**EXERCISE 1: Create a React Application named “cricketapp” with the following components:**

**npx create-react-app cricketapp**

**cd cricketapp**

* 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.

**D:\CTS\REACT JS\cricketapp\src\components\ListofPlayers.js**

import React, { useState } from 'react';

const ListofPlayers = () => {

    const players = [

        { name: "Virat Kohli", score: 85 },

        { name: "Rohit Sharma", score: 92 },

        { name: "KL Rahul", score: 45 },

        { name: "Hardik Pandya", score: 78 },

        { name: "Rishabh Pant", score: 55 },

        { name: "Ravindra Jadeja", score: 35 },

        { name: "Bhuvneshwar Kumar", score: 25 },

        { name: "Jasprit Bumrah", score: 15 },

        { name: "Mohammed Shami", score: 68 },

        { name: "Yuzvendra Chahal", score: 42 },

        { name: "Shikhar Dhawan", score: 88 }

    ];

    const lowScorePlayers = players.filter(player => player.score < 70);

    return (

        <div>

            <h2>List of Players</h2>

            <h3>All Players:</h3>

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

                <div key={index}>

                    {player.name} - Score: {player.score}

                </div>

            ))}

            <h3>Players with scores below 70:</h3>

            {lowScorePlayers.map((player, index) => (

                <div key={index}>

                    {player.name} - Score: {player.score}

                </div>

            ))}

        </div>

    );

};

export default ListofPlayers;

* IndianPlayers
  1. Display the Odd Team Player and Even Team players using the Destructuring features of ES6
  2. Declare two arrays T20players and RanjiTrophy players and merge the two arrays and display them using the Merge feature of ES6

**D:\CTS\REACT JS\cricketapp\src\components\IndianPlayers.js**

import React, { useState } from 'react';

const IndianPlayers = () => {

    const teamPlayers = [

        "Virat Kohli", "Rohit Sharma", "KL Rahul", "Hardik Pandya",

        "Rishabh Pant", "Ravindra Jadeja", "Bhuvneshwar Kumar", "Jasprit Bumrah"

    ];

    const [p1, p2, p3, p4, p5, p6, p7, p8] = teamPlayers;

    const oddTeamPlayers = [p1, p3, p5, p7];

    const evenTeamPlayers = [p2, p4, p6, p8];

    const T20players = ["Virat Kohli", "Rohit Sharma", "KL Rahul", "Hardik Pandya"];

    const RanjiTrophyPlayers = ["Prithvi Shaw", "Ajinkya Rahane", "Cheteshwar Pujara", "Wriddhiman Saha"];

    const mergedPlayers = [...T20players, ...RanjiTrophyPlayers];

    return (

        <div>

            <h2>Indian Players</h2>

            <h3>Odd Team Players:</h3>

            {oddTeamPlayers.map((player, index) => (

                <div key={index}>{player}</div>

            ))}

            <h3>Even Team Players:</h3>

            {evenTeamPlayers.map((player, index) => (

                <div key={index}>{player}</div>

            ))}

            <h3>Merged Players (T20 + Ranji Trophy):</h3>

            {mergedPlayers.map((player, index) => (

                <div key={index}>

                    {player} {index < T20players.length ? "(T20)" : "(Ranji)"}

                </div>

            ))}

        </div>

    );

};

export default IndianPlayers;

**D:\CTS\REACT JS\cricketapp\src\App.js**

import React, { useState } from 'react';

import ListofPlayers from './components/ListofPlayers';

import IndianPlayers from './components/IndianPlayers';

const App = () => {

  const [flag, setFlag] = useState(true);

  return (

    <div>

      <h1>Cricket App</h1>

      <button onClick={() => setFlag(!flag)}>

        Toggle View (Current flag: {flag.toString()})

      </button>

      <br /><br />

      {flag ? <ListofPlayers /> : <IndianPlayers />}

    </div>

  );

};

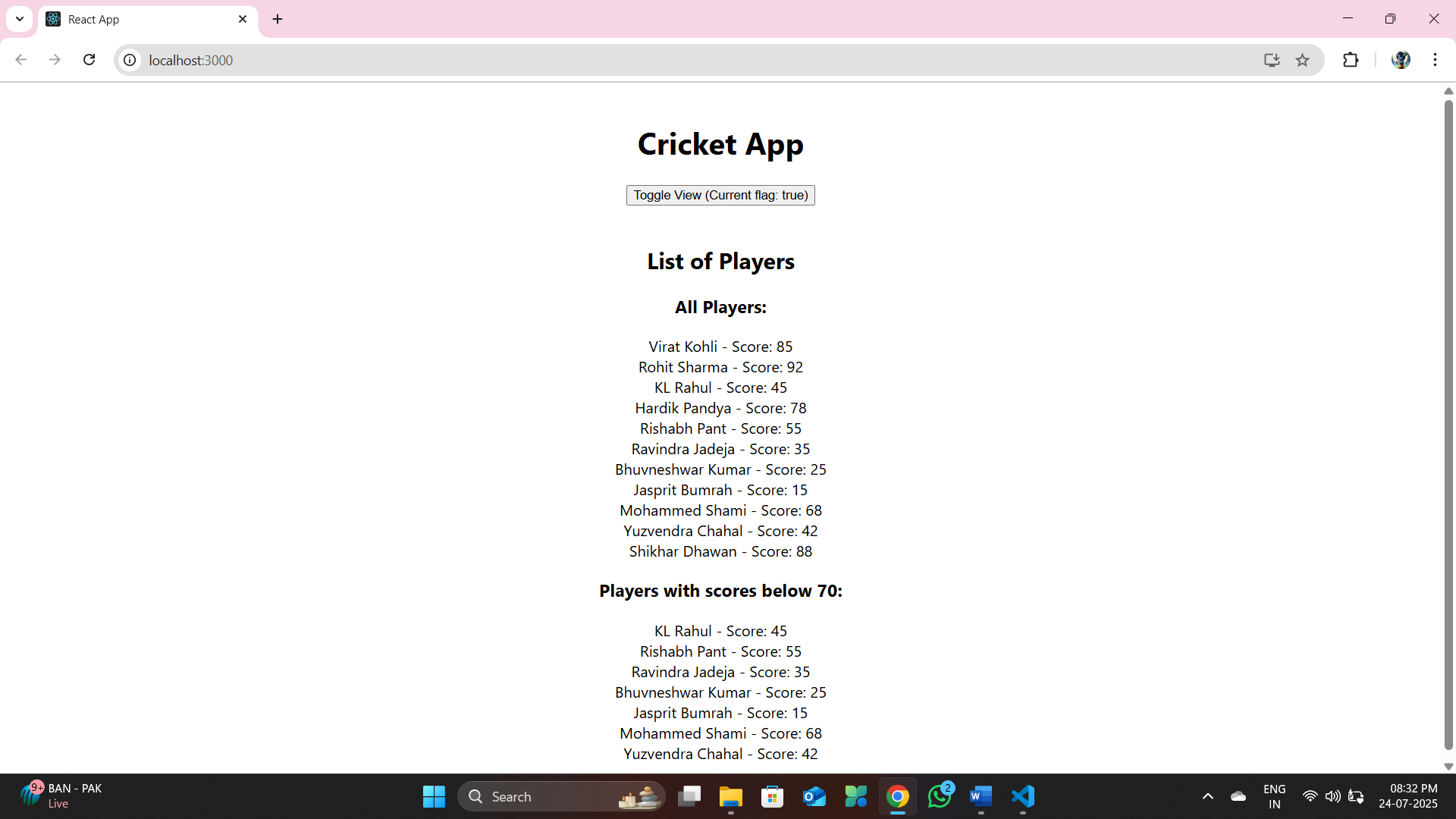
export default App;

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

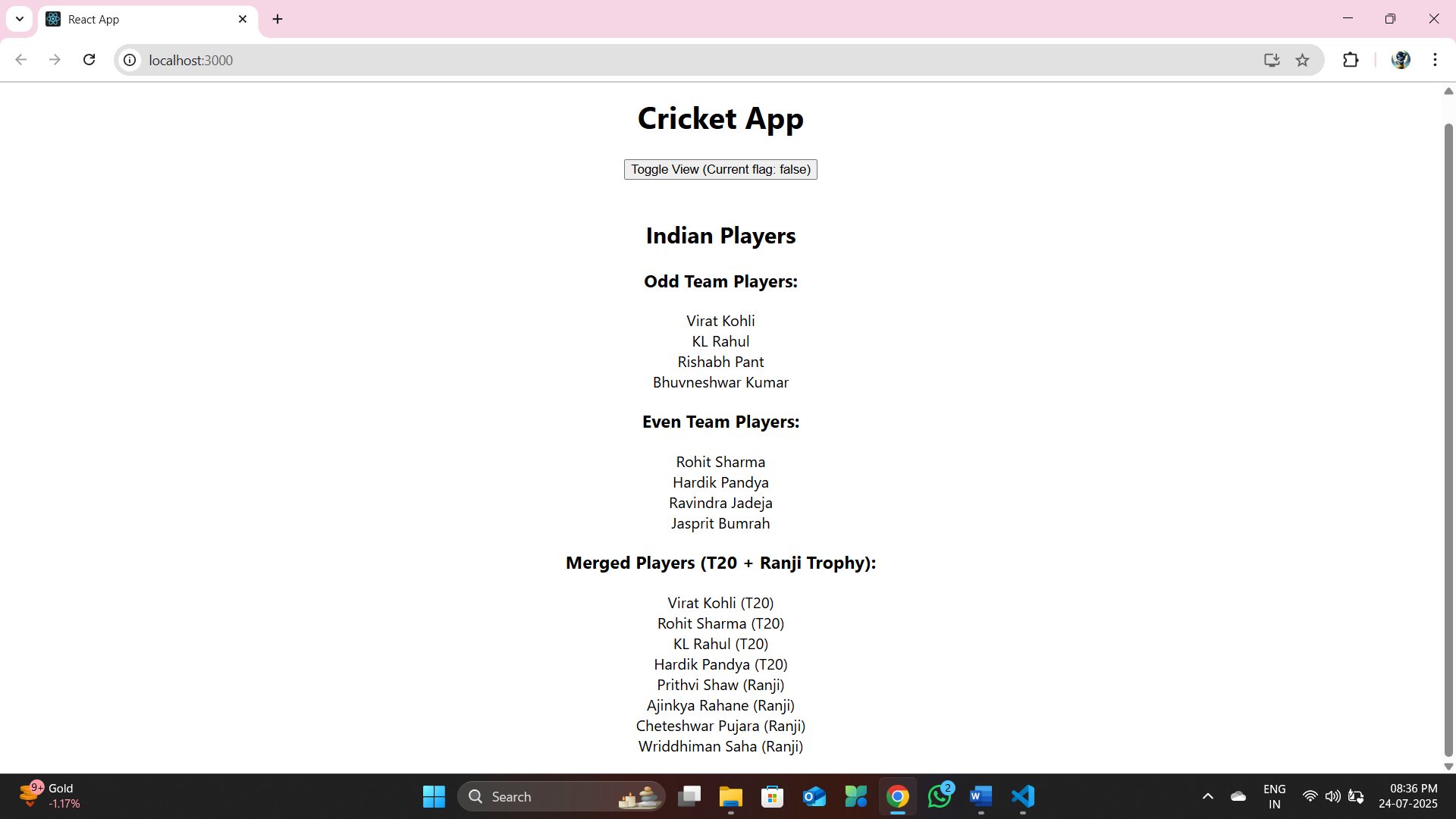
**Output:**

**npm start**

When Flag=true



When Flag=false



**EXERCISE 2: 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.

**npx create-react-app officespacerentalapp**

**cd officespacerentalapp**

**D:\CTS\REACT JS\officespacerentalapp\src\App.js**

import React from "react";

import "./App.css";

function App() {

  const offices = [

    {

      id: 1,

      name: "WeWork",

      rent: 45000,

      address: "Tidel Park, Chennai",

      image: "/office1.jpg"

    },

    {

      id: 2,

      name: "Regus",

      rent: 65000,

      address: "OMR, Chennai",

      image: "/office2.jpg"

    },

    {

      id: 3,

      name: "SmartWorks",

      rent: 55000,

      address: "Guindy, Chennai",

      image: "/office3.jpg"

    },

    {

      id: 4,

      name: "CTS",

      rent: 70000,

      address: "Sholinganallur, Chennai",

      image: "/office4.jpg"

    }

  ];

  const headingStyle = {

    textAlign: "center",

    color: "#333"

  };

  return (

    <div className="App">

      <h1 style={headingStyle}>Office Space Rental App</h1>

      <div className="office-container">

        {offices.map((office) => (

          <div key={office.id} className="office-card">

            <img src={office.image} alt={office.name} width="100%" />

            <h2>{office.name}</h2>

            <p>

              <strong>Address:</strong> {office.address}

            </p>

            <p

              style={{

                color: office.rent > 60000 ? "green" : "red",

                fontWeight: "bold"

              }}

            >

              Rent: ₹{office.rent}

            </p>

          </div>

        ))}

      </div>

    </div>

  );

}

export default App;

**officespacerentalapp\src\App.css**

.App {

  text-align: center;

  font-family: Arial, sans-serif;

}

.office-container {

  display: flex;

  flex-wrap: wrap;

  justify-content: center;

  gap: 20px;

}

.office-card {

  border: 1px solid #ccc;

  padding: 10px;

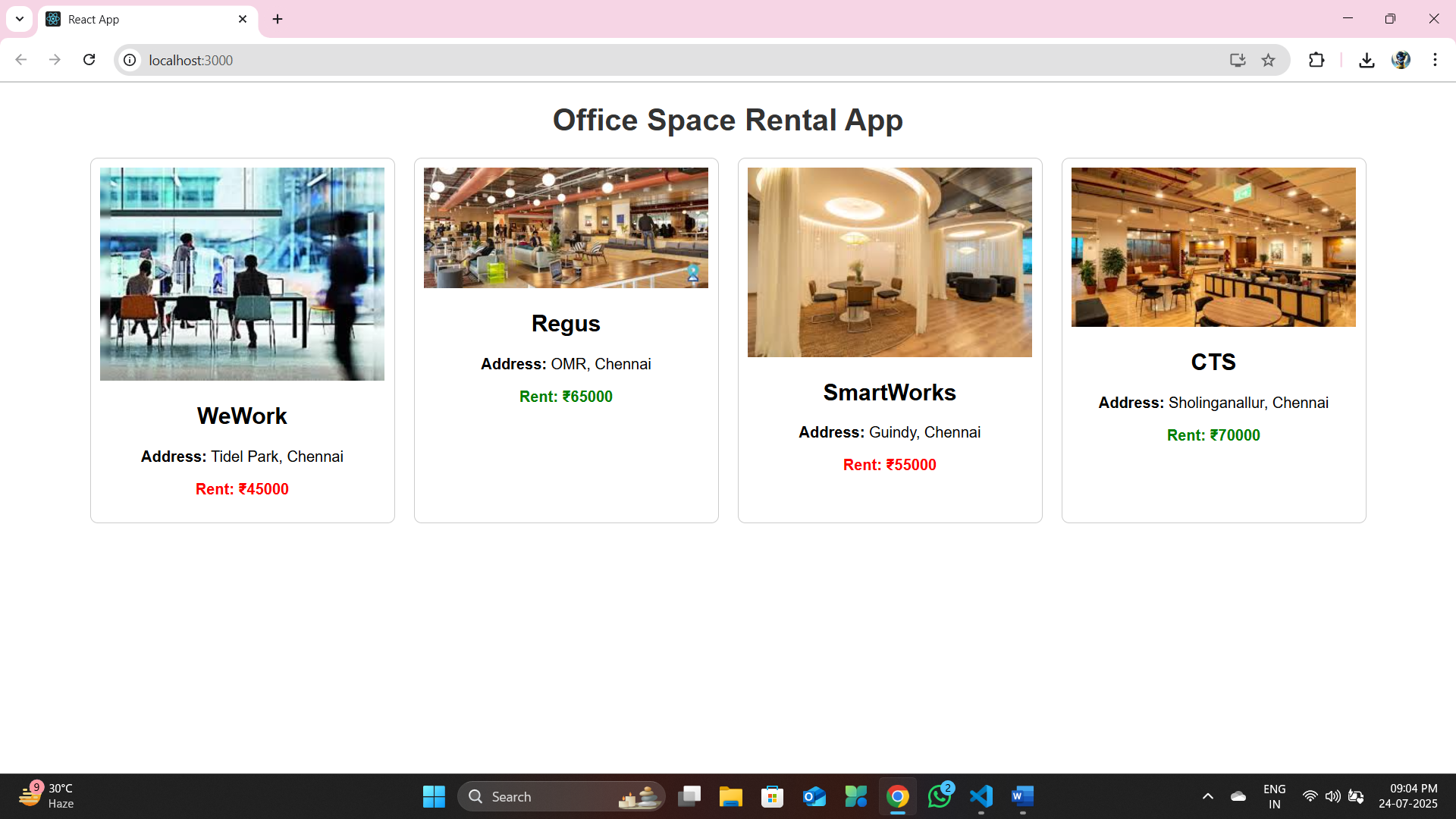
  border-radius: 8px;

  width: 300px;

}

**Output :**

**npm start**



**EXERCISE 3: Create a React Application “eventexamplesapp” to handle various events of the form elements in HTML.**

**npx create-react-app eventexamplesapp**

**D:\CTS\REACT JS\eventexamplesapp\src\App.js**

import React from 'react';

import Counter from './Counter';

import Welcome from './Welcome';

import SyntheticClick from './SyntheticClick';

import CurrencyConvertor from './CurrencyConvertor';

function App() {

  return (

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

      <h1>Event Handling Examples</h1>

      <Counter />

      <Welcome />

      <SyntheticClick />

      <CurrencyConvertor />

    </div>

  );

}

export default App;

* 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.

**D:\CTS\REACT JS\eventexamplesapp\src\Counter.js**

import React, { Component } from 'react';

class Counter extends Component {

    constructor(props) {

        super(props);

        this.state = {

            count: 0

        };

    }

    increment = () => {

        this.setState({ count: this.state.count + 1 }, () => {

            this.sayHello();

        });

    };

    sayHello = () => {

       alert('Hello! Counter has been increased.');

    };

    decrement = () => {

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

    };

