**1. ReactJS-HOL**

**Objectives**

1. **Define SPA and its benefits:**

A Single-Page Application (SPA) is a web application that loads a single HTML page and dynamically updates that page as the user interacts with the app. This provides a more fluid and desktop-like user experience by avoiding full page reloads.

1. **Define React and identify its working:**

React is a JavaScript library for building user interfaces. It works by creating a component-based structure where each component is a small, reusable piece of code that controls a part of the UI.

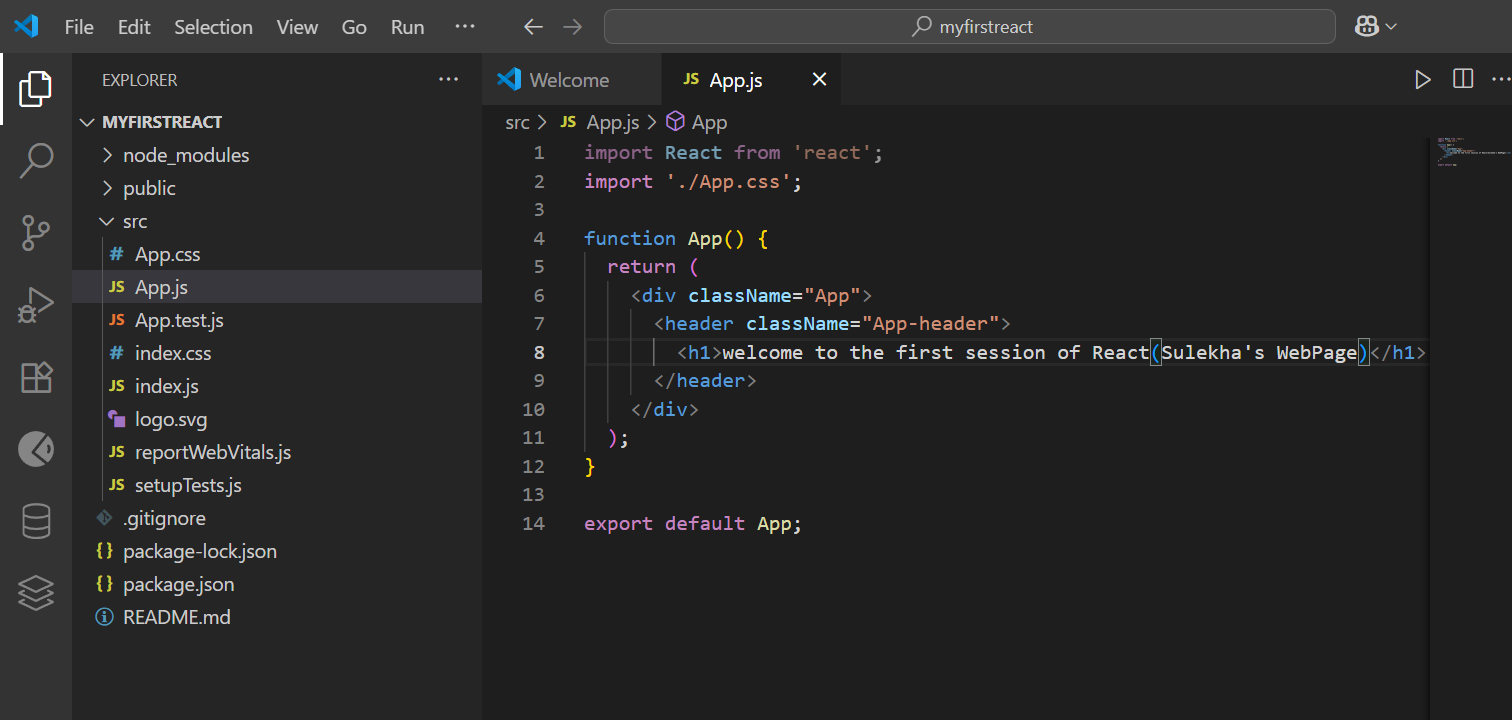
1. **Identify the differences between SPA and MPA:**
   1. SPA (Single-Page Application): Loads a single HTML page and updates content dynamically. Faster and more responsive user experience after the initial load.
   2. MPA (Multi-Page Application): Each user action (e.g., clicking a link) triggers a full page reload from the server. This can be slower and less fluid.
2. **Explain Pros & Cons of Single-Page Application:**
   1. Pros: Improved user experience, faster performance (after initial load), and easier debugging with modern browsers.
   2. Cons: Slower initial load time, SEO challenges (can be mitigated), and requires JavaScript to be enabled.
3. **Explain about React:**

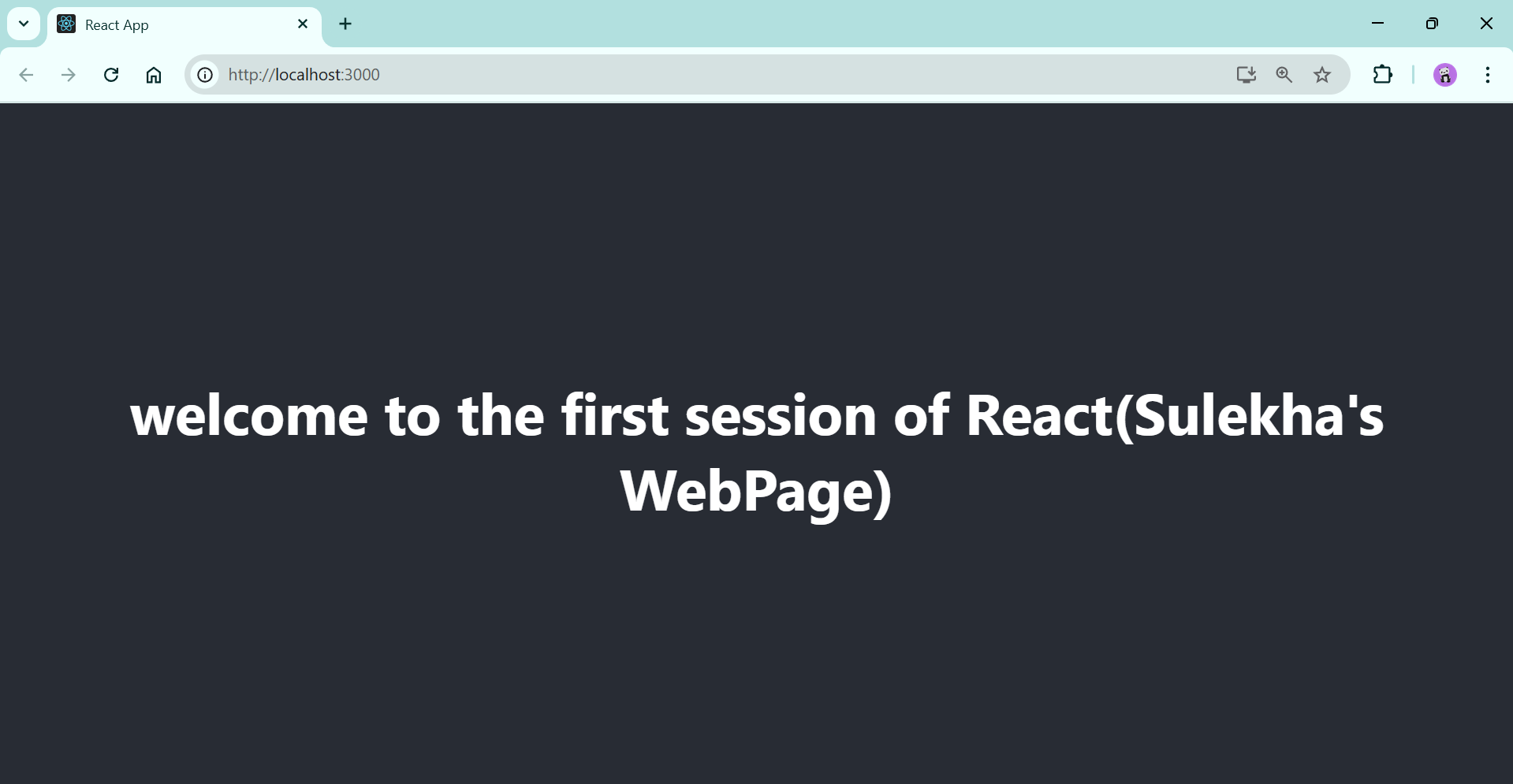
React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called "components."

