Name:Duggirala Bhagya Vivekini

Superset ID: 5673888

Cognizant Digital-Nurture-4.0-JavaFSE- Handson WEEK-6

ReactJS Hands-on Lab: Create "myfirstreact" Application

Install create-react-app:

npm install -g create-react-app

Create a New React Application

npx create-react-app myfirstreact

Navigate into Your Application's Directory

bash

cd myfirstreact

Open the Project in Visual Studio Code

code .

Modify App.js File

import React from 'react';

import './App.css';

function App() {

return (

<div className="App">

<header className="App-header">

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

</header>

</div>

);

}

export default App;

Run the React Application

npm start

Output:

Welcome to the first session of React

ReactJS Hands-on Lab: Student Management Portal - Multiple Components

Create a New React Application

npx create-react-app StudentApp

Navigate into Your Application's Directory

cd StudentApp

Open the Project in Visual Studio Code

code .

Create the Home.js Component

import React from 'react';

function Home() {

return (

<div>

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

</div>

);

}

export default Home;

Create the About.js Component

import React from 'react';

function About() {

return (

<div>

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

</div>

);

}

export default About;

Create the Contact.js Component

import React from 'react';

function Contact() {

return (

<div>

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

</div>

);

}

export default Contact;

Run the React Application

npm start

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

ReactJS Hands-on Lab: Score Calculator Application with Functional Component and Styling

Create a New React Application

npx create-react-app scorecalculatorapp

Navigate into Your Application's Directory

cd scorecalculatorapp

Open the Project in Visual Studio Code

code .

Create the CalculateScore.js Functional Component

import React from 'react';

function CalculateScore(props) {

const { name, school, total, goal } = props;

const average = total / 5;

return (

<div className="score-card">

<h3>Student Score Card</h3>

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

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

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

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

<p><strong>Average Score:</strong> {average.toFixed(2)}</p> {/\* Display average with 2 decimal places \*/}

{average >= goal ? (

<p className="status-pass">Status: Passed (Goal Achieved!)</p>

) : (

<p className="status-fail">Status: Failed (Goal Not Achieved)</p>

)}

</div>

);

}

export default CalculateScore;

Create Stylesheets Folder and mystyle.css

/\* src/Stylesheets/mystyle.css \*/

.App {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 20px;

background-color: #f4f4f4;

padding: 20px;

border-radius: 8px;

}

.score-card {

border: 1px solid #ddd;

padding: 20px;

margin: 20px auto;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

background-color: #fff;

width: 80%;

max-width: 500px;

}

.score-card h3 {

color: #333;

border-bottom: 2px solid #007bff;

padding-bottom: 10px;

margin-bottom: 15px;

}

.score-card p {

font-size: 1.1em;

line-height: 1.5;

margin-bottom: 8px;

}

.score-card strong {

color: #007bff;

}

.status-pass {

color: green;

font-weight: bold;

font-size: 1.2em;

}

.status-fail {

color: red;

font-weight: bold;

font-size: 1.2em;

}

Modify App.js to Invoke CalculateScore and Apply Styles

import React from 'react';

import './App.css';

import './Stylesheets/mystyle.css';

import CalculateScore from './Components/CalculateScore'; // Import CalculateScore component

function App() {

return (

<div className="App">

<h1>Student Score Calculator</h1>

<CalculateScore

name="Alice Johnson"

school="Springfield High"

total={450} // Example total marks

goal={75} // Example goal average score

/>

{/\* You can add more CalculateScore components with different props here \*/}

<CalculateScore

name="Bob Williams"

school="Riverside Academy"

total={320}

goal={70}

/>

</div>

);

}

export default App;

Run the React Application

npm start

Output:

Student Score Calculator

[Score Card for Alice Johnson]

Student Score Card Name: Alice Johnson

School: Springfield High

Total Marks: 450

Goal: 75

Average Score: 90.00

Status: Passed (Goal Achieved!)

[Score Card for Bob Williams]

Student Score Card Name: Bob Williams

School: Riverside Academy

Total Marks: 320

Goal: 70

Average Score: 64.00

Status: Failed (Goal Not Achieved)

**ReactJS Hands-on Lab: Blog Application with Lifecycle Hooks and Fetch API**

**Create a New React Application**

npx create-react-app blogapp

**Navigate into Your Application's Directory**

cd blogapp

**Open the Project in Visual Studio Code**

code .

**Create Post.js (Data Structure)**

class Post {

constructor(id, title, body) {

this.id = id;

this.title = title;

this.body = body;

}

}

export default Post;

**Create the Posts.js Class Component**

import React from 'react';

import Post from './Post';

class Posts extends React.Component {

constructor(props) {

super(props);

this.state = {

posts: [],

error: null,

};

}

loadPosts = async () => {

try {

const response = await fetch('https://jsonplaceholder.typicode.com/posts');

if (!response.ok) {

throw new Error(`HTTP error! status: ${response.status}`);

}

const data = await response.json();

const loadedPosts = data.map(item => new Post(item.id, item.title, item.body));

this.setState({ posts: loadedPosts, error: null });

} catch (error) {

console.error("Error fetching posts:", error);

this.setState({ error: "Failed to load posts. Please try again later." });

}

};

componentDidMount() {

this.loadPosts();

}

componentDidCatch(error, errorInfo) {

console.error("Component Did Catch an Error:", error, errorInfo);

alert("An error occurred in a component: " + error.message);

this.setState({ error: "An unexpected error occurred." });

}

render() {

const { posts, error } = this.state;

if (error) {

return <div style={{ color: 'red', textAlign: 'center' }}>Error: {error}</div>;

}

if (posts.length === 0) {

return <div style={{ textAlign: 'center' }}>Loading posts...</div>;

}

return (

<div style={{ padding: '20px', maxWidth: '800px', margin: 'auto' }}>

<h2 style={{ textAlign: 'center', color: '#333' }}>Blog Posts</h2>

{posts.map(post => (

<div key={post.id} style={{ border: '1px solid #eee', borderRadius: '8px', padding: '15px', marginBottom: '15px', backgroundColor: '#f9f9f9', boxShadow: '0 2px 5px rgba(0,0,0,0.1)' }}>

<h3 style={{ color: '#007bff', marginBottom: '5px' }}>{post.title}</h3>

<p style={{ color: '#555', lineHeight: '1.6' }}>{post.body}</p>

</div>

))}

</div>

);

}

}

export default Posts;

**Modify App.js to Render the Posts Component**

import React from 'react';

import './App.css';

import Posts from './Posts';

function App() {

return (

<div className="App">

{/\* You can add a main heading for your blog application here \*/}

<Posts /> {/\* Render the Posts component \*/}

</div>

);

}

export default App;

**Run the React Application**

npm start

**Output**

Blog Posts

[Post 1]

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

[Post 2]

qui est esse

est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores

neque et voluptates illo harum dolor eum et non reiciendis sed qui

ReactJS Hands-on Lab: Styling Components with CSS Modules and Conditional Styling

Initial Project Setup

cd CohortDashboard

Restore the Node packages:

npm install

Open the application using VS Code:

code .

Create a New CSS Module:

/\* src/CohortDetails.module.css \*/

.box {

width: 300px;

display: inline-block;

margin: 10px; /\* Overall 10px margin \*/

padding-top: 10px;

padding-bottom: 10px;

padding-left: 20px;

padding-right: 20px;

border: 1px solid black;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

vertical-align: top;

}

dt {

font-weight: 500;

}

Modify the Cohort Details Component

import React from 'react';

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

function CohortDetails({ cohort }) {

const titleStyle = {

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

};

return (

<div className={styles.box}>

<h3 style={titleStyle}>{cohort.name}</h3> {/\* Apply inline conditional style \*/}

<dl> {/\* Using <dl> for definition list \*/}

<dt>Status:</dt>

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

<dt>Start Date:</dt>

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

<dt>End Date:</dt>

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

<dt>Mentor:</dt>

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

</dl>

</div>

);

}

export default CohortDetails;

Key Changes to make in your CohortDetails.js:

const exampleCohort1 = {

name: "Java FSE Batch 1",

status: "ongoing",

startDate: "2024-01-15",

endDate: "2024-06-15",

mentor: "John Doe"

};

const exampleCohort2 = {

name: "Python DS Batch 3",

status: "completed",

startDate: "2023-09-01",

endDate: "2024-03-01",

mentor: "Jane Smith"

};

Ensure App.js Renders CohortDetails Components

import React from 'react';

import './App.css';

import CohortDetails from './components/CohortDetails';

function App() {

const cohort1 = {

name: "Digital Nurture Java FSE (Ongoing)",

status: "ongoing",

startDate: "2024-01-15",

endDate: "2024-07-15",

mentor: "Alice Mentor"

};

const cohort2 = {

name: "Digital Nurture DotNet FSE (Completed)",

status: "completed",

startDate: "2023-08-01",

endDate: "2024-01-30",

mentor: "Bob Mentor"

};

return (

<div className="App">

<h1>Academy Dashboard</h1>

<div style={{ display: 'flex', flexWrap: 'wrap', justifyContent: 'center' }}>

<CohortDetails cohort={cohort1} />

<CohortDetails cohort={cohort2} />

</div>

</div>

);

}

export default App;

Run the React Application

npm start

Output:

