**WEEK-6**

**1. ReactJS-HOL**

**App.js:**

import './App.css';

function App() {

  return (

    <div className="App">

      <h1 className="center-heading">

        Welcome the first session of React

      </h1>

    </div>

  );

}

export default App;

**App.css:**

.App {

  text-align: center;

  padding-top: 100px;

}

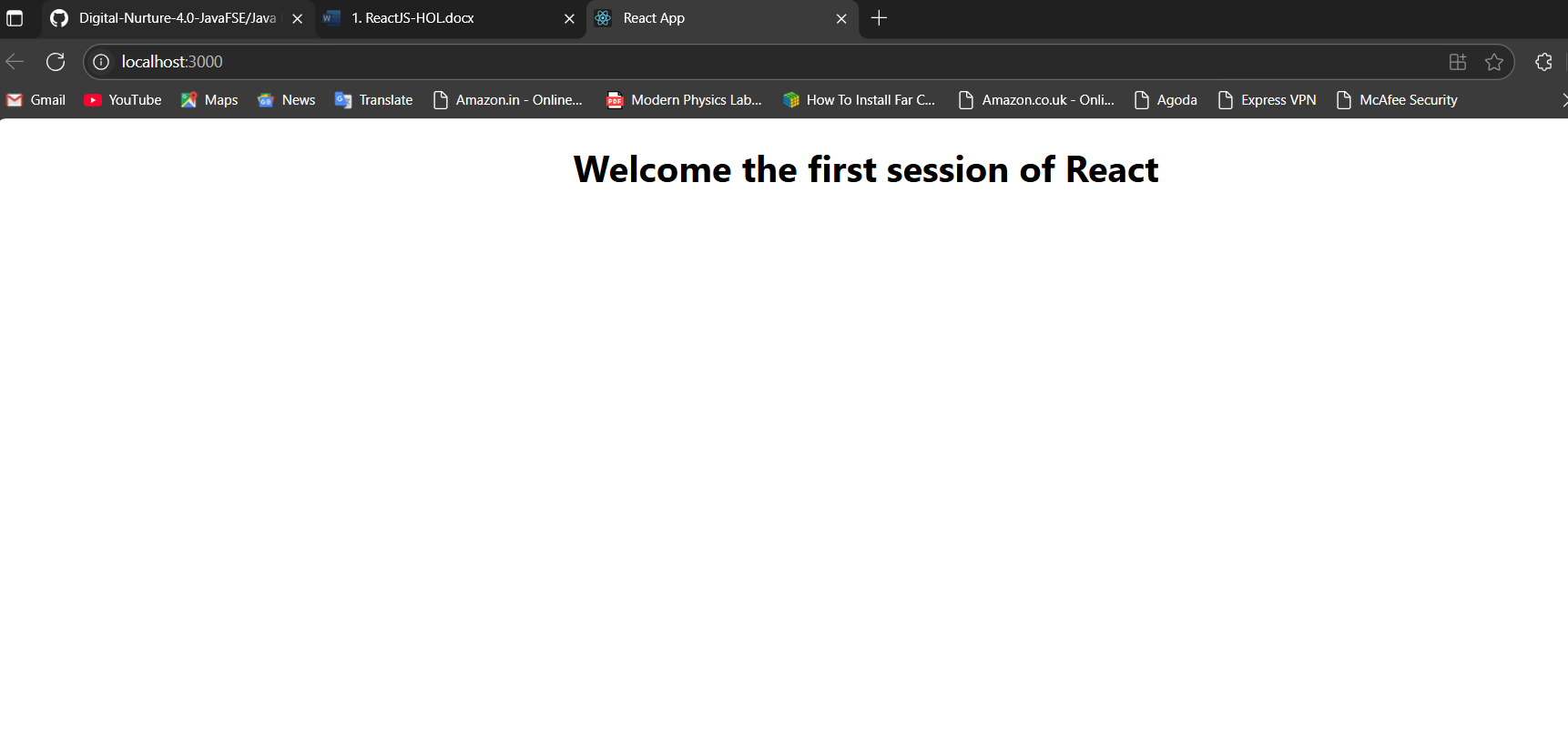
.center-heading {

  color: #333;

  font-size: 2rem;

