**Task 1:**

**App.jsx:**

import React from 'react';

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

import Home from './components/Home';

import SearchRecipes from './components/SearchRecipes';

import ViewRecipes from './components/ViewRecipes';

import AddRecipe from './components/AddRecipe';

function App() {

return (

<Router>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/search-recipes" element={<SearchRecipes />} />

<Route path="/view-recipes" element={<ViewRecipes />} />

<Route path="/add-recipe" element={<AddRecipe />} />

</Routes>

</Router>

);

}

export default App;

**AddReceipe.js:**

import React, { useState } from 'react';

const AddRecipe = () => {

const [recipeName, setRecipeName] = useState('');

const [category, setCategory] = useState('');

const [ingredients, setIngredients] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

// Logic to add the recipe (we'll add this functionality later)

console.log({ recipeName, category, ingredients });

};

return (

<div>

<h1>Add a New Recipe</h1>

<form onSubmit={handleSubmit}>

<div>

<label>Recipe Name:</label>

<input

type="text"

value={recipeName}

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

required

/>

</div>

<div>

<label>Category:</label>

<select

value={category}

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

required

>

<option value="">Select a category</option>

<option value="Breakfast">Breakfast</option>

<option value="Lunch">Lunch</option>

<option value="Dinner">Dinner</option>

<option value="Dessert">Dessert</option>

</select>

</div>

<div>

<label>Ingredients (comma-separated):</label>

<input

type="text"

value={ingredients}

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

required

/>

</div>

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

</form>

</div>

);

};

export default AddRecipe;

**ViewReceipe.js**

import React from 'react';

const ViewRecipes = () => {

return (

<div>

<h1>No recipes available</h1>

</div>

);

};

export default ViewRecipes;

**SearchRecipes.js:**

// src/components/SearchRecipes.js

import React, { useState } from 'react';

const SearchRecipes = () => {

const [query, setQuery] = useState('');

const [recipes, setRecipes] = useState([]);

const [loading, setLoading] = useState(false);

const [error, setError] = useState(null);

const handleSearch = async () => {

if (!query) return;

setLoading(true);

setError(null);

try {

const response = await fetch(`https://www.themealdb.com/api/json/v1/1/search.php?s=${query}`);

const data = await response.json();

setRecipes(data.meals || []);

} catch (err) {

setError('Failed to fetch recipes.');

} finally {

setLoading(false);

}

};

return (

<div>

<h2>Search Recipes</h2>

<input

type="text"

placeholder="Search for recipes"

value={query}

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

/>

<button onClick={handleSearch}>Search</button>

{loading && <p>Loading...</p>}

{error && <p>{error}</p>}

{recipes.length === 0 && !loading && <p>No recipes found</p>}

<div>

{recipes.map((recipe) => (

<div key={recipe.idMeal}>

<h3>{recipe.strMeal}</h3>

<img src={recipe.strMealThumb} alt={recipe.strMeal} width="200" />

</div>

))}

</div>

</div>

);

};

export default SearchRecipes;

**Home.js:**

import React from 'react';

import { Link } from 'react-router-dom';

const Home = () => {

return (

<div>

<h1>Welcome to the Recipe Finder</h1>

<nav>

<ul>

<li><Link to="/search-recipes">Search Recipes</Link></li>

<li><Link to="/view-recipes">View Recipes</Link></li>

<li><Link to="/add-recipe">Add Recipe</Link></li>

</ul>

</nav>

</div>

);

};

export default Home;

**Task 2 and 3:**

import React, { createContext, useState, useContext } from 'react';

const RecipeContext = createContext();

export const useRecipeContext = () => {

return useContext(RecipeContext);

};

export const RecipeProvider = ({ children }) => {

const [recipes, setRecipes] = useState([]);

const [categoryFilter, setCategoryFilter] = useState('');

const addRecipe = (newRecipe) => {

setRecipes((prevRecipes) => [...prevRecipes, newRecipe]);

};

const filteredRecipes = categoryFilter

? recipes.filter((recipe) => recipe.category === categoryFilter)

: recipes;

return (

<RecipeContext.Provider value={{ recipes: filteredRecipes, addRecipe, setCategoryFilter }}>

