CAPSTONE PROJECT

Smart Study Planner: A Client-Side Task Management Application

Student Name : Mohammed Zubair A
College Name : Sir M. Visvesvaraya Institute of Technology
Department : Information Science and Engineering

edunet
foundation

OUTLINE

- **Problem Statement** (Should not include solution)
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment (Step by Step Procedure)**
- **Result**
- **Conclusion**
- **Future Scope(Optional)**
- **References**

Problem Statement

- Students often struggle with organizing study schedules and tracking a multitude of academic tasks, leading to stress and missed deadlines.

- Existing digital planners can be overly complex, expensive, or lack features specifically tailored for academic purposes.

- There is a clear need for a lightweight, visually engaging, and easy-to-use planner that helps students manage their goals effectively.

- This tool must allow students to create, view, and interact with their study tasks in a simple and motivating way.

System Approach

Frontend Technologies:

- **HTML5:** Used to build the fundamental structure and semantic layout of the web application, including the task list, modals, and widgets.
- **CSS3:** Employed for all visual styling, creating a modern and clean user interface. This includes responsive layouts using Flexbox/Grid, animations for a dynamic feel, and a mobile-first design.
- **JavaScript (ES6+):** The core engine for all interactivity, handling dynamic task creation, updates, completion, deletion, and real-time progress calculations.

Data Storage Approach:

- **Browser Local Storage:** To save all user-generated tasks and their statuses directly in the browser, ensuring data persists between sessions without requiring a backend database.

Design Approach:

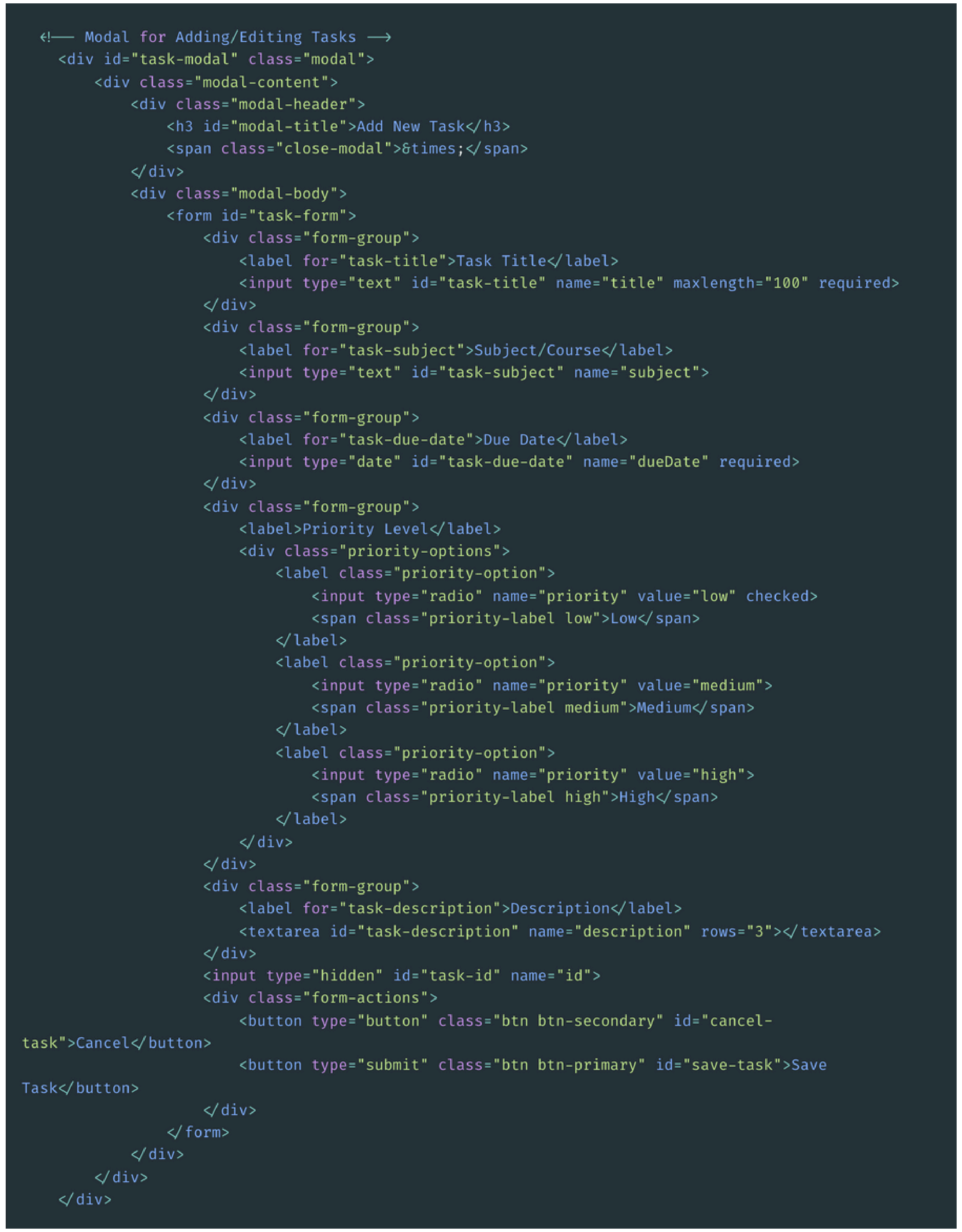
- A clean, intuitive, and user-friendly interface designed to minimize distraction and maximize productivity.
- A fully mobile-responsive design ensuring a seamless experience on desktops, tablets, and smartphones.
- Modular code with separate files for HTML, CSS, and JavaScript to ensure maintainability and scalability.

Algorithm & Deployment

Step-by-Step Procedure:

UI Scaffolding (HTML):

- The application structure was defined with a main container, a task display section, and a modal template for adding/editing tasks.

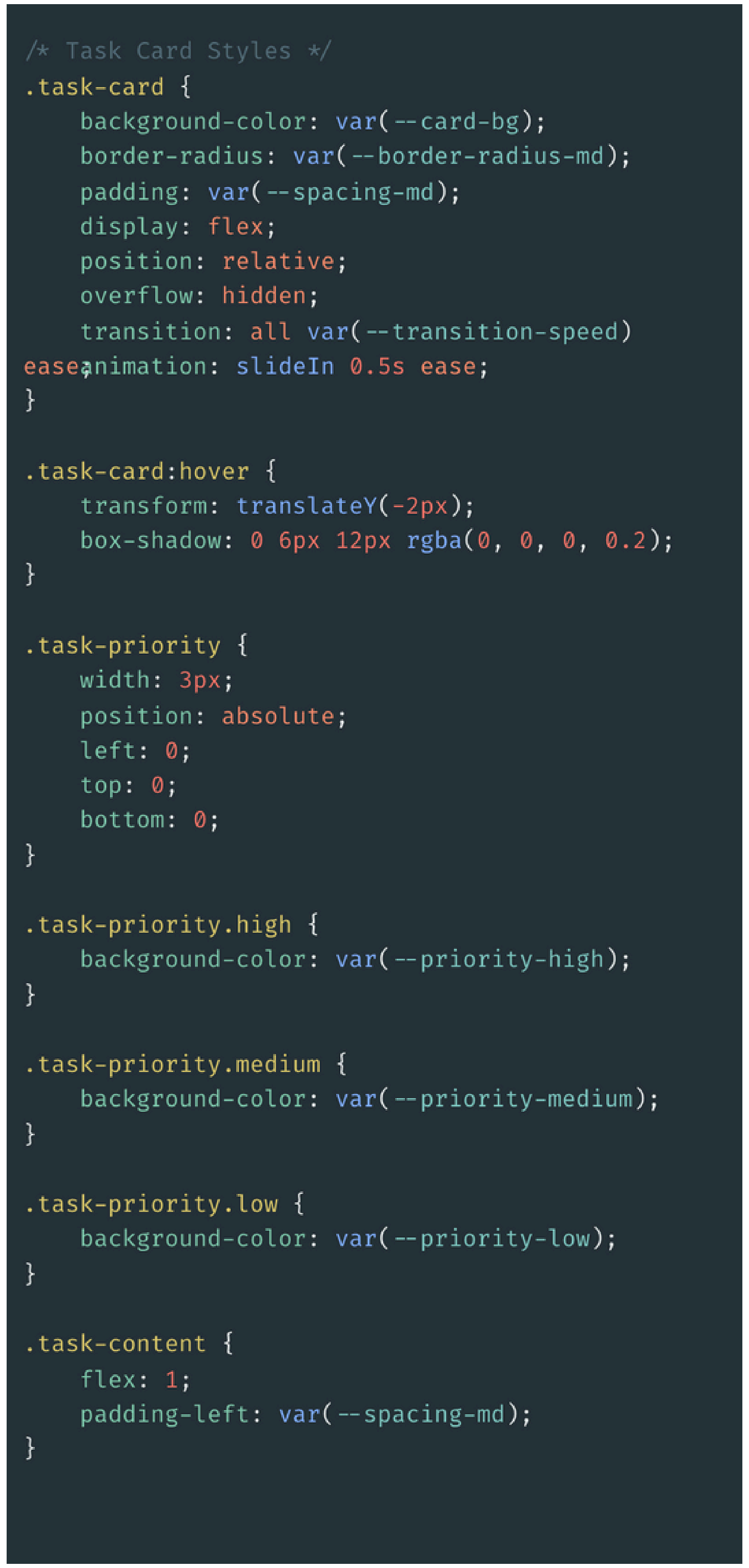


```html
<!-- Modal for Adding/Editing Tasks -->
<div id="task-modal" class="modal">
    <div class="modal-content">
        <div class="modal-header">
            <h3 id="modal-title">Add New Task</h3>
            <span class="close-modal">&times;</span>
        </div>
        <div class="modal-body">
            <form id="task-form">
                <div class="form-group">
                    <label for="task-title">Task Title</label>
                    <input type="text" id="task-title" name="title" maxlength="100" required>
                </div>
                <div class="form-group">
                    <label for="task-subject">Subject/Course</label>
                    <input type="text" id="task-subject" name="subject">
                </div>
                <div class="form-group">
                    <label for="task-due-date">Due Date</label>
                    <input type="date" id="task-due-date" name="dueDate" required>
                </div>
                <div class="form-group">
                    <label>Priority Level</label>
                    <div class="priority-options">
                        <label class="priority-option">
                            <input type="radio" name="priority" value="low" checked>
                            <span class="priority-label low">Low</span>
                        </label>
                        <label class="priority-option">
                            <input type="radio" name="priority" value="medium">
                            <span class="priority-label medium">Medium</span>
                        </label>
                        <label class="priority-option">
                            <input type="radio" name="priority" value="high">
                            <span class="priority-label high">High</span>
                        </label>
                    </div>
                </div>
                <div class="form-group">
                    <label for="task-description">Description</label>
                    <textarea id="task-description" name="description" rows="3"></textarea>
                </div>
                <input type="hidden" id="task-id" name="id">
                <div class="form-actions">
                    <button type="button" class="btn btn-secondary" id="cancel-
task">Cancel</button>
                    <button type="submit" class="btn btn-primary" id="save-task">Save
Task</button>
                </div>
            </form>
        </div>
    </div>
</div>
```

Algorithm & Deployment

Frontend Styling (CSS):

- A modern, dark-mode theme was implemented to be easy on the eyes.
- Task cards were designed with priority-colored borders, and interactive elements were given hover effects and smooth transitions.

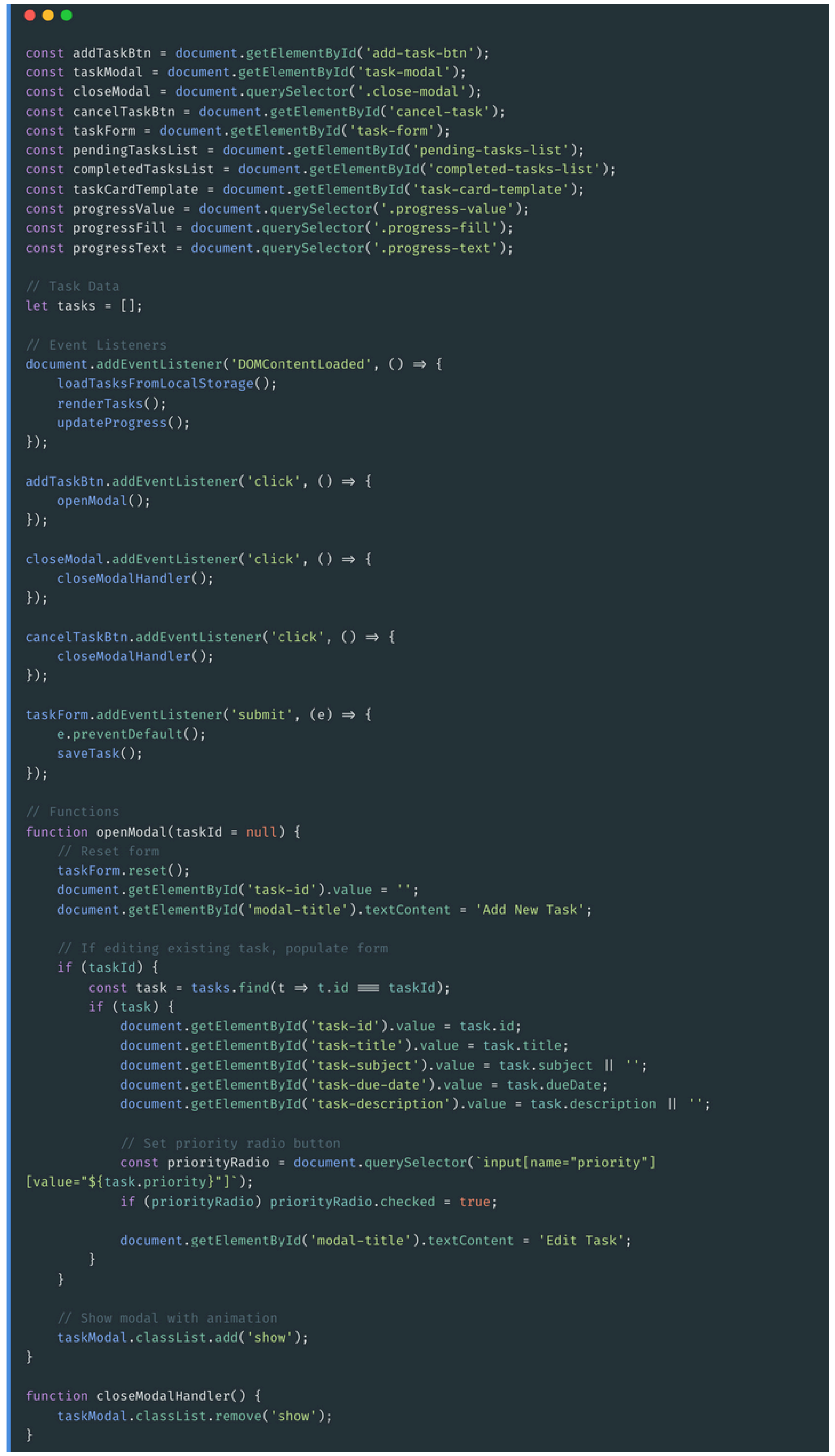


```css
/* Task Card Styles */
.task-card {
    background-color: var(--card-bg);
    border-radius: var(--border-radius-md);
    padding: var(--spacing-md);
    display: flex;
    position: relative;
    overflow: hidden;
    transition: all var(--transition-speed)
ease;animation: slideIn 0.5s ease;
}

.task-card:hover {
    transform: translateY(-2px);
    box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);
}

.task-priority {
    width: 3px;
    position: absolute;
    left: 0;
    top: 0;
    bottom: 0;
}

.task-priority.high {
    background-color: var(--priority-high);
}

.task-priority.medium {
    background-color: var(--priority-medium);
}

.task-priority.low {
    background-color: var(--priority-low);
}

.task-content {
    flex: 1;
    padding-left: var(--spacing-md);
}
```

Algorithm & Deployment

Core Logic Implementation (JavaScript):

- Functions were developed to handle CRUD (Create, Read, Update, Delete) operations for tasks.
- Logic was written to serialize the task list into a JSON string for storage in localStorage and parse it back on page load.

