React Router and Single Page Application

Vijesh M. Nair
Assistant Professor
Dept. of CSE (AI-ML)

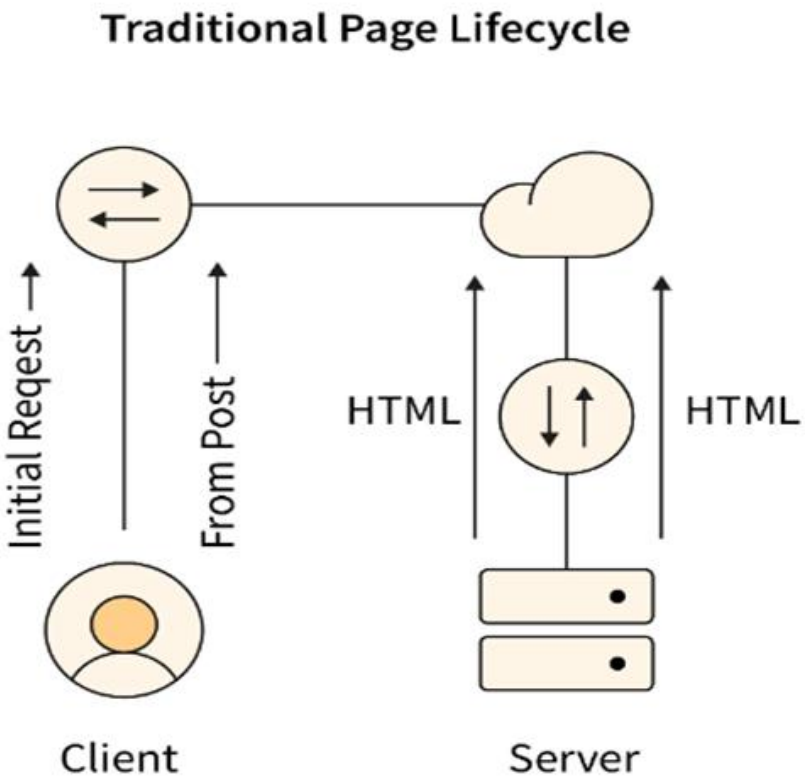
Single Page Applications (SPA)

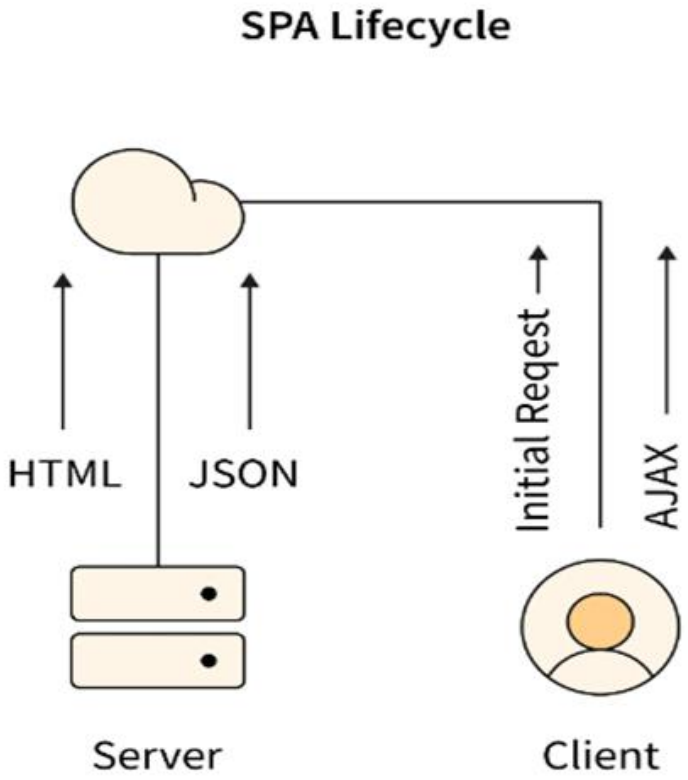
- The upgradation of the javascript performance of web browsers gave birth to Single Page Applications (SPAs).
- A single-page application is a web application that works in a browser and does not require a page reload during use.
- It loads all the files on the initial load and then updates its DOM dynamically on user interaction.
- Unlike the multi-page applications, in single-page applications, only the needed assets are rendered, and the page gets updated.
- As all the assets are not required to be fetched again, the bandwidth also becomes low.
- In SPAs, the logic is executed in the browser rather than the server. This makes the app faster.