}

**Output:**



**2. ReactJS-HOL**

**Home.js:**

import React, { Component } from 'react';

class Home extends Component {

    render() {

        return (

            <div>

                <h1>Welcome to the Home page of Student Management Portal</h1>

            </div>

        );

    }

}

export default Home;

**About.js:**

import React, { Component } from 'react';

class About extends Component {

    render() {

        return (

            <div>

                <h1>Welcome to the About page of the Student Management Portal</h1>

            </div>

        );

    }

}

export default About;

**Contact.js:**

import React, { Component } from 'react';

class Contact extends Component {

    render() {

        return (

            <div>

                <h1>Welcome to the Contact page of the Student Management Portal</h1>

            </div>

        );

    }

}

export default Contact;

**App.js:**

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 />

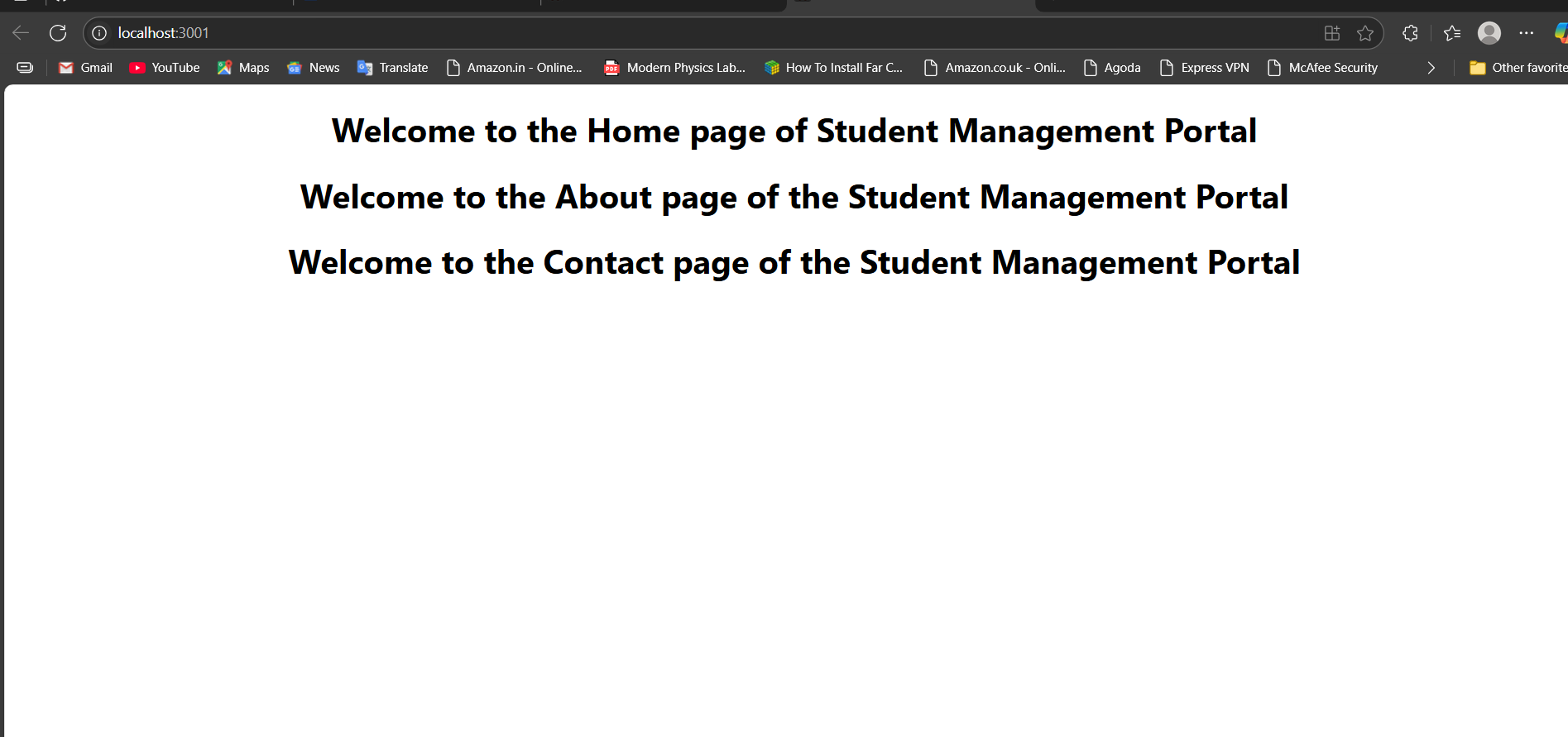
    </div>

  );

}

export default App;

**Output:**



**3. ReactJS-HOL**

**CalculateScore.js:**

import React from 'react';

import '../Stylesheets/mystyle.css';

const CalculateScore = (props) => {

  const averageScore = props.Total / props.Goal;

  return (

    <div className="container">

      <h2>Student Score Calculator</h2>

      <table>

        <tbody>

          <tr>

            <td>Name:</td>

            <td>{props.Name}</td>

          </tr>

          <tr>

            <td>School:</td>

            <td>{props.School}</td>

          </tr>

          <tr>

            <td>Total:</td>

            <td>{props.Total}</td>

          </tr>

          <tr>

            <td>Goal:</td>

            <td>{props.Goal}</td>

          </tr>

          <tr>

            <td>Average Score:</td>

            <td>{averageScore.toFixed(2)}</td>

          </tr>

        </tbody>

      </table>

    </div>

  );

};

export default CalculateScore;

**mystyle.css:**

.container {

  width: 400px;

  margin: 50px auto;

  padding: 20px;

  border: 2px solid #333;

  border-radius: 10px;

  background-color: #f9f9f9;

  font-family: Arial, sans-serif;

}

h2 {

  text-align: center;

  color: #333;

  margin-bottom: 20px;

}

table {

  width: 100%;

  border-collapse: collapse;

}

td {

  padding: 10px;

  border: 1px solid #ddd;

  font-size: 16px;

}

td:first-child {

  background-color: #e7e7e7;

  font-weight: bold;

  width: 40%;

}

td:last-child {

  background-color: #fff;

}

tr:nth-child(1) td:last-child {

  color: #1976d2;

  font-weight: bold;

}

tr:nth-child(2) td:last-child {

  color: #d32f2f;

  font-weight: bold;

}

tr:nth-child(3) td:last-child {

  color: #f57c00;

  font-weight: bold;

}

tr:nth-child(4) td:last-child {

  color: #7b1fa2;

  font-weight: bold;

}

tr:nth-child(5) td:last-child {

  color: #388e3c;

  font-weight: bold;

}

**App.js:**

import React from 'react';

import CalculateScore from './components/CalculateScore';

import './App.css';

function App() {

  return (

    <div className="App">

      <CalculateScore

        Name="John Doe"

        School="ABC High School"

        Total={85}

        Goal={100}

      />

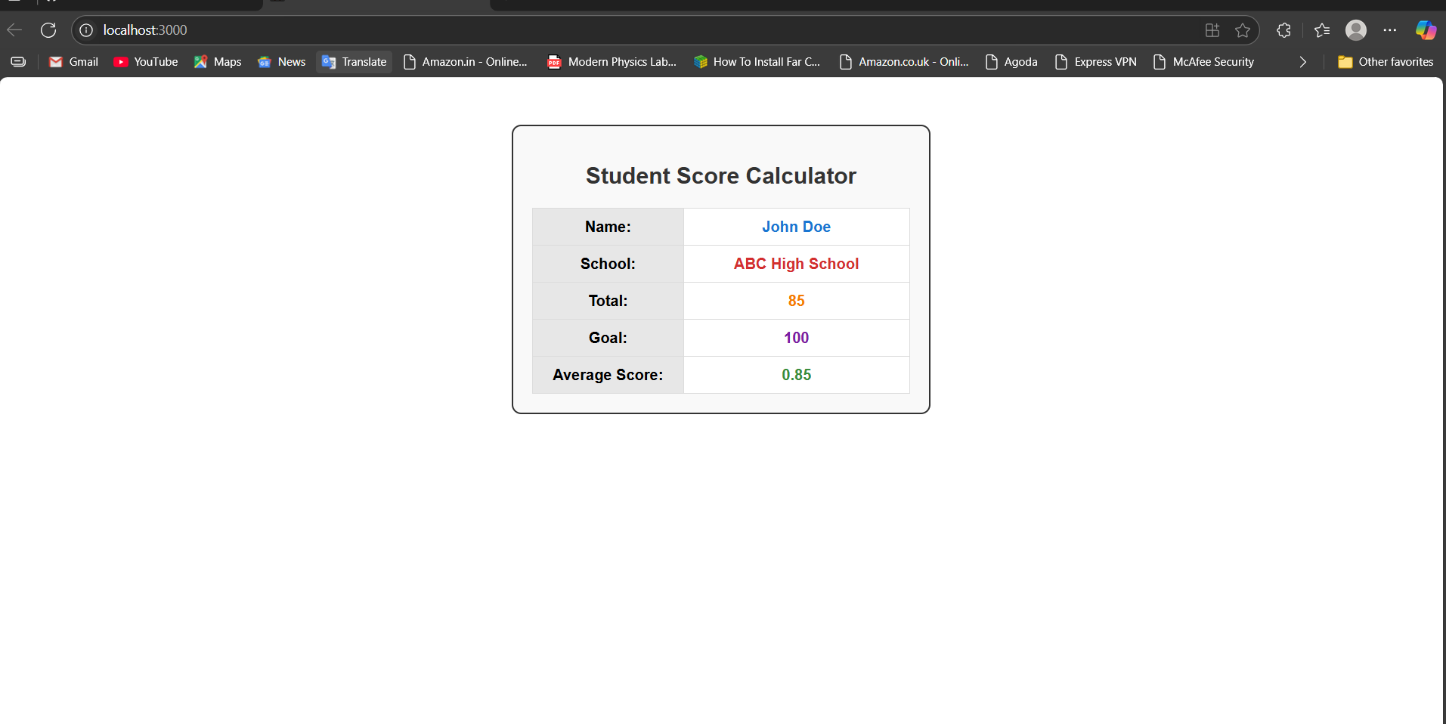
    </div>

  );

}

export default App;

**Output:**



**4. ReactJS-HOL**

**Post.js:**

class Post {

    constructor(id, title, body) {

        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: []

        };

    }

    loadPosts() {

        const englishPosts = [

            {

                id: 1,

                title: "Getting Started with React Development",

                body: "React is a powerful JavaScript library for building user interfaces. It allows developers to create reusable components and manage application state efficiently. React's component-based architecture makes it easy to build complex applications by breaking them down into smaller, manageable pieces."

            },

            {

                id: 2,

                title: "Understanding React Lifecycle Methods",

                body: "React lifecycle methods are special methods that get called at different stages of a component's life. ComponentDidMount is called after the component is mounted to the DOM, componentDidUpdate is called after updates, and componentWillUnmount is called before the component is destroyed. These methods help us control component behavior."

            },

            {

                id: 3,

                title: "Building Your First React Component",

                body: "Components are the building blocks of React applications. They can be either class-based or functional components. Class components have access to lifecycle methods and state management, while functional components use hooks for similar functionality. Both approaches have their advantages and use cases."

            },

            {

                id: 4,

                title: "State Management in React Applications",

                body: "State is a crucial concept in React that allows components to maintain and update data over time. When state changes, React automatically re-renders the component to reflect the new data. This reactive nature makes React applications dynamic and interactive, providing a smooth user experience."

            },

            {

                id: 5,

                title: "React Hooks: A Modern Development Approach",

                body: "React Hooks were introduced to allow functional components to use state and other React features. useState for state management, useEffect for side effects, and custom hooks enable developers to write cleaner, more reusable code. Hooks have revolutionized how we write React applications."

            },

            {

                id: 6,

                title: "Best Practices for React Development",

                body: "Following best practices in React development ensures maintainable and scalable applications. This includes proper component organization, efficient state management, performance optimization, and following naming conventions. Good practices lead to better code quality and team collaboration."

            },

            {

                id: 7,

                title: "Handling Events in React Components",

                body: "Event handling in React is straightforward but follows specific patterns. React uses SyntheticEvents which wrap native events to provide consistent behavior across browsers. Understanding how to properly bind event handlers and manage event data is essential for interactive applications."

            },

            {

                id: 8,

                title: "React Props and Component Communication",

                body: "Props are how parent components communicate with child components in React. They allow data to flow down the component tree and enable component reusability. Understanding how to properly pass and use props is fundamental to building effective React applications."

