**Mandatory Exercises for React Week-6:**

**Objectives**

* Define SPA and its benefits
* Define React and identify its working
* Identify the differences between SPA and MPA
* Explain Pros & Cons of Single-Page Application
* Explain about React
* Define virtual DOM
* Explain Features of React

In this hands-on lab, you will learn how to:

* Set up a react environment
* Use create-react-app

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **30 minutes.**

Create a new React Application with the name “myfirstreact”, Run the application to print “welcome to the first session of React” as heading of that page.

1. To create a new React app, Install Nodejs and Npm from the following link:

<https://nodejs.org/en/download/>

1. Install Create-react-app by running the following command in the command prompt:



1. To create a React Application with the name of “myfirstreact”, type the following command:



1. Once the App is created, navigate into the folder of myfirstreact by typing the following command:



1. Open the folder of myfirstreact in Visual Studio Code
2. Open the App.js file in Src Folder of myfirstreact
3. Remove the current content of “App.js”
4. Replace it with the following:



1. Run the following command to execute the React application:



1. Open a new browser window and type “localhost:3000” in the address bar



**1. Define SPA (Single-Page Application) and its Benefits**

* **Definition:**  
  A SPA is a web application that loads a single HTML page and dynamically updates content without reloading the whole page.
* **Benefits:**
  + Faster navigation (no full page reload)
  + Smooth, app-like user experience
  + Reduced server load
  + Efficient data loading using APIs

**2. Define React and Identify its Working**

* **React:**  
  An open-source JavaScript library for building user interfaces, mainly for SPAs.
* **Working:**
  + Uses a **component-based architecture**
  + Employs a **Virtual DOM** to efficiently update UI
  + Data flows in **one direction** (unidirectional binding)

**3. Differences Between SPA and MPA**

| **Feature** | **SPA** | **MPA** |
| --- | --- | --- |
| Page reload | No full reload | Reloads entire page |
| Speed | Faster after first load | Slower due to multiple requests |
| SEO | Harder to optimize | Easier for SEO |
| Development focus | Client-side rendering | Server-side rendering |

**4. Pros & Cons of SPA**

**Pros:**

* Fast and responsive
* Offline support possible
* Rich, interactive UI

**Cons:**

* SEO challenges
* Larger initial load time
* Browser history handling can be tricky

