HTML:  
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width,initial-scale=1" />

<title>Smart Study Planner</title>

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap" rel="stylesheet">

<link rel="stylesheet" href="style.css">

<link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@800&display=swap" rel="stylesheet">

</head>

<body>

<header>

<h1>

<span class="icon">📚</span>

<span class="gradient-text">Smart Study Planner</span>

</h1>

</header>

<main>

<!-- Left column: Form -->

<section id="task-form">

<h2 id="formTitle">Add New Task</h2>

<form id="form">

<input type="text" id="taskName" placeholder="Task name" required>

<input type="text" id="subject" placeholder="Subject" required>

<input type="date" id="dueDate">

<select id="priority">

<option value="low">Low Priority</option>

<option value="medium">Medium Priority</option>

<option value="high">High Priority</option>

</select>

<textarea id="notes" placeholder="Notes (optional)"></textarea>

<button type="submit" id="submitBtn">Add Task</button>

</form>

</section>

<!-- Right column: Tasks -->

<section id="task-list">

<h2>Your Tasks</h2>

<ul id="tasks"></ul>

</section>

<!-- Completed Tasks -->

<section id="completed-list">

<h2>Completed Tasks</h2>

<ul id="completedTasks"></ul>

</section>

<!-- Progress (below form, half width) -->

<section id="progress">

<h2>Progress</h2>

<div class="progress-bar">

<div id="progress-fill"></div>

</div>

<p id="progress-text">0% Completed</p>

</section>

</main>

<!-- Delete Confirmation Modal -->

<div id="deleteModal" aria-hidden="true">

<div class="modal-content" role="dialog" aria-modal="true" aria-labelledby="modalTitle">

<h3 id="modalTitle">Delete Task</h3>

<p>Are you sure you want to delete this task? This cannot be undone.</p>

<div class="modal-actions">

<button id="confirmDelete">Yes, Delete</button>

<button id="cancelDelete">Cancel</button>

</div>

</div>

</div>

<script src="script.js"></script>

</body>

</html>  
  
  
**CSS:**  
/\* ===== RESET & GLOBAL ===== \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: "Poppins", sans-serif;

}

body {

background: linear-gradient(135deg, #74ebd5 0%, #9face6 100%);

color: #333;

min-height: 100vh;

display: flex;

flex-direction: column;

align-items: center;

padding: 24px;

}

/\* ===== HEADER ===== \*/

header h1 {

font-family: "Montserrat", sans-serif;

font-size: clamp(1.5rem, 4vw, 5rem);

/\* smaller min size to fit small screens \*/

font-weight: 800;

text-align: center;

letter-spacing: 1px;

color: #22313f;

/\* fallback for emoji \*/

white-space: nowrap;

/\* prevent wrapping into multiple lines \*/

margin-bottom: 10px;

}

header h1 .gradient-text {

background: linear-gradient(90deg, #ff6ec7, #6a82fb, #00f260);

background-size: 300% 300%;

-webkit-background-clip: text;

-webkit-text-fill-color: transparent;

animation: gradientShift 6s ease infinite;

}

@keyframes gradientShift {

0% {

background-position: 0% 50%;

}

50% {

background-position: 100% 50%;

}

100% {

background-position: 0% 50%;

}

}

/\* Extra small devices \*/

@media (max-width: 400px) {

header h1 {

font-size: 1.4rem;

/\* further reduce font size \*/

}

}

/\* ===== MAIN GRID ===== \*/

main {

width: 100%;

max-width: 1100px;

display: grid;

grid-template-columns: 1fr 1fr;

gap: 24px;

align-items: start;

grid-template-areas:

"progress progress"

"form tasks"

"completed tasks";

/\* optional if you want completed tasks below form \*/

}

/\* Assign grid areas \*/

#task-form {

grid-area: form;

}

#task-list {

grid-area: tasks;

}

#progress {

grid-area: progress;

}

/\* ===== SECTIONS (cards) ===== \*/

section {

background: #fff;

padding: 18px;

border-radius: 14px;

box-shadow: 0 6px 18px rgba(0, 0, 0, 0.08);

transition: transform .18s;

}

section:hover {

transform: translateY(-3px);

}

h2 {

margin-bottom: 12px;

font-size: 1.15rem;

color: #274156;

}

/\* ===== FORM ELEMENTS ===== \*/

form input,

form select,

form textarea,

form button {

width: 100%;

margin: 10px 0;

padding: 12px 14px;

border-radius: 10px;

border: 1px solid #d7dbe6;

font-size: 0.95rem;

transition: box-shadow .18s, border-color .18s;

}

form input:focus,

form select:focus,

form textarea:focus {

outline: none;

border-color: #6c5ce7;

box-shadow: 0 4px 12px rgba(108, 92, 231, 0.08);

