**Task Management Application Documentation**

**Introduction**

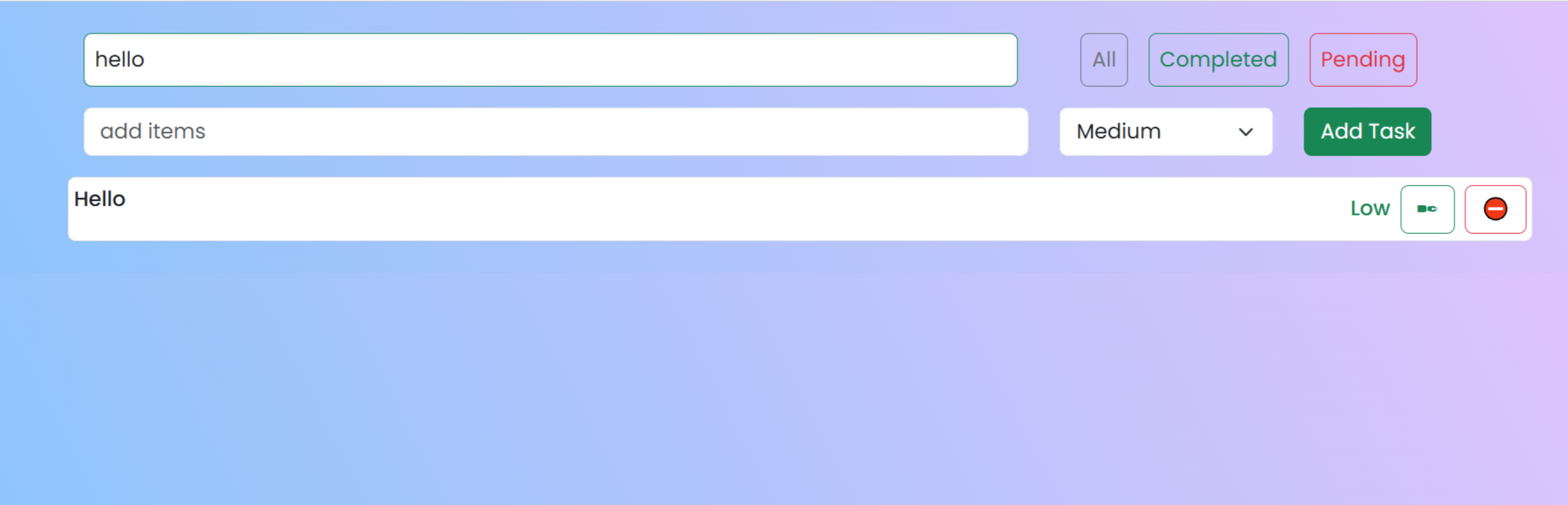
This project is a Task Management Application built using **React**. It allows users to:

* Add tasks with priority levels.
* Edit tasks.
* Delete tasks.
* Search tasks by name.
* Filter tasks by status (All, Completed, Pending).
* Highlight tasks when selected.

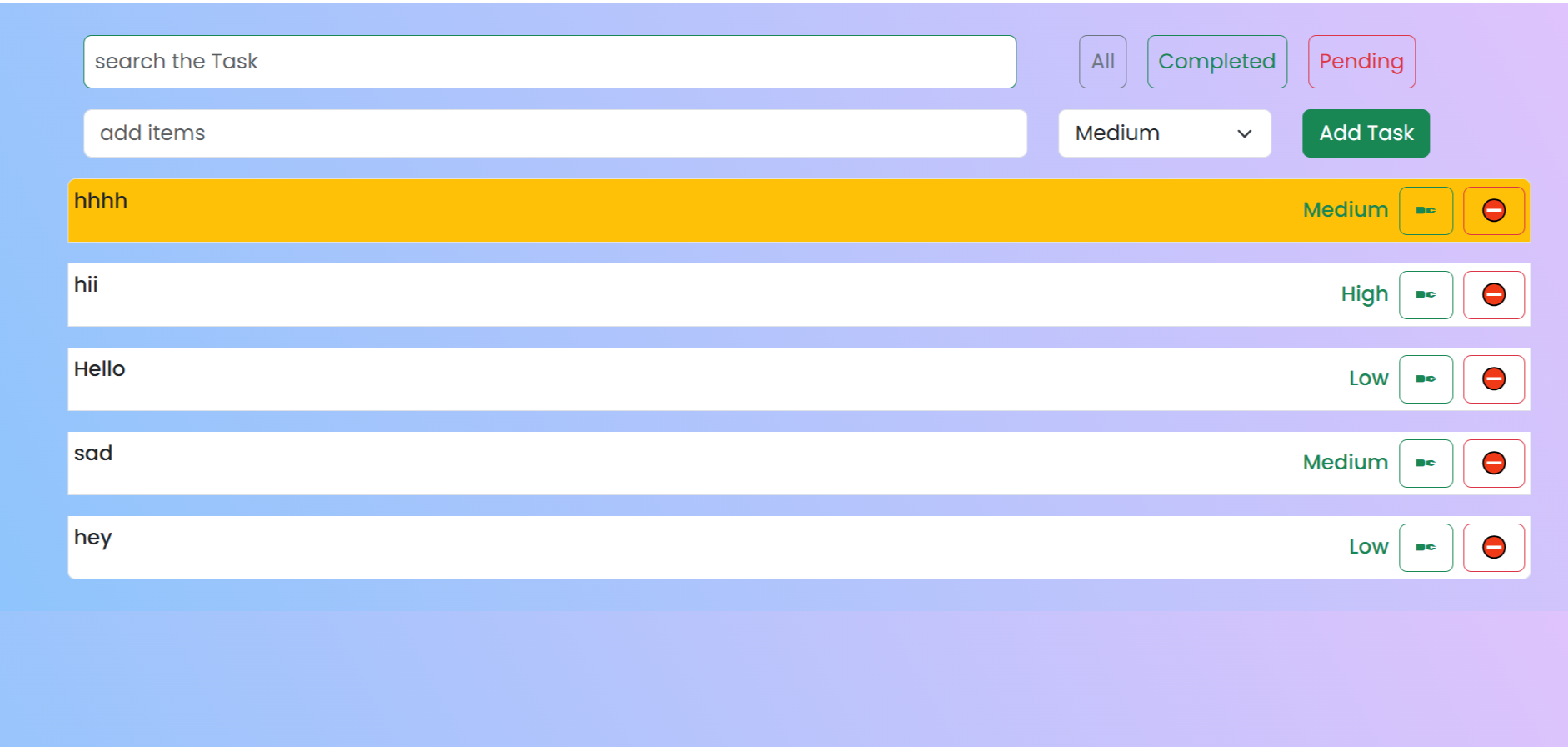
This application uses a REST API to fetch and persist tasks, ensuring that tasks are stored and retrieved from the backend.

