We can do routing in react using a very famous library `react-router-dom`. React internally doesn't posses any direct of routing users from one page to another. That's where this library helps.

How to install react-router-dom

`npm install react-router-dom`

How to configure react-router-dom

To configure react-router-dom we need to do a couple of steps:

- Wrap your App component inside a new BrowserRouter component provided by react-router-dom

```
//inside main.jsx
import { StrictMode } from 'react'

import { createRoot } from 'react-dom/client'

import App from './App.jsx'

import './index.css'

import { BrowserRouter } from 'react-router-dom'



createRoot(document.getElementById('root')).render(

  <StrictMode>

      <BrowserRouter>

            <App />

      </BrowserRouter>

  </StrictMode>,

)
```

- Then in the app component we have to use two components `Routes` and `Route`. Both of these are given by react-router-dom. `Route` component defined on which path a particular component must be rendered. `Routes` component keeps all the collection of individual `Route` together.

```
//App.jsx
import { Route, Routes } from "react-router-dom";

function App() {

  return (
 <>

 <div>

  Navbar       //it wont change we will redirect to other page also navbar like sticky

 </div>

        <Routes>

          <Route path="/play" element={<PlayGame />} />    {/*localhost:3898 something /play it will redirect to play game page everytime it is refresing page  */}

          <Route path="/start" element={<StartGame />} />

          <Route path="*" element={<div>not found</div>} />

        </Routes>

        </>

  );

}

export default App;
```

- Here we have defined 2 routes, one if /start and another is /play. If the user goes to the /play route then component is rendered, and for /start route component is rendered.

- If the user goes to route that is specifically not defined here, then we show Not Found to them. This is achieved by saying `path="*"` in one of the routes.

And with this , react-router-dom is setup completely, now we can use Link component or navigator from react-Router-dom to move the user from one page to another without any refreshing of the page.

Link component:

Link component helps us to put a hyperlink (just like anchor tag) on the dom, where is the user click then they will be redirected to a particular route and it's corresponding mapped component is rendered **without any page refresh.**

```
import { Link } from 'react-router-dom';
function PlayGame() {

  return (

    <div>

      <h1>Play Game</h1>

      <Link to="/start">Start Game</Link>   {/*if you dont want to refresh the whole page while redirecting the new page  when you write like this ny using link tag it just redirect the page without refresing the whole page*/}

    </div>

  );

}
export default PlayGame;
```

Here we are providing a link to `/start` route using the Link tag. When rendered on the dom it converts to an anchor tag, but doesn't refresh the page link an anchor tag.

Navigator in react-router-dom

If we don't want the user to click on the hyperlink to move to another page and instead move the user to different pages programatically, then we can use navigator object.
Let's say we have a form and we first do a form validation and then redirect the user to a page, for this usecase navigator is a perfect choice.
To use navigator we can call the inbuilt `useNavigate` hook inside react-router-dom.

```
import { useNavigate } from "react-router-dom";

import TextInputFormContainer from "../../../Components/TextInputForm/TextInputFormContainer";

function StartGame() {
    const navigate = useNavigate();  //by using this method user can redirect one page to other page

    function handleSubmit() {

        navigate('/play');

    }

    return(

        <div>

            <h1>Start Game</h1>

            <TextInputFormContainer onSubmit={handleSubmit} />

        </div>

    )

}

export default StartGame;
```

use Params hook :

useParam hook is created by react-router-dom. It returns an object of key/value pairs of the dynamic params form the current URL. Child routes inherits all params from their parent routes.

```
//Router.jsx in cryptocurrency project
import { Route, Routes } from "react-router-dom";
import Home from "../../Pages/Home";
import CoinDetailsPage from "../../Pages/CoinDetailsPage";
function Routing() {

    return(
    <Routes>

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

        <Route path="/details/:coinId" element={<CoinDetailsPage />} />
{/*here details is detials is constant part after that coinId is we can change whything it will show coinDetailsPage */}

    </Routes>

    )
}
export default Routing;
{/*if you want to fetch what value{ coinId} passed in URL you can use Params*/}
```

```
// CoinDetailsPage
import { useParams } from "react-router-dom";
function CoinDetailsPage() {
    const { coinId } = useParams();
    return(
       <div>Coin Details Page {coinId}</div>
    )

}

export default CoinDetailsPage;
```