**Superset-Id: 6364597  
Name: SRINJOY PAUL**

**Week-6 REACT HANDSON**

**OBJECTIVES:  
1) Define SPA and its benefits   
Answer:** A Single Page Application (SPA) is a type of web application that loads just one HTML file and dynamically modifies its content based on user actions without reloading the entire page.

**Benefits:**  
It offers quicker transitions between views, lowers the burden on the server, delivers a seamless and fluid user experience, and avoids full-page reloads.

1. **Define React and identify its working  
   Answer:** React is a JavaScript library used to create user interfaces by composing reusable UI components. It operates by using a virtual DOM, which allows it to update only the elements in the actual DOM that have changed, ensuring efficient rendering.
2. **Identify the differences between SPA and MPA  
   Answer:   
   SPA (Single Page Application):**  
   Uses one HTML page; navigation is handled on the client side, and content changes dynamically without reloading the entire page.

**MPA (Multi Page Application):**  
Consists of multiple HTML pages; each user interaction results in a new page being fetched from the server.

1. **Explain Pros & Cons of Single-Page Application**

**Answer:   
Pros:**Offers quick page transitions, improves overall user experience, and consumes less bandwidth after the initial load.

**Cons:**Can face difficulties with search engine optimization (SEO), may have a larger initial loading time, and relies heavily on JavaScript to function properly.

1. **Define virtual DOM  
   Answer:** The virtual DOM is an in-memory representation of the actual DOM. React leverages it to track changes efficiently and update only the affected parts of the real DOM, enhancing performance.

**6)Explain Features of React  
Answer:**React offers a component-driven structure, utilizes a virtual DOM for efficient updates, supports one-way data binding, uses JSX for writing UI, and promotes reusable interface components.

1. **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.**

**Code:**

import React from "react";

function App() {

  return (

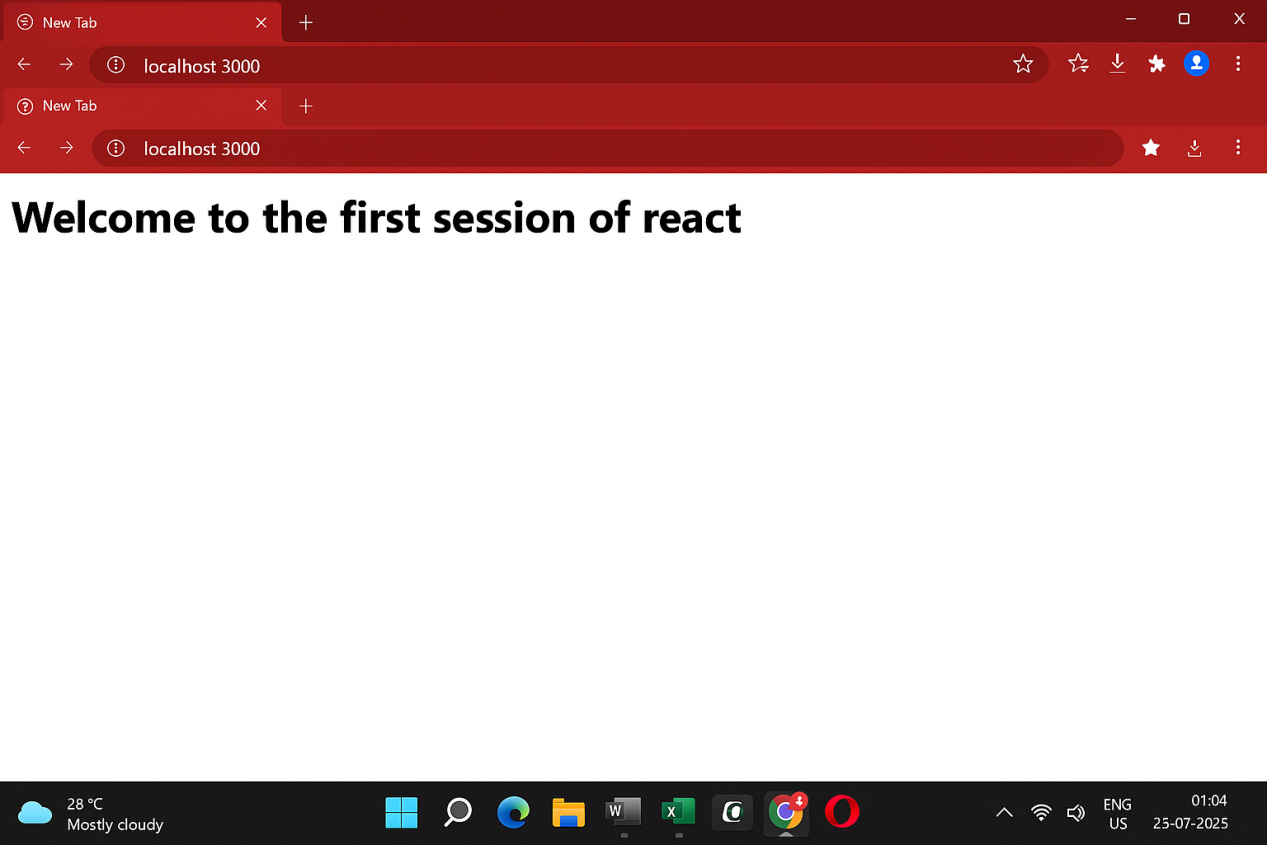
    <h1>Welcome to the first session of react</h1>