1. **Explain Features of React:** Component-based architecture, declarative views, JSX syntax, Virtual DOM, and a one-way data flow.
2. **Define Virtual DOM:**

The Virtual DOM is a lightweight copy of the real DOM. When the state of a React component changes, React first updates the Virtual DOM, then efficiently calculates the minimum number of changes needed to update the real DOM. This process, known as "reconciliation," significantly improves performance.

Code:



Output

**2. ReactJS-HOL**

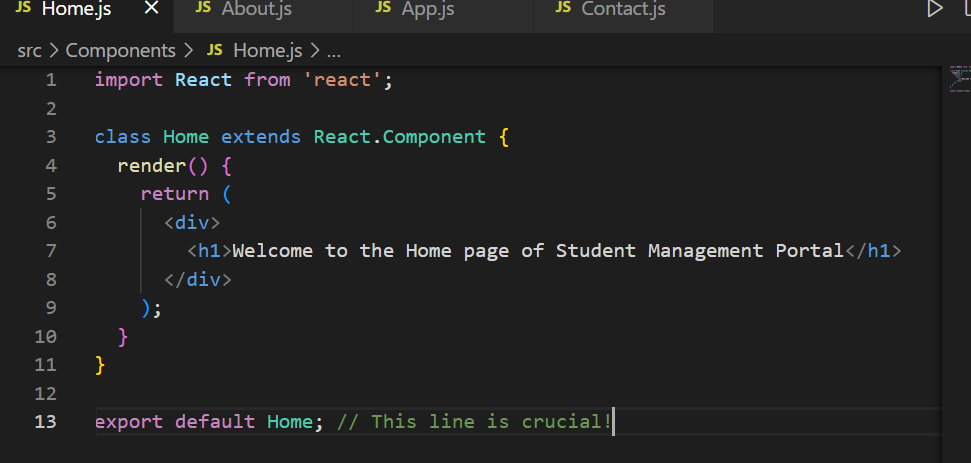
**Objectives**

1. **Explain React components:** A React component is an independent, reusable piece of code that represents a part of a user interface (UI).
2. **Identify the differences between components and JavaScript functions:** React components are specialized functions or classes that take properties (props) as input and return React elements that describe what should appear on the screen. Standard JavaScript functions perform specific tasks and return an output.
3. **Identify the types of components:** The two main types of components in React are Class Components and Function Components.
4. **Explain class component:** Class components are defined using ES6 class syntax, extend React.Component, and have a render() method that returns JSX. They can manage their own state and lifecycle methods.
5. **Explain function component:** Function components are defined using a JavaScript function and are simpler to write. With React Hooks, they can also manage state and handle lifecycle events.
6. **Define component constructor:** For class components, the constructor() is a special method called when a component is created, used to initialize state and bind event handlers. It's crucial to call
7. **Define render() function:** The render() method is the only mandatory method in a class component, responsible for returning the JSX that describes what the component should render. It should be a pure function.

