**1.Objectives**

* List the features of ES6
* Explain JavaScript let
* Identify the differences between var and let
* Explain JavaScript const
* Explain ES6 class fundamentals
* Explain ES6 class inheritance
* Define ES6 arrow functions
* Identify set(), map()

In this hands-on lab, you will learn how to:

* Use map() method of ES6
* Apply arrow functions of ES6
* Implement Destructuring features of ES6

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **60 minutes.**

Create a React Application named “cricketapp” with the following components:

1. ListofPlayers

* Declare an array with 11 players and store details of their names and scores using the map feature of ES6



* Filter the players with scores below 70 using arrow functions of ES6.



1. IndianPlayers
   1. Display the Odd Team Player and Even Team players using the Destructuring features of ES6



* 1. Declare two arrays T20players and RanjiTrophy players and merge the two arrays and display them using the Merge feature of ES6



Display these two components in the same home page using a simple if else in the flag variable.

npx create-react-app cricketapp

cd cricketapp

**ListOfPlayers.js:**

// src/components/ListOfPlayers.js

import React from 'react';

const players = [

{ name: 'Jack', score: 50 },

{ name: 'Michael', score: 70 },

{ name: 'John', score: 40 },

{ name: 'Ann', score: 61 },

{ name: 'Elisabeth', score: 61 },

{ name: 'Sachin', score: 95 },

{ name: 'Dhoni', score: 100 },

{ name: 'Virat', score: 84 },

{ name: 'Jadeja', score: 64 },

{ name: 'Raina', score: 75 },

{ name: 'Rohit', score: 80 },

];

const ListOfPlayers = () => {

const below70 = players.filter((item) => item.score <= 70);

return (

<div>

<h2>List of Players</h2>

<ul>

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

<li key={index}>

Mr. {item.name} <span>{item.score}</span>

</li>

))}

</ul>

<h2>List of Players having Scores Less than 70</h2>

<ul>

{below70.map((item, index) => (

<li key={index}>

Mr. {item.name} <span>{item.score}</span>

</li>

))}

</ul>

</div>

);

};

export default ListOfPlayers;

**IndianPlayers.js :**

// src/components/IndianPlayers.js

import React from 'react';

export function OddPlayers([first, , third, , fifth]) {

return (

<div>

<h2>Odd Players</h2>

<ul>

<li>First : {first}</li>

<li>Third : {third}</li>

<li>Fifth : {fifth}</li>

</ul>

</div>

);

}

export function EvenPlayers([, second, , fourth, , sixth]) {

return (

<div>

<h2>Even Players</h2>

<ul>

<li>Second : {second}</li>

<li>Fourth : {fourth}</li>

<li>Sixth : {sixth}</li>

</ul>

</div>

);

}

export function IndianPlayersMerged() {

const T20Players = [

'Mr. First Player',

'Mr. Second Player',

'Mr. Third Player',

];

const RanjiPlayers = ['Mr. Fourth Player', 'Mr. Fifth Player', 'Mr. Sixth Player'];

const AllPlayers = [...T20Players, ...RanjiPlayers];

return (

<div>

<h2>List of Indian Players Merged:</h2>

<ul>

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

<li key={index}>{item}</li>

))}

</ul>

</div>

);

}

**App.js :**

// src/App.js

import React from 'react';

import ListOfPlayers from './components/ListOfPlayers';

import { OddPlayers, EvenPlayers, IndianPlayersMerged } from './components/IndianPlayers';

function App() {

const flag = true; // Change to false to switch component

const team = ['Sachin1', 'Dhoni2', 'Virat3', 'Rohit4', 'Yuvaraj5', 'Raina6'];