Working of Single Page Applications

- In Single page applications, the server sends an **HTML file only** on the **first request**. In all the subsequent requests, **only data** is passed to and from the server.
- Javascript is used to define all the logic. Only the **changes are rewritten** on the page, and the complete page is not rerendered. This creates a **seamless user experience**.
- They are **mobile-friendly**. The frameworks that are used to create SPA can also help in the creation of mobile apps by reusing codes.
- Single-page applications are widely used by **Facebook**, **Gmail**, and **Twitter** are some famous examples of SPAs.

Working of Single Page Applications





React Router

- The Router in React JS is a pure JavaScript package and framework that allows you to use React to create complicated client-side apps.
- Initially launched in 2013, it has become one of the most prominent routing libraries in today's online applications.
- Handles client and server-side routing in React applications.
- It enables the creation of single-page web or mobile apps that allow navigating without refreshing the page.
- It also allows to use browser history features while preserving the right application view.
- A Router in React JS routes using a component-based architecture. It offers various routing components as required by the application.

Step-by-Step Guide to Setup React Router

Step 1: Installing React Router DOM

Step 2: Adding / importing React Router

Step 3: Routing and Rendering Components

1. **Creating Multiple Components**
2. **Define routes**
3. **Link to Navigate to Components**

Step 4: Final Code

1 React Router DOM

- Is an **npm package** that enables you to implement **dynamic routing** in a web app.
- It is a fully-featured client and server-side routing library for React.
- React Router Dom is used to build **single-page applications** i.e. applications that have many pages or components but the page is never refreshed instead the content is dynamically fetched based on the URL.
- This process is called **Routing** and it is made possible with the help of React Router Dom.

Installing React Router

To install react-router in your application write the following command in your terminal

```
// Installing
npm i react-router-dom
```

2 React Router Components

Importing React Router

```
// Importing
import { BrowserRouter, Routes, Route } from "react-router-dom";
```

- **BrowserRouter**: BrowserRouter is a router implementation that uses the HTML5 history API(pushState, replaceState, and the popstate event) to keep your UI in sync with the URL. It is the parent component that is used to store all of the other components.
- **Routes**: It's a new component introduced in the v6 and an upgrade of the component. The main advantages of Routes over Switch are:
  - Relative s and s
  - Routes are chosen based on the best match instead of being traversed in order.
- **Route:** Route is the conditionally shown component that renders some UI when its path matches the current URL.
- **Link:** The link component is used to create links to different routes and implement navigation around the application. It works like an HTML anchor tag.

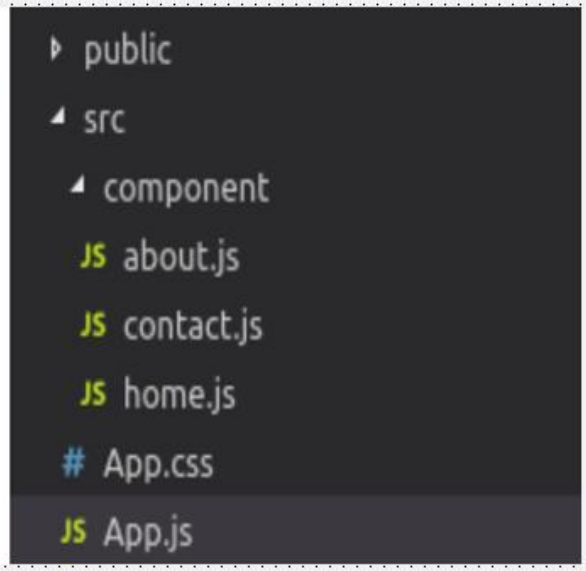
3 Routing and Rendering Components

1. Creating Multiple Components

To use React Router, let us first create a few components in the React application. In your project directory, create a folder named **component** inside the src folder and now add 3 files named.