            },

            {

                id: 9,

                title: "Conditional Rendering in React",

                body: "React provides several ways to conditionally render components and elements. Whether using if statements, ternary operators, or logical AND operators, conditional rendering allows you to show different content based on application state and user interactions."

            },

            {

                id: 10,

                title: "React Performance Optimization Tips",

                body: "Optimizing React applications involves several techniques including proper key usage in lists, avoiding unnecessary re-renders, using React.memo for functional components, and implementing shouldComponentUpdate for class components. These optimizations ensure smooth user experiences."

            }

        ];

        setTimeout(() => {

            const postsArray = englishPosts.map(post =>

                new Post(post.id, post.title, post.body)

            );

            this.setState({ posts: postsArray });

        }, 1000);

    }

    componentDidMount() {

        console.log('Component mounted - loading posts');

        this.loadPosts();

    }

    componentDidCatch(error, errorInfo) {

        console.error('Error caught by componentDidCatch:', error, errorInfo);

        alert(`An error occurred: ${error.message}`);

    }

    render() {

        return (

            <div style={{ padding: '20px' }}>

                <h1>Blog Posts</h1>

                {this.state.posts.length === 0 ? (

                    <p>Loading posts...</p>

                ) : (

                    this.state.posts.map(post => (

                        <div key={post.id} style={{

                            marginBottom: '20px',

                            padding: '15px',

                            border: '1px solid #ddd',

                            borderRadius: '8px',

                            backgroundColor: '#f9f9f9'

                        }}>

                            <h2 style={{ color: '#333', marginBottom: '10px' }}>{post.title}</h2>

                            <p style={{ lineHeight: '1.6', color: '#666' }}>{post.body}</p>

                        </div>

                    ))

                )}

            </div>

        );

    }

}

export default Posts;

**App.js:**

import React from 'react';

import './App.css';

import Posts from './Posts';