return (

<div className="App">

{flag ? (

<>

<ListOfPlayers />

</>

) : (

<>

<OddPlayers {...[team]} />

<EvenPlayers {...[team]} />

<IndianPlayersMerged />

</>

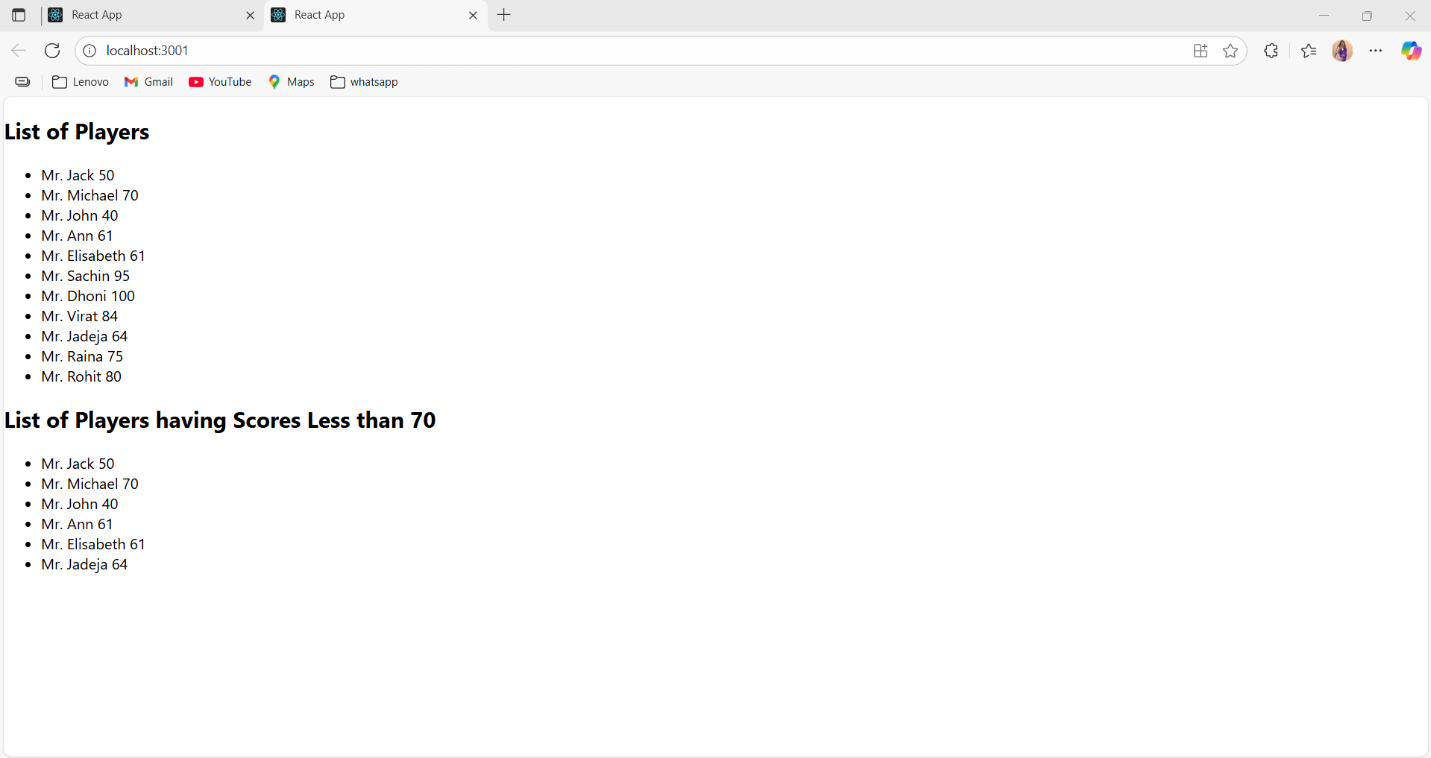
)}

</div>

);

}export default App;

**OUTPUT :**

****

**2.Objectives**

* Define JSX
* Explain about ECMA Script
* Explain React.createElement()
* Explain how to create React nodes with JSX
* Define how to render JSX to DOM
* Explain how to use JavaScript expressions in JSX
* Explain how to use inline CSS in JSX

In this hands-on lab, you will learn how to:

* Use JSX syntax in React applications
* Use inline CSS in JSX

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: 60 minutes.

Create a React Application named “officespacerentalapp” which uses React JSX to create elements, attributes and renders DOM to display the page.

Create an element to display the heading of the page.

Attribute to display the image of the office space

Create an object of office to display the details like Name, Rent and Address.

Create a list of Object and loop through the office space item to display more data.

To apply Css, Display the color of the Rent in Red if it’s below 60000 and in Green if it’s above 60000.

