex;
   align-items: center;
   gap: 15px;
}

.header-left h1 {
   color: #EAEAEA;
   font-weight: 700;
   letter-spacing: 1px;
   margin: 0;
   font-family: 'Poppins', sans-serif;
   font-size: 1.5rem;
}

.logo-icon {
   width: 35px;
   height: 35px;
   fill: #9D00FF; /* Using Neon Purple for the logo */
}

.header-nav .nav-pills .nav-link {
   border-radius: 50px;
   transition: all 0.3s ease;
   font-weight: 500;
```

```
    color: #00FFFF; /* Electric Blue for links */
}

.header-nav .nav-pills .nav-link:hover {
    transform: translateY(-3px);
    background-color: #00FFFF;
    color: #2A2F36;
}

.header-nav .nav-pills .nav-link.active {
    background: #9D00FF;
    color: white;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.3);
}

.card {
    background-color: #333842;
    border: 1px solid #454d58;
    border-radius: 16px;
    box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);
    margin-bottom: 20px;
    color: #EAEAEA;
}

.card-header-title {
    color: #9D00FF;
}
```

```css
/* Dashboard Preview Cards */
.preview-card {
   background-color: #3f4651;
   min-height: 150px;
   cursor: pointer;
   transition: transform 0.2s ease, box-shadow 0.2s ease;
   display: flex;
   flex-direction: column;
   justify-content: center;
   align-items: center;
   text-align: center;
}
.preview-card:hover {
   transform: translateY(-5px);
   box-shadow: 0 8px 16px rgba(0, 0, 0, 0.3);
}
.preview-card h5 {
   font-weight: 600;
   color: #00FFFF;
}
.preview-card .card-body {
   padding: 1rem;
}
.preview-card-content {
   font-size: 0.9em;
   color: #EAEAEA;
}
.preview-card .bi {
   font-size: 1.5rem;
   margin-bottom: 5px;
   color: #00FFFF;
}

/* Pawxel styles */
#pawxelToggle {
   position: fixed;
   bottom: 20px;
   right: 20px;
   width: 65px;
   height: 65px;
   border-radius: 50%;
   background: #9D00FF;
   display: flex;
   align-items: center;
   justify-content: center;
   box-shadow: 0 6px 18px rgba(0, 0, 0, 0.4);
   cursor: pointer;
   z-index: 1050;
   transition: transform 0.2s ease;
   font-size: 2rem;
   line-height: 1;
}
```

```
#pawxelToggle:hover {
    transform: scale(1.1);
}

#pawxelChat {
    position: fixed;
    bottom: 100px;
    right: 20px;
    width: 320px;
    max-height: 420px;
    background: #333842;
    border-radius: 20px;
    box-shadow: 0 8px 24px rgba(0, 0, 0, 0.4);
    display: none;
    flex-direction: column;
    overflow: hidden;
    z-index: 1100;
    color: #EAEAEA;
}

#pawxelHeader {
    background: #1a1e24;
    padding: 10px;
    display: flex;
    align-items: center;
    gap: 10px;
    color: #EAEAEA;
```

```
}

#pawxelHeader h6 {
   margin: 0;
   font-weight: 600;
}

.cat-avatar {
   width: 40px;
   height: 40px;
}

#pawxelWindow {
   flex-grow: 1;
   padding: 12px;
   overflow-y: auto;
   background: #2a2f36;
   display: flex;
   flex-direction: column;
}

.chat-bubble {
   max-width: 75%;
   padding: 10px 14px;
   border-radius: 18px;
   margin: 6px 0;
   position: relative;
```

```
}

.chat-user {
   background: #00FFFF;
   align-self: flex-end;
   border-bottom-right-radius: 2px;
   color: #2A2F36;
}

.chat-user::after {
   content: "";
   position: absolute;
   right: -8px;
   bottom: 0;
   border-width: 8px 0 0 8px;
   border-style: solid;
   border-color: #00FFFF transparent;
}

.chat-bot {
   background: #9D00FF;
   color: white;
   align-self: flex-start;
   border-bottom-left-radius: 2px;
}

.chat-bot::after {
   content: "";
   position: absolute;
   left: -8px;
   bottom: 0;
   border-width: 8px 8px 0 0;
   border-style: solid;
   border-color: #9D00FF transparent;
}

#pawxelForm {
   display: flex;
   padding: 8px;
   gap: 6px;
   border-top: 1px solid #454d58;
}

#pawxelForm input {
   flex-grow: 1;
   background-color: #2a2f36;
   border: 1px solid #454d58;
   color: #EAEAEA;
}

#pawxelForm input::placeholder {
   color: #708090;
}
```

```
#pawxelForm button {
    background: #00FFFF;
    color: #2A2F36;
    border: none;
}

/* Task list styles */
.task-input-group {
    display: flex;
    flex-wrap: wrap;
    gap: 10px;
    margin-bottom: 20px;
}

#taskInput, #taskDueDatetime {
    background-color: #2a2f36;
    border-radius: 50px;
    padding: 10px 20px;
    border: 1px solid #454d58;
    flex-grow: 1;
    color: #EAEAEA;
}

#taskInput::placeholder, #taskDueDatetime::placeholder {
    color: #708090;
}
```

```
#addTaskBtn {
    border-radius: 50px;
    background: #9D00FF;
    border: none;
    color: white;
    transition: all 0.2s ease;
}
#addTaskBtn:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}

.filters button {
    border-radius: 50px;
    margin-right: 5px;
    font-weight: 500;
    background-color: #3f4651;
    border: 1px solid #454d58;
    color: #EAEAEA;
}

.filters button.active {
    background-color: #00FFFF;
    color: #2A2F36;
    border-color: #00FFFF;
}
```

```
.task-list-item {
   display: flex;
   align-items: center;
   justify-content: space-between;
   background-color: #3f4651;
   padding: 15px;
   border-radius: 12px;
   margin-bottom: 10px;
   border-left: 5px solid #00FFFF;
   transition: all 0.3s ease;
   cursor: move; /* Drag cursor */
}