function App() {

  return (

    <div className="App">

      <Posts />

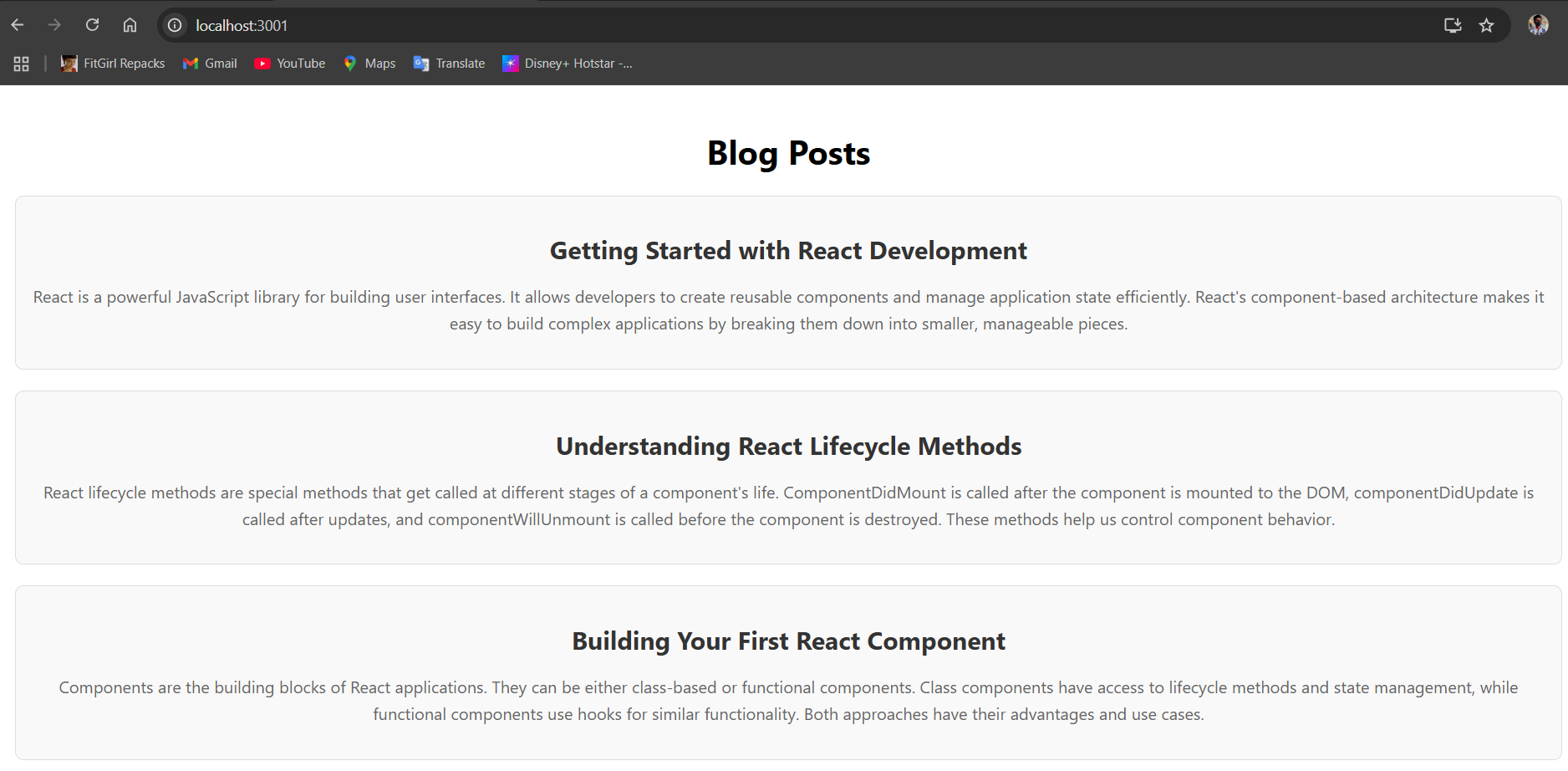
    </div>

  );

}

export default App;

**Output:**



**5. ReactJS-HOL**

**CohortDetails.js:**

import React from 'react';

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

const CohortDetails = ({ cohort }) => {

  console.log('CohortDetails rendering:', cohort);

  console.log('Styles object:', styles);

  const titleStyle = {

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

  };

  return (

    <div className={styles.box}>

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

      <dl>

        <dt>Started On</dt>

        <dd>{cohort.startDate}</dd>

        <dt>Current Status</dt>

        <dd>{cohort.status}</dd>

        <dt>Coach</dt>

        <dd>{cohort.coach}</dd>

        <dt>Trainer</dt>

        <dd>{cohort.trainer}</dd>

      </dl>

    </div>

  );

};

export default CohortDetails;

**CohortDetails.module.css:**

.box {

  width: 300px;

  display: inline-block;

  margin: 10px;

  padding: 10px 20px;

  border: 1px solid black;

  border-radius: 10px;

  vertical-align: top;

  box-sizing: border-box;

}

dt {

  font-weight: 500;

  margin-top: 10px;

  margin-bottom: 5px;

}

dd {

  margin-left: 20px;

  margin-bottom: 5px;

}

**App.css:**

.App {

  text-align: left;

  padding: 20px;

}

.cohorts-container {

  display: flex;

  flex-wrap: wrap;

  gap: 20px;

  align-items: flex-start;

}

h1 {

  margin-bottom: 30px;

  font-size: 24px;

  font-weight: bold;

}

**App.js:**

import React from 'react';

import CohortDetails from './CohortDetails';

import './App.css';

function App() {

  console.log('App component is rendering');

  const cohorts = [

    {

      id: 1,

      name: 'INTADMDF10 -.NET FSD',

      startDate: '22-Feb-2022',

      status: 'Scheduled',

      coach: 'Aathma',

      trainer: 'Jojo Jose'

    },

    {

      id: 2,

      name: 'ADM21JF014 -Java FSD',

      startDate: '10-Sep-2021',

      status: 'Ongoing',

      coach: 'Apoorv',

      trainer: 'Elisa Smith'

    },

    {

      id: 3,

      name: 'CDBJ F21025 -Java FSD',

      startDate: '24-Dec-2021',

      status: 'Ongoing',

      coach: 'Aathma',

      trainer: 'John Doe'

    }

  ];

  return (

    <div className="App">

      <h1>Cohorts Details</h1>

      <div className="cohorts-container">

        {cohorts.map(cohort => (

          <CohortDetails key={cohort.id} cohort={cohort} />

        ))}

      </div>

    </div>

  );

}

export default App;

**Output:**

