**React**

1. **Cricketapp**

**//index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>React App</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**//App.css**

.App {

text-align: center;

}

.App-logo {

height: 40vmin;

pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

.App-logo {

animation: App-logo-spin infinite 20s linear;

}

}

.App-header {

background-color: #282c34;

min-height: 100vh;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

font-size: calc(10px + 2vmin);

color: white;

}

.App-link {

color: #61dafb;

}

@keyframes App-logo-spin {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

**//App.js**

import React, { useState } from 'react';

import './App.css';

import ListofPlayers from './ListofPlayers';

import IndianPlayers from './IndianPlayers';

function App() {

// Flag variable to toggle between components

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

return (

<div className="App" style={{ backgroundColor: '#f8f9fa', minHeight: '100vh', padding: '20px' }}>

<div style={{ textAlign: 'center', marginBottom: '30px' }}>

<h1 style={{ color: '#2c3e50', fontSize: '2.5rem', marginBottom: '20px' }}>

🏏 Cricket App Dashboard

</h1>

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

<button

onClick={() => setFlag(true)}

style={{

backgroundColor: flag ? '#3498db' : '#bdc3c7',

color: 'white',

border: 'none',

padding: '12px 24px',

margin: '0 10px',

borderRadius: '25px',

cursor: 'pointer',

fontSize: '16px',

fontWeight: 'bold',

transition: 'all 0.3s ease'

}}

>

Show List of Players

</button>

<button

onClick={() => setFlag(false)}

style={{

backgroundColor: !flag ? '#e74c3c' : '#bdc3c7',

color: 'white',

border: 'none',

padding: '12px 24px',

margin: '0 10px',

borderRadius: '25px',

cursor: 'pointer',

fontSize: '16px',

fontWeight: 'bold',

transition: 'all 0.3s ease'

}}

>

Show Indian Players

</button>

</div>

<div style={{

backgroundColor: '#34495e',

color: 'white',

padding: '10px 20px',

borderRadius: '20px',

display: 'inline-block',

fontSize: '18px',

fontWeight: 'bold'

}}>

Current View: {flag ? 'List of Players (Flag = true)' : 'Indian Players (Flag = false)'}

</div>

</div>

{/\* Simple if-else using flag variable \*/}

<div>

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

</div>

</div>

);

}

export default App;

**//App.test.js**

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

render(<App />);

const linkElement = screen.getByText(/learn react/i);

expect(linkElement).toBeInTheDocument();

});

**//IndianPlayers.js**

import React from 'react';

function IndianPlayers() {

// Team players for destructuring

const teamPlayers = [

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

"Rishabh Pant", "Ravindra Jadeja", "Jasprit Bumrah", "Mohammed Shami",

"Yuzvendra Chahal", "Bhuvneshwar Kumar", "Shikhar Dhawan"

];

// Destructuring to get odd and even positioned players

const [first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh] = teamPlayers;

// Odd positioned players (1st, 3rd, 5th, etc.)

const oddTeamPlayers = [first, third, fifth, seventh, ninth, eleventh];

// Even positioned players (2nd, 4th, 6th, etc.)

const evenTeamPlayers = [second, fourth, sixth, eighth, tenth];

// T20 players array

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

// Ranji Trophy players array

const RanjiTrophyPlayers = ["Prithvi Shaw", "Devdutt Padikkal", "Sarfaraz Khan", "Ishan Kishan", "Washington Sundar"];

// Merge the two arrays using ES6 spread operator

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

return (

<div style={{ padding: '20px', backgroundColor: '#fff8dc', margin: '10px', borderRadius: '8px' }}>

<h2 style={{ color: '#2c3e50', textAlign: 'center' }}>🇮🇳 Indian Players</h2>

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

<h3 style={{ color: '#e67e22' }}>Odd Team Players (1st, 3rd, 5th positions...):</h3>

<div style={{ display: 'flex', flexWrap: 'wrap', gap: '10px' }}>

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

<div key={index} style={{

backgroundColor: '#fff',

padding: '10px 15px',

borderRadius: '20px',

border: '2px solid #f39c12',

fontWeight: 'bold',

color: '#d35400'

}}>

{player}

</div>

))}

</div>

</div>

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

<h3 style={{ color: '#8e44ad' }}>Even Team Players (2nd, 4th, 6th positions...):</h3>

<div style={{ display: 'flex', flexWrap: 'wrap', gap: '10px' }}>

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

<div key={index} style={{

backgroundColor: '#fff',

padding: '10px 15px',

borderRadius: '20px',

border: '2px solid #9b59b6',

fontWeight: 'bold',

color: '#6c3483'

}}>

{player}

</div>

))}

</div>

</div>

<div>

<h3 style={{ color: '#16a085' }}>Merged Players (T20 + Ranji Trophy):</h3>

<div style={{ marginBottom: '15px' }}>

<h4 style={{ color: '#2980b9' }}>T20 Players:</h4>

<div style={{ display: 'flex', flexWrap: 'wrap', gap: '8px', marginBottom: '10px' }}>

{T20players.map((player, index) => (

<span key={index} style={{

backgroundColor: '#3498db',

color: 'white',

padding: '5px 10px',

borderRadius: '15px',

fontSize: '14px'

}}>

{player}

</span>

))}

</div>

<h4 style={{ color: '#27ae60' }}>Ranji Trophy Players:</h4>

<div style={{ display: 'flex', flexWrap: 'wrap', gap: '8px', marginBottom: '15px' }}>

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

<span key={index} style={{

backgroundColor: '#2ecc71',

color: 'white',

padding: '5px 10px',

borderRadius: '15px',

fontSize: '14px'

}}>

{player}

</span>

))}

</div>

</div>

<h4 style={{ color: '#c0392b' }}>All Merged Players:</h4>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(150px, 1fr))', gap: '10px' }}>

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

<div key={index} style={{

backgroundColor: index < T20players.length ? '#ebf3fd' : '#eafaf1',

padding: '10px',

borderRadius: '5px',

border: index < T20players.length ? '1px solid #3498db' : '1px solid #2ecc71',

textAlign: 'center',

fontWeight: 'bold'

}}>

{player}

<br />

<small style={{ color: '#7f8c8d' }}>

{index < T20players.length ? 'T20' : 'Ranji'}

</small>

</div>

))}

</div>

</div>

</div>

);

}

export default IndianPlayers;

**//ListofPlayers.js**

import React from 'react';

function ListofPlayers() {

// Array with 11 players and their scores using ES6 map feature

const players = [

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

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

{ name: "KL Rahul", score: 65 },

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

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

{ name: "Ravindra Jadeja", score: 45 },

{ name: "Jasprit Bumrah", score: 25 },

{ name: "Mohammed Shami", score: 30 },

{ name: "Yuzvendra Chahal", score: 15 },

{ name: "Bhuvneshwar Kumar", score: 40 },

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

];

// Filter players with scores below 70 using arrow functions

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

return (

<div style={{ padding: '20px', backgroundColor: '#f0f8ff', margin: '10px', borderRadius: '8px' }}>

<h2 style={{ color: '#2c3e50', textAlign: 'center' }}>🏏 List of Players</h2>

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

<h3 style={{ color: '#27ae60' }}>All Players:</h3>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(200px, 1fr))', gap: '10px' }}>

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

<div key={index} style={{

backgroundColor: '#fff',

padding: '10px',

borderRadius: '5px',

border: '1px solid #ddd',

textAlign: 'center'

}}>

<strong>{player.name}</strong><br />

Score: {player.score}

</div>

))}

</div>

</div>

<div>

<h3 style={{ color: '#e74c3c' }}>Players with Scores Below 70:</h3>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(200px, 1fr))', gap: '10px' }}>

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

<div key={index} style={{

backgroundColor: '#ffe6e6',

padding: '10px',

borderRadius: '5px',

border: '1px solid #ff9999',

textAlign: 'center'

}}>

<strong>{player.name}</strong><br />

Score: {player.score}

</div>

))}

</div>

</div>

</div>

);

}

export default ListofPlayers;

**//index.css**

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

**//index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**//reportWebVital.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

getTTFB(onPerfEntry);

});

}

};

export default reportWebVitals;

**OUTPUT:**

1. **Officespacerentalapp**

**//index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>React App</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**//App.css**

/\* App Container \*/

.App {

max-width: 1200px;

margin: 0 auto;

padding: 20px;

font-family: 'Arial', sans-serif;

background-color: #f8f9fa;

min-height: 100vh;

}

/\* Header Styles \*/

.app-header {

text-align: center;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: white;

padding: 40px 20px;

border-radius: 15px;

margin-bottom: 30px;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

}

.main-heading {

font-size: 3rem;

margin: 0 0 10px 0;

font-weight: bold;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

}

.subtitle {

font-size: 1.2rem;

margin: 0;

opacity: 0.9;

}

/\* Main Content \*/

.main-content {

margin-bottom: 40px;

}

/\* Section Headings \*/

.section-heading {

font-size: 2.2rem;

color: #2c3e50;

text-align: center;

margin: 40px 0 30px 0;

font-weight: bold;

}

/\* Featured Office Section \*/

.featured-office {

margin-bottom: 50px;

}

.office-card.featured {

max-width: 600px;

margin: 0 auto;

background: linear-gradient(135deg, #ffecd2 0%, #fcb69f 100%);

border: 3px solid #f39c12;

transform: scale(1.05);