**5. React**

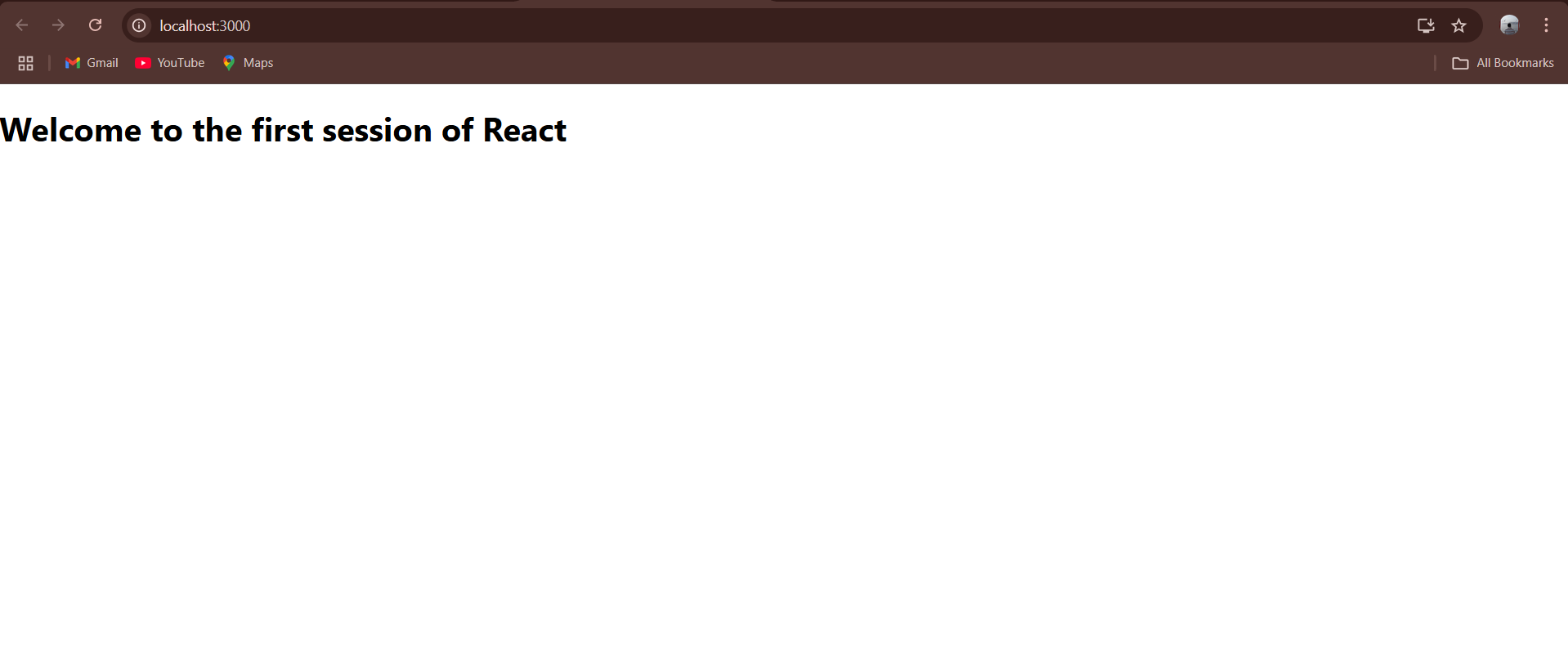
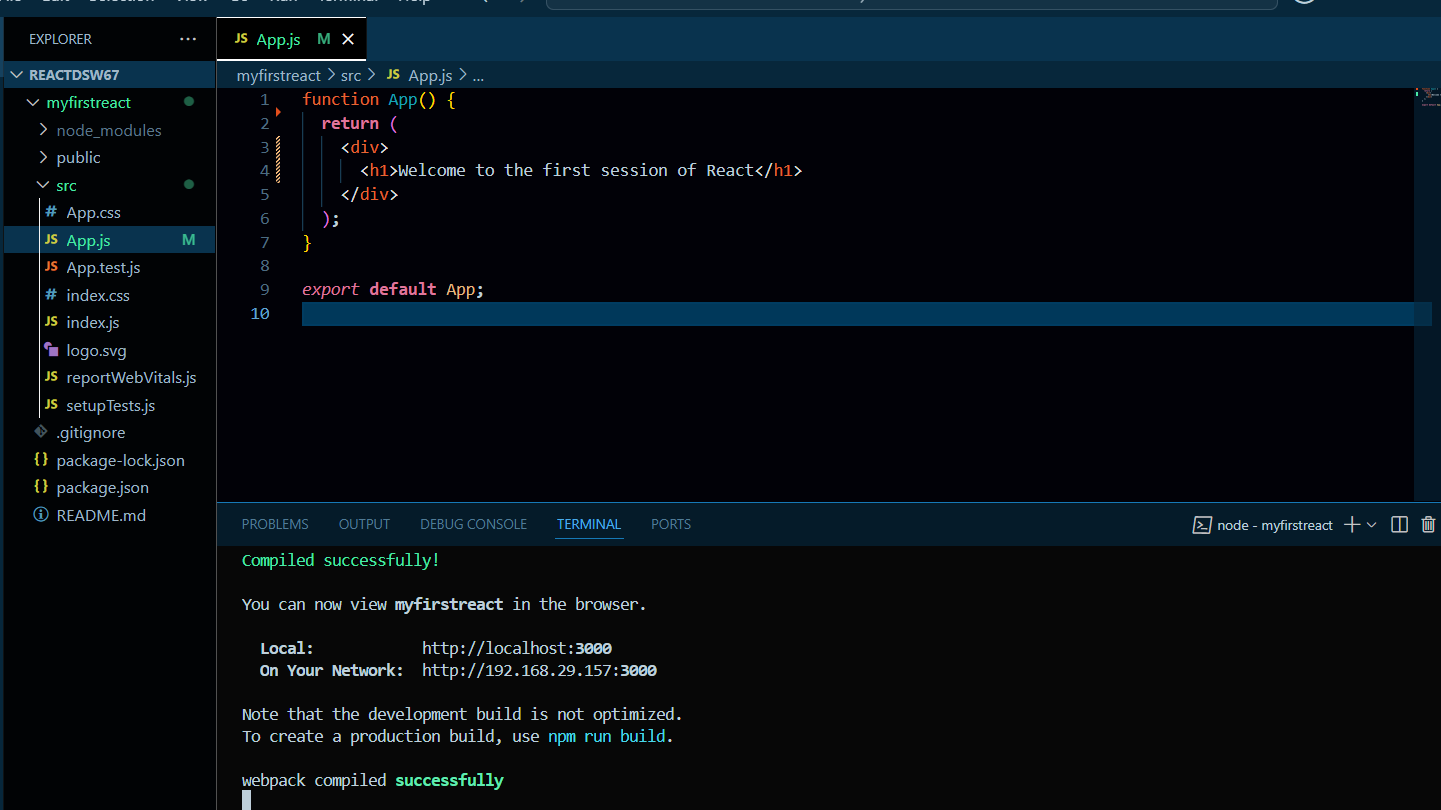
React is a **JavaScript library** created by Facebook for building reusable UI components and efficiently updating the DOM.

**6. Virtual DOM**

* **Definition:**  
  A lightweight copy of the actual DOM maintained in memory.
* **Purpose:**  
  Improves performance by updating only the changed parts of the DOM instead of re-rendering everything.

