**React Hooks Assignment**

**Objective**

This assignment is designed to help students understand the **purpose and practical applications** of React Hooks. By completing hands-on tasks, students will gain experience working with the following major hooks:

* useState
* useEffect
* useContext
* useReducer
* useRef
* useMemo
* useCallback

The final project will be a **Task Manager App**, incorporating multiple hooks in a meaningful way to encourage best practices and performance optimizations.

**Project Overview: Task Manager App**

Students will build a **Task Manager App** that allows users to **add, edit, delete, and filter tasks**, while ensuring performance efficiency using React Hooks. The app will also include a **dark mode toggle** and **real-time task-saving feature**.

**Features:**

* Add, update, and delete tasks
* Mark tasks as completed
* Filter tasks based on status
* Dark mode toggle using useContext
* Autosave feature using useEffect
* Optimized rendering using useMemo and useCallback
* State management using useReducer

**Assignment Instructions**

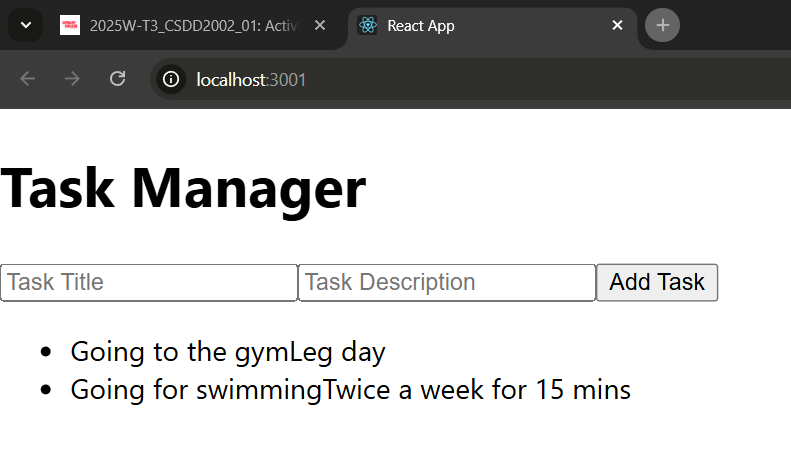
**1. useState: Managing Task State**

**Task:**

* Implement a form that allows users to add tasks with a **title** and **description**.
* Use useState to manage the list of tasks.

**Real-world Use Cases:**

1. Handling form inputs dynamically.
2. Managing UI state like modal visibility.
3. Toggling dark/light mode.



**2. useEffect: Side Effects and Data Persistence**

**Task:**

* Implement an **autosave** feature that saves tasks to localStorage whenever a new task is added or removed.
* Display an alert when the task list is updated.

A screenshot of a computer

AI-generated content may be incorrect.

**Real-world Use Cases:**

1. Fetching data from APIs when a component mounts.
2. Listening to window resize events.
3. Syncing state with local storage or a database.

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

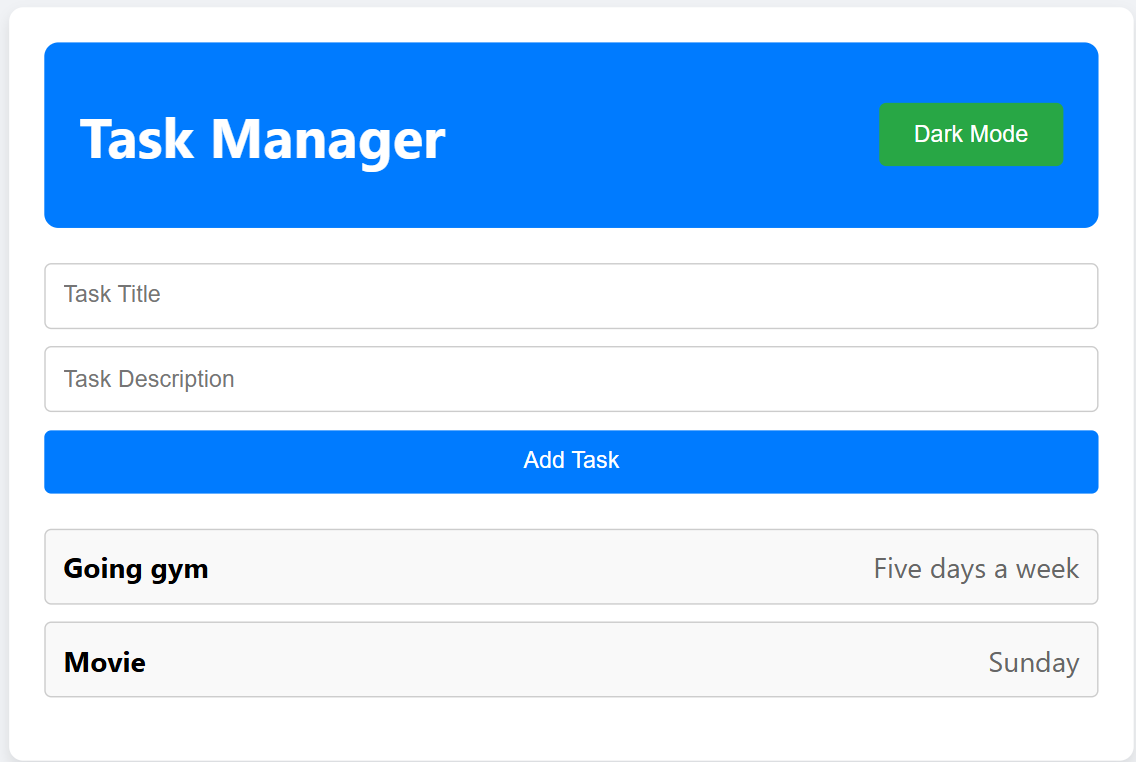
**3. useContext: Theme Toggle (Dark Mode)**