    render() {

        return (

            <div>

                <h2>Counter: {this.state.count}</h2>

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

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

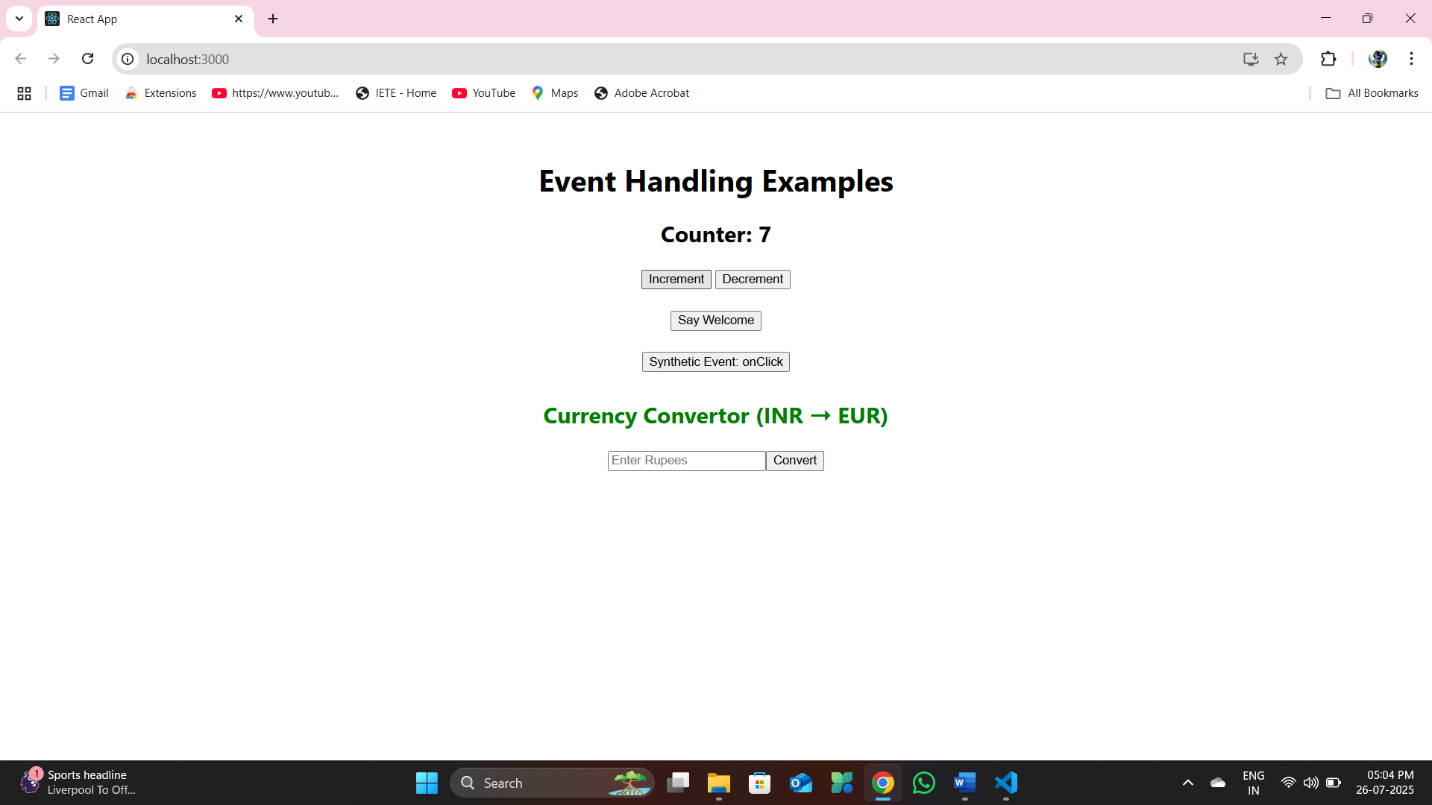
            </div>

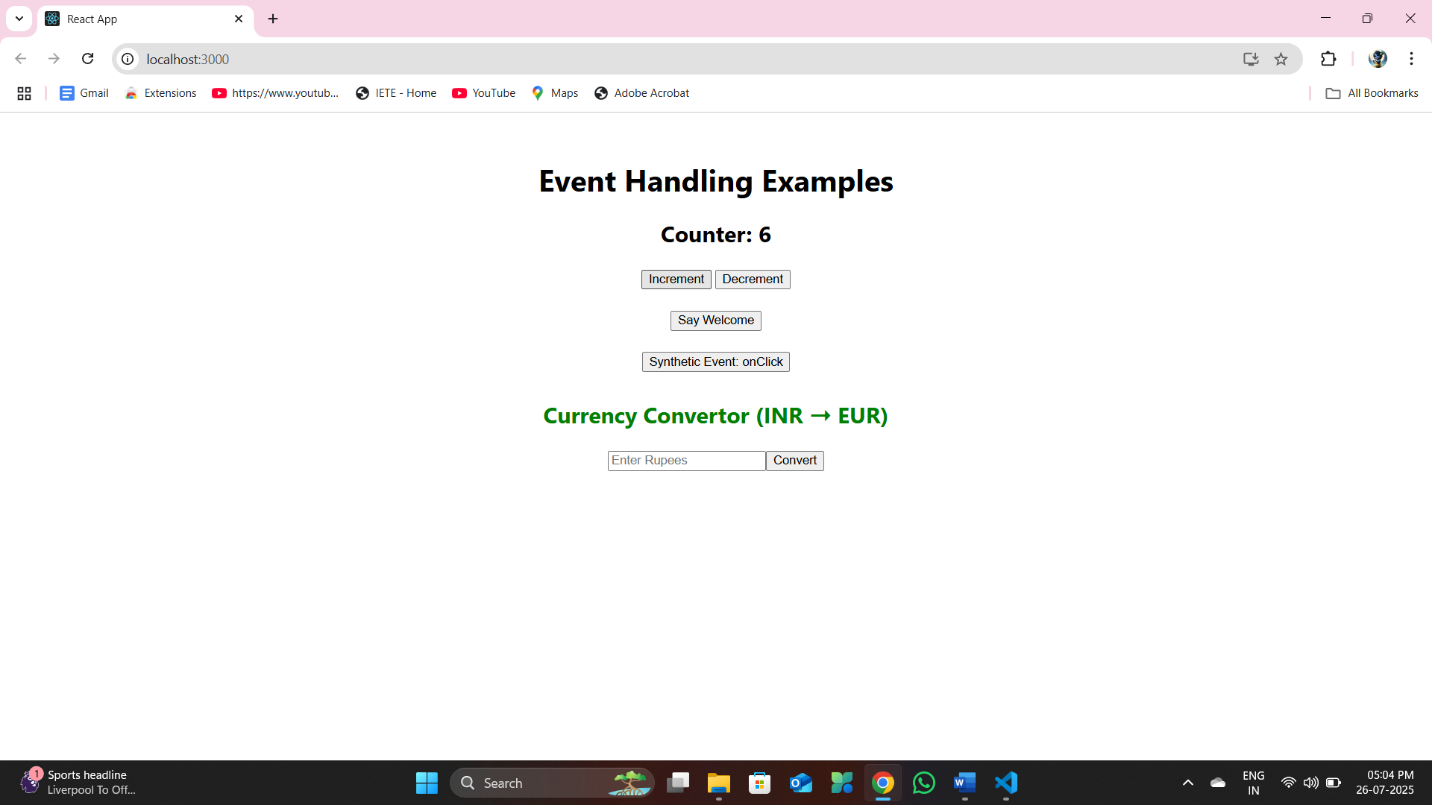
        );

    }

}

export default Counter;

****



* Create a button “Say Welcome” which invokes the function which takes “welcome” as an argument.

**D:\CTS\REACT JS\eventexamplesapp\src\Welcome.js**

import React from 'react';

function sayWelcome(message) {

    alert(`Say ${message}`);

}

function Welcome() {

    return (

        <div style={{ marginTop: '20px' }}>

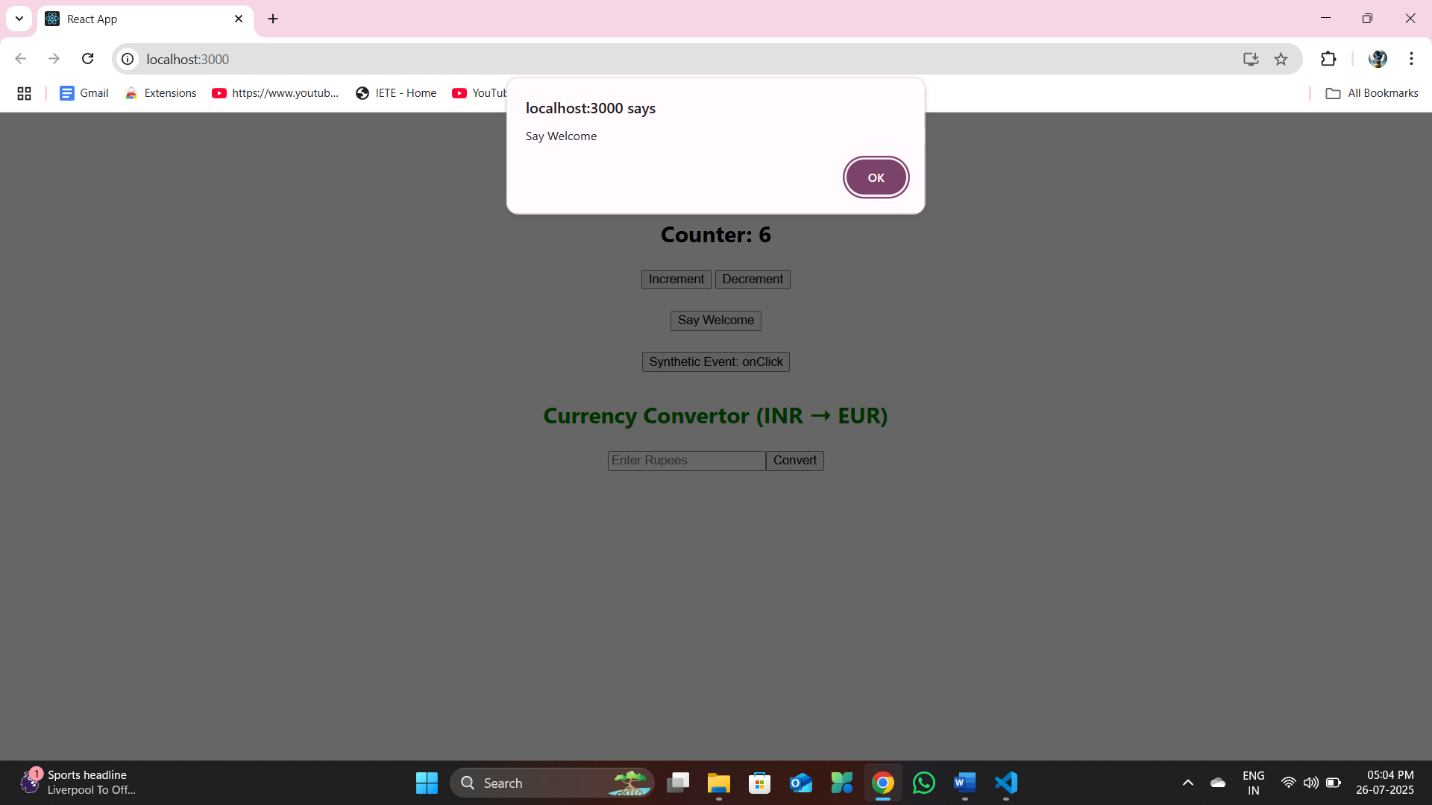
            <button onClick={() => sayWelcome('Welcome')}>Say Welcome</button>

        </div>

    );

}

export default Welcome;

****

* Create a button which invokes synthetic event “OnPress” which display “I was clicked”

**D:\CTS\REACT JS\eventexamplesapp\src\SyntheticClick.js**

import React from 'react';

function SyntheticClick() {

    const handleClick = (event) => {

        alert('I was clicked');

        console.log('Synthetic event object:', event);

    };

    return (

        <div style={{ marginTop: '20px' }}>

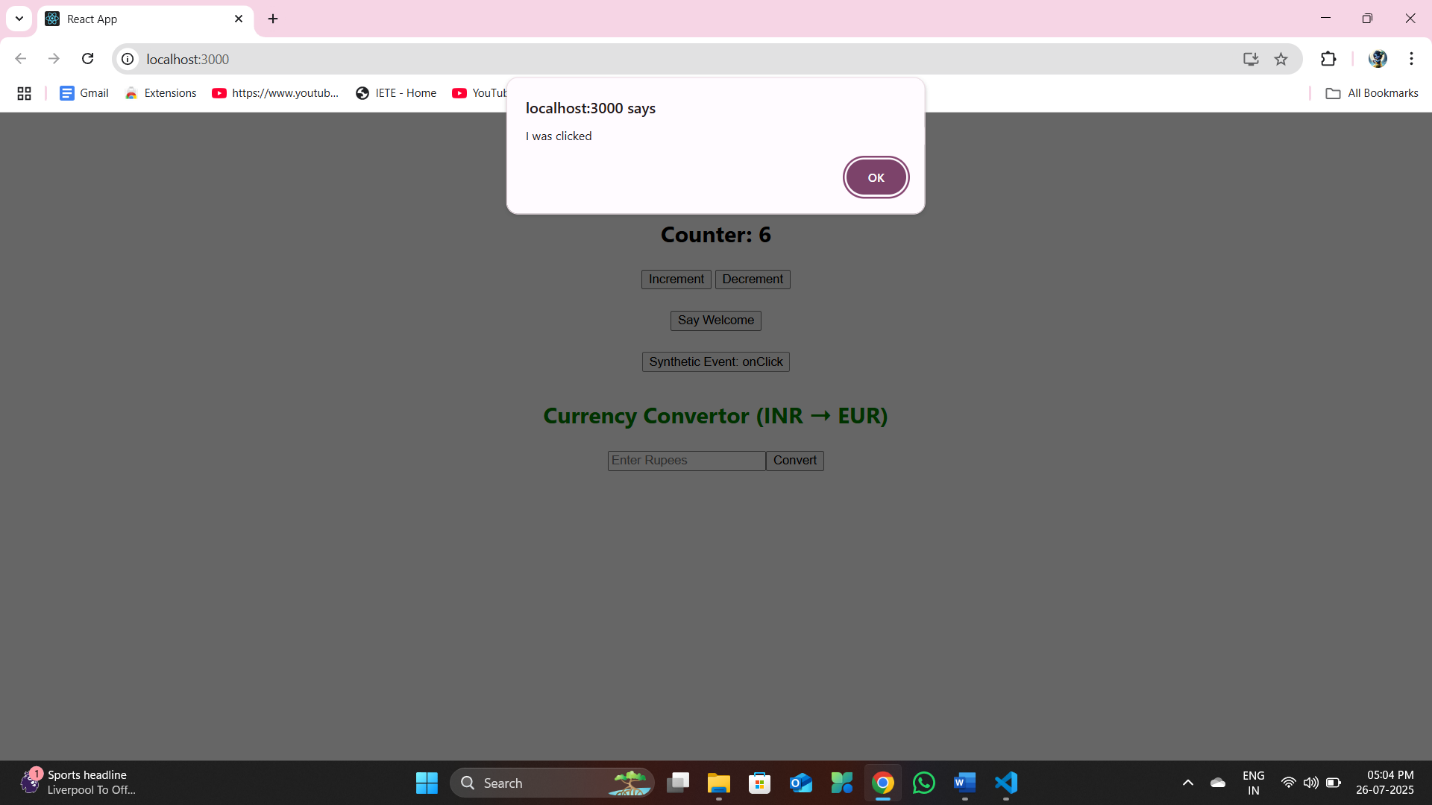
            <button onClick={handleClick}>Synthetic Event: onClick</button>

        </div>

    );

}

export default SyntheticClick;



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.