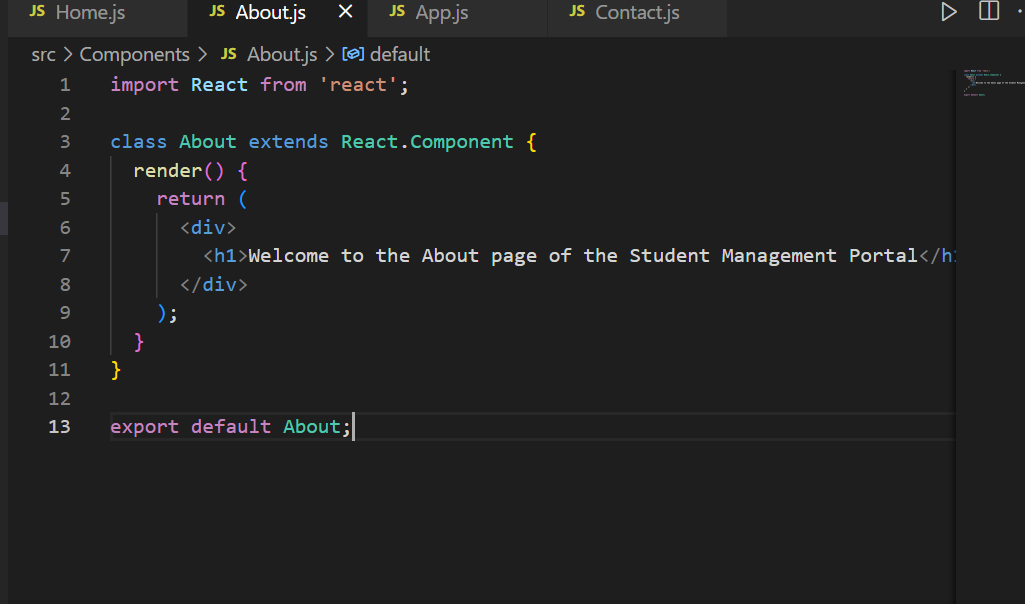
**Student Management Portal**

**Code:**

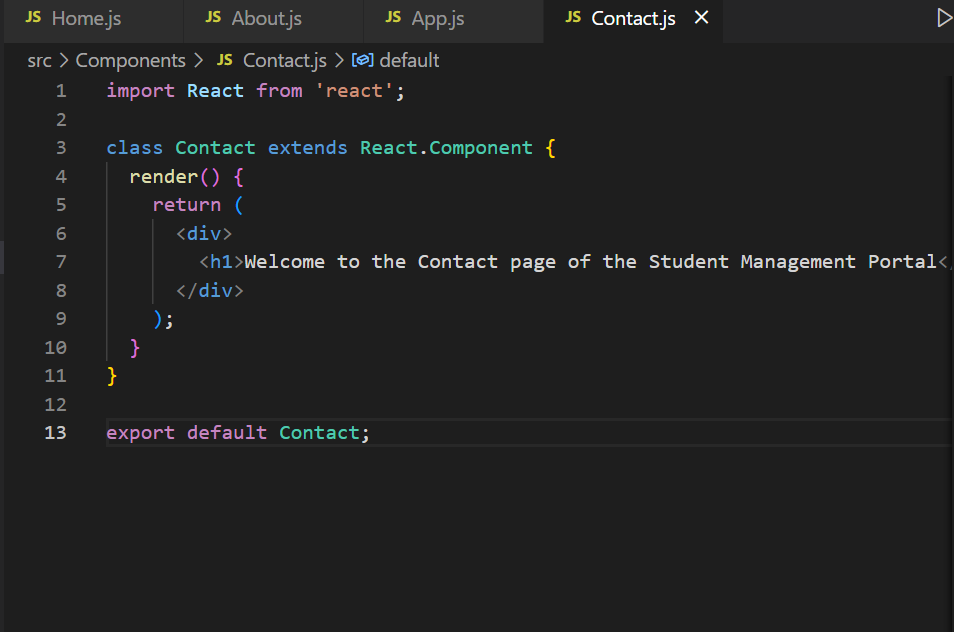
Home.js

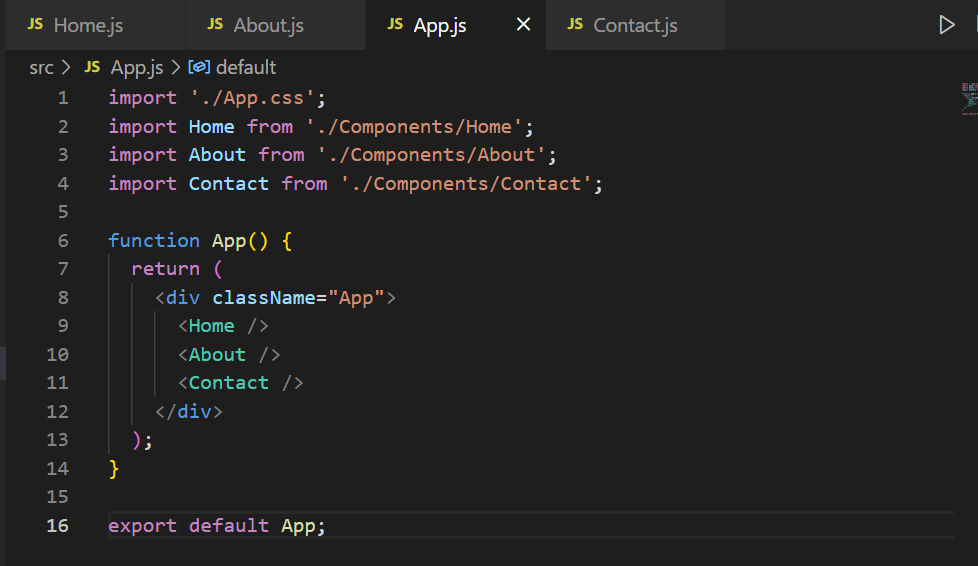


About.js

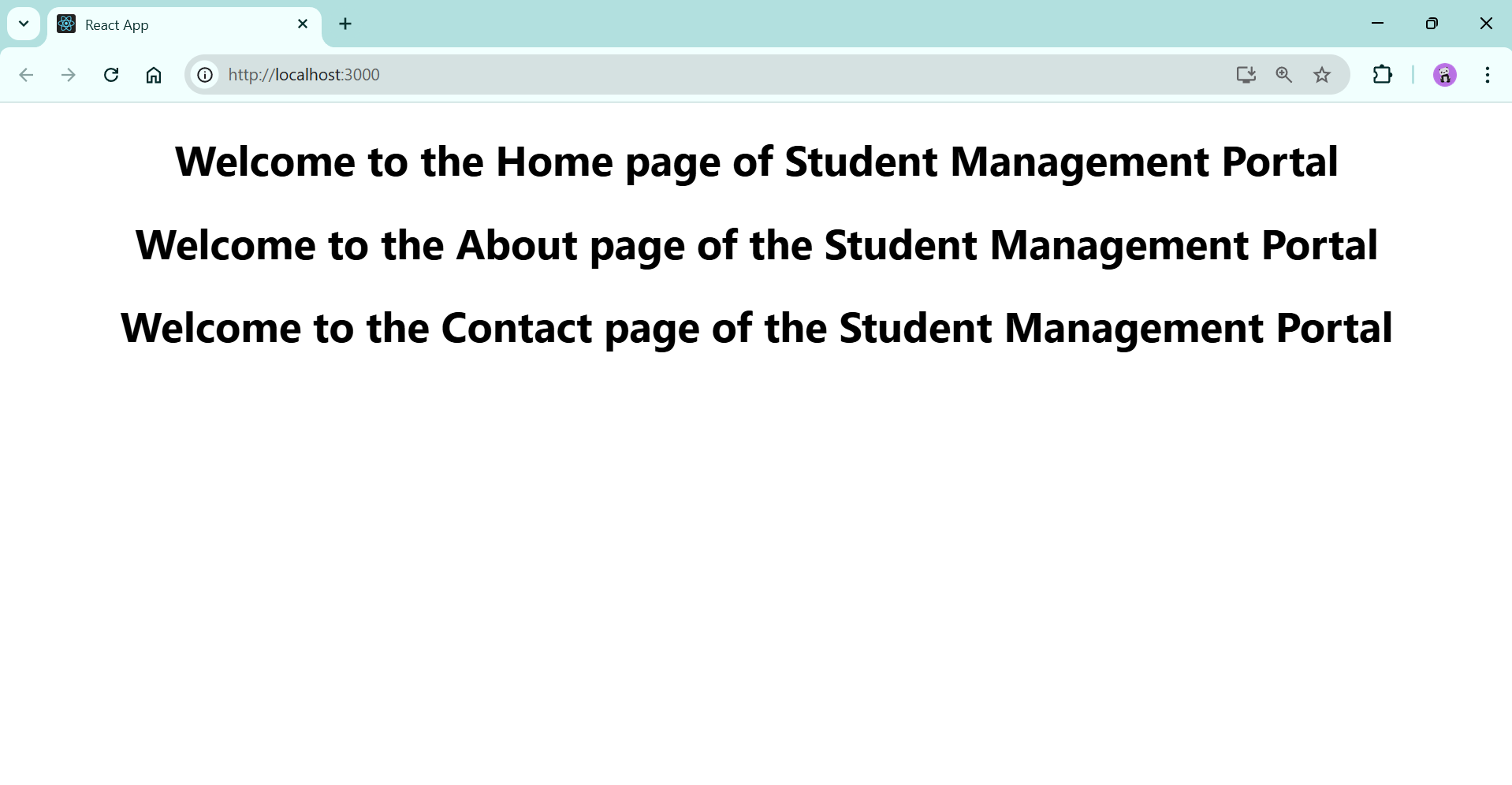


Contact.js



App.js 

**Output**



**3. ReactJS-HOL**

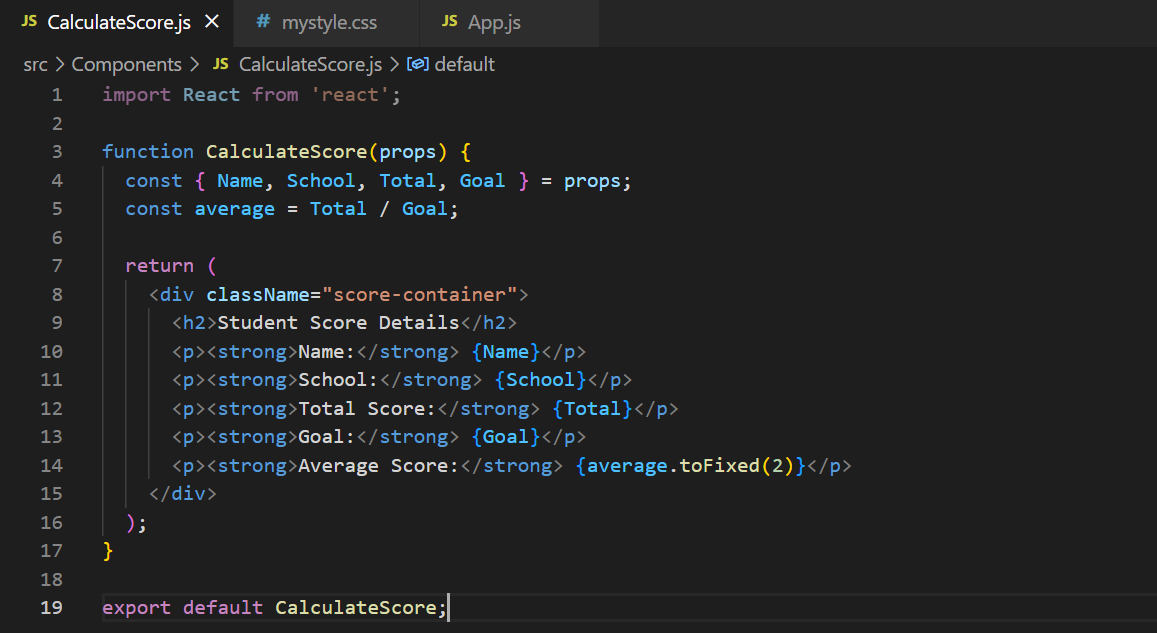
**Objectives**

1. **Explain React components**: Independent, reusable pieces of UI code.
2. **Identify the differences between components and JavaScript functions**: React components are specialized functions or classes that take props and return React elements (UI), while JavaScript functions perform tasks and return outputs.
3. **Identify the types of components**: The two main types are Class Components and Function Components.
4. **Explain class component**: Defined using ES6 class syntax, extending React.Component, and including a render() method. Can manage state and lifecycle methods.
5. **Explain function component**: Defined using a JavaScript function. Simpler to write and, with Hooks, can manage state and lifecycle events.
6. **Define component constructor**: (Relevant to Class Components) A special method called upon component creation, used for state initialization and binding methods.
7. **Define render() function**: The mandatory method in a class component, responsible for returning the JSX that describes the UI. Should be a pure function.

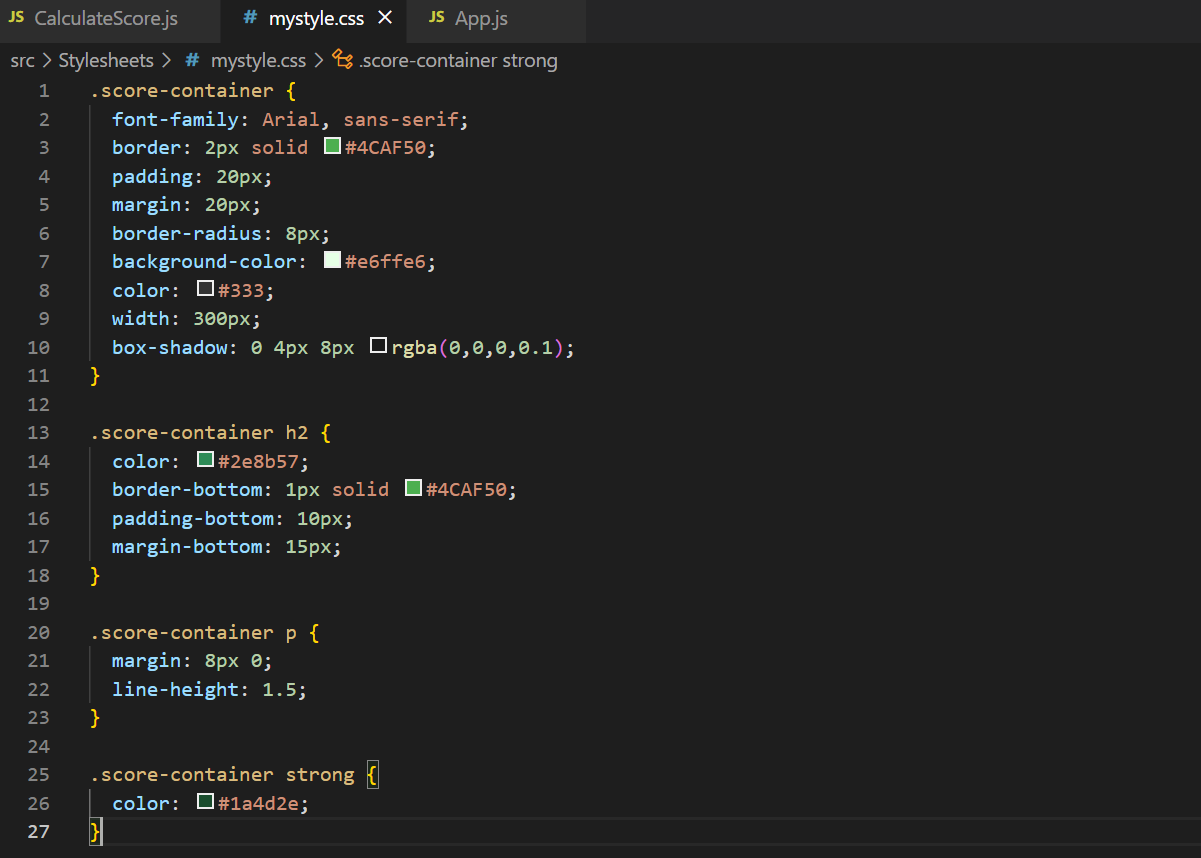
**Student Management Portal to Calculate Score**

**Code:**

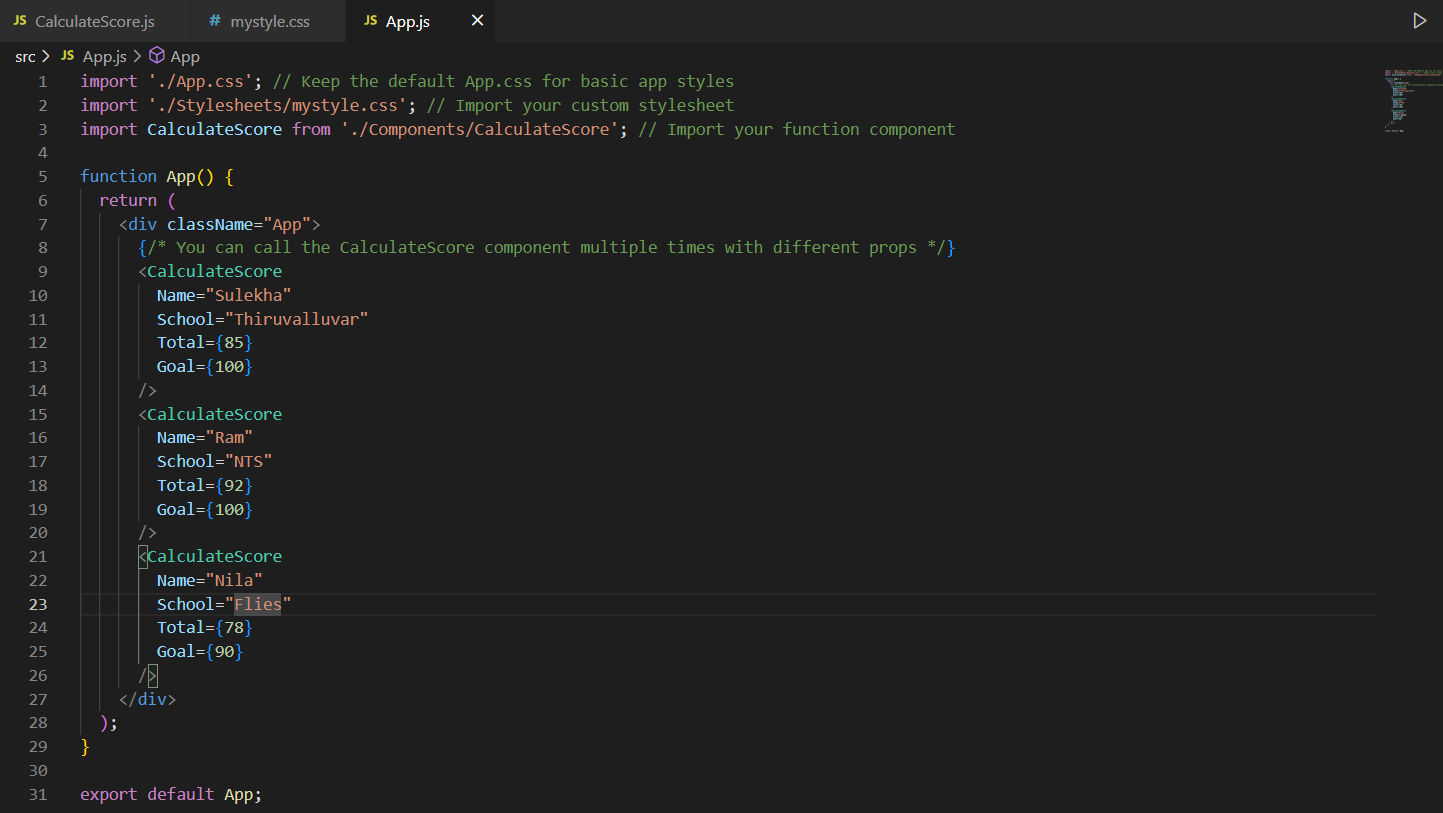
CalculateScore.js

****

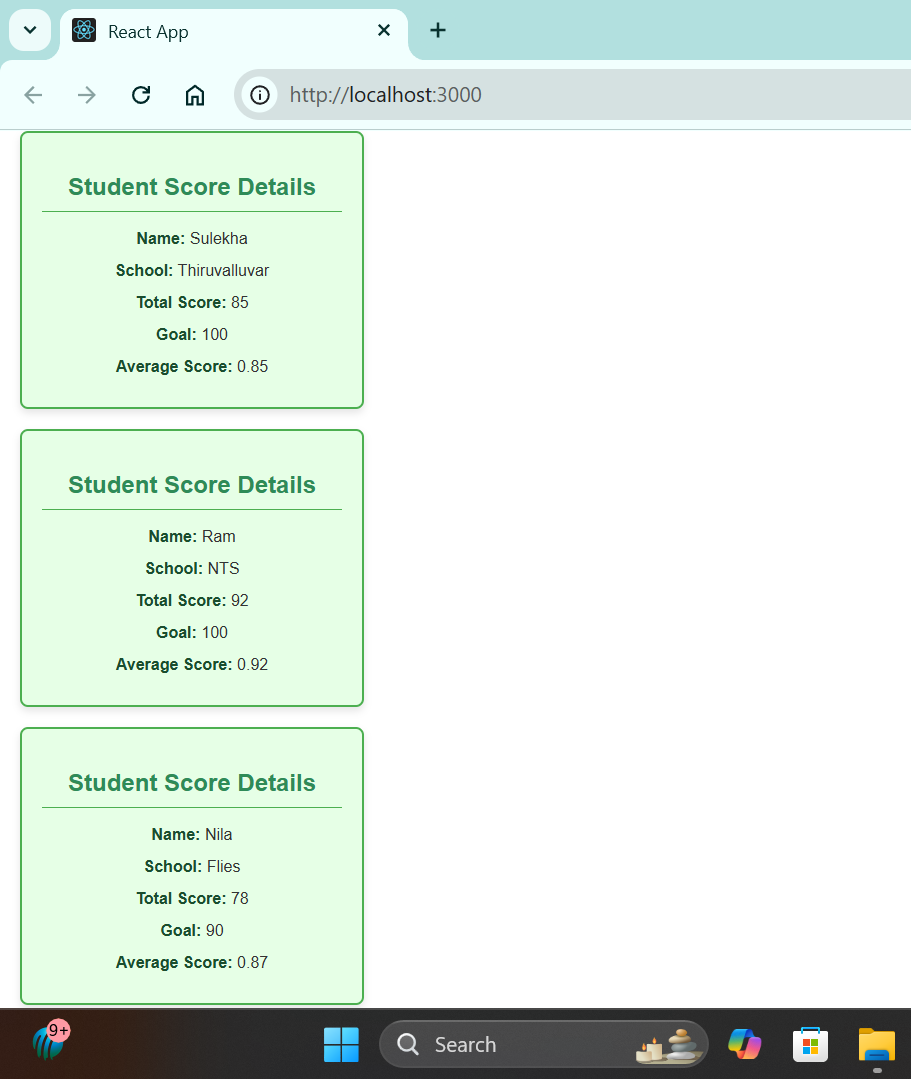
MyStyle.css



App.js



**Output:**

****

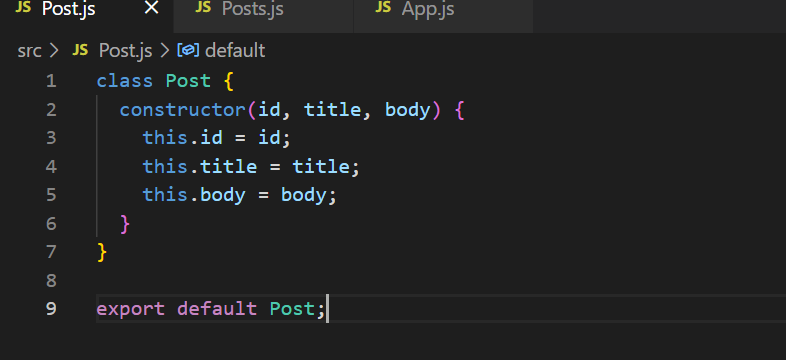
**4. ReactJS-HOL**

**Objectives**

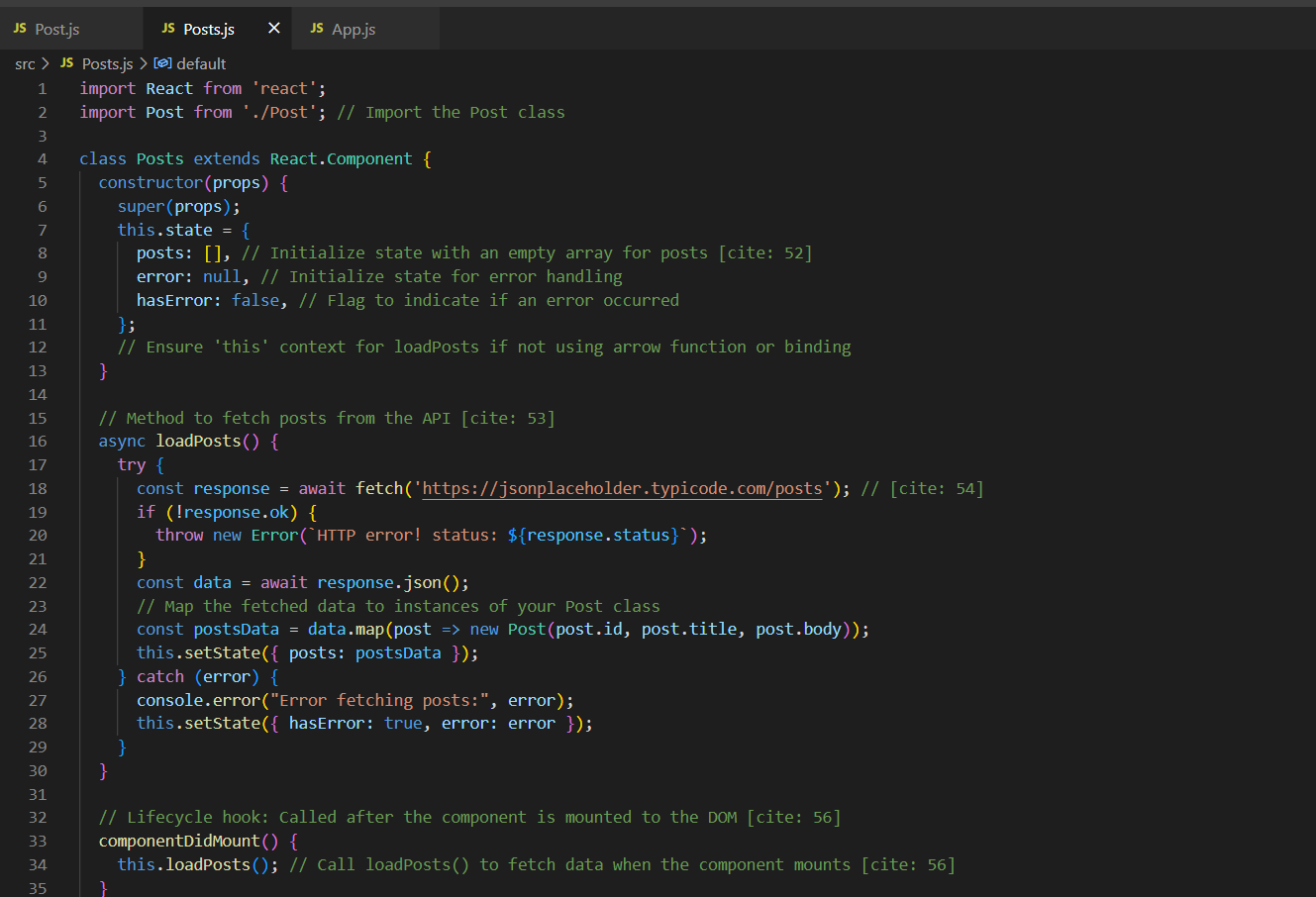
1. **Explain the need and Benefits of component lifecycle**: The component lifecycle refers to the various phases a component goes through from its creation to its destruction. Lifecycle methods (hooks) allow developers to execute code at specific points in a component's life, enabling control over rendering, data fetching, and interaction with the DOM.
2. **Identify various life cycle hook methods**: Key lifecycle methods include constructor(), render(), componentDidMount(), componentDidUpdate(), componentWillUnmount(), and error handling methods like componentDidCatch().
3. **List the sequence of steps in rendering a component**:
   1. **Mounting (Initial Render)**: constructor() -> render() -> React updates DOM -> componentDidMount().
   2. **Updating (Re-renders)**: render() -> React updates DOM -> componentDidUpdate().
   3. **Unmounting (Removal)**: componentWillUnmount().
   4. **Error Handling**: componentDidCatch().

