**Building a Simple Single Page Application (SPA) with React.js**

**Introduction to React.js**

React.js is a **JavaScript library** used for building user interfaces, especially **Single Page Applications (SPA)**. It allows developers to create reusable components and manage the user interface efficiently.

**What is a Single Page Application (SPA)?**

A Single Page Application (SPA) is a web application that loads a single HTML page and dynamically updates the content without reloading the entire page. This makes the app faster and smoother.

### ****What is an SPA (Single Page Application)?****

A **Single Page Application (SPA)** is a web application that **loads a single HTML page** and dynamically updates the content **without reloading the entire page** when the user navigates between different views.

### ****How Does an SPA Work?****

1. **Initial Page Load**
   * When you open http://localhost:3000/, the browser loads the main index.html file along with JavaScript files (React app).
   * The React application **renders only one page**, which contains all the necessary logic to dynamically update the view.
2. **Navigation Without Refresh**
   * When you click on a navigation link (e.g., **"Profile"** or **"Contact Us"**), React Router **updates the URL** but does not reload the entire page.
   * Instead of making a request to the server to fetch a new HTML file, **React dynamically updates the content** inside the existing page using JavaScript.
3. **Why No Page Refresh?**
   * Traditional websites send a request to the server and reload the entire page.
   * In an SPA, React **intercepts** the navigation request, prevents the full page reload, and only updates the **necessary components** on the page.

### ****How Is This Implemented in Your React App?****

In App.js, you used **React Router**:

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

* <Router>: Wraps the app to handle routing without reloading the page.
* <Link>: Used instead of <a> tags to navigate without refreshing.
* <Routes> and <Route>: Define which component to render when the URL changes.

For example:

<Link to="/profile">Profile</Link>

* Clicking **"Profile"** updates the URL (/profile).
* React Router detects the change and **renders the Profile component** inside the current page.
* The page **does not refresh**, only the view updates.

### ****Benefits of SPA****

✅ **Faster navigation** (no full page reload)  
✅ **Smooth user experience** (like a desktop app)  
✅ **Reduced server load** (fewer requests)  
✅ **Better performance** (only load necessary components)

This is why **your app does not refresh when navigating**—React handles the routing internally! 🚀

### ****Real-Time Examples of SPA (Single Page Applications)****

Here are some real-world applications that use **SPA architecture** to provide seamless navigation **without refreshing the page**:

### ****1️⃣ Gmail****

🔹 **How it works**:

* When you click on an email, the URL updates, but the inbox does not reload.
* Only the email content area changes dynamically.
* Clicking **Compose** opens a modal without reloading the page.

🔹 **Why it’s an SPA**:

* Only parts of the page update dynamically.
* Fast and smooth experience like a desktop app.

### ****2️⃣ Facebook****

🔹 **How it works**:

* Clicking on a friend's profile updates the URL, but the page does not refresh.
* New posts load dynamically as you scroll (infinite scrolling).
* Chat messages update in real-time without a page reload.

🔹 **Why it’s an SPA**:

* Keeps the user experience smooth.
* Only updates necessary parts instead of reloading the whole page.

In the program we will use

1. **React Router** to switch between pages.
2. **Bootstrap** for styling the navigation menu.
3. **React Components** (Home.js, Profile.js, ContactUs.js) are reusable blocks of code.

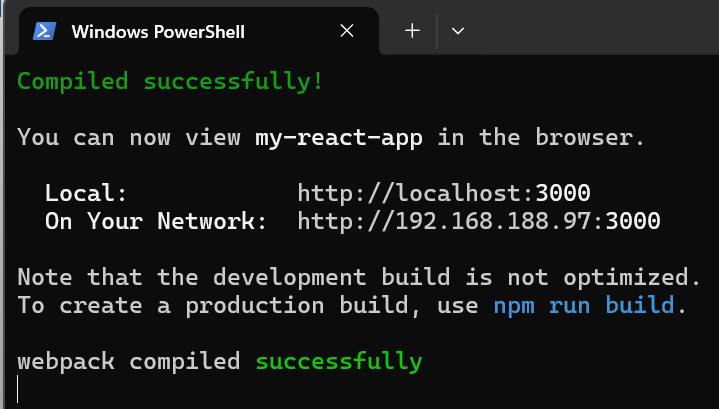
**Step 1: Install React.js**

Before we start coding, we need to set up a React project.