- home.js
- about.js
- contact.js

Note: Remove the default styling provided with App.css



Routing and Rendering Components

src > component > JS about.js > ...

```
// About.js
import React from 'react';

function About () {
    return <div>
        <h2>APSIT: An institute that grooms industry - ready futuristic technocrats!</h2>

        Read more about us at :
        <a href="https://www.apsit.edu.in/about-us/">
        https://www.apsit.edu.in
        </a>
    </div>
}
export default About;
```

src > component > JS home.js > Home

```
// Home.js
import React from 'react';

function Home (){
    return <h1>Welcome to the world of Technocrats @ APSIT!</h1>
}

export default Home;
```

src > component > JS contact.js > ...

```
// Contact.js
import React from 'react';

function Contact (){
return <address>
            You can find us here:<br />
            A. P. Shah Institute of Technology<br />
            Survey No. 12, 13, Opp. Hypercity Mall, <br />
            Kasarvadavali, Ghodbunder Road, Thane West, Thane, Maharashtra, India 400615
        </address>
}

export default Contact;
```

Routing and Rendering Components

2. Define routes

Our root component is the App.js component, where our React code gets rendered initially. We will be creating all our routes in it.

To define the routes, we are going to use two things: **Routes** and **Route**. Import both components from react-router-dom and use them to route the components. The syntax for defining the routes is as follows:

```
<Routes>
        <Route exact path='/' element={< Home />}></Route>
        <Route exact path='/about' element={< About />}></Route>
        <Route exact path='/contact' element={< Contact />}></Route>
</Routes>
```

Routing and Rendering Components

2. Define routes

```
<Routes>
        <Route exact path='/' element={< Home />}></Route>
        <Route exact path='/about' element={< About />}></Route>
        <Route exact path='/contact' element={< Contact />}></Route>
</Routes>
```

Let's understand this syntax in detail:

- **<Routes>-** The <Routes> component is used to define the different routes that are available in your application. Each individual route is defined using a <Route> component.
- **<Route>-** The <Route> component takes in a path prop, which is a string that defines the URL path that the route should match.
- **path-** The path prop defines the route path.
- **element-** The component prop defines the React component to render when the route path matches the current URL.

