**Exercise 6**

**OBJECTIVES**

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

React is a single-page application (SPA) framework, and by default, doesn't support page navigation. React Router enables dynamic routing, allowing users to navigate without reloading the page. It makes user experience seamless by preserving the application state between views. Routing improves code organization and separation of concerns by mapping paths to components. It allows deep linking, browser history support, and nested route handling.

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

**<BrowserRouter>:** Wraps the app and manages the browser history.

**<Routes>:** Groups multiple <Route> components, enabling routing logic.

**<Route>:** Defines the path and the component to render when matched.

**<Link>:** Used for navigation without reloading the page (replaces <a>).

**<Navigate>** and **<Outlet>** help with redirection and nested routes respectively.

1. **List of the types of Router Components**

**BrowserRouter**: Uses HTML5 history API for clean URLs and is commonly used.

**HashRouter**: Uses the hash portion (#) of the URL, useful in static hosting.

**MemoryRouter**: Stores the URL history in memory (useful for tests or non-browser environments).

**StaticRouter**: Used mainly in server-side rendering (e.g., Next.js or SSR apps).

Each type serves a different use case depending on the environment and hosting.

1. **Parameter passing via url.**

URL parameters allow passing dynamic values through routes (e.g., /user/:id).

In the route component, you define a path like <Route path="/user/:id" element={<User />} />. You can access parameters using useParams() hook in the User component. Example: /user/101 will pass id=101 to the component. This technique enables dynamic rendering based on user interaction or data.

**HANDS ON PRACTICE**

1. **Create a new React app using *create-react-app* tool with the as “TrainersApp”**

Npx create-react-app trainersapp

1. **Add a new file called *trainer.js* inside the src folder and define a class named as “Trainer” with the following properties**
   1. **TrainerId**
   2. **Name**
   3. **Email**
   4. **Phone**
   5. **Technology**
   6. **Skills**

**Trainer.js**

class Trainer{

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

  {

    this.trainerId = trainerId;

    this.name = name;

    this.email = email;

    this.phone = phone;

    this.technology = technology;

    this.skills = skills;

  }

}

export default Trainer;

1. **Create a new TrainersMock.js file which will contain the mock trainer data. Refer the following screenshot for mock data**

**TrainersMock.js**

import Trainer from './Trainer';

const trainers = [

  new Trainer('sandeep7',

    'Sandeep Rishi J B',

    'sandeeprishijb@cognizant.com',

    '9673912341',

    'Python',

    ['Python', 'Django', 'Flask', 'Angular']

  ),

  new Trainer('shrimath17',

    'Shrimath Y',

    'shrimathy@cognizant.com',

    '9351353113',

    'Java',

    ['Java', 'Spring', 'Hibernate', 'React.js']

  ),

  new Trainer('ragul25',

    'Ragul',

    'ragulp@cognizant.com',

    '9874847771',

    'Python',

    ['Python', 'Django', 'Flask', 'React']

  ),

  new Trainer('rahul14',

    'Rahul C',

    'rahulc@cognizant.com',

    '7847715687',

    'Python',

    ['Python', 'Machine Learning', 'Data Science']

  )

]

export default trainers;

1. **Install the support for React router for the dom.**

npm install react-router-dom@6

1. **Create new component named as TrainersList inside Trainerlist.js file. The component should accept the trainer’s data as parameter and render it as a list. The list should display names of each trainers which must be clickable like a hyper link. Refer the following screenshot for the component layout.**

**TrainersList.js**

import React from 'react';

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;

1. **Create a new component named as Home inside Home.js which will be responsible for displaying the following**

**Home.js**

import React from 'react';

function Home()

{

  return(

    <div>

      <h2>Welcome to the Trainers App</h2>

      <p>Explore our expert trainers and their skills</p>

    </div>

  );

}

export default Home;

1. **Create a new component named TrainerDetail in *TrainerDetails.js* file.**

**The component should retrieve a parameter named id from the URL with the help of “useParams” hook from the React router DOM library.**

**It should query the mock trainer data using the id and display the trainer details as show in screenshot.**

**Modify the TrainersList component to add Links to TrainerDetail component while passing the ID. Define a route in App component for the same**

**TrainersList.js**

**TrainersDetails.js**

import React from 'react';

import trainers from './TrainersMock';

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

function TrainersDetails()

{

  const {id} = useParams();

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

  if(!trainer) return <p>Trainer not found!</p>

  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 TrainersDetails;

1. **App.js**

import React from 'react';

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

import Home from './Home';

import TrainersList from './TrainersList';

import trainers from './TrainersMock';

import TrainersDetails from './TrainersDetails';

function App()

{

  return(

    <BrowserRouter>

    <nav>

      <Link to='/' >Home</Link> | <Link to='/trainers'>Trainers</Link>

    </nav>

    <Routes>

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

      <Route path='/trainers' element={<TrainersList trainers={trainers}/>}/>

      <Route path='/trainer/:id' element={<TrainersDetails/>}/>

    </Routes>

    </BrowserRouter>

  );

}

export default App;

1. **Output**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**