}

/\* Office Grid \*/

.office-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(350px, 1fr));

gap: 25px;

padding: 20px;

}

/\* Office Card Styles \*/

.office-card {

background: white;

border-radius: 15px;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

overflow: hidden;

transition: all 0.3s ease;

border: 2px solid #e9ecef;

}

.office-card:hover {

transform: translateY(-10px);

box-shadow: 0 15px 40px rgba(0, 0, 0, 0.2);

border-color: #667eea;

}

/\* Office Image \*/

.office-image {

width: 100%;

height: 250px;

object-fit: cover;

border-bottom: 3px solid #eee;

}

/\* Office Details \*/

.office-details {

padding: 25px;

}

.office-name {

font-size: 1.5rem;

color: #2c3e50;

margin: 0 0 15px 0;

font-weight: bold;

}

.office-address {

font-size: 1rem;

color: #7f8c8d;

margin: 10px 0;

line-height: 1.5;

}

.office-rent {

font-size: 1.8rem;

font-weight: bold;

margin: 15px 0;

padding: 10px 15px;

border-radius: 25px;

background-color: #f8f9fa;

text-align: center;

border: 2px solid currentColor;

}

/\* Office Features \*/

.office-features {

margin: 20px 0;

padding: 15px;

background-color: #f8f9fa;

border-radius: 10px;

border-left: 4px solid #667eea;

}

.features-list {

list-style: none;

padding: 0;

margin: 10px 0 0 0;

display: flex;

flex-wrap: wrap;

gap: 8px;

}

.feature-item {

background: linear-gradient(135deg, #667eea, #764ba2);

color: white;

padding: 5px 12px;

border-radius: 20px;

font-size: 0.9rem;

font-weight: 500;

}

/\* Contact Button \*/

.contact-btn {

width: 100%;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: white;

border: none;

padding: 15px 25px;

border-radius: 25px;

font-size: 1.1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

margin-top: 15px;

}

.contact-btn:hover {

transform: translateY(-2px);

box-shadow: 0 8px 20px rgba(102, 126, 234, 0.4);

}

/\* Statistics Section \*/

.statistics {

background: white;

padding: 40px;

border-radius: 15px;

margin: 40px 0;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

}

.stats-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

gap: 20px;

margin-top: 30px;

}

.stat-card {

text-align: center;

padding: 25px;

background: linear-gradient(135deg, #f093fb 0%, #f5576c 100%);

color: white;

border-radius: 15px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

}

.stat-card h3 {

margin: 0 0 15px 0;

font-size: 1.1rem;

}

.stat-number {

font-size: 2.5rem;

font-weight: bold;

margin: 0;

}

.stat-number.red {

color: #ff6b6b;

}

.stat-number.green {

color: #51cf66;

}

/\* Footer \*/

.app-footer {

text-align: center;

padding: 30px;

background: #2c3e50;

color: white;

border-radius: 15px;

margin-top: 40px;

}

/\* Responsive Design \*/

@media (max-width: 768px) {

.App {

padding: 10px;

}

.main-heading {

font-size: 2rem;

}

.office-grid {

grid-template-columns: 1fr;

padding: 10px;

}

.stats-grid {

grid-template-columns: 1fr;

}

}

/\* Color Classes for Rent \*/

.red {

color: #e74c3c !important;

}

.green {

color: #27ae60 !important;

}

**//App.js**

import React from 'react';

import './App.css';

function App() {

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

const sampleOffice = {

name: "Tech Hub Premium",

rent: 75000,

address: "123 Business District, Mumbai, Maharashtra",

image: "https://images.unsplash.com/photo-1497366216548-37526070297c?w=400&h=300&fit=crop"

};

// Create a list of office objects to display more data

const officeSpaces = [

{

id: 1,

name: "Modern Co-Working Space",

rent: 45000,

address: "456 Innovation Street, Bangalore, Karnataka",

image: "https://images.unsplash.com/photo-1497366811353-6870744d04b2?w=400&h=300&fit=crop",

features: ["WiFi", "Conference Room", "Parking"]

},

{

id: 2,

name: "Executive Business Center",

rent: 85000,

address: "789 Corporate Avenue, Delhi, NCR",

image: "https://images.unsplash.com/photo-1497366754035-f200968a6e72?w=400&h=300&fit=crop",

features: ["Reception", "Security", "Cafeteria"]

},

{

id: 3,

name: "Creative Studio Space",

rent: 35000,

address: "321 Art District, Pune, Maharashtra",

image: "https://images.unsplash.com/photo-1497215842964-222b430dc094?w=400&h=300&fit=crop",

features: ["Natural Light", "Open Space", "Kitchen"]

},

{

id: 4,

name: "Premium Office Suite",

rent: 120000,

address: "654 Financial Center, Hyderabad, Telangana",

image: "https://images.unsplash.com/photo-1497215728101-856f4ea42174?w=400&h=300&fit=crop",

features: ["Private Office", "Meeting Rooms", "Gym"]

},

{

id: 5,

name: "Startup Hub",

rent: 25000,

address: "987 Tech Park, Chennai, Tamil Nadu",

image: "https://images.unsplash.com/photo-1497366412874-3415097a27e7?w=400&h=300&fit=crop",

features: ["24/7 Access", "Printer", "Lounge"]

},

{

id: 6,

name: "Corporate Headquarters",

rent: 200000,

address: "111 Business Plaza, Gurgaon, Haryana",

image: "https://images.unsplash.com/photo-1486406146926-c627a92ad1ab?w=400&h=300&fit=crop",

features: ["Multiple Floors", "Auditorium", "VIP Parking"]

}

];

// Function to determine rent color based on amount

const getRentColor = (rent) => {

return rent < 60000 ? 'red' : 'green';

};

// Function to format rent with currency

const formatRent = (rent) => {

return `₹${rent.toLocaleString()}/month`;

};

return (

<div className="App">

{/\* Create an element to display the heading of the page \*/}

<header className="app-header">

<h1 className="main-heading">🏢 Office Space Rental Portal</h1>

<p className="subtitle">Find Your Perfect Workspace</p>

</header>

<main className="main-content">

{/\* Featured Office Section \*/}

<section className="featured-office">

<h2 className="section-heading">✨ Featured Office Space</h2>

<div className="office-card featured">

{/\* Attribute to display the image of the office space \*/}

<img

src={sampleOffice.image}

alt={sampleOffice.name}

className="office-image"

/>

<div className="office-details">

<h3 className="office-name">{sampleOffice.name}</h3>

<p className="office-address">📍 {sampleOffice.address}</p>

<p className="office-rent" style={{ color: getRentColor(sampleOffice.rent) }}>

💰 {formatRent(sampleOffice.rent)}

</p>

</div>

</div>

</section>

{/\* All Office Spaces Section \*/}

<section className="office-listings">

<h2 className="section-heading">🏢 Available Office Spaces</h2>

<div className="office-grid">

{/\* Loop through the office space items to display more data \*/}

{officeSpaces.map((office) => (

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

{/\* Attribute to display the image of each office space \*/}

<img

src={office.image}

alt={office.name}

className="office-image"

/>

<div className="office-details">

<h3 className="office-name">{office.name}</h3>

<p className="office-address">📍 {office.address}</p>

{/\* Apply CSS to display rent color: Red if below 60000, Green if above \*/}

<p className="office-rent" style={{ color: getRentColor(office.rent) }}>

💰 {formatRent(office.rent)}

</p>

<div className="office-features">

<strong>Features:</strong>

<ul className="features-list">

{office.features.map((feature, index) => (

<li key={index} className="feature-item">{feature}</li>

))}

</ul>

</div>

<button className="contact-btn">Contact Now</button>

</div>

</div>

))}

</div>

</section>

{/\* Statistics Section \*/}

<section className="statistics">

<h2 className="section-heading">📊 Rental Statistics</h2>

<div className="stats-grid">

<div className="stat-card">

<h3>Total Offices</h3>

<p className="stat-number">{officeSpaces.length + 1}</p>

</div>

<div className="stat-card">

<h3>Budget-Friendly (Under ₹60,000)</h3>

<p className="stat-number red">

{officeSpaces.filter(office => office.rent < 60000).length}

</p>

</div>

<div className="stat-card">

<h3>Premium (Above ₹60,000)</h3>

<p className="stat-number green">

{officeSpaces.filter(office => office.rent >= 60000).length}

</p>

</div>

</div>

</section>

</main>

<footer className="app-footer">

<p>© 2025 Office Space Rental Portal - Find Your Dream Workspace</p>

</footer>

</div>

);

}

export default App;

**//App.test.js**

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

render(<App />);

const linkElement = screen.getByText(/learn react/i);

expect(linkElement).toBeInTheDocument();

});

**//index.css**

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

**//index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**//reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

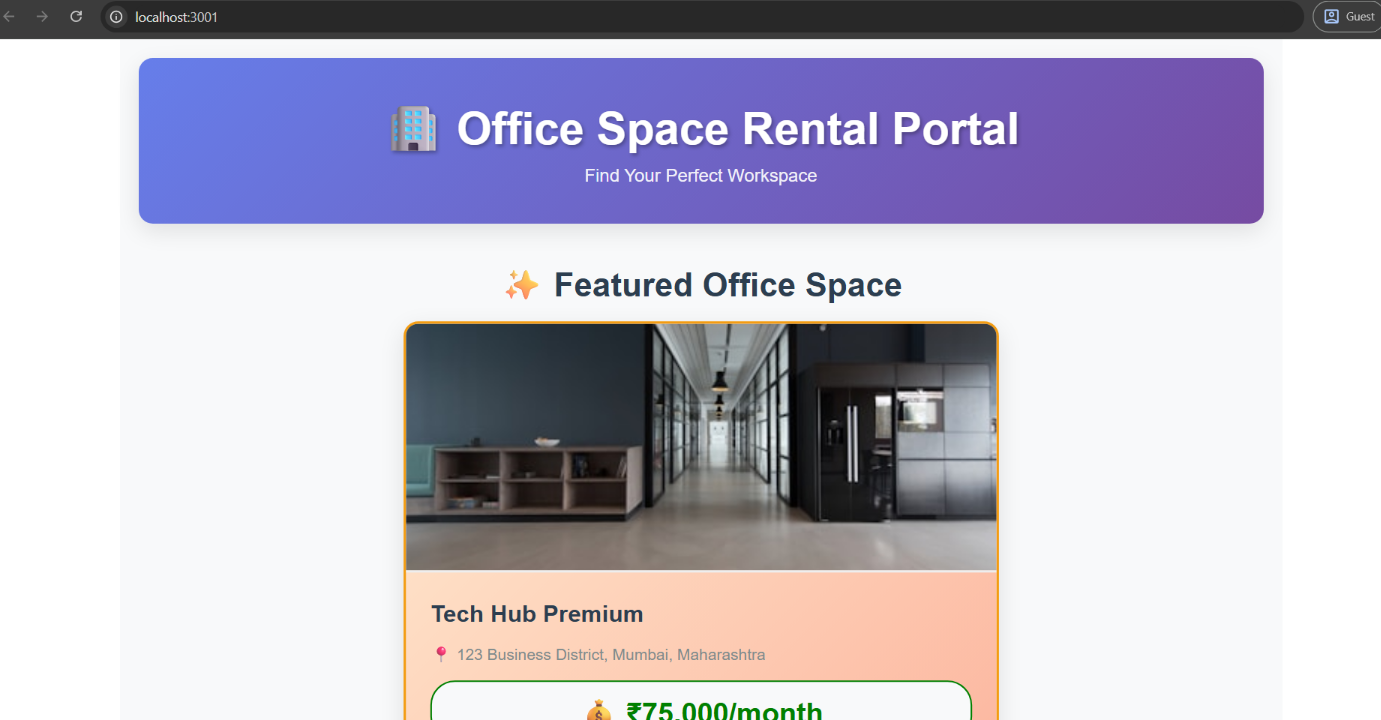
getTTFB(onPerfEntry);

});

}

};

export default reportWebVitals;

**OUTPUT:**

1. **Eventexamplesapp**

**//index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>React App</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**//App.css**

/\* App Container \*/

.App {

max-width: 1200px;

margin: 0 auto;

padding: 20px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

min-height: 100vh;

}

/\* Header Styles \*/

.app-header {

text-align: center;

background: white;

color: #2c3e50;

padding: 40px 20px;

border-radius: 20px;

margin-bottom: 30px;

box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);

}

.app-header h1 {

font-size: 3rem;

margin: 0 0 10px 0;

background: linear-gradient(135deg, #667eea, #764ba2);

-webkit-background-clip: text;

-webkit-text-fill-color: transparent;

background-clip: text;

}

.app-header p {

font-size: 1.2rem;

margin: 0;

color: #7f8c8d;

}

/\* Main Content \*/

.main-content {

display: grid;

gap: 30px;

margin-bottom: 40px;

}

/\* Section Styles \*/

section {

background: white;

padding: 30px;

border-radius: 15px;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

transition: transform 0.3s ease;

}

section:hover {

transform: translateY(-5px);

}

section h2 {

color: #2c3e50;

font-size: 1.8rem;

margin: 0 0 20px 0;

text-align: center;

border-bottom: 3px solid #667eea;

padding-bottom: 10px;

}

/\* Counter Section \*/

.counter-section {

background: linear-gradient(135deg, #ffecd2 0%, #fcb69f 100%);

}

.counter-display {

text-align: center;

margin: 30px 0;

}

.counter-value {

font-size: 4rem;

font-weight: bold;

color: #2c3e50;

background: white;

padding: 20px 40px;

border-radius: 50px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

display: inline-block;

min-width: 150px;

}

.counter-buttons {

display: flex;

justify-content: center;

gap: 20px;

margin: 30px 0;

flex-wrap: wrap;

}

/\* Button Styles \*/

.btn {

padding: 15px 25px;

border: none;

border-radius: 25px;

font-size: 1.1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

min-width: 180px;

}

.btn:hover {

transform: translateY(-3px);

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.3);

}

.increment-btn {

background: linear-gradient(135deg, #4CAF50, #45a049);

color: white;

}

.decrement-btn {

background: linear-gradient(135deg, #f44336, #d32f2f);

color: white;

}

.welcome-btn {

background: linear-gradient(135deg, #2196F3, #1976D2);

color: white;

display: block;

margin: 20px auto;

}

.synthetic-btn {

background: linear-gradient(135deg, #FF9800, #F57C00);

color: white;

display: block;

margin: 20px auto;

}

/\* Info Box \*/

.info-box {

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 10px;

margin-top: 20px;

border-left: 5px solid #667eea;

}

.info-box ul {

margin: 10px 0;

padding-left: 20px;

}

.info-box li {

margin: 5px 0;

color: #2c3e50;

}

/\* Welcome Section \*/

.welcome-section {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: white;

}

.welcome-section h2 {

color: white;

border-bottom-color: white;

}

/\* Synthetic Event Section \*/

.synthetic-event-section {

background: linear-gradient(135deg, #ffeaa7 0%, #fab1a0 100%);

}

.description {

text-align: center;

font-style: italic;

color: #7f8c8d;

margin: 15px 0;

}

.event-info {

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 10px;

margin-top: 20px;

}

.event-info h4 {

color: #2c3e50;

margin: 0 0 10px 0;

}

/\* Currency Converter Styles \*/

.currency-converter {

margin-top: 20px;

}

.converter-container {

max-width: 500px;

margin: 0 auto;

}

.converter-form {

background: #f8f9fa;

padding: 25px;

border-radius: 15px;

margin-bottom: 20px;

}

.input-group {

margin-bottom: 20px;

}

.input-group label {

display: block;

font-weight: bold;

margin-bottom: 10px;

color: #2c3e50;

font-size: 1.1rem;

}

.currency-symbol {

font-size: 1.3rem;

color: #e67e22;

margin-right: 5px;

}

.currency-input {

width: 100%;

padding: 15px;

border: 2px solid #ddd;

border-radius: 10px;

font-size: 1.1rem;

transition: border-color 0.3s ease;

box-sizing: border-box;

}

.currency-input:focus {

border-color: #667eea;

outline: none;

box-shadow: 0 0 10px rgba(102, 126, 234, 0.3);

}

.button-group {

display: flex;

gap: 15px;

justify-content: center;

}

.convert-btn {

background: linear-gradient(135deg, #00b894, #00a085);

color: white;

flex: 1;

}

.convert-btn:disabled {

background: #bdc3c7;

cursor: not-allowed;

transform: none;

}

.reset-btn {

background: linear-gradient(135deg, #636e72, #2d3436);

color: white;

flex: 1;

}

/\* Result Container \*/

.result-container {

background: linear-gradient(135deg, #74b9ff 0%, #0984e3 100%);

color: white;

padding: 25px;

border-radius: 15px;

text-align: center;

animation: slideIn 0.5s ease;

}

@keyframes slideIn {

from {

opacity: 0;

transform: translateY(20px);

}

to {

opacity: 1;

transform: translateY(0);

}

}

.conversion-display {

display: flex;

align-items: center;

justify-content: center;

gap: 20px;

margin: 20px 0;

flex-wrap: wrap;

}

.from-currency, .to-currency {

font-size: 1.5rem;

font-weight: bold;

background: white;

color: #2c3e50;

padding: 10px 20px;

border-radius: 25px;

}

.arrow {

font-size: 2rem;

font-weight: bold;

}

.rate-info {

font-size: 0.9rem;

opacity: 0.9;

margin: 10px 0 0 0;

}

/\* Additional Events Section \*/

.additional-events {

background: linear-gradient(135deg, #fd79a8 0%, #e84393 100%);

color: white;

}

.additional-events h2 {

color: white;

border-bottom-color: white;

}

.event-examples-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 20px;

margin-top: 20px;

}

.event-card {

background: white;

color: #2c3e50;

padding: 20px;

border-radius: 10px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

}

.event-card h4 {

margin: 0 0 15px 0;

color: #667eea;

text-align: center;

}

.event-input {

width: 100%;

padding: 10px;

border: 2px solid #ddd;

border-radius: 5px;

font-size: 1rem;

transition: all 0.3s ease;

box-sizing: border-box;

}

.hover-box {

background: #f8f9fa;

padding: 20px;

border-radius: 10px;

text-align: center;

cursor: pointer;

transition: all 0.3s ease;

border: 2px dashed #667eea;

}

.hover-box:hover {

background: #667eea;

color: white;

transform: scale(1.05);

