**WEEK – 6(Additional HandsOn)**

**React**

**6. ReactJS-HOL**

**Explain the need and benefits of React Router:**

**Need:**In single-page applications (SPAs) built with React, navigation and URL management must be handled on the client side. React Router is essential for mapping different URL paths to specific components, enabling navigation without full page reloads. Without it, users would not be able to bookmark pages, use the back/forward browser buttons, or directly access deep links within the app.

**Benefits:**

* Seamless navigation—no full page reloads, resulting in faster transitions and better user experience.
* Improved user experience—supports use of browser features like history, bookmarks, back/forward buttons, and deep links.
* Declarative and flexible—routes and navigation logic are maintained in code, allowing for easy changes and clear organization.
* Dynamic routing—routes can be generated at runtime based on app state or data.
* Nested and modular routing—makes it easy to manage complex navigation in large applications

**Identify the Components in React Router:**

Components in React Router

React Router uses several key components to handle routing:

* **BrowserRouter:**Keeps the UI in sync with the URL using the HTML5 history API.
* **HashRouter:**Uses the hash portion of the URL (after #) for routing. Useful when server-side configuration is restricted.
* **MemoryRouter:**Stores the history of your “URL” in memory (useful for testing or non-browser environments).
* **Routes:**A container that holds all the Route components.
* **Route**: Declares a mapping from a URL path to a specific React component.
* **Link/ NavLink**: Used to provide navigation without refreshing the page—NavLink adds styling when the route is active.
* **useParams Hook:**Allows you to access dynamic parameters in the route from within your component.

**List the types of Router Components:**

React Router offers several router components, each serving a different environment or use case:

| **Router Type** | **Purpose/Usage** |
| --- | --- |
| BrowserRouter | For web apps using clean URLs with the HTML5 history API |
| HashRouter | Uses URL hash (for static file web servers with no rewrite) |
| MemoryRouter | In-memory route storage (testing, React Native, non-browser) |
| StaticRouter | For server-side rendering (SSR) |
| NativeRouter | For React Native mobile applications |

**Parameter passing via url:**

* Defining Route with Parameters:  
  Add a : before a parameter name in the route's path prop:

jsx

<Route path="/post/:id" element={<Post />} />

* Accessing Parameters:  
  In the component rendered by the route, use the useParams hook:

jsx

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

function Post() {

const { id } = useParams();

return <div>Post ID: {id}</div>;

}

This allows for dynamic rendering based on the current URL.

**CODE:**

**src/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;

**src/TrainersMock.js:**

import Trainer from './Trainer';

const trainers = [

new Trainer(1, 'John Doe', 'john@example.com', '1234567890', 'React', ['JSX', 'Hooks', 'Routing']),

new Trainer(2, 'Jane Smith', 'jane@example.com', '9876543210', 'Node.js', ['Express', 'MongoDB', 'API']),

new Trainer(3, 'Robert Brown', 'robert@example.com', '1122334455', 'Python', ['Django', 'Flask', 'ML']),

];

export default trainers;

**src/Home.js:**

import React from 'react';

function Home() {

return (

<div>

<h1>Welcome to the Cognizant Academy Trainer App</h1>

<p>This app helps you explore trainer details and skills.</p>

</div>

);

}

export default Home;

**src/TrainerList.js:**

import React from 'react';

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

function TrainerList({ trainers }) {

return (

<div>

<h2>Trainer List</h2>

<ul>

{trainers.map(trainer => (

<li key={trainer.trainerId}>

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

</li>

))}

</ul>

</div>

);

}

export default TrainerList;

src/TrainerDetails.js:

import React from 'react';

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

import trainers from './TrainersMock';

function TrainerDetails() {

const { id } = useParams();

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

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

return (

<div>

<h2>Trainer Details</h2>

<p><strong>ID:</strong> {trainer.trainerId}</p>

<p><strong>Name:</strong> {trainer.name}</p>

<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 TrainerDetails;

**src/App.js:**

import React from 'react';

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

import Home from './Home';

import TrainerList from './TrainerList';

import TrainerDetails from './TrainerDetails';

import trainers from './TrainersMock';

