**WEEK – 6 : HANDS-ON EXERCISE**

**React**

**Exercise 1: React SPA(Single Page Application) Setup**

**Scenario:**

Create a simple app called myfirstreact that shows the message “Welcome to the first session of React” on the screen.

**Step 1: Install Node.js and npm**

Go to the official [Node.js download page](https://nodejs.org/en/download/) and install the LTS version. npm will be installed automatically along with it.

**Step 2: Install Create React App CLI Tool**

npm install -g create-react-app

**Step 3: Create the React Application**

npx create-react-app myfirstreact

**Step 4: Navigate into the Application Directory**

cd myfirstreact

**Step 5: Open Project in Visual Studio Code**

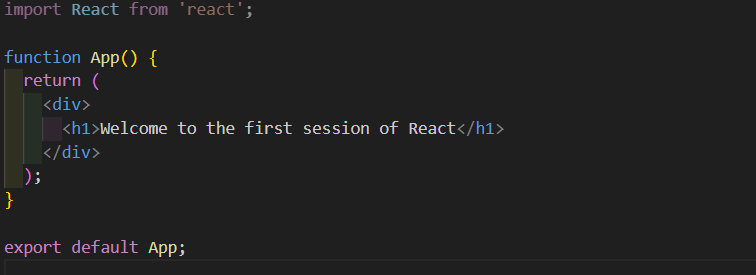
code .