.task-list-item:hover {
   box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

.task-list-item.completed {
   background-color: #708090;
   border-left: 5px solid #9D00FF;
   color: #2A2F36;
   text-decoration: line-through;
}

.task-list-item.completed .task-due-date {
   color: #333;
}
```

```
.task-content {
   flex-grow: 1;
   font-weight: 500;
   display: flex;
   flex-direction: column;
   color: #EAEAEA;
}

.task-list-item.completed .task-content {
   color: #333;
}

.task-content .task-text {
   font-size: 1.1em;
   font-weight: 500;
}

.task-content .task-due-date {
   font-size: 0.8rem;
   color: #B0B0B0;
   margin-top: 4px;
}

.task-actions button {
   border: none;
   background: none;
```

```
    cursor: pointer;
    font-size: 1.2rem;
    color: #B0B0B0;
    transition: color 0.2s ease;
    margin-left: 10px;
}

.task-actions button:hover {
    color: #EAEAEA;
}

.task-actions .edit-btn:hover {
    color: #9D00FF;
}

.task-actions .delete-btn:hover {
    color: #00FFFF;
}

/* Calendar page specific styles */
.calendar-nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 15px;
    font-family: 'Poppins', sans-serif;
    font-weight: 600;
```

```
}
.calendar-nav button {
   background: none;
   border: none;
   font-size: 1.5rem;
   color: #00FFFF;
   cursor: pointer;
   transition: transform 0.2s ease;
}
.calendar-nav button:hover {
   transform: scale(1.1);
}
.calendar-grid {
   display: grid;
   grid-template-columns: repeat(7, 1fr);
   gap: 10px;
   text-align: center;
}
.calendar-day-header {
   font-weight: bold;
   color: #9D00FF;
}
.calendar-day-box {
   background-color: #3f4651;
   padding: 10px 5px;
   border-radius: 8px;
   min-height: 80px;
```

```
    display: flex;
    flex-direction: column;
    border: 1px solid #454d58;
    color: #EAEAEA;
}
.calendar-day-box.has-tasks {
    background-color: #00FFFF;
    color: #2A2F36;
    border-color: #00FFFF;
    position: relative;
    overflow: hidden;
}
.calendar-day-box.has-tasks::after {
    content: "•";
    position: absolute;
    bottom: 5px;
    right: 5px;
    font-size: 2em;
    color: #2A2F36;
    line-height: 1;
}
.calendar-day-box .day-number {
    font-weight: bold;
}
.calendar-day-box.empty-day {
    background-color: #2a2f36;
    color: #555;
    border: 1px solid #3f4651;
}


/* Progress page specific styles */
.progress-container {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    padding: 20px;
}
.progress-circle {
    width: 150px;
    height: 150px;
    border-radius: 50%;
    background: conic-gradient(#00FFFF var(--percentage), #3f4651 var(--percentage));
    display: flex;
    align-items: center;
    justify-content: center;
    margin-bottom: 20px;
}
.progress-inner-circle {
    width: 120px;
    height: 120px;
    background-color: #333842;
    border-radius: 50%;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
}
.progress-percentage {
    font-size: 2.5rem;
    font-weight: bold;
    color: #00FFFF;
    font-family: 'Roboto Mono', monospace;
}
.progress-label {
    font-size: 0.9rem;
    color: #B0B0B0;
}
.progress-stats {
    text-align: center;
}
.progress-stats h4 {
    font-weight: 500;
}
.progress-stats p {
    color: #EAEAEA;
}

/* Weekly Report page specific styles */
.report-list .list-group-item {
    display: flex;
```

```
      justify-content: space-between;
      align-items: center;
      padding: 15px;
      border: none;
      border-bottom: 1px solid #454d58;
      transition: background-color 0.2s ease;
      background-color: #333842;
      color: #EAEAEA;
    }
    .report-list .list-group-item:first-child {
      border-top: 1px solid #454d58;
    }
    .report-list .list-group-item:hover {
      background-color: #3f4651;
    }
    .report-title {
      font-weight: 500;
    }
    .report-date {
      font-size: 0.9em;
      color: #B0B0B0;
    }

  </style>
</head>
<body>
```

```
    <div class="header-bar">
        <div class="header-left">
            <svg class="logo-icon" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 576 512">
                <path d="M304 32c0 26.5-21.5 48-48 48s-48-21.5-48-48s21.5-48 48-48s48 21.5 48 48zM48 224c0-17.7 14.3-32 32-32H384c17.7 0 32 14.3 32 32V448h64V224c0-17.7 14.3-32 32-32s32 14.3 32 32V448c0 17.7-14.3 32-32 32H384c-17.7 0-32-14.3-32-32V256H80c-17.7 0-32 14.3-32 32s14.3 32 32 32H496v32H80c-17.7 0-32 14.3-32 32s14.3 32 32 32H496v32H80c-17.7 0-32 14.3-32 32s14.3 32 32 32H496c17.7 0 32-14.3 32-32V256c0-53-43-96-96-96H80c-53 0-96 43-96 96z"/>
            </svg>
            <h1>Task N Go</h1>
        </div>
        <div class="header-nav">
            <ul class="nav nav-pills">
                <li class="nav-item"><a class="nav-link active nav-pill" data-target="home">Tasks</a></li>
                <li class="nav-item"><a class="nav-link nav-pill" data-target="calendar">Calendar</a></li>
                <li class="nav-item"><a class="nav-link nav-pill" data-target="progress">Progress</a></li>
                <li class="nav-item"><a class="nav-link nav-pill" data-target="report">Weekly Report</a></li>
            </ul>
        </div>
    </div>

    <div class="container pb-5">
        <div id="home" class="page active">
            <div class="card p-4">
                <h4 class="mb-3 card-header-title">My Task List</h4>
                <form id="taskForm" class="task-input-group">
                  <input type="text" id="taskInput" class="form-control" placeholder="Add a new task..." required>
                  <input type="datetime-local" id="taskDueDatetime" class="form-control" placeholder="Due Date (Optional)">
                  <button type="submit" id="addTaskBtn" class="btn">Add</button>
                </form>