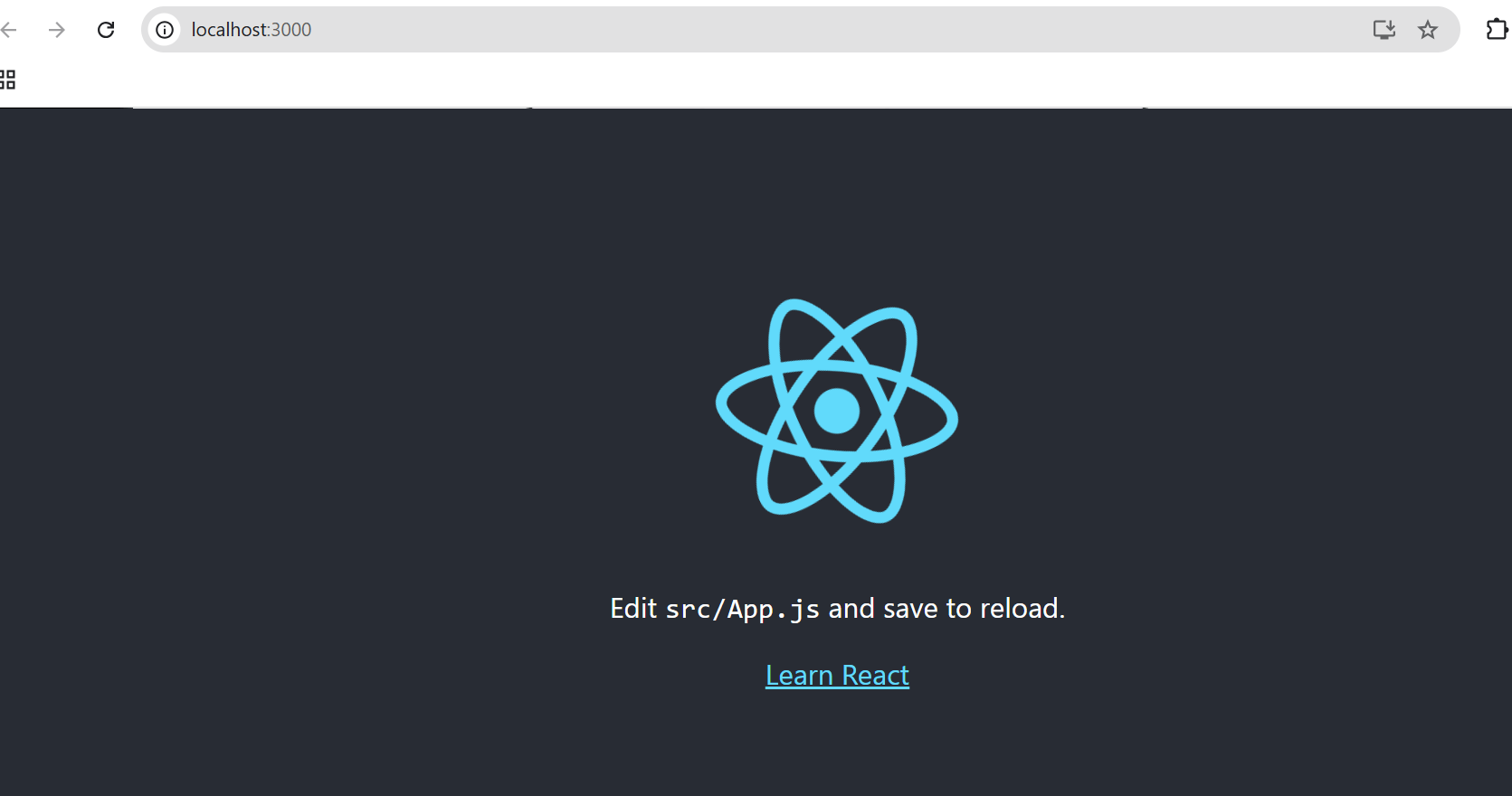
1. Navigate to a convenient folder where you want to create the project🡪 **Open a terminal**
2. Run the following command:
3. npx create-react-app my-react-app
4. if scripts not allowed,error is showing then run the command first🡪

Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass

1. Once the installation is complete, navigate into the project folder:
2. cd my-react-app
3. Start the development server:
4. npm start



This will open a new browser window at [**http://localhost:3000/**](http://localhost:3000/) with the default React welcome page.

