







App.tsx  
  
import "./App.css";

import ListGroup from "./components/ListGroup";

import "bootstrap/dist/css/bootstrap.css";

import { useState, useEffect } from "react";

interface City {

id: number;

name: string;

description: string;

}

function App() {

const [searchQuery, setSearchQuery] = useState("");

const [newCityName, setNewCityName] = useState("");

const [newCityDescription, setNewCityDescription] = useState("");

const [cities, setCities] = useState(() => {

const savedCities = localStorage.getItem("cities");

return savedCities

? JSON.parse(savedCities)

: [

{

id: 1,

name: "Colombo",

description:

"This is colombo",

},

{

id: 2,

name: "Kandy",

description:

"This is kandy",

},

{

id: 3,

name: "Galle",

description: "This is galle",

},

];

});

const [selectedCity, setSelectedCity] = useState<null | City>(null);

useEffect(() => {

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

}, [cities]);

const handleAddCity = (e: React.FormEvent) => {

e.preventDefault();

if (!newCityName.trim() || !newCityDescription.trim()) return;

const newCity = {

id: cities.length + 1,

name: newCityName.trim(),

description: newCityDescription.trim(),

};

setCities([...cities, newCity]);

setNewCityName("");

setNewCityDescription("");

};

const filteredCities = cities.filter(

(city: { id: number; name: string; description: string }) =>

city.name.toLowerCase().includes(searchQuery.toLowerCase())

);

const handleSelectCity = (city: {

id: number;

name: string;

description: string;

}) => {

setSelectedCity(city);

};

const handleReset = () => {

setSelectedCity(null);

};

return (

<>

<h1>City Explorer</h1>

<div className="container mt-5">

<div className="d-flex justify-content-between align-items-center mb-3">

<input

type="text"

className="form-control me-2"

placeholder="Search cities..."

value={searchQuery}

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

/>

</div>

<ListGroup

items={filteredCities}

heading="Available Cities"

onSelectCity={handleSelectCity}

selectedCity={selectedCity}

/>

<form onSubmit={handleAddCity} className="mb-4">

<h2>Add a New City</h2>

<div className="mb-3">

<input

type="text"

className="form-control mb-2"

placeholder="Enter city name..."

value={newCityName}

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

/>

<input

type="text"

className="form-control mb-2"

placeholder="Enter city description..."

value={newCityDescription}

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

/>

<button type="submit" className="btn btn-success mt-5">

Add City

</button>

<br />

<button

className="btn btn-danger mt-3"

onClick={handleReset}

disabled={!selectedCity}

>

Reset Selection

</button>

</div>

</form>

</div>

</>

);

}

export default App;

ListGroup.tsx

interface City {

id: number;

name: string;

description: string;

}

interface Props {

items: City[];

heading: string;

onSelectCity: (city: City) => void;

selectedCity: City | null;

}

const ListGroup = ({ items, heading, onSelectCity, selectedCity }: Props) => {

return (

<>

<h2>{heading}</h2>

<ul className="list-group mb-3">

{items.map((item) => (

<li

key={item.id}

className={`list-group-item ${selectedCity?.id === item.id ? 'active' : ''}`}

onClick={() => onSelectCity(item)}

>

<div className="d-flex justify-content-between align-items-center">

<h5 className="mb-1">{item.name}</h5>

</div>

</li>

))}

</ul>

{selectedCity && (

<div className="alert alert-info">

<p className="mb-0">{selectedCity.description}</p>

</div>

)}

</>

);

};

export default ListGroup;