# 12-****React Router****

**react-router-dom** is a package in the React ecosystem that provides routing capabilities for single-page applications. It's a part of the larger **react-router** library which handles routing for React applications

Routing in React.js refers to the process of managing the navigation and rendering of different components based on the URL of the application.

**Phase 1**

1. Create a react project; Install react-router-dom library as dependency
2. Setup BrowserRoute in index.js
3. Setup Routes and Route components in App.jsx
   * 1. Different Route paths can home / , about, users, contact, login. Create the same pages as separate react component and import it in App.jsx and also make sure the same is passed to element prop of Route component. Ref:

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

b. All Routecomponents are wrapped in Routes component

1. Set up Link components in App.jsx for the same routes home / about, users, contact, login Ref :

**<Link to="/">Home</Link>**

1. You can further refactor your code by making a separate Navbar and AllRoutes component
   1. Navbar can have all the links and AllRoutes can have all the different routes Ref :

**``jsx**

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

**export default function Navbar() {**

**return (**

**<div**

**style={{**

**display: "flex",**

**alignItems: "center",**

**justifyContent: "space-around",**

**width: "80%",**

**margin: "auto"**

**}}**

**>**

**<Link to="/">Home</Link>**

**....**

**</div>**

**);**

**}**

**```**

**```jsx**

**import { Route, Routes } from "react-router-dom";**

**export default function AllRoutes() {**

**return (**

**<Routes>**

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

**...**

**</Routes>**

**);**

**}**

**```**

**Phase 2**

1. Instead of hardcoding Link components. You can create a list of Link components with links array which can look something like this ( Refactoring Step )

**`const links = [**

**{ path: "/", title: "Home" },**

**{ path: "/about", title: "About" },**

**...**

**];**

1. NavLink are good alternative if you want to also showcase the active style as well. You can replace Link components with NavLink components in that case. ( They are also part of react-router-dom library. implement active and default styling Ref : // inline styling

<NavLink

style={({ isActive }) => {

return isActive ? activeStyle : defaultStyle;

}}

key={link.path}

to={link.path}

>

{link.title}

</NavLink>

// module.css

<NavLink

style={({ isActive }) => {

return isActive ? activeStyle : defaultStyle;

}}

key={link.path}

to={link.path}

>

{link.title}

</NavLink>

**Phase 3**

1. In Users Page, on mount phase make an API call, get the data and display the same on UI ( Info about the user like Image, Name and also create a Link component for each user. This Link component will update the link to /users/${user.id} and it can have text More Info. Clicking on this Link will take user to /users/1 page
2. Create a dynamic Route component which will render SingleUserPage or UserPage ( whatever you named it ). Whenever user tries to access this users/variable route. It’ll render this single user page. Ref :

<Route path="/users/:id" element={<SingleUserPage />}></Route>

In SingleUserPage or UserPage Component; Retrive the user if from the url by using useParams hook that comes from react-router-dom library

import { useState, useEffect } from "react";

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

function SingleUserPage() {

let params = useParams();

....

1. As soon as the component mounts; Make a network request in this UserPage component; get the data specific to that particular user by hitting single user endpoint ( using id ) . Display the same data onto UI.
2. Please make sure that also maintain loading and error state and display relevant UI wherever network requests are involved
3. When the link entered by the user does not match any of the Route. Please maintain a NotFound Page.. This is be a simple NotFound which display page not found message.