function App() {

return (

<Router>

<div>

<nav style={{ marginBottom: '20px' }}>

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

<Link to="/trainers">Trainer List</Link>

</nav>

<Routes>

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

<Route path="/trainers" element={<TrainerList trainers={trainers} />} />

<Route path="/trainer/:id" element={<TrainerDetails />} />

</Routes>

</div>

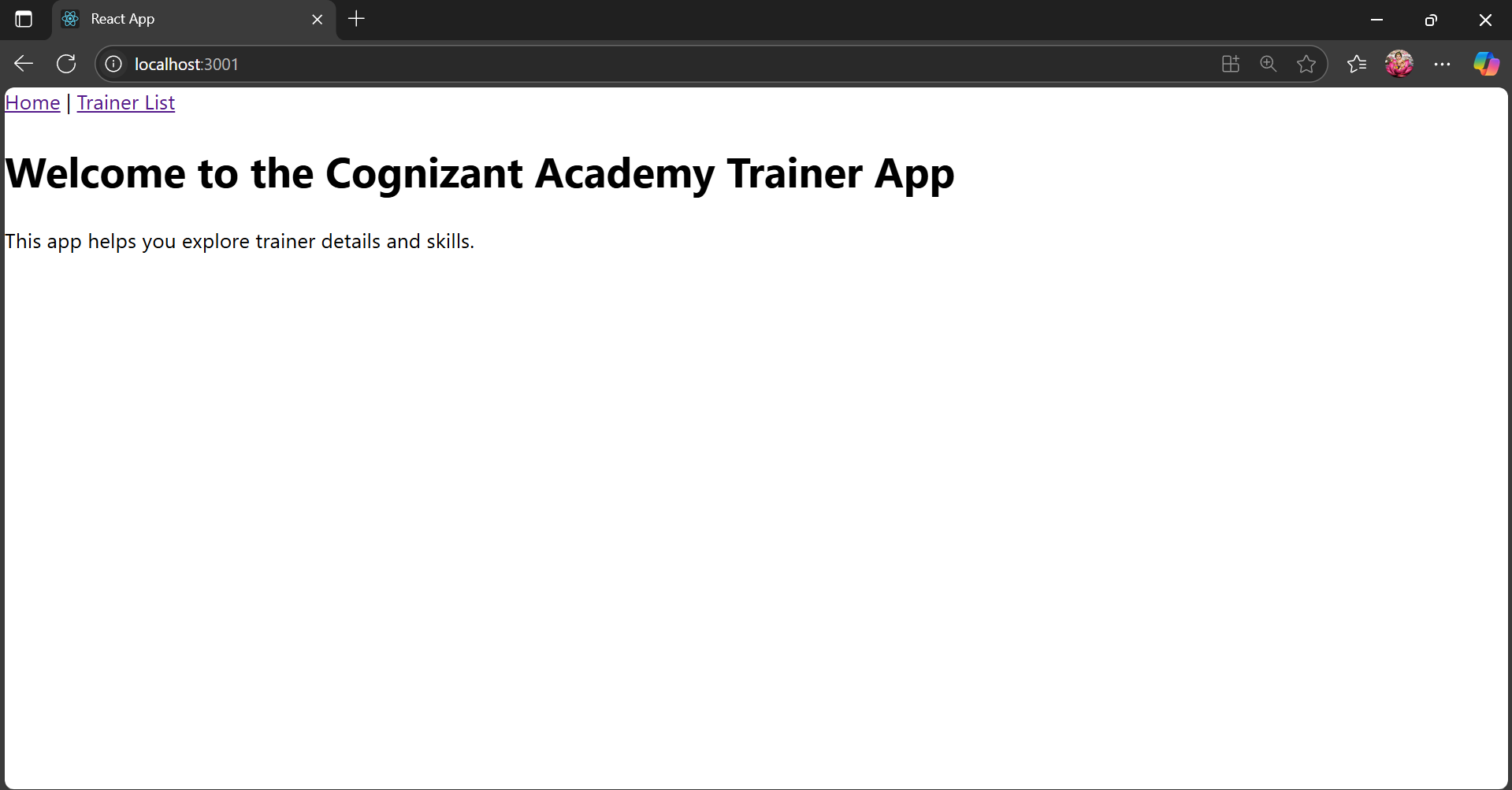
</Router>

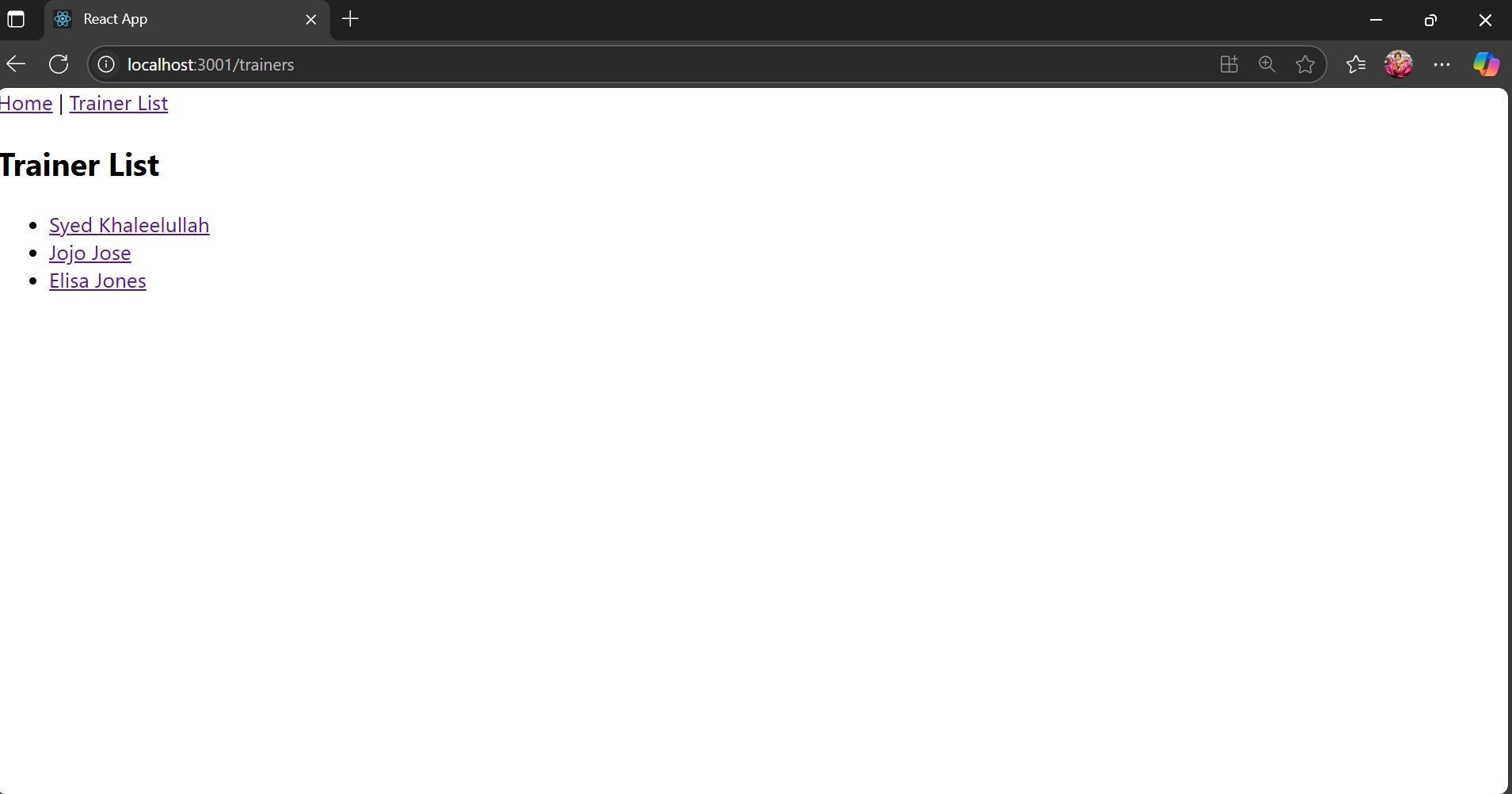
);