    );

}

export default App;

**Output:**



**OBJECTIVES**

1. **Explain React components  
   Answer:**React components are modular and self-contained building blocks of a user interface. They handle their own data (state) and are designed to be reused throughout the application.
2. **Identify the differences between components and JavaScript functions  
   Answer:**While JavaScript functions are used for executing logic or calculations, React components are designed to return visual elements (using JSX) and also handle UI updates and internal state.
3. **Identify the types of components  
   Answer:  
   Class components:**Built using ES6 classes, these components can hold their own state and utilize lifecycle methods**.  
   Function components:**Written as standard JavaScript functions that return JSX; they can handle state and effects using hooks (from React 16.8 onward).
4. **Explain class component  
   Answer:**A class-based React component is created by extending React.Component. It includes features like constructors and lifecycle methods for managing behavior and data.
5. **Explain function component  
   Answer:**A function component is a basic JavaScript function that returns JSX to display UI. Starting with React 16.8, hooks can be used in them to manage state and lifecycle-like behavior.
6. **Define component constructor**

**Answer:**In class components, the constructor(props) method is used to set up initial state and bind class methods to the component instance.

1. **Define render() function  
   Answer:**The render() method is essential in class components and is responsible for returning the JSX that defines what appears on the screen.
2. **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.**

**Home.js:**

import React from 'react';

function Home() {

return (

<div>

<h2>Welcome to the Home page of Student Management Portal</h2>

</div>

);

}

export default Home;

**About.js:**

import React from 'react';

function About() {

return (

<div>

<h2>Welcome to the About page of the Student Management Portal</h2>

</div>

);

}

export default About;

**Contact.js:**

import React from 'react';

function Contact() {

return (

<div>

<h2>Welcome to the Contact page of the Student Management Portal</h2>

</div>

);

}

export default Contact;

**App.js:**

import React from 'react';

import Home from './Components/Home';

import About from './Components/About';

import Contact from './Components/Contact';