**Step 6: Modify App.js**

* Navigate to the src folder
* Open App.js

**CODE:**

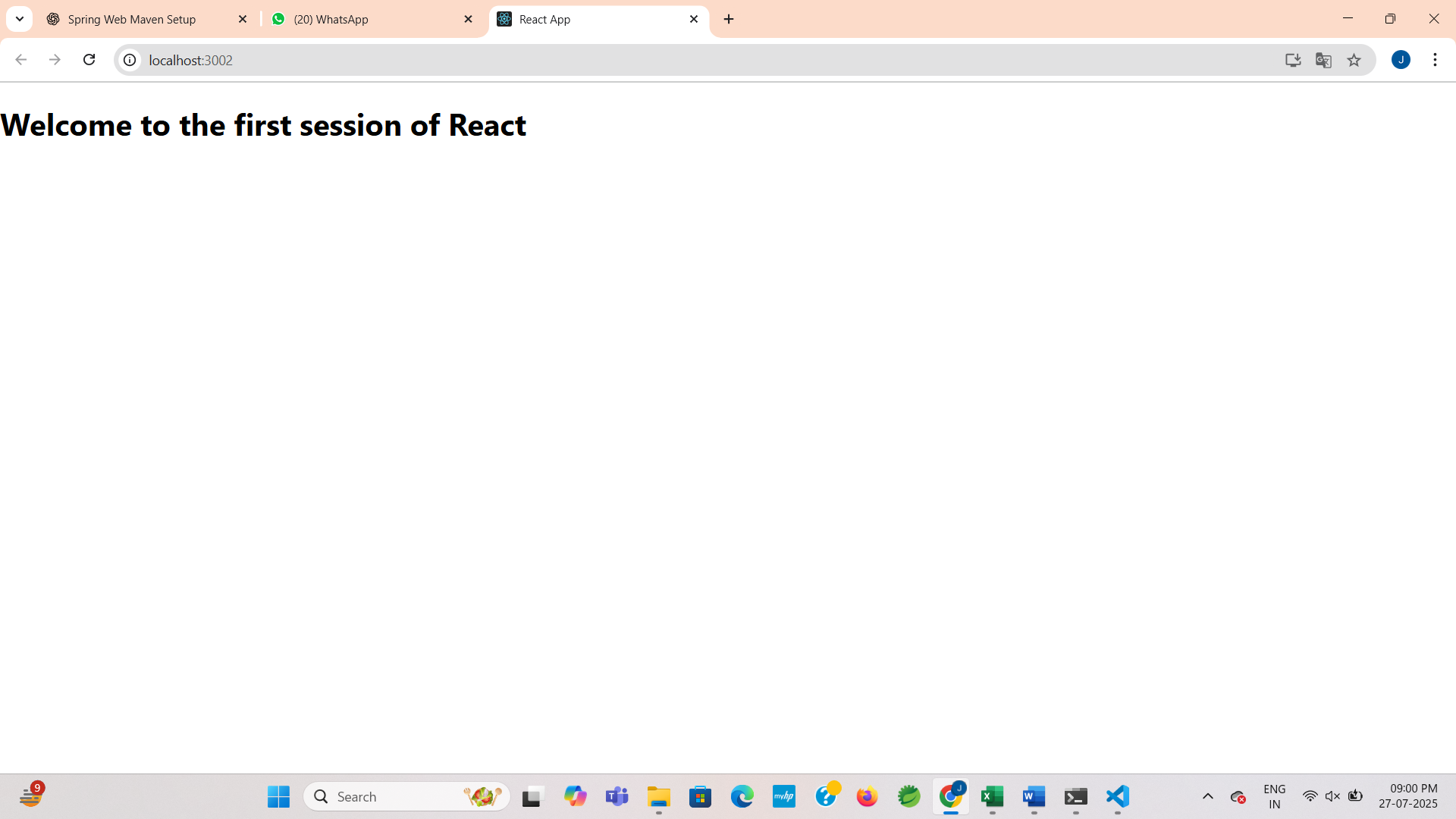
**App.js:**



**Step 7: Run the React App**

npm start

**OUTPUT:**



**Exercise 2: Building a Basic React Student Management Portal**

**Scenario:**

Create a simple Student Management Portal using React with three components: Home, About, and Contact, each displaying a welcome message. Learn how to structure, create, and render components in a React project.

**Step 1: Create a React App**

npx create-react-app StudentApp

cd StudentApp

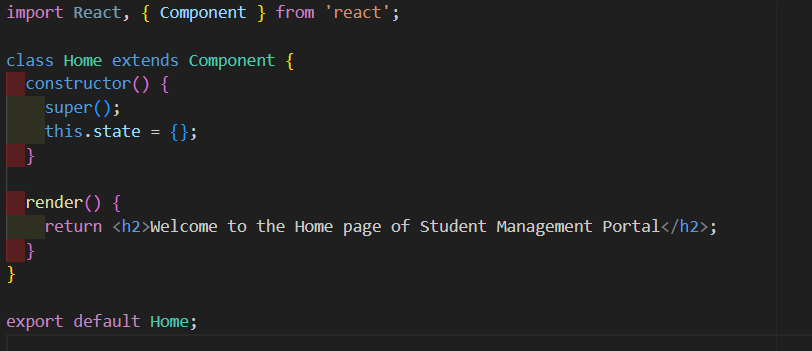
**Step 2 : Create Components Folder**

Inside the src directory:

* Create a folder named Components.

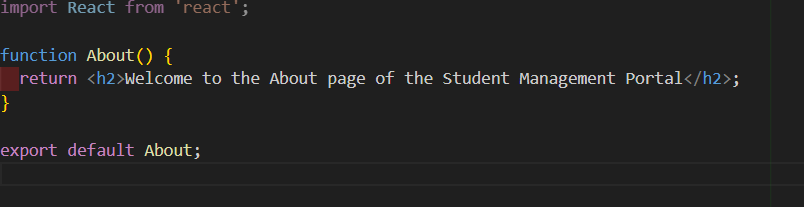
**Step 3: Create Home.js**

**CODE:**

****

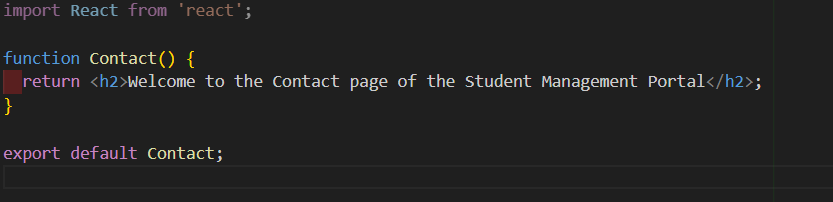
**Step 4: Create About.js**

**CODE:**

****

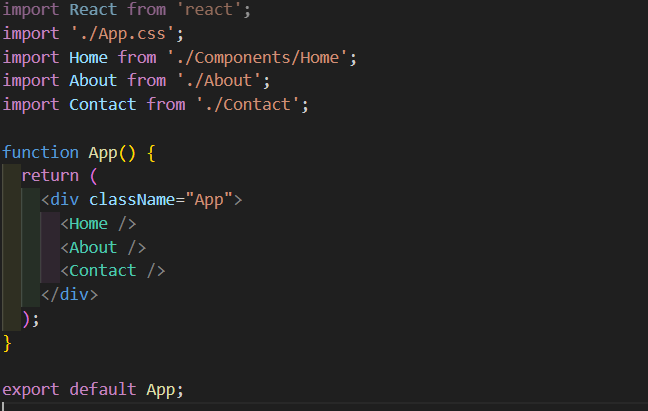
**Step 5: Create Contact.js**

**CODE:**

****

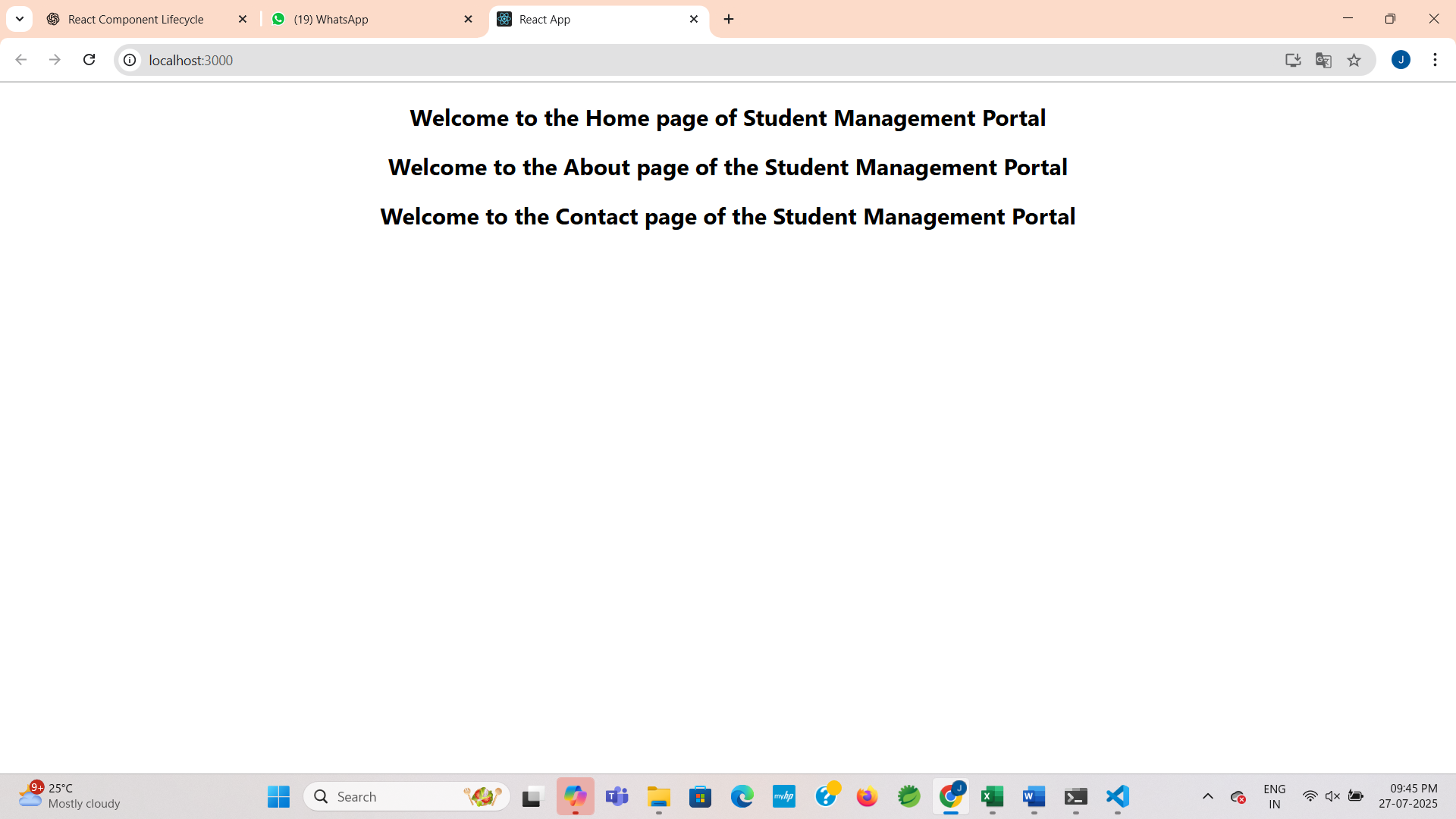
**Step 6:** **Edit App.js to Render All Components.**

**CODE:**

****

**Step 7: Run the Application**

**OUTPUT:**



**Exercise 3 : Student Score Calculator Using React Functional Components**

**Scenario:**

Create a React app to display a student’s average score using a functional component called CalculateScore. It takes Name, School, Total, and Goal as input props and shows the result in a styled format.

**Step 1 : Create React App**

npx create-react-app scorecalculatorapp

* After it finishes, navigate into the app:

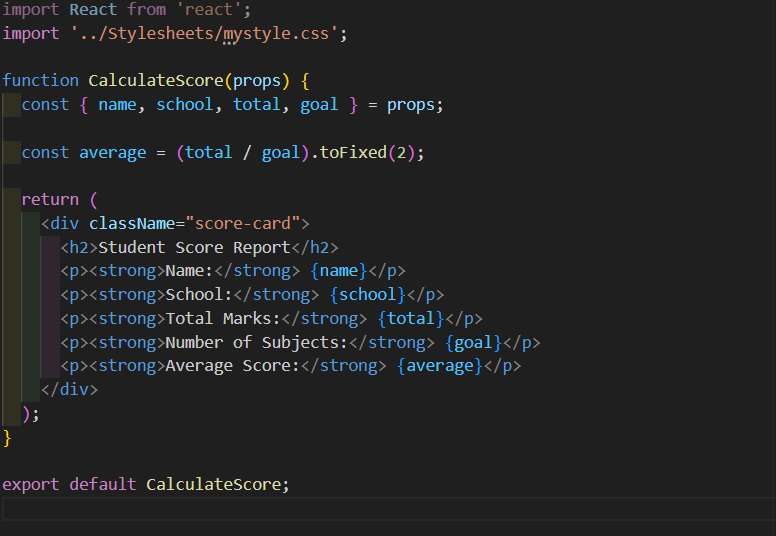
cd scorecalculatorapp

**Step 2: Create Components Folder and File**

Inside the src directory, create a folder named Components.

**Step 3: Create CalculateScore.js**

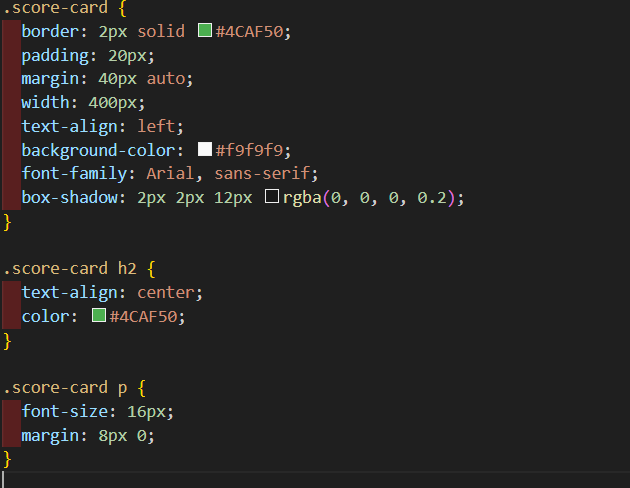
**CODE:**

****

**Step 4: Add Styles - mystyle.css**

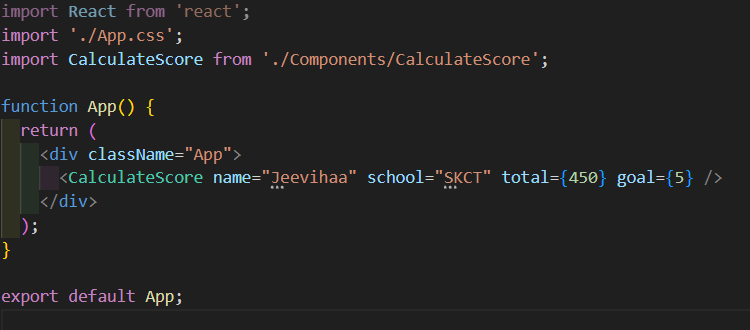
Create a folder in src named Stylesheets and inside that, create mystyle.css.

**CODE:**

****

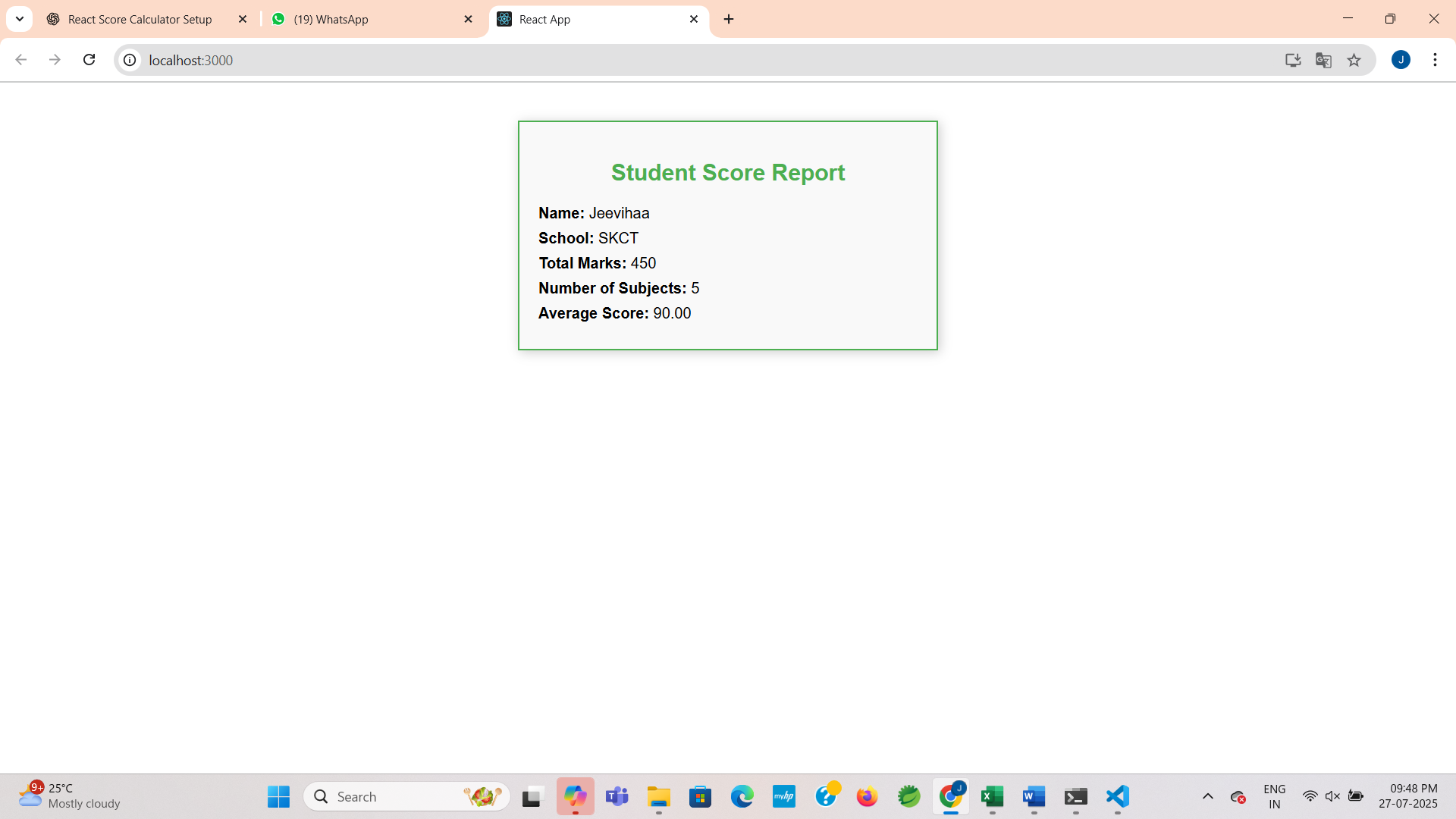
**Step 5 : Modify App.js to Render the Component**

**CODE:**

****

**Step 6: Run the application**

**OUTPUT:**



**Exercise 4: Understanding and Implementing React Component Lifecycle Methods**

**Scenario:**

You are building a simple blog app that fetches and displays posts from an external API. To manage data fetching and error handling, you will implement React class components using lifecycle methods like componentDidMount() and componentDidCatch().

**Step 1: Create React App**

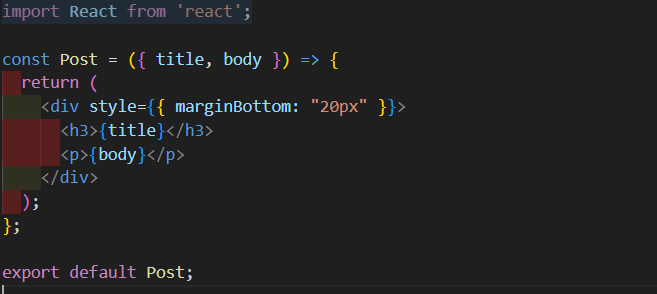
npx create-react-app blogapp

cd blogapp

code .

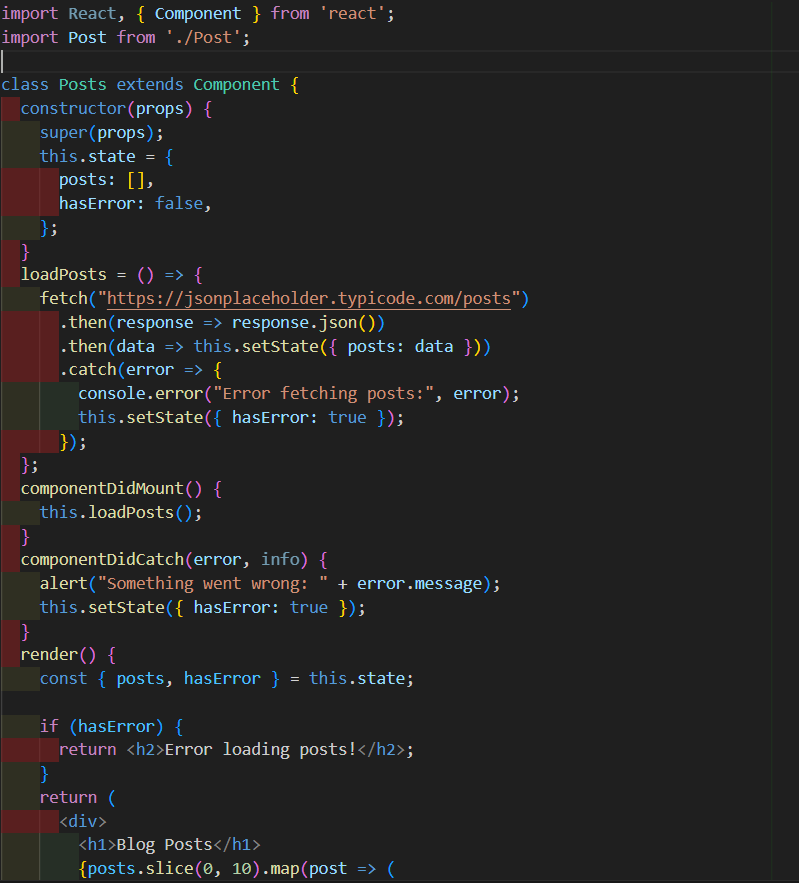
**Step 2: Create Post.js**

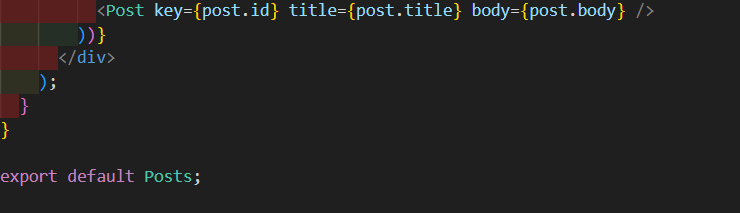
**CODE:**

****

**Step 3: Create Class Component Posts.js**

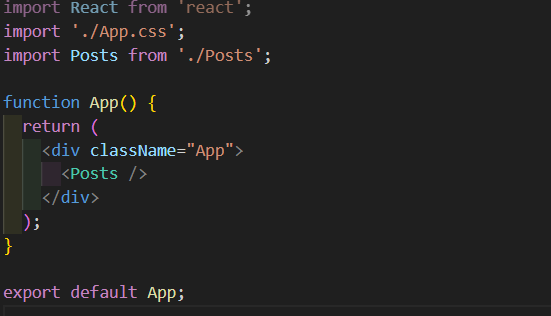
**CODE:**





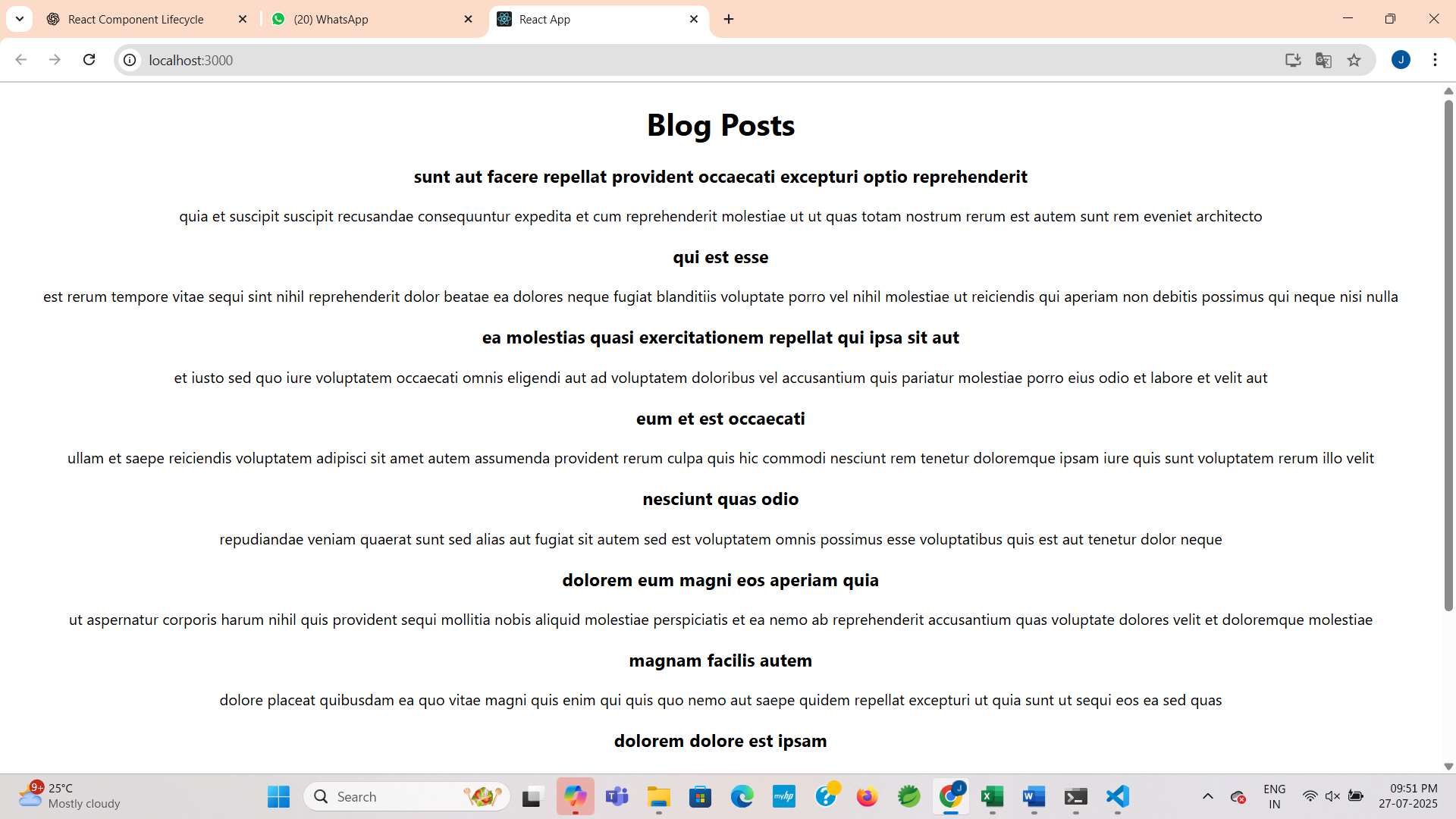