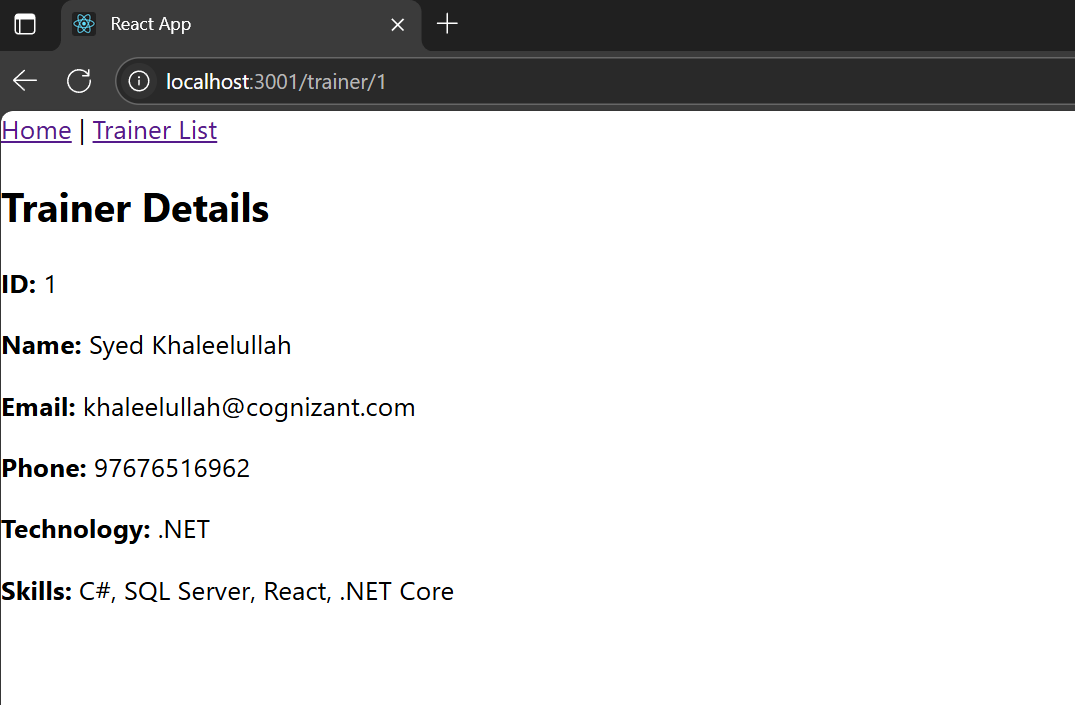
}

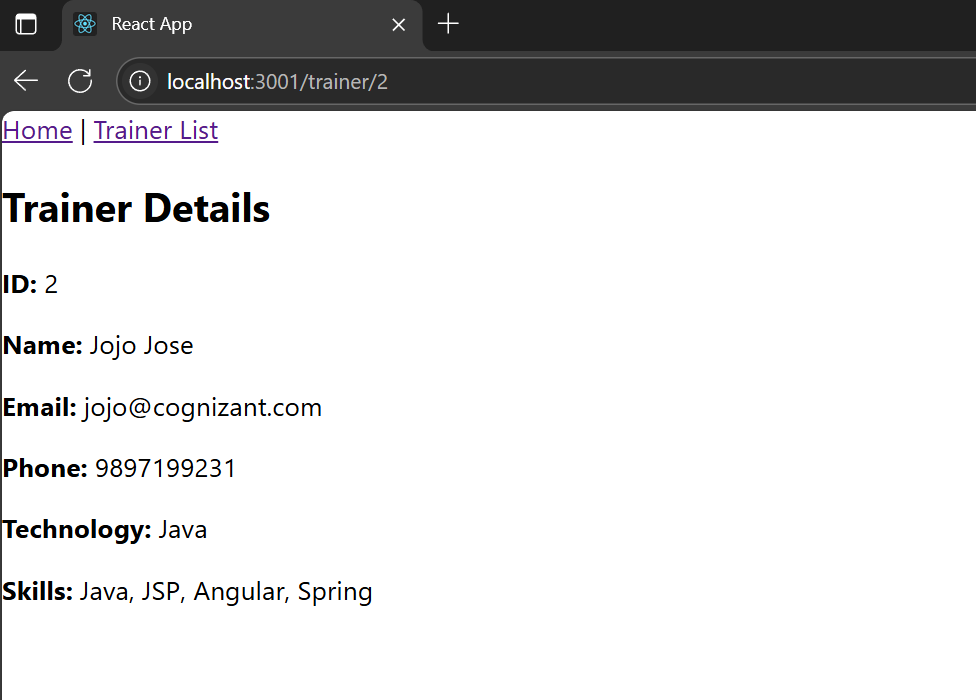
export default App;

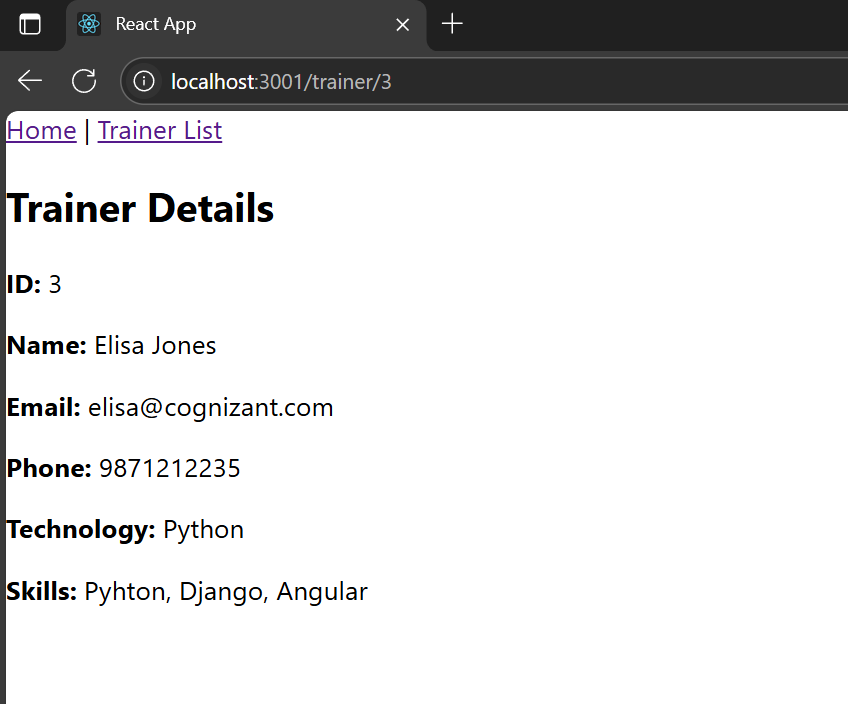
**OUTPUT:**

****

****

****

****

****