**8. ReactJS-HOL**

**CountPeople.js**

import React, { Component } from 'react';

class CountPeople extends Component {

constructor(props) {

super(props);

this.state = {

entryCount: 0,

exitCount: 0

};

}

updateEntry = () => {

this.setState((prevState) => ({

entryCount: prevState.entryCount + 1

}));

};

updateExit = () => {

this.setState((prevState) => ({

exitCount: prevState.exitCount + 1

}));

};

render() {

const { entryCount, exitCount } = this.state;

const buttonStyle = {

backgroundColor: 'lightgreen',

padding: '5px 10px',

border: '1px solid green',

marginRight: '10px',

cursor: 'pointer'

};

const boxStyle = {

display: 'inline-block',

margin: '20px',

textAlign: 'center'

};

return (

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

<div style={boxStyle}>

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

<span>{entryCount} People Entered!!!</span>

</div>

<div style={boxStyle}>

<button style={buttonStyle} onClick={this.updateExit}>Exit</button>

<span>{exitCount} People Left!!!</span>

</div>

</div>

);

}

}

export default CountPeople;

**App.js**

import React from 'react';

import CountPeople from './CountPeople';

function App() {

return (

<div>

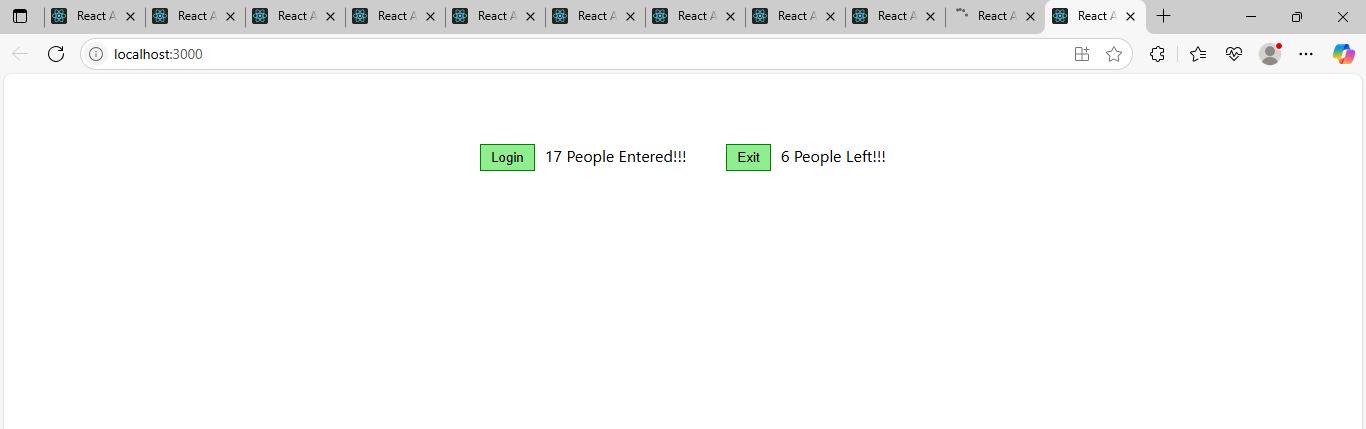
<CountPeople />

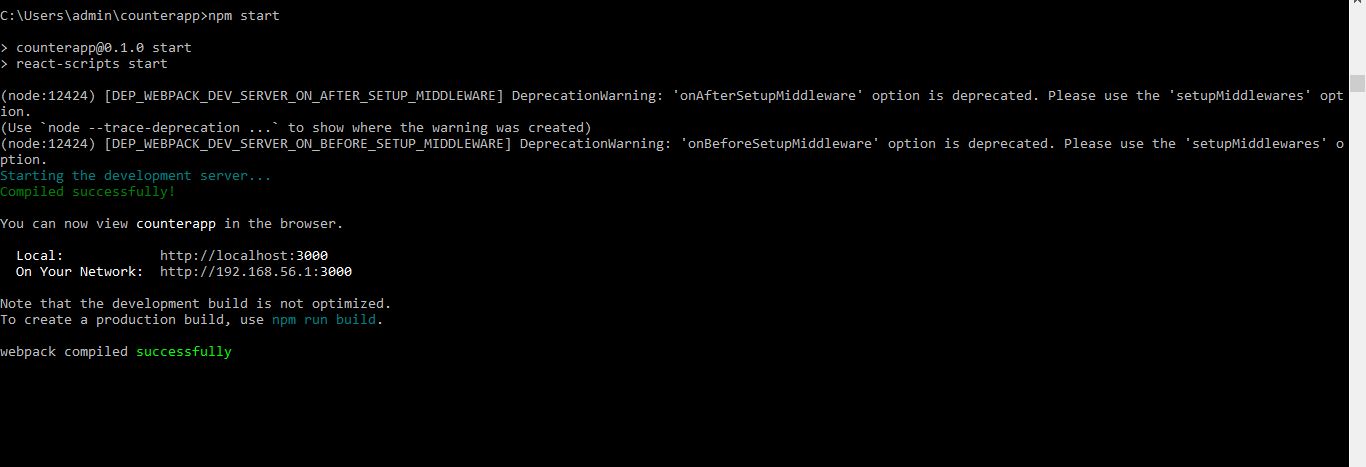
</div>

);

}

export default App;





**What is React State?**

In React, **state** refers to a **built-in object** that holds **dynamic data** for a component. State enables a component to keep track of information that **can change over time**, and when it does, the component **re-renders** to reflect those changes on the screen.

**Why is State Important?**

* React components are **reactive**—they update automatically when their state changes.
* State is used to **control user interactions**, **form inputs**, **counters**, **dynamic content**, etc.
* It allows for **interactive UIs** without manual DOM manipulation.

**Key Points:**

* State is **local** to a component.
* State is created using:
  + this.state in class components
  + useState() hook in functional components
* State should **never be modified directly**. Use setState() in class components and the setter function from useState() in functional components.