

Program:

Dashboard.js

```
import React, { useState } from "react";

import "./Dashboard.css";

const StatusColors = {

  Pending: "#ff5252",

  InProgress: "#ffc107",

  Completed: "#4caf50",

};

const App3 = () => {

  const [tasks, setTasks] = useState([

    { id: 1, title: "Task 1", status: "Pending" },

    { id: 2, title: "Task 2", status: "InProgress" },

    { id: 3, title: "Task 3", status: "Completed" },

  ]);

  const [inputText, setInputText] = useState("");

  const [editingTaskId, setEditingTaskId] = useState(null);

  const handleStatusChange = (taskId, newStatus) => {

    const updatedTasks = tasks.map((task) =>

      task.id === taskId ? { ...task, status: newStatus } : task

    );

    setTasks(updatedTasks);

  };

  const handleTaskAdd = (newTaskTitle) => {

    const newTask = {

      id: tasks.length + 1,

      title: newTaskTitle,

      status: "Pending",

    };

    setTasks([...tasks, newTask]);

    setInputText("");

  };

};
```

```
const handleTaskEdit = (taskId, newTitle) => {  
  const updatedTasks = tasks.map((task) =>  
    task.id === taskId ? { ...task, title: newTitle } : task  
  );  
  setTasks(updatedTasks);  
  setEditingTaskId(null);  
};  
  
const handleTaskRemove = (taskId) => {  
  const updatedTasks = tasks.filter((task) => task.id !== taskId);  
  setTasks(updatedTasks);  
};  
  
return (  
  <div className="app-container">  
    <h1 className="app-title">Project Management Dashboard</h1>  
    <div className="task-list">  
      {tasks.map((task) => (  
        <div key={task.id} className="task">  
          <h3 onClick={() => setEditingTaskId(task.id)}>{task.title}</h3>  
          <div className="status-buttons">  
            <button  
              style={{ backgroundColor: StatusColors.Pending }}  
              onClick={() => handleStatusChange(task.id, "Pending")}  
            >  
              Pending  
            </button>  
            <button  
              style={{ backgroundColor: StatusColors.InProgress }}  
              onClick={() => handleStatusChange(task.id, "InProgress")}  
            >  
              In Progress
```

```

    </button>

    <button

      style={{ backgroundColor: StatusColors.Completed }}

      onClick={() => handleStatusChange(task.id, "Completed")}

    >

      Completed

    </button>

  </div>

  <button

    className="remove-task-button"

    onClick={() => handleTaskRemove(task.id)}

  >

    Remove Task

  </button>

</div>

)}}

</div>

<div className="add-task">

  <input

    type="text"

    placeholder="New Task Title"

    value={inputText}

    onChange={(e) => setInputText(e.target.value)}

  />

  <button onClick={() => handleTaskAdd(inputText)}>Add Task</button>

</div>

{editingTaskId !== null && (

  <div className="modal">

    <div className="modal-content">

      <h2>Edit Task</h2>

      <input

        type="lable"

        value={

```

```

      tasks.find((task) => task.id === editingTaskId)?.title || ""
    }
    onChange={(e) => setInputText(e.target.value)}
  />
  <div className="modal-buttons">
    <button onClick={() => setEditingTaskId(null)}>Cancel</button>
    <button
      onClick={() => {
        const editedTask = tasks.find(
          (task) => task.id === editingTaskId
        );
        if (editedTask) {
          handleTaskEdit(editingTaskId, inputText);
        }
      }}
    >
      Save
    </button>
  </div>
</div>
</div>
)}
</div>
);
};
export default App3;

```

Dashboard.css

```

body {
  margin: 0;
  padding: 0;
  font-family: Arial, sans-serif;
}
.app-container {

```

```
background-color: #f1f1f1;

padding: 20px;

}

.app-title{

color: #333;

margin-bottom: 20px;

text-align: center;

}

.task-list {

display: flex;

flex-wrap: wrap;

}

.task {

border: 1px solid #ccc;

background-color: #fff;

padding: 10px;

margin: 10px;

width: 400px;

border-radius: 8px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

}

/* ... (existing styles) ... */

.remove-task-button {

background-color: #f44336;

color: #fff;

border: none;

padding: 8px 16px;

cursor: pointer;

border-radius: 4px;

margin-top: 10px;

}

.remove-task-button:hover {

background-color: #d32f2f;
```

```
}

.remove-task-button:focus {

outline: none;

}

h3 {

margin-top: 0;

margin-bottom: 10px;

}

.status-buttons {

display: flex;

justify-content: space-between;

margin-top: 10px;

}

.status-buttons button {

color: #fff;

border: none;

padding: 5px 10px;

cursor: pointer;

border-radius: 4px;

}

.pending {

background-color: #ff5252;

}

.inprogress {

background-color: #ffc107;

}

.completed {

background-color: #4caf50;

}

.add-task {

display: flex;

margin-top: 20px;

align-items: center;
```

```
}  
  
.add-task input {  
padding: 8px;  
margin-right: 10px;  
border: 1px solid #ccc;  
border-radius: 4px;  
}  
  
.add-task button {  
background-color: #3f51b5;  
color: #fff;  
border: none;  
padding: 8px 16px;  
cursor: pointer;  
border-radius: 4px;  
}
```

OUTPUT :