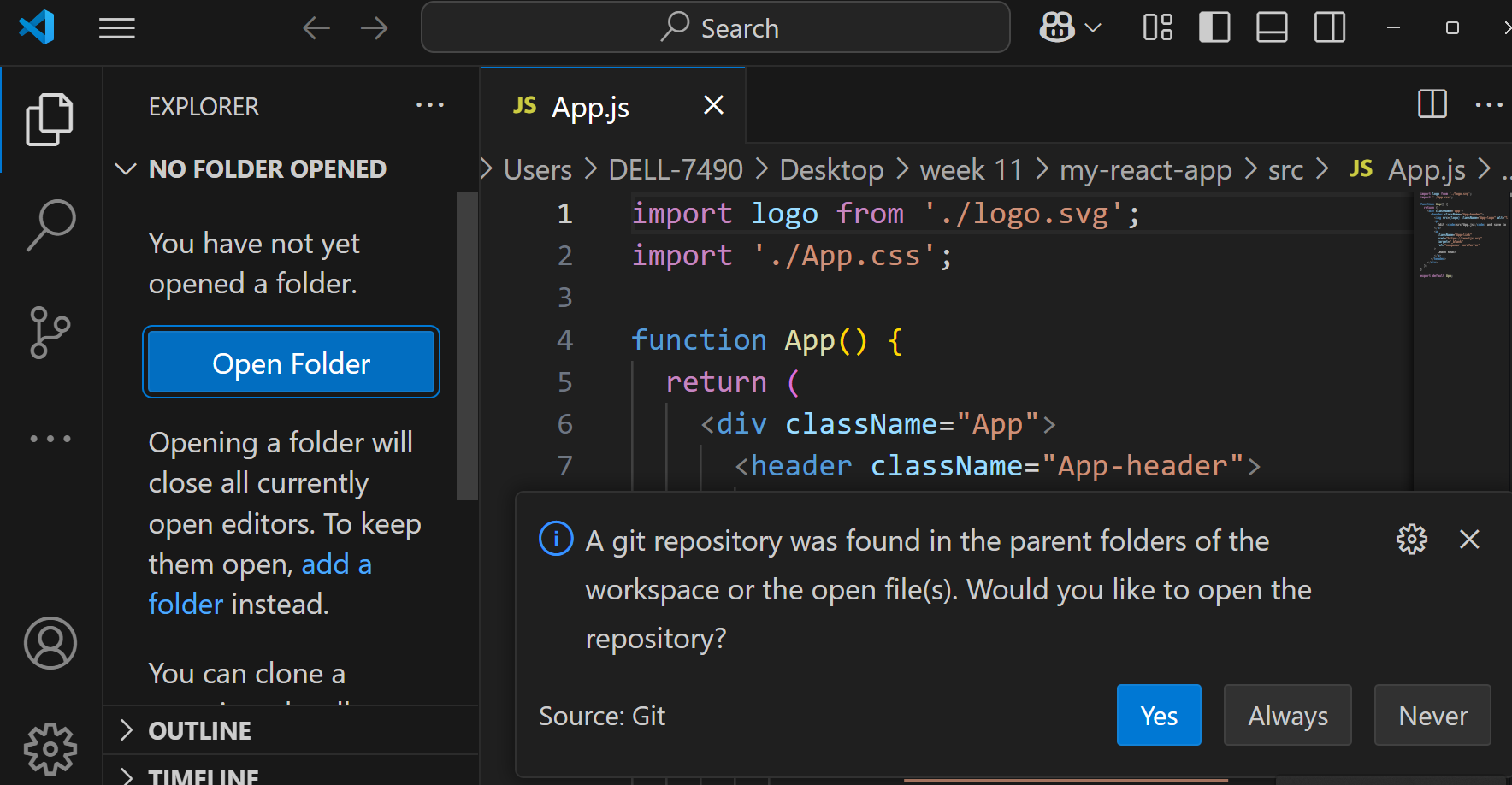


**Step 2: Install Dependencies**

**Where to put Step 2?**

* **Open the project folder:** Navigate to your React app's directory (e.g., my-react-app).
* **Modify the App.js file:**
  + Go to src/App.js.

