**WEEK-6**

**1. ReactJS-HOL**

Prerequisites

Before you begin, make sure you have:

Node.js & npm installed

Visual Studio Code or any code editor

A modern web browser (Chrome, Firefox, Edge)

To verify Node.js and npm are installed:

node -v

npm -v

Getting Started

Follow these steps to create and run your first React application:

1. Install Create React App (CRA) globally

npm install -g create-react-app

2. Create a new React App

npx create-react-app myfirstreact

This creates a folder called myfirstreact with the React boilerplate.

3. Navigate into the project folder

cd myfirstreact

4. Modify App.js

import React from 'react';

function App() {

return (

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

);

}

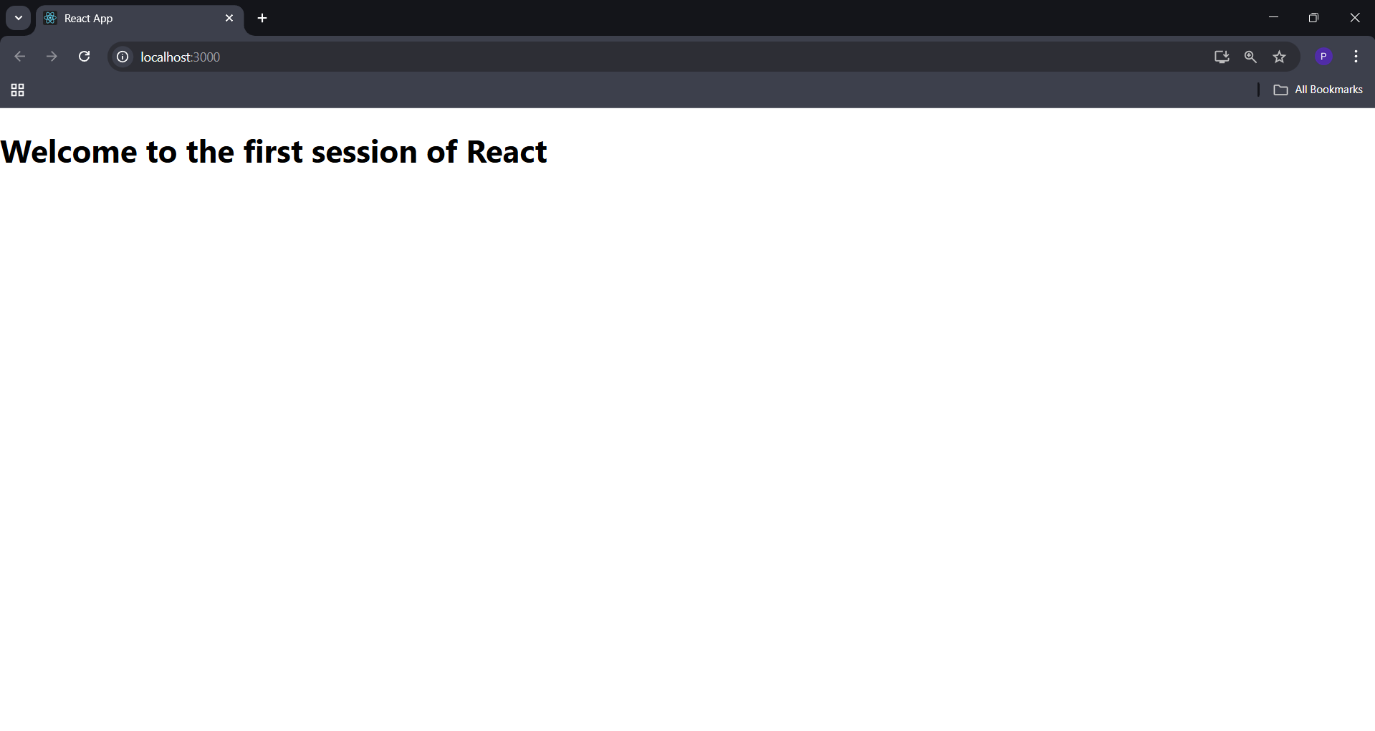
export default App;

Running the Application

Start the development server with:

npm start

**OUTPUT:**



**2. ReactJS-HOL**

**1. Create the app**

**npx create-react-app studentapp**

**cd studentapp**

**2. Create component structure**

**Inside src/Components, create three files:**

**Home.js**

**About.js**

**Contact.js**

**3. Sample Component Code**

**Example: Home.js**

**import React, { Component } from 'react';**

**class Home extends Component {**

**render() {**

**return (**

**<div>**

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

**</div>**

**);**

**}**

**}**

**export default Home;**

**Repeat similarly for About.js and Contact.js with respective messages.**

**4. Modify App.js to include all components**

**import React from 'react';**

**import './App.css';**

**import Home from './Components/Home';**

**import About from './Components/About';**

**import Contact from './Components/Contact';**

**function App() {**

**return (**

**<div className="container">**

**<h1>Student Management Portal</h1>**

**<Home />**

**<About />**

**<Contact />**

**</div>**

**);**

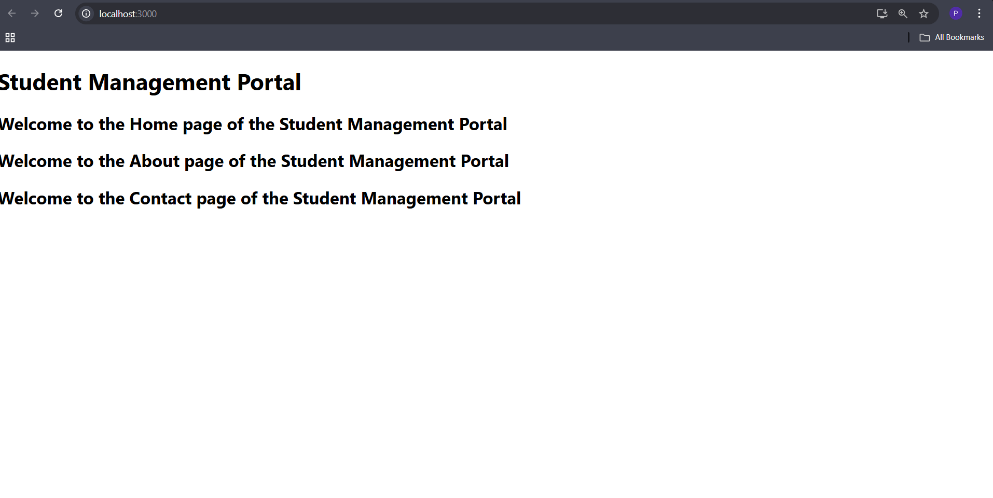
**}**

**export default App;**

**Run the Application**

**npm start**

**OUTPUT:**

****

**3. ReactJS-HOL**

Getting Started

1. Create React App

npx create-react-app scorecalculatorapp

cd scorecalculatorapp

2. Create Component

Inside src, create a folder named:

src/components

Inside it, create the file: CalculateScore.js

Paste the following code into it:

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>

)

3. Add Styles

Create a file: src/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;

}

4. Update App.js

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

function App() {

return (

<div>

<CalculateScore

Name={"Steeve"}

School={"DNV Public School"}

total={284}

goal={3}

/>

</div>

);

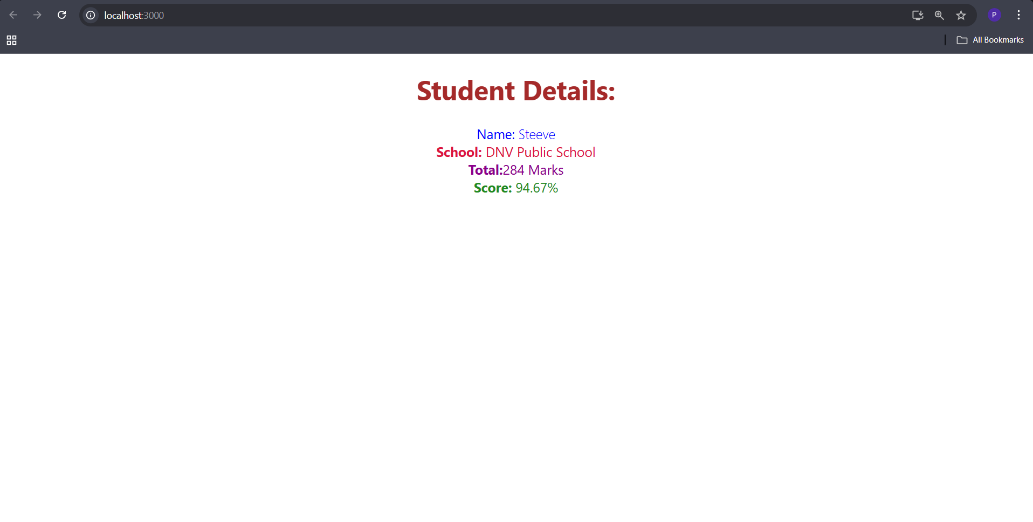
}

export default App;

Run the Application

npm start

**OUTPUT:**

****

**4. ReactJS-HOL**

Installation

1. Create the app using create-react-app

npx create-react-app blogapp

cd blogapp

2. Replace/Add the following files as described below

Post.js

Implementation

Post Model (Post.js)

class Post {

constructor(id, title, body) {

this.id = id;

this.title = title;

this.body = body;

}

}

Posts Component

constructor: Initializes state with empty posts.

loadPosts(): Fetches data from https://jsonplaceholder.typicode.com/posts.

componentDidMount(): Triggers API call after mounting.

componentDidCatch(): Catches any render-time errors.

render(): Maps post data to HTML using <h2> and <p>. Renders the Posts component in the root application view.

Usage

npm start

**5. ReactJS-HOL**

Project Setup

# 1. Clone or unzip the project

cd cohort-dashboard

# 2. Install dependencies

npm install

# 3. Start the development server

npm start

Styling Techniques Used

Component/CohortDetails.module.css

Scoped styles for the component using a CSS Module:

.box {

width: 300px;

display: inline-block;

margin: 10px;

padding: 10px 20px;

border: 1px solid black;

border-radius: 10px;

}

dt {

font-weight: 500;

}

Conditional Inline Styling

The <h3> element uses a green color if the cohort status is ongoing, and blue otherwise:

const statusColor = cohort.currentStatus.toLowerCase().trim() === 'ongoing' ? 'green' : 'blue';

...

<div className={styles.box}>

<h3 style={{ color: statusColor }}>

{props.cohort.cohortCode} -

<span>{props.cohort.technology}</span>

</h3>

...

<div>

Final Output Preview

Each cohort is shown in a styled box with borders and padding

The title (<h3>) turns green for ongoing cohorts and blue otherwise

Labels are semi-bold (font-weight: 500) using the <dt> tag style

OUTPUT:

