# HANDS ON 1: CREATING FIRST REACT APP

# React and Single-Page Application Concepts

## Define SPA and its benefits

SPA (Single-Page Application) is a web application that interacts with the user by dynamically rewriting the current page, rather than loading entire new pages from the server. This results in a smoother user experience.  
  
Benefits of SPA:  
- Faster navigation  
- Reduced server load  
- Better user experience  
- Efficient development with frameworks like React/Angular

## Define React and identify its working

React is a JavaScript library developed by Facebook for building user interfaces. It allows developers to build reusable UI components.  
  
Working of React:  
- Uses a component-based architecture  
- Maintains a virtual DOM  
- Efficiently updates and renders components based on data changes

## Identify the differences between SPA and MPA

SPA (Single-Page Application):  
- Loads a single HTML page  
- Uses JavaScript to update content dynamically  
- Faster interactions after initial load  
  
MPA (Multi-Page Application):  
- Loads a new HTML page for each interaction  
- Better for SEO  
- More complex navigation and server requests

## Explain Pros & Cons of Single-Page Application

Pros:  
- Faster loading after initial page load  
- Seamless user experience  
- Efficient client-side routing  
  
Cons:  
- Poor SEO support  
- Initial load time may be high  
- Can be difficult to secure

## Explain about React

React is an open-source JavaScript library for building user interfaces, primarily maintained by Meta (Facebook). It's used to build fast and interactive UIs by breaking them down into reusable components.

## Define virtual DOM

The Virtual DOM (VDOM) is a programming concept where a virtual representation of the UI is kept in memory. React uses this to update only the changed parts of the real DOM, improving performance.

## Explain Features of React

Features of React:  
- JSX (JavaScript XML) for writing HTML in JavaScript  
- Component-based architecture  
- Virtual DOM for optimized rendering  
- Unidirectional data flow  
- Strong community and ecosystem

**CODE:**

**App.js:**

import logo from './logo.svg';

import './App.css';

function App() {

  return (

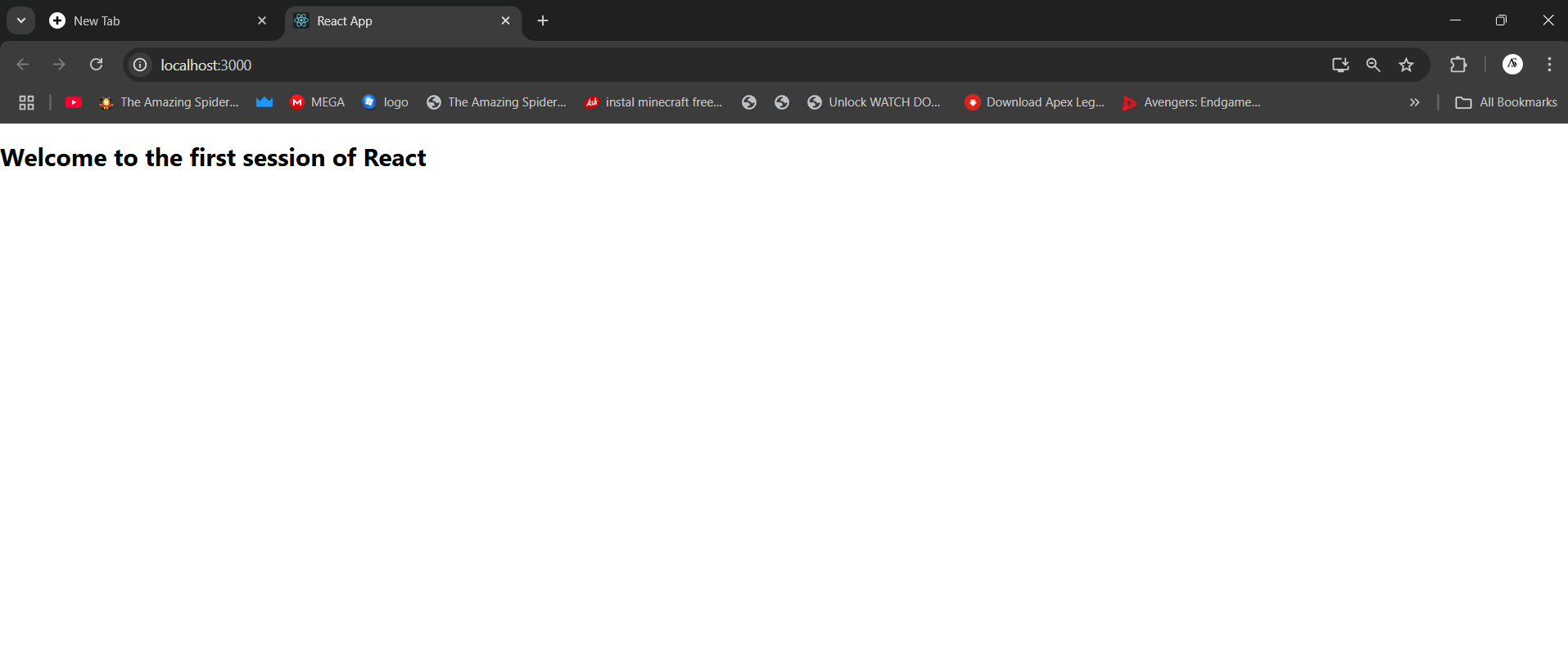
    <h1>Welcome to the first session of React</h1>

  );

}

export default App;

**OUTPUT:**



# HANDS ON 2: CREATING REACT APP FOR STUDENT MANAGEMENT PORTAL

# React Components Concepts

## Explain React components

React components are the building blocks of a React application's UI. A component is a JavaScript function or class that optionally accepts inputs (called props) and returns a React element that describes how a section of the UI should appear.

## Identify the differences between components and JavaScript functions

React Components:  
- Return JSX (UI markup)  
- Can have state and lifecycle methods  
- Used to build UI  
  
JavaScript Functions:  
- Return values or perform operations  
- Do not inherently manage state or UI  
- Used for general logic and operations

## Identify the types of components

Types of React Components:  
1. Class Components  
2. Function Components

## Explain class component

Class components are ES6 classes that extend from React.Component and must include a render() method. They can manage their own state and have access to lifecycle methods.

## Explain function component

Function components are simpler and are written as JavaScript functions. With the introduction of Hooks, they can now manage state and use lifecycle features.

## Define component constructor

The constructor is a special method used in class components for initializing state and binding event handlers. It is called before the component is mounted.

## Define render() function

The render() function is required in class components. It returns the JSX that defines the UI for that component. It is called every time the component’s state or props change.

**CODE:**

**COMPONENTS:**

**About.js:**

import React, { Component } from 'react';

class About extends Component {

    render() {

        return (

            <div>

                <h3>Welcome to the About Page of Student Management Portal</h3>

            </div>

        );

    }

}

export default About;

**Contact.js:**

import React, { Component } from 'react';

class Contact extends Component {

    render() {

        return (

            <div>

                <h3>Welcome to the Contact Page of Student Management Portal</h3>

            </div>

        );

    }

}

export default Contact;

**Home.js:**

import React, { Component } from 'react';

class Home extends Component {

    render() {

        return (

            <div>

                <h3>Welcome to the Home Page of Student Management Portal</h3>

            </div>

        );

    }

}

export default Home;

**App.js:**

import logo from './logo.svg';

import './App.css';

import Home from './Components/Home';

import About from './Components/About';

import Contact from './Components/Contact';

function App() {

  return (

    <div className="container">

      <Home />

      <About />

      <Contact />

    </div>

  );

}

export default App;

**App.css:**

.App {

  text-align: center;

}

.App-logo {

  height: 40vmin;

  pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

  .App-logo {

    animation: App-logo-spin infinite 20s linear;

  }

}

.App-header {

  background-color: #282c34;

  min-height: 100vh;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  font-size: calc(10px + 2vmin);

  color: white;

}

.App-link {

  color: #61dafb;

}

.container {

  display: flex;

  flex-direction: column;  /\* stack children vertically \*/

  justify-content: center; /\* center vertically \*/

  align-items: center;     /\* center horizontally \*/

  min-height: 100vh;       /\* full viewport height \*/

  text-align: center;      /\* optional: center text \*/

}

@keyframes App-logo-spin {

  from {

    transform: rotate(0deg);

  }

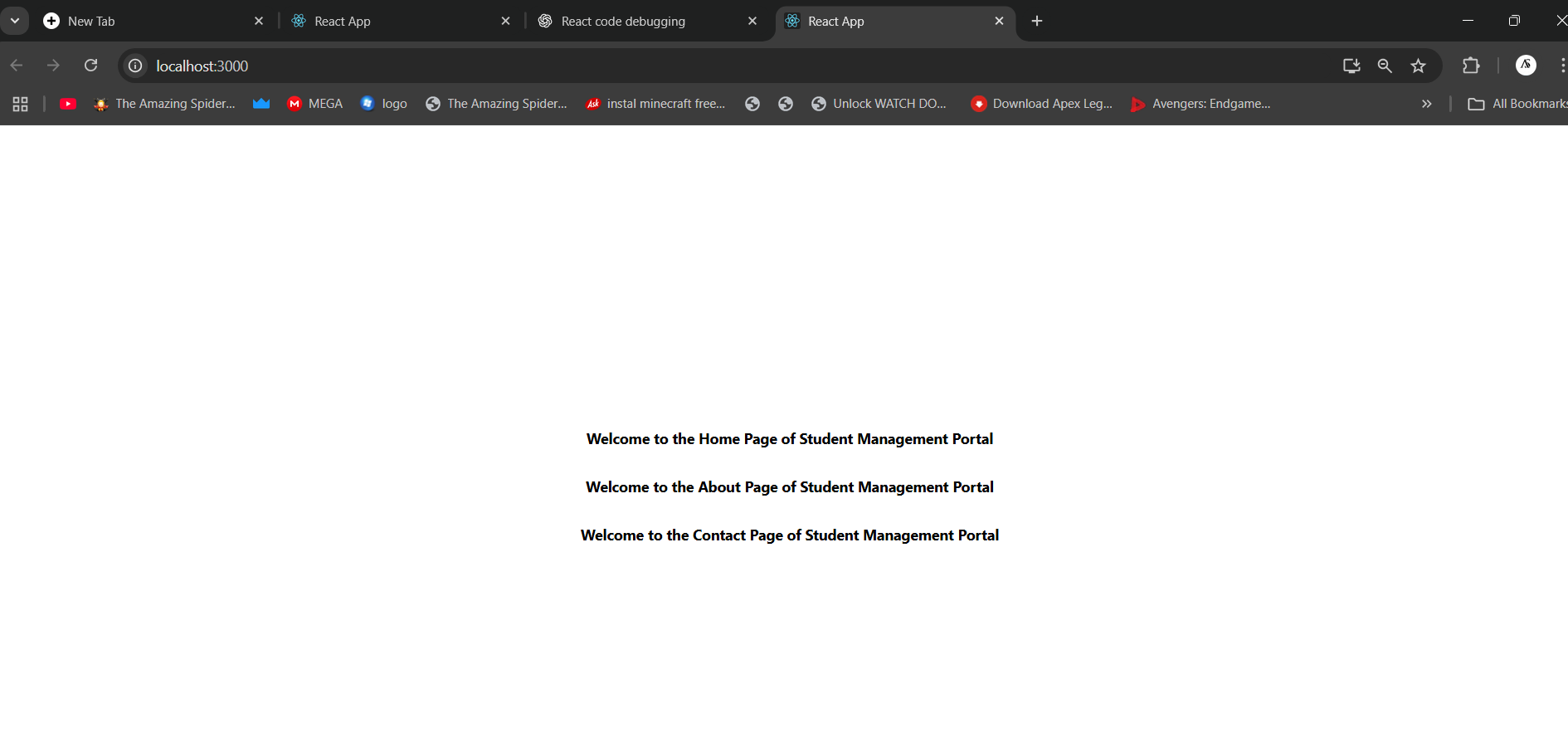
  to {

    transform: rotate(360deg);

  }

}

**OUTPUT:**

****

# HANDS ON 3: CREATING SCORE CALCULATOR REACT APP

# React Component Lifecycle Concepts

## Explain the need and Benefits of component life cycle

The component lifecycle in React allows developers to control the behavior of components at specific points during their existence (mounting, updating, and unmounting). This helps in:  
- Managing side effects (e.g., API calls)  
- Cleaning up resources (e.g., timers)  
- Improving performance and stability  
- Enhancing debugging and control

## Identify various life cycle hook methods

React Class Component Lifecycle Methods:  
- constructor()  
- static getDerivedStateFromProps()  
- render()  
- componentDidMount()  
- shouldComponentUpdate()  
- getSnapshotBeforeUpdate()  
- componentDidUpdate()  
- componentWillUnmount()  
React Function Component Hooks:  
- useEffect()  
- useState()  
- useContext()  
- useRef()

## List the sequence of steps in rendering a component

Component Rendering Lifecycle (Class Component):  
1. constructor()  
2. getDerivedStateFromProps()  
3. render()  
4. componentDidMount() (for initial render)  
  
When updating:  
1. getDerivedStateFromProps()  
2. shouldComponentUpdate()  
3. render()  
4. getSnapshotBeforeUpdate()  
5. componentDidUpdate()

**CODE:**

**COMPONENTS:**

**CalculateScore.js:**

import './Stylesheets/mystyle.css';

const percentToDecimal = (decimal) => {

  return (decimal.toFixed(2) + '%');

}

const calcScore = (total, goal) => {

  return percentToDecimal(total / goal);

}

export const CalculateScore = ({ Name, School, total, goal }) => (

  <div className="formatstyle">

    <h1><font color="Brown">Student Details:</font></h1>

    <div className="Name">

      <b><span> Name: </span></b>

      <span>{Name}</span>

    </div>

    <div className="School">

      <b><span> School: </span></b>

      <span>{School}</span>

    </div>

    <div className="Total">

      <b><span>Total:</span></b>

      <span>{total}</span>

      <span> Marks</span>

    </div>

    <div className="Score">

      <b>Score:</b>

      <span>

        {calcScore(total, goal)}

      </span>

    </div>

  </div>

);

**Stylesheets:**

**mystyle.css:**

.Name{

    font-weight:300;

color:blue;

}

.School

{

    color:crimson;

}

.Total

{

color:darkmagenta;

}

.formatstyle

{

text-align:center;

font-size:large;

}

.Score

{

color:forestgreen;

}

**APP.js:**

import { CalculateScore } from '../src/Components/CalculateScore';

function App() {

  return (

    <div>

      <CalculateScore

        Name="Steeve"

        School="DNV Public School"

        total={284}

        goal={3}

      />

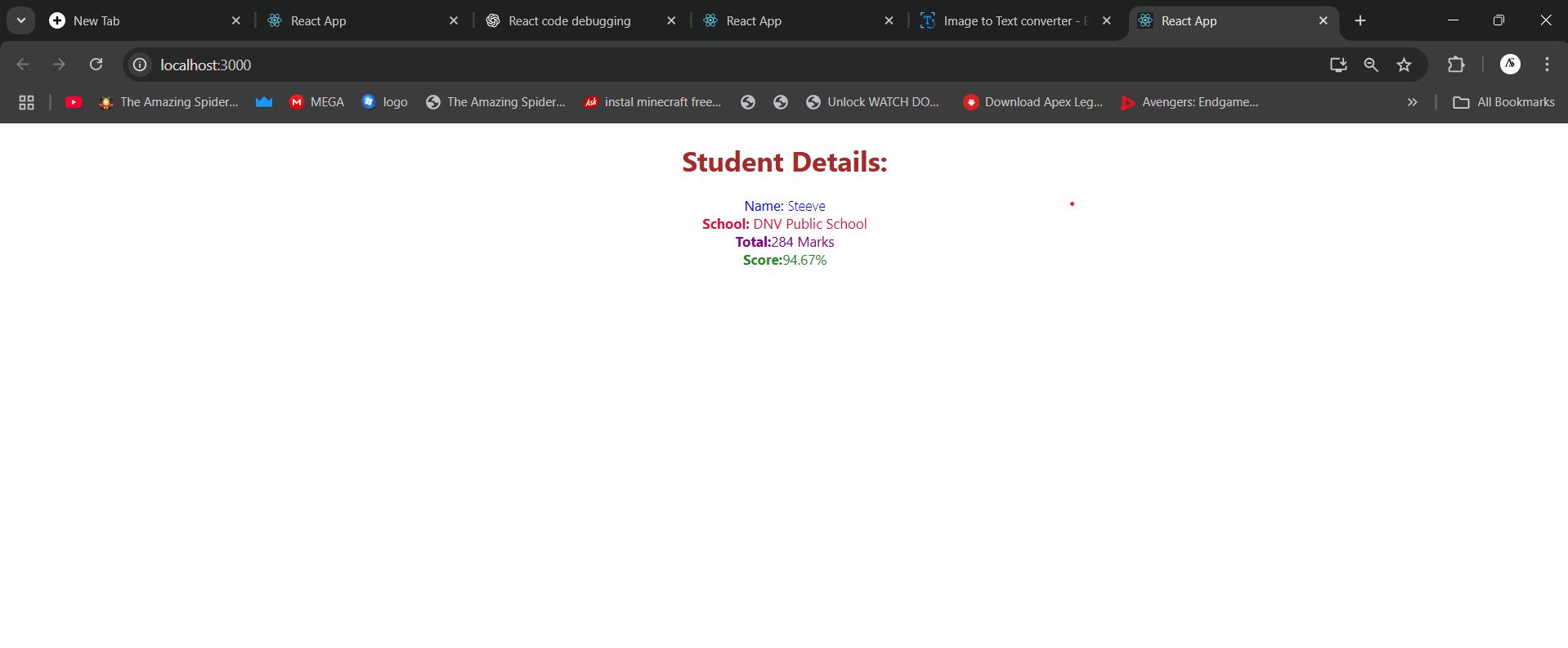
    </div>

  );

}

export default App;

**OUTPUT:**

****

# **HANDS ON 4: CREATING ‘blogapp’ REACT APP**

**CODE:**

**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';

// Figure 3: Posts Component

class Posts extends Component {

  constructor(props) {

    super(props);

    // Initialize state with empty posts list

    this.state = {

      posts: []

    };

  }

  // loadPosts() method

  loadPosts() {

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

      .then(response => response.json())

      .then(data => {

        const posts = data.map(

          p => new Post(p.id, p.title, p.body)

        );

        this.setState({ posts });

      })

      .catch(error => console.error('Error:', error));

  }

  // componentDidMount()

  componentDidMount() {

    this.loadPosts();

  }

  // componentDidCatch()

  componentDidCatch(error, info) {

    alert('An error occurred: ' + error);

  }

  // render() method

  render() {

    return (

      <div>

        <h1>Posts</h1>

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

          <div key={post.id}>

            <h2>{post.title}</h2>

            <p>{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 />

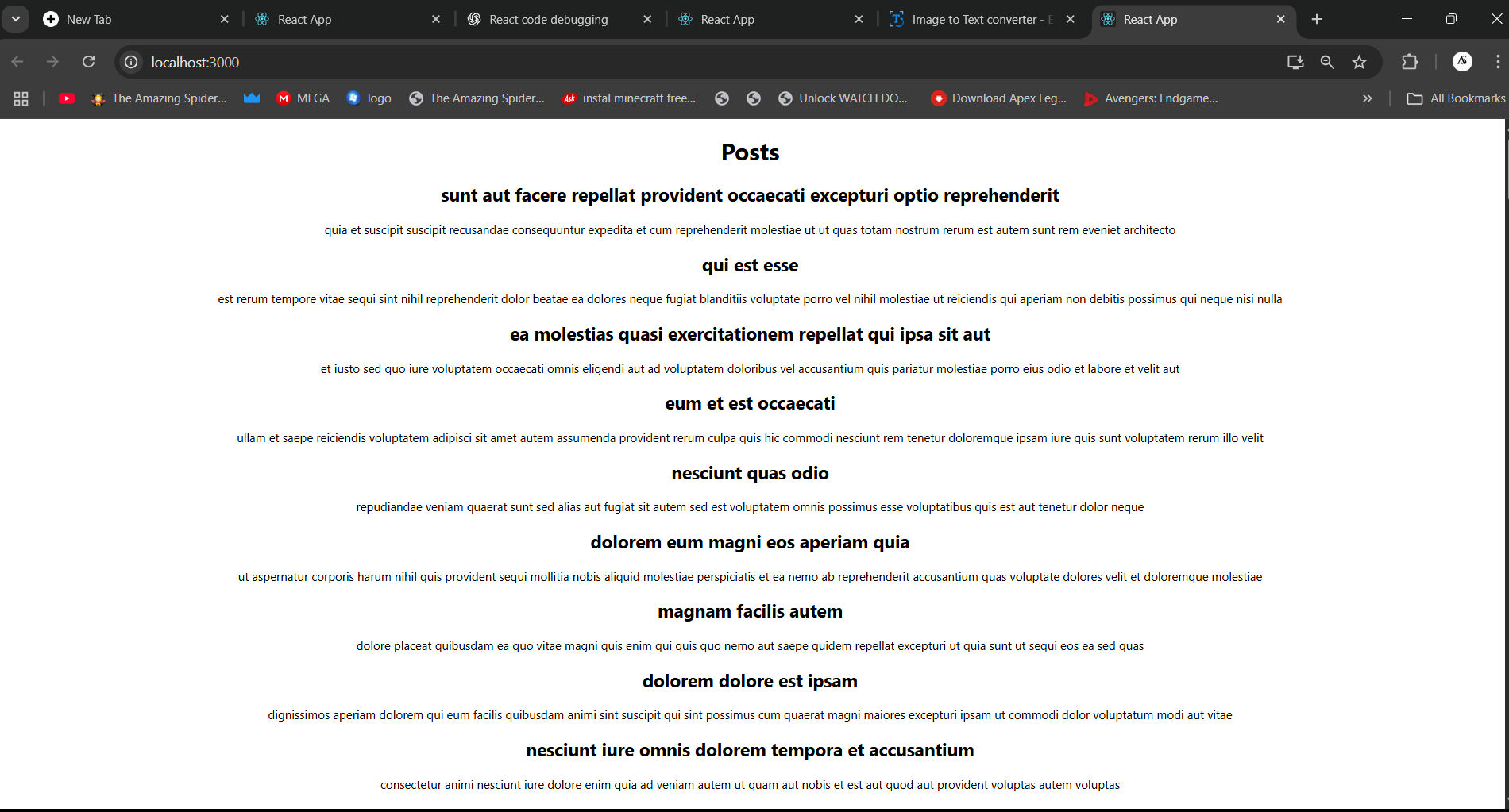
    </div>

  );

}

export default App;

**OUTPUT:**

****

# **HANDS ON 5: COHORT TRACKER**

**CODE:**

**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;

}

**Index.css:**

body {

  margin: 0;

  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

    sans-serif;

  -webkit-font-smoothing: antialiased;

  -moz-osx-font-smoothing: grayscale;

}

code {

  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

    monospace;

}

**App.js:**

import CohortDetails from './CohortDetails';

function App() {

  const cohorts = [

    {

      cohortCode: 'INTADMDF10',

      technology: '.NET FSD',

      startDate: '22-Feb-2022',

      currentStatus: 'Scheduled',

      coachName: 'Aathma',

      trainerName: 'Jojo Jose',

    },

    {

      cohortCode: 'ADM21JF014',

      technology: 'Java FSD',

      startDate: '10-Sep-2021',

      currentStatus: 'Ongoing',

      coachName: 'Apoorrv',

      trainerName: 'Elisa Smith',

    },

    {

      cohortCode: 'CDBJF21025',

      technology: 'Java FSD',

      startDate: '24-Dec-2021',

      currentStatus: 'Ongoing',

      coachName: 'Aathma',

      trainerName: 'John Doe',

    },

  ];

  return (

    <div>

      <h1>Cohorts Details</h1>

      {cohorts.map((cohort) => (

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

      ))}

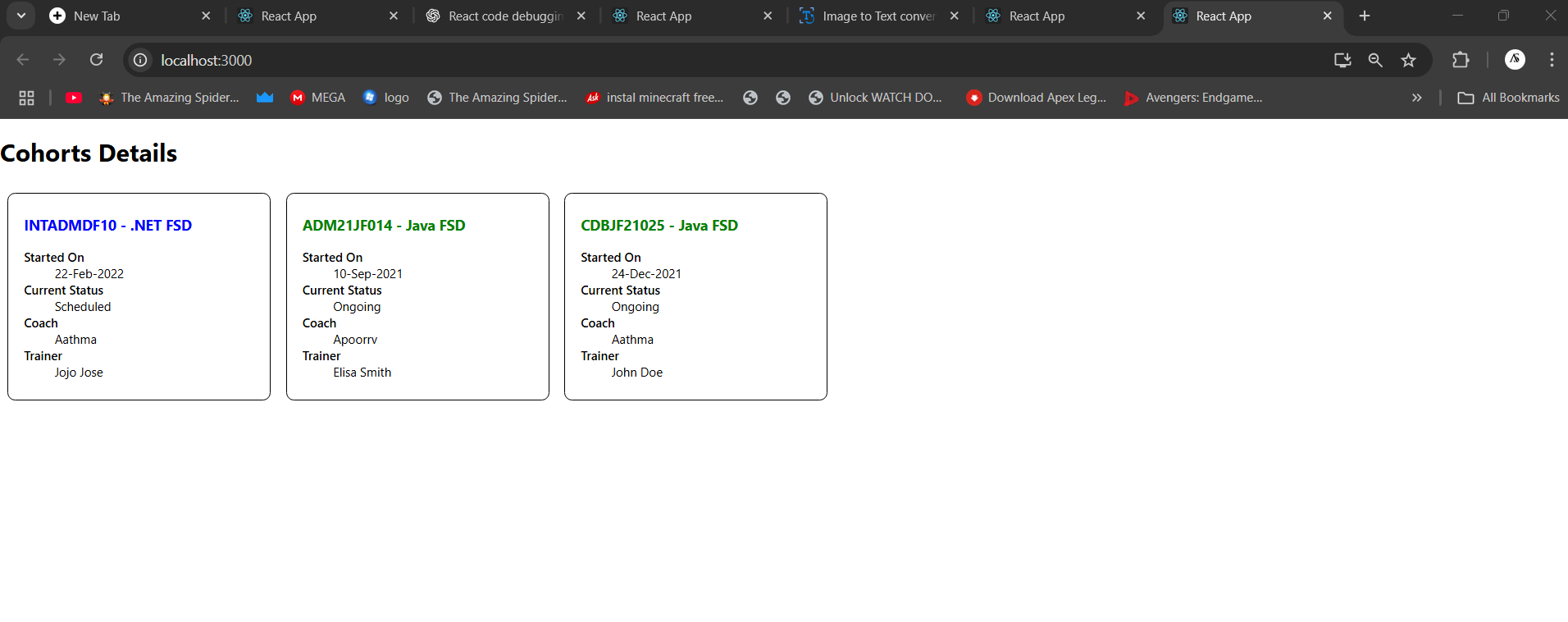
    </div>

  );

}

export default App;

**OUTPUT:**

****