Routing and Rendering Components

3. Link to Navigate to Components

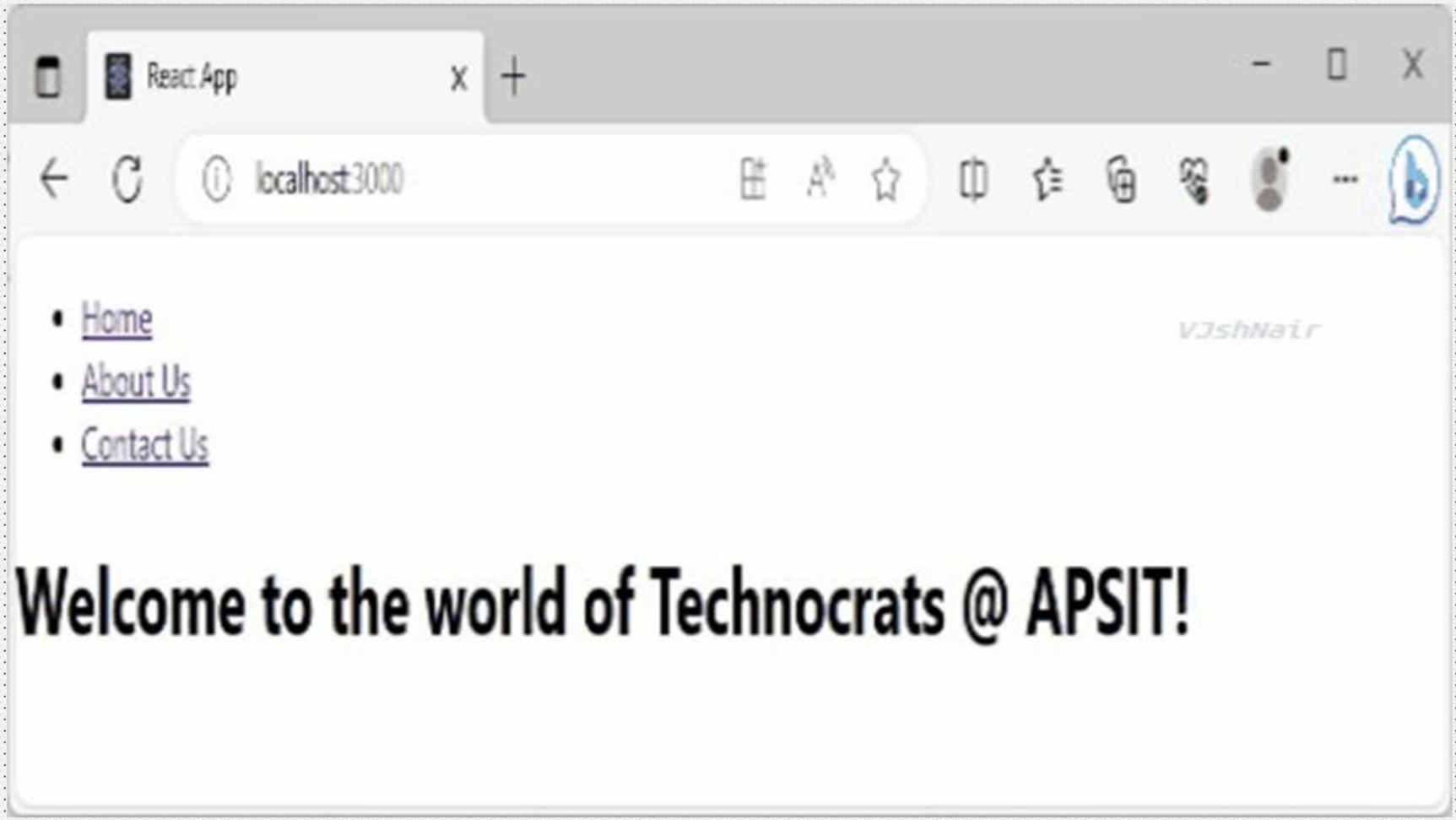
- The <Link> component of react-router-dom to create clickable links that change the URL of the browser and render a different component based on the new URL.
- This component takes in a *to* prop, which is a string that defines the URL path that the link should navigate to.
- When the user clicks on a <Link> component, the URL of the browser will change to the path specified in the *to* prop, and the appropriate component associated with that route will be rendered.

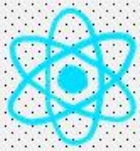
Final Code

src > JS App.js > App > render

```
1   import React, { Component } from 'react';
2   import { BrowserRouter as Router,Routes, Route, Link } from 'react-router-dom';
3   import Home from './component/home';
4   import About from './component/about';
5   import Contact from './component/contact';
6   import './App.css';
7
8   class App extends Component {
9   render() {
10      return (
11      <Router>
12          <div className="App">
13              <ul className="App-header">
14              <li>
15                  <Link to="/">Home</Link>
16              </li>
17              <li>
18                  <Link to="/about">About Us</Link>
19              </li>
20              <li>
21                  <Link to="/contact">Contact Us</Link>
22              </li>
23              </ul>
24          <Routes>
25                  <Route exact path='/' element={< Home />}></Route>
26                  <Route exact path='/about' element={< About />}></Route>
27                  <Route exact path='/contact' element={< Contact />}></Route>
28          </Routes>
29          </div>
30      </Router>
31  );
32  }
33  }
34
35  export default App;
```

Output


React App
localhost:3000
Home
About Us
Contact Us
VJshNair
Welcome to the world of Technocrats @ APSIT!



Thank You!