                <div class="d-flex mb-3 filters">
                  <button class="btn active" data-filter="all">All</button>
                  <button class="btn" data-filter="active">Active</button>
                  <button class="btn" data-filter="completed">Completed</button>
                </div>

                <div id="taskList" class="list-group">
                  <p id="noTasksMessage" class="text-center text-muted mt-3" style="display: none;">No tasks here. Add some!</p>
                </div>
            </div>

            <div class="row mt-4">
                <div class="col-md-4">
                  <div class="card preview-card" data-target="calendar">
                    <div class="card-body">
                      <h5>Calendar</h5>
                      <div class="preview-card-content" id="calendarPreview">
                        </div>
                    </div>
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="card preview-card" data-target="progress">
                        <div class="card-body">
                            <h5>Progress</h5>
                            <div class="preview-card-content" id="progressPreview">
                                </div>
                        </div>
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="card preview-card" data-target="report">
                        <div class="card-body">
                            <h5>Weekly Report</h5>
                            <div class="preview-card-content" id="reportPreview">
                                </div>
                        </div>
                    </div>
                </div>
            </div>
</div>

<div id="calendar" class="page" style="display: none;">
    <div class="card p-3">
        <h4 class="mb-3 text-center card-header-title">Calendar View</h4>
        <div id="calendarContent">
```

```
            </div>
        </div>
    </div>

    <div id="progress" class="page" style="display: none;">
        <div class="card p-3">
            <h4 class="mb-3 text-center card-header-title">Your Progress</h4>
            <div id="progressContent" class="progress-container">
                </div>
        </div>
    </div>

    <div id="report" class="page" style="display: none;">
        <div class="card p-3">
            <h4 class="mb-3 text-center card-header-title">Weekly Report</h4>
            <div id="reportContent">
                </div>
        </div>
    </div>
</div>

<div id="pawxelToggle" title="Chat with Pawxel">
    <span role="img" aria-label="Cat emoji">🐱</span>
</div>

<div id="pawxelChat">
    <div id="pawxelHeader">
```

```
        <img src="images/cat.jpg" class="cat-avatar" alt="cat">
        <h6>Pawxel</h6>
    </div>
    <div id="pawxelWindow"></div>
    <form id="pawxelForm">
        <input id="pawxelInput" class="form-control" placeholder="Ask Pawxel...">
        <button class="btn">Send</button>
    </form>
</div>

<script>
    // Store tasks in a global array
    let tasks = [];
    const STORAGE_KEY = 'pawxel_tasks';
    let currentMonthView = new Date(); // Tracks the month for the calendar view

    // --- Page Navigation Logic ---
    $(document).ready(function() {
        // Initial setup
        $(".page").hide();
        $("#home").show(); // Show the main 'Tasks' page by default
    });

    $(".nav-pill, .preview-card").on("click", function() {
        const target = $(this).data("target");

        // Handle navigation pills
        $(".nav-pill").removeClass("active");
        $(`.nav-pill[data-target="${target}"]`).addClass("active");

        // Handle page display
        $(".page").hide();
        const targetPage = $("#" + target);
        targetPage.show();

        // Auto-update the content of the target page
        if (target === "calendar") {
            renderCalendar(currentMonthView);
        } else if (target === "progress") {
            renderProgress();
        } else if (target === "report") {
            renderReport();
        }
    });

    // --- Pawxel Chat Logic ---
    $("#pawxelToggle").on("click", () => {
        $("#pawxelChat").fadeToggle();
        // Scroll to bottom when opening
        if ($("#pawxelChat").is(":visible")) {
            $("#pawxelWindow").scrollTop($("#pawxelWindow")[0].scrollHeight);
        }
    });
```

```
function pawxelReply(text) {
    const $win = $("#pawxelWindow");
    const addBot = (m) => {
        $win.append(`<div class="chat-bubble chat-bot">${m}</div>`);
        $win.scrollTop($win[0].scrollHeight);
    };

    const lowerCaseText = text.toLowerCase();
    let replyMessage;

    if (lowerCaseText.includes("hello") || lowerCaseText.includes("hi")) {
        replyMessage = "Hi there! I'm Pawxel 🐾. How can I help with your tasks today?";
    } else if (lowerCaseText.includes("add") && lowerCaseText.includes("task")) {
        const taskText = text.replace(/add\s*a*\s*new*\s*task\s*about*/i, '').replace(/please/i, '').trim();
        if (taskText) {
            addTask(taskText);
            replyMessage = `Got it! I've added "**${taskText}**" to your list. Purr-fect!`;
        } else {
            replyMessage = "I can't add an empty task. What do you want to add?";
        }
    } else if (lowerCaseText.includes("tasks")) {
        if (tasks.length > 0) {
            const taskListText = tasks.slice(0, 3).map(t => `- **${t.text}** (${t.isCompleted ? 'Completed' : 'Active'})`).join('<br>');
            const moreText = tasks.length > 3 ? `<br>... and ${tasks.length - 3} more.` : '';
            replyMessage = `Here are your top tasks:<br>${taskListText}${moreText}`;
        } else {
          replyMessage = "You don't have any tasks right now. Time to relax! 😌";
        }
      } else if (lowerCaseText.includes("thank")) {
        replyMessage = "You're welcome! Happy to help. Meow. 🐾";
      } else if (lowerCaseText.includes("joke")) {
        replyMessage = "Why did the cat go to the library? Because it wanted a good 'tail'! 📖😹";
      } else {
        replyMessage = "Meow! I can help you with **adding tasks**, listing your **tasks**, or just say **hello**!";
      }
      addBot(replyMessage);
    }

    $("#pawxelForm").on("submit", function(e) {
      e.preventDefault();
      let txt = $("#pawxelInput").val().trim();
      if (!txt) return;
      $("#pawxelWindow").append(`<div class="chat-bubble chat-user">${txt}</div>`);
      $("#pawxelInput").val(");
      setTimeout(() => pawxelReply(txt), 500);
    });