Hint:





npx create-react-app officespacerentalapp

cd officespacerentalapp

**App.js:**

import React from 'react';

import './App.css'; // Optional, for global styles

import img from './office.jpg'; // Save image in src folder

function App() {

const element = "Office Space";

const officeSpaces = [

{ Name: "DBS", Rent: 50000, Address: "Chennai" },

{ Name: "Regus", Rent: 65000, Address: "Hyderabad" },

{ Name: "WeWork", Rent: 40000, Address: "Bangalore" },

];

return (

<div>

<h1>{element} , at Affordable Range</h1>

{

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

let colors = [];

if (item.Rent <= 60000) {

colors.push("textRed");

} else {

colors.push("textGreen");

}

return (

<div key={index} style={{

border: "1px solid black",

width: "300px",

display: "inline-block",

padding: "10px 20px",

margin: "10px",

borderRadius: "10px"

}}>

<img src={img} width="100%" alt="Office Space" />

<h2>Name: {item.Name}</h2>

<h3 className={colors.join(" ")}>Rent: Rs. {item.Rent}</h3>

<h3>Address: {item.Address}</h3>

</div>

);

})

}

</div>

);

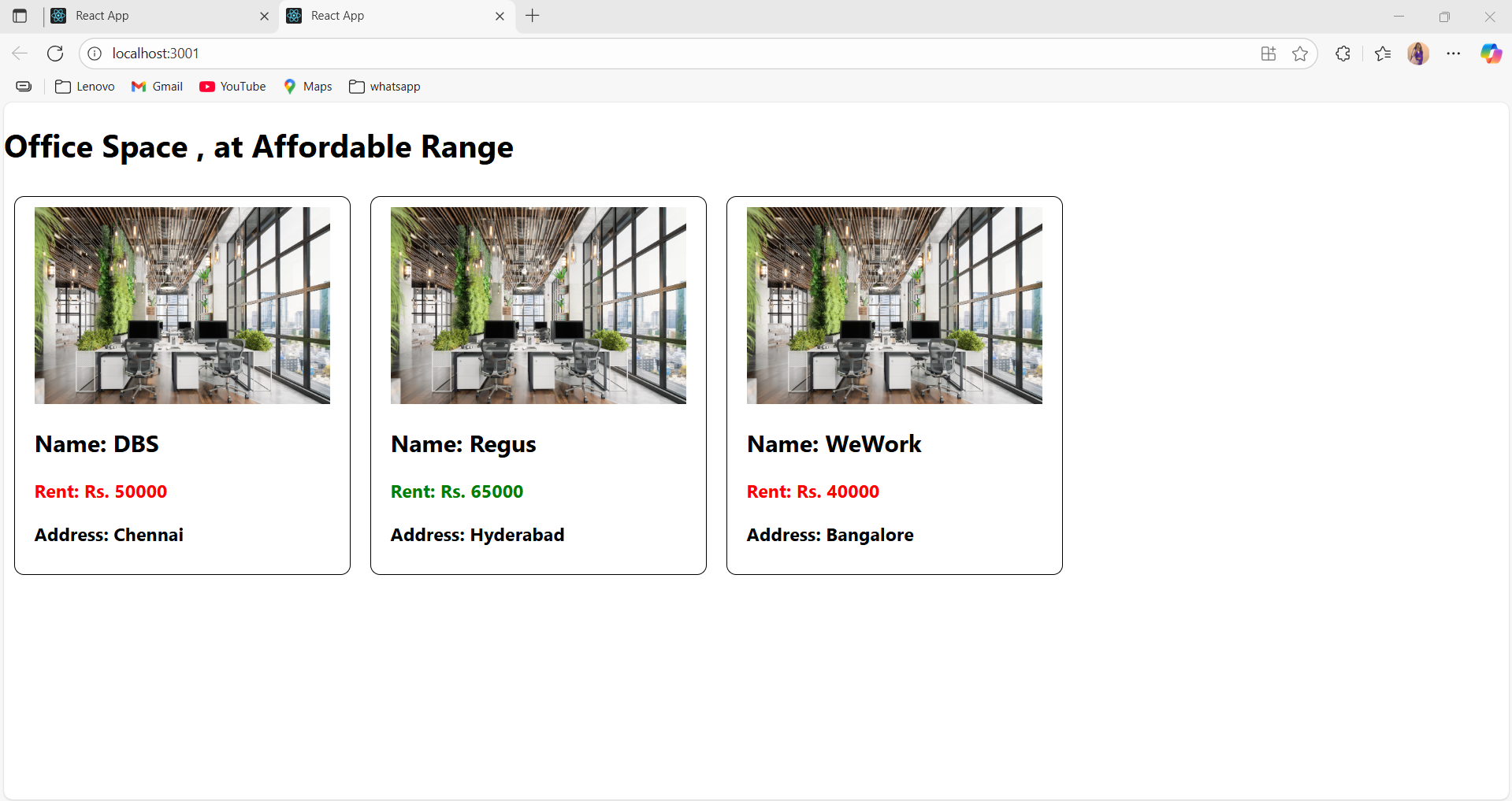
}

export default App;

Open the terminal and run:

**npm start**

**OUTPUT:**

****

**3.Objectives**

* Explain React events
* Explain about event handlers
* Define Synthetic event
* Identify React event naming convention

In this hands-on lab, you will learn how to:

* Implement Event handling concept in React applications
* Use this keyword
* Use synthetic event

Prerequisites

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

Notes

Estimated time to complete this lab: 90 minutes.

Create a React Application “eventexamplesapp” to handle various events of the form elements in HTML.

1. Create “Increment” button to increase the value of the counter and “Decrement” button to decrease the value of the counter. The “Increase” button should invoke multiple methods.
   1. To increment the value
   2. Say Hello followed by a static message.
2. Create a button “Say Welcome” which invokes the function which takes “welcome” as an argument.
3. Create a button which invokes synthetic event “OnPress” which display “I was clicked”

Create a “CurrencyConvertor” component which will convert the Indian Rupees to Euro when the Convert button is clicked.

Handle the Click event of the button to invoke the handleSubmit event and handle the conversion of the euro to rupees.

npx create-react-app eventexamplesapp

cd eventexamplesapp

**App.js :**

import React, { Component } from 'react';

import CurrencyConverter from './CurrencyConverter';

class App extends Component {

constructor() {

super();

this.state = {

count: 5,

};

this.handleClick = this.handleClick.bind(this);

}

increment = () => {

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

this.sayHello();

};

sayHello = () => {

alert('Hello! Member1');

};

decrement = () => {

this.setState({ count: this.state.count - 1 });

};

sayWelcome = (msg) => {

alert(msg);

};

handleClick(event) {

alert('I was clicked');

}

render() {

return (

<div>

<h3>{this.state.count}</h3>

<button onClick={this.increment}>Increment</button><br />

<button onClick={this.decrement}>Decrement</button><br />

<button onClick={() => this.sayWelcome('welcome')}>Say welcome</button><br />

<button onClick={this.handleClick}>Click on me</button><br /><br />

<CurrencyConverter />

</div>

);

}

}

export default App;

**4.Objectives**

* Explain about conditional rendering in React
* Define element variables
* Explain how to prevent components from rendering

In this hands-on lab, you will learn how to:

* Implement conditional rendering in React applications

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: 60 minutes.

Create a React Application named “ticketbookingapp” where the guest user can browse the page where the flight details are displayed whereas the logged in user only can book tickets.

The Login and Logout buttons should accordingly display different pages. Once the user is logged in the User page should be displayed. When the user clicks on Logout, the Guest page should be displayed.

Hint:







npx create-react-app ticketbookingapp

cd ticketbookingapp

**UserGreeting.js :**

import React from 'react';

function UserGreeting() {

return <h1>Welcome back</h1>;

}

export default UserGreeting;

**GuestGreeting.js:**

import React from 'react';

function GuestGreeting() {

return <h1>Please sign up.</h1>;

}

export default GuestGreeting;

**Greeting.js:**

import React from 'react';

import UserGreeting from './UserGreeting';

import GuestGreeting from './GuestGreeting';

function Greeting(props) {

const isLoggedIn = props.isLoggedIn;

if (isLoggedIn) {

return <UserGreeting />;

}

return <GuestGreeting />;

}

export default Greeting;

**App.js:**

import React, { Component } from 'react';

import Greeting from './Greeting';

import LoginButton from './LoginButton';

import LogoutButton from './LogoutButton';

class App extends Component {

constructor(props) {

super(props);

this.handleLoginClick = this.handleLoginClick.bind(this);

this.handleLogoutClick = this.handleLogoutClick.bind(this);

this.state = { isLoggedIn: false };

}

handleLoginClick() {

this.setState({ isLoggedIn: true });

}

handleLogoutClick() {

this.setState({ isLoggedIn: false });

}

render() {

const isLoggedIn = this.state.isLoggedIn;

let button;

if (isLoggedIn) {

button = <LogoutButton onClick={this.handleLogoutClick} />;

} else {

button = <LoginButton onClick={this.handleLoginClick} />;

}

return (

<div style={{ padding: '50px' }}>

<Greeting isLoggedIn={isLoggedIn} />

{button}

</div>

);

}

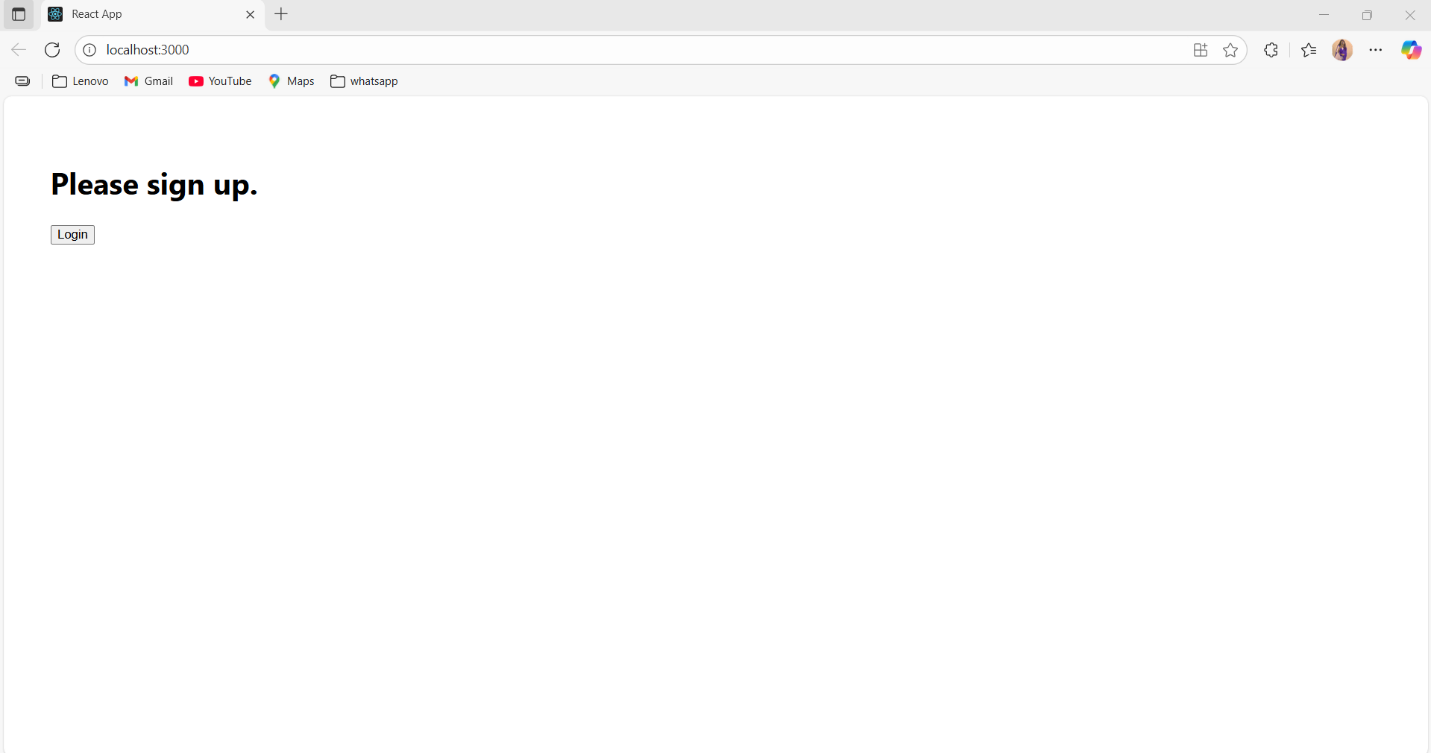
}

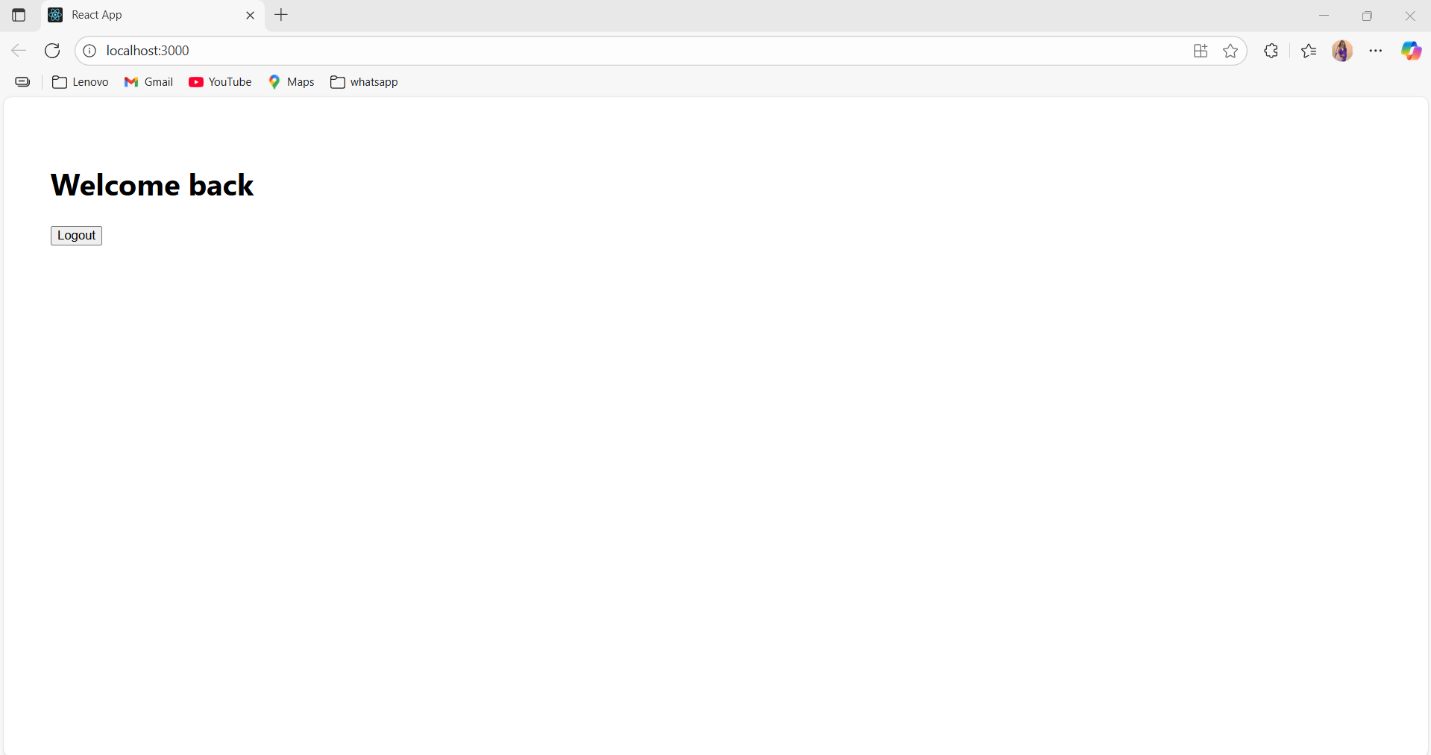
export default App;

In terminal :

**npm start**

**OUTPUT :**

****

****

**5.Objectives**

* Explain various ways of conditional rendering
* Explain how to render multiple components
* Define list component
* Explain about keys in React applications
* Explain how to extract components with keys
* Explain React Map, map() function

In this hands-on lab, you will learn how to:

* Implement conditional rendering in React applications

**Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: 60 minutes.

Create a React App named “bloggerapp” in with 3 components.

1. Book Details
2. Blog Details
3. Course Details

Implement this with as many ways possible of Conditional Rendering.

Hint:

****

****

****

npx create-react-app bloggerapp

cd bloggerapp

**BookDetails.js:**

import React from 'react';

const books = [

{ id: 101, bname: 'Master React', price: 670 },

{ id: 102, bname: 'Deep Dive into Angular 11', price: 800 },

{ id: 103, bname: 'Mongo Essentials', price: 450 },

];

const BookDetails = () => {

return (

<div className="st2">

<h1>Book Details</h1>

<ul>

{books.map((book) => (

<div key={book.id}>

<h3>{book.bname}</h3>

<h4>{book.price}</h4>

</div>

))}

</ul>

</div>

);

};

export default BookDetails;

**BlogDetails.js:**

import React from 'react';

const blogs = [

{

id: 1,

title: 'React Learning',

author: 'Stephen Biz',

content: 'Welcome to learning React!',

},

{

id: 2,

title: 'Installation',

author: 'Schewzdenier',

content: 'You can install React from npm.',

},

];

const BlogDetails = ({ show }) => {

if (!show) return null; // Conditional rendering

return (

<div className="v1">

<h1>Blog Details</h1>

{blogs.map((blog) => (

<div key={blog.id}>

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

<h4><b>{blog.author}</b></h4>

<p>{blog.content}</p>

</div>

))}

</div>

);

};

export default BlogDetails;

**App.js:**

import './App.css';

import BookDetails from './BookDetails';

import BlogDetails from './BlogDetails';

import CourseDetails from './CourseDetails';

function App() {

const showBlogs = true; // change this to false to test conditional rendering

return (

<div className="App">

<div style={{ display: 'flex', justifyContent: 'space-around' }}>

<BookDetails />

<BlogDetails show={showBlogs} />

<CourseDetails />

</div>

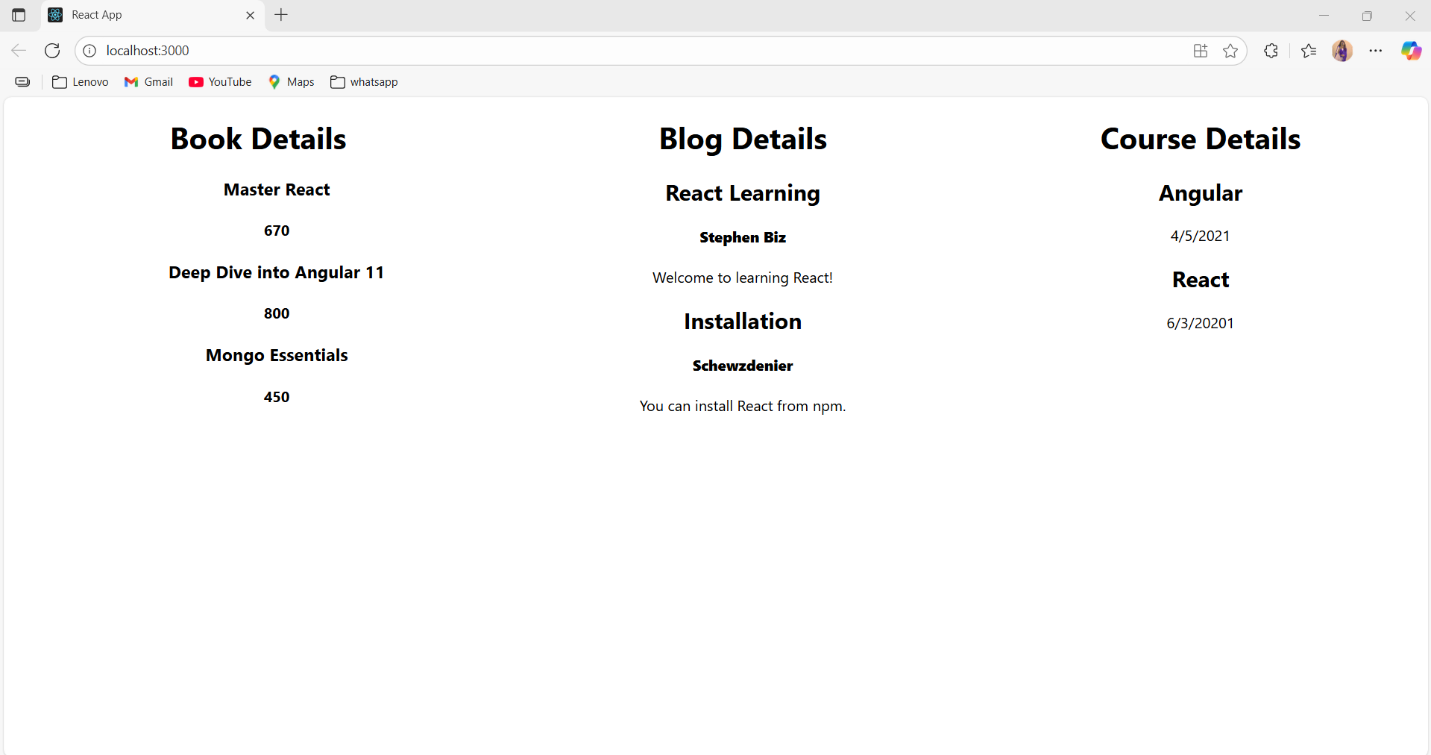
</div>

);

}

export default App;

**OUTPUT :**

****