}

.converter-info {

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 10px;

margin-top: 20px;

}

.converter-info h4 {

color: #2c3e50;

margin: 0 0 10px 0;

}

/\* Footer \*/

.app-footer {

text-align: center;

padding: 30px;

background: white;

color: #2c3e50;

border-radius: 15px;

margin-top: 30px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

}

/\* Responsive Design \*/

@media (max-width: 768px) {

.App {

padding: 10px;

}

.app-header h1 {

font-size: 2rem;

}

.counter-value {

font-size: 3rem;

padding: 15px 30px;

}

.counter-buttons {

flex-direction: column;

align-items: center;

}

.btn {

min-width: 250px;

}

.conversion-display {

flex-direction: column;

gap: 10px;

}

.button-group {

flex-direction: column;

}

}

**//App.js**

import React, { useState } from 'react';

import './App.css';

import CurrencyConvertor from './CurrencyConvertor';

function App() {

// State for counter

const [counter, setCounter] = useState(0);

// Method to increment the counter

const incrementCounter = () => {

setCounter(prevCounter => prevCounter + 1);

};

// Method to say hello with static message

const sayHello = () => {

alert('Hello! Welcome to Event Examples App - React Learning Journey!');

};

// Method to decrement the counter

const decrementCounter = () => {

setCounter(prevCounter => prevCounter - 1);

};

// Multiple methods handler for Increment button

const handleIncrementWithMultipleMethods = () => {

incrementCounter(); // a. To increment the value

sayHello(); // b. Say Hello followed by a static message

};

// Function that takes "welcome" as an argument

const sayMessage = (message) => {

alert(`${message}! Thank you for using our Event Examples App!`);

};

// Synthetic event handler for OnPress

const handleOnPress = (e) => {

e.preventDefault();

console.log('Synthetic Event Object:', e);

alert('I was clicked! This is a synthetic event example.');

};

return (

<div className="App">

{/\* Header Section \*/}

<header className="app-header">

<h1>🎯 Event Examples App</h1>

<p>Learn React Event Handling with Interactive Examples</p>

</header>

<main className="main-content">

{/\* Counter Section \*/}

<section className="counter-section">

<h2>📊 Counter with Event Handlers</h2>

<div className="counter-display">

<span className="counter-value">{counter}</span>

</div>

<div className="counter-buttons">

{/\* 1. Increment button that invokes multiple methods \*/}

<button

onClick={handleIncrementWithMultipleMethods}

className="btn increment-btn"

title="Increments counter AND shows hello message"

>

➕ Increment (Multiple Methods)

</button>

{/\* Decrement button \*/}

<button

onClick={decrementCounter}

className="btn decrement-btn"

>

➖ Decrement

</button>

</div>

<div className="info-box">

<p><strong>Increment Button:</strong> Calls two methods:</p>

<ul>

<li>✅ Increments the counter value</li>

<li>✅ Shows "Hello" message with static text</li>

</ul>

</div>

</section>

{/\* Welcome Button Section \*/}

<section className="welcome-section">

<h2>👋 Function with Arguments</h2>

{/\* 2. Button that invokes function with "welcome" as argument \*/}

<button

onClick={() => sayMessage('Welcome')}

className="btn welcome-btn"

>

🎉 Say Welcome

</button>

<p className="description">

This button calls a function that takes "welcome" as an argument

</p>

</section>

{/\* Synthetic Event Section \*/}

<section className="synthetic-event-section">

<h2>⚡ Synthetic Event Example</h2>

{/\* 3. Button with synthetic event "OnPress" \*/}

<button

onClick={handleOnPress}

className="btn synthetic-btn"

onMouseEnter={(e) => console.log('Mouse entered button:', e.type)}

onMouseLeave={(e) => console.log('Mouse left button:', e.type)}

>

🖱️ OnPress Event (Check Console)

</button>

<p className="description">

This button demonstrates synthetic events. Click and check the console!

</p>

<div className="event-info">

<h4>Synthetic Event Features:</h4>

<ul>

<li>✅ Cross-browser compatibility</li>

<li>✅ Event object logging</li>

<li>✅ Mouse enter/leave events</li>

<li>✅ Click event handling</li>

</ul>

</div>

</section>

{/\* Currency Converter Section \*/}

<section className="currency-section">

<h2>💰 Form Event Handling</h2>

<CurrencyConvertor />

<div className="converter-info">

<h4>Form Events Demonstrated:</h4>

<ul>

<li>✅ onSubmit event with preventDefault()</li>

<li>✅ onChange event for input handling</li>

<li>✅ Form validation</li>

<li>✅ State management with events</li>

</ul>

</div>

</section>

{/\* Additional Event Examples \*/}

<section className="additional-events">

<h2>🎨 More Event Examples</h2>

<div className="event-examples-grid">

<div className="event-card">

<h4>Keyboard Events</h4>

<input

type="text"

placeholder="Type something..."

onKeyDown={(e) => console.log('Key down:', e.key)}

onKeyUp={(e) => console.log('Key up:', e.key)}

className="event-input"

/>

</div>

<div className="event-card">

<h4>Focus Events</h4>

<input

type="text"

placeholder="Click to focus..."

onFocus={(e) => e.target.style.backgroundColor = '#e3f2fd'}

onBlur={(e) => e.target.style.backgroundColor = 'white'}

className="event-input"

/>

</div>

<div className="event-card">

<h4>Mouse Events</h4>

<div

className="hover-box"

onMouseOver={(e) => e.target.textContent = 'Mouse Over! 🎯'}

onMouseOut={(e) => e.target.textContent = 'Hover over me!'}

>

Hover over me!

</div>

</div>

</div>

</section>

</main>

<footer className="app-footer">

<p>© 2025 Event Examples App - Master React Event Handling</p>

</footer>

</div>

);

}

export default App;

**//App.test.js**

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

render(<App />);

const linkElement = screen.getByText(/learn react/i);

expect(linkElement).toBeInTheDocument();

});

**//CurrencyConverter.js**

import React, { useState } from 'react';

function CurrencyConvertor() {

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

const [euros, setEuros] = useState('');

const [conversionRate] = useState(0.011); // 1 INR = 0.011 EUR (approximate)

// Handle the Submit event for conversion

const handleSubmit = (e) => {

e.preventDefault();

if (rupees && !isNaN(rupees)) {

const convertedAmount = (parseFloat(rupees) \* conversionRate).toFixed(2);

setEuros(convertedAmount);

} else {

alert('Please enter a valid amount in Rupees');

}

};

// Handle input change

const handleInputChange = (e) => {

setRupees(e.target.value);

setEuros(''); // Clear euros when input changes

};

// Reset function

const handleReset = () => {

setRupees('');

setEuros('');

};

return (

<div className="currency-converter">

<h2>💱 Currency Converter (INR to EUR)</h2>

<div className="converter-container">

<form onSubmit={handleSubmit} className="converter-form">

<div className="input-group">

<label htmlFor="rupees">

<span className="currency-symbol">₹</span>

Enter Amount in Indian Rupees:

</label>

<input

type="number"

id="rupees"

value={rupees}

onChange={handleInputChange}

placeholder="Enter amount in INR"

className="currency-input"

step="0.01"

min="0"

/>

</div>

<div className="button-group">

<button

type="submit"

className="convert-btn"

disabled={!rupees}

>

Convert to EUR

</button>

<button

type="button"

onClick={handleReset}

className="reset-btn"

>

Reset

</button>

</div>

</form>

{euros && (

<div className="result-container">

<h3>Conversion Result:</h3>

<div className="conversion-display">

<span className="from-currency">

₹{parseFloat(rupees).toLocaleString('en-IN')}

</span>

<span className="arrow">→</span>

<span className="to-currency">

€{euros}

</span>

</div>

<p className="rate-info">

Exchange Rate: 1 INR = €{conversionRate}

</p>

</div>

)}

</div>

</div>

);

}

export default CurrencyConvertor;

**//index.css**

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

**//index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**//reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

getTTFB(onPerfEntry);

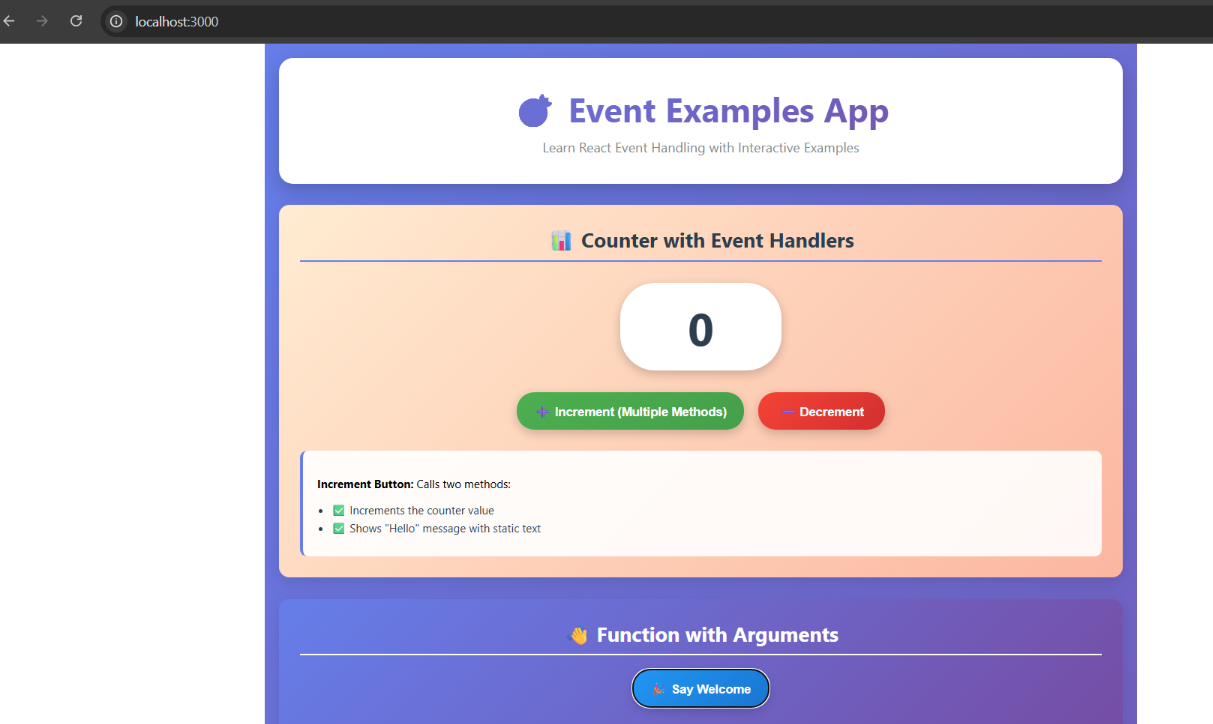
});

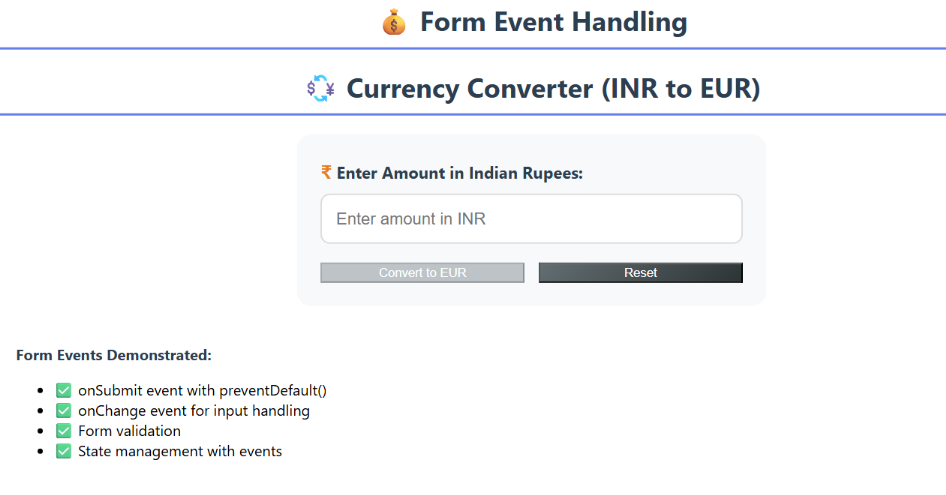
}

};

export default reportWebVitals;

**OUTPUT:**





1. **Ticketbookingapp**

**//index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>React App</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**//App.css**

/\* Global Styles \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

min-height: 100vh;

}

.App {

min-height: 100vh;

display: flex;

flex-direction: column;

}

/\* Header Styles \*/

.app-header {

background: white;

box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);

position: sticky;

top: 0;

z-index: 100;

}

.header-content {

max-width: 1200px;

margin: 0 auto;

padding: 20px;

display: flex;

justify-content: space-between;

align-items: center;

}

.header-content h1 {

color: #2c3e50;

font-size: 2.5rem;

background: linear-gradient(135deg, #667eea, #764ba2);

-webkit-background-clip: text;

-webkit-text-fill-color: transparent;

background-clip: text;

}

.nav-section {

display: flex;

align-items: center;

gap: 15px;

}

.user-info {

display: flex;

align-items: center;

gap: 15px;

}

.welcome-text {

font-weight: bold;

color: #2c3e50;

font-size: 1.1rem;

}

.nav-btn {

padding: 12px 24px;

border: none;

border-radius: 25px;

font-size: 1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

box-shadow: 0 4px 15px rgba(0, 0, 0, 0.2);

}

.nav-btn.login {

background: linear-gradient(135deg, #4CAF50, #45a049);

color: white;

}

.nav-btn.logout {

background: linear-gradient(135deg, #f44336, #d32f2f);

color: white;

}

.nav-btn:hover {

transform: translateY(-2px);

box-shadow: 0 6px 20px rgba(0, 0, 0, 0.3);

}

/\* Main Content \*/

.main-content {

flex: 1;

max-width: 1200px;

margin: 0 auto;

padding: 30px 20px;

width: 100%;

}

/\* Guest Page Styles \*/

.guest-page {

background: white;

border-radius: 15px;

padding: 30px;

box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);

}

.page-header {

text-align: center;

margin-bottom: 40px;

padding: 30px;

background: linear-gradient(135deg, #ffeaa7 0%, #fab1a0 100%);

border-radius: 15px;

}

.page-header h1 {

color: #2c3e50;

font-size: 2.5rem;

margin-bottom: 15px;

}

.guest-message {

font-size: 1.2rem;

color: #7f8c8d;

margin-bottom: 20px;

}

.login-btn {

background: linear-gradient(135deg, #4CAF50, #45a049);

color: white;

border: none;

padding: 15px 30px;

border-radius: 25px;

font-size: 1.1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

}

.login-btn:hover {

transform: translateY(-3px);

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.3);

}

/\* User Page Styles \*/

.user-page {

background: white;

border-radius: 15px;

padding: 30px;

box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);

}

.user-header {

text-align: center;

margin-bottom: 40px;

padding: 30px;

background: linear-gradient(135deg, #74b9ff 0%, #0984e3 100%);

color: white;

border-radius: 15px;

}

.user-header h1 {

font-size: 2.5rem;

margin-bottom: 15px;

}

.logout-btn {

background: linear-gradient(135deg, #f44336, #d32f2f);

color: white;

border: none;

padding: 12px 24px;

border-radius: 25px;

font-size: 1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

margin-top: 15px;

}

.logout-btn:hover {

transform: translateY(-2px);

box-shadow: 0 6px 20px rgba(244, 67, 54, 0.4);

}

/\* Flights Container \*/

.flights-container {

margin-bottom: 50px;

}

.flights-container h2 {

color: #2c3e50;

font-size: 2rem;

margin-bottom: 30px;

text-align: center;

border-bottom: 3px solid #667eea;

padding-bottom: 15px;

}

.flights-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(400px, 1fr));

gap: 25px;

margin-top: 30px;

}

/\* Flight Card Styles \*/

.flight-card {

background: white;

border-radius: 15px;

padding: 25px;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

transition: all 0.3s ease;

border: 2px solid transparent;

}

.flight-card:hover {

transform: translateY(-5px);

box-shadow: 0 15px 40px rgba(0, 0, 0, 0.2);

}

.flight-card.guest {

border-color: #f39c12;

background: linear-gradient(135deg, #ffeaa7 0%, #fff 100%);

}

.flight-card.user {

border-color: #2ecc71;

background: linear-gradient(135deg, #d1f2eb 0%, #fff 100%);

}

.flight-header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 20px;

}

.airline {

font-size: 1.4rem;

color: #2c3e50;

font-weight: bold;

margin: 0;

}

.price {

font-size: 1.5rem;

font-weight: bold;

color: #e74c3c;

background: #fff;

padding: 8px 15px;

border-radius: 20px;

border: 2px solid #e74c3c;

}

/\* Flight Route \*/

.flight-route {

display: flex;

align-items: center;

justify-content: space-between;

margin: 20px 0;

padding: 20px;

background: #f8f9fa;

border-radius: 10px;

}

.route-info {

text-align: center;

flex: 1;

}

.city {

display: block;

font-size: 1.3rem;

font-weight: bold;

color: #2c3e50;

margin-bottom: 5px;

}

.time {

display: block;

font-size: 1rem;

color: #7f8c8d;

}

.route-arrow {

text-align: center;

margin: 0 20px;

}

.duration {

display: block;

font-size: 0.9rem;

color: #7f8c8d;

margin-bottom: 5px;

}

.arrow {

font-size: 2rem;

color: #667eea;

font-weight: bold;

}

/\* Flight Footer \*/

.flight-footer {

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 20px;

padding-top: 20px;

border-top: 1px solid #ecf0f1;

}

.seats {

color: #27ae60;

font-weight: bold;

font-size: 1rem;

}

.book-btn {

padding: 12px 24px;

border: none;

border-radius: 25px;

font-size: 1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

}

.book-btn.enabled {

background: linear-gradient(135deg, #2ecc71, #27ae60);

color: white;

box-shadow: 0 4px 15px rgba(46, 204, 113, 0.3);

}

.book-btn.enabled:hover {

transform: translateY(-2px);

box-shadow: 0 6px 20px rgba(46, 204, 113, 0.5);

}

.book-btn.disabled {

background: #bdc3c7;

color: #7f8c8d;

cursor: not-allowed;

}

/\* Guest Features \*/

.guest-features {

background: #f8f9fa;

padding: 30px;

border-radius: 15px;

margin-top: 40px;

}

.guest-features h3 {

color: #2c3e50;

margin-bottom: 20px;

text-align: center;

