**EXERCISE 6 REACTJS-HOL**

**1. Explain the need and benefits of React Router**

React applications are single-page applications (SPAs) that dynamically update the content without reloading the entire page. To achieve seamless navigation between different views or pages, client-side routing is essential. React Router provides this functionality by mapping URLs to specific components, allowing users to navigate the application smoothly. The main benefits of React Router include efficient navigation without full page reloads, dynamic routing based on state or data, and support for nested routes to create complex layouts. It also allows easy handling of URL parameters and uses a declarative approach, making the routing logic simple and easy to maintain.

**2. Identify the Components in React Router**

React Router consists of several core components that enable routing in a React application. The <BrowserRouter> component wraps the entire application and provides routing capabilities using the HTML5 history API. The <Routes> component acts as a container for multiple <Route> elements, each of which defines a specific path and maps it to a corresponding component. Navigation within the app is achieved using <Link> and <NavLink> components, where <NavLink> provides active link styling. Additionally, React Router offers hooks such as useNavigate() for programmatic navigation, useParams() to access route parameters, and useLocation() to retrieve the current location details.

**3. List the Types of Router Components**

There are different types of router components in React Router. The **BrowserRouter** uses the HTML5 history API and is the most common choice for modern web applications. The **HashRouter** uses the URL hash (#) for navigation and is useful for older browsers or static hosting environments. The **MemoryRouter** keeps the routing history in memory and is generally used for testing or environments without a browser history. For mobile applications built with React Native, the **NativeRouter** is used to provide similar routing functionality.

**4. Parameter Passing via URL**

React Router allows passing parameters through URLs in two main ways: path parameters and query parameters. Path parameters are defined in the route path using a colon (:) before the parameter name, such as /user/:id. These parameters can be accessed using the useParams() hook inside the component. For example, if the URL is /user/101, id will hold the value 101. Query parameters, on the other hand, are appended to the URL after a question mark, such as /search?query=react. These can be accessed using the useLocation() hook combined with URLSearchParams. Both methods make it easy to pass and retrieve dynamic data through the URL.

**HANDON-EXERCISE:**

**CODE STRUCTURE:**

****

**CODE:**

**Home.js**

function Home() {

  return (

    <div>

      <h2>Welcome to Cognizant Academy Trainers Page</h2>

    </div>

  );

}

export default Home;

**trainer.js**

class Trainer{

    constructor(trainerId,name,email,phone,technology,skills){

        this.trainerId=trainerId;

        this.name=name;

        this.phone=phone;

        this.technology=technology;

        this.skills=skills;

    }

}

export default Trainer**;**

**TrainerDetails.js**

import { useParams } from "react-router-dom";

function TrainerDetail({ trainers }) {

  const { id } = useParams();

  const trainer = trainers.find((t) => t.trainerId === id);

  if (!trainer) {

    return <h3>Trainer not found</h3>;

  }

  return (

    <div>

      <h2>{trainer.name}</h2>

      <p><strong>Email:</strong> {trainer.email}</p>

      <p><strong>Phone:</strong> {trainer.phone}</p>

      <p><strong>Technology:</strong> {trainer.technology}</p>

      <p><strong>Skills:</strong> {trainer.skills.join(", ")}</p>

    </div>

  );

}

export default TrainerDetail;

**TrainersLists.js**

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

function TrainersList({ trainers }) {

  return (

    <div>

      <h2>Trainers List</h2>

      <ul>

        {trainers.map((trainer) => (

          <li key={trainer.trainerId}>

            <Link to={`/trainer/${trainer.trainerId}`}>{trainer.name}</Link>

          </li>

        ))}

      </ul>

    </div>

  );

}

export default TrainersList;

**trainersMock.js**

import Trainer from "./trainer";

const trainersMock=[

    new Trainer('t-syed8','Syed Khaleelullah','khaleelullah@cognizant.com','97676516962','.NET',['c#','SQL Server','React','.NET core']),

    new Trainer('t-jojo','Jojo Jose','jojo@cognizant.com','97676599921','Java',['Java','JSP','Angular','Spring']),

    new Trainer('t-elisa','Elisa Jones','elisa@cognizant.com','93876516962','Python',['Python','Django','Angular']),

]

export default trainersMock;

**app.js**

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

import Home from "./Home";

import TrainersList from "./TrainersList";

import trainersMock from "./trainersMock";

import TrainerDetail from "./TrainerDetail";

function App() {

  return (

    <Router>

      <div>

        <h1>Cognizant Academy Trainers App</h1>

      </div>

      <nav>

        <Link to="/">Home</Link> |{" "}

        <Link to="/trainers">Show Trainers</Link>

      </nav>

      <Routes>

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

        <Route path="/trainers" element={<TrainersList trainers={trainersMock} />} />

        <Route path="/trainer/:id" element={<TrainerDetail trainers={trainersMock} />} />

      </Routes>

    </Router>

  );

}

export default App;

**OUTPUT:**