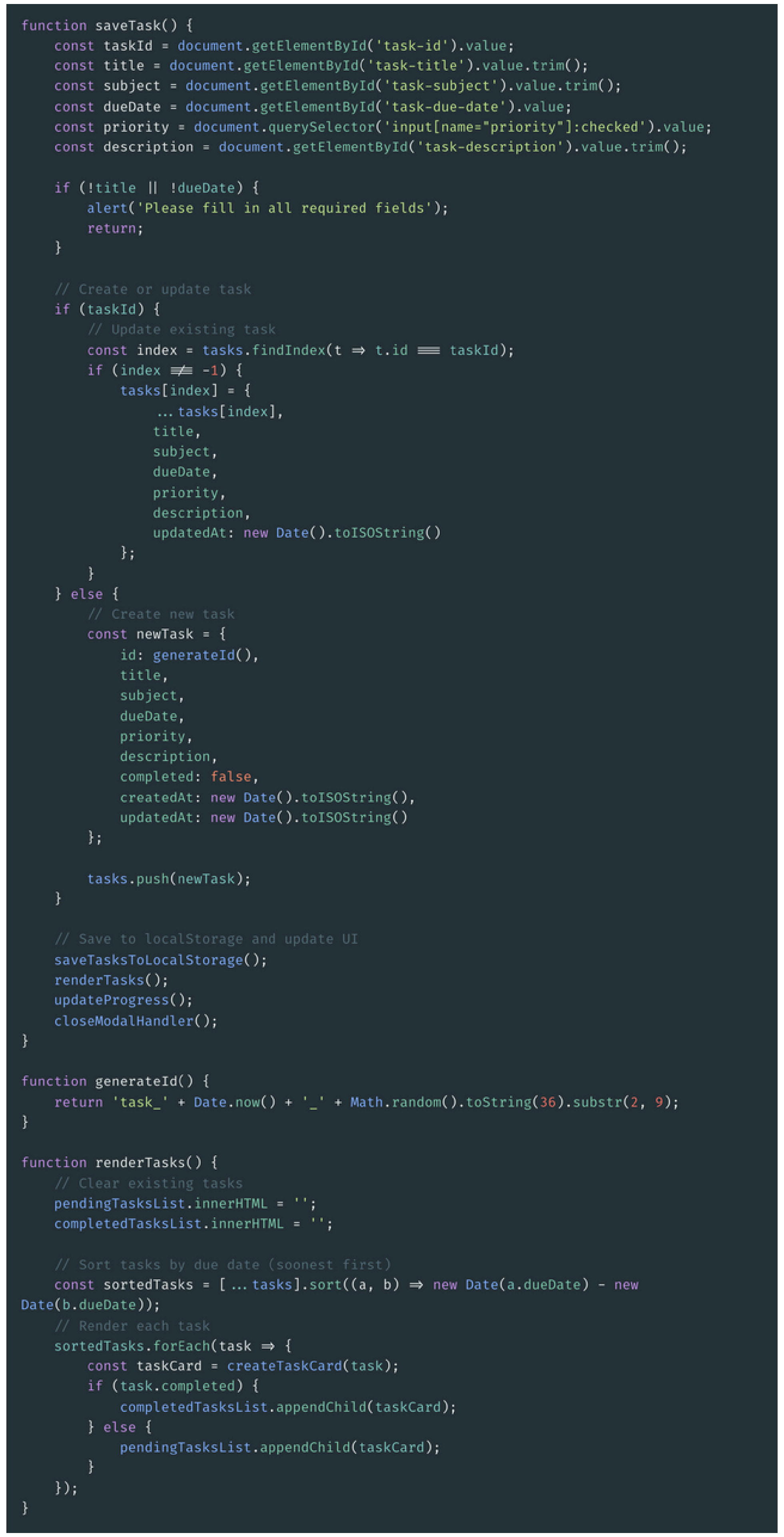


```javascript
const addTaskBtn = document.getElementById('add-task-btn');
const taskModal = document.getElementById('task-modal');
const closeModal = document.querySelector('.close-modal');
const cancelTaskBtn = document.getElementById('cancel-task');
const taskForm = document.getElementById('task-form');
const pendingTasksList = document.getElementById('pending-tasks-list');
const completedTasksList = document.getElementById('completed-tasks-list');
const taskCardTemplate = document.getElementById('task-card-template');
const progressValue = document.querySelector('.progress-value');
const progressFill = document.querySelector('.progress-fill');
const progressText = document.querySelector('.progress-text');

// Task Data
let tasks = [];

// Event Listeners
document.addEventListener('DOMContentLoaded', () => {
    loadTasksFromLocalStorage();
    renderTasks();
    updateProgress();
});

addTaskBtn.addEventListener('click', () => {
    openModal();
});

closeModal.addEventListener('click', () => {
    closeModalHandler();
});

cancelTaskBtn.addEventListener('click', () => {
    closeModalHandler();
});

taskForm.addEventListener('submit', (e) => {
    e.preventDefault();
    saveTask();
});

// Functions
function openModal(taskId = null) {
    // Reset form
    taskForm.reset();
    document.getElementById('task-id').value = '';
    document.getElementById('modal-title').textContent = 'Add New Task';

    // If editing existing task, populate form
    if (taskId) {
        const task = tasks.find(t => t.id === taskId);
        if (task) {
            document.getElementById('task-id').value = task.id;
            document.getElementById('task-title').value = task.title;
            document.getElementById('task-subject').value = task.subject || '';
            document.getElementById('task-due-date').value = task.dueDate;
            document.getElementById('task-description').value = task.description || '';

            // Set priority radio button
            const priorityRadio = document.querySelector(`input[name="priority"]
[value="${task.priority}"]`);
            if (priorityRadio) priorityRadio.checked = true;

            document.getElementById('modal-title').textContent = 'Edit Task';
        }
    }

    // Show modal with animation
    taskModal.classList.add('show');
}

function closeModalHandler() {
    taskModal.classList.remove('show');
}
```

Algorithm & Deployment

Interactive Feature Development (JavaScript):

- Implemented an "Add Task" function that captures user input from the modal and renders a new task on the dashboard.
- Developed "Mark as Complete" and "Delete" functionalities, which update the task's state in the UI and localStorage.
- Created a function for real-time calculation and display of the task completion percentage in the progress widget.



```javascript
function saveTask() {
    const taskId = document.getElementById('task-id').value;
    const title = document.getElementById('task-title').value.trim();
    const subject = document.getElementById('task-subject').value.trim();
    const dueDate = document.getElementById('task-due-date').value;
    const priority = document.querySelector('input[name="priority"]:checked').value;
    const description = document.getElementById('task-description').value.trim();

    if (!title || !dueDate) {
        alert('Please fill in all required fields');
        return;
    }

    // Create or update task
    if (taskId) {
        // Update existing task
        const index = tasks.findIndex(t => t.id === taskId);
        if (index !== -1) {
            tasks[index] = {
                ...tasks[index],
                title,
                subject,
                dueDate,
                priority,
                description,
                updatedAt: new Date().toISOString()
            };
        }
    } else {
        // Create new task
        const newTask = {
            id: generateId(),
            title,
            subject,
            dueDate,
            priority,
            description,
            completed: false,
            createdAt: new Date().toISOString(),
            updatedAt: new Date().toISOString()
        };

        tasks.push(newTask);
    }

    // Save to localStorage and update UI
    saveTasksToLocalStorage();
    renderTasks();
    updateProgress();
    closeModalHandler();
}

function generateId() {
    return 'task_' + Date.now() + '_' + Math.random().toString(36).substr(2, 9);
}

function renderTasks() {
    // Clear existing tasks
    pendingTasksList.innerHTML = '';
    completedTasksList.innerHTML = '';

    // Sort tasks by due date (soonest first)
    const sortedTasks = [...tasks].sort((a, b) => new Date(a.dueDate) - new
Date(b.dueDate));
    // Render each task
    sortedTasks.forEach(task => {
        const taskCard = createTaskCard(task);
        if (task.completed) {
            completedTasksList.appendChild(taskCard);
        } else {
            pendingTasksList.appendChild(taskCard);
        }
    });
}
```