}

.guest-features ul {

list-style: none;

max-width: 400px;

margin: 0 auto;

}

.guest-features li {

padding: 8px 0;

font-size: 1.1rem;

}

/\* Bookings Container \*/

.bookings-container {

margin-top: 50px;

padding-top: 30px;

border-top: 3px solid #667eea;

}

.bookings-container h2 {

color: #2c3e50;

font-size: 2rem;

margin-bottom: 30px;

text-align: center;

}

.no-bookings {

text-align: center;

padding: 40px;

background: #f8f9fa;

border-radius: 15px;

color: #7f8c8d;

font-size: 1.2rem;

}

.bookings-list {

display: grid;

gap: 20px;

}

.booking-card {

background: linear-gradient(135deg, #74b9ff 0%, #0984e3 100%);

color: white;

padding: 25px;

border-radius: 15px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

}

.booking-header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 15px;

border-bottom: 1px solid rgba(255, 255, 255, 0.3);

padding-bottom: 10px;

}

.status {

background: #27ae60;

padding: 5px 15px;

border-radius: 15px;

font-size: 0.9rem;

font-weight: bold;

}

.booking-details p {

margin: 8px 0;

font-size: 1rem;

}

/\* Booking Form \*/

.booking-form-container {

max-width: 600px;

margin: 0 auto;

background: #f8f9fa;

padding: 40px;

border-radius: 15px;

box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);

}

.booking-form-container h2 {

color: #2c3e50;

text-align: center;

margin-bottom: 30px;

}

.selected-flight {

background: linear-gradient(135deg, #667eea, #764ba2);

color: white;

padding: 20px;

border-radius: 10px;

margin-bottom: 30px;

text-align: center;

}

.flight-summary {

margin-top: 15px;

}

.flight-summary p {

margin: 5px 0;

}

.flight-summary .price {

font-size: 1.5rem;

font-weight: bold;

color: #f1c40f;

}

.booking-form h3 {

color: #2c3e50;

margin-bottom: 20px;

}

.form-group {

margin-bottom: 20px;

}

.form-group label {

display: block;

font-weight: bold;

margin-bottom: 8px;

color: #2c3e50;

}

.form-group input {

width: 100%;

padding: 12px;

border: 2px solid #ddd;

border-radius: 8px;

font-size: 1rem;

transition: border-color 0.3s ease;

}

.form-group input:focus {

border-color: #667eea;

outline: none;

box-shadow: 0 0 10px rgba(102, 126, 234, 0.3);

}

.form-buttons {

display: flex;

gap: 15px;

margin-top: 30px;

}

.confirm-btn {

flex: 1;

background: linear-gradient(135deg, #2ecc71, #27ae60);

color: white;

border: none;

padding: 15px;

border-radius: 8px;

font-size: 1.1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

}

.confirm-btn:hover {

transform: translateY(-2px);

box-shadow: 0 6px 20px rgba(46, 204, 113, 0.4);

}

.cancel-btn {

flex: 1;

background: linear-gradient(135deg, #f44336, #d32f2f);

color: white;

border: none;

padding: 15px;

border-radius: 8px;

font-size: 1.1rem;

font-weight: bold;

cursor: pointer;

transition: all 0.3s ease;

}

.cancel-btn:hover {

transform: translateY(-2px);

box-shadow: 0 6px 20px rgba(244, 67, 54, 0.4);

}

/\* Footer \*/

.app-footer {

background: white;

padding: 30px 20px;

text-align: center;

box-shadow: 0 -4px 20px rgba(0, 0, 0, 0.1);

margin-top: auto;

}

.footer-content {

max-width: 1200px;

margin: 0 auto;

}

.footer-links {

margin-top: 10px;

font-weight: bold;

color: #667eea;

}

/\* Features Info \*/

.features-info {

background: white;

padding: 30px;

border-radius: 15px;

margin: 30px auto;

max-width: 1200px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

}

.features-info h3 {

text-align: center;

color: #2c3e50;

margin-bottom: 30px;

font-size: 1.8rem;

}

.features-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));

gap: 30px;

}

.feature-card {

background: linear-gradient(135deg, #74b9ff 0%, #0984e3 100%);

color: white;

padding: 25px;

border-radius: 15px;

text-align: center;

}

.feature-card h4 {

margin-bottom: 15px;

font-size: 1.3rem;

}

.feature-card ul {

list-style: none;

text-align: left;

}

.feature-card li {

padding: 5px 0;

font-size: 1rem;

}

/\* Responsive Design \*/

@media (max-width: 768px) {

.header-content {

flex-direction: column;

gap: 20px;

text-align: center;

}

.header-content h1 {

font-size: 2rem;

}

.flights-grid {

grid-template-columns: 1fr;

}

.flight-route {

flex-direction: column;

gap: 15px;

}

.route-arrow {

margin: 10px 0;

}

.arrow {

transform: rotate(90deg);

}

.flight-footer {

flex-direction: column;

gap: 15px;

text-align: center;

}

.form-buttons {

flex-direction: column;

}

.features-grid {

grid-template-columns: 1fr;

}

}

**//App.js**

import React, { useState } from 'react';

import './App.css';

import GuestPage from './GuestPage';

import UserPage from './UserPage';

function App() {

// State to track if user is logged in

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

const [username, setUsername] = useState('');

// Handle login

const handleLogin = () => {

const name = prompt('Enter your username:');

if (name && name.trim()) {

setUsername(name.trim());

setIsLoggedIn(true);

}

};

// Handle logout

const handleLogout = () => {

setIsLoggedIn(false);

setUsername('');

alert('You have been successfully logged out!');

};

return (

<div className="App">

{/\* Header with Navigation \*/}

<header className="app-header">

<div className="header-content">

<h1>🛫 FlightBook Pro</h1>

<div className="nav-section">

{isLoggedIn ? (

<div className="user-info">

<span className="welcome-text">Welcome, {username}</span>

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

🚪 Logout

</button>

</div>

) : (

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

🔑 Login

</button>

)}

</div>

</div>

</header>

{/\* Main Content - Conditional Rendering \*/}

<main className="main-content">

{isLoggedIn ? (

<UserPage onLogout={handleLogout} username={username} />

) : (

<GuestPage onLogin={handleLogin} />

)}

</main>

{/\* Footer \*/}

<footer className="app-footer">

<div className="footer-content">

<p>© 2025 FlightBook Pro - Your Trusted Flight Booking Partner</p>

<div className="footer-links">

<span>Status: {isLoggedIn ? '🟢 Logged In' : '🔴 Guest Mode'}</span>

</div>

</div>

</footer>

{/\* Features Info \*/}

<div className="features-info">

<h3>🌟 App Features:</h3>

<div className="features-grid">

<div className="feature-card">

<h4>👥 Guest Mode</h4>

<ul>

<li>Browse all flights</li>

<li>View prices & schedules</li>

<li>No booking access</li>

</ul>

</div>

<div className="feature-card">

<h4>🔐 User Mode</h4>

<ul>

<li>Full flight booking</li>

<li>Manage bookings</li>

<li>Passenger details</li>

</ul>

</div>

</div>

</div>

</div>

);

}

export default App;

**//App.test.js**

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

render(<App />);

const linkElement = screen.getByText(/learn react/i);

expect(linkElement).toBeInTheDocument();

});

**//GuestPage.js**

import React from 'react';

function GuestPage({ onLogin }) {

// Sample flight data for browsing

const flights = [

{

id: 1,

airline: "Air India",

from: "Mumbai",

to: "Delhi",

departure: "08:30 AM",

arrival: "10:45 AM",

duration: "2h 15m",

price: "₹6,500",

seats: "Available"

},

{

id: 2,

airline: "IndiGo",

from: "Bangalore",

to: "Chennai",

departure: "02:15 PM",

arrival: "03:30 PM",

duration: "1h 15m",

price: "₹4,200",

seats: "Available"

},

{

id: 3,

airline: "SpiceJet",

from: "Delhi",

to: "Goa",

departure: "11:00 AM",

arrival: "01:30 PM",

duration: "2h 30m",

price: "₹8,900",

seats: "Available"

},

{

id: 4,

airline: "Vistara",

from: "Hyderabad",

to: "Mumbai",

departure: "06:45 PM",

arrival: "08:15 PM",

duration: "1h 30m",

price: "₹7,300",

seats: "Available"

},

{

id: 5,

airline: "GoAir",

from: "Kolkata",

to: "Pune",

departure: "09:20 AM",

arrival: "12:10 PM",

duration: "2h 50m",

price: "₹5,800",

seats: "Available"

}

];

return (

<div className="guest-page">

<div className="page-header">

<h1>✈️ Flight Search - Browse Flights</h1>

<p className="guest-message">

Welcome Guest! Browse available flights below.