**7. Features of React**

* Declarative UI
* Component-based
* Virtual DOM
* JSX (JavaScript + XML)
* Unidirectional data flow
* Strong community support

Outputs and Codes:-  
  


## **Objectives**

* Explain React components
* Identify the differences between components and JavaScript functions
* Identify the types of components
* Explain class component
* Explain function component
* Define component constructor
* Define render() function

In this hands-on lab, you will learn how to:

* Create a class component
* Create multiple components
* Render a component

## **Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

## **Notes**

Estimated time to complete this lab: **30 minutes.**

Create a react app for Student Management Portal named StudentApp and create a component named Home which will display the Message “Welcome to the Home page of Student Management Portal”. Create another component named About and display the Message “Welcome to the About page of the Student Management Portal”. Create a third component named Contact and display the Message “Welcome to the Contact page of the Student Management Portal”. Call all the three components.

1. Create a React project named “StudentApp” type the following command in terminal of Visual studio:



1. Create a new folder under Src folder with the name “Components”. Add a new file named “Home.js”
2. Type the following code in Home.js



1. Under Src folder add another file named “About.js”
2. Repeat the same steps for Creating “About” and “Contact” component by adding a new file as ”About.js”, “Contact.js” under “Src” folder and edit the code as mentioned for “Home” Component.
3. Edit the App.js to invoke the Home, About and Contact component as follows:



1. In command Prompt, navigate into StudentApp and execute the code by typing the following command:



1. Open browser and type “localhost:3000” in the address bar:



**Part 1 – Theory**

**1. React Components**

* **Definition:**  
  Components are the building blocks of a React application.  
  Each component is a **reusable piece of UI** that can contain HTML, CSS, and JavaScript logic.

**2. Difference Between Components and JavaScript Functions**

| **React Component** | **JavaScript Function** |
| --- | --- |
| Returns JSX (UI) | Returns primitive values or objects |
| Used to build UI in React | Used for general programming tasks |
| Can have lifecycle methods (in class components) | No lifecycle methods |
| Can manage state | No built-in state management |

**3. Types of Components**

1. **Class Components**
   * ES6 classes that extend React.Component
   * Can have **state** and **lifecycle methods**
2. **Function Components**
   * JavaScript functions that return JSX
   * Can use **React Hooks** to manage state and lifecycle

**4. Class Component**

* A component defined as a class that:
  + Extends React.Component
  + Has a **render()** method to return JSX
  + Can use **constructor** to initialize state and bind methods

**5. Function Component**

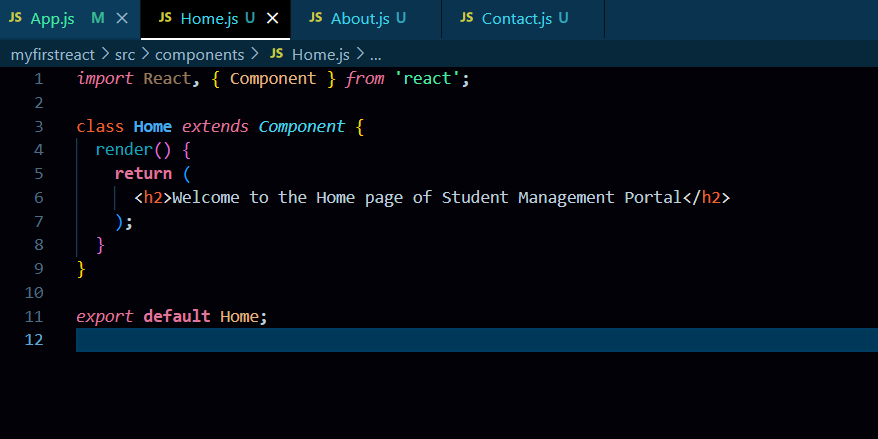
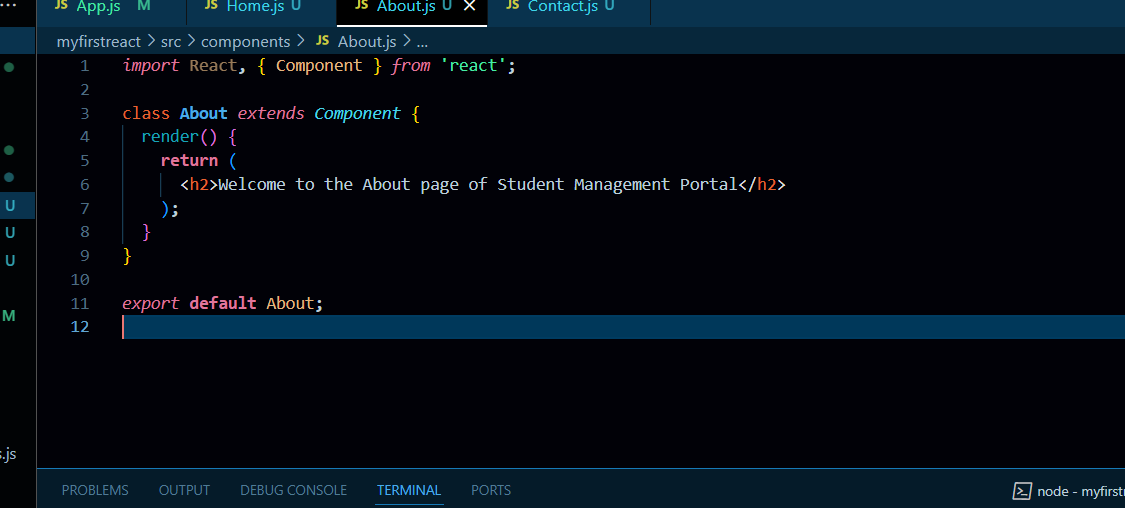
* A simple JavaScript function returning JSX
* Can use Hooks (useState, useEffect) for state and lifecycle

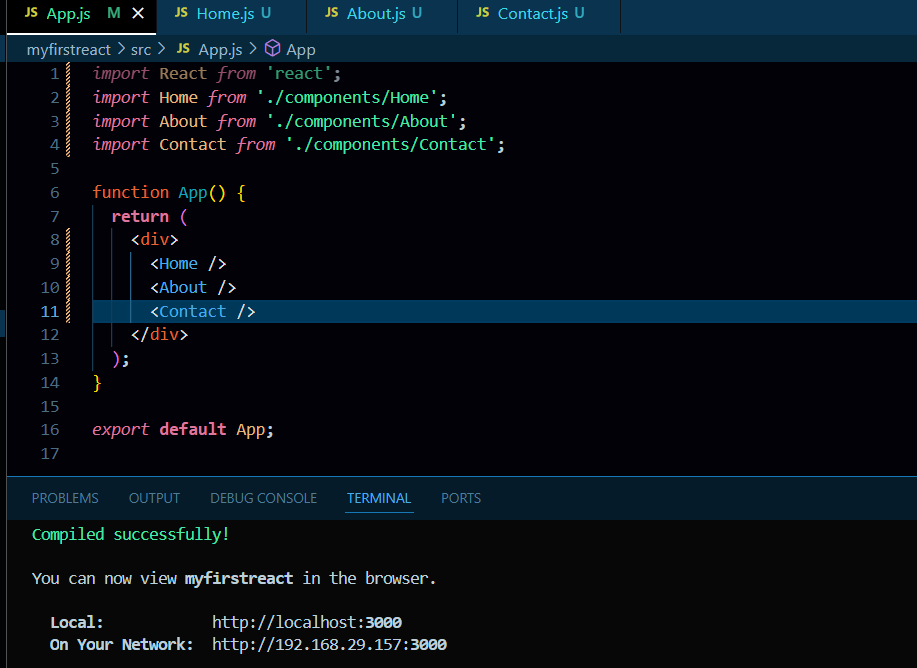
**6. Component Constructor**

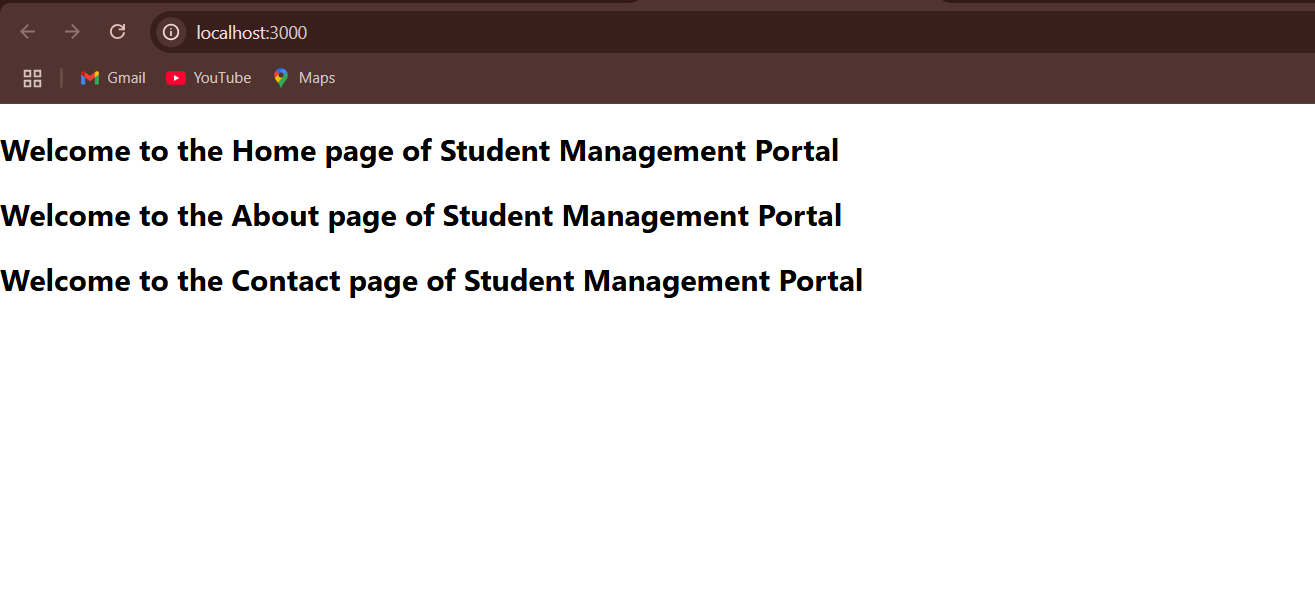
* A special method in **class components**
* Used to initialize **state** and bind **event handlers**
* Example:

**7. render() Function**

* Required in **class components**
* Must return JSX
* Gets called whenever the state or props change

Codes and Output:  
  
  
  






**Objectives**

* Explain React components
* Identify the differences between components and JavaScript functions
* Identify the types of components
* Explain class component
* Explain function component
* Define component constructor
* Define render() function

In this hands-on lab, you will learn how to:

* Create a function component
* Apply style to components
* Render a component

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **30 minutes.**

Create a react app for Student Management Portal named scorecalculatorapp and create a function component named “CalculateScore” which will accept Name, School, Total and goal in order to calculate the average score of a student and display the same.

1. Create a React project named “scorecalculatorapp” type the following command in terminal of Visual studio:



1. Create a new folder under Src folder with the name “Components”. Add a new file named “CalculateScore.js”
2. Type the following code in CalculateScore.js





1. Create a Folder named Stylesheets and add a file named “mystyle.css” in order to add some styles to the components:



1. Edit the App.js to invoke the CalculateScore functional component as follows:



1. In command Prompt, navigate into scorecalculatorapp and execute the code by typing the following command:



1. Open browser and type “localhost:3000” in the address bar:



**Part 1 – Theory**

**1. React Components**

* **Definition:**  
  Components are reusable, independent building blocks in React that describe a part of the UI.
* **Purpose:**  
  Break the UI into smaller, maintainable pieces.

**2. Difference Between Components and JavaScript Functions**

| **React Component** | **JavaScript Function** |
| --- | --- |
| Returns **JSX** (UI elements) | Returns values, objects, or performs tasks |
| Used to build UI in React apps | Used for general programming logic |
| Can manage **state** and **props** | No built-in state or props concept |
| May have lifecycle methods (class) | No lifecycle methods |

**3. Types of Components**

1. **Class Components**
   * Defined using ES6 class syntax
   * Can have state and lifecycle methods
   * Must have a render() method
2. **Function Components**
   * Simple JavaScript functions that return JSX
   * Can use Hooks (useState, useEffect) for state/lifecycle

**4. Class Component**

* Extends React.Component
* Contains a render() method
* Example:

class MyClass extends React.Component {

render() {

return <h1>Hello</h1>;

}

}

**5. Function Component**

* Returns JSX directly from a function
* Example:

function MyFunc() {

return <h1>Hello</h1>;

}

**6. Component Constructor**

* Used **only in class components**
* Initializes state or binds methods
* Example:

constructor(props) {

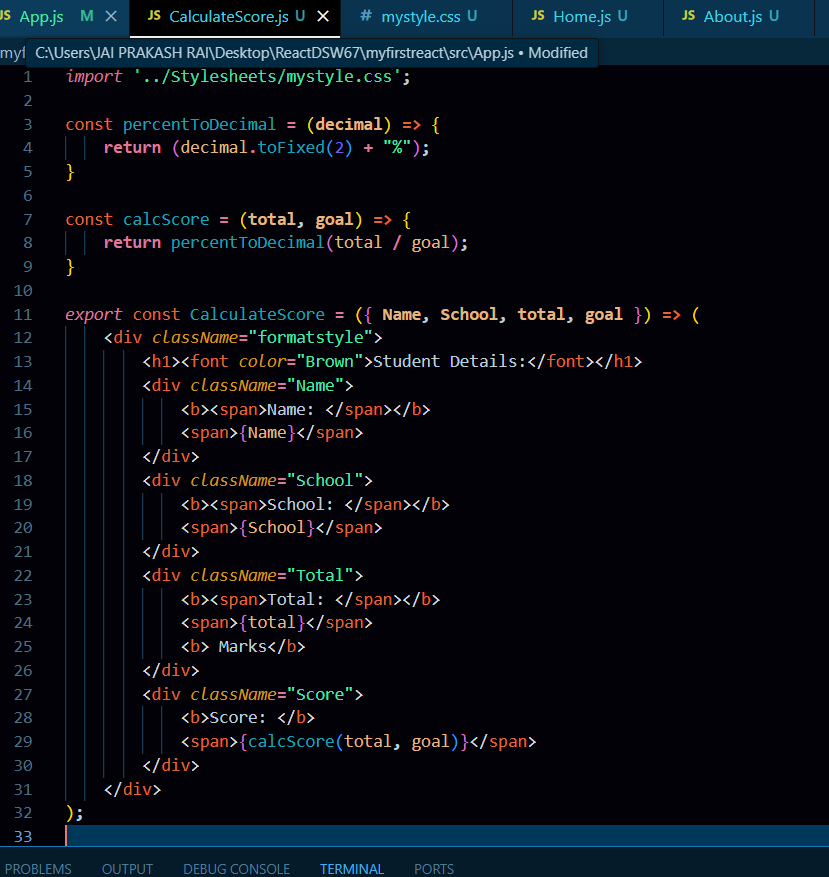
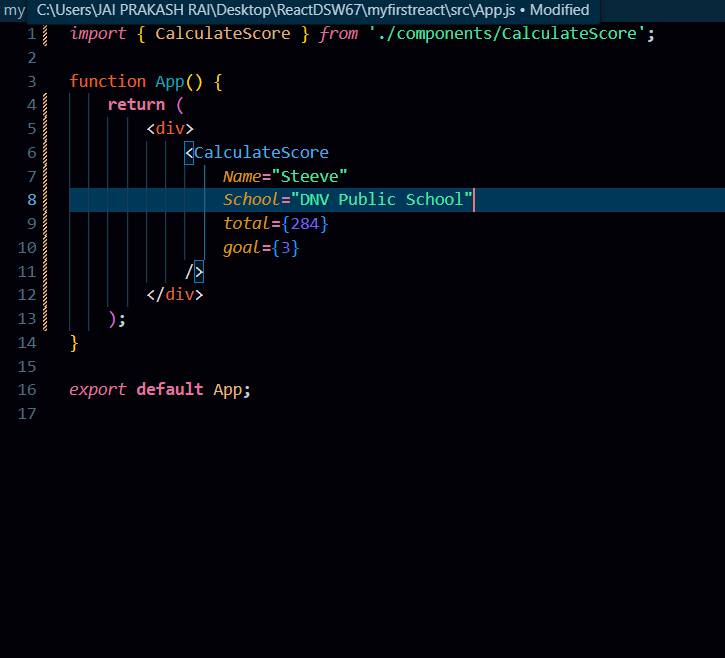
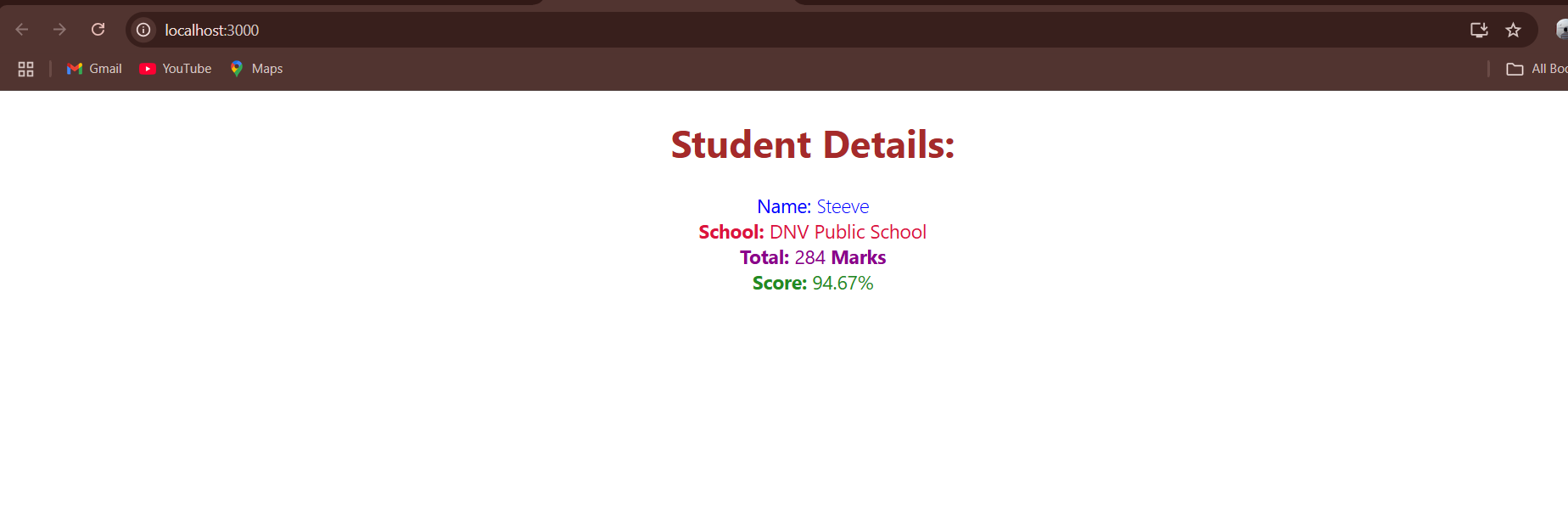
super(props);

this.state = {};

}

**7. render() Function**

* Only in **class components**
* Returns JSX to display UI
* Runs whenever state/props change

Codes and Outputs:-  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**Objectives**

* Explain the need and Benefits of component life cycle
* Identify various life cycle hook methods
* List the sequence of steps in rendering a component

In this hands-on lab, you will learn how to:

* Implement componentDidMount() hook
* Implementing componentDidCatch() life cycle hook.

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **60 minutes.**

1. Create a new react application using *create-react-app* tool with the name as “blogapp”
2. Open the application using VS Code
3. Create a new file named as **Post.js** in **src folder** with following properties



*Figure 2: Post class*

1. Create a new class based component named as **Posts** inside **Posts.js** file



*Figure 3: Posts Component*

1. Initialize the component with a list of Post in state of the component using the constructor
2. Create a new method in component with the name as **loadPosts()** which will be responsible for using Fetch API and assign it to the component state created earlier. To get the posts use the url (<https://jsonplaceholder.typicode.com/posts>)



*Figure 4: loadPosts() method*

1. Implement the **componentDidMount()** hook to make calls to **loadPosts()** which will fetch the posts



*Figure 5: componentDidMount() hook*

1. Implement the **render()** which will display the title and post of posts in html page using heading and paragraphs respectively.



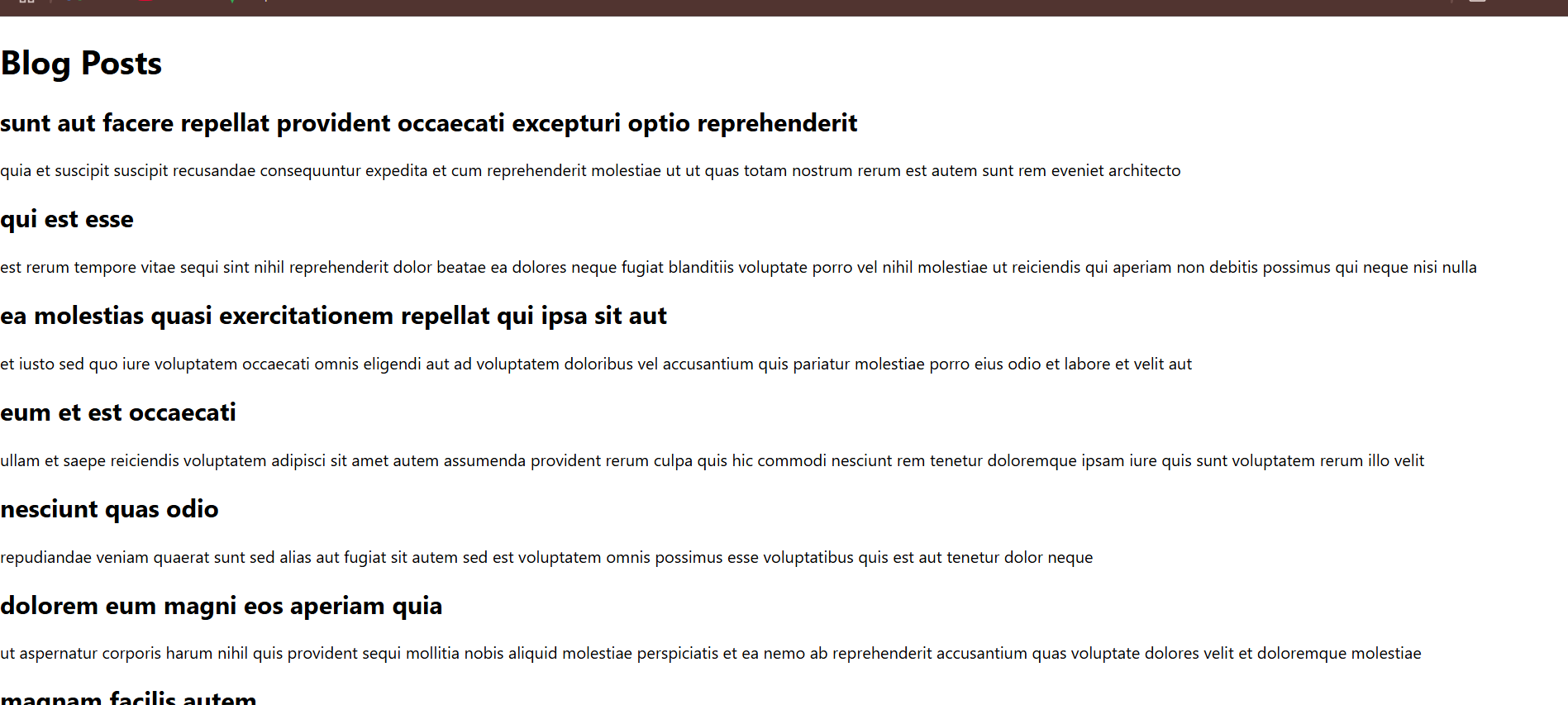
*Figure 6: render() method*

1. Define a **componentDidCatch()** method which will be responsible for displaying any error happing in the component as alert messages.



