React Hands-On Labs Report

Hands-on Lab 1: Introduction to React and SPA

## Objectives

* • Define SPA and its benefits
* • Define React
* • Explain the virtual DOM
* • List features of React

## Steps

1. Install Node.js and NPM from https://nodejs.org/en/download/

2. Install Create React App using: npm install -g create-react-app

3. Create new app: npx create-react-app myfirstreact

4. Navigate into the app: cd myfirstreact

5. Open the folder in Visual Studio Code

6. Open src/App.js and remove all existing code

7. Replace with the following code to show a heading

8. Run the app using: npm start

9. Visit http://localhost:3000 in your browser

## Code

import React from 'react';  
  
function App() {  
 return (  
 <div>  
 <h1>Welcome to the first session of React</h1>  
 </div>  
 );  
}  
  
export default App;

## Simulated Output (Python)

print("Welcome to the first session of React")

A screenshot of a computer

AI-generated content may be incorrect.

Hands-on Lab 2: React Components - Class and Function

## Objectives

* • Explain React components
* • Class vs Function components
* • Render multiple components

## Steps

1. Create new app: npx create-react-app StudentApp

2. Navigate to the src folder and create a Components folder

3. Inside Components, create Home.js, About.js, Contact.js

4. Add class components to each file with a unique welcome message

5. Edit App.js to import and render all three components

6. Run the app using: npm start

7. Open http://localhost:3000 in browser

## Code

// Home.js  
import React, { Component } from 'react';  
  
class Home extends Component {  
 render() {  
 return <h2>Welcome to the Home page of Student Management Portal</h2>;  
 }  
}  
  
export default Home;

// About.js  
import React, { Component } from 'react';  
  
class About extends Component {  
 render() {  
 return <h2>Welcome to the About page of the Student Management Portal</h2>;  
 }  
}  
  
export default About;

// Contact.js  
import React, { Component } from 'react';  
  
class Contact extends Component {  
 render() {  
 return <h2>Welcome to the Contact page of the Student Management Portal</h2>;  
 }  
}  
  
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>  
 <Home />  
 <About />  
 <Contact />  
 </div>  
 );  
}  
  
export default App;

## Simulated Output (Python)

print("Welcome to the Home page of Student Management Portal")  
print("Welcome to the About page of the Student Management Portal")  
print("Welcome to the Contact page of the Student Management Portal")

A screenshot of a computer

AI-generated content may be incorrect.

Hands-on Lab 3: Functional Component & Styling

## Objectives

* • Function component usage
* • Apply styling using CSS

## Steps

1. Create app: npx create-react-app scorecalculatorapp

2. Inside src, create Components and Stylesheets folders

3. Add CalculateScore.js component in Components

4. Add mystyle.css file in Stylesheets with styles

5. Update App.js to use the CalculateScore component

6. Run the app using: npm start

7. Open http://localhost:3000 in browser

## Code

// CalculateScore.js  
import React from 'react';  
import '../Stylesheets/mystyle.css';  
  
function CalculateScore({ name, school, total, goal }) {  
 const average = total / goal;  
 return (  
 <div className="score-box">  
 <h2>Student Score Report</h2>  
 <p>Name: {name}</p>  
 <p>School: {school}</p>  
 <p>Average Score: {average}</p>  
 </div>  
 );  
}  
  
export default CalculateScore;

/\* mystyle.css \*/  
.score-box {  
 border: 1px solid black;  
 padding: 20px;  
 width: 300px;  
 margin: 10px;  
 border-radius: 10px;  
}

// App.js  
import React from 'react';  
import CalculateScore from './Components/CalculateScore';  
  
function App() {  
 return (  
 <div>  
 <CalculateScore name="John Doe" school="Cognizant Academy" total={450} goal={5} />  
 </div>  
 );  
}  
  
export default App;

## Simulated Output (Python)

print("Student Score Report")  
print("Name: John Doe")  
print("School: Cognizant Academy")  
print("Average Score: 90.0")

A screenshot of a computer error

AI-generated content may be incorrect.

# Hands-on Lab 4: Component Lifecycle Methods

## Objectives

* • Use componentDidMount()
* • Use componentDidCatch()
* • Fetch and render posts

## Steps

1. Create app: npx create-react-app blogapp

2. Create Post.js component to display title and body

3. Create Posts.js class component to fetch and display posts using Fetch API

4. Initialize state and define loadPosts() method

5. Call loadPosts() from componentDidMount()

6. Handle any rendering errors using componentDidCatch()

7. Display fetched posts in render() method using Post component

8. Import and render Posts in App.js

9. Run the app using npm start

10. Open http://localhost:3000 in browser

## Code

// Post.js  
import React from 'react';  
  
function Post({ title, body }) {  
 return (  
 <div>  
 <h3>{title}</h3>  
 <p>{body}</p>  
 </div>  
 );  
}  
  
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 = async () => {  
 try {  
 const res = await fetch('https://jsonplaceholder.typicode.com/posts');  
 const data = await res.json();  
 this.setState({ posts: data.slice(0, 3) });  
 } catch (err) {  
 this.setState({ error: err.message });  
 }  
 };  
  
 componentDidMount() {  
 this.loadPosts();  
 }  
  
 componentDidCatch(error, info) {  
 alert("An error occurred: " + error);  
 }  
  
 render() {  
 return (  
 <div>  
 {this.state.posts.map(post => (  
 <Post key={post.id} title={post.title} body={post.body} />  
 ))}  
 </div>  
 );  
 }  
}  
  
export default Posts;

// App.js  
import React from 'react';  
import Posts from './Posts';  
  
function App() {  
 return (  
 <div>  
 <Posts />  
 </div>  
 );  
}  
  
export default App;

## Simulated Output (Python)

print("Title: sunt aut facere repellat...")  
print("Body: quia et suscipit...")  
print("Title: qui est esse")  
print("Body: est rerum tempore vitae...")

A screenshot of a computer

AI-generated content may be incorrect.

Hands-on Lab 5: CSS Modules and Inline Styling

## Objectives

* • Apply CSS Modules
* • Style components conditionally

## Steps

1. Unzip and open provided React app

2. Install node packages using npm install

3. Create CohortDetails.module.css in src folder

4. Define a 'box' class and <dt> tag styles

5. Edit CohortDetails.js to import and use the CSS module

6. Style h3 element inline based on cohort status

7. Import and render CohortDetails in App.js

8. Run the app using npm start

9. Visit http://localhost:3000 in browser

## 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;  
}

// CohortDetails.js  
import React from 'react';  
import styles from './CohortDetails.module.css';  
  
function CohortDetails({ name, status }) {  
 const statusColor = status === 'ongoing' ? 'green' : 'blue';  
  
 return (  
 <div className={styles.box}>  
 <h3 style={{ color: statusColor }}>{name}</h3>  
 <dl>  
 <dt>Status:</dt>  
 <dd>{status}</dd>  
 </dl>  
 </div>  
 );  
}  
  
export default CohortDetails;

// App.js  
import React from 'react';  
import CohortDetails from './CohortDetails';  
  
function App() {  
 return (  
 <div>  
 <CohortDetails name="React Bootcamp" status="ongoing" />  
 <CohortDetails name="Angular Training" status="completed" />  
 </div>  
 );  
}  
  
export default App;

## Simulated Output (Python)

print("React Bootcamp - Status: ongoing (green text)")  
print("Angular Training - Status: completed (blue text)")

A screenshot of a computer

AI-generated content may be incorrect.