You will output as :



*This message in* ***VS Code*** *appears because a* ***Git repository*** *was detected in the parent folders of your workspace.*

### *What should you do?*

*It depends on your needs:*

1. ***If you want to use Git for version control*** *(recommended):*
   * *Click* ***"Yes"*** *to open the repository.*
   * *This allows you to track changes, commit updates, and push code to platforms like GitHub.*
2. ***If you don't need Git or it's not relevant right now:***
   * *Click* ***"Never"*** *to disable this prompt permanently.*
   * *Click* ***"Always"*** *if you always want it to open in the future.*

***Best Option?*** *If you’re unsure but might use Git later,* ***click "Yes"*** *for now. It won’t affect your project but will allow Git-based operations if needed.*

* Edit the file to implement Step 2 (routing, API call, or other enhancements).
* **Save changes and restart the app if necessary.**

### ****Stop the Server and Run the Command****

1. In your current terminal (PowerShell), press **Ctrl + C** to stop the React development server.

npm install react-router-dom

1. After installation, restart your React app:

npm start

We need to install **React Router** and **Bootstrap** to build our SPA.

Run these commands inside the project folder:

npm install react-router-dom bootstrap react-bootstrap

Now, we are ready to start coding!

**Step 3: Create the Required Files**

In the src folder, create the following files:

* index.js
* Layout.js
* Home.js
* Profile.js
* ContactUs.js

All these files though have js extension they will be holding jsx code.

### ****What is JSX? Is it Different from JavaScript?****

### ****🔹 JSX (JavaScript XML)****

JSX stands for **JavaScript XML**. It is a **syntax extension** for JavaScript that looks similar to HTML. It allows developers to write UI components using an HTML-like syntax inside JavaScript.

📌 **Example of JSX:**

Jsx-🡪

const element = <h1>Hello, World!</h1>;

* This looks like **HTML**, but it is actually **JSX**.
* JSX makes it easier to create and structure UI elements in React.
* Under the hood, JSX gets **converted into JavaScript**.

### ****🔹 How JSX Works (Behind the Scenes)****

When JSX is used, it **compiles down to plain JavaScript** using Babel (a JavaScript compiler).

📌 **JSX Code:**

Jsx🡪

const element = <h1>Hello, World!</h1>;

📌 **How Babel Converts it to JavaScript:**

Js-🡪

const element = React.createElement("h1", null, "Hello, World!");

* React.createElement("h1", null, "Hello, World!") creates a virtual DOM element.
* React then updates the real DOM efficiently.

**Step 4: Code Implementation**

1. **Modify/replace app.js in src file with this code**

import React from "react";

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

import Home from "./home";

import Profile from "./profile";

import ContactUs from "./contactus";

function App() {

  return (

    <Router>

      <div>

        <nav>

          <ul>

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

            <li><Link to="/profile">Profile</Link></li>

            <li><Link to="/contactus">Contact Us</Link></li>

          </ul>

        </nav>

        <Routes>

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

          <Route path="/profile" element={<Profile />} />

          <Route path="/contactus" element={<ContactUs />} />

        </Routes>

      </div>

    </Router>

  );

}

export default App;

## ****🔹 Step-by-Step Explanation****

### ****1️⃣ Importing Dependencies****

import React from "react";

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

import Home from "./home";

import Profile from "./profile";

import ContactUs from "./contactus";

* **import React from "react";**
  + This imports **React**, which is required for every React component.
* **import { BrowserRouter as Router, Route, Routes, Link } from "react-router-dom";**
  + This imports functions from **React Router**, which helps in **client-side routing**.
  + **Router** → Wraps the app to enable routing.
  + **Routes** → A container that holds all route definitions.
  + **Route** → Defines the individual paths and their components.
  + **Link** → Used for navigation (like <a> in HTML but without full page reload).
* **import Home from "./home";**
  + This imports the Home.js component.
* **import Profile from "./profile";**
  + This imports the Profile.js component.
* **import ContactUs from "./contactus";**
  + This imports the ContactUs.js component.

