TASK SCHEDULER PROJECT:
HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Task Scheduler</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: linear-gradient(120deg, #f6d365, #fda085);
      margin: 0;
      padding: 20px;
      color: #333;
    }

    h1 {
      text-align: center;
      color: #fff;
    }

    .task-container {
      max-width: 800px;
      margin: 0 auto;
      background: #fff;
      border-radius: 10px;
      padding: 20px;
      box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
    }

    .input-group {
      display: flex;
      flex-direction: column;
      margin-bottom: 15px;
    }

    .input-group label {
      margin-bottom: 5px;
    }

    .input-group input, .input-group select, .input-group button {
      padding: 10px;
      border: 1px solid #ddd;
      border-radius: 5px;
    }
```

```
.input-group button {
    background: #f6d365;
    color: #fff;
    border: none;
    cursor: pointer;
}

.input-group button:hover {
    background: #fda085;
}

.tasks {
    margin-top: 20px;
}

.task {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    margin-bottom: 10px;
    background-color: #f9f9f9;
}

.task.completed {
    background-color: #d4edda;
    text-decoration: line-through;
}

.task button {
    margin-left: 10px;
    padding: 5px 10px;
    border: none;
    border-radius: 5px;
    background: #ff6b6b;
    color: #fff;
    cursor: pointer;
}

.task button:hover {
    background: #ff4b4b;
}

.filter-group {
    margin-bottom: 20px;
```

```
            display: flex;
            justify-content: space-between;
        }

        .filter-group select {
            padding: 10px;
            border: 1px solid #ddd;
            border-radius: 5px;
        }
    </style>
</head>
<body>
    <h1>Task Scheduler</h1>
    <div class="task-container">
        <div class="input-group">
            <label for="taskName">Task Name:</label>
            <input type="text" id="taskName" placeholder="Enter task name">
        </div>
        <div class="input-group">
            <label for="dueDate">Due Date:</label>
            <input type="date" id="dueDate">
        </div>
        <div class="input-group">
            <label for="priority">Priority:</label>
            <select id="priority">
                <option value="High">High</option>
                <option value="Medium">Medium</option>
                <option value="Low">Low</option>
            </select>
        </div>
        <div class="input-group">
            <button onclick="addTaskUI()">Add Task</button>
        </div>

        <div class="filter-group">
            <select id="filterCriteria" onchange="filterTasks()">
                <option value="all">All Tasks</option>
                <option value="completed">Completed</option>
                <option value="notCompleted">Not Completed</option>
            </select>
        </div>

        <div class="tasks" id="tasks"></div>
    </div>

    <script src="scripttt.js"></script>
</body>
```

```
</html>
```

JAVSCRIPT CODE:

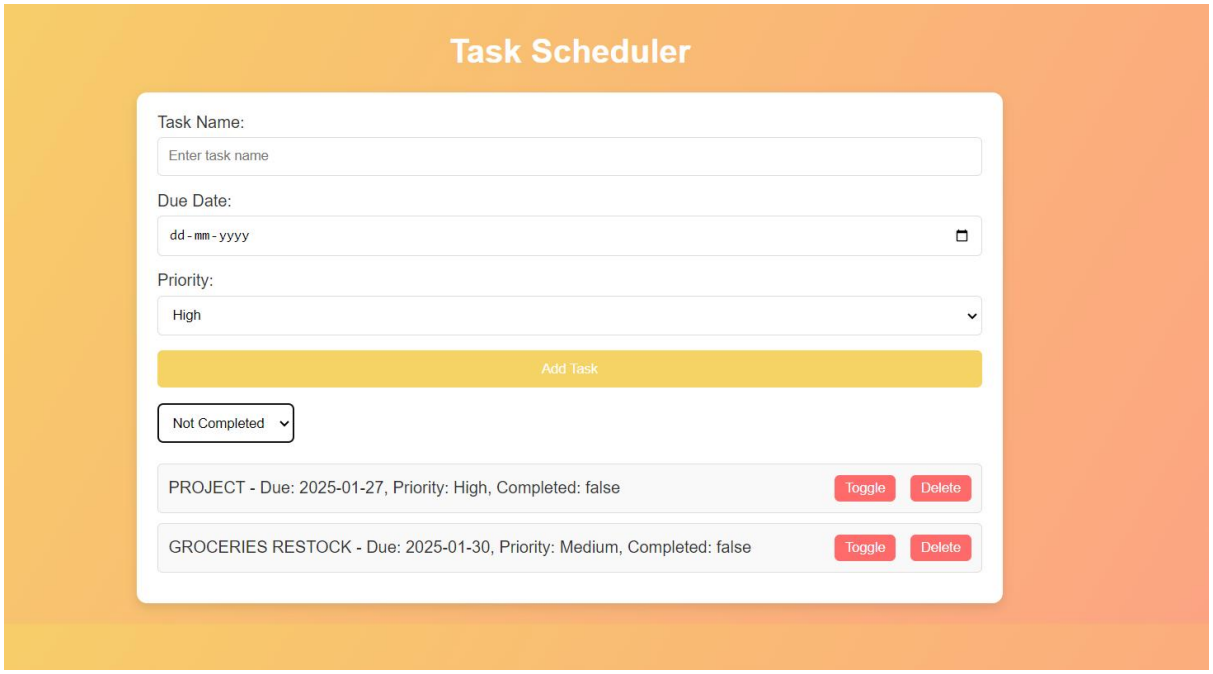
```
class Task {
  constructor(taskName, dueDate, priority) {
    this.taskName = taskName;
    this.dueDate = dueDate;
    this.priority = priority;
    this.completed = false;
  }