*Figure 7: componentDidCatch() hook*

1. Add the Posts component to App component.
2. Build and Run the application using *npm start* command.

Output:-  
  
  


**Objectives**

* Understanding the need for styling react component
* Working with CSS Module and inline styles

In this hands-on lab, you will learn how to:

* Style a react component
* Define styles using the CSS Module
* Apply styles to components using className and style properties

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **30 minutes.**

My Academy team at Cognizant want to create a dashboard containing the details of ongoing and completed cohorts. A react application is created which displays the detail of the cohorts using react component. You are assigned the task of styling these react components.

Download and build the attached react application.



1. Unzip the react application in a folder
2. Open command prompt and switch to the react application folder
3. Restore the node packages using the following commands



*Figure 1: Restore packages*

1. Open the application using VS Code
2. Create a new CSS Module in a file called “CohortDetails.module.css”
3. Define a css class with the name as “box” with following properties

*Width = 300px;*

*Display = inline block;*

*Overall 10px margin*

*Top and bottom padding as 10px*

*Left and right padding as 20px*

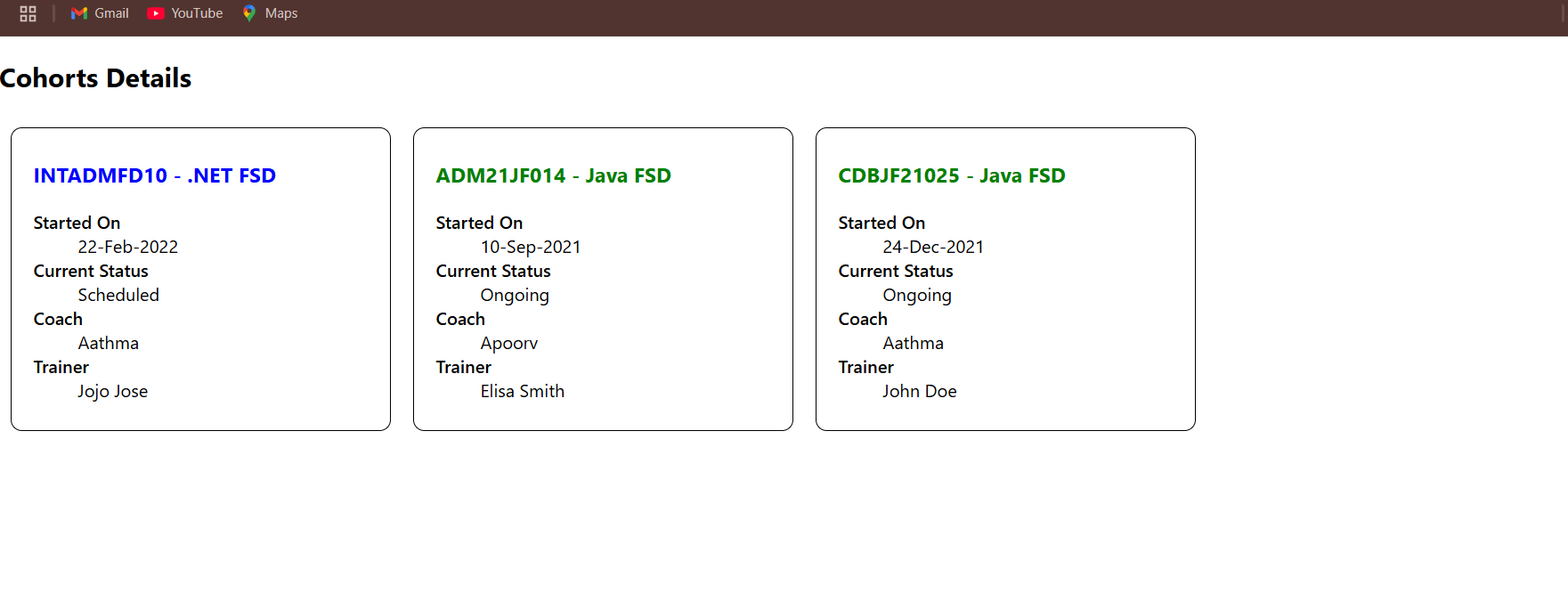
*1 px border in black color*

*A border radius of 10px*

1. Define a css style for html <dt> element using tag selector. Set the font weight to 500.
2. Open the cohort details component and import the CSS Module
3. Apply the box class to the container div
4. Define the style for <h3> element to use “green” color font when cohort status is “ongoing” and “blue” color in all other scenarios.
5. Final result should look similar to the below image



*Figure 2: Final Result*

  
  
**Thank YOU<3**