**Ticket Booking App**

**REACT**

Vaishnavi

31/7/2025

Conditional rendering in React allows components to render different UI outputs based on certain conditions, similar to using if statements in JavaScript. This feature is useful when building interactive applications that change content based on user actions, like login/logout functionality.

In this lab, we will create a **Ticket Booking Application** named ticketbookingapp.

**Guest users** can only view flight details.

**Logged-in users** can book tickets.

The interface will switch between **Guest Page** and **User Page** based on the login status, using **conditional rendering techniques**.

**Objective:**

Conditional rendering in React is the technique of displaying UI elements based on specific conditions or logic. It works similarly to regular JavaScript conditions (like if, ternary ? :, or logical &&) and helps React decide what to display depending on the component's state or received props. This is commonly used to show different views for users (like login vs. dashboard), handle loading states, or toggle visibility.  
**Example:**

{isLoggedIn ? <Dashboard /> : <Login />}

Here, Dashboard is shown if isLoggedIn is true; otherwise, Login is displayed.

In React, element variables are used to store JSX content in a variable before returning it inside the component. This helps simplify the render logic, especially when there’s a need to conditionally display certain parts of the UI. Instead of writing logic inside JSX, it's handled beforehand using variables.  
**Example:**

let message;

if (isConnected) {

message = <p>You are connected</p>;

} else {

message = <p>You are not connected</p>;

}

return <div>{message}</div>;

This method keeps the JSX clean and improves readability when the logic becomes complex.

React provides different ways to stop a component from rendering when it's not needed. One approach is to return null from a component — React will not render anything in this case. Another way is using conditional logic in JSX so that the component only renders under certain conditions.  
**Common ways include:**

* **short-circuit logic:**

{isVisible && <MyComponent />}

* **Excluding it in the parent JSX entirely based on logic.**  
  This approach improves performance and avoids rendering unnecessary components in the UI.

**Implementation:**

**Step 1: Create a React App**

****

**Step 2: Use the Component in App.js**

import React, { useState } from 'react';

import GuestPage from './components/GuestPage';

import UserPage from './components/UserPage';

import './App.css';

function App() {

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

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

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

  return (

    <div className="App">

      <header>

        <h1>✈️ Ticket Booking App</h1>

        {isLoggedIn ? (

          <button onClick={handleLogout} className="logout-btn">Logout</button>

        ) : (

          <button onClick={handleLogin} className="login-btn">Login</button>

        )}

      </header>

      {isLoggedIn ? <UserPage /> : <GuestPage />}

    </div>

  );

}

export default App;

**UserPage.jsx**

import React from 'react';

import FlightDetails from './FlightDetails';

function UserPage() {

  return (

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

      <h2>Welcome, User!</h2>

      <FlightDetails showBooking={true} />

    </div>

  );

}

export default UserPage;

**FlightDetails.jsx**

import React from 'react';

import '../App.css';

const flights = [

  { id: 1, name: 'Vistara UK-101', from: 'Pune', to: 'Goa', price: 3200, time: '08:30 AM' },

  { id: 2, name: 'Akasa Air QP-212', from: 'Delhi', to: 'Ahmedabad', price: 3900, time: '10:15 AM' },

  { id: 3, name: 'GoFirst G8-525', from: 'Lucknow', to: 'Jaipur', price: 4100, time: '01:45 PM' }

];

function FlightDetails({ showBooking }) {

  return (

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

      <h3>🛫 Flights for Your Friend</h3>

      <div className="flight-list">

        {flights.map(flight => (

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

            <h4>{flight.name}</h4>

            <p>{flight.from} ➡️ {flight.to}</p>

            <p>Departure: {flight.time}</p>

            <p>Price: ₹{flight.price}</p>

            {showBooking && <button className="book-btn">Book Now</button>}

          </div>

        ))}

      </div>

    </div>

  );

}

export default FlightDetails;

**GuestPage.jsx**

import React from 'react';

import FlightDetails from './FlightDetails';

function GuestPage() {

  return (

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

      <h2>Welcome Guest!</h2>

      <p>Please log in to book tickets.</p>

      <FlightDetails />

    </div>

  );

}

export default GuestPage;

**App.css**

.flight-list {

  display: flex;

  gap: 20px;

  flex-wrap: wrap;

  justify-content: center;

  padding: 10px;

}

.flight-card {

  background-color: #f0f8ff;

  border: 2px solid #008cba;

  border-radius: 12px;

  padding: 16px;

  width: 230px;

  text-align: center;

  box-shadow: 2px 2px 10px rgba(0,0,0,0.1);

  transition: transform 0.2s;

}

.flight-card:hover {

  transform: scale(1.05);

}

.book-btn {

  margin-top: 10px;

  background-color: #007acc;

  color: white;

  border: none;

  padding: 8px 12px;

  border-radius: 8px;

  cursor: pointer;

}

.book-btn:hover {

  background-color: #005f99;

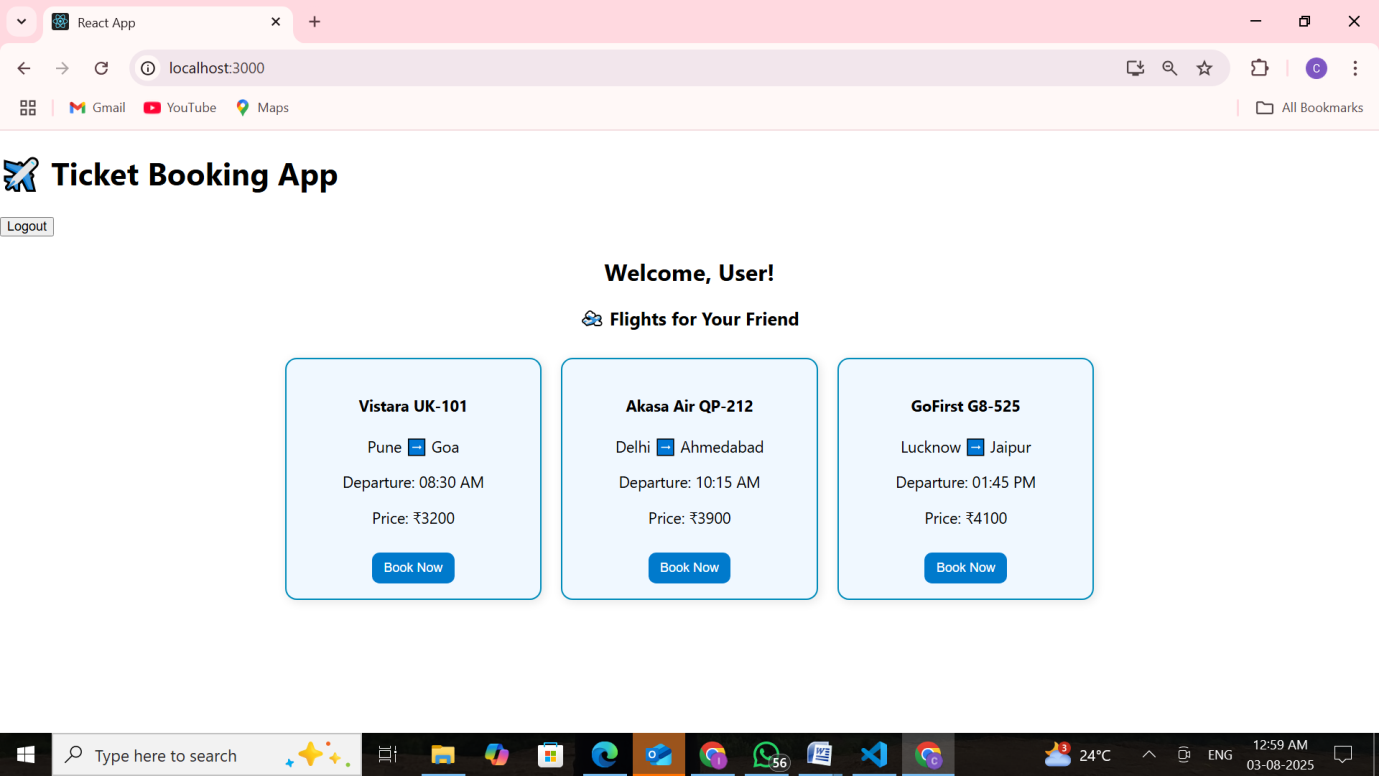
}

**Step 4: Run the App**

****

**Output:**

After Login:



After Logout:

