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

import { DragDropContext, Droppable, Draggable } from "react-beautiful-dnd";

import "tailwindcss/tailwind.css";

const TaskManager = () => {

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

const [taskInput, setTaskInput] = useState("");

const [taskDescription, setTaskDescription] = useState("");

const [taskPriority, setTaskPriority] = useState("Low");

const [darkMode, setDarkMode] = useState(false);

useEffect(() => {

const storedTasks = JSON.parse(localStorage.getItem("tasks")) || [];

setTasks(storedTasks);

}, []);

useEffect(() => {

localStorage.setItem("tasks", JSON.stringify(tasks));

}, [tasks]);

const addTask = () => {

if (taskInput.trim()) {

setTasks([...tasks, {

id: Date.now(),

title: taskInput,

description: taskDescription,

priority: taskPriority,

completed: false

}]);

setTaskInput("");

setTaskDescription("");

setTaskPriority("Low");

}

};

const toggleTaskCompletion = (taskId) => {

setTasks(tasks.map(task => task.id === taskId ? { ...task, completed: !task.completed } : task));

};

const deleteTask = (taskId) => {

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

};

const handleDragEnd = (result) => {

if (!result.destination) return;

const reorderedTasks = [...tasks];

const [movedTask] = reorderedTasks.splice(result.source.index, 1);

reorderedTasks.splice(result.destination.index, 0, movedTask);

setTasks(reorderedTasks);

};

return (

<div className={`${darkMode ? "bg-gray-900 text-white" : "bg-white text-black"} min-h-screen p-4`}>

<button onClick={() => setDarkMode(!darkMode)} className="mb-4 p-2 border rounded">

Toggle Dark Mode

</button>

<div className="mb-4">

<input

type="text"

placeholder="Task Title"

value={taskInput}

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

className="p-2 border rounded mr-2"

/>

<input

type="text"

placeholder="Task Description"

value={taskDescription}

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

className="p-2 border rounded mr-2"

/>

<select

value={taskPriority}

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

className="p-2 border rounded mr-2"

>

<option value="Low">Low</option>

<option value="Medium">Medium</option>

<option value="High">High</option>

</select>

<button onClick={addTask} className="p-2 bg-blue-500 text-white rounded">Add Task</button>

</div>

<DragDropContext onDragEnd={handleDragEnd}>

<Droppable droppableId="tasks">

{(provided) => (

<div {...provided.droppableProps} ref={provided.innerRef}>

{tasks.map((task, index) => (

<Draggable key={task.id} draggableId={task.id.toString()} index={index}>

{(provided) => (

<div

ref={provided.innerRef}

{...provided.draggableProps}

{...provided.dragHandleProps}

className="flex justify-between p-2 border-b items-center"

>

<input

type="checkbox"

checked={task.completed}

onChange={() => toggleTaskCompletion(task.id)}

className="mr-2"

/>

<div>

<p className={task.completed ? "line-through" : ""}>{task.title}</p>

<p className="text-sm text-gray-500">{task.description}</p>

<p className={`text-xs ${task.priority === "High" ? "text-red-500" : task.priority === "Medium" ? "text-yellow-500" : "text-green-500"}`}>Priority: {task.priority}</p>

</div>

<button onClick={() => deleteTask(task.id)} className="text-red-500">X</button>

</div>

)}

</Draggable>

))}

{provided.placeholder}

</div>

)}

</Droppable>

</DragDropContext>

</div>

);

};

export default TaskManager;