### ****2️⃣ Defining the**** App ****Component****

function App() {

* This defines the App function, which is a **React component**.

### ****3️⃣ Wrapping Everything Inside**** <Router>

<Router>

* This enables **React Router** for the app, allowing navigation without reloading the page.

### ****4️⃣ Creating a Navigation Menu****

<nav>

<ul>

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

<li><Link to="/profile">Profile</Link></li>

<li><Link to="/contactus">Contact Us</Link></li>

</ul>

</nav>

* **<nav>** → Represents a navigation bar.
* **<ul>** → An unordered list for links.
* **<Link to="/">Home</Link>**
  + This is a **React Router link**.
  + It navigates to the **Home page (/)** without reloading the whole site.
* **<Link to="/profile">Profile</Link>**
  + Navigates to /profile.
* **<Link to="/contactus">Contact Us</Link>**
  + Navigates to /contactus.

🚀 **This is how a Single Page Application (SPA) works** – it changes views **without refreshing the browser**.

### ****5️⃣ Defining Routes****

<Routes>

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

<Route path="/profile" element={<Profile />} />

<Route path="/contactus" element={<ContactUs />} />

</Routes>

* **<Routes>** → This acts as a container for all **routes**.
* **<Route path="/" element={<Home />} />**
  + When the URL is /, the Home component is rendered.
* **<Route path="/profile" element={<Profile />} />**
  + When the URL is /profile, the Profile component is rendered.
* **<Route path="/contactus" element={<ContactUs />} />**
  + When the URL is /contactus, the ContactUs component is rendered.

### ****6️⃣ Exporting the App Component****

export default App;

* This makes the App component available for import in index.js.

## ****📌 Real-World Example****

Think of a **Single Page Application (SPA)** like **Facebook**:

* When you **click on "Profile"**, Facebook updates the view **without refreshing the page**.
* When you **click on "Home"**, it loads the home page dynamically **without a page reload**.

Your app behaves the same way!

* Clicking **"Profile"** loads profile.js dynamically.
* Clicking **"Contact Us"** loads contactus.js dynamically.
* The **URL changes**, but the page **does not reload**.

1. **Modify index.js (Entry Point of the App)**

***We got two index.js which one to modify:***

### *****Explanation of Both***** *index.js* *****Files:*****

1. ***src/index.js (You Need to Modify This)***
   * *This is the entry point for your React application.*
   * *It contains the ReactDOM render method and imports the root component (App.js).*
   * *If you are adding* ***React Router****, this is where you will wrap the <App /> component with <BrowserRouter>.*
2. ***public/index.html (Do Not Modify for Routing)***
   * *This is the HTML file that React injects content into.*
   * *It contains the <div id="root"></div>, where React mounts the app.*
   * *You don’t need to modify this for React Router setup*.

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

## ****🔹 Step-by-Step Explanation****

### ****1️⃣ Import Statements****

import React from 'react';

import ReactDOM from 'react-dom/client';

* **import React from 'react';**
  + Imports the **React library** to use React features like components.
* **import ReactDOM from 'react-dom/client';**
  + Imports ReactDOM which handles rendering React components into the actual HTML page.
  + **In React 18+, ReactDOM.createRoot() is used instead of ReactDOM.render()** for better performance.

import './index.css';

* **Imports a CSS file (index.css)** for global styling.
* This applies styles to the entire application.

import App from './App';

* **Imports the main component (App.js)**.
* This is the **entry point** of the app where we define routes and page structure.

import reportWebVitals from './reportWebVitals';

* **Imports a function (reportWebVitals)** used for measuring the app’s performance.
* It’s optional and can be removed if you don’t need performance tracking.