Algorithm & Deployment

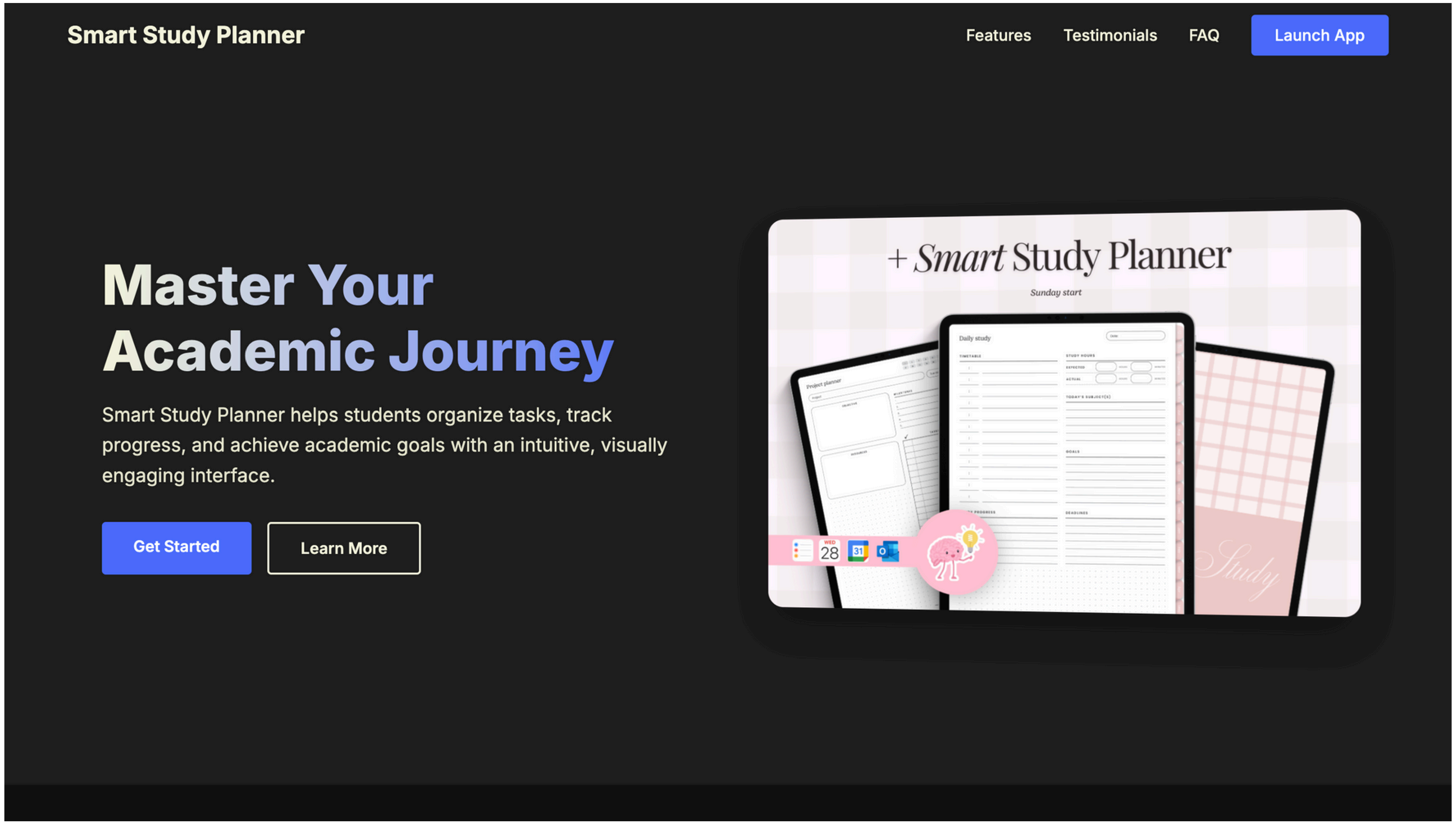
Testing:

- Performed manual testing to ensure all features work as expected.
- Checked for cross-browser compatibility on Chrome, Firefox, and Edge.
- Thoroughly tested the mobile responsiveness on various screen sizes.

Deployment:

- The final code was pushed to a GitHub repository.
- The application was deployed as a static website using
- GitHub Pages, making it freely and publicly accessible via a URL

Result



Smart Study Planner

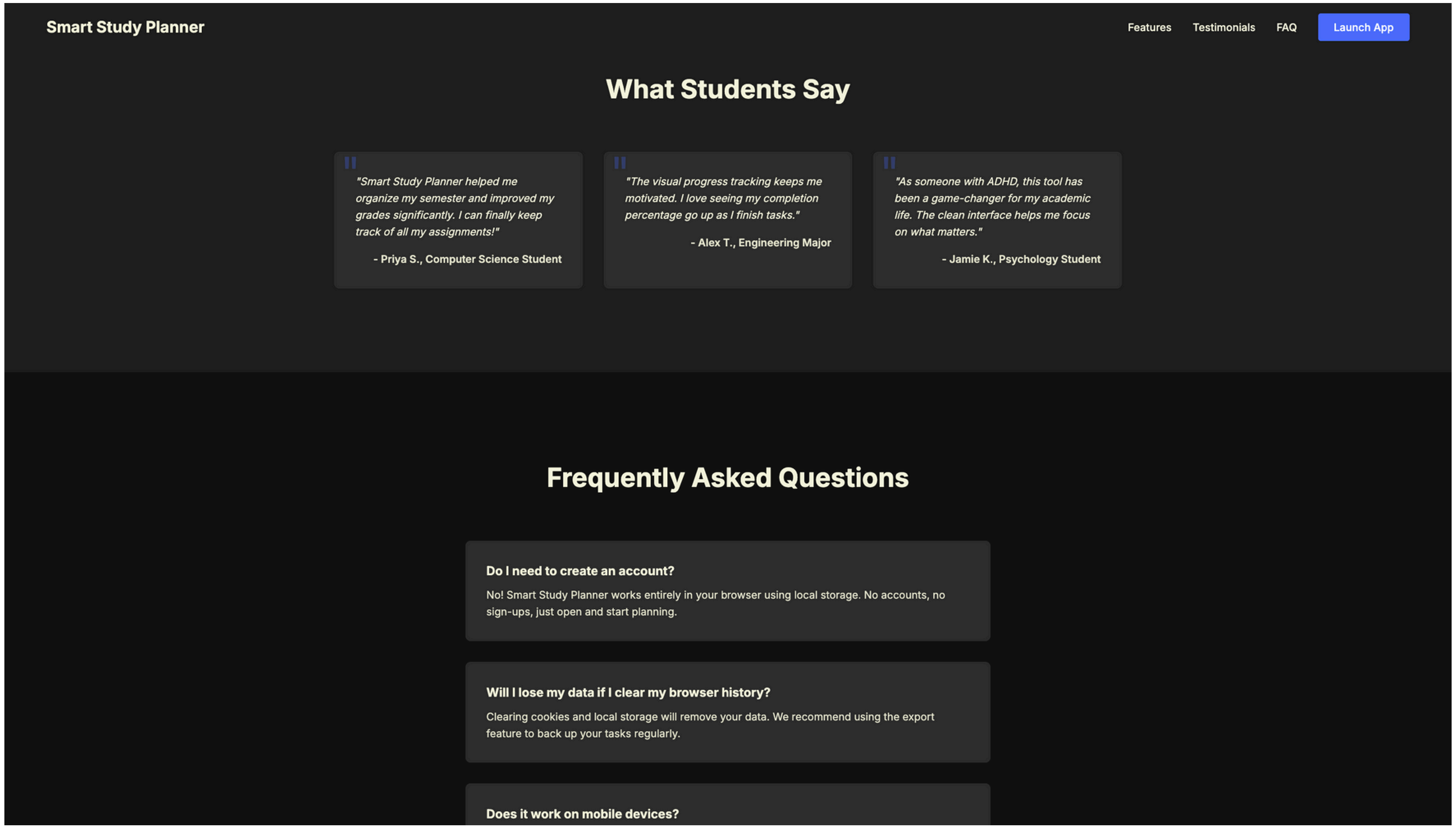
Features Testimonials FAQ Launch App

Master Your Academic Journey

Smart Study Planner helps students organize tasks, track progress, and achieve academic goals with an intuitive, visually engaging interface.

Get Started Learn More

Result



What Students Say

"Smart Study Planner helped me organize my semester and improved my grades significantly. I can finally keep track of all my assignments!"

- Priya S., Computer Science Student

"The visual progress tracking keeps me motivated. I love seeing my completion percentage go up as I finish tasks."

- Alex T., Engineering Major

"As someone with ADHD, this tool has been a game-changer for my academic life. The clean interface helps me focus on what matters."

- Jamie K., Psychology Student

Frequently Asked Questions

Do I need to create an account?

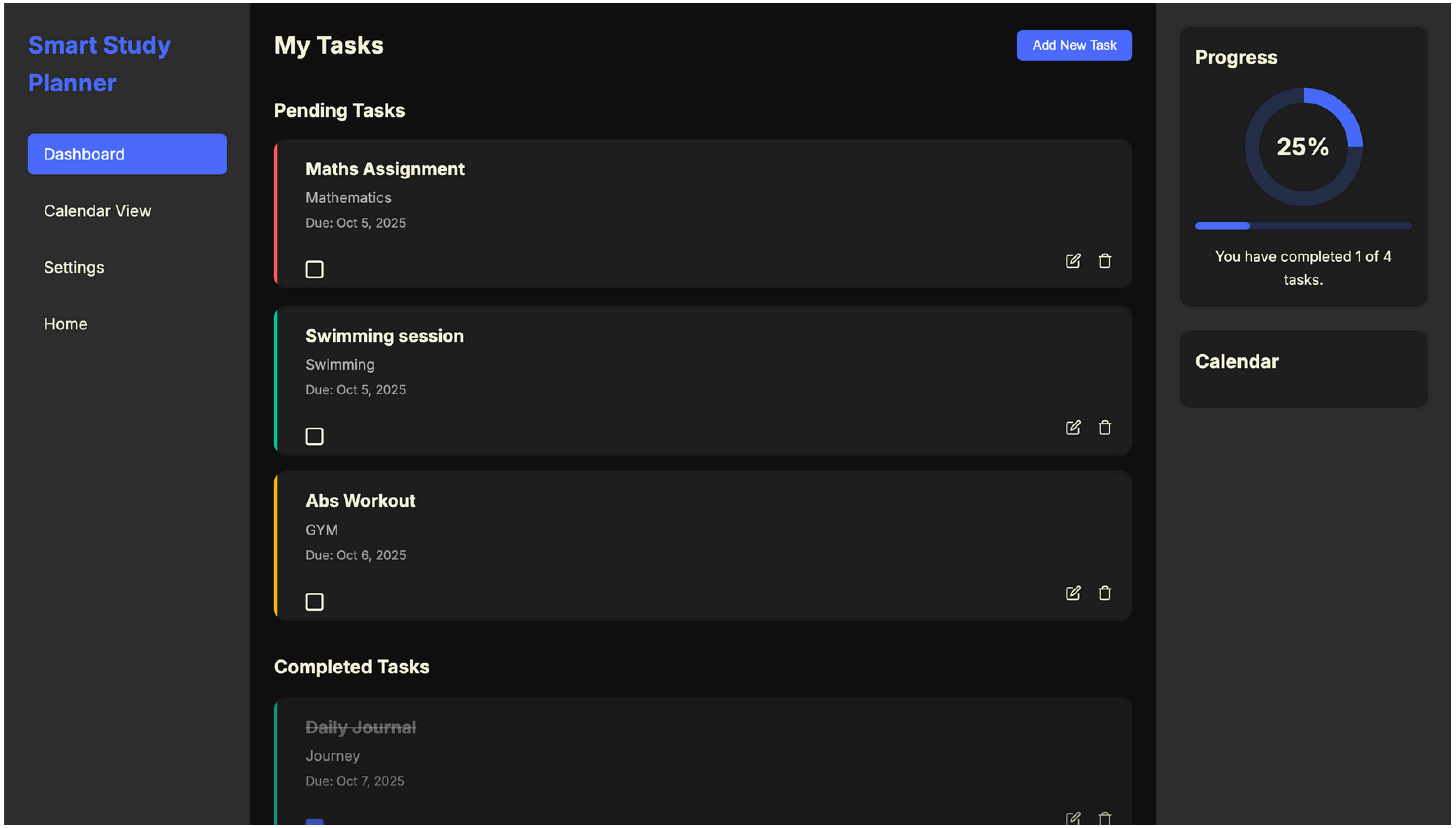
No! Smart Study Planner works entirely in your browser using local storage. No accounts, no sign-ups, just open and start planning.

Will I lose my data if I clear my browser history?

Clearing cookies and local storage will remove your data. We recommend using the export feature to back up your tasks regularly.

Does it work on mobile devices?

Result



Smart Study Planner

Dashboard

Calendar View

Settings

Home

My Tasks

Add New Task

Pending Tasks

Maths Assignment
Mathematics
Due: Oct 5, 2025

Swimming session
Swimming
Due: Oct 5, 2025

Abs Workout
GYM
Due: Oct 6, 2025

Completed Tasks

~~**Daily Journal**~~
Journey
Due: Oct 7, 2025

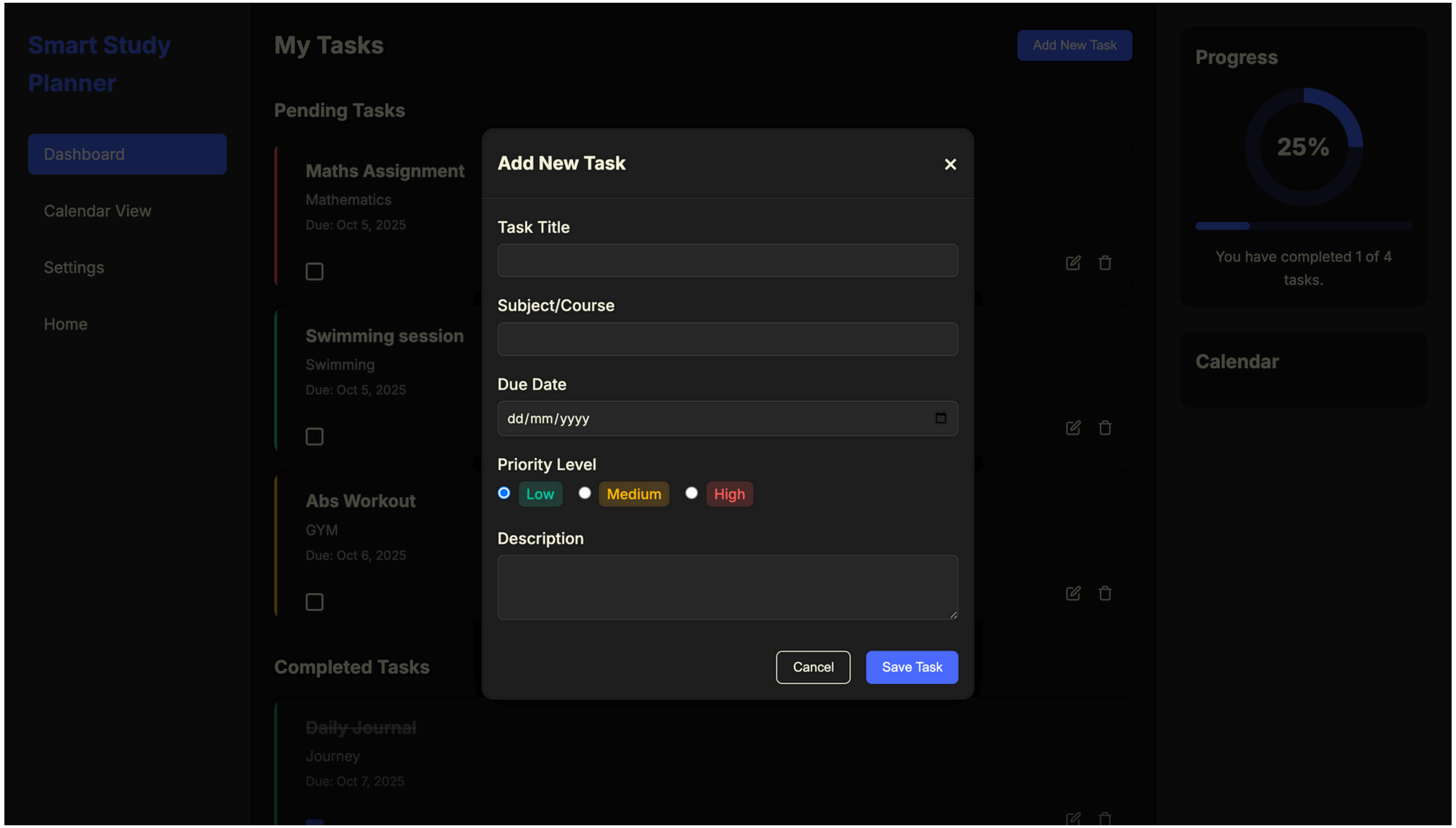
Progress

25%

You have completed 1 of 4 tasks.

Calendar

Result



Smart Study Planner

Dashboard

Calendar View

Settings

Home

My Tasks

Add New Task

Pending Tasks

Maths Assignment
Mathematics
Due: Oct 5, 2025

Swimming session
Swimming
Due: Oct 5, 2025

Abs Workout
GYM
Due: Oct 6, 2025

Completed Tasks

~~Daily Journal~~
Journey
Due: Oct 7, 2025

Progress

25%

You have completed 1 of 4 tasks.

Calendar

Add New Task

Task Title

Subject/Course

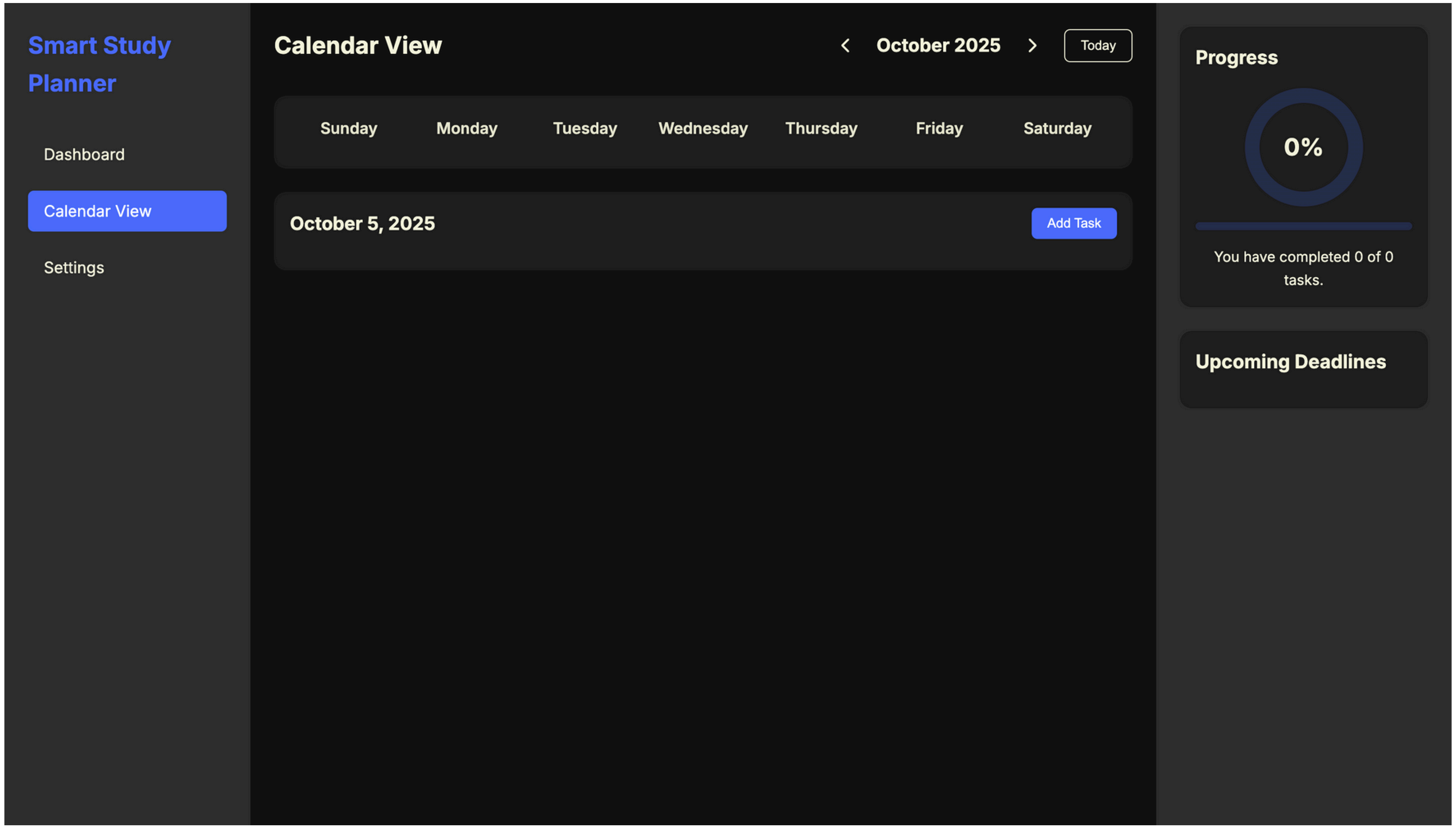
Due Date
dd/mm/yyyy

Priority Level
[✓ Low] Medium High

Description

Cancel
Save Task

Result



Smart Study Planner

Dashboard

Calendar View

Settings

Calendar View

October 2025

Today

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday

October 5, 2025

Add Task

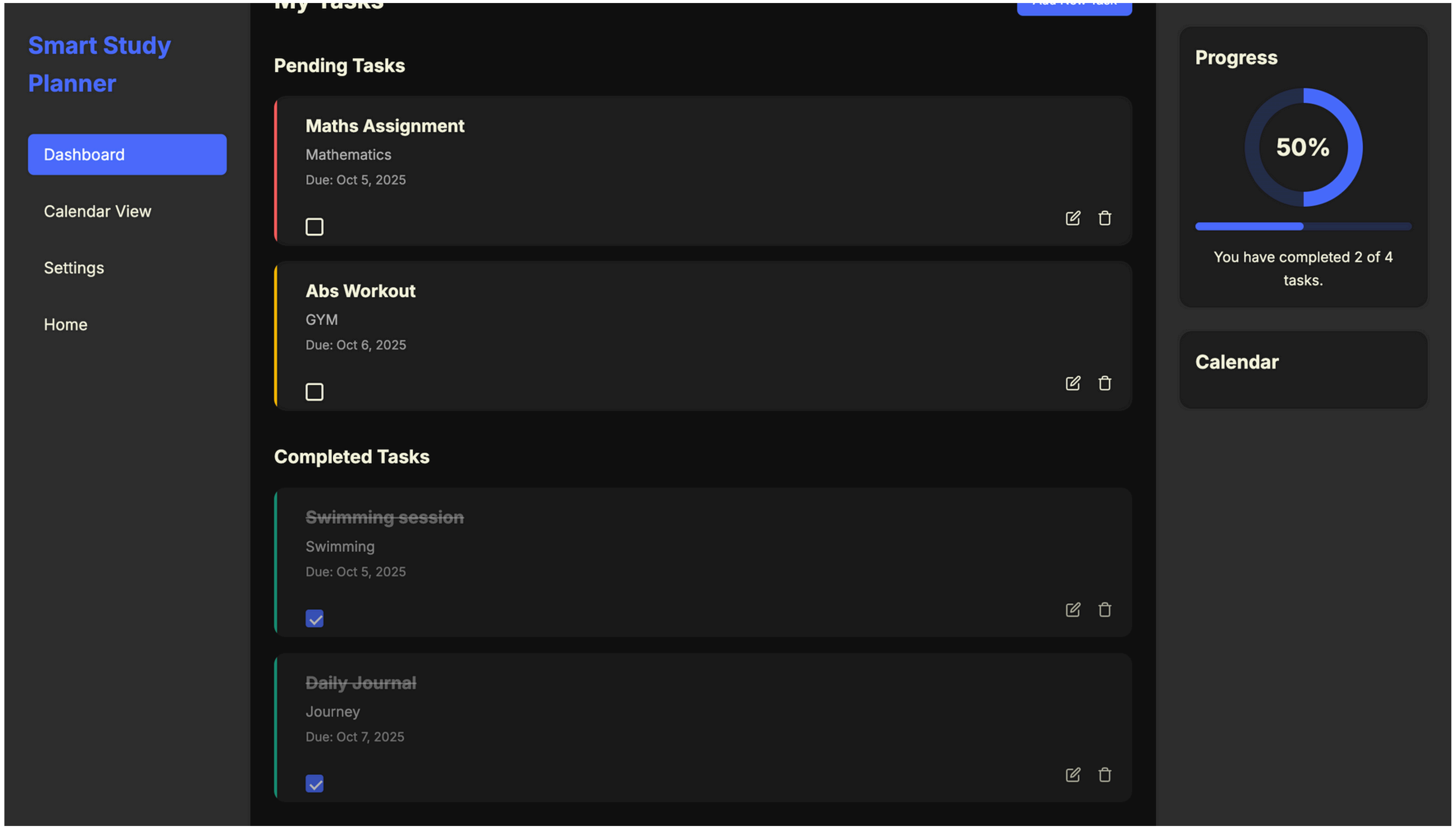
Progress

0%

You have completed 0 of 0 tasks.

Upcoming Deadlines

Result



Smart Study Planner

- Dashboard
- Calendar View
- Settings
- Home

My Tasks

Pending Tasks

Maths Assignment
Mathematics
Due: Oct 5, 2025

Abs Workout
GYM
Due: Oct 6, 2025

Completed Tasks

~~**Swimming session**~~
Swimming
Due: Oct 5, 2025

~~**Daily Journal**~~
Journey
Due: Oct 7, 2025

Progress

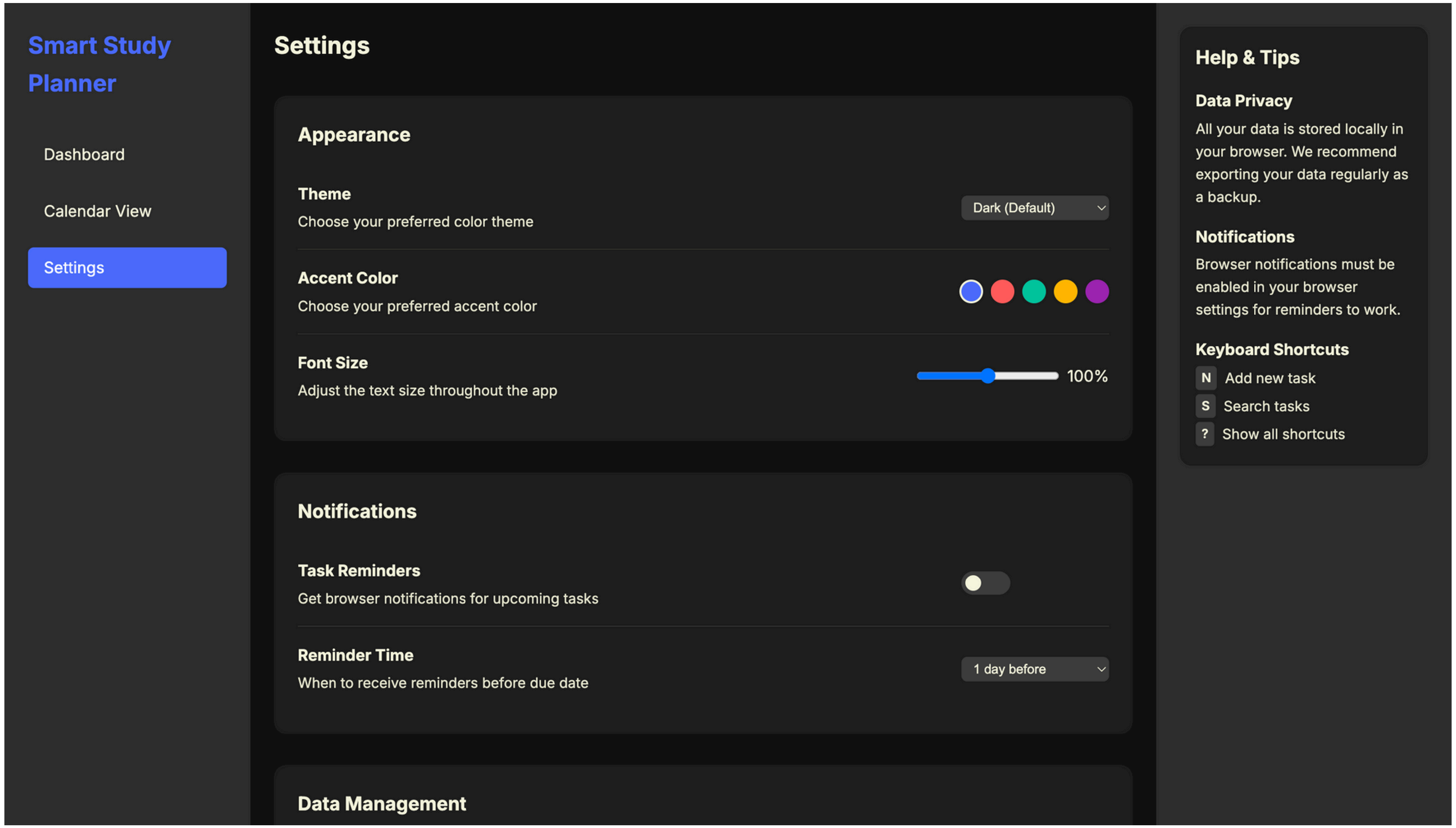
50%

You have completed 2 of 4 tasks.

Calendar

edunet foundation

Result



Smart Study Planner

Dashboard

Calendar View

Settings

Settings

Appearance

Theme
Choose your preferred color theme — Dark (Default)

Accent Color
Choose your preferred accent color

Font Size
Adjust the text size throughout the app — 100%

Notifications

Task Reminders
Get browser notifications for upcoming tasks

Reminder Time
When to receive reminders before due date — 1 day before

Data Management

Help & Tips

Data Privacy
All your data is stored locally in your browser. We recommend exporting your data regularly as a backup.

Notifications
Browser notifications must be enabled in your browser settings for reminders to work.

Keyboard Shortcuts

N	Add new task
S	Search tasks
?	Show all shortcuts

GITHUB AND DEPLOYMNET LINK

- **Github Link : https://github.com/Mdzub7/Smart-Study-Planner**
- **Deployment link: https://mdzub7.github.io/Smart-Study-Planner/**

Conclusion

- The Smart Study Planner provides an effective, no-cost solution for students to organize their academic lives and enhance productivity.
- This project successfully eliminates the need for complex and expensive productivity software by focusing on core, essential features.
- It delivers an engaging and interactive user experience that motivates students to stay on top of their tasks.
- The use of client-side technologies makes the application fast, secure, and accessible even when offline.

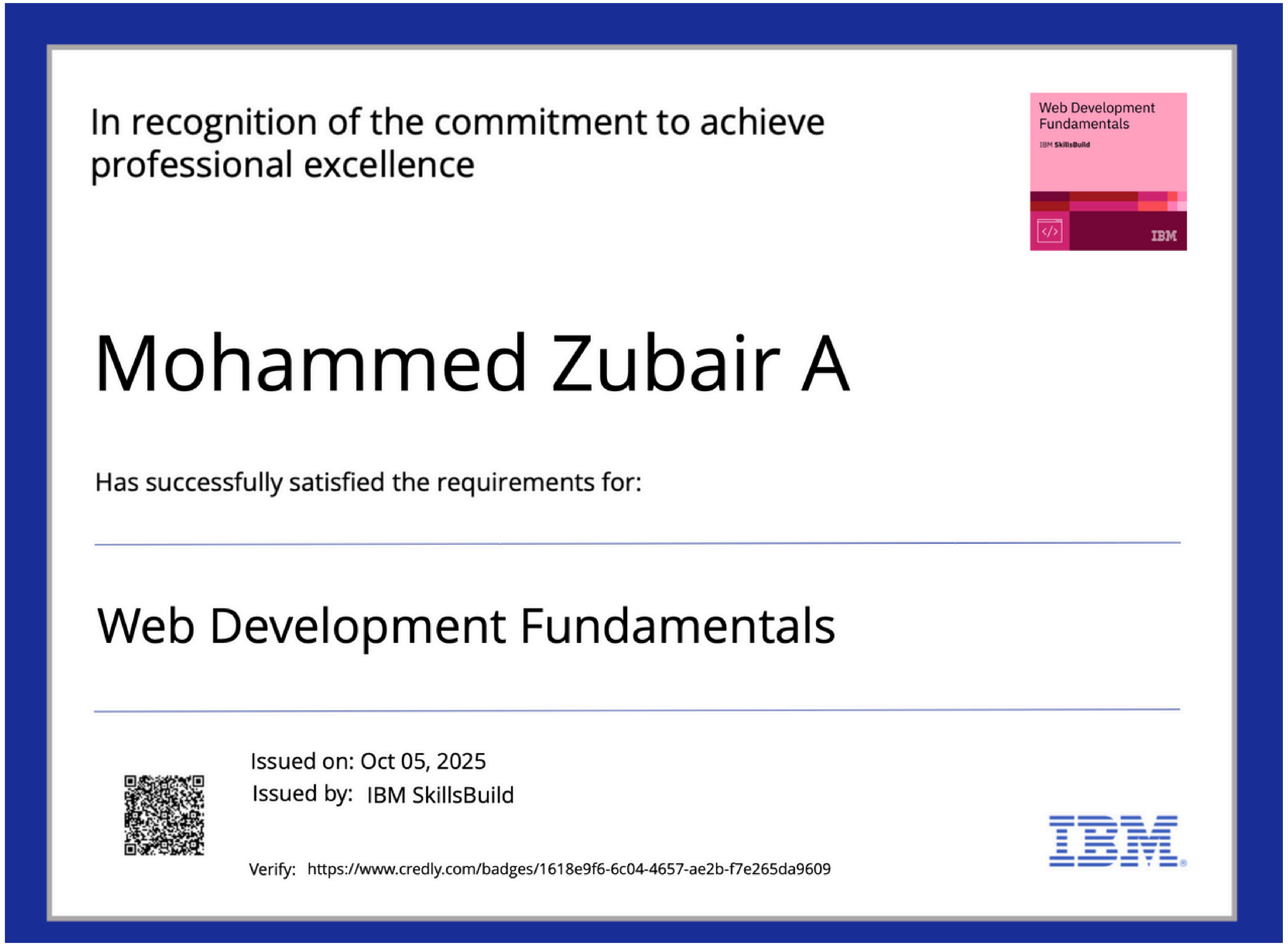
Future scope

- **Cloud Synchronization:** Integrate with a service like Firebase to allow users to sync their tasks across multiple devices.
- **Push Notifications:** Add browser-based push notifications as reminders for approaching deadlines.
- **Backend & User Accounts:** Develop a Node.js backend to support user accounts, enabling data backup and collaborative features.
- **AI-Powered Suggestions:** Implement a feature that suggests optimal study times based on a user's task load and habits.

References

- **MDN Web Docs -** For comprehensive documentation on HTML5, CSS3, and JavaScript.
- **W3Schools -** For tutorials and references on web development technologies.
- **CSS-Tricks -** For advanced CSS techniques and design patterns.
- **Google Fonts -** For web fonts used in the project design.

Certificates



In recognition of the commitment to achieve professional excellence

Mohammed Zubair A

Has successfully satisfied the requirements for:

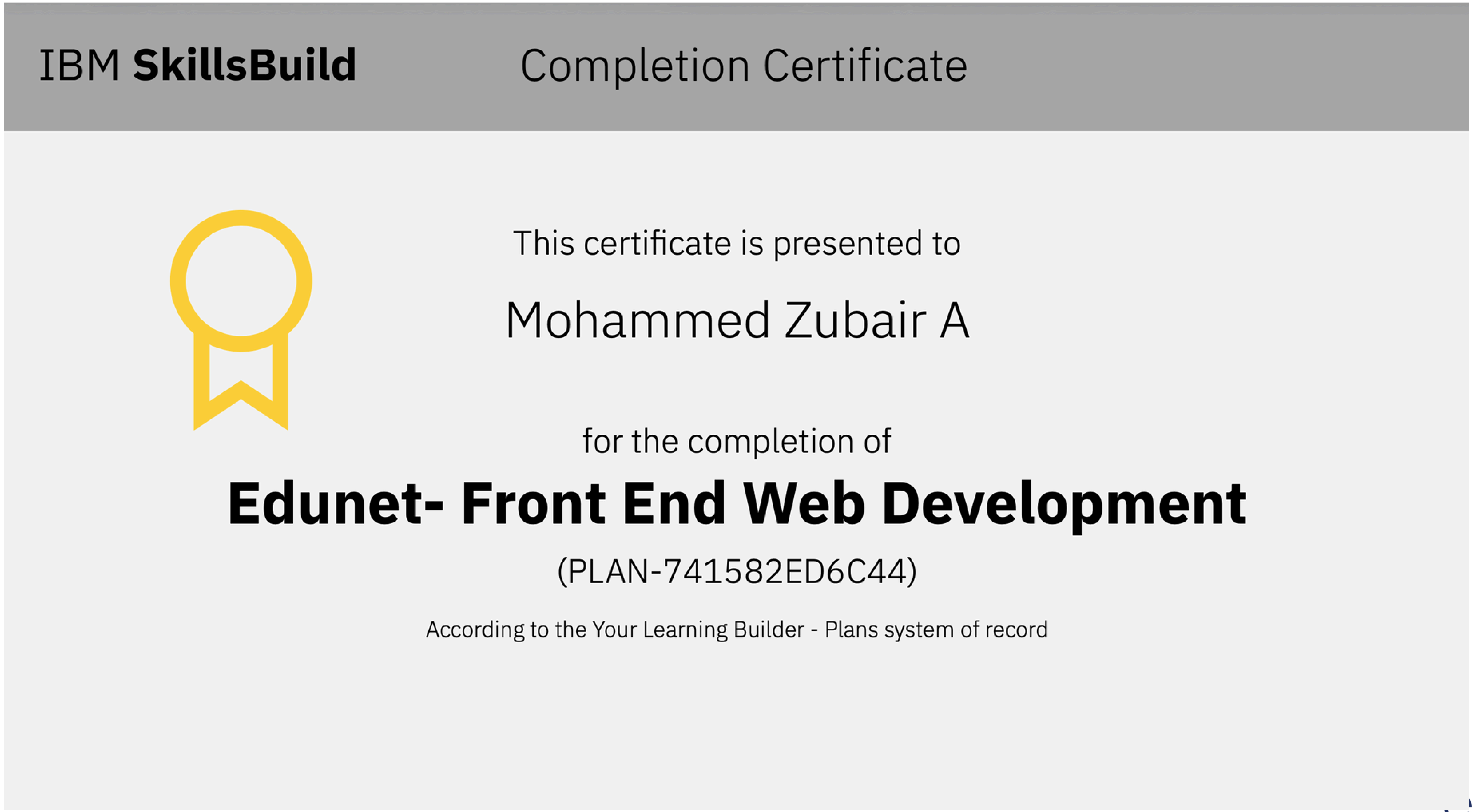
Web Development Fundamentals

Issued on: Oct 05, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/1618e9f6-6c04-4657-ae2b-f7e265da9609

Web Development Fundamentals
IBM SkillsBuild
IBM

Certificates



IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Mohammed Zubair A

for the completion of

Edunet- Front End Web Development

(PLAN-741582ED6C44)

According to the Your Learning Builder - Plans system of record

THANK YOU

edunet
foundation