<strong> Please login to book tickets.</strong>

</p>

<button onClick={onLogin} className="login-btn">

🔑 Login to Book Tickets

</button>

</div>

<div className="flights-container">

<h2>Available Flights</h2>

<div className="flights-grid">

{flights.map(flight => (

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

<div className="flight-header">

<h3 className="airline">{flight.airline}</h3>

<span className="price">{flight.price}</span>

</div>

<div className="flight-route">

<div className="route-info">

<span className="city">{flight.from}</span>

<span className="time">{flight.departure}</span>

</div>

<div className="route-arrow">

<span className="duration">{flight.duration}</span>

<div className="arrow">→</div>

</div>

<div className="route-info">

<span className="city">{flight.to}</span>

<span className="time">{flight.arrival}</span>

</div>

</div>

<div className="flight-footer">

<span className="seats">✅ {flight.seats}</span>

<button className="book-btn disabled" disabled>

🔒 Login Required

</button>

</div>

</div>

))}

</div>

</div>

<div className="guest-features">

<h3>🌟 Available Features for Guests:</h3>

<ul>

<li>✅ Browse all available flights</li>

<li>✅ View flight details and prices</li>

<li>✅ Check flight schedules</li>

<li>❌ Book tickets (Login required)</li>

<li>❌ Manage bookings (Login required)</li>

</ul>

</div>

</div>

);

}

export default GuestPage;

**//UserPage.js**

import React, { useState } from 'react';

function UserPage({ onLogout, username }) {

const [bookings, setBookings] = useState([]);

const [showBookingForm, setShowBookingForm] = useState(false);

const [selectedFlight, setSelectedFlight] = useState(null);

// Sample flight data for booking

const flights = [

{

id: 1,

airline: "Air India",

from: "Mumbai",

to: "Delhi",

departure: "08:30 AM",

arrival: "10:45 AM",

duration: "2h 15m",

price: 6500,

seats: "Available"

},

{

id: 2,

airline: "IndiGo",

from: "Bangalore",

to: "Chennai",

departure: "02:15 PM",

arrival: "03:30 PM",

duration: "1h 15m",

price: 4200,

seats: "Available"

},

{

id: 3,

airline: "SpiceJet",

from: "Delhi",

to: "Goa",

departure: "11:00 AM",

arrival: "01:30 PM",

duration: "2h 30m",

price: 8900,

seats: "Available"

},

{

id: 4,

airline: "Vistara",

from: "Hyderabad",

to: "Mumbai",

departure: "06:45 PM",

arrival: "08:15 PM",

duration: "1h 30m",

price: 7300,

seats: "Available"

},

{

id: 5,

airline: "GoAir",

from: "Kolkata",

to: "Pune",

departure: "09:20 AM",

arrival: "12:10 PM",

duration: "2h 50m",

price: 5800,

seats: "Available"

}

];

const handleBookFlight = (flight) => {

setSelectedFlight(flight);

setShowBookingForm(true);

};

const handleConfirmBooking = (passengerDetails) => {

const newBooking = {

id: Date.now(),

flight: selectedFlight,

passenger: passengerDetails,

bookingDate: new Date().toLocaleDateString(),

status: "Confirmed"

};

setBookings([...bookings, newBooking]);

setShowBookingForm(false);

setSelectedFlight(null);

alert(`✅ Booking confirmed for ${passengerDetails.name}!`);

};

return (

<div className="user-page">

<div className="user-header">

<h1>✈️ Welcome, {username}!</h1>

<p>Book your flights and manage your bookings</p>

<button onClick={onLogout} className="logout-btn">

🚪 Logout

</button>

</div>

{!showBookingForm ? (

<>

{/\* Available Flights for Booking \*/}

<div className="flights-container">

<h2>🎯 Book Your Flight</h2>

<div className="flights-grid">

{flights.map(flight => (

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

<div className="flight-header">

<h3 className="airline">{flight.airline}</h3>

<span className="price">₹{flight.price.toLocaleString()}</span>

</div>

<div className="flight-route">

<div className="route-info">

<span className="city">{flight.from}</span>

<span className="time">{flight.departure}</span>

</div>

<div className="route-arrow">

<span className="duration">{flight.duration}</span>

<div className="arrow">→</div>

</div>

<div className="route-info">

<span className="city">{flight.to}</span>

<span className="time">{flight.arrival}</span>

</div>

</div>

<div className="flight-footer">

<span className="seats">✅ {flight.seats}</span>

<button

className="book-btn enabled"

onClick={() => handleBookFlight(flight)}

>

🎫 Book Now

</button>

</div>

</div>

))}

</div>

</div>

{/\* My Bookings Section \*/}

<div className="bookings-container">

<h2>📋 My Bookings</h2>

{bookings.length === 0 ? (

<div className="no-bookings">

<p>No bookings yet. Book your first flight above!</p>

</div>

) : (

<div className="bookings-list">

{bookings.map(booking => (

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

<div className="booking-header">

<h4>Booking #{booking.id}</h4>

<span className="status">{booking.status}</span>

</div>

<div className="booking-details">

<p><strong>Flight:</strong> {booking.flight.airline}</p>

<p><strong>Route:</strong> {booking.flight.from} → {booking.flight.to}</p>

<p><strong>Passenger:</strong> {booking.passenger.name}</p>

<p><strong>Date:</strong> {booking.bookingDate}</p>

<p><strong>Amount:</strong> ₹{booking.flight.price.toLocaleString()}</p>

</div>

</div>

))}

</div>

)}

</div>

</>

) : (

<BookingForm

flight={selectedFlight}

onConfirm={handleConfirmBooking}

onCancel={() => setShowBookingForm(false)}

/>

)}

</div>

);

}

// Booking Form Component

function BookingForm({ flight, onConfirm, onCancel }) {

const [passengerName, setPassengerName] = useState('');

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

const [phone, setPhone] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

if (passengerName && email && phone) {

onConfirm({

name: passengerName,

email: email,

phone: phone

});

} else {

alert('Please fill all fields');

}

};

return (

<div className="booking-form-container">

<h2>🎫 Complete Your Booking</h2>

<div className="selected-flight">

<h3>Selected Flight:</h3>

<div className="flight-summary">

<p><strong>{flight.airline}</strong></p>

<p>{flight.from} → {flight.to}</p>

<p>{flight.departure} - {flight.arrival}</p>

<p className="price">₹{flight.price.toLocaleString()}</p>

</div>

</div>

<form onSubmit={handleSubmit} className="booking-form">

<h3>Passenger Details:</h3>

<div className="form-group">

<label>Full Name:</label>

<input

type="text"

value={passengerName}

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

placeholder="Enter passenger name"

required

/>

</div>

<div className="form-group">

<label>Email:</label>

<input

type="email"

value={email}

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

placeholder="Enter email address"

required

/>

</div>

<div className="form-group">

<label>Phone Number:</label>

<input

type="tel"

value={phone}

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

placeholder="Enter phone number"

required

/>

</div>

<div className="form-buttons">

<button type="submit" className="confirm-btn">

✅ Confirm Booking

</button>

<button type="button" onClick={onCancel} className="cancel-btn">

❌ Cancel

</button>

</div>

</form>

</div>

);

}

export default UserPage;

**//index.css**

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

**//index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**//reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

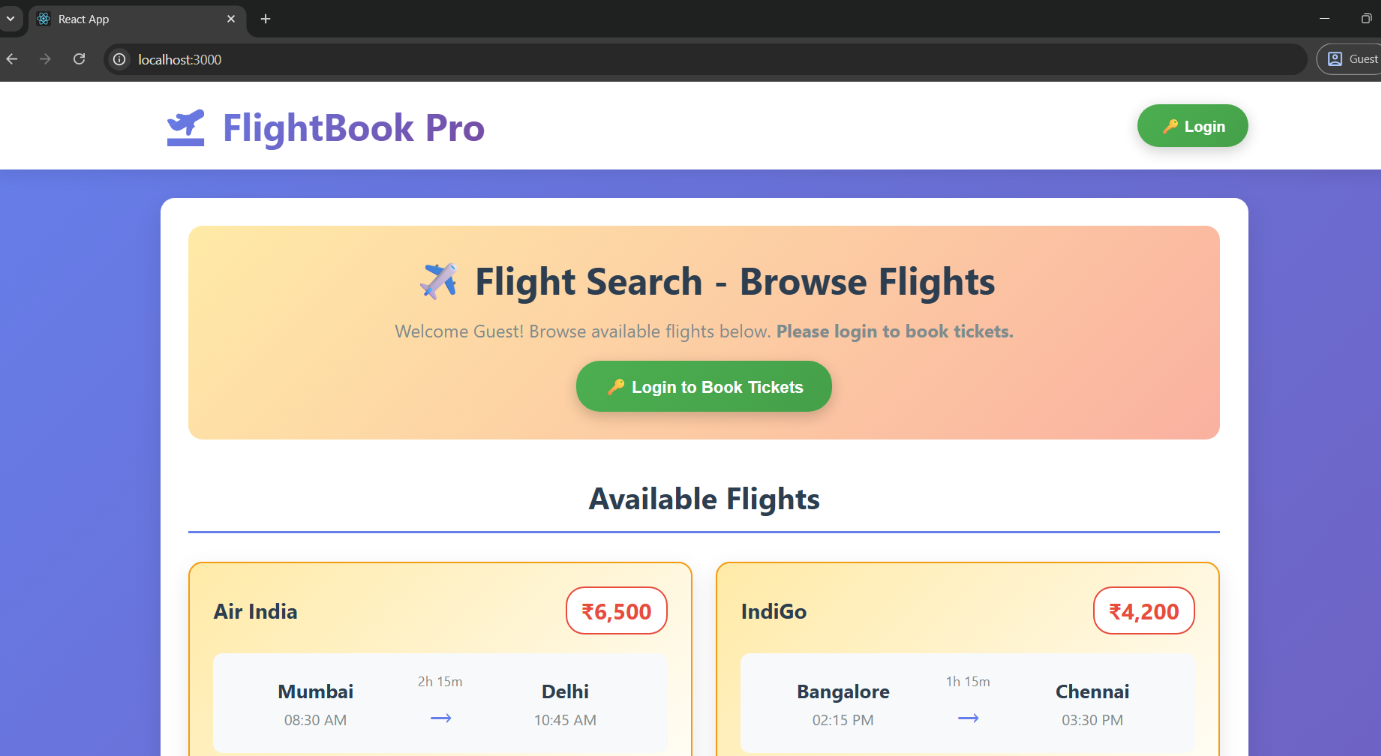
getTTFB(onPerfEntry);