}

textarea {

min-height: 84px;

resize: vertical;

}

form button {

margin-top: 8px;

background: linear-gradient(90deg, #6c5ce7, #7b6ef0);

color: #fff;

border: none;

font-weight: 600;

padding: 12px;

border-radius: 12px;

cursor: pointer;

}

form button:hover {

transform: translateY(-2px);

}

/\* ===== TASK LIST ===== \*/

#tasks {

list-style: none;

padding-left: 0;

margin: 0;

}

.task-card {

background: #fff;

border-radius: 10px;

padding: 15px;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

margin-bottom: 20px;

/\* ✅ Space between cards \*/

}

.task-card h3 {

margin-bottom: 6px;

font-size: 1rem;

color: #203040;

}

.task-card p {

font-size: 0.88rem;

color: #4b5866;

margin: 4px 0;

}

/\* task action buttons row \*/

.task-actions {

margin-top: 10px;

display: flex;

gap: 10px;

}

.task-actions button {

flex: 1;

padding: 8px 10px;

border-radius: 8px;

border: none;

cursor: pointer;

font-size: 0.85rem;

}

#completed-list {

grid-column: 1;

/\* full width below both columns \*/

margin-top: 20px;

}

#completedTasks {

list-style: none;

padding-left: 0;

margin: 0;

}

#completedTasks .task-card {

opacity: 0.75;

text-decoration: line-through;

background: #f0f9f0;

}

/\* individual button styles \*/

.complete-btn {

background: #00b894;

color: #fff;

}

.complete-btn:hover {

filter: brightness(.95);

}

.edit-btn {

background: #6c5ce7;

color: #fff;

}

.delete-btn {

background: #e74c3c;

color: #fff;

}

/\* ===== PRIORITY COLORS ===== \*/

.priority-high {

border-left: 6px solid #e74c3c;

background: #fff5f5;

}

.priority-medium {

border-left: 6px solid #f39c12;

background: #fffaf2;

}

.priority-low {

border-left: 6px solid #27ae60;

background: #f4fff7;

}

/\* ===== PROGRESS BAR ===== \*/

.progress-bar {

width: 100%;

height: 22px;

background: #e5e9ef;

border-radius: 12px;

overflow: hidden;

margin-bottom: 8px;

}

#progress-fill {

height: 100%;

width: 0;

background: linear-gradient(90deg, #6c5ce7, #00cec9);

transition: width .45s ease;

}

#progress-text {

font-weight: 600;

color: #22313f;

}

/\* ===== MODAL ===== \*/

#deleteModal {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background: rgba(0, 0, 0, 0.48);

display: none;

justify-content: center;

align-items: center;

z-index: 9999;

}

#deleteModal.show {

display: flex;

}

#deleteModal .modal-content {

width: 92%;

max-width: 420px;

background: #fff;

padding: 18px;

border-radius: 12px;

box-shadow: 0 8px 30px rgba(0, 0, 0, 0.18);

text-align: center;

}

#deleteModal .modal-actions {

margin-top: 14px;

display: flex;

gap: 12px;

justify-content: center;

}

#deleteModal button {

padding: 10px 14px;

border-radius: 8px;

border: none;

cursor: pointer;

}

#confirmDelete {

background: #e74c3c;

color: #fff;

}

#cancelDelete {

background: #6c757d;

color: #fff;

}

/\* ===== RESPONSIVE ===== \*/

@media (max-width:900px) {

main {

grid-template-columns: 1fr;

grid-template-areas:

"form"

"progress"

"tasks";

}

}

**JS:**  
// ===== SELECT ELEMENTS =====

const form = document.getElementById("form");

const taskName = document.getElementById("taskName");

const subject = document.getElementById("subject");

const dueDate = document.getElementById("dueDate");

const priority = document.getElementById("priority");

const notes = document.getElementById("notes");

const tasksList = document.getElementById("tasks");

const completedList = document.getElementById("completedTasks");

const progressFill = document.getElementById("progress-fill");

const progressText = document.getElementById("progress-text");

const formTitle = document.getElementById("formTitle");

const submitBtn = document.getElementById("submitBtn");

// Modal Elements

const deleteModal = document.getElementById("deleteModal");

const confirmDeleteBtn = document.getElementById("confirmDelete");

const cancelDeleteBtn = document.getElementById("cancelDelete");

// ===== STATE =====

let tasks = JSON.parse(localStorage.getItem("tasks")) || [];

let editTaskId = null;

let taskToDelete = null;

// ===== INITIAL RENDER =====

renderTasks();

// ===== ADD / EDIT TASK =====

form.addEventListener("submit", (e) => {

e.preventDefault();

const name = taskName.value.trim();

if (!name) {

alert("Please enter a task name.");

return;

}

if (editTaskId) {

tasks = tasks.map(task =>

task.id === editTaskId ? {

...task,

name,

subject: subject.value.trim(),

dueDate: dueDate.value,

priority: priority.value,

notes: notes.value.trim()

} :

task

);

editTaskId = null;

formTitle.textContent = "Add New Task";

submitBtn.textContent = "Add Task";

} else {

const task = {

id: Date.now(),

name,

subject: subject.value.trim(),

dueDate: dueDate.value,

priority: priority.value,

notes: notes.value.trim(),

completed: false,

};

tasks.push(task);

}

saveTasks();

renderTasks();

form.reset();

});

// ===== SAVE TO LOCAL STORAGE =====

function saveTasks() {

localStorage.setItem("tasks", JSON.stringify(tasks));

}

// ===== RENDER TASKS =====

function renderTasks() {

tasksList.innerHTML = "";

completedList.innerHTML = "";

if (tasks.length === 0) {

tasksList.innerHTML = "<p>No tasks yet. Add one!</p>";

updateProgress();

return;

}

tasks.forEach(task => {

const li = document.createElement("li");

li.classList.add("task-card", `priority-${task.priority}`);

const dueText = task.dueDate ? formatDate(task.dueDate) : "No due date";

li.innerHTML = `

<h3>${escapeHtml(task.name)}</h3>

<p><b>Subject:</b> ${escapeHtml(task.subject || "")}</p>

<p><b>Due:</b> ${escapeHtml(dueText)}</p>

<p><b>Priority:</b> <span style="text-transform:capitalize">${escapeHtml(task.priority)}</span></p>

<p>${escapeHtml(task.notes || "")}</p>

<div class="task-actions">

<button class="complete-btn" onclick="toggleComplete(${task.id})">

${task.completed ? "✅ Completed" : "✔ Mark Complete"}

</button>

<button class="edit-btn" onclick="editTask(${task.id})">✏ Edit</button>

<button class="delete-btn" onclick="confirmDelete(${task.id})">🗑 Delete</button>

</div>

`;

if (task.completed) {

li.style.opacity = "0.7";

li.style.textDecoration = "line-through";

completedList.appendChild(li);

} else {

tasksList.appendChild(li);

}

});

updateProgress();

}

// ===== TOGGLE COMPLETE =====

function toggleComplete(id) {

tasks = tasks.map(task => (task.id === id ? {

...task,

completed: !task.completed

} : task));

saveTasks();

renderTasks();

}

// ===== EDIT TASK =====

function editTask(id) {

const task = tasks.find(t => t.id === id);

if (!task) return;

taskName.value = task.name;

subject.value = task.subject || "";

dueDate.value = task.dueDate || "";

priority.value = task.priority || "low";

notes.value = task.notes || "";

editTaskId = id;

formTitle.textContent = "Edit Task";

submitBtn.textContent = "Update Task";

taskName.scrollIntoView({

behavior: "smooth",

block: "center"

});

}

// ===== DELETE (Modal) =====

function confirmDelete(id) {

taskToDelete = id;

deleteModal.classList.add("show");

document.body.style.overflow = "hidden";

}

confirmDeleteBtn.addEventListener("click", () => {

if (taskToDelete) {

tasks = tasks.filter(task => task.id !== taskToDelete);

saveTasks();

renderTasks();

taskToDelete = null;

}

deleteModal.classList.remove("show");

document.body.style.overflow = "";

});

cancelDeleteBtn.addEventListener("click", () => {

taskToDelete = null;

deleteModal.classList.remove("show");

document.body.style.overflow = "";

});

window.addEventListener("click", (e) => {

if (e.target === deleteModal) {

taskToDelete = null;

deleteModal.classList.remove("show");

document.body.style.overflow = "";

}

});

// ===== UPDATE PROGRESS =====

function updateProgress() {

const total = tasks.length;

const completed = tasks.filter(t => t.completed).length;

const percent = total === 0 ? 0 : Math.round((completed / total) \* 100);

progressFill.style.width = percent + "%";

progressText.textContent = `${percent}% Completed`;

}

// ===== HELPERS =====

function formatDate(iso) {

try {

if (!iso) return "";

const d = new Date(iso + "T00:00:00");

return d.toLocaleDateString(undefined, {

year: "numeric",

month: "short",

day: "numeric"

});

} catch {

return iso;

}

}

function escapeHtml(str) {

if (!str) return "";

return String(str).replaceAll("&", "&amp;").replaceAll("<", "&lt;").replaceAll(">", "&gt;");

}