    // --- Task List Logic ---
    $(document).ready(function() {
      loadTasks();
```

```
renderAllTabs();

// Make the task list sortable using jQuery UI
$("#taskList").sortable({
    handle: ".task-content", // Only drag by the text area
    update: function(event, ui) {
        const newOrder = $("#taskList").sortable("toArray", { attribute: "data-id" });
        reorderTasks(newOrder);
    }
});

// Handle adding a new task
$("#taskForm").on("submit", function(e) {
    e.preventDefault();
    const taskText = $("#taskInput").val().trim();
    const dueDatetime = $("#taskDueDatetime").val();

    if (taskText) {
        addTask(taskText, dueDatetime);
        $("#taskInput").val(");
        $("#taskDueDatetime").val("); // Clear the date input
    }
});

// Handle task actions (delete, toggle complete) using event delegation
$("#taskList").on("click", ".task-actions .delete-btn", function(e) {
    e.stopPropagation(); // Prevent the parent item toggle from firing
```

```
            const id = $(this).closest(".task-list-item").data("id");
            deleteTask(id);
        });

        // Toggle task completion on list item click (excluding the actions area)
        $("#taskList").on("click", ".task-list-item", function(e) {
            // Ignore clicks on action buttons or editable span
            if ($(e.target).closest(".task-actions").length === 0 &&
$(e.target).attr('contenteditable') !== 'true') {
                const id = $(this).data("id");
                toggleTaskCompletion(id);
            }
        });

        // Edit task text
        $("#taskList").on("click", ".task-actions .edit-btn", function(e) {
            e.stopPropagation(); // Prevent the parent item toggle from firing
            const $item = $(this).closest(".task-list-item");
            const $textSpan = $item.find(".task-text");
            const id = $item.data("id");

            if ($textSpan.attr("contenteditable") === "true") {
                $textSpan.attr("contenteditable", "false");
                $(this).html("✏️");
                updateTaskText(id, $textSpan.text());
            } else {
                $textSpan.attr("contenteditable", "true").focus();
```

```
    // Set cursor to end of text
    const range = document.createRange();
    const sel = window.getSelection();
    range.selectNodeContents($textSpan[0]);
    range.collapse(false);
    sel.removeAllRanges();
    sel.addRange(range);

    $(this).html("✓");
  }
});

// Handle filtering
$(".filters button").on("click", function() {
  $(".filters button").removeClass("active");
  $(this).addClass("active");
  renderTasks($(this).data("filter"));
});

// Calendar navigation buttons (rendered inside renderCalendar)
$("#calendarContent").on('click', '#prevMonthBtn', function() {
  currentMonthView.setMonth(currentMonthView.getMonth() - 1);
  renderCalendar(currentMonthView);
});
$("#calendarContent").on('click', '#nextMonthBtn', function() {
  currentMonthView.setMonth(currentMonthView.getMonth() + 1);
  renderCalendar(currentMonthView);
```

```
    });
});

function saveTasks() {
    localStorage.setItem(STORAGE_KEY, JSON.stringify(tasks));
}

function loadTasks() {
    const storedTasks = localStorage.getItem(STORAGE_KEY);
    if (storedTasks) {
        tasks = JSON.parse(storedTasks);
    }
}

function renderAllTabs() {
    renderTasks($(".filters button.active").data("filter"));
    // Only render calendar if it's the active tab to prevent unnecessary rendering
    if ($('#calendar').is(':visible')) {
        renderCalendar(currentMonthView);
    }
    if ($('#progress').is(':visible')) {
        renderProgress();
    }
    if ($('#report').is(':visible')) {
        renderReport();
    }
    updateDashboardPreviews();
```

```
}


function renderTasks(filter = 'all') {
    const $taskList = $("#taskList");
    $taskList.empty();


    let filteredTasks = tasks;
    if (filter === 'active') {
        filteredTasks = tasks.filter(task => !task.isCompleted);
    } else if (filter === 'completed') {
        filteredTasks = tasks.filter(task => task.isCompleted);
    }


    if (filteredTasks.length === 0) {
        $("#noTasksMessage").show();
    } else {
        $("#noTasksMessage").hide();
        filteredTasks.forEach(task => {
            // Format due date
            let dueDateText = '';
            if (task.dueDatetime) {
                const date = new Date(task.dueDatetime);
                // Check if the date is valid before formatting
                if (!isNaN(date)) {
                     // Use a simpler format for display
                    const options = { year: 'numeric', month: 'short', day: 'numeric', hour: '2-digit', minute: '2-digit' };
```

```
                            dueDateText = `<span class="task-due-date">Due: ${date.toLocaleString('en-US', options)}</span>`;
                        }
                    }

                    const listItem = `
                        <div class="task-list-item ${task.isCompleted ? 'completed' : "}" data-id="${task.id}">
                            <div class="task-content">
                                <span class="task-text" contenteditable="false">${task.text}</span>
                                ${dueDateText}
                            </div>
                            <div class="task-actions">
                                <button class="edit-btn" title="Edit Task">✏️</button>
                                <button class="delete-btn" title="Delete Task">✕</button>
                            </div>
                        </div>
                    `;
                    $taskList.append(listItem);
                });
            }
        }