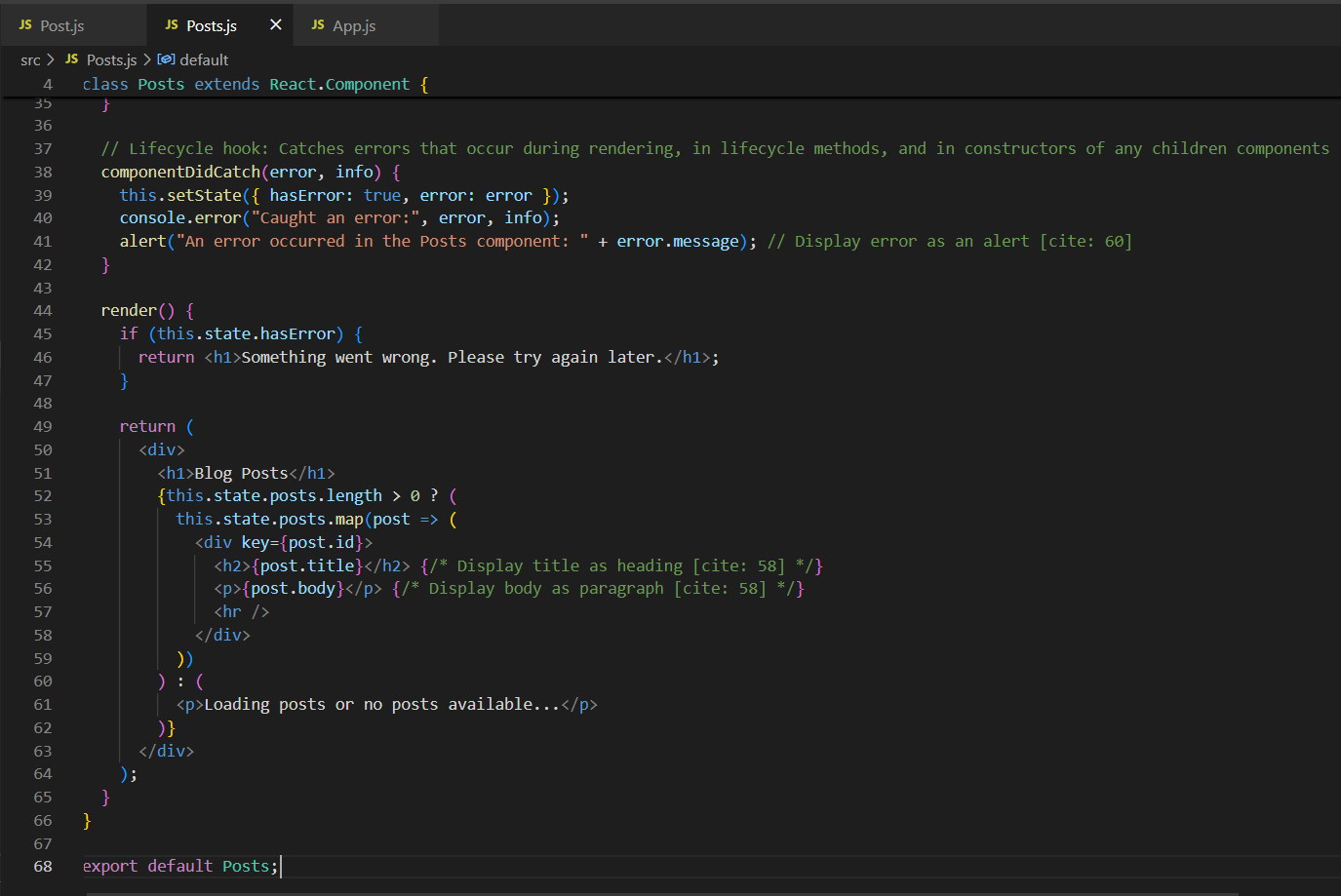
**Blog Application**

Post.js

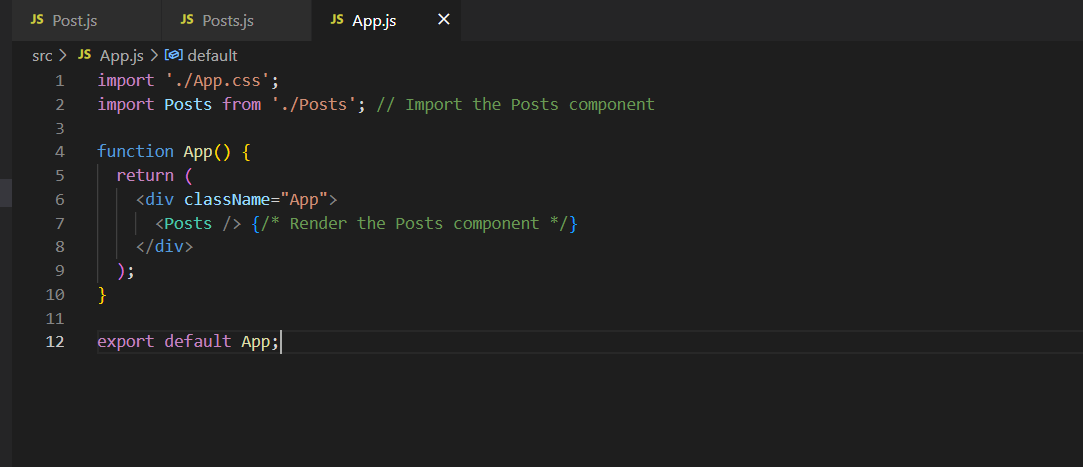


Posts.js

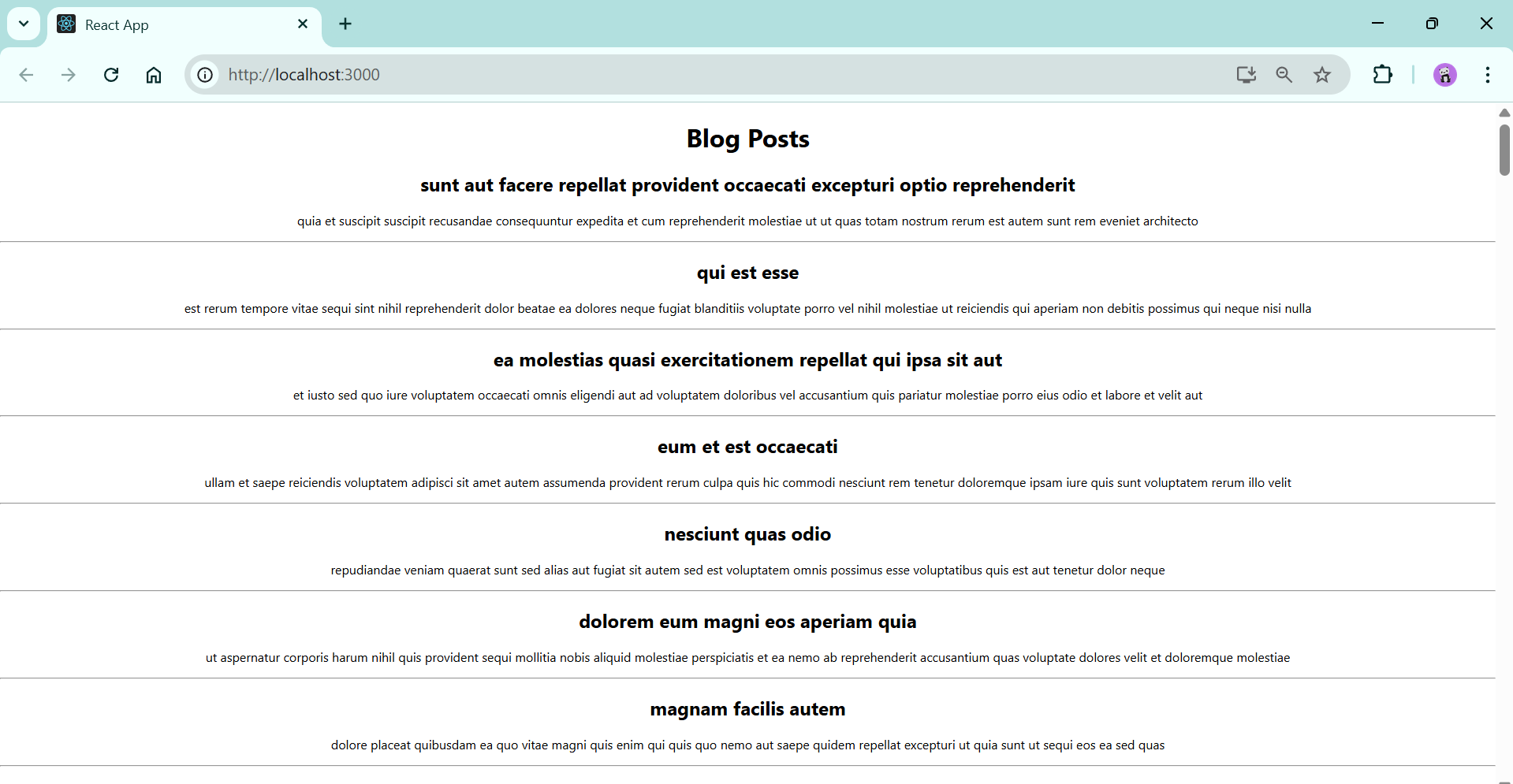
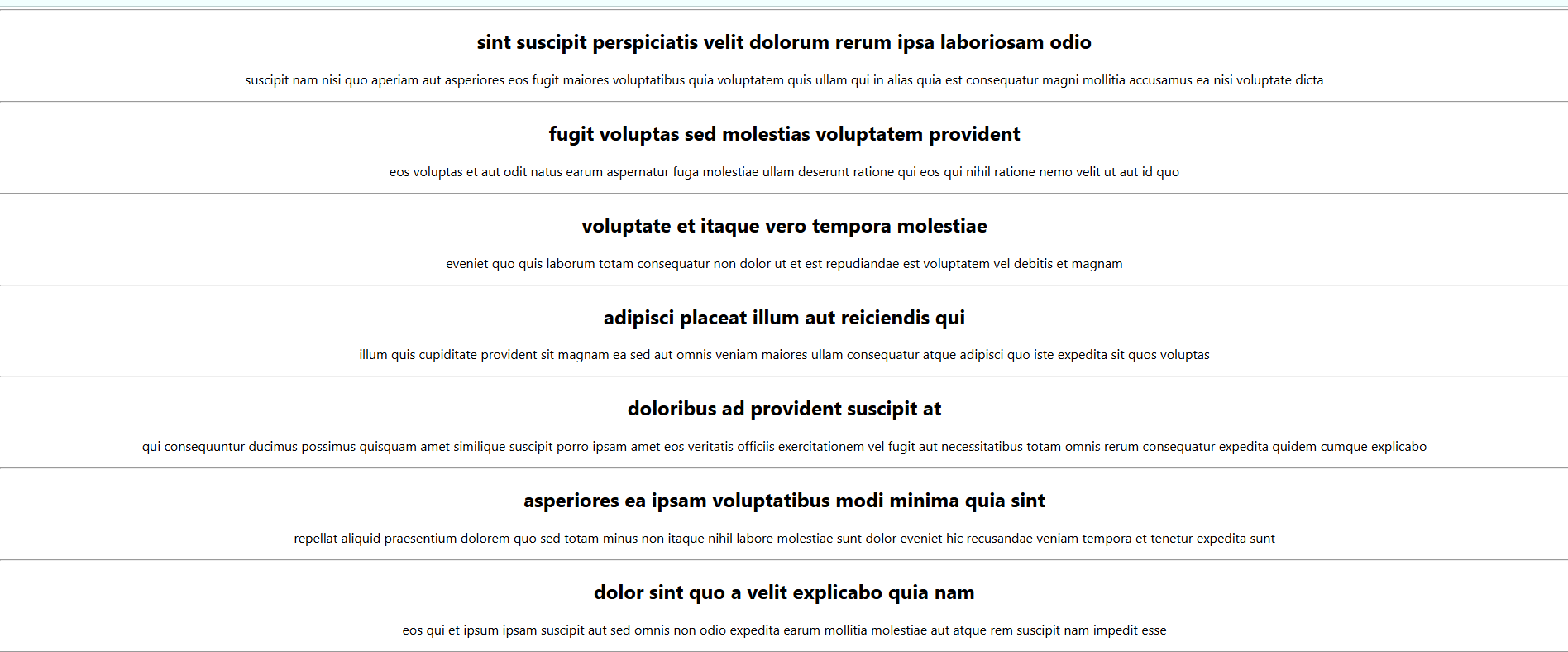