**D:\CTS\REACT JS\eventexamplesapp\src\CurrencyConvertor.js**

import React, { useState } from 'react';

function CurrencyConvertor() {

    const [rupees, setRupees] = useState('');

    const [euro, setEuro] = useState('');

    const handleSubmit = (e) => {

        e.preventDefault();

        const euroValue = (parseFloat(rupees) / 90).toFixed(2);

        setEuro(euroValue);

    };

    return (

        <div style={{ marginTop: '30px' }}>

            <h2 style={{ color: 'green' }}>Currency Convertor (INR ➝ EUR)</h2>

            <form onSubmit={handleSubmit}>

                <input

                    type="number"

                    placeholder="Enter Rupees"

                    value={rupees}

                    onChange={(e) => setRupees(e.target.value)}

                />

                <button type="submit" >Convert</button>

            </form>

            {rupees && euro && (

                alert(`Converting to Euro Amount is : ₹${rupees} = €${euro}`)

            )}

            {rupees && <p>Rupees: ₹{rupees}</p>}

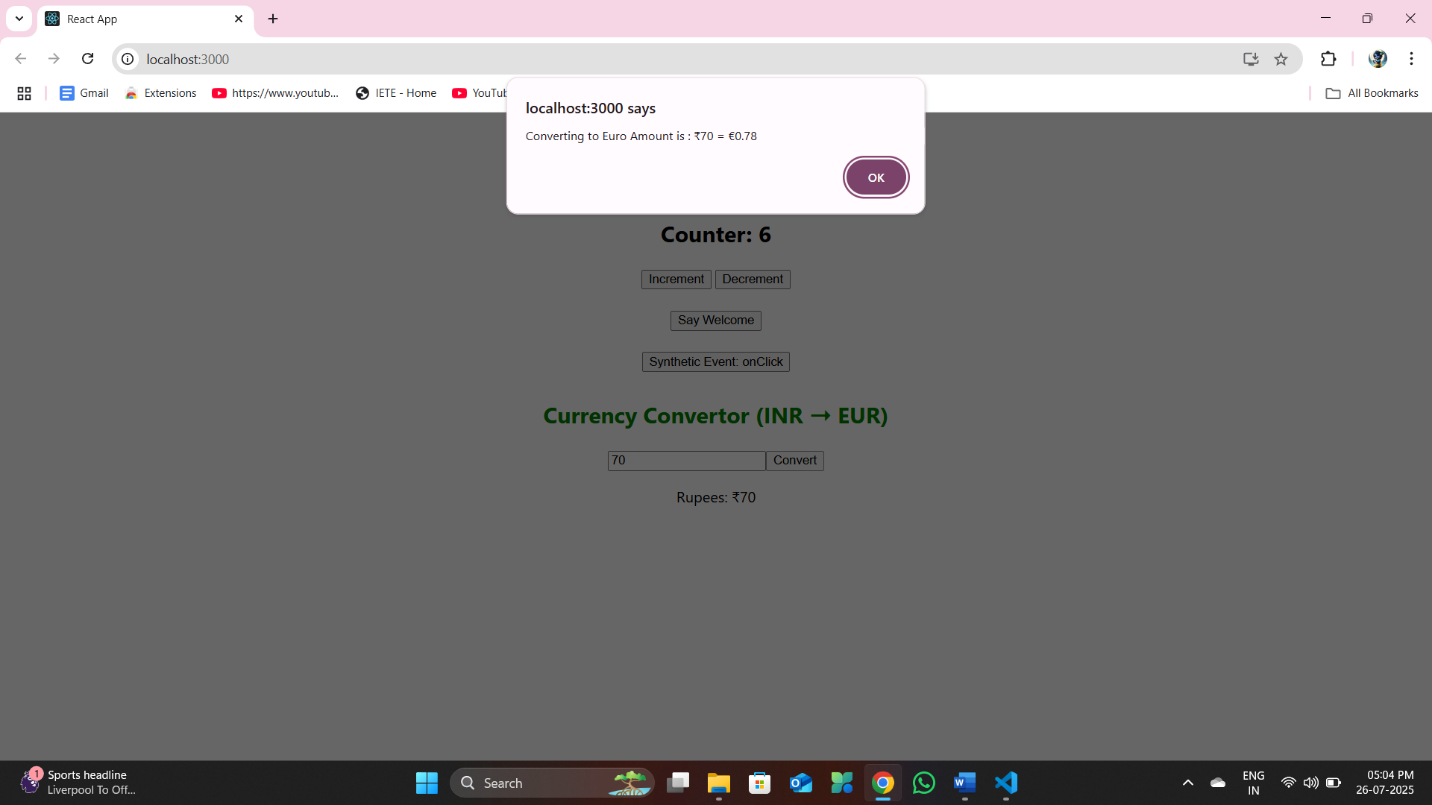
            {euro && <p>Euro: €{euro}</p>}

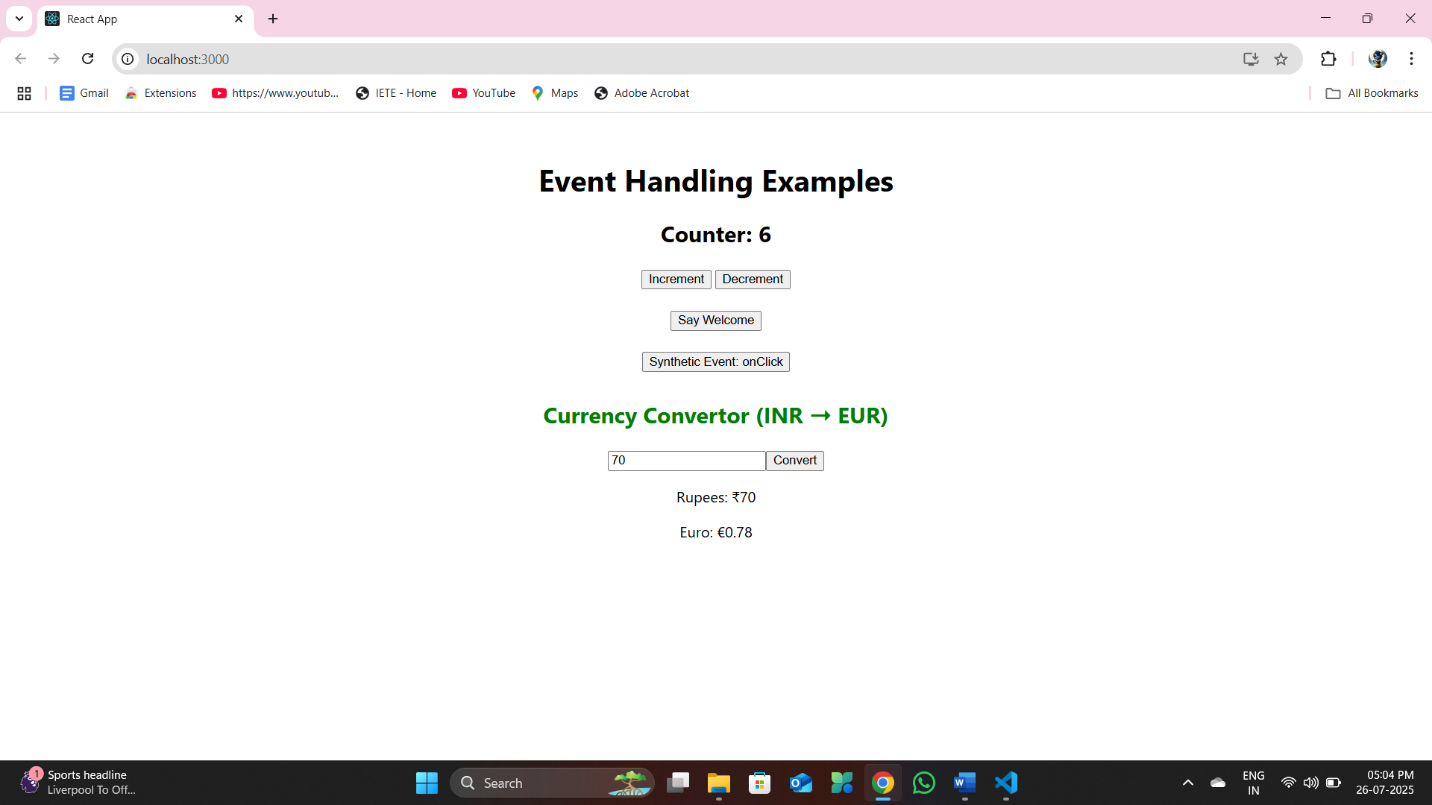
        </div>

    );

}

export default CurrencyConvertor;





**EXERCISE 4: 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.**

**npx create-react-app ticketbookingapp**

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.

**D:\CTS\REACT JS\ticketbookingapp\src\GuestPage.js**

import React from "react";

function GuestPage() {

    return (

        <div>

            <h2>Welcome Guest</h2>

            <p>Please login to book your flight tickets.</p>

            <h3>Available Flights</h3>

            <ul>

                <li>Chennai → Delhi</li>

                <li>Hyderabad → Mumbai</li>

                <li>Bangalore → Kolkata</li>

            </ul>

            <h1 style={{ color: "red" }}>Please Sign Up!!!</h1>

        </div>

    );

}

export default GuestPage;

**D:\CTS\REACT JS\ticketbookingapp\src\UserPage.js**

import React from "react";

function UserPage() {

    return (

        <div>

            <h2 style={{ color: "green" }}>Welcome User</h2>

            <p>You can now book your flight tickets below:</p>

            <button style={{ backgroundColor: "lightblue" }}>Book Chennai → Delhi</button><br /><br />

            <button style={{ backgroundColor: "lightblue" }}>Book Hyderabad → Mumbai</button><br /><br />

            <button style={{ backgroundColor: "lightblue" }}>Book Bangalore → Kolkata</button>

        </div>

    );

}

export default UserPage;

**D:\CTS\REACT JS\ticketbookingapp\src\App.js**

import React, { useState } from "react";

import GuestPage from "./GuestPage";

import UserPage from "./UserPage";

function App() {

  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const handleLogin = () => setIsLoggedIn(true);

  const handleLogout = () => setIsLoggedIn(false);

  let page;

  if (isLoggedIn) {

    page = <UserPage />;

  } else {

    page = <GuestPage />;