        // Function to reorder tasks based on the new visual order
        function reorderTasks(newOrder) {
            const reorderedTasks = newOrder.map(id => tasks.find(task => task.id == id));
            tasks = reorderedTasks.filter(t => t); // Filter out any undefined elements
```

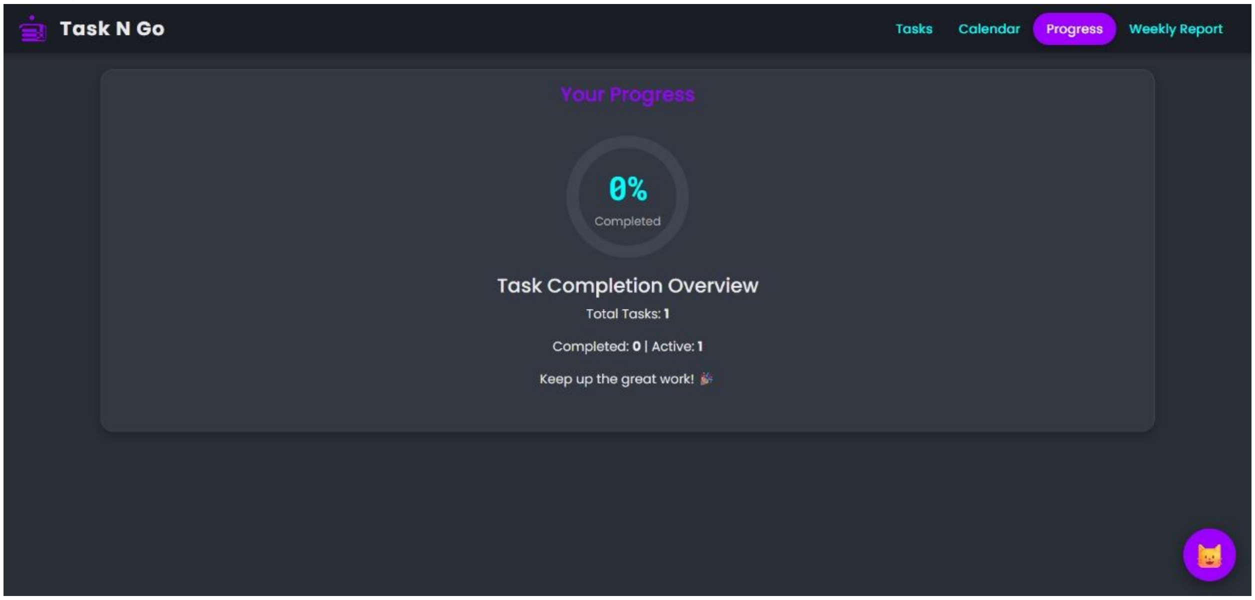
```
    saveTasks();
    // No need to re-render all tabs, just the task list might be enough, but for safety:
    renderAllTabs();
}

function addTask(text, dueDatetime = null) {
    const newTask = {
        id: Date.now(),
        text: text,
        isCompleted: false,
        dueDatetime: dueDatetime || null // Store as null if empty
    };
    tasks.unshift(newTask);
    saveTasks();
    renderAllTabs();
}

function deleteTask(id) {
    tasks = tasks.filter(task => task.id != id);
    saveTasks();
    renderAllTabs();
}

function toggleTaskCompletion(id) {
    const task = tasks.find(t => t.id == id);
    if (task) {
        task.isCompleted = !task.isCompleted;
        if(task.isCompleted){
            task.completionDate = new Date().toISOString();
        } else {
            delete task.completionDate;
        }
        saveTasks();
        renderAllTabs();
    }
}

function updateTaskText(id, newText) {
    const task = tasks.find(t => t.id == id);
    if (task) {
        task.text = newText;
        saveTasks();
        // No need for full renderAllTabs, but re-renderTasks to update the list view
        renderTasks($(".filters button.active").data("filter"));
    }
}

// --- Dynamic Content Rendering for other tabs ---

// Calendar logic
function renderCalendar(date = new Date()) {
    const $calendarContent = $("#calendarContent");
    $calendarContent.empty();
```

```
        const year = date.getFullYear();

        const month = date.getMonth();

        const firstDayOfMonth = new Date(year, month, 1);

        const lastDayOfMonth = new Date(year, month + 1, 0);

        const daysInMonth = lastDayOfMonth.getDate();

        // 0=Sunday, 1=Monday, ...

        const firstDayOfWeek = firstDayOfMonth.getDay();


        const daysOfWeek = ['Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat'];

        let calendarHtml = '<div class="calendar-nav mb-3"><button
id="prevMonthBtn">⍰</button><span>' + firstDayOfMonth.toLocaleDateString('en-US', {
month: 'long', year: 'numeric' }) + '</span><button id="nextMonthBtn">⍰</button></div>';

        calendarHtml += '<div class="calendar-grid">';


        // Day headers

        daysOfWeek.forEach(day => {

          calendarHtml += `<div class="calendar-day-header">${day}</div>`;

        });


        // Empty slots for the first week

        for (let i = 0; i < firstDayOfWeek; i++) {

          calendarHtml += '<div class="calendar-day-box empty-day"></div>';

        }


        // Days of the month

        for (let i = 1; i <= daysInMonth; i++) {

          const day = new Date(year, month, i);
        const hasTasks = tasks.some(t => {
            if (t.dueDatetime) {
                const dueDate = new Date(t.dueDatetime);
                // Compare year, month, and day
                return dueDate.getDate() === day.getDate() &&
                       dueDate.getMonth() === day.getMonth() &&
                       dueDate.getFullYear() === day.getFullYear();
            }
            return false;
        });
        calendarHtml += `<div class="calendar-day-box ${hasTasks ? 'has-tasks' : ''}">
                             <span class="day-number">${i}</span>
                             </div>`;
    }
    calendarHtml += '</div>';

    $calendarContent.html(calendarHtml);
    currentMonthView = date; // Update the global tracker
}

// Progress logic
function renderProgress() {
    const totalTasks = tasks.length;
    const completedTasks = tasks.filter(task => task.isCompleted).length;
    const percentage = totalTasks > 0 ? Math.round((completedTasks / totalTasks) * 100) : 0;
    const activeTasks = totalTasks - completedTasks;
    const $progressContent = $("#progressContent");
```

```
      const progressHtml = `
        <div class="progress-circle" style="--percentage: ${percentage}%;">
          <div class="progress-inner-circle">
            <span class="progress-percentage">${percentage}%</span>
            <span class="progress-label">Completed</span>
          </div>
        </div>
        <div class="progress-stats">
          <h4>Task Completion Overview</h4>
          <p>Total Tasks: <b>${totalTasks}</b></p>
          <p>Completed: <b>${completedTasks}</b> | Active: <b>${activeTasks}</b></p>
          <p>${percentage === 100 ? 'All tasks done! You are amazing! ✨' : 'Keep up the great work! 🎉'}</p>
        </div>
      `;

