**MiniProject Week-6**

**//TaskList**

**import React from "react";**

**import Task from "./Task";**

**function TaskList({ tasks, removeTask, selectTask }) { return (**

**<div>**

**<h2>Task List</h2>**

**{tasks.map((task) => (**

**<Task**

**key={task.id}**

**task={task}**

**removeTask={removeTask}**

**selectTask={selectTask}**

**/>**

**))}**

**</div>**

**);**

**}**

**export default TaskList;**

**//Task.js**

**import React from "react";**

**function Task({ task, removeTask, selectTask }) { const handleRemove = () => {**

**removeTask(task.id);**

**};**

**const handleSelect = () => {**

**selectTask(task);**

**};**

**return (**

**<div>**

**<h3>{task.title}</h3>**

**<p>{task.description}</p>**

**<button onClick={handleRemove}>Delete</button> <button onClick={handleSelect}>Edit</button> </div>**

**);**

**}**

**export default Task;**

**//AddForm.js**

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

**function AddForm({ addTask }) {**

**const [title, setTitle] = useState("");**

**const [description, setDescription] = useState("");**

**const handleSubmit = (e) => {**

**e.preventDefault();**

**const task = {**

**id: Math.floor(Math.random() \* 10000),**

**title,**

**description,**

**};**

**addTask(task);**

**setTitle("");**

**setDescription("");**

**};**

**return (**

**<div>**

**<h2>Add Task</h2>**

**<form onSubmit={handleSubmit}>**

**<label>**

**Title:**

**<input**

**type="text"**

**value={title}**

**onChange={(e) => setTitle(e.target.value)} />**

**</label>**

**<br />**

**<label>**

**Description:**

**<textarea**

**value={description}**

**onChange={(e) => setDescription(e.target.value)}**

**/>**

**</label>**

**<br />**

**<button type="submit">Add</button>**

**</form>**

**</div>**

**);**

**}**

**export default AddForm;**

**//UpdateTask**

**import React, { useState, useEffect } from "react";**

**function UpdateTask({ selectedTask, updateTask, setSelectedTask }) { const [title, setTitle] = useState("");**

**const [description, setDescription] = useState("");**

**useEffect(() => {**

**setTitle(selectedTask.title);**

**setDescription(selectedTask.description);**

**}, [selectedTask]);**

**const handleSubmit = (e) => {**

**e.preventDefault();**

**const updatedTask = {**

**id: selectedTask.id,**

**title,**

**description,**

**};**

**updateTask(selectedTask.id, updatedTask);**

**setSelectedTask(null);**

**};**

**return (**

**<div>**

**<h2>Update Task</h2>**

**<form onSubmit={handleSubmit}>**

**<label>**

**Title:**

**<input**

**type="text"**

**value={title}**

**onChange={(e) => setTitle(e.target.value)}**

**/>**

**</label>**

**<br />**

**<label>**

**Description:**

**<textarea**

**value={description}**

**onChange={(e) => setDescription(e.target.value)} />**

**</label>**

**<br />**

**<button type="submit">Update</button>**

**<button onClick={() => setSelectedTask(null)}>Cancel</button> </form>**

**</div>**

**);**

**}**

**export default UpdateTask;**

**//App.js**

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

**import TaskList from "./components/TaskList";**

**import AddForm from "./components/AddForm";**

**import UpdateTask from "./components/UpdateTask"; function App() {**

**const [tasks, setTasks] = useState([]);**

**const [selectedTask, setSelectedTask] = useState(null); const addTask = (task) => {**

**setTasks([...tasks, task]);**

**};**

**const removeTask = (id) => {**

**setTasks(tasks.filter((task) => task.id !== id));**

**};**

**const selectTask = (task) => {**

**setSelectedTask(task);**

**};**

**const updateTask = (id, updatedTask) => {**

**setTasks(tasks.map((task) => (task.id === id ? updatedTask : task))); setSelectedTask(null);**

**};**

**return (**

**<div>**

**<h1>Task Manager</h1>**

**<TaskList tasks={tasks} removeTask={removeTask} selectTask={selectTask} /> <AddForm addTask={addTask} />**

**{selectedTask && (**

**<UpdateTask**

**selectedTask={selectedTask}**

**updateTask={updateTask}**

**setSelectedTask={setSelectedTask}**

**/>**

**)}**

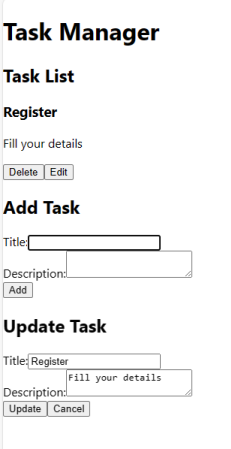
**</div>**

**);**

**}**

**export default App;**

**Output:**

****