**Task:**

* Implement a **dark mode toggle** using useContext.
* The theme should persist across page reloads.

**Real-world Use Cases:**

1. Managing global themes (dark/light mode).
2. Sharing user authentication state across components.
3. Managing language preferences in multi-lingual apps.

A screenshot of a computer

AI-generated content may be incorrect.

**4. useReducer: Complex State Management**

**Task:**

* Replace useState with useReducer to manage tasks.
* Implement actions for adding, updating, and deleting tasks.

**Real-world Use Cases:**

1. Managing complex state logic (shopping cart, forms with multiple fields).
2. Handling undo/redo functionality.
3. Managing authentication state.

A computer code on a dark background

AI-generated content may be incorrect.

A computer screen shot of text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**5. useRef: Managing References and Performance Optimization**

**Task:**

* Use useRef to store a reference to the task input field.
* Auto-focus the input field when the component mounts.

**Real-world Use Cases:**

1. Storing references to DOM elements (focus management, input fields).
2. Tracking previous values without triggering re-renders.
3. Controlling animations.

wA screenshot of a computer

AI-generated content may be incorrect.

**6. useMemo: Performance Optimization**

**Task:**

* Optimize task filtering performance using useMemo.
* Prevent unnecessary re-renders when filtering tasks.

**Real-world Use Cases:**

1. Optimizing expensive computations (filtering, sorting large datasets).
2. Preventing unnecessary re-renders in performance-critical components.
3. Caching API response transformations.

**Filtering all tasks:**

A screenshot of a task manager

AI-generated content may be incorrect.

**Filtering completed tasks:**

A screenshot of a computer

AI-generated content may be incorrect.

**Incomplete Tasks:**

A screenshot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a program code

AI-generated content may be incorrect.

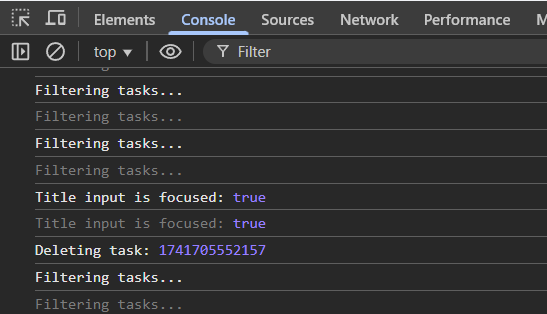
**7. useCallback: Memoizing Functions**

**Task:**

* Use useCallback to optimize the onClick event handlers for updating and deleting tasks.
* Prevent functions from being re-created on every render.

**Real-world Use Cases:**

1. Preventing unnecessary re-renders in child components.
2. Optimizing event handlers in large applications.
3. Reducing memory usage in frequently re-rendered components.



**Expected Outcomes**

By the end of this assignment, students should be able to:

* Implement all major React Hooks in a practical project.
* Understand the difference between useState and useReducer.
* Optimize component performance using useMemo and useCallback.
* Manage global state using useContext.
* Work with side effects using useEffect.

**Optional Challenges (Advanced)**

1. **Drag-and-Drop Feature**: Implement task reordering using React DnD.
2. **API Integration**: Save and fetch tasks from a backend API.
3. **Animations**: Add animations using framer-motion.
4. **Undo/Redo Functionality**: Implement undo/redo for task actions.

**Submission Guidelines**

* Submit a GitHub repository with completed code.
* Include a short README explaining your implementation.
* Deploy the project on Vercel or Netlify and share the live link.