function App() {

return (

<div className="App">

<Home />

<About />

<Contact />

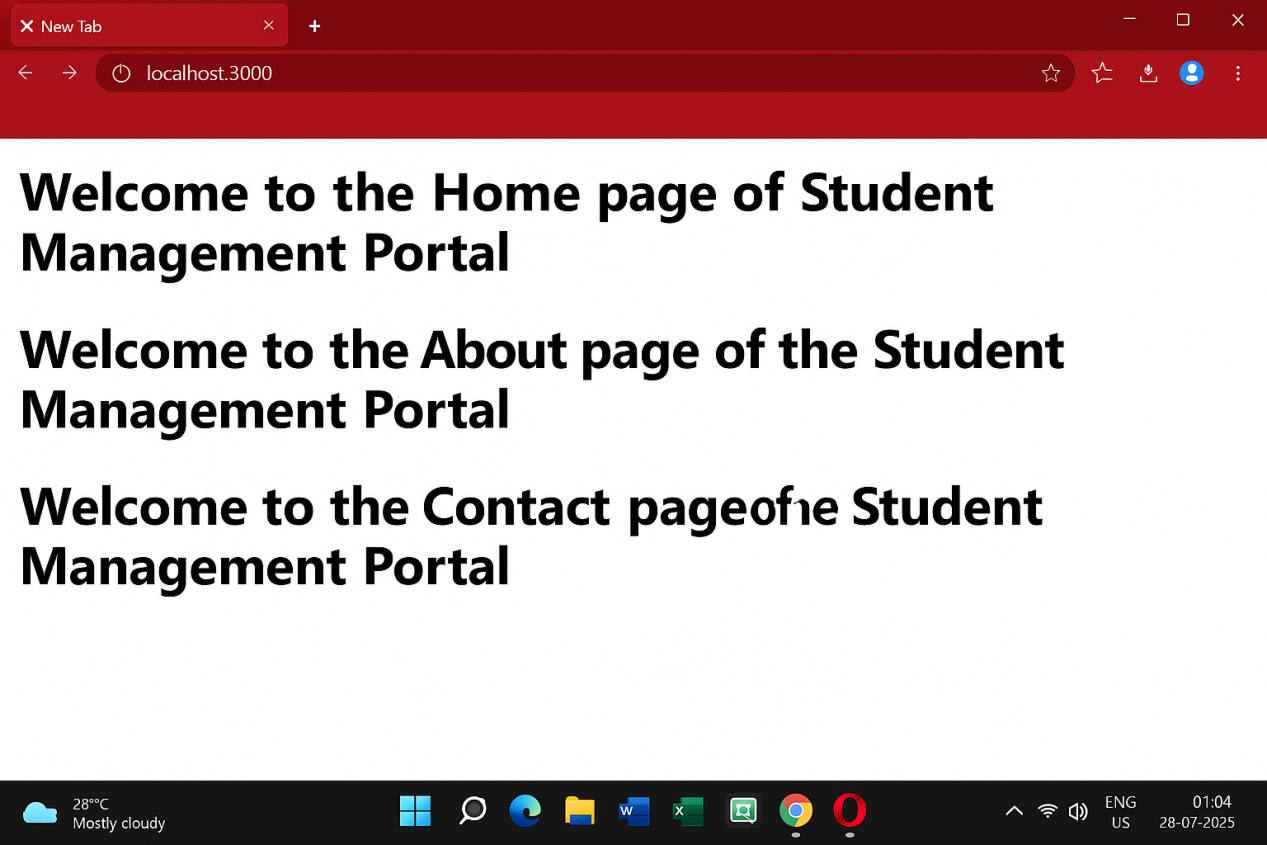
</div>

);

}

export default App;

**Output:**

****

**OBJECTIVES  
1) Explain React components  
Answer:**React components are self-contained and reusable units used to build user interfaces. Each component returns elements that describe how a part of the UI should appear. By breaking the UI into smaller parts, components help organize the application structure and make it easier to develop and maintain.

**2)Identify the differences between components and JavaScript functions  
Answer:  
Goal: T**raditional JavaScript functions are used to execute logic or return data, while React components are designed to produce JSX that structures the visual part of an app.

**Functionality:** React components are meant to participate in UI rendering and can maintain their own data (state), handle effects, and use lifecycle hooks (in class or function form). Regular JavaScript functions cannot do this.