  getTaskDetail() {
    return `${this.taskName} - Due: ${this.dueDate}, Priority: ${this.priority}, Completed:
${this.completed}`;
  }

  toggleCompletion() {
    this.completed = !this.completed;
  }
}
let taskList = [];
function addTaskUI() {
  const taskName = document.getElementById('taskName').value.trim();
  const dueDate = document.getElementById('dueDate').value;
  const priority = document.getElementById('priority').value;

  if (!taskName || !dueDate) {
    alert('Please fill out all fields before adding a task.');
    return;
  }

  const task = new Task(taskName, dueDate, priority);
  taskList.push(task);
  renderTasks();
  saveTasks();
  clearInputs();
}
function renderTasks(filter = 'all') {
  const tasksContainer = document.getElementById('tasks');
  tasksContainer.innerHTML = ''; // Clear existing tasks

  const filteredTasks = taskList.filter((task) => {
    if (filter === 'completed') return task.completed;
    if (filter === 'notCompleted') return !task.completed;
    return true;
  });
```

```
    filteredTasks.forEach((task, index) => {
        const taskDiv = document.createElement('div');
        taskDiv.className = `task ${task.completed ? 'completed' : ''}`;
        taskDiv.innerHTML = `
            <span>${task.getTaskDetail()}</span>
            <div>
                <button onclick="toggleTaskCompletion(${index})">Toggle</button>
                <button onclick="deleteTask(${index})">Delete</button>
            </div>
        `;
        tasksContainer.appendChild(taskDiv);
    });
}
function toggleTaskCompletion(index) {
    taskList[index].toggleCompletion();
    renderTasks();
    saveTasks();
}
function deleteTask(index) {
    taskList.splice(index, 1);
    renderTasks();
    saveTasks();
}
function filterTasks() {
    const filterCriteria = document.getElementById('filterCriteria').value;
    renderTasks(filterCriteria);
}
function clearInputs() {
    document.getElementById('taskName').value = '';