{children}

</RecipeContext.Provider>

);

};

**App.jsx:**

import React from 'react';

import { BrowserRouter as Router, Route, Switch, Link } from 'react-router-dom';

import { RecipeProvider } from './context/RecipeContext'; // Import the RecipeProvider

import Home from './components/Home';

import SearchRecipes from './components/SearchRecipes';

import ViewRecipes from './components/ViewRecipes';

import AddRecipe from './components/AddRecipe';

function App() {

return (

<RecipeProvider> {/\* Wrap your app with the RecipeProvider \*/}

<Router>

<div>

<nav>

<Link to="/">Home</Link>

<Link to="/search">Search Recipes</Link>

<Link to="/view">View Recipes</Link>

<Link to="/add">Add Recipe</Link>

</nav>

<Switch>

<Route path="/" exact component={Home} />

<Route path="/search" component={SearchRecipes} />

<Route path="/view" component={ViewRecipes} />

<Route path="/add" component={AddRecipe} />

</Switch>

</div>

</Router>

</RecipeProvider>

);

}

export default App;

**AddReceipe.js:**

import React, { useState } from 'react';

import { useRecipeContext } from '../context/RecipeContext'; // Import the custom hook

const AddRecipe = () => {

const [name, setName] = useState('');

const [category, setCategory] = useState('');

const [ingredients, setIngredients] = useState('');

const { addRecipe } = useRecipeContext(); // Use the context to get addRecipe function

const handleSubmit = (e) => {

e.preventDefault();

const newRecipe = {

name,

category,

ingredients: ingredients.split(',').map((ingredient) => ingredient.trim()),

};

addRecipe(newRecipe); // Add the new recipe to the global state

setName('');

setCategory('');

setIngredients('');

};

return (

<div>

<h2>Add Recipe</h2>

<form onSubmit={handleSubmit}>

<div>

<label>Recipe Name</label>

<input

type="text"

value={name}

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

required

/>

</div>

<div>

<label>Category</label>

<input

type="text"

value={category}

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

required

/>

</div>

<div>

<label>Ingredients</label>

<input

type="text"

value={ingredients}

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

required

/>

</div>

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

</form>

</div>

);

};

export default AddRecipe;

**ViewReceipe.js:**

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

import { useRecipeContext } from '../context/RecipeContext'; // Import the custom hook

const ViewRecipes = () => {

const { recipes, setCategoryFilter } = useRecipeContext(); // Get recipes from context

const [apiRecipes, setApiRecipes] = useState([]);

const [loading, setLoading] = useState(false);

const [error, setError] = useState(null);

// Fetch recipes from the API when the component mounts

useEffect(() => {

const fetchRecipes = async () => {

setLoading(true);

setError(null);

try {

const response = await fetch('https://www.themealdb.com/api/json/v1/1/search.php?s=');

const data = await response.json();

setApiRecipes(data.meals || []);

} catch (err) {

setError('Failed to fetch recipes from API');

} finally {

setLoading(false);

}

};

fetchRecipes();

}, []);

const handleCategoryChange = (e) => {

setCategoryFilter(e.target.value); // Update the category filter in the global state

};

const allRecipes = [...apiRecipes, ...recipes]; // Combine API recipes and local recipes

return (

<div>

<h2>View Recipes</h2>

<label>Filter by Category:</label>

<select onChange={handleCategoryChange}>

<option value="">All Categories</option>

<option value="Breakfast">Breakfast</option>

<option value="Lunch">Lunch</option>

<option value="Dinner">Dinner</option>

<option value="Dessert">Dessert</option>

</select>

{loading && <p>Loading recipes...</p>}

{error && <p>{error}</p>}

{allRecipes.length === 0 && !loading && <p>No recipes found.</p>}

<div>

{allRecipes.map((recipe, index) => (

<div key={index}>

<h3>{recipe.name}</h3>

<p>{recipe.category}</p>

<ul>

{recipe.ingredients.map((ingredient, i) => (

<li key={i}>{ingredient}</li>

))}

</ul>

</div>

))}

</div>

</div>

);

};

export default ViewRecipes;