### ****2️⃣ Creating the Root Element****

const root = ReactDOM.createRoot(document.getElementById('root'));

* **document.getElementById('root')** → Finds the <div id="root"></div> inside public/index.html.
* **ReactDOM.createRoot()** → Creates a root where React components will be rendered.

### ****3️⃣ Rendering the App Component****

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

* **root.render(...)** → This tells React to render your app inside <div id="root"></div>.
* **<React.StrictMode>**
  + Used **only in development** to highlight potential issues.
  + Helps detect deprecated APIs, side effects, and unexpected behaviors.

### ****4️⃣ Measuring Performance (Optional)****

reportWebVitals();

* This function is used to **measure the performance of the app**.
* It can log important metrics like:
  + Page Load Time
  + Rendering Speed
  + User Experience Issues
* By default, it does nothing, but you can modify it like this:

reportWebVitals(console.log);

* + This logs performance data in the browser console.

## ****🚀 What Happens When You Run This File?****

1. The browser loads index.html, which contains:

<div id="root"></div>

1. index.js finds this <div id="root"></div>.
2. ReactDOM.createRoot() creates a **React root** inside it.
3. root.render(<App />) inserts the entire App.js component into the page.
4. The app is displayed in the browser!

## ****📌 Summary****

| **Line of Code** | **Purpose** |
| --- | --- |
| import React from 'react'; | Imports React for using JSX and components |
| import ReactDOM from 'react-dom/client'; | Imports ReactDOM to render React components in HTML |
| import './index.css'; | Imports global CSS styles |
| import App from './App'; | Imports the main app component |
| import reportWebVitals from './reportWebVitals'; | Imports a function for performance measurement (optional) |
| const root = ReactDOM.createRoot(document.getElementById('root')); | Finds the root div and prepares it for rendering |
| root.render(<React.StrictMode><App /></React.StrictMode>); | Renders the App component inside the root div |
| reportWebVitals(); | (Optional) Starts performance tracking |

**2. Create Layout.js (Main Layout with Navigation Menu)**

import { Container, Nav, Navbar } from "react-bootstrap";

import { Outlet, Link } from "react-router-dom";

const Layout = () => {

return (

<>

<Navbar bg="primary" data-bs-theme="dark">

<Container>

<Navbar.Brand href="/">React SPA</Navbar.Brand>

<Nav className="me-auto">

<Nav.Link as={Link} to="/">Home</Nav.Link>

<Nav.Link as={Link} to="/profile">Profile</Nav.Link>

<Nav.Link as={Link} to="/contact">Contact Us</Nav.Link>

</Nav>

</Container>

</Navbar>

<Container className="mt-4">

<Outlet />

</Container>

</>

);

};

export default Layout;

**3. Create Home.js (Home Page Component)**

const Home = () => {

return <h1>Welcome to CMRIT Home Page</h1>;

};

export default Home;

**4. Create Profile.js (Profile Page Component)**

const Profile = () => {

return (

<>

<h1>Welcome ABC</h1>

<p>CSE - D</p>

</>

);

};

export default Profile;

**5. Create ContactUs.js (Contact Page Component)**

const ContactUs = () => {

return <h1>CMRIT, Kandlakoya Village, Medchal, Hyderabad</h1>;

};

export default ContactUs;

**Step 5: Run the Application**

1. Make sure you're in the project folder.
2. Start the React development server:
3. npm start
4. Open your browser and visit [**http://localhost:3000/**](http://localhost:3000/).

**Step 6: Test the Application**

* Click on **Home** → Displays Welcome to CMRIT Home Page.
* Click on **Profile** → Displays Welcome ABC and CSE - D.
* Click on **Contact Us** → Displays CMRIT, Kandlakoya Village, Medchal, Hyderabad.

**Final Output (As Seen in the Images)**

1. **Homepage**
2. Welcome to CMRIT Home Page
3. **Profile Page**
4. Welcome ABC
5. CSE - D
6. **Contact Us Page**
7. CMRIT, Kandlakoya Village, Medchal, Hyderabad

**Summary**

**React is used to build modern web applications**.

1. **We created an SPA** that does not refresh the page when navigating.