**WEEK – 7(Additional HandsOn)**

**React**

**14. ReactJS-HOL**

**Objectives**

**• Explain the need and Benefits of React Context API**

1. Need and Benefits of React Context API

🔹 What is Context API?

The React Context API is a built-in feature that lets you share data globally across components without prop drilling (passing props manually through each level).

🔹 Why do we need Context?

* To avoid passing props down multiple levels
* To share global state (like themes, language, user authentication, etc.)
* **Benefits:**

| **Benefit** | **Description** |
| --- | --- |
| Eliminates Prop Drilling | No need to pass props through every nested component |
| Cleaner & Scalable Code | Makes components more independent and reusable |
| Centralized Data Handling | Share data (theme, auth, language, cart, etc.) globally |
| Built into React | No need for third-party state management tools |

**• Working with createContext()**

Step 1: Create a Context

import React, { createContext } from 'react';

const ThemeContext = createContext(); // default value optional

Step 2: Create a Provider Component

function ThemeProvider({ children }) {

const theme = 'dark';

return (

<ThemeContext.Provider value={theme}>

{children}

</ThemeContext.Provider>

);

}

Step 3: Use Context in Child Components

import React, { useContext } from 'react';

function Header() {

const theme = useContext(ThemeContext);

return <h1 style={{ color: theme === 'dark' ? 'white' : 'black' }}>Welcome</h1>;

}

**• List the types of Router Components**

React Router provides several components for navigation and route handling.

**Commonly Used Router Components:**

| **Component** | **Purpose** |
| --- | --- |
| <BrowserRouter> | Wraps the app and enables routing using HTML5 history API |
| <Routes> | Wraps all your <Route /> elements |
| <Route> | Defines a path and the component to render |
| <Link> | Navigation link (like <a>, but without reloading the page) |
| <NavLink> | Like <Link> but with active styling for selected routes |
| <Outlet> | Used in nested routing to render child components |
| useNavigate() | A hook to navigate programmatically |
| useParams() | A hook to read route parameters |
| useLocation() | A hook to get current location object |

**Basic Example:**

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

function App() {

return (

<BrowserRouter>

<nav>

<Link to="/">Home</Link> | <Link to="/about">About</Link>

</nav>

<Routes>

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

<Route path="/about" element={<About />} />

</Routes>

</BrowserRouter>

);

}

**CODE:**

**src/ThemeContext.js:**

import React from 'react';

// Create a Context with default theme 'light'

const ThemeContext = React.createContext('light');

export default ThemeContext;

**src/EmployeeList.js:**

import React, { useContext } from 'react';

import EmployeeCard from './EmployeeCard';

import ThemeContext from './ThemeContext';

function EmployeeList() {

  // Consume theme value from context

  const theme = useContext(ThemeContext);

  // Example list of employees (replace with real data if needed)

  const employees = [

    { id: 1, name: 'John Doe', position: 'Developer' },

    { id: 2, name: 'Jane Smith', position: 'Designer' }

  ];

  return (

    <div className={`employee-list ${theme}`}>

      <h2>Employee List</h2>

      {employees.map(emp => (

        <EmployeeCard key={emp.id} employee={emp} />

      ))}

    </div>

  );

}

export default EmployeeList;

**src/EmployeeCard.js:**

import React, { useContext } from 'react';

import ThemeContext from './ThemeContext';

function EmployeeCard({ employee }) {

  const theme = useContext(ThemeContext);

  return (

    <div className={`employee-card ${theme}`}>

      <h3>{employee.name}</h3>

      <p>{employee.position}</p>

      <button className={theme === 'dark' ? 'btn-dark' : 'btn-light'}>

        Contact

      </button>

    </div>

  );

}

export default EmployeeCard;