**Step 4: Use Posts in App.js**

**CODE:**

****

**Step 5: Run the Application**

**OUTPUT:**



**Exercise 5 : Styling React Components with CSS Modules and Inline Styles**

**Scenario:**

Your are building a dashboard to display details of cohorts. The React app is ready, and you're assigned to style the CohortDetails component using CSS Modules and inline styles.

**Step 1 : Open the Project**

Unzip the given React app.

* Open terminal or command prompt:

cd cohorttracker

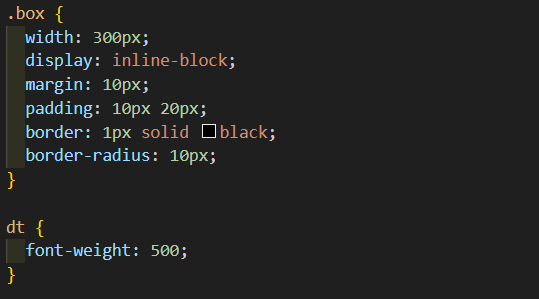
npm install

* Open the project in VS Code:

code .

**Step 2 : Create the CSS Module**

**CODE:**

****

**Step 3 : Create** **CohortDetails.js**

**CODE:**

****

**Step 4 : Create App.js**

**CODE:**

****

**Step 5 : Run the application**

**OUTPUT:**