**Naming convention:** React components are usually named with an uppercase first letter (like UserCard), whereas ordinary functions can start with lowercase or any naming format**.**

1. **Identify the types of components  
   Answer:  
   Class Components:** Created using ES6 class syntax, these components can handle internal state and respond to component lifecycle events.  
   **Function Components:** Defined using JavaScript functions, they were initially stateless but now can control state and side effects through hooks.
2. **Explain class component  
   Answer:**A class component is defined as a JavaScript class that inherits from React.Component. It must include a render() method that returns JSX content. It can also: **Use this.state to store local state  
   Use lifecycle methods such as componentDidMount  
   Access input data through this.props**
3. **Explain function component  
   Answer:**A function component is written as a standard JavaScript function that returns JSX elements to render the UI. Since React 16.8, these components can also handle state and lifecycle behavior using React hooks like useState and useEffect.
4. **Define component constructor**

**Answer:**In class-based components, the constructor is a special method that gets invoked first. It’s mainly used to: **Set up the initial state with this.state  
Bind class methods to the component instance if needed**This runs before the component appears on screen.

1. **Define render() function**

**Answer:**The render() method is necessary in class components and is responsible for returning JSX that represents the component’s UI. **It updates the view whenever there are changes in props or state  
It defines what should be displayed on the screen**

**3) 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.**

**CalculateScore.js:**

import React from "react";

import "../Stylesheets/mystyle.css";

function CalculateScore() {

const name = "John Doe";

const school = "Springfield High";

const total = 450;

const goal = 500;

const average = ((total / goal) \* 100).toFixed(2);

return (

<div className="score-container">

<h2>Student Score Summary</h2>

<p><strong>Name:</strong> {name}</p>

<p><strong>School:</strong> {school}</p>

<p><strong>Total Score:</strong> {total}</p>

<p><strong>Goal:</strong> {goal}</p>

<p><strong>Average Score:</strong> {average}%</p>

</div>

);

}

export default CalculateScore;

**mystyle.css:**

.score-container {

border: 2px solid #4CAF50;

padding: 20px;

margin: 20px;

background-color: #f0f8ff;

font-family: Arial, sans-serif;

}

.score-container h2 {

color: #4CAF50;

}

**App.js:**

import React from 'react';

import CalculateScore from './Components/CalculateScore';