**src/App.js:**

import React from 'react';

import EmployeeList from './EmployeeList';

import ThemeContext from './ThemeContext';

function App() {

  const theme = 'light'; // Change to 'dark' to test dark mode

  return (

    <ThemeContext.Provider value={theme}>

      <div className={`app-container ${theme}`}>

        <EmployeeList />

      </div>

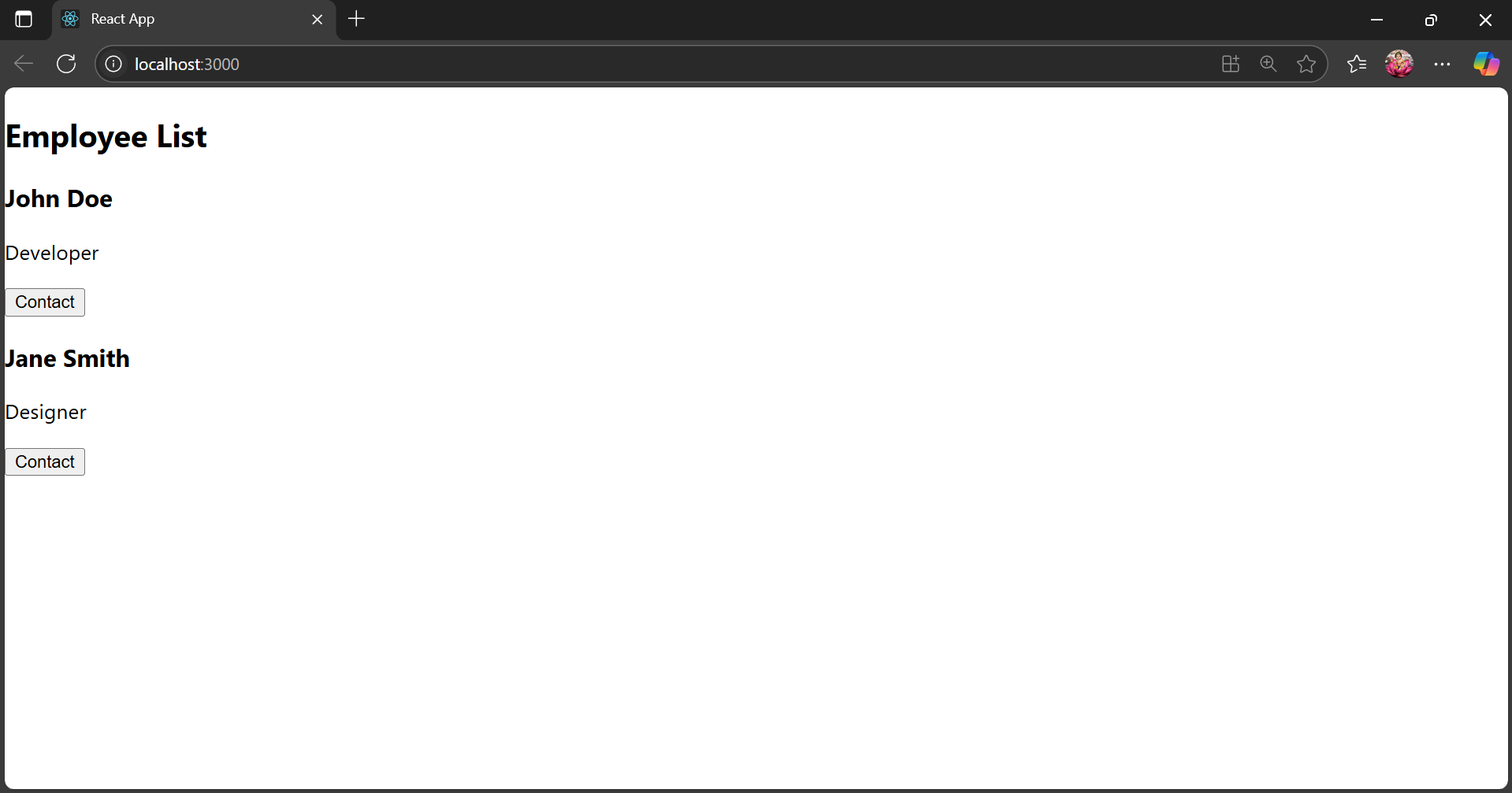
    </ThemeContext.Provider>

  );

}

export default App;

**OUTPUT:**

****