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 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

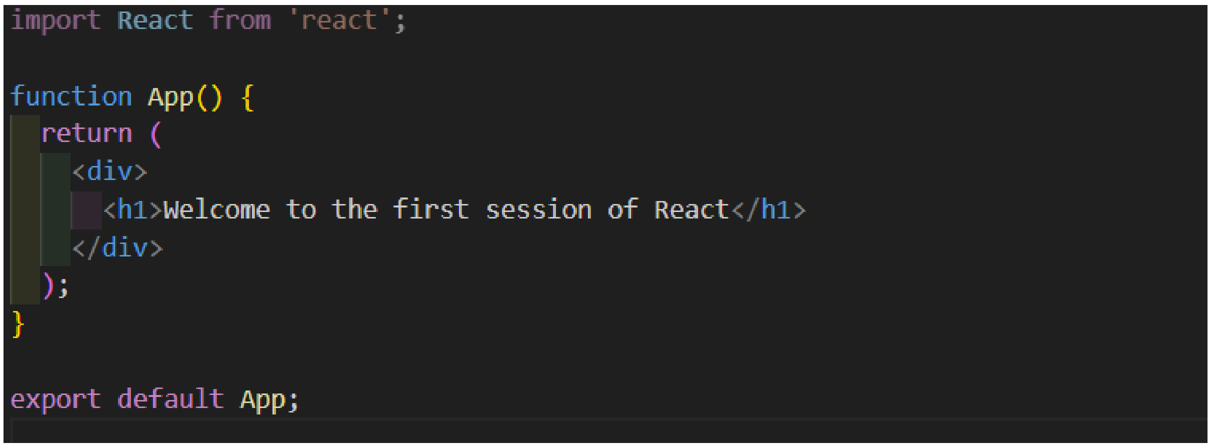
```
code .
```

Step 6: Modify App.js

- Navigate to the src folder
- Open App.js

CODE:

App.js:

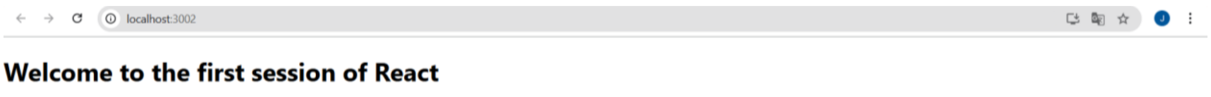


```
import React from 'react';

function App() {
  return (
    <div>
      <h1>Welcome to the first session of React</h1>
    </div>
  );
}

export default App;
```

Step 7: Run the React App

npm start

OUTPUT:



localhost:3002

Welcome to the first session of React

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
```

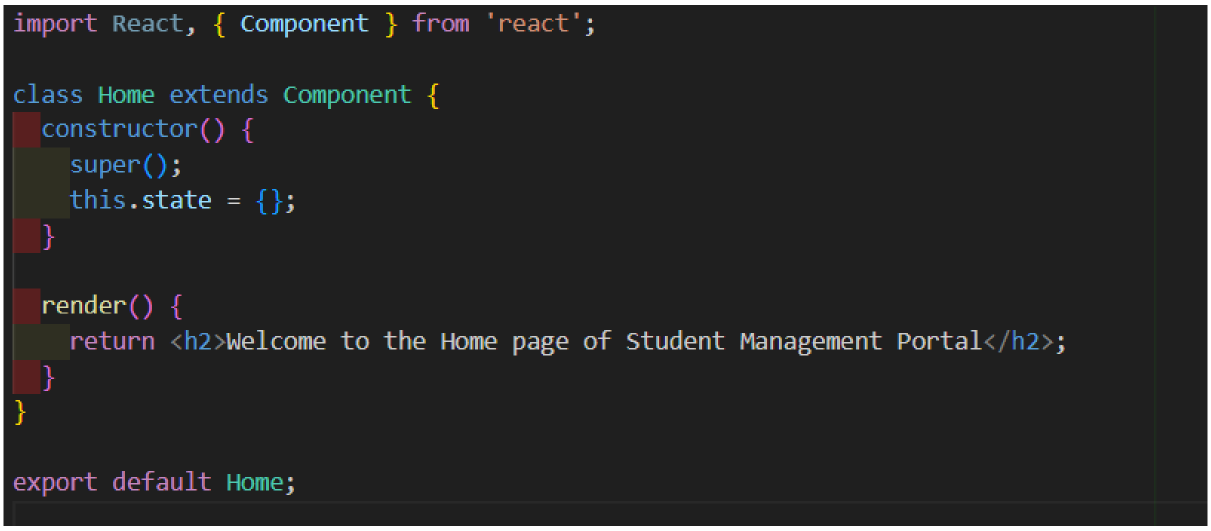
Step 2 : Create Components Folder

Inside the src directory:

- Create a folder named Components.

Step 3: Create Home.js

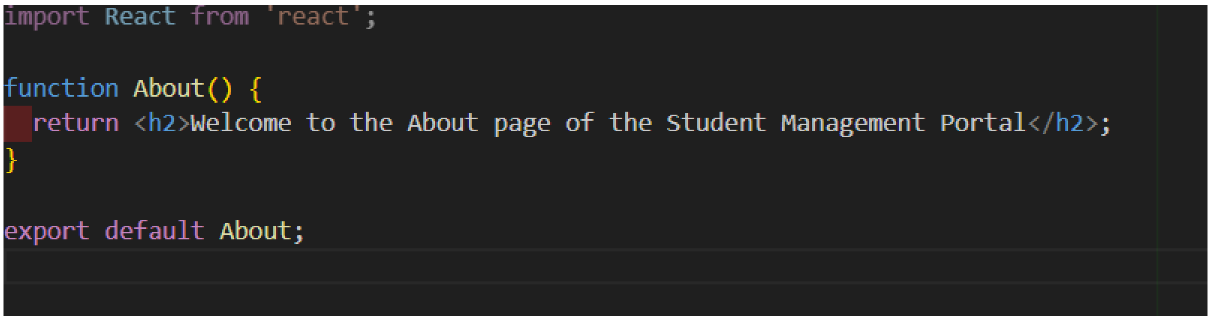
CODE:



```
import React, { Component } from 'react';

class Home extends Component {
  constructor() {
    super();
    this.state = {};
  }

  render() {
    return <h2>Welcome to the Home page of Student Management Portal</h2>;
  }
}

export default Home;
```

Step 4: Create About.js

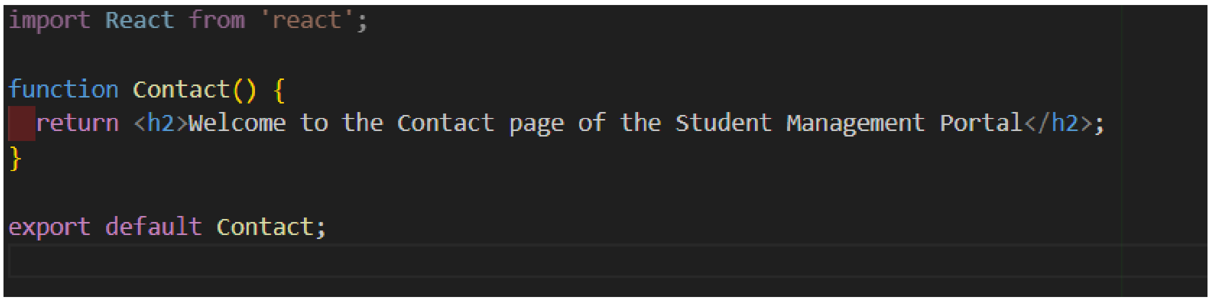
CODE:



```
import React from 'react';

function About() {
  return <h2>Welcome to the About page of the Student Management Portal</h2>;
}

export default About;
```

Step 5: Create Contact.js

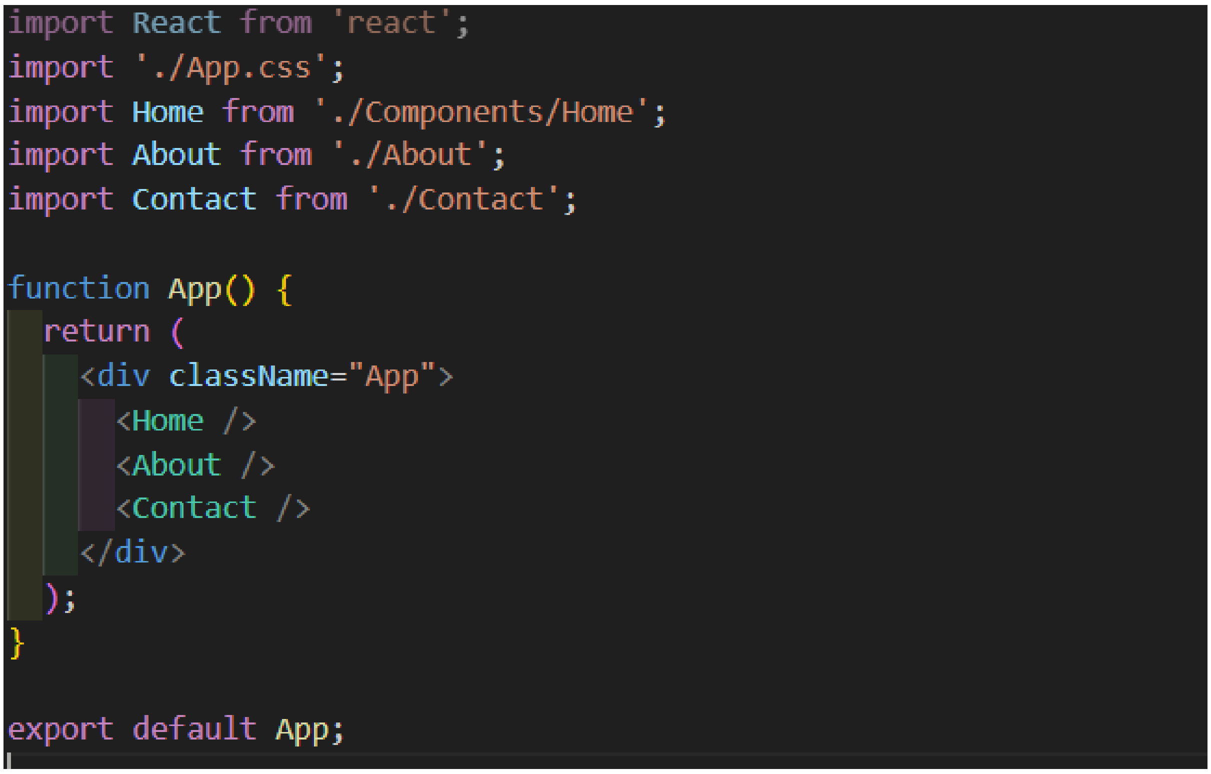
CODE:



```
import React from 'react';

function Contact() {
  return <h2>Welcome to the Contact page of the Student Management Portal</h2>;
}

export default Contact;
```

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

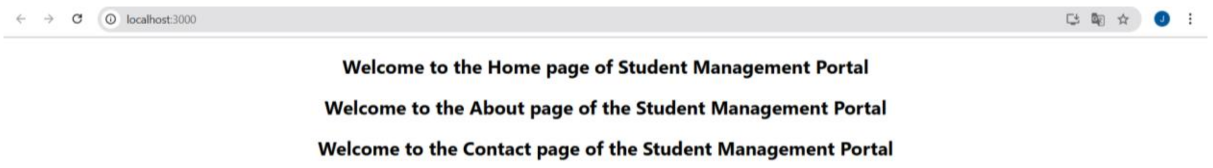
CODE:



```
import React from 'react';
import './App.css';
import Home from './Components/Home';
import About from './About';
import Contact from './Contact';

function App() {
  return (
    <div className="App">
      <Home />
      <About />
      <Contact />
    </div>
  );
}

export default App;
```

Step 7: Run the Application

OUTPUT:



```
Welcome to the Home page of Student Management Portal
Welcome to the About page of the Student Management Portal
Welcome to the Contact page of the Student Management Portal
```

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:

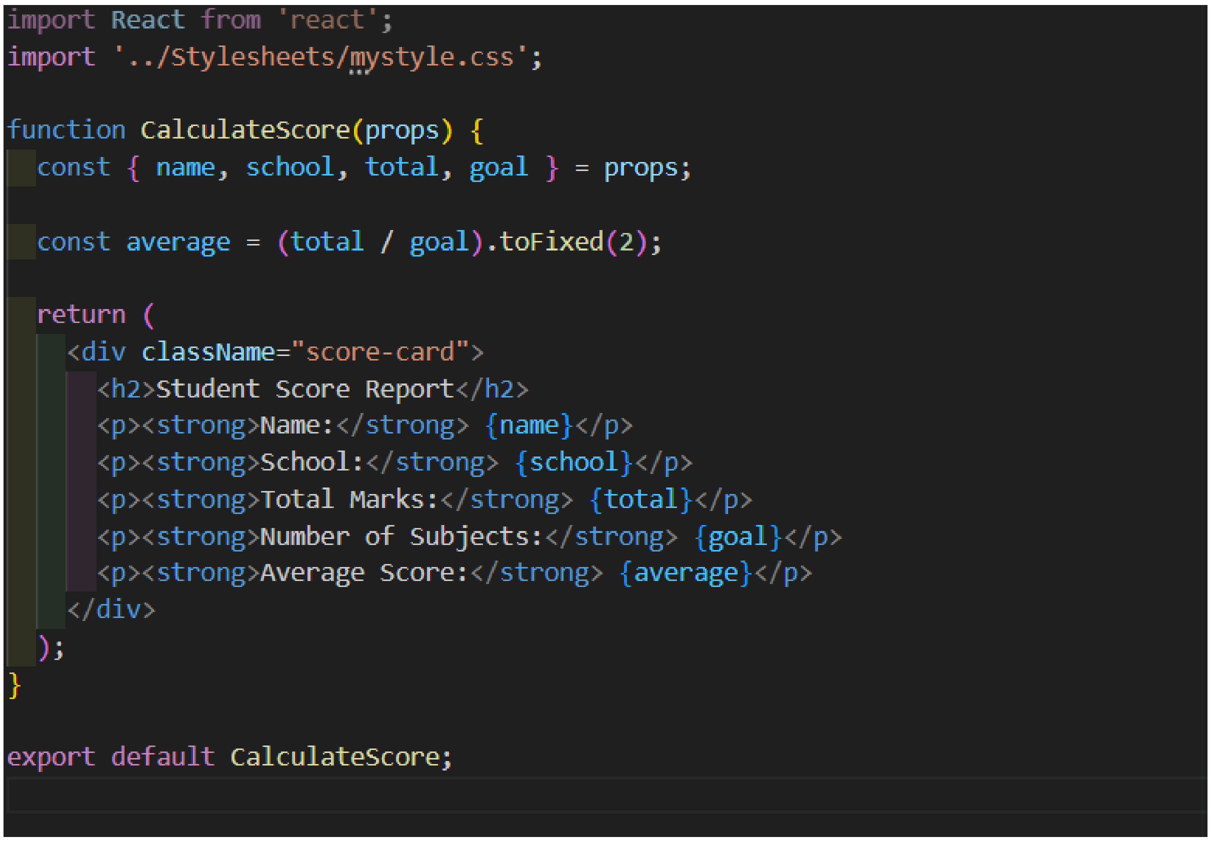
  cd scorecalculatorapp

Step 2: Create Components Folder and File

Inside the src directory, create a folder named Components.

Step 3: Create CalculateScore.js

CODE:

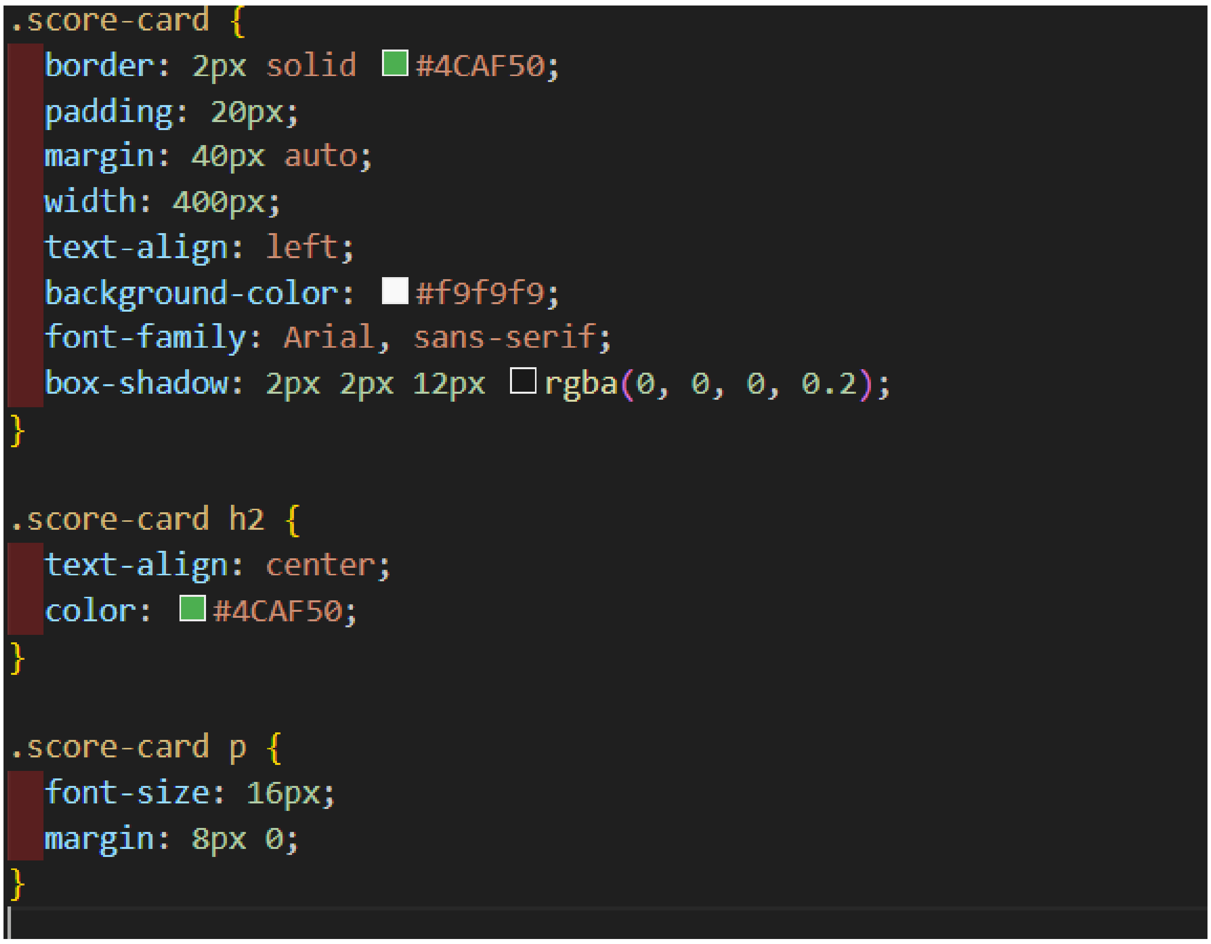


```
import React from 'react';
import '../Stylesheets/mystyle.css';

function CalculateScore(props) {
  const { name, school, total, goal } = props;

  const average = (total / goal).toFixed(2);

  return (
    <div className="score-card">
      <h2>Student Score Report</h2>
      <p><strong>Name:</strong> {name}</p>
      <p><strong>School:</strong> {school}</p>
      <p><strong>Total Marks:</strong> {total}</p>
      <p><strong>Number of Subjects:</strong> {goal}</p>
      <p><strong>Average Score:</strong> {average}</p>
    </div>
  );
}

export default CalculateScore;
```

Step 4: Add Styles - mystyle.css

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

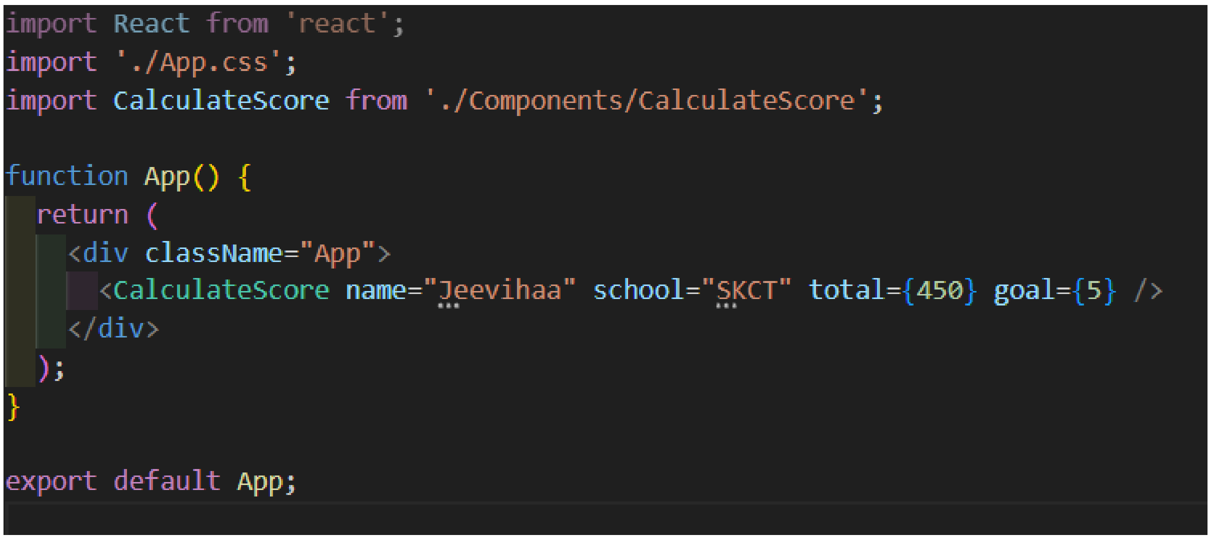
CODE:



```css
.score-card {
    border: 2px solid #4CAF50;
    padding: 20px;
    margin: 40px auto;
    width: 400px;
    text-align: left;
    background-color: #f9f9f9;
    font-family: Arial, sans-serif;
    box-shadow: 2px 2px 12px rgba(0, 0, 0, 0.2);
}

.score-card h2 {
    text-align: center;
    color: #4CAF50;
}

.score-card p {
    font-size: 16px;
    margin: 8px 0;
}
```

Step 5 : Modify App.js to Render the Component

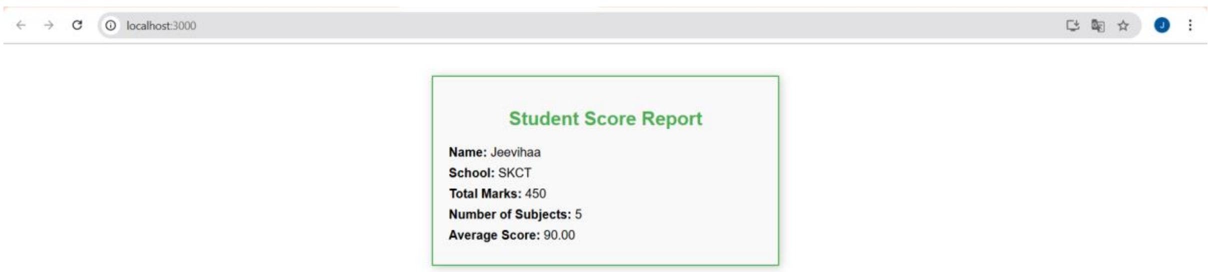
CODE:



```javascript
import React from 'react';
import './App.css';
import CalculateScore from './Components/CalculateScore';

function App() {
  return (
    <div className="App">
      <CalculateScore name="Jeevihaa" school="SKCT" total={450} goal={5} />
    </div>
  );
}

export default App;
```

Step 6: Run the application

OUTPUT:



localhost:3000

Student Score Report

Name: Jeevihaa
School: SKCT
Total Marks: 450
Number of Subjects: 5
Average Score: 90.00

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

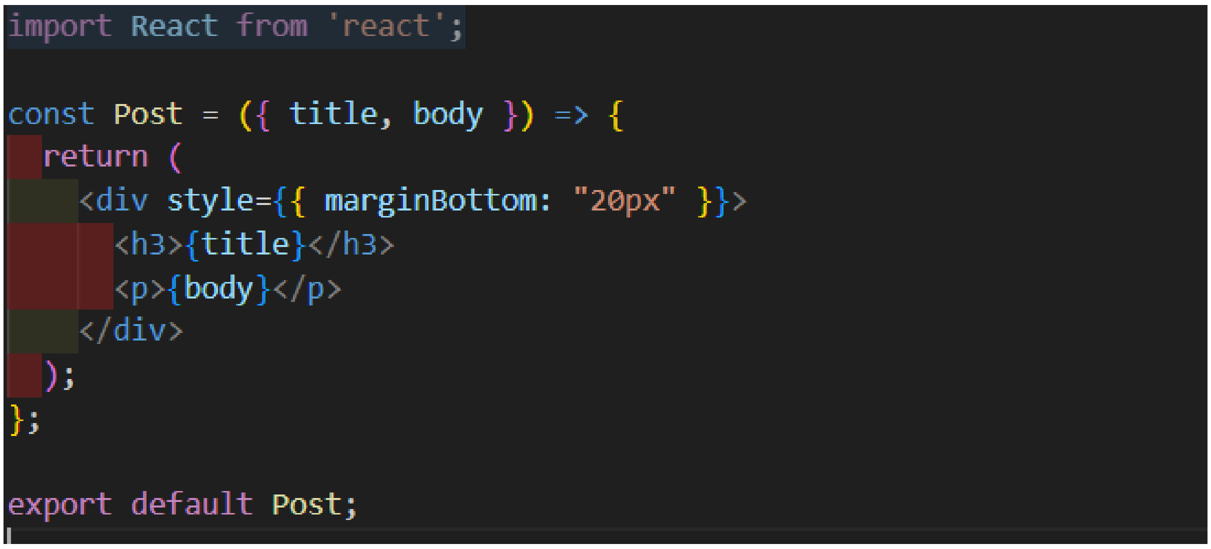
npx create-react-app blogapp

cd blogapp

code .

Step 2: Create Post.js

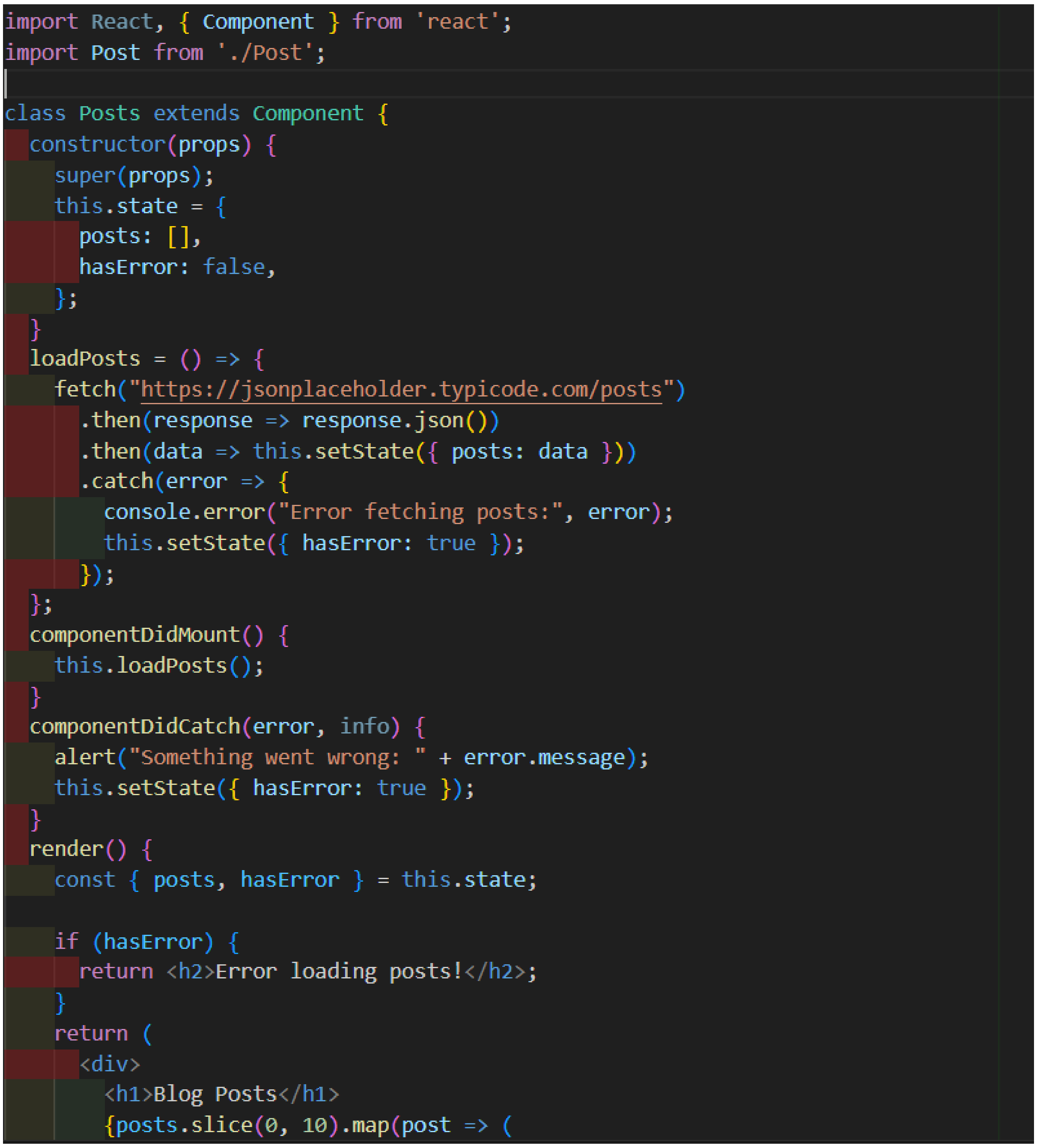
CODE:



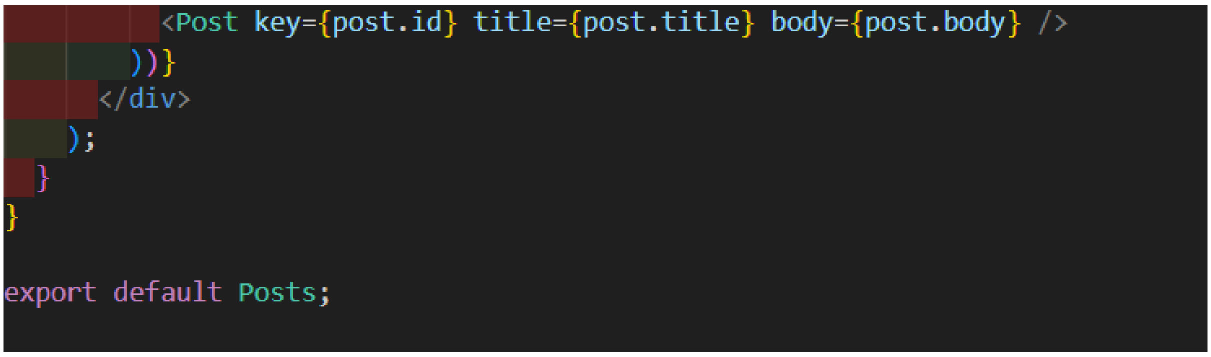
```
import React from 'react';

const Post = ({ title, body }) => {
  return (
    <div style={{ marginBottom: "20px" }}>
      <h3>{title}</h3>
      <p>{body}</p>
    </div>
  );
};

export default Post;
```

Step 3: Create Class Component Posts.js

CODE:



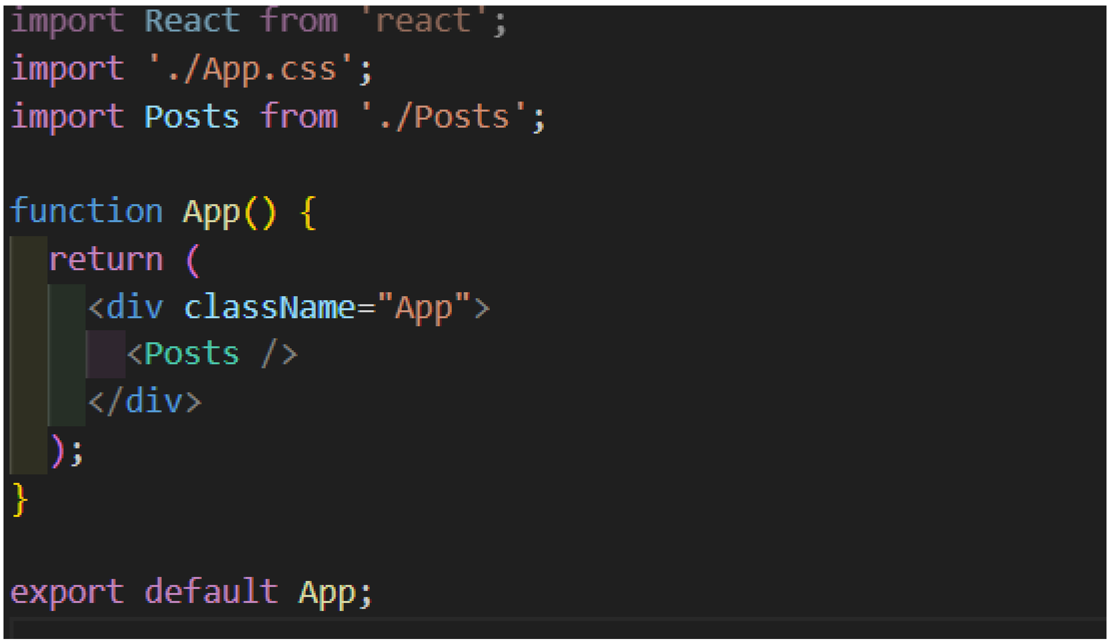
```
import React, { Component } from 'react';
import Post from './Post';

class Posts extends Component {
  constructor(props) {
    super(props);
    this.state = {
      posts: [],
      hasError: false,
    };
  }
  loadPosts = () => {
    fetch("https://jsonplaceholder.typicode.com/posts")
      .then(response => response.json())
      .then(data => this.setState({ posts: data }))
      .catch(error => {
        console.error("Error fetching posts:", error);
        this.setState({ hasError: true });
      });
  };
  componentDidMount() {
    this.loadPosts();
  }
  componentDidCatch(error, info) {
    alert("Something went wrong: " + error.message);
    this.setState({ hasError: true });
  }
  render() {
    const { posts, hasError } = this.state;

    if (hasError) {
      return <h2>Error loading posts!</h2>;
    }
    return (
      <div>
        <h1>Blog Posts</h1>
        {posts.slice(0, 10).map(post => (
```



```
        <Post key={post.id} title={post.title} body={post.body} />
      ))}
    </div>
  );
}
}

export default Posts;
```

Step 4: Use Posts in App.js

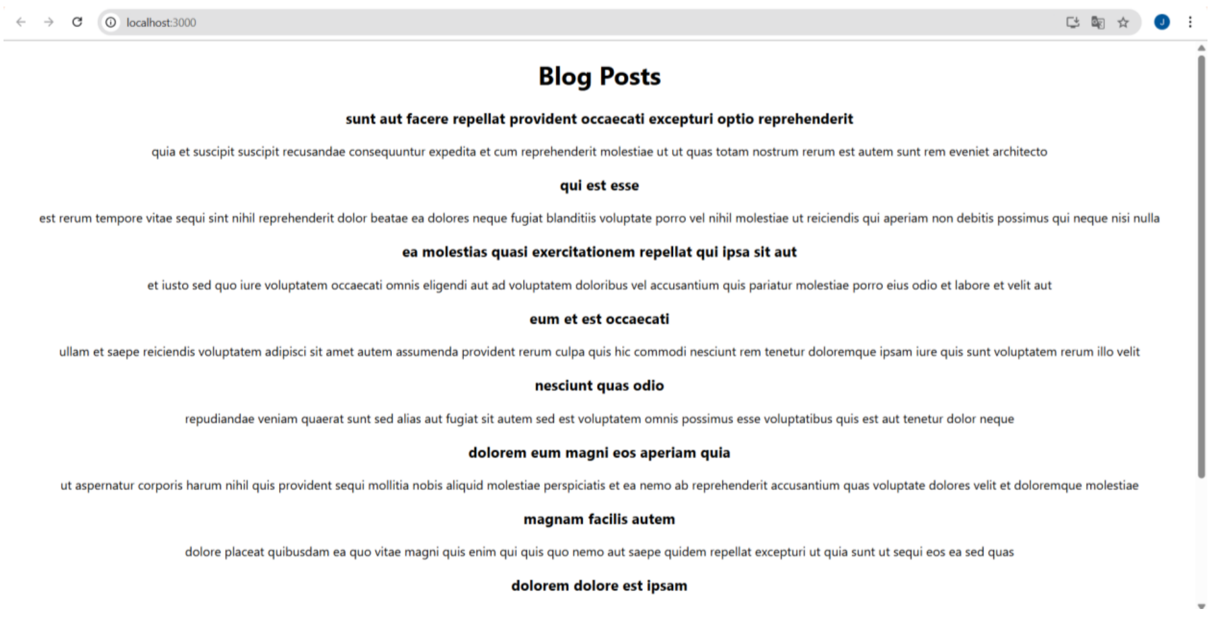
CODE:



```
import React from 'react';
import './App.css';
import Posts from './Posts';

function App() {
  return (
    <div className="App">
      <Posts />
    </div>
  );
}

export default App;
```

Step 5: Run the Application

OUTPUT:



Blog Posts

sunt aut facere repellat provident occaecati excepturi optio reprehenderit

quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto

qui est esse

est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla

ea molestias quasi exercitationem repellat qui ipsa sit aut

et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et velit aut

eum et est occaecati

ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque ipsam iure quis sunt voluptatem rerum illo velit

nesciunt quas odio

repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque

dolorem eum magni eos aperiam quia

ut aspernatur corporis harum nihil quis provident sequi mollitia nobis aliquid molestiae perspiciatis et ea nemo ab reprehenderit accusantium quas voluptate dolores velit et doloremque molestiae

magnam facilis autem

dolore placeat quibusdam ea quo vitae magni quis enim qui quis quo nemo aut saepe quidem repellat excepturi ut quia sunt ut sequi eos ea sed quas

dolorem dolore est ipsam

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:

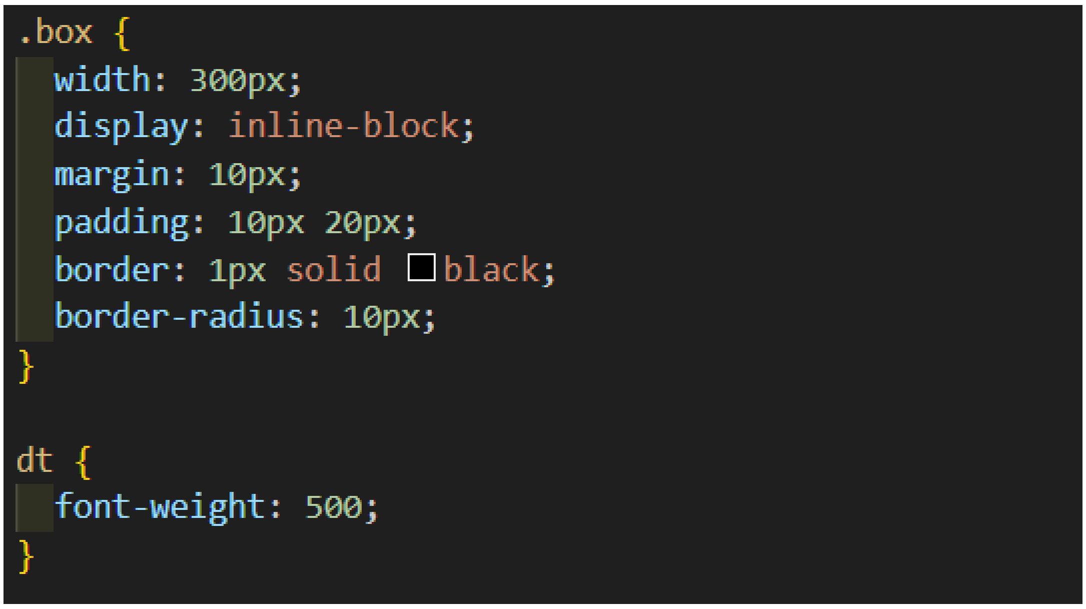
```
cd cohorttracker

npm install
```

- Open the project in VS Code:

```
code .
```

Step 2 : Create the CSS Module

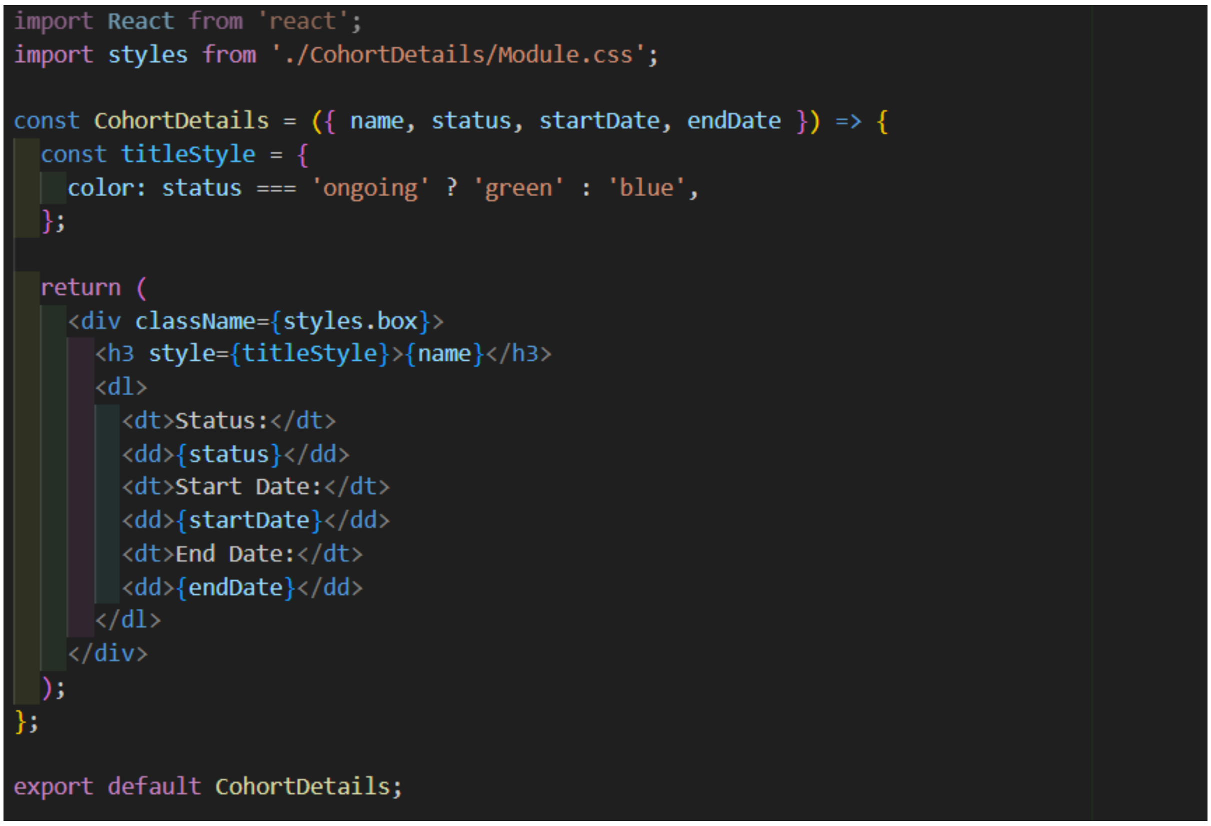
CODE:



```css
.box {
  width: 300px;
  display: inline-block;
  margin: 10px;
  padding: 10px 20px;
  border: 1px solid black;
  border-radius: 10px;
}

dt {
  font-weight: 500;
}
```

Step 3 : Create CohortDetails.js

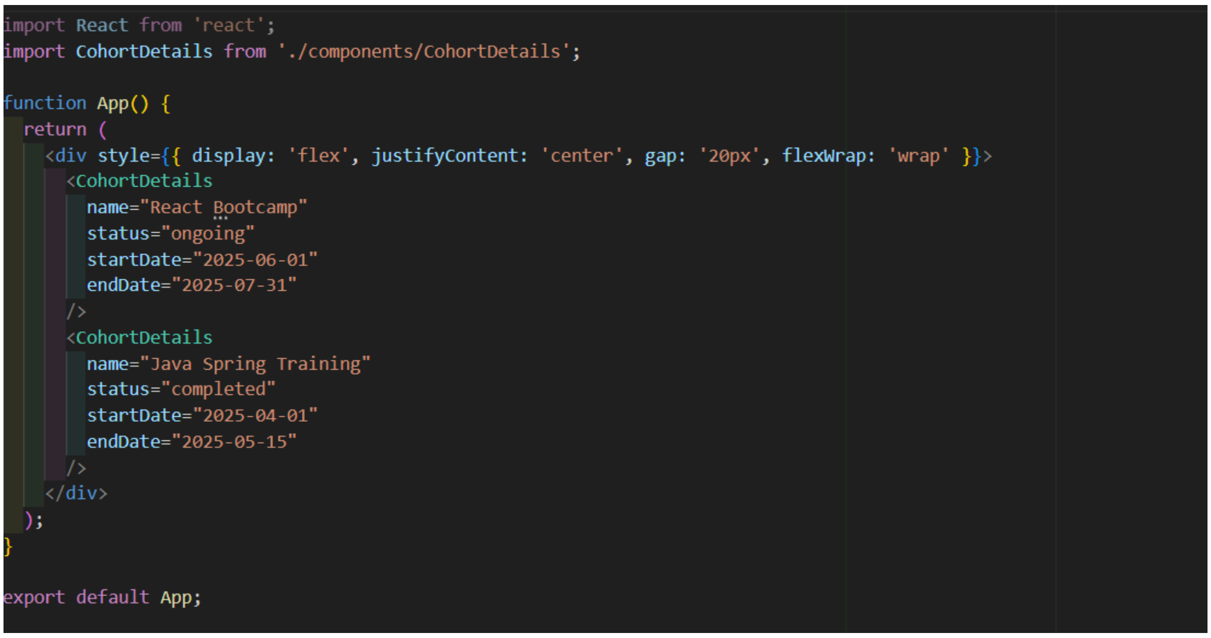
CODE:



```javascript
import React from 'react';
import styles from './CohortDetails/Module.css';

const CohortDetails = ({ name, status, startDate, endDate }) => {
  const titleStyle = {
    color: status === 'ongoing' ? 'green' : 'blue',
  };

  return (
    <div className={styles.box}>
      <h3 style={titleStyle}>{name}</h3>
      <dl>
        <dt>Status:</dt>
        <dd>{status}</dd>
        <dt>Start Date:</dt>
        <dd>{startDate}</dd>
        <dt>End Date:</dt>
        <dd>{endDate}</dd>
      </dl>
    </div>
  );
};

export default CohortDetails;
```

Step 4 : Create App.js

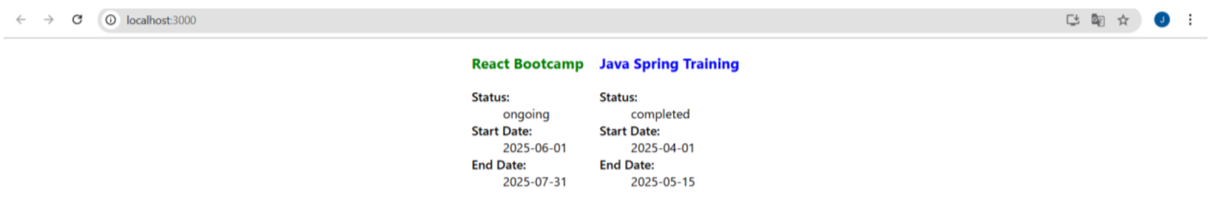
CODE:



```
import React from 'react';
import CohortDetails from './components/CohortDetails';

function App() {
  return (
    <div style={{ display: 'flex', justifyContent: 'center', gap: '20px', flexWrap: 'wrap' }}>
      <CohortDetails
        name="React Bootcamp"
        status="ongoing"
        startDate="2025-06-01"
        endDate="2025-07-31"
      />
      <CohortDetails
        name="Java Spring Training"
        status="completed"
        startDate="2025-04-01"
        endDate="2025-05-15"
      />
    </div>
  );
}

export default App;
```

Step 5 : Run the application

OUTPUT:



```
localhost:3000

React Bootcamp     Java Spring Training

Status:            Status:
    ongoing            completed
Start Date:        Start Date:
    2025-06-01         2025-04-01
End Date:          End Date:
    2025-07-31         2025-05-15
```