      $progressContent.html(progressHtml);
    }

    // Weekly Report Logic (Completed)
    function renderReport() {
      const $reportContent = $("#reportContent");
      $reportContent.empty();

      const oneWeekAgo = new Date();
      oneWeekAgo.setDate(oneWeekAgo.getDate() - 7);
      const completedTasksInWeek = tasks.filter(task =>
        task.isCompleted && task.completionDate && new Date(task.completionDate) > oneWeekAgo
      );


      let reportHtml = `<div class="card-body">
                <h5>Last 7 Days (Since ${oneWeekAgo.toLocaleDateString()})</h5>
                <p class="text-muted">You completed **${completedTasksInWeek.length}** task(s) this week.</p>`;


      if (completedTasksInWeek.length > 0) {
        reportHtml += `<ul class="list-group report-list">`;
        completedTasksInWeek.forEach(task => {
          const completionTime = new Date(task.completionDate).toLocaleDateString('en-US', {weekday: 'short', month: 'numeric', day: 'numeric', hour: '2-digit', minute: '2-digit'});
          reportHtml += `
            <li class="list-group-item">
              <span class="report-title">${task.text}</span>
              <span class="report-date">Completed: ${completionTime}</span>
            </li>
          `;
        });
        reportHtml += `</ul>`;
      } else {
         reportHtml += `<p class="text-muted">No tasks were completed in the last week.</p>`;
      }
```

```
      // General Stats
      const totalTasks = tasks.length;
      const completedTotal = tasks.filter(t => t.isCompleted).length;
      const completionRate = totalTasks > 0 ? Math.round((completedTotal / totalTasks) * 100) : 0;

      reportHtml += `<div class="mt-4">
              <h5>Overall Performance</h5>
              <p>Total Tasks Completed: <b>${completedTotal}</b></p>
              <p>Overall Completion Rate: <b>${completionRate}%</b></p>
            </div>
           </div>`;

      $reportContent.html(reportHtml);
    }

    // Dashboard Preview Logic (Completed)
    function updateDashboardPreviews() {
      const totalTasks = tasks.length;
      const completedTasks = tasks.filter(task => task.isCompleted).length;
      const activeTasks = totalTasks - completedTasks;

      // Calendar Preview
      const tasksDueThisMonth = tasks.filter(t => t.dueDatetime && new Date(t.dueDatetime).getMonth() === new Date().getMonth()).length;
      $("#calendarPreview").html(`<p>${tasksDueThisMonth} task(s) due this month.</p>`);
      // Progress Preview
      const percentage = totalTasks > 0 ? Math.round((completedTasks / totalTasks) * 100) : 0;
      $("#progressPreview").html(`<p>Completion: **${percentage}%**</p><p>${activeTasks} active tasks.</p>`);

      // Report Preview
      const oneWeekAgo = new Date();
      oneWeekAgo.setDate(oneWeekAgo.getDate() - 7);
      const completedInWeek = tasks.filter(task =>
        task.isCompleted && task.completionDate && new Date(task.completionDate) > oneWeekAgo
      ).length;
      $("#reportPreview").html(`<p>Completed **${completedInWeek}** task(s) this week.</p>`);
    }

  </script>
</body>
</html>
```

13. Screenshots of Final Output



Your Progress

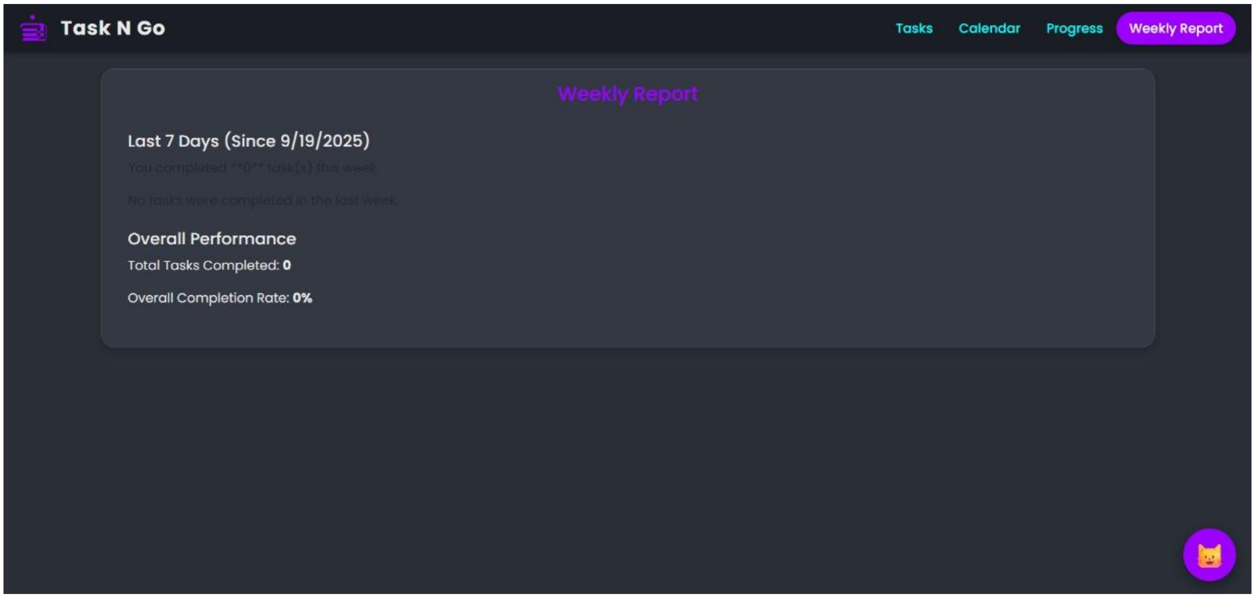
0%
Completed

Task Completion Overview

Total Tasks: **1**

Completed: **0** | Active: **1**

Keep up the great work! 🎉



Weekly Report

Last 7 Days (Since 9/19/2025)

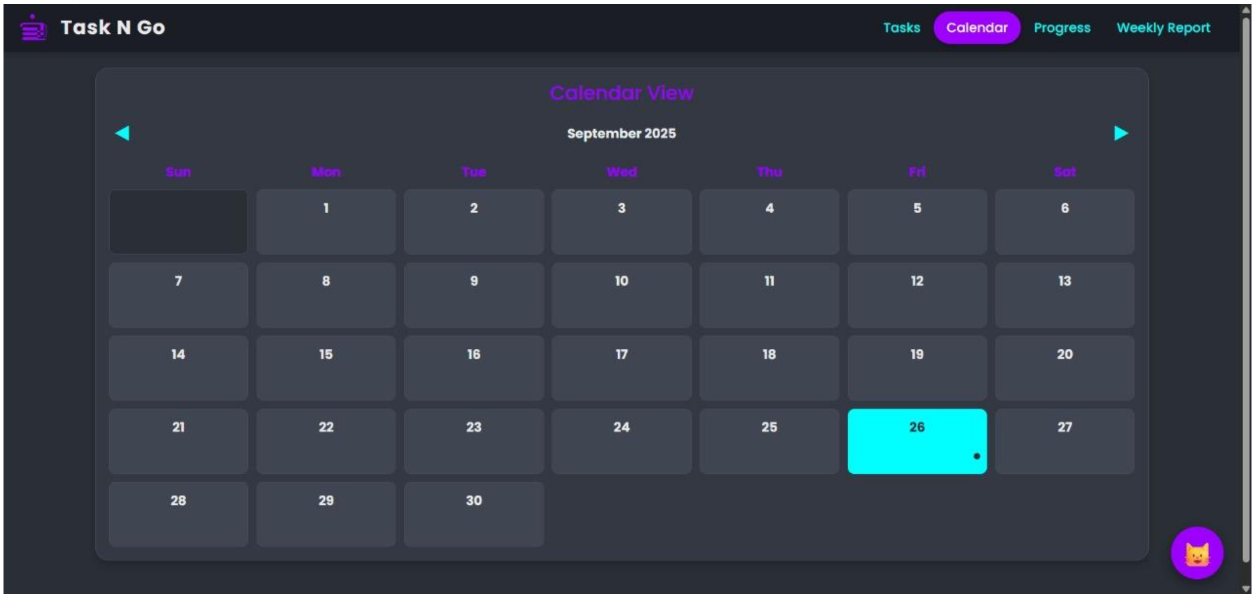
You completed **0** task(s) this week.

No tasks were completed in the last week.

Overall Performance

Total Tasks Completed: **0**

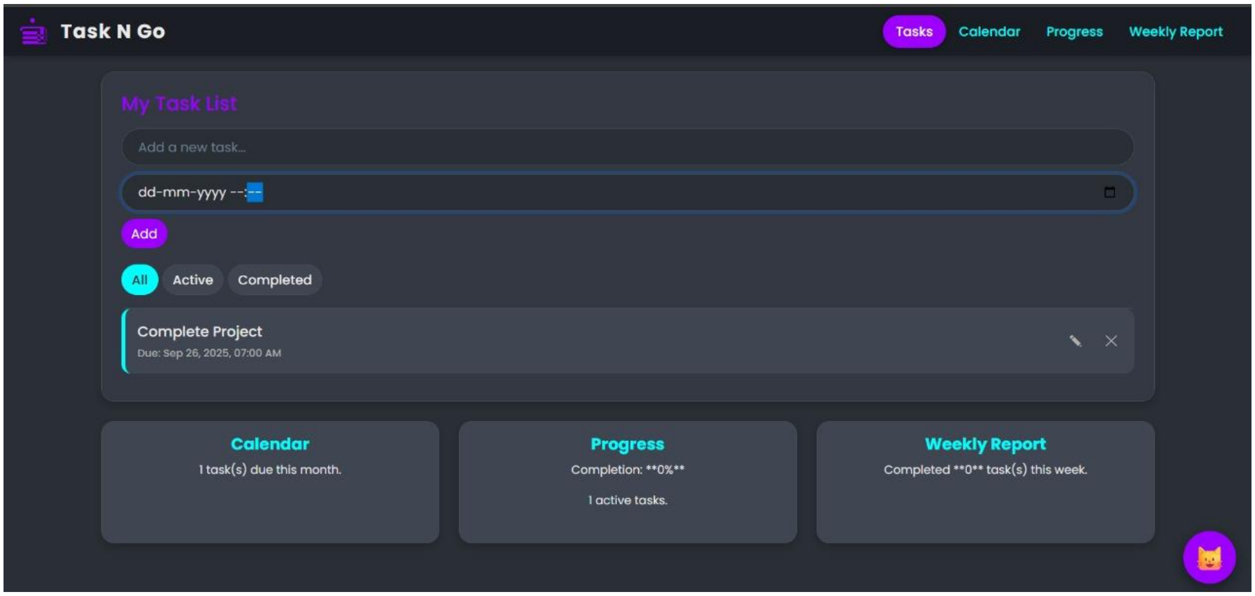
Overall Completion Rate: **0%**



Calendar View

September 2025

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26 ●	27
28	29	30				



My Task List

Add a new task...

dd-mm-yyyy --:--

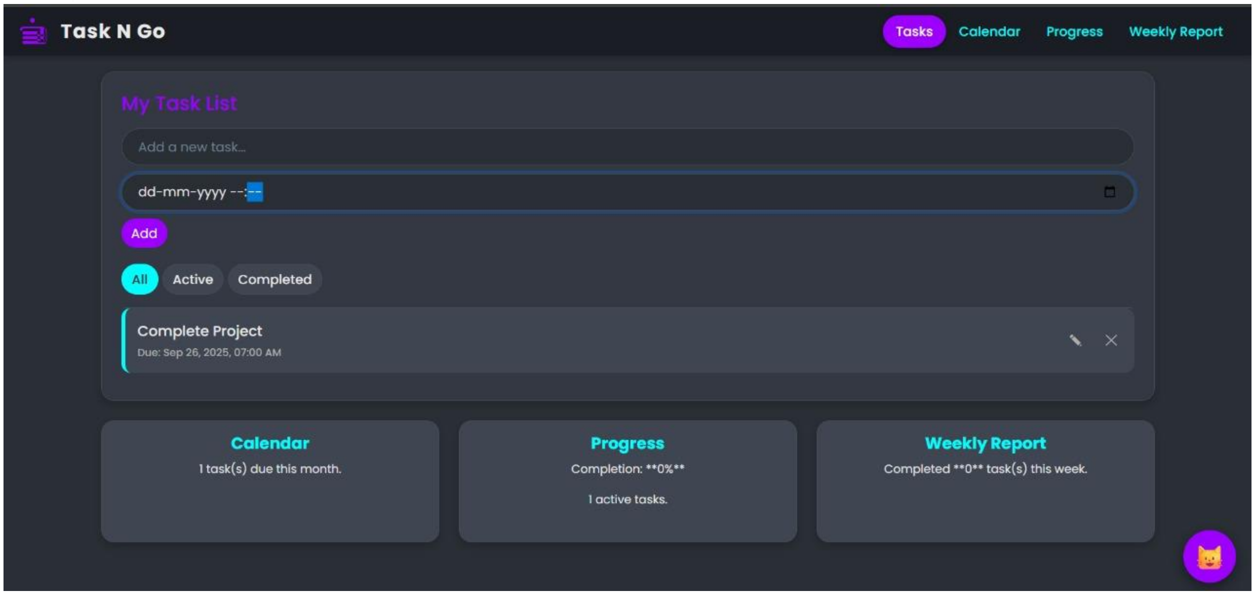
Complete Project
Due: Sep 26, 2025, 07:00 AM

Calendar
1 task(s) due this month.

Progress
Completion: **0%**

1 active tasks.

Weekly Report
Completed **0** task(s) this week.



My Task List

Add a new task...

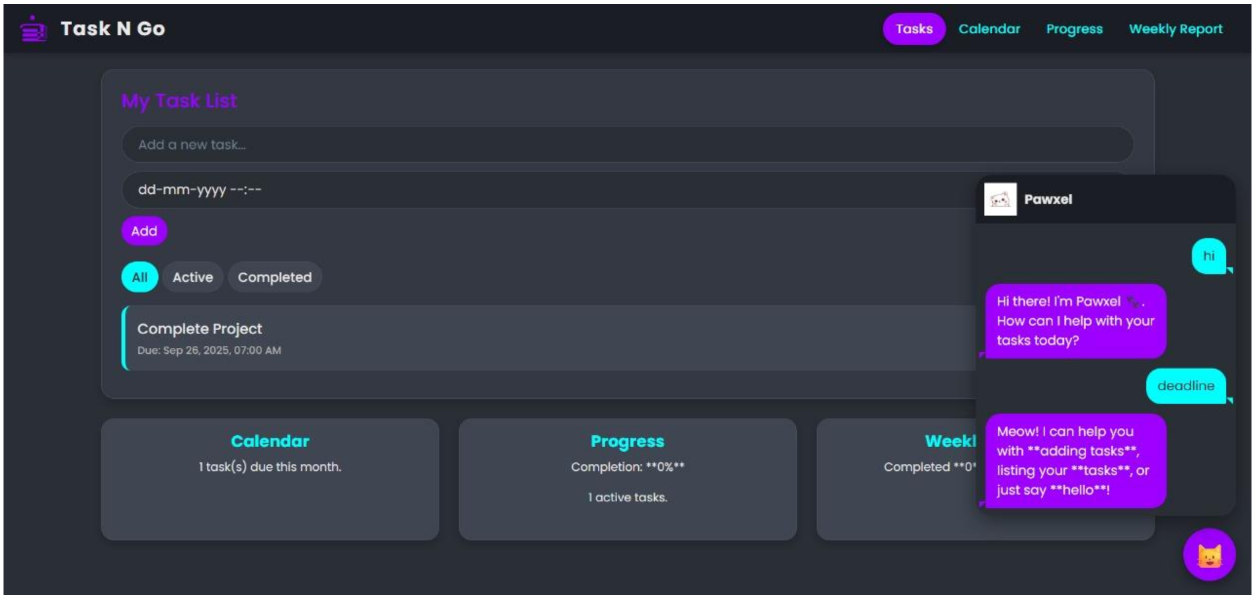
dd-mm-yyyy --:--

Complete Project
Due: Sep 26, 2025, 07:00 AM

Calendar
1 task(s) due this month.

Progress
Completion: **0%**
1 active tasks.

Weekly Report
Completed **0** task(s) this week.



My Task List

Add a new task...

dd-mm-yyyy --:--

Complete Project
Due: Sep 26, 2025, 07:00 AM

Calendar
1 task(s) due this month.

Progress
Completion: **0%**
1 active tasks.

Weekl
Completed **0*

Pawxel

hi

Hi there! I'm Pawxel 🐾. How can I help with your tasks today?

deadline

Meow! I can help you with **adding tasks**, listing your **tasks**, or just say **hello**!

11. Conclusion

The To-Do List Application successfully demonstrates core web development skills by integrating HTML, CSS, JavaScript, Bootstrap, and jQuery to build a responsive and interactive task management tool. Users can efficiently add, edit, delete, and filter tasks, enhancing productivity and user experience. This project not only reinforces foundational programming concepts but also showcases practical implementation of dynamic UI behavior and responsive design—making it a solid stepping stone for more advanced web applications.

12. References

- Mozilla Developer Network (MDN). *HTML, CSS, and JavaScript documentation*. Retrieved from https://developer.mozilla.org
- Bootstrap Official Documentation. *Responsive design and components*. Retrieved from https://getbootstrap.com
- jQuery Documentation. *Simplifying JavaScript interactions*. Retrieved from https://api.jquery.com
- W3Schools. *Web development tutorials and examples*. Retrieved from https://www.w3schools.com