function App() {

return (

<div className="App">

<CalculateScore />

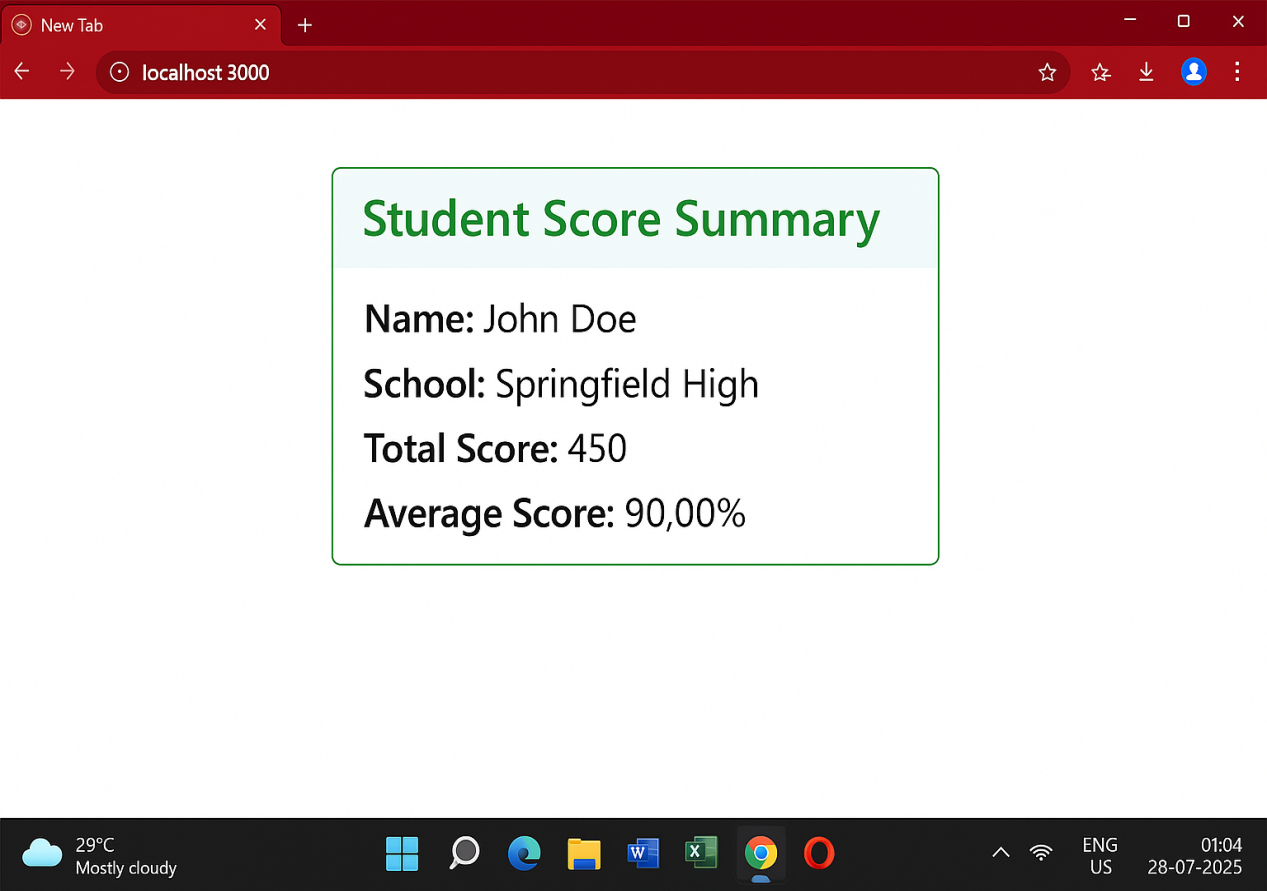
</div>

);

}

export default App;

**Output:**

****

**OBJECTIVES  
1)Explain the need and Benefits of component life cycle  
Answer:  
Need:**Knowing the React component lifecycle is important because it helps developers handle how a component behaves throughout its lifespan—from being created and displayed, to being updated, and finally taken out of the DOM.

**Benefits:**

* Enables proper setup of data when a component first mounts (like calling an API)
* Allows the interface to refresh based on changes in state or props
* Ensures cleanup of resources such as intervals or event listeners before the component is unmounted, avoiding memory issues
* Improves performance by controlling when updates should happen

1. **Identify various life cycle hook methods  
   Answer:**The key lifecycle hooks you’ll encounter in class components include:

* constructor() – initializes state and binds methods
* componentDidMount() – runs after the component is added to the DOM
* componentDidUpdate() – triggers after the component re-renders due to updates
* componentWillUnmount() – called right before the component is deleted
* (Other notable methods include shouldComponentUpdate() and getDerivedStateFromProps().)

**3)List the sequence of steps in rendering a component  
Answer:**The typical rendering flow of a React class-based component follows these steps:

* **Mounting:**

1)The constructor() runs first to set up initial state

2)The render() method generates the component’s structure

3)componentDidMount() is called right after the component appears on the screen for tasks like data fetching

* **Updating:**1)When props or state change, React re-invokes render()  
  2)After the update, componentDidUpdate() is executed
* **Unmounting:**

1)Before the component is removed from the 2)page, componentWillUnmount() is invoked to do final cleanup

1. **Create a new react application using *create-react-app* tool with the name as “blogapp”**

**Post.js:**

class Post {

constructor(userId, id, title, body) {

this.userId = userId;

this.id = id;

this.title = title;

this.body = body;

}

}

export default Post;

**Posts.js:**

import React, { Component } from 'react';

import Post from './Post';

class Posts extends Component {

constructor(props) {

super(props);

this.state = {

posts: [],

error: null

};

}

loadPosts = () => {

fetch('https://jsonplaceholder.typicode.com/posts')

.then(response => {

if (!response.ok) {

throw new Error("Network response was not ok.");

}

return response.json();

})

.then(data => {

const postObjects = data.map(

item => new Post(item.userId, item.id, item.title, item.body)

);

this.setState({ posts: postObjects });

})

.catch(error => {

this.setState({ error });

throw error; // Triggers componentDidCatch

});

};

componentDidMount() {

this.loadPosts();

}

componentDidCatch(error, info) {

alert("An error occurred while loading posts.");

console.error("Error caught in component:", error, info);

}

render() {

const { posts, error } = this.state;

if (error) {

return <p style={{ color: 'red' }}>Something went wrong!</p>;

}

return (

<div>

<h2>Blog Posts</h2>

{posts.map(post => (

<div key={post.id} style={{ borderBottom: "1px solid #ccc", marginBottom: "15px" }}>

<h3>{post.title}</h3>

<p>{post.body}</p>

</div>

))}

</div>

);

}

}

export default Posts;

**App.js:**

import React from 'react';

import Posts from './Posts';

function App() {

return (

<div className="App">

<Posts />

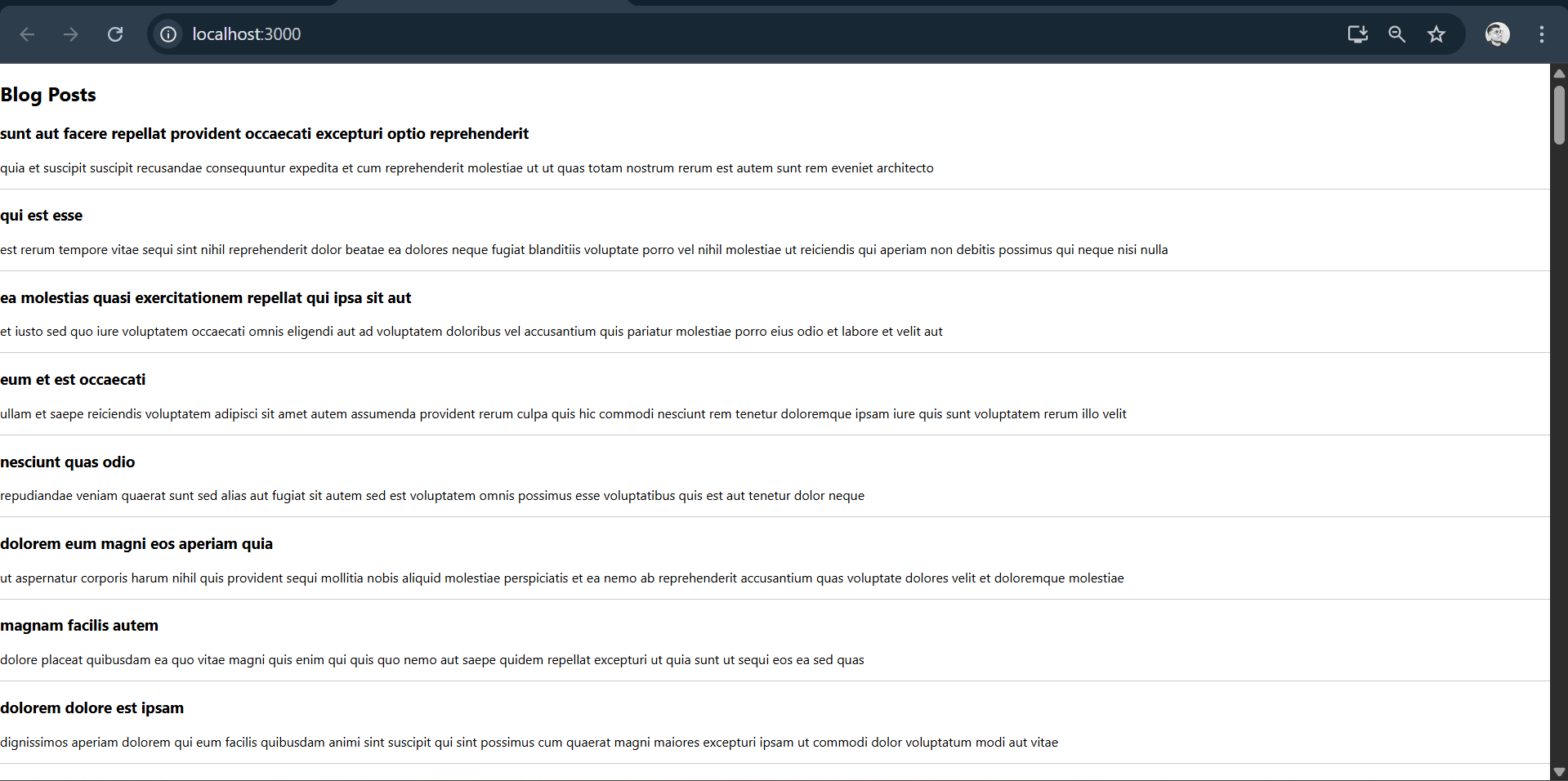
</div>

);

}

export default App;

**Output:**



**OBJECTIVES**

1. **Understanding the need for styling react component  
   Answer:**Applying styles to React components is crucial for creating attractive, intuitive, and brand-consistent user interfaces. Styling helps define the layout, draw attention to important content, and ensures the interface functions well on various screen sizes and resolutions.
2. **Working with CSS Modules and Inline Styles  
   Answer:  
    CSS Modules:**CSS Modules allow you to write CSS in a modular way where class names are locally scoped by default. This eliminates the risk of style clashes. In React, you bring in the CSS Module file (like styles.module.css) and use it via an object that maps class names to unique hashed names.

**Inline Styles:**Inline styling is done by setting the style attribute on a React element. It uses a JavaScript object to define CSS properties (e.g., { color: "red", fontSize: "20px" }). These styles affect only the targeted element, preventing interference, though they don't support features like media queries or pseudo-classes.

**How to Write Styles Using CSS Modules**

1. First, create a CSS file ending in .module.css, for example:  
      Button.module.css
2. Then, define your CSS classes inside it like this:

css

/\* Button.module.css \*/

.primary {

background-color: blue;

color: white;

padding: 10px 20px;

border-radius: 4px;

}

1. Now, bring this CSS module into your React component:

javascript

import styles from './Button.module.css';

4) Apply styles to components using the className and style properties

**Using className with CSS Modules:  
Assign a class from the imported CSS Module:**

javascript

<button className={styles.primary}>Click Me</button>

* **Using style for inline styles:**Apply a JavaScript object directly to the style prop:

javascript

<button style={{ backgroundColor: "blue", color: "white" }}>Click Me</button>

**5)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**

**CohortDetails.module.css:**

.box {

width: 300px;

display: inline-block;

margin: 10px;

padding: 10px 20px;

border: 1px solid black;

border-radius: 10px;

}

dt {

font-weight: 500;

}

**CohortDetails.js:**

import React from 'react';

import styles from './CohortDetails.module.css';

function CohortDetails({ cohort }) {

const statusStyle = {

color: cohort.status === 'ongoing' ? 'green' : 'blue'

};

return (

<div className={styles.box}>

<h3 style={statusStyle}>{cohort.name}</h3>

<dl>

<dt>Trainer:</dt>

<dd>{cohort.trainer}</dd>

<dt>Status:</dt>

<dd>{cohort.status}</dd>

<dt>Start Date:</dt>

<dd>{cohort.startDate}</dd>

<dt>End Date:</dt>

<dd>{cohort.endDate}</dd>

</dl>

</div>

);

}

export default CohortDetails;

**App.js:**

import React from 'react';

import CohortDetails from './CohortDetails';

function App() {

const cohorts = [

{

name: 'React Bootcamp',

trainer: 'John Doe',

status: 'ongoing',

startDate: '2025-07-01',

endDate: '2025-08-15'

},

{

name: 'Node.js Essentials',

trainer: 'Jane Smith',

status: 'completed',

startDate: '2025-06-01',

endDate: '2025-07-15'

}

];

return (

<div>

{cohorts.map((cohort, index) => (

<CohortDetails key={index} cohort={cohort} />

))}

</div>

);

}

export default App;

**Output:**