App.js



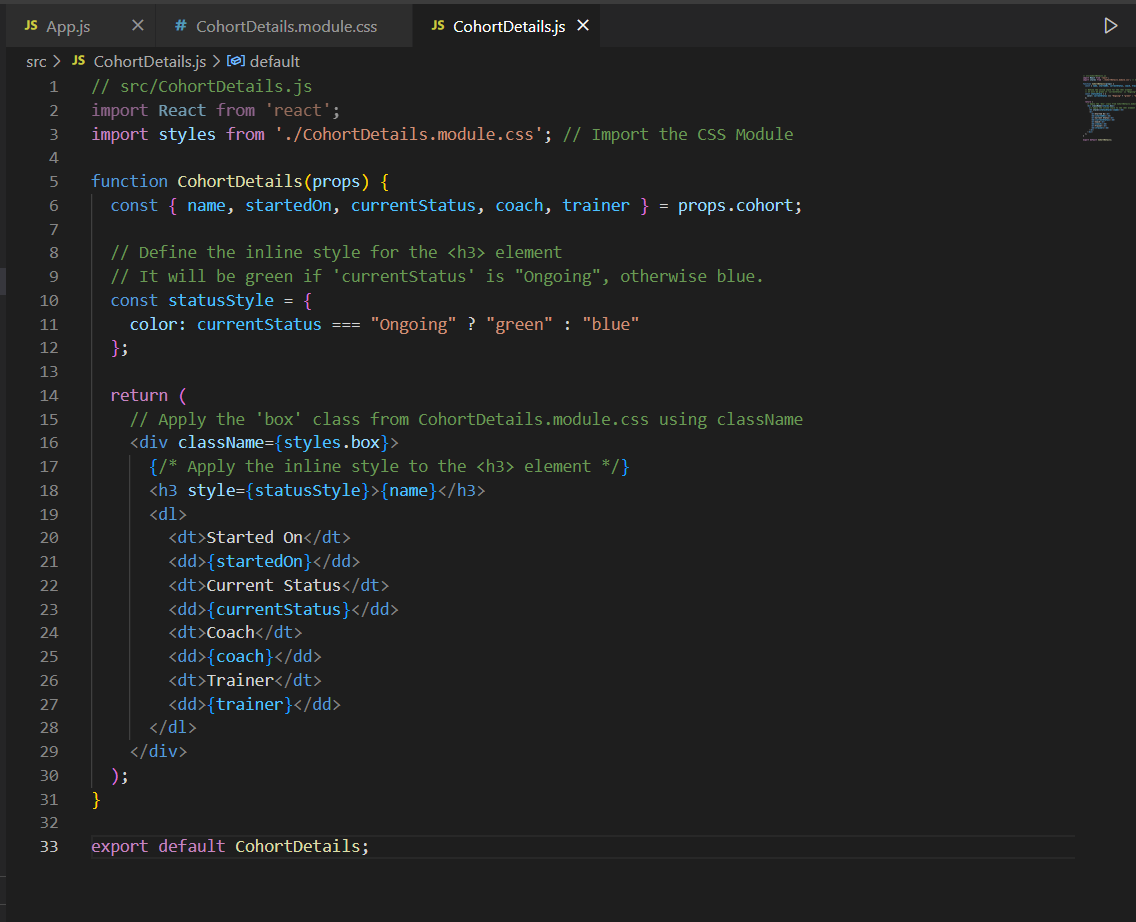
**Output:**

**** **** 

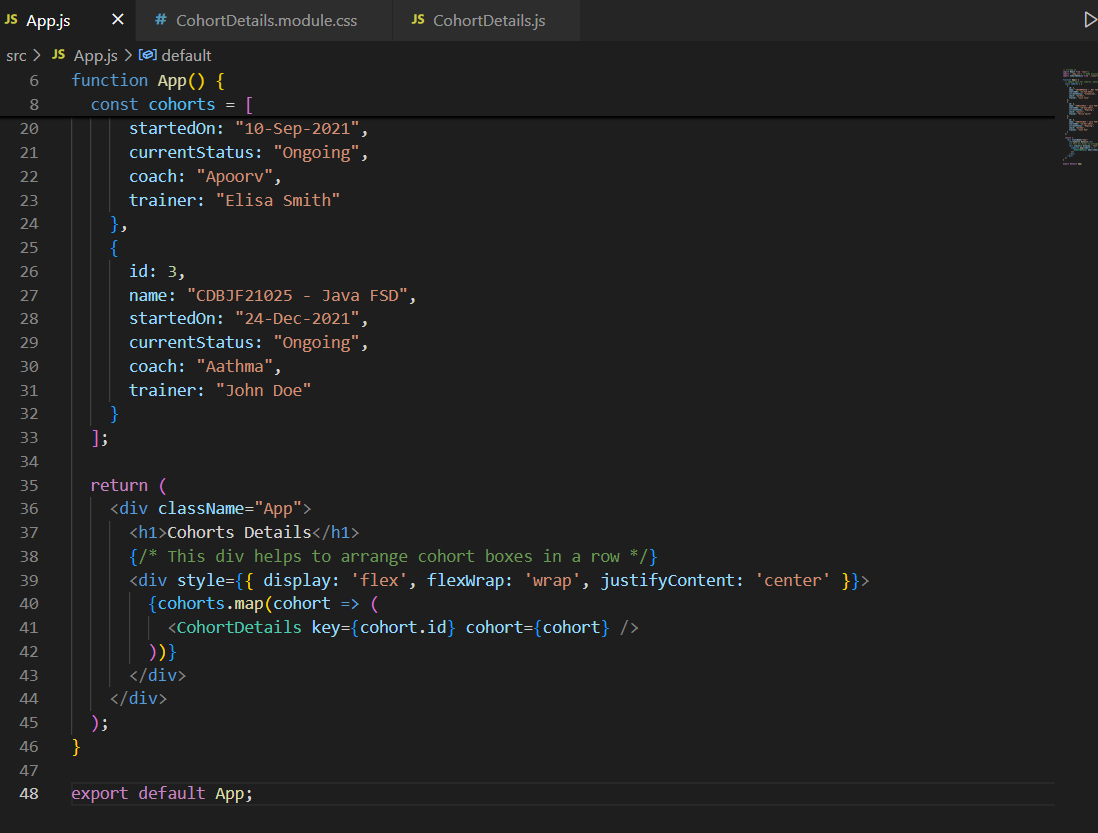
5.ReactJS-HOL

**Code:**

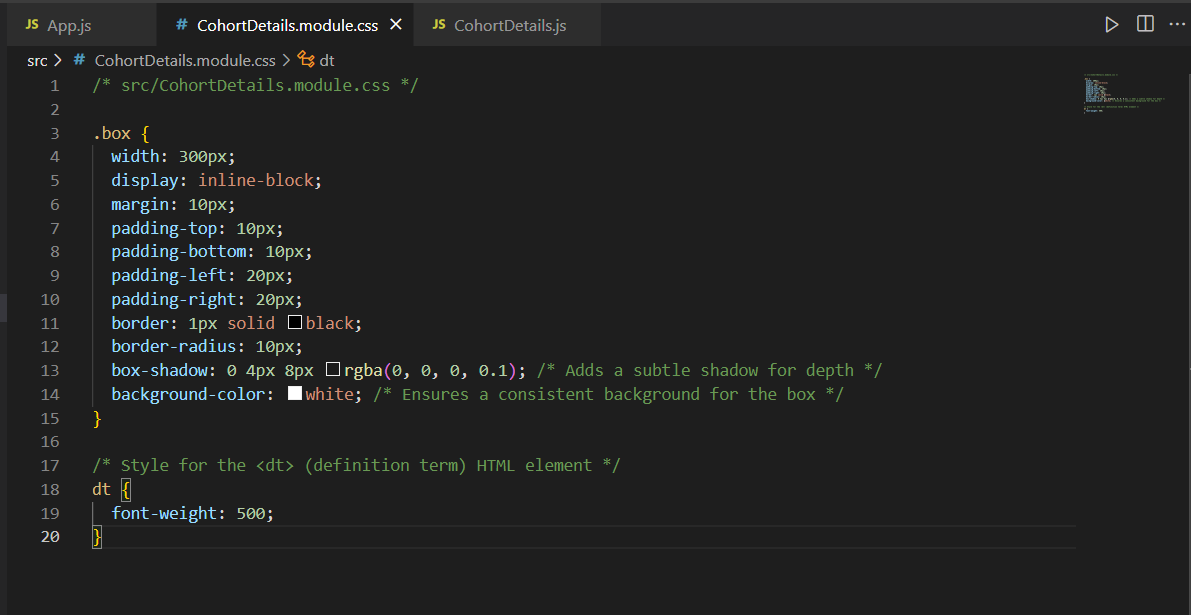
CohortDetails.js



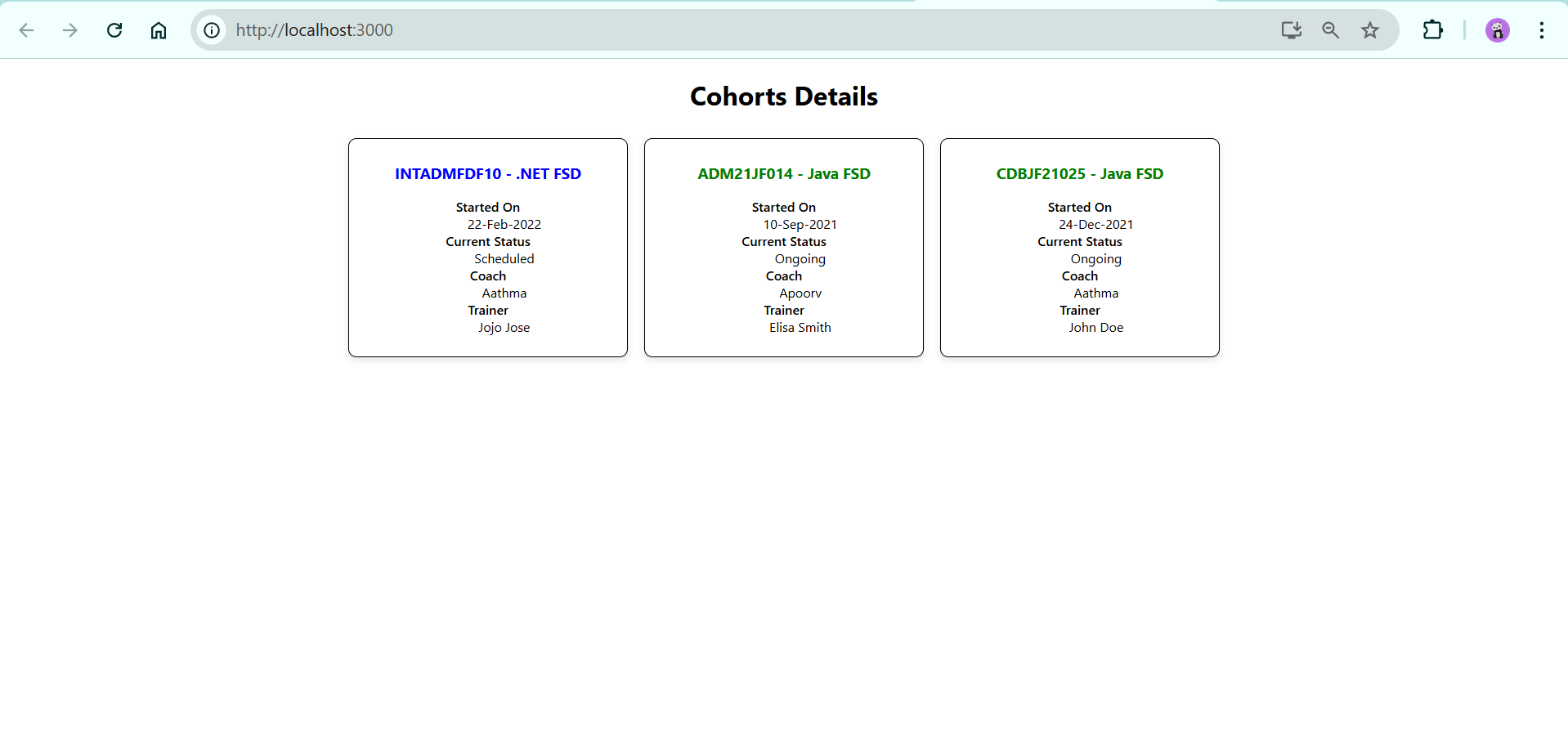
App.js



CohortDetails.modules.css



**Ouput:**

****