Routing in React

React Router

- React Router is a standard library system built on top of the React and used to **create routing** in the React application using **React Router Package**.
- Routing is a process in which a user is **directed to different pages** based on their action or request.
- It provides the **synchronous URL** on the browser with data that will be displayed on the web page.
- It is mainly used for developing **single page web applications(SPAs) that displays multiple views in an application**.

Setting up the Router Package

Three different packages for routing are:

- **react-router:** It provides the **core routing components and functions** for the React Router applications.
- **react-router-native:** It is used for **mobile applications**.
- **react-router-dom:** It is used for **web applications design**.

react-router vs react-router-dom

To use **react routing** in web application, first, you need to install **react-router-dom modules** in your application as you cannot install **react-router** directly into your application.

Command to Install react-router-dom is

npm install react-router-dom --save

Preparing the Project for Routing

Step 1: Create the components that you want to include in SPA.

Note: Create the components in a separate folder for convenience(For Example: components folder).

Home.js

```
import React, { Component } from 'react';
class Home extends Component {
   state = {  count:0, name:"MVJ" }
   render() {
      return (
         <>
            <h1 class="badge bg-primary text-wrap text-uppercase"> Hello,
{this.state.name},{this.state.count}</h1>
            <p>                How are you?   </p>
        </>
      );
   }
}
export default Home;
```

FuncDemo.js

```
import React from 'react';
function FuncDemo() {
    return (
    <h1>Hello, Demo!!!</h1>
    );
}
export default FuncDemo;
```

Hooksdemo.js

```
import React, { useState } from 'react';
const      Hooksdemo=()=> {
   const [state,setState] = useState(0);
   return (
   <>
      <button onClick={()=>(setState(state+1))}>+</button>
      <span>{state}</span>
      <button onClick={()=>(setState(state-1))}>-</button>
   </>
   );
}

export default Hooksdemo;
```

HooksListDemo.js

```
import React, {useState } from 'react';                //imrs
const HooksListDemo = () =>{
   const [message] = useState("List of Letters");
   const [list,setList] = useState(
      [
         {id:1,name:"A"},
         {id:2,name:"B"},
         {id:3,name:"C"},
         {id:4,name:"D"},
      ]
   );
   return (
    <>
                <h1> {message} </h1>
                <ul>  { list.length? list.map(lst => (<li key={lst.id}>{lst.id}-{lst.name}</li>)):null }        </ul>
    </>
    );
}
export default HooksListDemo;
```

Step 2: Import all the components into App.js file

```
import Home from './components/Home';
import FuncDemo from './components/FuncDemo';
import HooksDemo from './components/HooksDemo';
import HooksListDemo from './components/HooksListDemo';
```

Step 3: Import components which helps us to **implement the Routing** from **react-router-dom**

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

<BrowserRouter> :

It is used for handling the dynamic URL.

It is a router implementation that uses the HTML5 history API (**pushstate**, **replacestate**, and **popstate** events) to **keep your UI in sync with the URL**.

It is the **parent component** used to store all other components.

<Link>:

This component is used to **create links** which allows us to **navigate** on different **URLs** and render its content **without reloading the webpage**. It works as an HTML anchor tag.

Prop:

to=“path”: Refers to the path of the component to navigate

<Route> :

A route is a conditionally shown component that provides UI when its **path matches the current URL**.

It is used to **define and render component** based on the specified path.

The route component will help us establish the link between the **component's UI and the URL**.

Prop:

exact: This matches the exact value with the URL.

Path="path": Path specifies a pathname of the component.

element={<ComponentaName/>}: Refers to the component that will render if the path matches.

<Routes> :

To read a single component, wrap all route inside the Routes component.

Step 4: Add Navigation using Link component inside BrowserRouter(Alias Router) component

```
<Router>
        <ul class="nav nav-tabs">
            <li class="nav-item">
                <Link class="nav-link active" to="/">Home</Link>
            </li>
            <li class="nav-item">
                <Link class="nav-link" to="/FuncDemo">FuncDemo</Link>
            </li>
            <li class="nav-item">
                <Link class="nav-link" to="/HooksDemo">HooksDemo</Link>
            </li>
            <li class="nav-item">
                <Link class="nav-link" to="/HooksListDemo">HooksListDemo</Link>
            </li>
        </ul>
</Router>
```

npm install bootstrap@5.2.2

import 'bootstrap/dist/css/bootstrap.css'; **in index.js file**

Step 5: Implement Routing using Routes and Route components in App.js file

```
<Router>
        …// retain previous stuff here
        <Routes>
            <Route path="/" element={<Home/>}/>
            <Route path="/FuncDemo" element={< FuncDemo />}/>
            <Route path="/HooksDemo" element={< HooksDemo />}/>
            <Route path="/HooksListDemo" element={< HooksListDemo />}/>
        </Routes>
</Router>
```

Step 6: Save App.js file and Run the command

npm start

Routing with props

- With **React Router v6**, since you're using the element, you just pass a prop to the component as **you normally would**.

Example:

```
<Route path="/FuncDemo"
element={<FuncDemo name={"MVJ"}/>}/>
```

Now Modify the following file to access the props in function component (changes shown in bold face text)

FuncDemo.js

```
import React from 'react';
function FuncDemo(props) {
    return (
    <h1>Hello, {props.name} Demo!!!</h1>
    );
}
export default FuncDemo;
```

Routing with Query Parameters

A custom hook that builds on useLocation to parse the query string.

Create the Function Component with the following code

QueryParam.js

```
import React from "react";
import {Link, useLocation } from "react-router-dom";
function useQuery() {
  const  { search } = useLocation();
  return React.useMemo(() => new URLSearchParams(search), [search]);
}
export default function QueryParamsExample() {
  let query = useQuery();
  let name = query.get("name");
  return (
    <div>
      <div>
        <ul>
          <li> <Link to="/QueryParam?name=netflix">Netflix</Link> </li>
        </ul>
        <h1>{ name? "The name in the query string is " + name : "There is no name in the query string“ }</h1>
      </div>
    </div>
  );
}
```

Update the App.js file to include QueryParam.js component in routing

<Routes>

…//retain your previous stuff here

<Route path="/QueryParam" element={<QueryParamsExample />}/>

</Routes>