**Features**

1. **Task CRUD Operations:**
   * Create new tasks with a description and priority level.
   * Edit existing tasks (update their description and priority).
   * Delete tasks from the task list.
2. **Search Functionality:**
   * Users can search for specific tasks by entering keywords in real time.



1. **Filter Functionality:**
   * Filter tasks by:
     + **All** : Displays all tasks.
     + **Completed** : Shows only completed tasks tasks.
     + **Pending** : Shows tasks that are still pending.
2. **UI Highlights:**
   * Highlight a selected task for better visualization.



1. **Toast Notifications:**
   * Real-time feedback (success or error) for actions like adding, editing, or deleting tasks.

**Technologies Used**

**Frontend**

* **React.js:** Main library for building the user interface.
* **React Hooks:** useState and useEffect for managing state and lifecycle methods.
* **Bootstrap:** Used for responsive and styled UI components.
* **React-Toastify:** Provides toast notifications for user actions.
* **Axios:** For API communication (fetching, posting, updating, and deleting data).

**Backend**

* A REST API is used to handle tasks, with endpoints for GET, POST, PUT, and DELETE requests.

**Project Structure**

The project is divided into modular components for scalability and reusability:

**1. Components**

1. **TaskManagement.js(Parent Component):**
   * The main parent component that combines all other components.
   * Manages the state of the application.
   * Fetches tasks from the API.
   * Handles logic for adding, editing, deleting, searching, and filtering tasks.
2. **TaskItem.js (Individual Task):**
   * Represents an individual task item.
   * Handles task editing, deleting, and highlighting logic.
3. **TaskForm.jsx**
   * Displays the form for adding new tasks.
   * Allows the user to input a task description and select a priority.
4. **SearchBar.jsx**
   * Provides a search bar for filtering tasks by name in real time.
5. **TaskFilter.jsx**
   * Provides buttons to filter tasks by their status (All, Completed, Pending).
6. **Api.js**
   * Contains the Axios instance for communicating with the REST API.

**How It Works**

Give a walkthrough of the application's functionality:

1. **Fetching Tasks:**
   * "When the application loads, it fetches all tasks from the backend using the GET /tasks endpoint and stores them in the state using useEffect."
2. **Adding Tasks:**
   * "Users can add new tasks using the form. The addTask function sends the task data to the backend using the POST /tasks endpoint. On success, the new task is added to the local state and displayed in the UI."
3. **Editing Tasks:**
   * "Users can edit a task by clicking the edit button. The updated data is sent to the backend using the PUT /tasks/:id endpoint, and the state is updated locally to reflect the changes."
4. **Deleting Tasks:**
   * "Tasks can be deleted by clicking the delete button. The DELETE /tasks/:id endpoint is called, and the task is removed from both the backend and the local state."
5. **Search and Filter:**
   * "Tasks can be searched by name in real time using the handleSearch function. Filtering is done by status (All, Completed, Pending) using the handleFilter function."

**Future Enhancements**

Suggest improvements to the application:

* "We can add due dates for tasks and sort tasks by priority or due date."
* "Implement user authentication to allow multiple users to manage their tasks."
* "Introduce drag-and-drop functionality to reorder tasks based on priority."
* "Use a progress bar to show overall task completion."

**Final Output**

