React Routing

React Router

Traditional multi-page web application uses server based routing where the user requests for a page, the request goes to the server, and when they navigate through different parts of the web page it a new request will be sent to the server to request for a new page. The browser will basically request new web page everytime the user interacts with the web page.

That's the traditional MPA model, on the other hand, Single Page Application model uses client side routing that basically loads in only single HTML page. When the user navigates to a different route, React Router will intercept the URL and update the browser history without re-requesting the request to the server. The page is dynamically updated with the corresponding component based on the route. The web application will only request API to the server for data to populate for the web page. This improves user experience because it doesn't need to go to the server for every navigation.

React Router isn't package by default from the core library and in order to leverage it you will need to install it into your React application.

```
npm install react-router-dom
```

How to Implement Routing

To implement routing you would need couple components to manage the user's navigation history.

BrowserRouter

This component is the backbone of React Router, it will keep your UI in sync with the URL. So whenever you make changes to the URL it will result in the correct corresponding component being rendered. This means that any navigation will result also in browser history allowing the user to easily go back or go forward in their page history.

> Under the hood, this component uses the HTML 5 History API which consists of `pushState, replaceState, popState` to monitor and manipulate the application's browsing history.

To leverage this component you would wrap your entire application in BrowserRouter component in order to give React Router the control of your application and enable routing correctly.

```
app.render(
    <BrowserRouter>
        <Content />
    </BrowserRouter>
);
```

In this simple example, we have wrapped our main web application with BrowserRouter enabling the routing capability for our application.

Routes and Route

These two components are used to group all of the app's routes. **Each Route is used to specify a path to the React component that you would like to render when the URL matches to the given path.**

As a example you can define routing to your web application like such:

```
app.render(
    <BrowserRouter>
        <Routes>
            <Route path="/" element={<MainPage />} />
            <Route path="/about-me" element={<SidePage />}/>
        </Routes>
    </BrowserRouter>
);
```

The Route components contains the `/` URL path that the MainPage component will be rendered and `/about-me` for the SidePage.

As you can see, the component that you're rendering will need to be put into brackets and referenced within it, otherwise, JSX will not know that it is a React component that you would like to render.

Link

You can't have navigation without the navigation bar right? Otherwise, how else are your user going to be browsing around your web application?

You can create navigation links to your application by using the Link component.

```
app.render(
    <BrowserRouter>
        <nav>
            <Link to="/">Main Page</Link>
            <Link to="/about-me">About Me</Link>
        </nav>
        <Routes>
            <Route path="/" element={<MainPage />} />
            <Route path="/about-me" element={<SidePage />} />
        </Routes>
    </BrowserRouter>
);
```

The navigation bar will be displayed on the top and when you click on the different navigation tab, it will route you to the corresponding component for each path.

useNavigate

You can also programmatically navigate the page, say after you click a button inside the component or after handling an event.

```
export default function MainPage() {
    let navigate = useNavigate();

    function handleSubmit() {
        navigate("/about-me")
    }

    return (
        <div>
            <button onClick={handleSubmit}>Click me</button>
            <p>This is a page about me personally!</p>
        </div>
    )
}
```

Revision #3
Created 2025-11-21 02:14:09 UTC by Tamarine
Updated 2025-11-21 04:21:31 UTC by Tamarine