    document.getElementById('dueDate').value = '';
    document.getElementById('priority').value = 'High';
}
async function saveTasks() {
    const taskJSON = JSON.stringify(taskList);
    localStorage.setItem('tasks', taskJSON);
}
async function loadTasks() {
    const savedTasks = localStorage.getItem('tasks');
    if (savedTasks) {
        taskList = JSON.parse(savedTasks).map((task) => {
            const { taskName, dueDate, priority, completed } = task;
            const newTask = new Task(taskName, dueDate, priority);
            newTask.completed = completed;
            return newTask;
        });
        renderTasks();
```

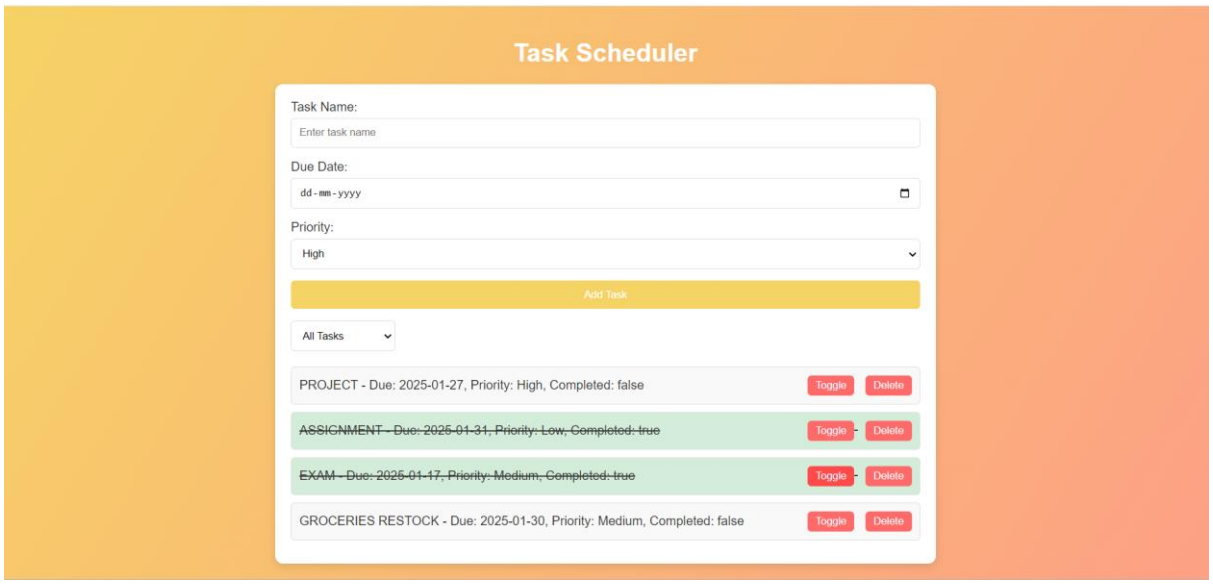
```
    }
}
window.onload = loadTasks;
```

OUTPUT:

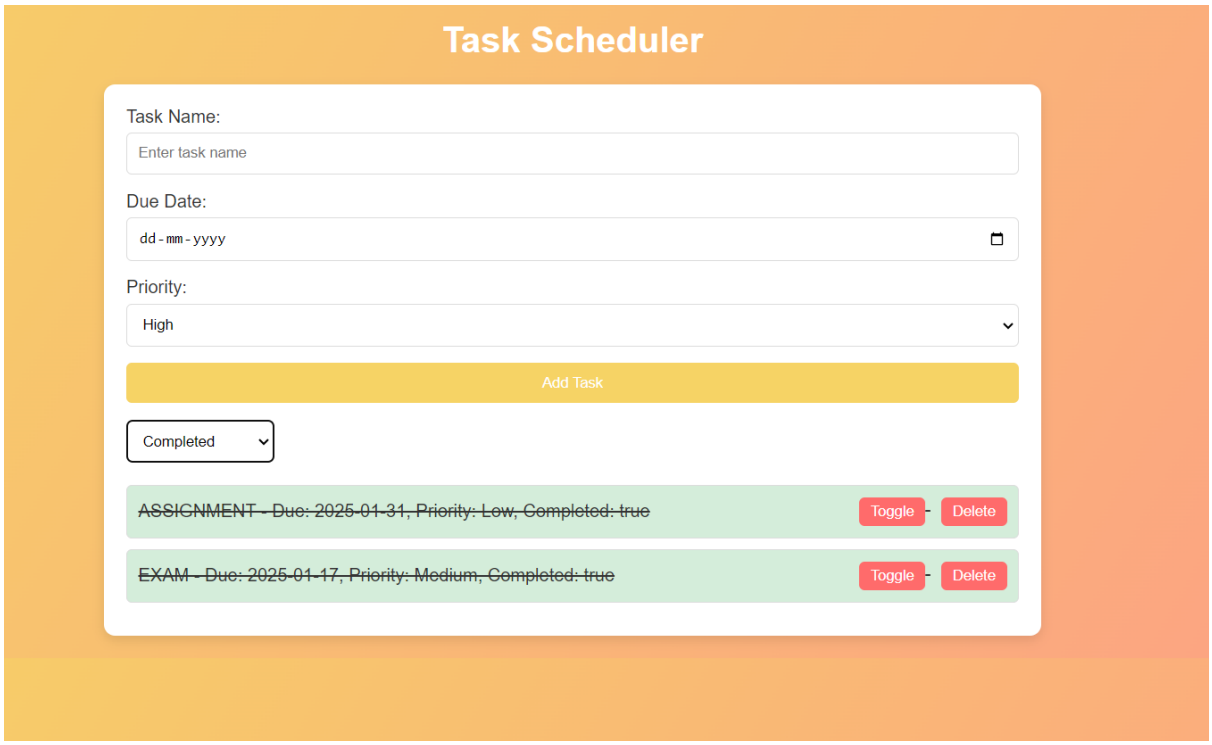
ADDING TASKS



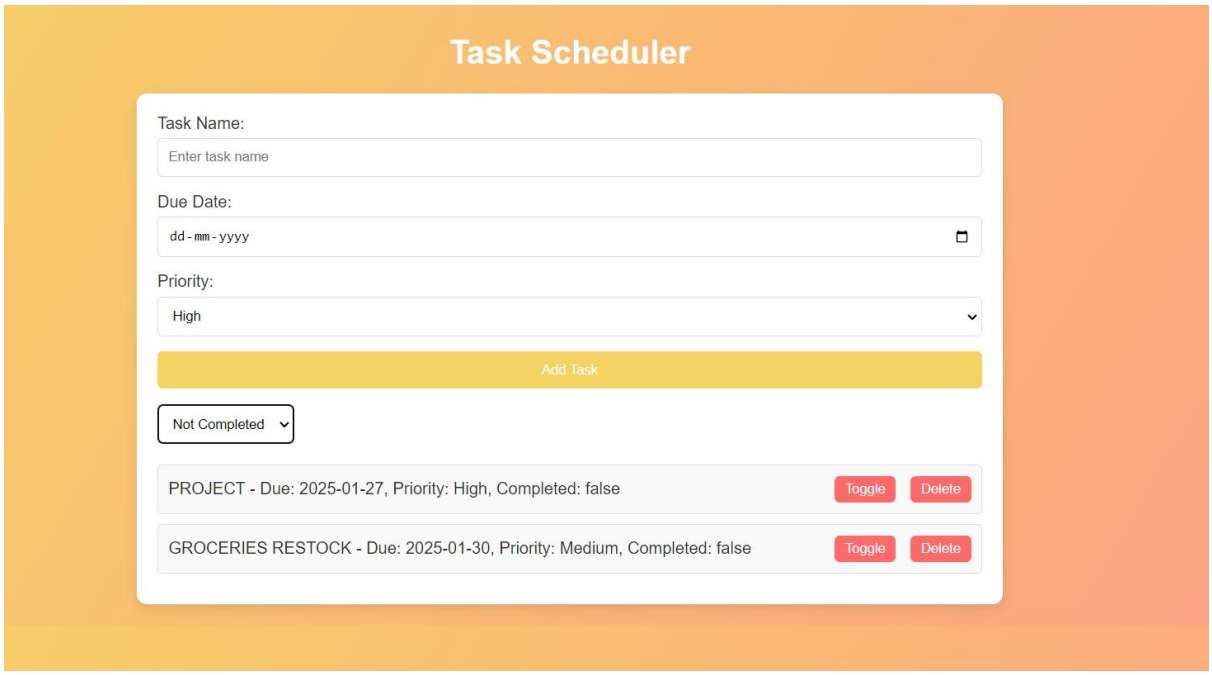
IF COMPLETED TOGGLING IT:



FILTERING COMPLETED TASKS:



FILTERING INCOMPLETED TASKS:



DELETING COMPLETED TASKS: