**Digital Nurture 4.0 – Week 6**

**Mandatory hands-on**

**File Name:1.ReactJS – HOL**

**Objective:**

The objective of this lab is to define Single-Page Application (SPA) and its benefits, understand what React is and how it works, differentiate between SPA and Multi-Page Application (MPA), explain the pros and cons of Single-Page Applications, describe the concept of virtual DOM, and highlight the key features of React.

**Steps:**

**Step 1:**

To create a new React app, Install Nodejs and Npm from the following link:  
<https://nodejs.org/en/download/>

**Step 2:**

Install Create-react-app by running the following command in the command prompt:

npm install -g create-react-app

**Step 3:**

To create a React Application with the name of “myfirstreact”, type the following command:

npx create-react-app myfirstreac

**Step 4:**

Once the App is created, navigate into the folder of myfirstreact by typing the following command:

cd myfirstreact

**Step 5:**

Open the folder of myfirstreact in Visual Studio Code

**Step 6:**

Open the App.js file in Src Folder of myfirstreact

**Step 7:**

Remove the current content of “App.js”

**Step 8:**

import React from 'react';

function App() {

return (

<div>

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

</div>

);

}

export default App;

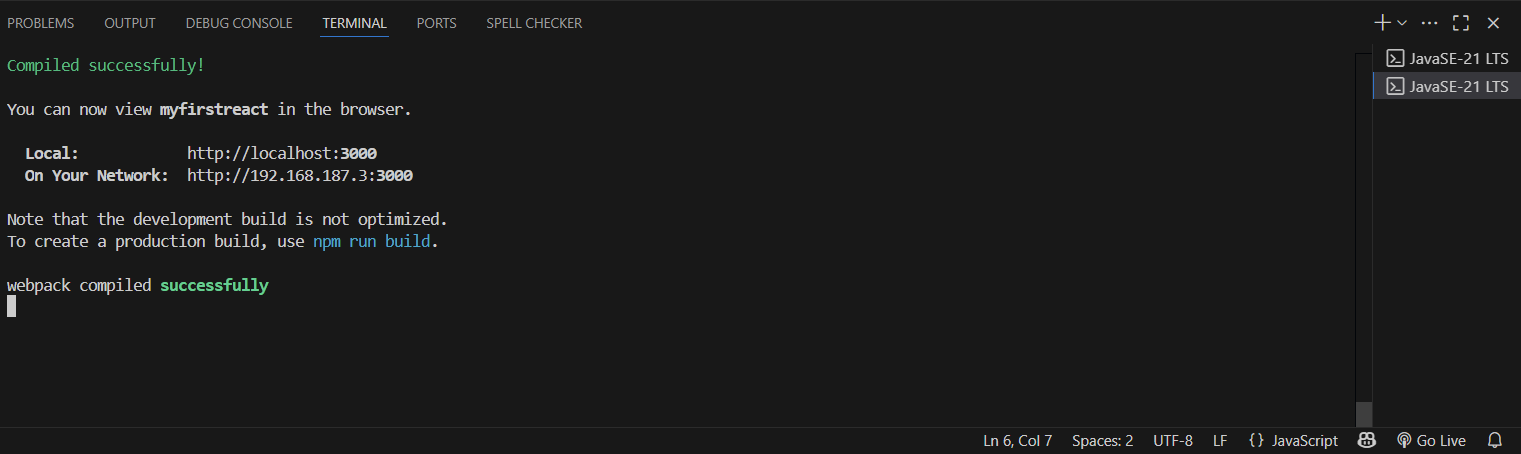
**Step 9:**

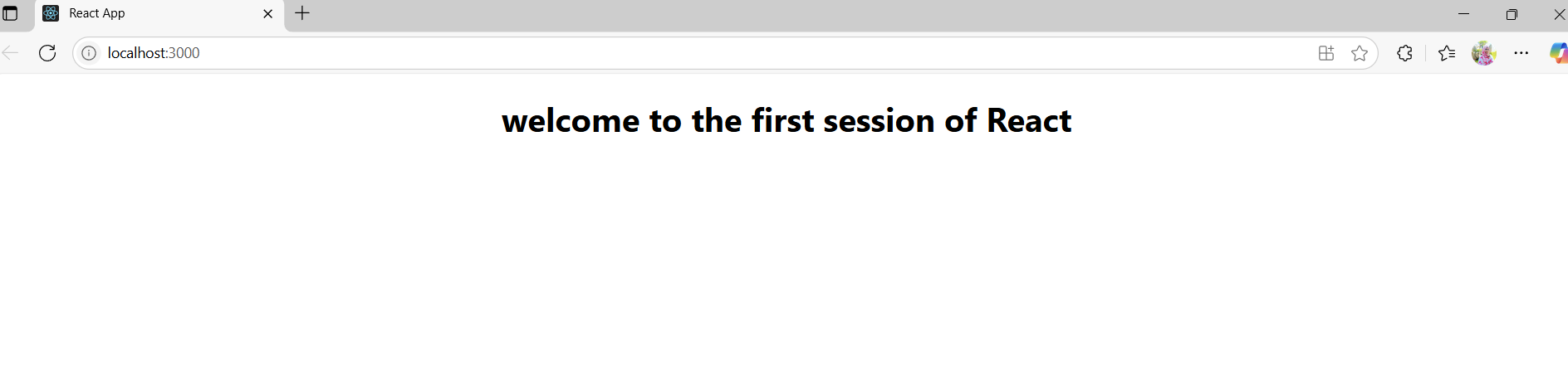
npm start

**Step 10:**

<http://localhost:3000>

**Output:**

****

****

**File Name:2.ReactJS – HOL**

**Objective:**

To learn how to create and render class components in React by building a simple Student Management Portal with Home, About, and Contact components.

**Steps:**

**Step 1: Create a React project named studentapp**

Open terminal:

npx create-react-app StudentApp

**Step 2: Create a folder named Components**

In the StudentApp/src folder:

* Create a folder named Components
* Inside Components, add a file named Home.js

**Step 3: Add the following code to 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;

**Step 4:** Add another file named About.js inside the src/Components folder

**Step 5: Repeat for About.js and Contact.js**

**About.js**

import React, { Component } from 'react';

class About extends Component {

render() {

return (

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

);

}

}

export default Contact;

**Step 6: Edit App.js to invoke all three components**

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;

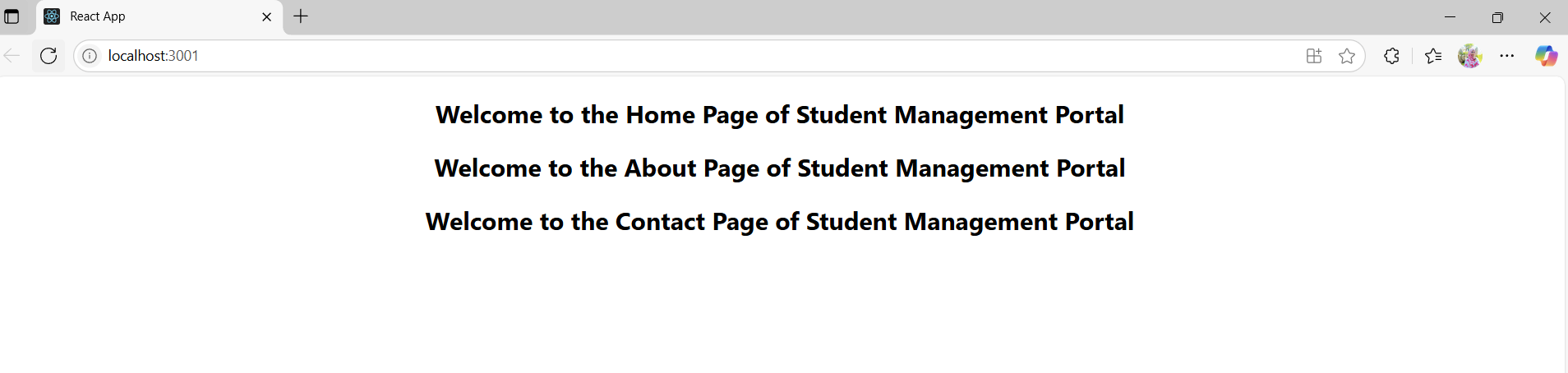
**Step 7: Run the React app**

npm start

**Step 8:**

<http://localhost:3000>

**Output:**

****

**File Name:3.ReactJS – HOL**

**Objective:**

This lab aims to help students understand React components by exploring their differences from JavaScript functions, types of components, and core concepts such as the constructor and render() function. The lab provides hands-on experience in creating function components, styling them using external CSS, and rendering components effectively.

**Steps:**

**Step 1: Go to CalculateScore.js inside src/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>

);

**Step 2: Go to mystyle.css inside src/Components**

**mystyle.css**

.Name {

font-weight: 300;

color: blue;

}

.School {

color: crimson;

}

.Total {

color: darkmagenta;

}

.formatstyle {

text-align: center;

font-size: large;

}

.Score {

color: forestgreen;

}

**Step 3: Go to App.js inside src/**

**App.js**

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

function App() {

return (

<div>

<CalculateScore

Name={"Steeve"}

School={"DNV Public School"}

total={284}

goal={3}

/>

</div>

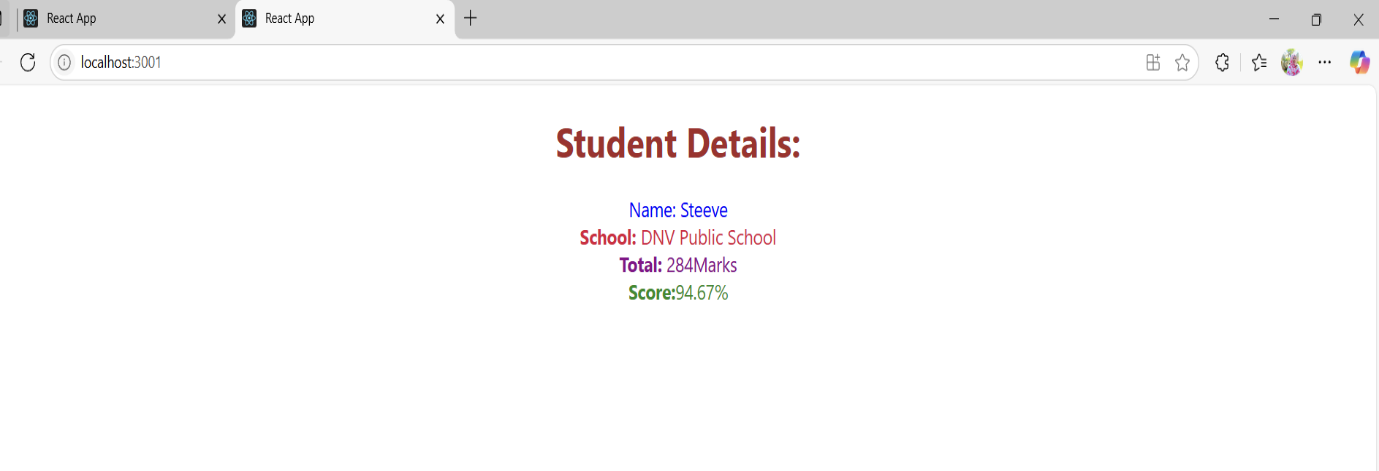
);}

export default App;

**Step 4:** **Run Your App**

npm start

**Output:**

****

**File Name:4.ReactJS – HOL**

**Objective :**

The objective of this lab is to understand and implement React class component lifecycle methods by using componentDidMount() to fetch data from an external API (https://jsonplaceholder.typicode.com/posts) and componentDidCatch() to handle any rendering errors, while dynamically displaying a list of blog posts using class-based components.

**Steps:**

**Step 1:** Create a React App named blogapp using create-react-app and open it in Visual Studio Code.

**Step 2: Inside the src/ folder, create a file named Post.js and define a Post class:**

class Post {

constructor(id, title, body) {

this.id = id;

this.title = title;

this.body = body;

}

}

export default Post;

**Step 3: Create another file named Posts.js and define a class-based React component:**

import React from 'react';

import Post from './Post';

class Posts extends React.Component {

constructor(props) {

super(props);

this.state = {

posts: []

};

}

**Step 4: Inside Posts.js, define the loadPosts() method to fetch data from the API:**

loadPosts() {

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

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

.then(data => {

const postList = data.map(p => new Post(p.id, p.title, p.body));

this.setState({ posts: postList });

})

.catch(error => {

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

});

}

**Step 5: Call loadPosts() in the componentDidMount() lifecycle method:**

componentDidMount() {

this.loadPosts();

}

**Step 6: Add componentDidCatch() to catch and display any rendering errors:**

componentDidCatch(error, info) {

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

}

**Step 7: Use the render() method to display the post titles and bodies:**

js

Copy code

render() {

return (

<div>

<h1>All Posts</h1>

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

<div key={post.id}>

<h2>{post.title}</h2>

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

</div>

))}

</div>

);

}

}

export default Posts;

**Step 8: Open App.js and import the Posts component:**

import React from 'react';

import './App.css';

import Posts from './Posts';

function App() {

return (

<div className="App">

<Posts />

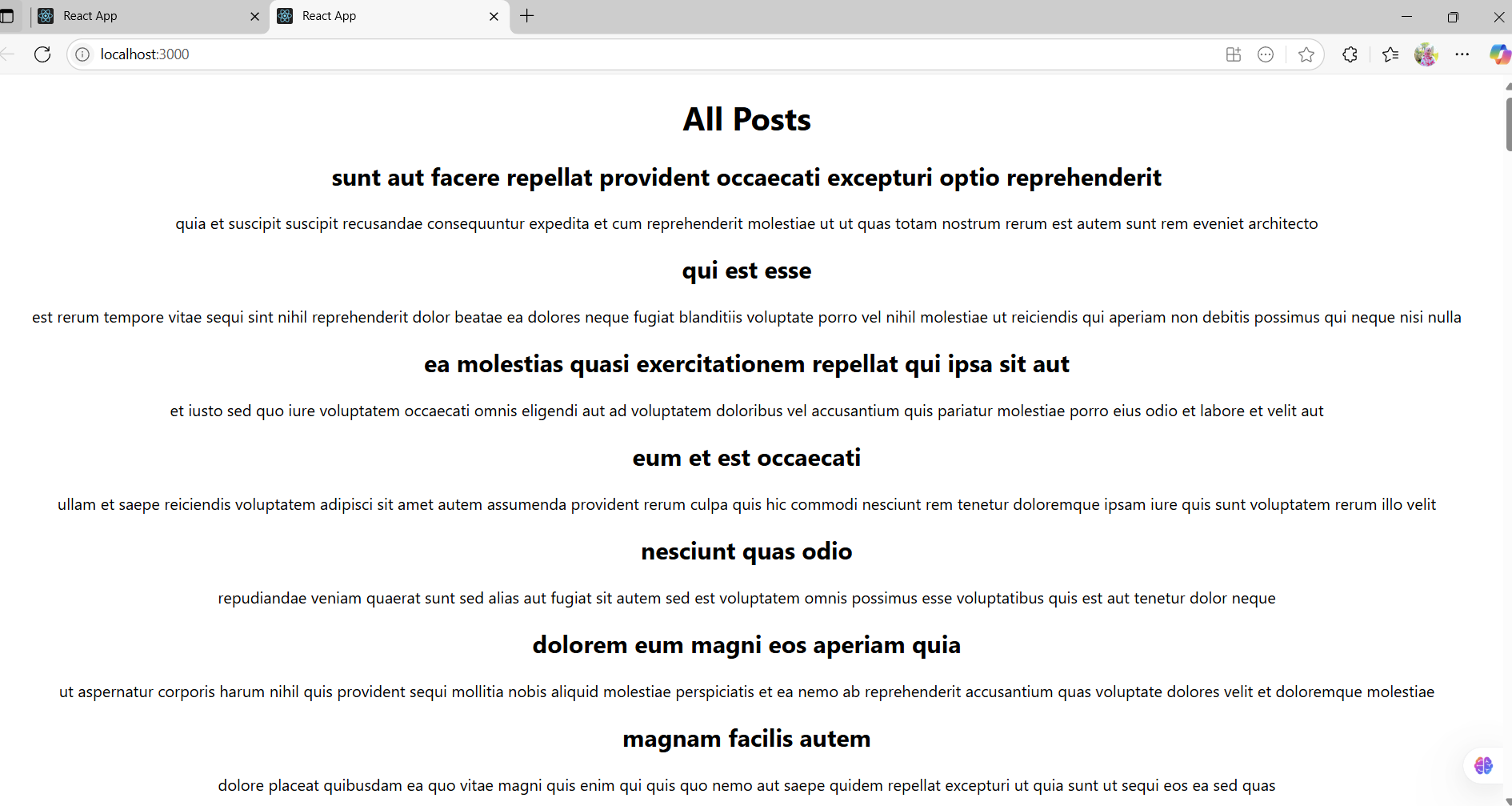
</div>

);

}

export default App;

**Output:**

****

**File Name:5.ReactJS – HOL**

**Objective:**

The objective of this hands-on lab is to learn how to style React components using CSS Modules and inline styles. You will create a reusable component to display cohort details, apply custom styling via a CSS module file, and dynamically change the heading color based on the cohort status. This activity helps reinforce best practices for component-based styling and conditional rendering in React.

**Steps:**

**Step 1: Open the Project in Visual Studio Code**

1. Locate your unzipped React app folder (e.g., cohorttracker).
2. Open **Visual Studio Code**.
3. Go to **File → Open Folder**, and select the React app folder.
4. Once opened, in the **Explorer panel**, expand the src folder and find the components folder.

**Step 2: Create the CSS Module**

CohortDetails.module.css

.box {

width: 300px;

display: inline-block;

margin: 10px;

padding: 10px 20px;

border: 1px solid black;

border-radius: 10px;

vertical-align: top;

}

dt {

font-weight: 500;

}

**Step 3: Create the React Component File**

CohortDetails.jsx

import React from 'react';

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

const CohortDetails = ({ title, status, startDate, coach, trainer }) => {

const titleColor = status === 'Ongoing' ? 'green' : 'blue';

return (

<div className={styles.box}>

<h3 style={{ color: titleColor }}>{title}</h3>

<dl>

<dt>Started On</dt>

<dd>{startDate}</dd>

<dt>Current Status</dt>

<dd>{status}</dd>

<dt>Coach</dt>

<dd>{coach}</dd>

<dt>Trainer</dt>

<dd>{trainer}</dd>

</dl>

</div>

);

};

export default CohortDetails;

**Step 4: Use the Component in App.js**

import React from 'react';

import CohortDetails from './components/CohortDetails.jsx';

function App() {

return (

<div>

<h1>Cohorts Dashboard</h1>

<CohortDetails

title="INTADMDF10 - .NET FSD"

status="Scheduled"

startDate="22-Feb-2022"

coach="Aathma"

trainer="Jojo Jose"

/>

<CohortDetails

title="ADM21JF014 - Java FSD"

status="Ongoing"

startDate="10-Sep-2021"

coach="Apoorv"

trainer="Elisa Smith"

/>

<CohortDetails

title="CDBJF21025 - Java FSD"

status="Ongoing"

startDate="24-Dec-2021"

coach="Aathma"

trainer="John Doe"

/>

</div>

);

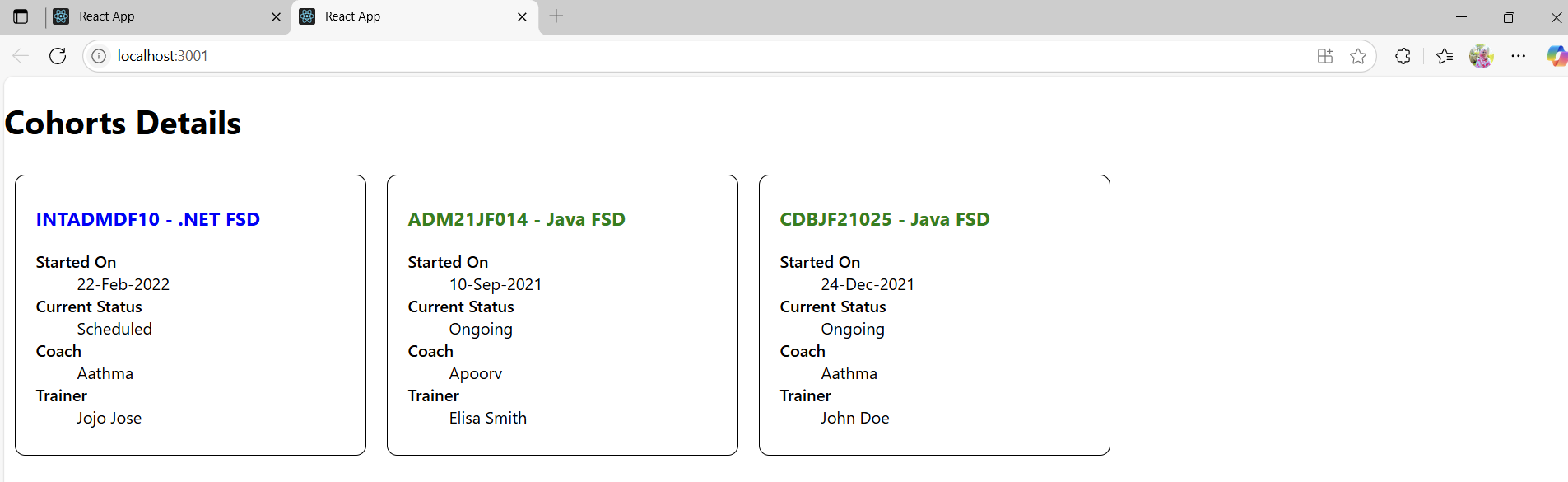
}

export default App;

**Step 5:Run the Code:**

npm start

**Output:**

****

**Additional Hands-On**

**File Name:6.ReactJS – HOL**

**Objective:**

The objective of the Trainer component (defined in trainer.js) is to serve as a model class that structures and encapsulates trainer-related data used across the application. It defines key properties such as TrainerId, Name, Email, Phone, Technology, and Skills, providing a consistent blueprint for creating and managing trainer objects. This ensures standardized data handling when rendering trainer lists, passing props, or displaying detailed trainer information within the React components.

**Steps:**

**Step 1: Create the React App**

Open terminal and run:

npx create-react-app TrainersApp

**Step 2:** **Create trainer.js inside src/**

class Trainer {

constructor(trainerId, name, email, phone, technology, skills) {

this.trainerId = trainerId;

this.name = name;

this.email = email;

this.phone = phone;

this.technology = technology;

this.skills = skills;

}

}

export default Trainer;

**Step 3:** **Create TrainersMock.js with mock data**

import Trainer from "./trainer";

const trainersMock = [

new Trainer(

"t-syed8",

"Syed Khaleelullah",

"khaleelullah@cognizant.com",

"97676516962",

".NET",

["C#", "SQL Server", "React", ".NET Core"]

),

new Trainer(

"t-jojo",

"Jojo Jose",

"jojo@cognizant.com",

"9897199231",

"Java",

["Java", "JSP", "Angular", "Spring"]

),

new Trainer(

"t-elisa",

"Elisa Jones",

"elisa@cognizant.com",

"9871212235",

"Python",

["Python", "Django", "Angular"]

)

];

export default trainersMock;

**Step 4:TrainerList.js**

import React from "react";

import { Link } from "react-router-dom";

function TrainerList({ trainers }) {

return (

<div>

<h2>Trainers List</h2>

<ul>

{trainers.map((t) => (

<li key={t.trainerId}>

<Link to={`/trainer/${t.trainerId}`}>{t.name}</Link>

</li>

))}

</ul>

</div>

);

}

export default TrainerList;

**Step 5:** **Home.js**

import React from "react";

function Home() {

return (

<div>

<h2>Welcome to My Academy trainers page</h2>

</div>

);

}

export default Home;

**Step 6:** **TrainerDetails.js**

import React from "react";

import { useParams } from "react-router-dom";

import trainersMock from "./TrainersMock";

function TrainerDetails() {

const { id } = useParams();

const trainer = trainersMock.find((t) => t.trainerId === id);

if (!trainer) return <p>Trainer not found</p>;

return (

<div>

<h2>Trainers Details</h2>

<p><strong>{trainer.name}</strong> ({trainer.technology})</p>

<p>{trainer.email}</p>

<p>{trainer.phone}</p>

<ul>

{trainer.skills.map((s, i) => (

<li key={i}>{s}</li>

))}

</ul>

</div>

);

}

export default TrainerDetails;

**Step 7:** **App.js**

import React from "react";

import { BrowserRouter as Router, Routes, Route, Link } from "react-router-dom";

import Home from "./Home";

import TrainerList from "./TrainerList";

import TrainerDetails from "./TrainerDetails";

import trainersMock from "./TrainersMock";

function App() {

return (

<Router>

<div>

<h1>My Academy Trainers App</h1>

<nav>

<Link to="/">Home</Link> | <Link to="/trainers">Show Trainers</Link>

</nav>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/trainers" element={<TrainerList trainers={trainersMock} />} />

<Route path="/trainer/:id" element={<TrainerDetails />} />

</Routes>

</div>

</Router>

);

}

export default App;

**Step 8:**

import React from "react";

import ReactDOM from "react-dom/client";

import App from "./App";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(<App />);

**Step 9:Install React Router**

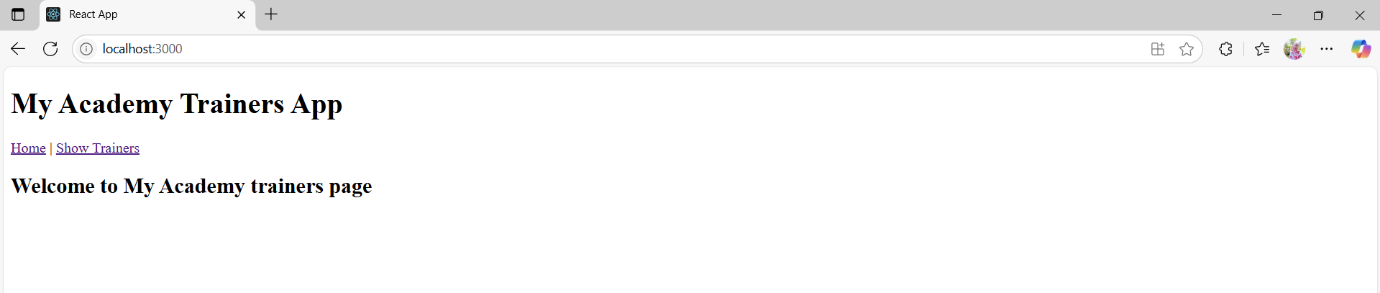
In your terminal, run:

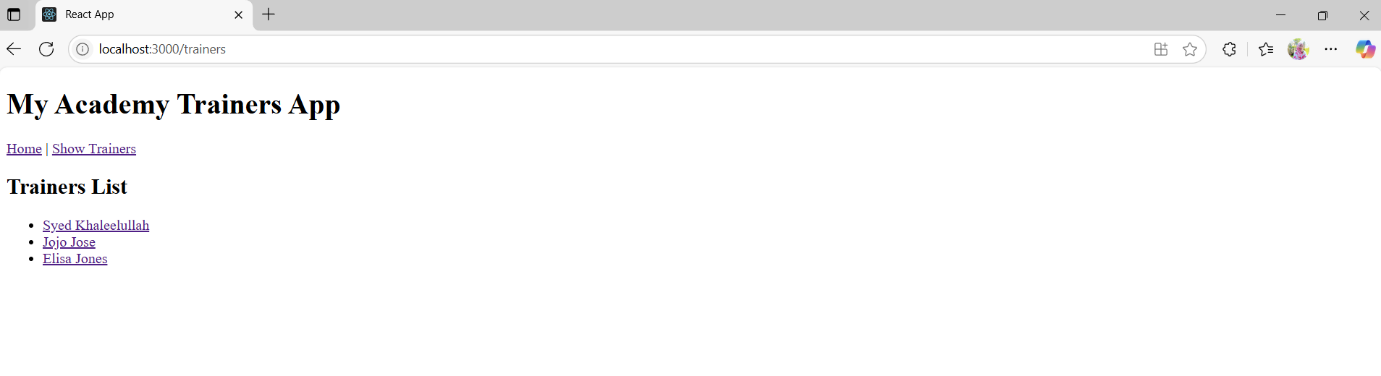
npm install react-router-dom@6

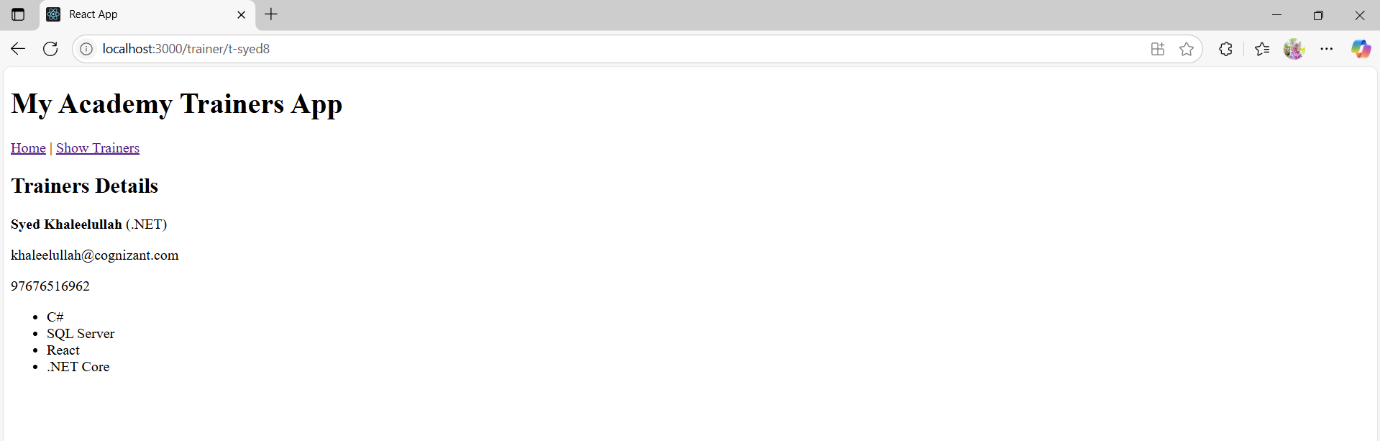
**Step 10: Run the App**

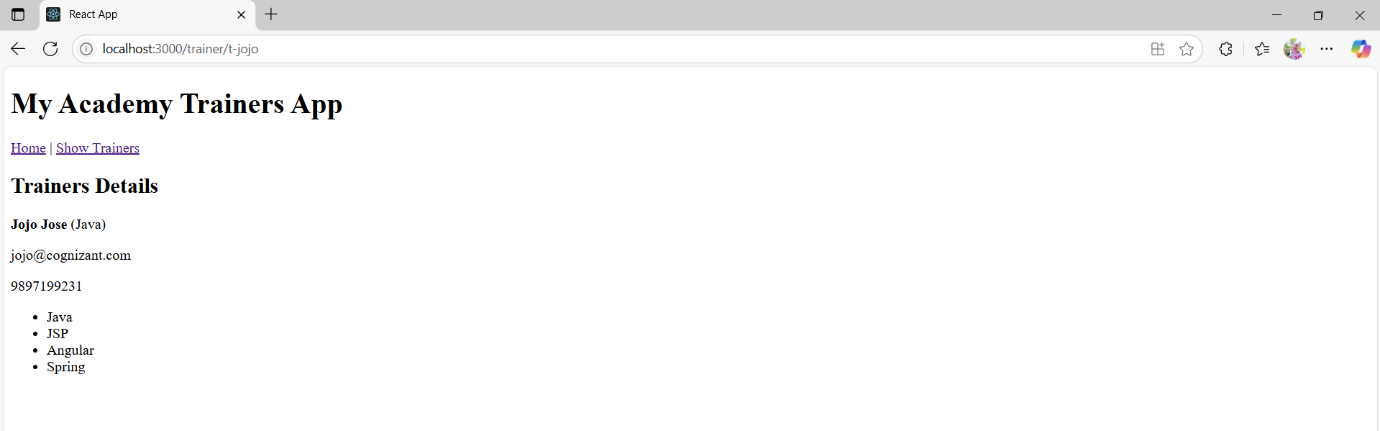
npm start

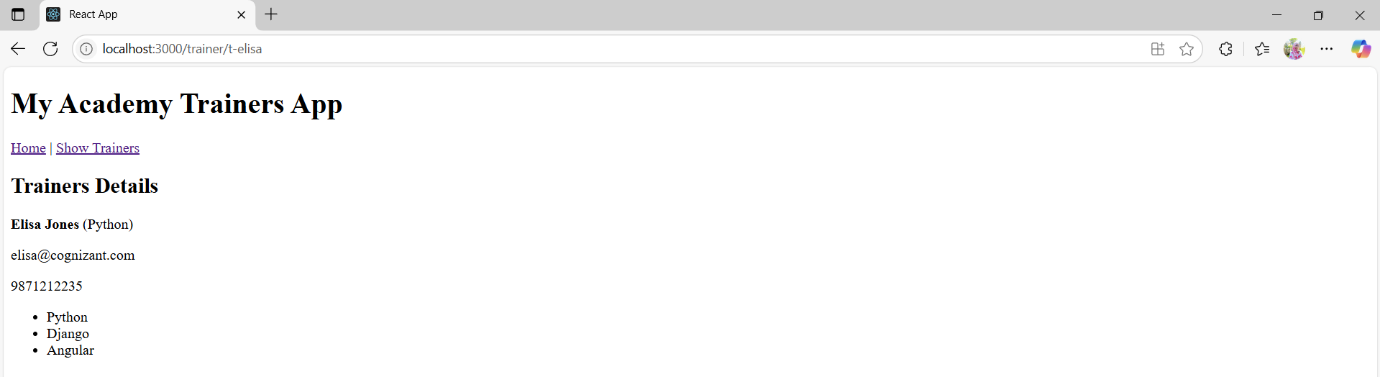
**Output:**

****









**File Name:7.ReactJS – HOL**

**Objective:**

The objective of this project is to create a visually structured and centered shopping cart table using React and CSS, where each item and its price are displayed in individual grid-like cells with clearly separated borders, matching a hand-drawn layout. The table includes two columns—Name and Price—with all text styled in green, and the entire layout enclosed in a single dark border to ensure a clean and organized presentation that mirrors the provided sketch.

**Steps:**

**Step 1: Set Up React App**

To begin building the shopping table layout, you need to set up a React development environment.

npx create-react-app shoppingapp

**Step 2:Create OnlineShopping.js**

import React from "react";

import Cart from "./Cart";

import "./App.css";

const OnlineShopping = () => {

const CartInfo = [

{ itemname: "Laptop", price: 80000 },

{ itemname: "TV", price: 120000 },

{ itemname: "Washing Machine", price: 50000 },

{ itemname: "Mobile", price: 30000 },

{ itemname: "Fridge", price: 70000 }

];

return (

<div className="mydiv">

<h1>Items Ordered :</h1>

<Cart item={CartInfo} />

</div>

);

};

export default OnlineShopping;

**Step 3: Create Cart.js**

import React from "react";

const Cart = ({ item }) => {

return (

<div className="table-container">

<table className="custom-table">

<thead>

<tr>

<th>Name</th>

<th>Price</th>

</tr>

</thead>

<tbody>

{item.map((item, index) => (

<tr key={index}>

<td>{item.itemname}</td>

<td>{item.price}</td>

</tr>

))}

</tbody>

</table>

</div>

);

};

export default Cart;

**Step 4:Create App.css**

.container {

display: flex;

flex-direction: column;

align-items: center;

margin-top: 50px;

}

h1 {

color: #2e7d32;

font-size: 32px;

font-weight: bold;

}

.cart-table {

border: 2px solid black;

border-collapse: separate;

border-spacing: 10px;

background-color: white;

}

.cart-table th,

.cart-table td {

padding: 0;

}

.cell {

border: 2px solid black;

padding: 10px 20px;

text-align: center;

font-size: 18px;

font-family: sans-serif;

background-color: white;

}

**Step 5: Create App.js**

import React from "react";

import OnlineShopping from "./OnlineShopping";

function App() {

return (

<div>

<OnlineShopping />

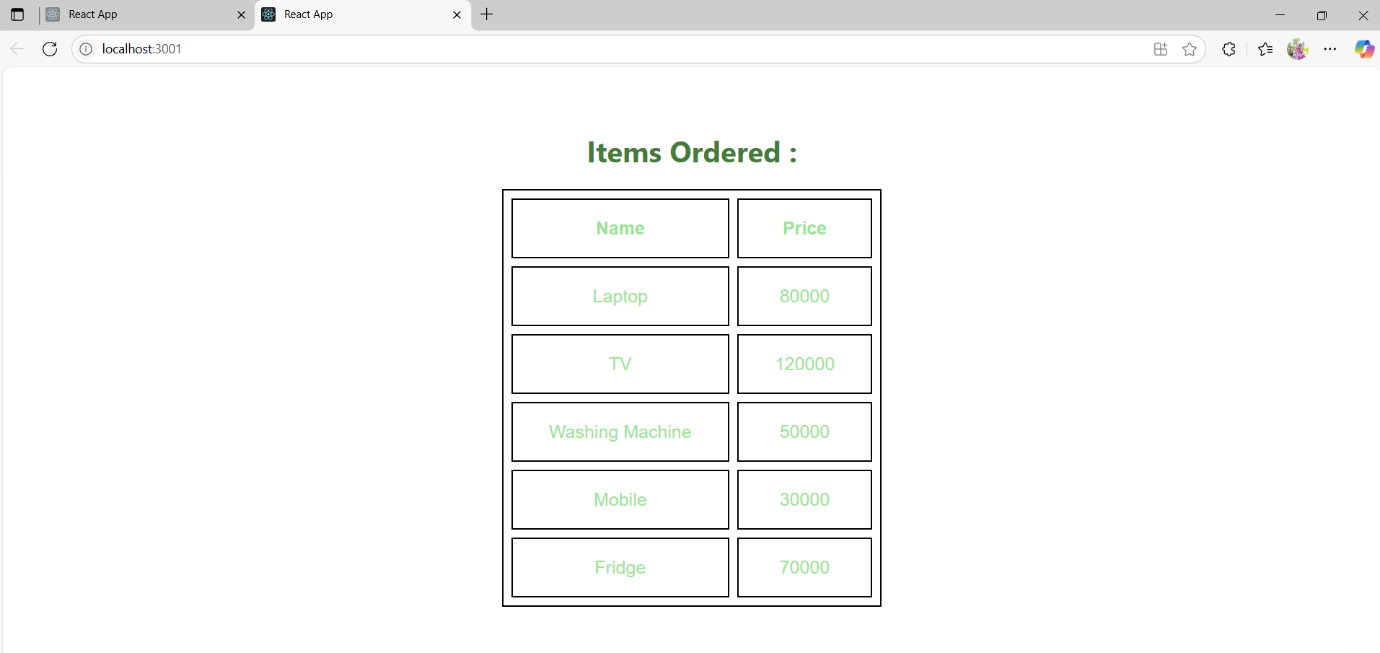
</div>

);

}

export default App;

**Output:**

****

**File Name:8.ReactJS – HOL**

**Objective:**

To develop a simple React application that uses state management in class-based components to count the number of people entering and exiting a facility (such as a mall). The project focuses on understanding the basics of React state and event handling using class components.

**Step 1: Install Prerequisites**

Make sure you have **Node.js**, **NPM**, and **Visual Studio Code** installed on your machine.

**Step 2: Create a New React App**

**npx create-react-app counterapp**

**cd counterapp**

**Step 3: Create the CountPeople Component**

In the src folder, create a new file named CountPeople.js.

**Step 4:** **Build the Component with Constructor and State**

**CountPeople.js:**

import React from 'react';

class CountPeople extends React.Component {

constructor(props) {

super(props);

this.state = {

entryCount: 0,

exitCount: 0

};

}

updateEntry = () => {

this.setState({ entryCount: this.state.entryCount + 1 });

};

updateExit = () => {

this.setState({ exitCount: this.state.exitCount + 1 });

};

render() {

return (

<div style={{ textAlign: 'center', marginTop: '50px' }}>

<h1>People Counter</h1>

<p>Number of people entered: {this.state.entryCount}</p>

<p>Number of people exited: {this.state.exitCount}</p>

<button onClick={this.updateEntry}>Login</button>

<button onClick={this.updateExit} style={{ marginLeft: '10px' }}>Exit</button>

</div>

);

}

}

export default CountPeople;

**Step 5:** **Use the Component in App.js**

import React from 'react';

import CountPeople from './CountPeople';

function App() {

return (

<div className="App">

<CountPeople />

</div>

);

}

export default App;

**Step 6: Start the App**

npm start

**Output:**