  }

  return (

    <div style={{ textAlign: "center", padding: "20px" }}>

      <h1>Flight Ticket Booking App</h1>

      {isLoggedIn ? (

        <button onClick={handleLogout}>Logout</button>

      ) : (

        <button onClick={handleLogin}>Login</button>

      )}

      <hr />

      {page}

    </div>

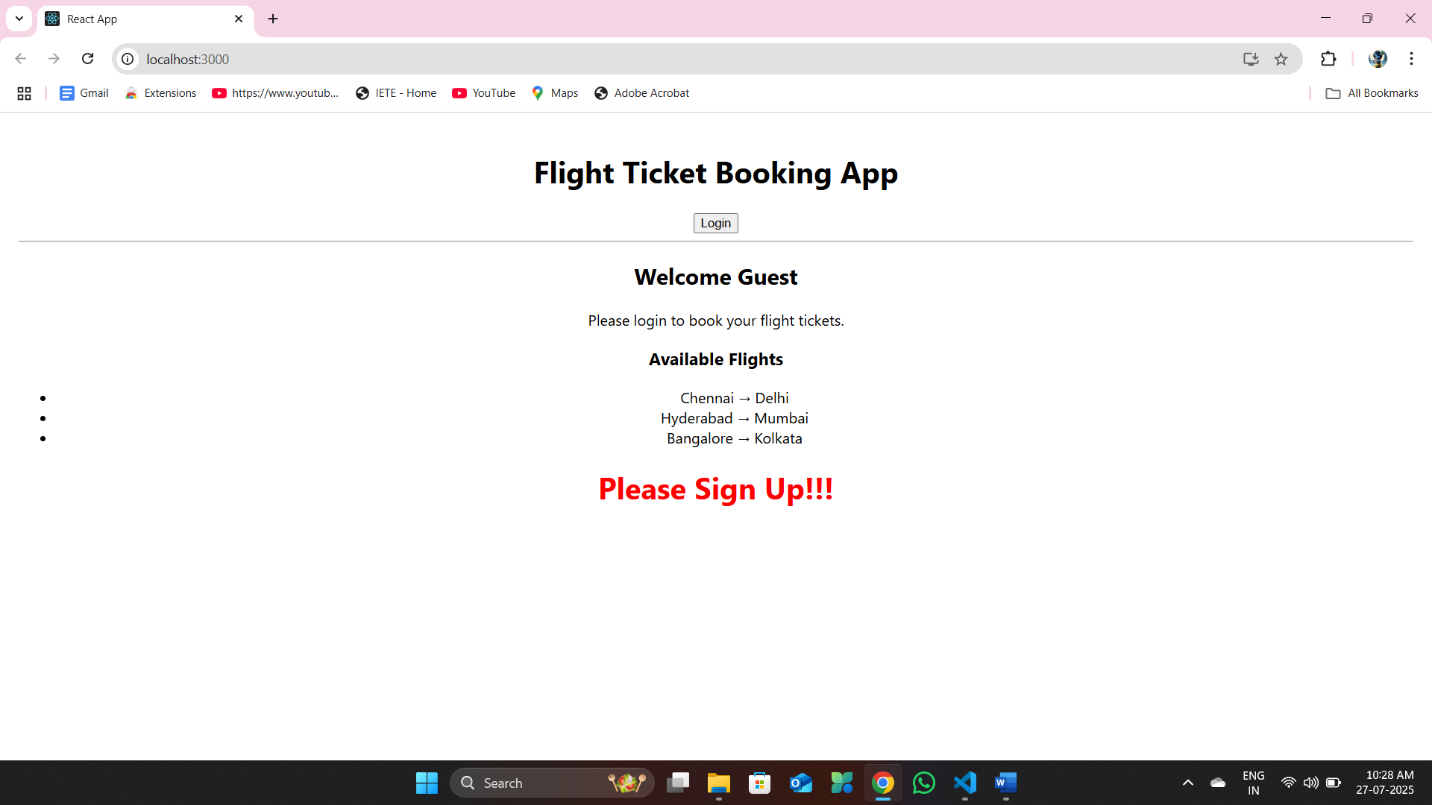
  );

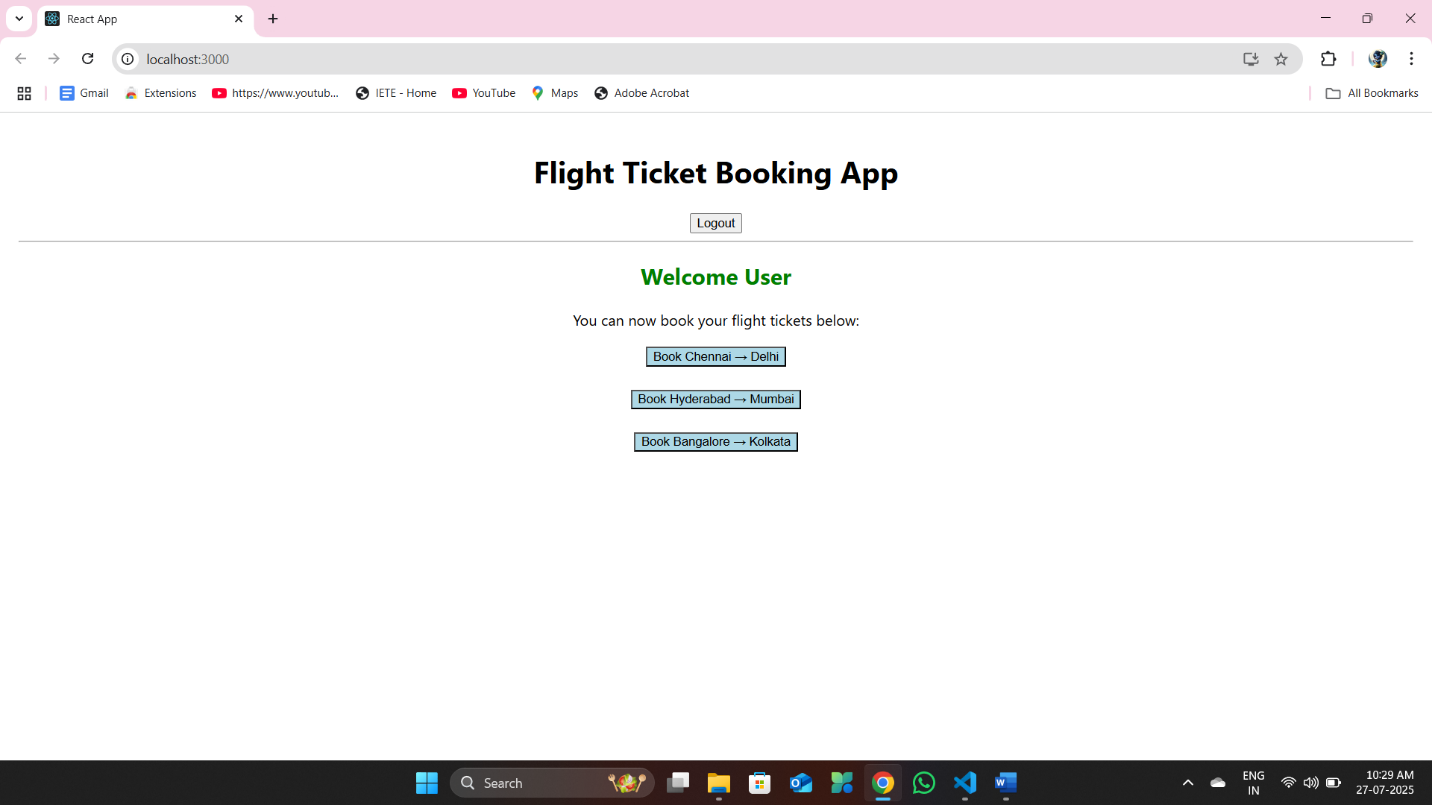
}

export default App;

**Output:**

**npm start**





**EXERCISE 5: Create a React App named “bloggerapp” in with 3 components.**

* Book Details
* Blog Details
* Course Details

Implement this with as many ways possible of Conditional Rendering.

**npx create-react-app bloggerapp**

**D:\CTS\REACT JS\bloggerapp\src\BlogDetails.js**

import React from 'react';

function BlogDetails() {

    const blogs = [

        {

            title: "React Learning",

            author: "Stephen Biz",

            content: "Welcome to learning React!"

        },

        {

            title: "Installation",

            author: "Schewzdenier",

            content: "You can install React from npm."

        }

    ];

    return (

        <div className="column">

            <h2>Blog Details</h2>

            {blogs.map((blog, index) => (

                <div key={index}>

                    <h3>{blog.title}</h3>

                    <strong>{blog.author}</strong>

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

                </div>

            ))}

        </div>

    );

}

export default BlogDetails;

**D:\CTS\REACT JS\bloggerapp\src\BookDetails.js**

import React from 'react';

function BookDetails() {

    const books = [

        { title: "Master React", price: "670" },

        { title: "Deep Dive into Angular 11", price: "800" },

        { title: "Mongo Essentials", price: "450" },

    ];

    return (

        <div className="column">

            <h2>Book Details</h2>

            {books.map((book, index) => (

                <div key={index}>

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

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

                </div>

            ))}

        </div>

    );

}

export default BookDetails;

**D:\CTS\REACT JS\bloggerapp\src\CourseDetails.js**

import React from 'react';

function CourseDetails() {

    const courses = [

        { name: "Angular", date: "4/5/2021" },

        { name: "React", date: "6/3/20201" },

    ];

    return (

        <div className="column">

            <h2>Course Details</h2>

            {courses.map((course, index) => (

                <div key={index}>

                    <h3>{course.name}</h3>

                    <p>{course.date}</p>

                </div>

            ))}

        </div>

    );

}

export default CourseDetails;

**D:\CTS\REACT JS\bloggerapp\src\App.js**

import React from 'react';

import './App.css';

import CourseDetails from './CourseDetails';

import BookDetails from './BookDetails';

import BlogDetails from './BlogDetails';

function App() {

  return (

    <div className="container">

      <CourseDetails />

      <BookDetails />

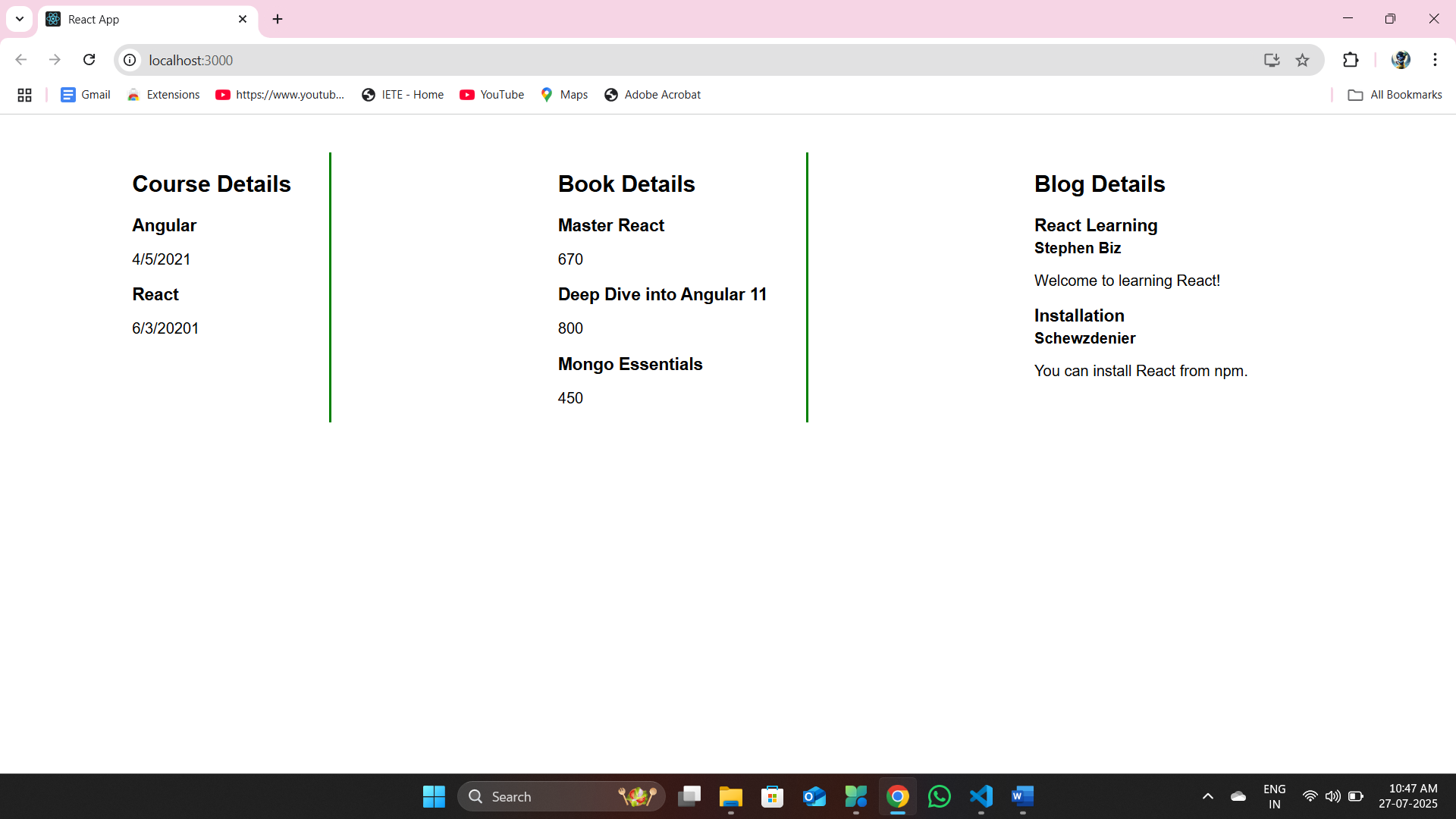
      <BlogDetails />

    </div>

  );

}

export default App;



**EXERCISE 6: Developers of Apps Centric Solutions have created an employee management application which supports light and dark themes for the buttons. The current solution uses the react state and props to provide the theme name to be used from App component to Employee List component and from there to Employee Card component. Quality assurance team analyzed the solutions and found the technique being used to be a substandard one. React architect suggested to use the react context API to share the theme name with nested child components instead of passing them down using props from the parent component.**

**You are assigned the task of converting the application form props only to React Context API.**

Application can be downloaded from below



* Unzip the application and open it using VS Code
* Go to terminal and execute *npm install* command to restore all the node modules
* Run the application once to see the output. Use npm start command.
* Explore the components present in **App.js**, **EmployeesList.js** and **EmployeeCard.js** files.
* Create a new file with the name as **ThemeContext.js**. Define a new context in the file with the name as ThemeContext and assign it a default value of ‘light’ and export it as default form the module.
* Open App component present in **App.js** file.
  1. Import the ThemeContext in App component.
  2. Define the theme context provider to be the entire JSX of the App component.
  3. Assign the value for the theme provider from the state of the component.
  4. Modify the call to EmployeeList component so that theme name is no longer passed as props.
* Go to EmployeeList component present in **EmployeeList.js** file and modify it so that theme name is not passed explicitly to its child component.
* Go to **EmployeeCard** component inside **EmployeeCard.js** file
  1. Import the ThemeContext into the component file
  2. Retrieve the value of the context with the help of **useContext()** and store it in a variable
  3. Use the variable to pass the className for the buttons.

**D:\CTS\REACT JS\employeesapp\src\App.js**

import logo from './logo.svg';

import './App.css';

import { EmployeesData } from './Employee';

import EmployeesList from './EmployeesList';

import { useState } from 'react';

import ThemeContext from './ThemeContext';

function App() {

  const Employees = EmployeesData;

  const [theme, setTheme] = useState('light');

  return (

    <ThemeContext.Provider value={theme}>

      <>

        <div>

          <label>SELECT A THEME </label>

          <select onChange={(e) => setTheme(e.target.value)}>

            <option value='light'>Light</option>

            <option value='dark'>Dark</option>

          </select>

        </div>

        <EmployeesList theme={theme} employees={Employees} />

      </>

    </ThemeContext.Provider>

  );

}

export default App;

**D:\CTS\REACT JS\employeesapp\src\EmployeesList.js**

import EmployeeCard from './EmployeeCard';

import React from 'react';

function EmployeesList(props) {

    return (

        <div>

            <h1>Employees List</h1>

            {

                props.employees.map(employee =>

                    <EmployeeCard employee={employee} key={employee.id} />

                )}

        </div>

    );

}

export default EmployeesList;

**D:\CTS\REACT JS\employeesapp\src\EmployeeCard.js**

import Styles from './EmployeeCard.module.css';

import React, { useContext } from 'react';

import ThemeContext from './ThemeContext';

function EmployeeCard(props) {

    const theme = useContext(ThemeContext);

    return (

        <div className={Styles.Card}>

            <h3>{props.employee.name}</h3>

            <p>{props.employee.email}</p>

            <p>{props.employee.phone}</p>

            <p>

                <a href="#" className={theme}>Edit</a>

                <a href="#" className={theme}>Delete</a>

            </p>

        </div>

    );

}

export default EmployeeCard;

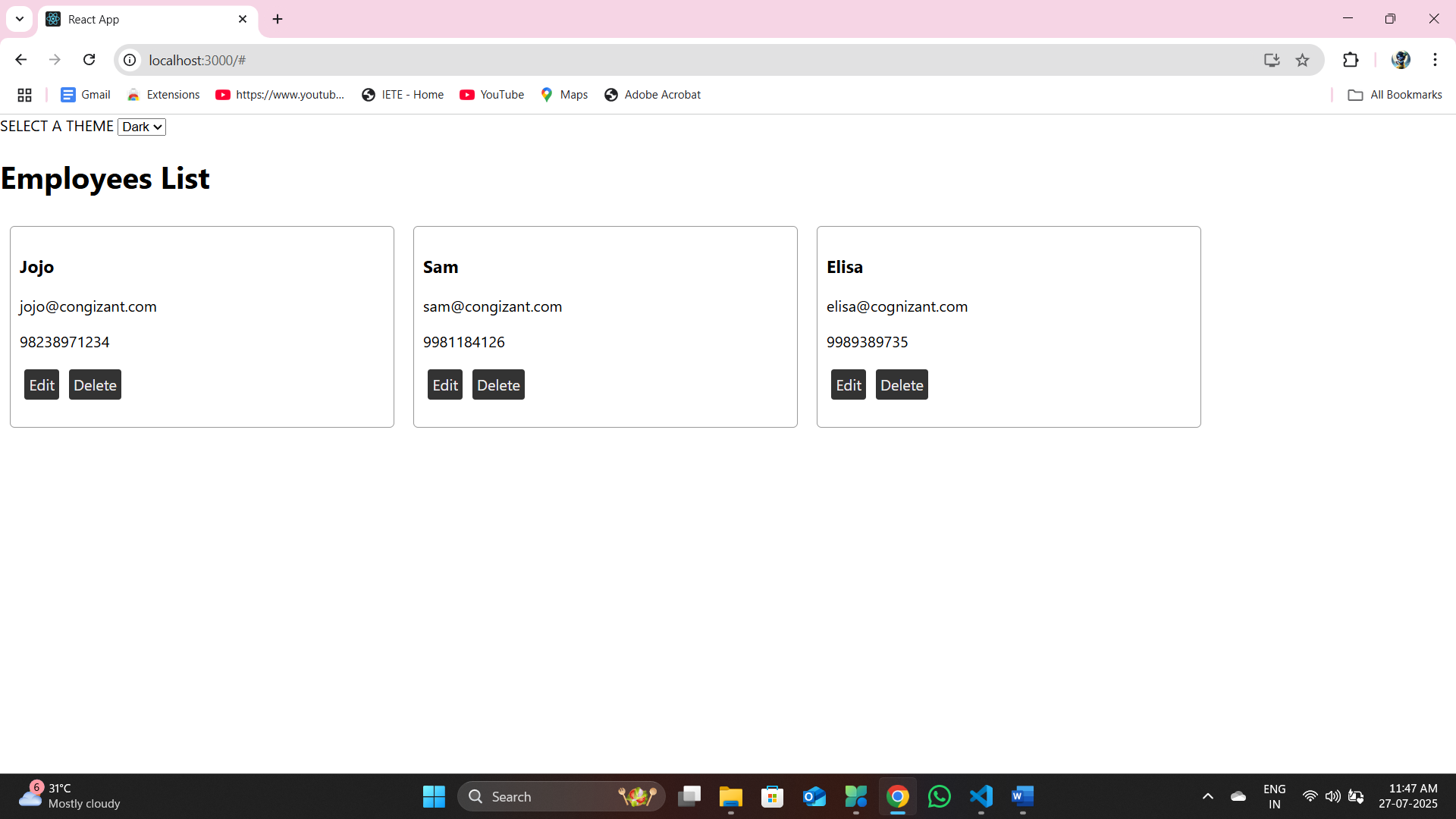
**D:\CTS\REACT JS\employeesapp\src\ThemeContext.js**

import React from 'react';

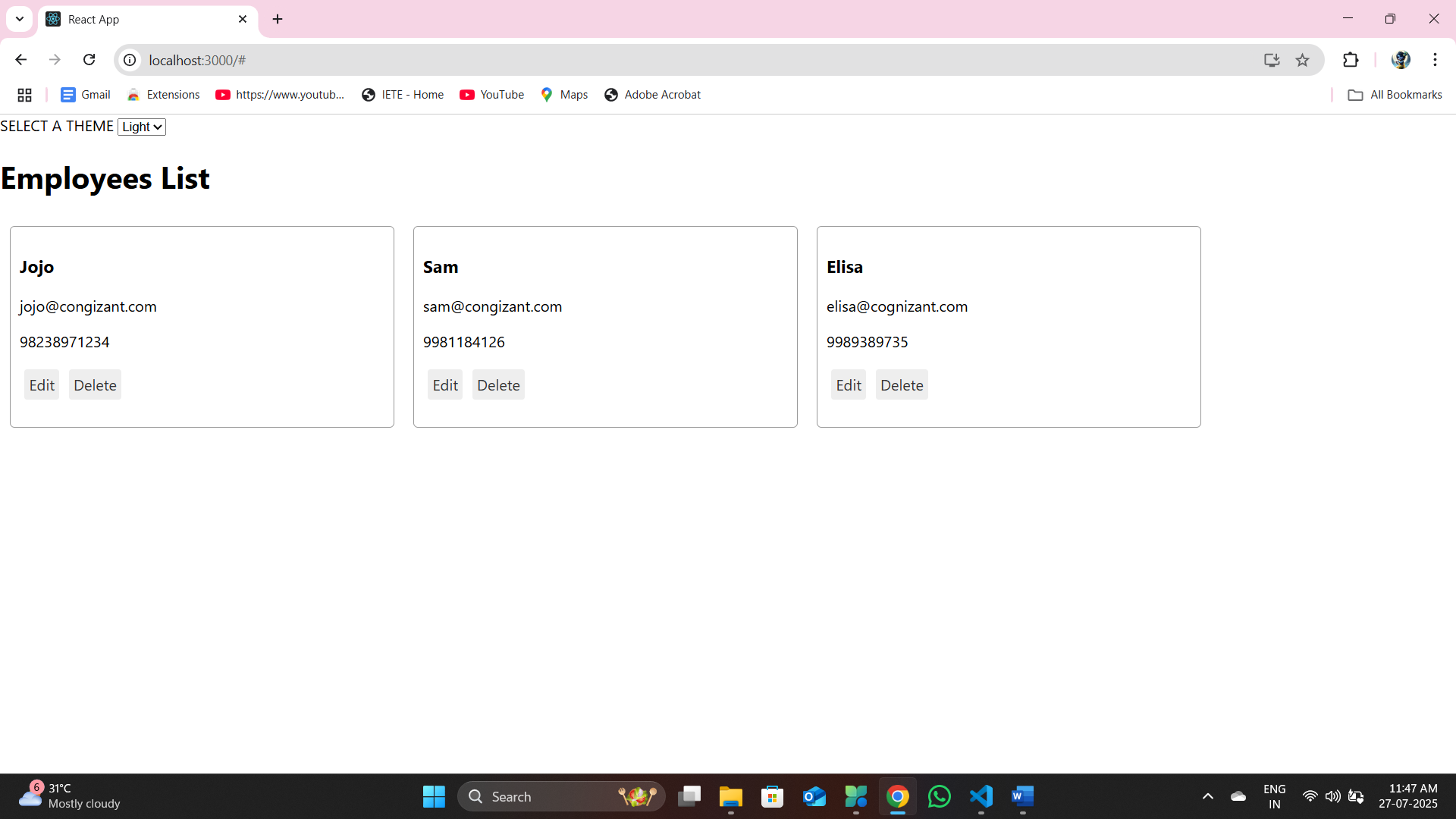
const ThemeContext = React.createContext('light');

export default ThemeContext;

**When toggled to ‘DARK’**



**When toggled to ‘LIGHT’**



**EXERCISE 7: Create a React App named “ticketraisingapp” which will help to raise a complaint and get it resolved.**

**npx create-react-app ticketraisingapp**

**Create a component named “ComplaintRegister” with a form containing a textbox to enter the employee name and a textarea to enter the complaint. Use “handleSubmit” event of the button to submit the complaint and generate a Reference number for further follow ups in the alert box.**

**D:\CTS\REACT JS\ticketraisingapp\src\App.js**

import React from 'react';

import ComplaintRegister from './ComplaintRegister';

function App() {

  return (

    <div className="App">

      <ComplaintRegister />

    </div>

  );

}

export default App;

**D:\CTS\REACT JS\ticketraisingapp\src\ComplaintRegister.js**

import React, { useState } from 'react';

const ComplaintRegister = () => {

    const [employeeName, setEmployeeName] = useState('');

    const [complaint, setComplaint] = useState('');

    const handleSubmit = (e) => {

        e.preventDefault();

        if (employeeName.trim() === '' || complaint.trim() === '') {

            alert("Please fill in all the fields.");

            return;

        }

        const transactionId = Math.floor(Math.random() \* 100); // 0–99

        alert(

            `Thanks ${employeeName}\nYour Complaint was Submitted.\nTransaction ID is: ${transactionId}`

        );

        setEmployeeName('');

        setComplaint('');

    };

    return (

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

            <h2 style={{ color: 'red', fontWeight: 'bold' }}>Register your complaints here!!!</h2>

            <form onSubmit={handleSubmit} style={{ display: 'inline-block', marginTop: '20px' }}>

                <div style={{ marginBottom: '10px', textAlign: 'left' }}>

                    <label>Name: </label>

                    <input

                        type="text"

                        value={employeeName}

                        onChange={(e) => setEmployeeName(e.target.value)}

                        style={{ marginLeft: '10px' }}

                    />

                </div>

                <div style={{ marginBottom: '10px', textAlign: 'left' }}>

                    <label>Complaint: </label>

                    <textarea

                        value={complaint}

                        onChange={(e) => setComplaint(e.target.value)}

                        rows="2"

                        cols="22"

                        style={{ marginLeft: '10px', verticalAlign: 'top' }}

                    ></textarea>

                </div>

                <div style={{ textAlign: 'center' }}>

                    <button type="submit">Submit</button>

                </div>

            </form>

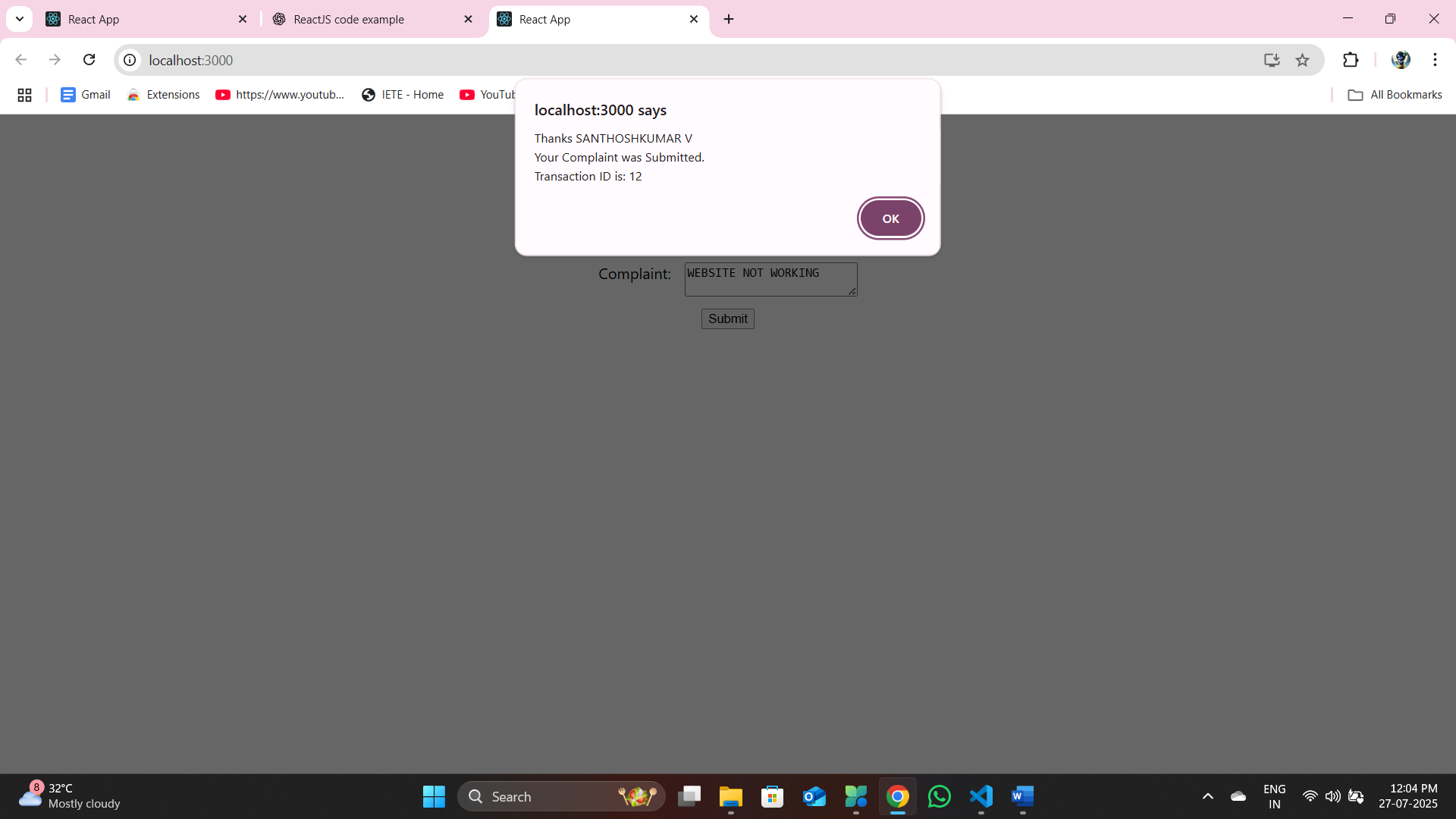
        </div>

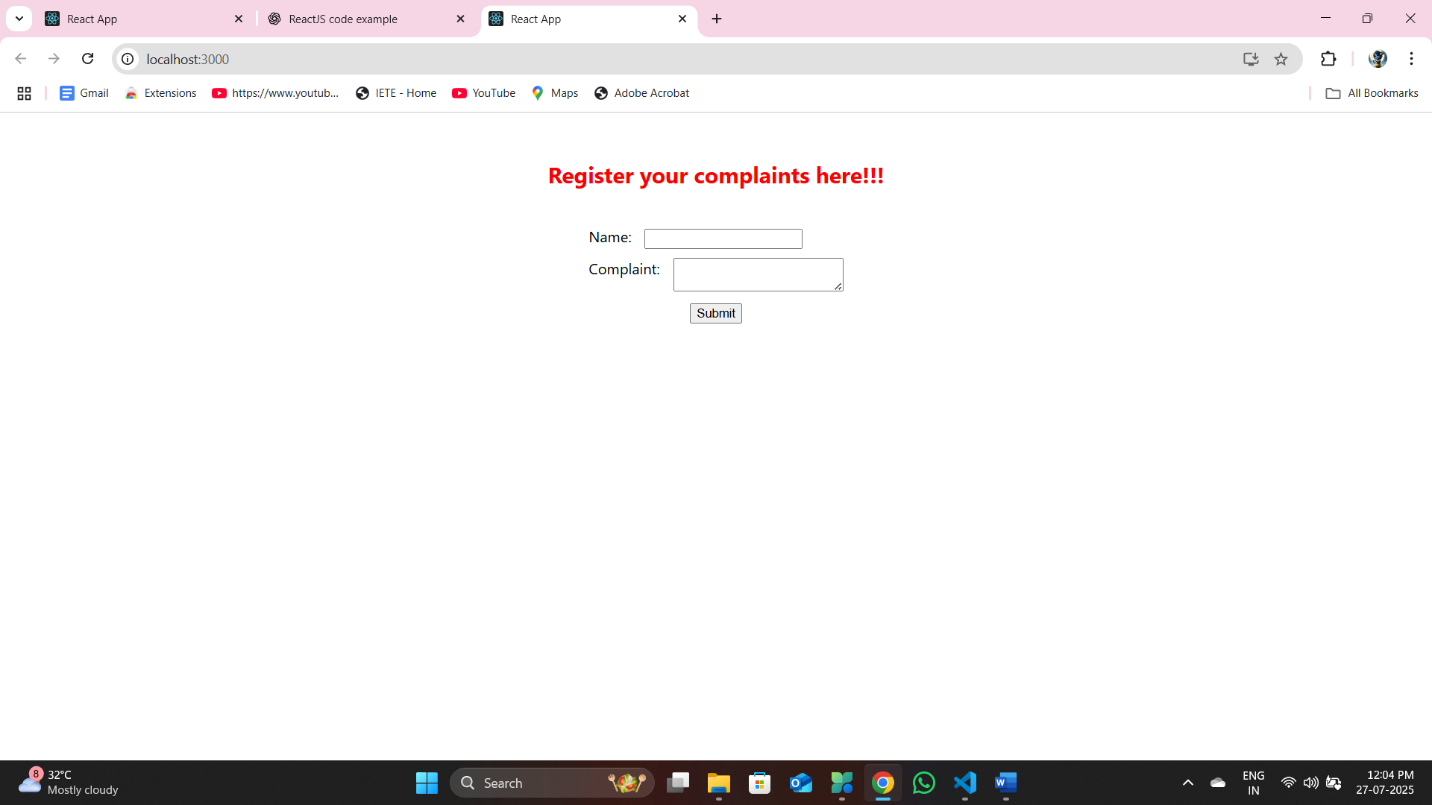
    );

};

export default ComplaintRegister;

**Output :**





**EXERCISE 8: Create a React App named “mailregisterapp” which will have a component named “register.js”. Create a form which accepts the name, email and password and validate the fields as per the following:**

* **Name should have atleast 5 characters**
* **Email should have @ and .**
* **Password should have atleast 8 characters.**

**Ensure that validations are implemented through eventhandle and eventsubmit of a form.**

**npx create-react-app mailregisterapp**

**D:\CTS\REACT JS\mailregisterapp\src\App.js**

import React from 'react';

import Register from './Register';

function App() {

  return (

    <div className="App">

      <Register />

    </div>

  );

}

export default App;

**D:\CTS\REACT JS\mailregisterapp\src\Register.js**

import React, { useState } from 'react';

const Register = () => {

    const [name, setName] = useState('');

    const [email, setEmail] = useState('');

    const [password, setPassword] = useState('');

    const [errors, setErrors] = useState({});

    const validate = () => {

        let tempErrors = {};

        if (name.length < 5) {

            tempErrors.name = "Name must be at least 5 characters.";

        }

        if (!email.includes('@') || !email.includes('.')) {

            tempErrors.email = "Email must contain '@' and '.'";

        }

        if (password.length < 8) {

            tempErrors.password = "Password must be at least 8 characters.";

        }

        setErrors(tempErrors);

        return Object.keys(tempErrors).length === 0;

    };

    const handleSubmit = (e) => {

        e.preventDefault();

        if (validate()) {

            alert(`Registration Successful!\nName: ${name}\nEmail: ${email}`);

            // Reset form

            setName('');

            setEmail('');

            setPassword('');

            setErrors({});

        }

    };

    return (

        <div style={{ margin: '50px auto', maxWidth: '400px' }}>

            <h2 style={{ textAlign: 'center', color: 'blue' }}>User Registration</h2>

            <form onSubmit={handleSubmit}>

                <div style={{ marginBottom: '10px' }}>

                    <label>Name:</label><br />

                    <input

                        type="text"

                        value={name}

                        onChange={(e) => setName(e.target.value)}

                        style={{ width: '100%', padding: '8px' }}

                    />

                    {errors.name && <div style={{ color: 'red' }}>{errors.name}</div>}

                </div>

                <div style={{ marginBottom: '10px' }}>

                    <label>Email:</label><br />

                    <input

                        type="email"

                        value={email}

                        onChange={(e) => setEmail(e.target.value)}

                        style={{ width: '100%', padding: '8px' }}

                    />

                    {errors.email && <div style={{ color: 'red' }}>{errors.email}</div>}

                </div>

                <div style={{ marginBottom: '10px' }}>

                    <label>Password:</label><br />

                    <input

                        type="password"

                        value={password}

                        onChange={(e) => setPassword(e.target.value)}

                        style={{ width: '100%', padding: '8px' }}

                    />

                    {errors.password && <div style={{ color: 'red' }}>{errors.password}</div>}

                </div>

                <div style={{ textAlign: 'center' }}>

                    <button type="submit" style={{ padding: '10px 20px', backgroundColor: 'green', color: 'white' }}>

                        Register

                    </button>

                </div>

            </form>

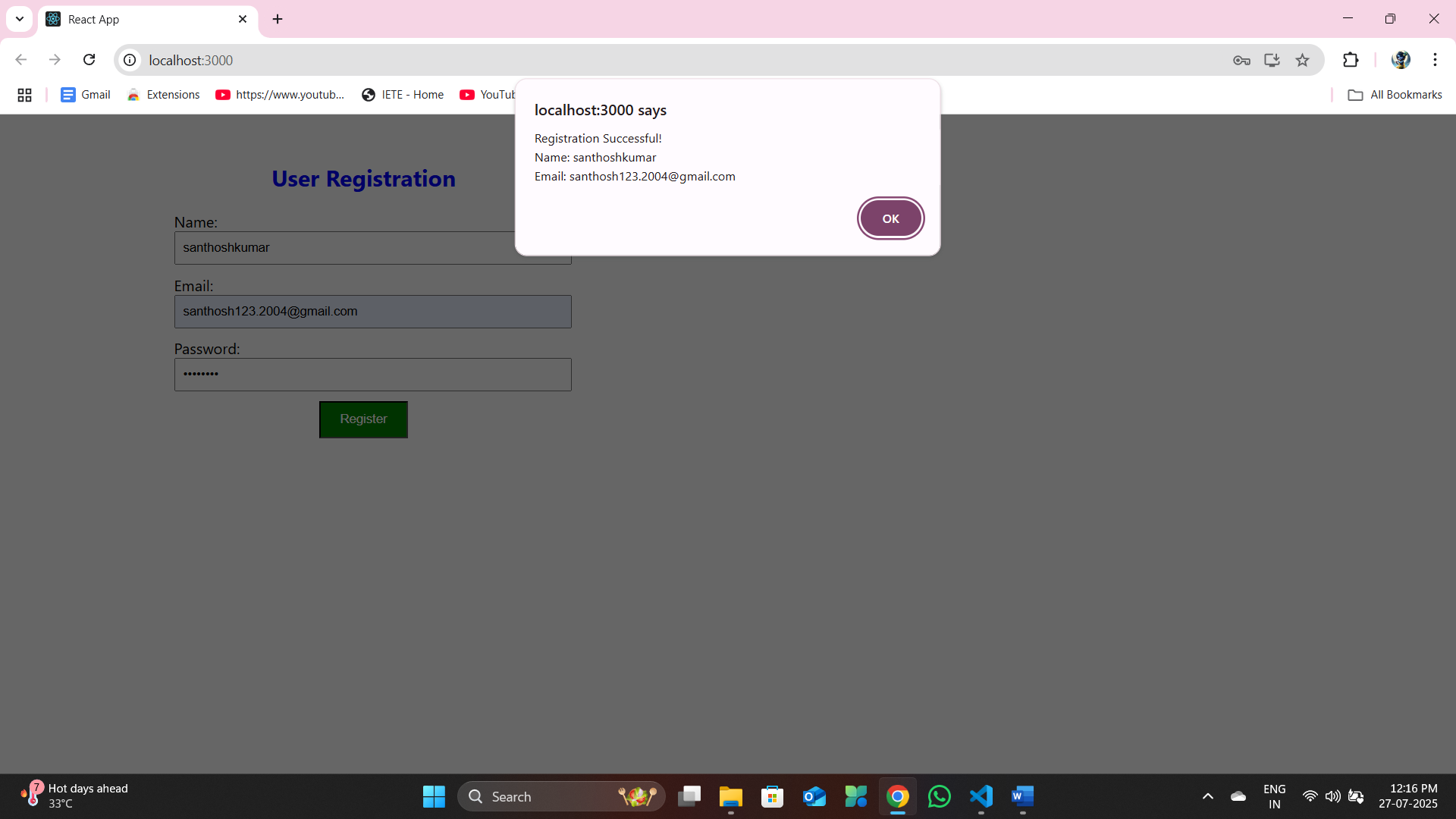
        </div>

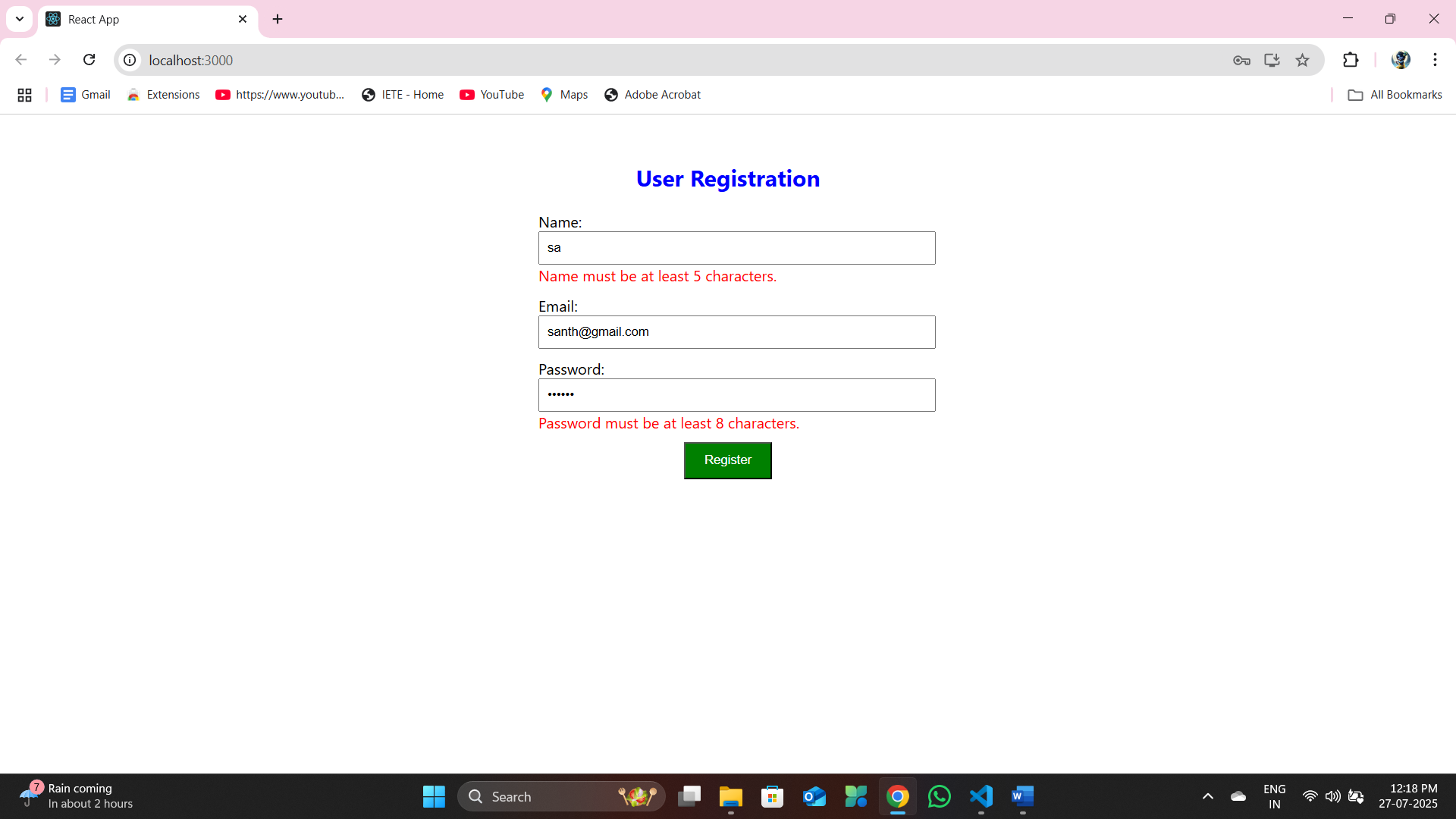
    );

};

export default Register;

**Output :**





**EXERCISE 9: Create a React Application “fetchuserapp” which will retrieve the user details from** [**https://api.randomuser.me/**](https://api.randomuser.me/) **and display the title, firstname and image of a user.**

**Create a component named “Getuser” and in the asynchronous method “ComponentDidMount ()” invoke the URL using fetch method and the response can be displayed in the render method of the**

**D:\CTS\REACT JS\fetchuserapp\src\App.js**

// src/App.js

import React from 'react';

import './App.css';

import Getuser from './Getuser';

function App() {

  return (

    <div className="App">

      <h1>Fetch User App</h1>

      <Getuser />

    </div>

  );

}

export default App;

**D:\CTS\REACT JS\fetchuserapp\src\Getuser.js**

// src/Getuser.js

import React, { Component } from 'react';

class Getuser extends Component {

    constructor(props) {

        super(props);

        this.state = {

            user: null,

            isLoading: true,

            error: null,

        };

    }

    async componentDidMount() {

        try {

            const response = await fetch('https://api.randomuser.me/');

            const data = await response.json();

            this.setState({ user: data.results[0], isLoading: false });

        } catch (error) {

            this.setState({ error: 'Failed to fetch user', isLoading: false });

        }

    }

    render() {

        const { user, isLoading, error } = this.state;

        if (isLoading) return <p>Loading user...</p>;

        if (error) return <p>{error}</p>;

        return (

            <div>

                <h2>User Information</h2>

                <p>Title: {user.name.title}</p>

                <p>First Name: {user.name.first}</p>

                <img src={user.picture.large} alt="User" />

            </div>

        );

    }

}

export default Getuser;

**Output:**