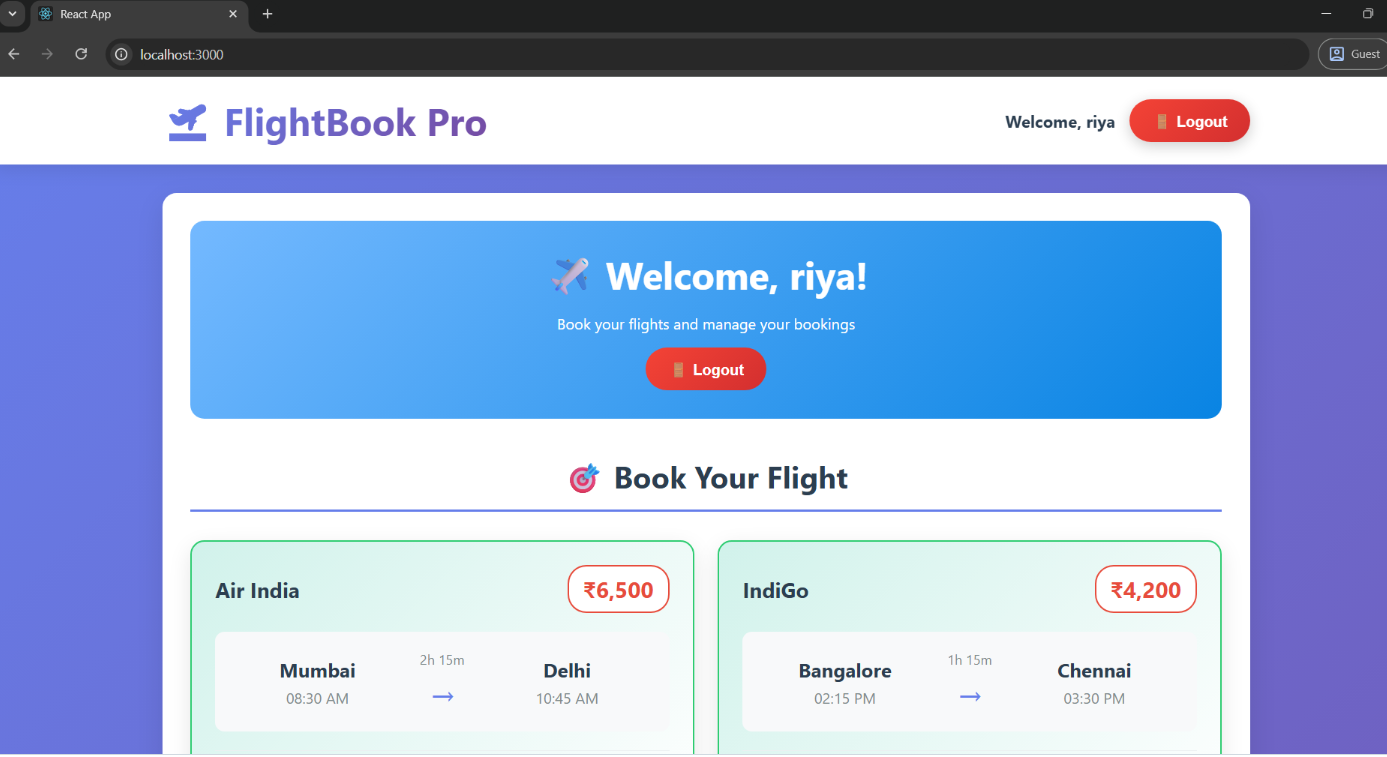
});

}

};

export default reportWebVitals;

**OUTPUT:**

****

1. **Bloggerapp**

**//index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>React App</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**//BlogDetails.js**

import React, { useState } from 'react';

const BlogDetails = () => {

// State for various conditional rendering scenarios

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

const [userRole, setUserRole] = useState('reader');

const [currentView, setCurrentView] = useState('all');

const [searchTerm, setSearchTerm] = useState('');

const [selectedCategory, setSelectedCategory] = useState('all');

// Sample blog data

const blogs = [

{

id: 1,

title: "Getting Started with React Hooks",

excerpt: "Learn the fundamentals of React Hooks and how they can simplify your component logic.",

category: "tutorial",

author: "Sarah Johnson",

publishDate: "2024-01-15",

readTime: 8,

likes: 245,

comments: 23,

tags: ["react", "hooks", "javascript"],

isPremium: false,

isPublished: true,

viewCount: 1250

},

{

id: 2,

title: "Advanced State Management Patterns",

excerpt: "Explore complex state management techniques including Context API and custom hooks.",

category: "advanced",

author: "Mike Chen",

publishDate: "2024-01-12",

readTime: 15,

likes: 189,

comments: 45,

tags: ["react", "state", "context"],

isPremium: true,

isPublished: true,

viewCount: 890

},

{

id: 3,

title: "Building Responsive Web Applications",

excerpt: "Master the art of creating applications that work seamlessly across all devices.",

category: "design",

author: "Emma Wilson",

publishDate: "2024-01-10",

readTime: 12,

likes: 156,

comments: 18,

tags: ["css", "responsive", "design"],

isPremium: false,

isPublished: true,

viewCount: 2100

},

{

id: 4,

title: "Future of Web Development",

excerpt: "A comprehensive look at emerging technologies and trends shaping the web development landscape.",

category: "trends",

author: "David Rodriguez",

publishDate: "2024-01-08",

readTime: 20,

likes: 78,

comments: 12,

tags: ["future", "trends", "technology"],

isPremium: true,

isPublished: false,

viewCount: 245

},

{

id: 5,

title: "JavaScript Performance Optimization",

excerpt: "Techniques and best practices for writing high-performance JavaScript applications.",

category: "performance",

author: "Lisa Park",

publishDate: "2024-01-05",

readTime: 18,

likes: 312,

comments: 67,

tags: ["javascript", "performance", "optimization"],

isPremium: false,

isPublished: true,

viewCount: 1850

}

];

// Filter blogs based on current conditions

const getFilteredBlogs = () => {

return blogs.filter(blog => {

// Search filter

const matchesSearch = blog.title.toLowerCase().includes(searchTerm.toLowerCase()) ||

blog.excerpt.toLowerCase().includes(searchTerm.toLowerCase()) ||

blog.tags.some(tag => tag.toLowerCase().includes(searchTerm.toLowerCase()));

// Category filter

const matchesCategory = selectedCategory === 'all' || blog.category === selectedCategory;

// User role and premium access

const hasAccess = !blog.isPremium || userRole === 'premium' || userRole === 'admin';

// Published status for different roles

const canView = blog.isPublished || userRole === 'admin' || (userRole === 'author' && blog.author === 'Current User');

return matchesSearch && matchesCategory && hasAccess && canView;

});

};

return (

<div className="blog-details">

<div className="component-header">

<h2>📝 Blog Details Component</h2>

<p>Advanced Conditional Rendering with User Roles & Filters</p>

</div>

{/\* 1. Authentication State Management \*/}

<div className="conditional-section">

<h3>1. Complex Authentication Flow</h3>

<div className="user-controls">

<div>

<button

className={isLoggedIn ? "logout-btn" : "login-btn"}

onClick={() => setIsLoggedIn(!isLoggedIn)}

>

{isLoggedIn ? "Logout" : "Login"}

</button>

{/\* Conditional role selector - only shown when logged in \*/}

{isLoggedIn && (

<div className="role-selector">

<label>Role: </label>

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

<option value="reader">Reader</option>

<option value="author">Author</option>

<option value="premium">Premium User</option>

<option value="admin">Administrator</option>

</select>

</div>

)}

</div>

{/\* Status indicator with nested conditionals \*/}

<div>

{isLoggedIn ? (

<span style={{ color: '#27ae60', fontWeight: 'bold' }}>

✅ Logged in as {userRole === 'admin' ? '👑 Administrator' :

userRole === 'premium' ? '⭐ Premium User' :

userRole === 'author' ? '✍️ Author' : '👤 Reader'}

</span>

) : (

<span style={{ color: '#e74c3c', fontWeight: 'bold' }}>

❌ Not logged in - Limited access

</span>

)}

</div>

</div>

</div>

{/\* 2. View Mode Switching \*/}

<div className="conditional-section">

<h3>2. Dynamic View Switching</h3>

<div className="view-navigation">

{['all', 'published', 'draft', 'premium'].map(view => (

<button

key={view}

className={currentView === view ? 'active' : ''}

onClick={() => setCurrentView(view)}

disabled={!isLoggedIn && (view === 'draft' || view === 'premium')}

>

{view.charAt(0).toUpperCase() + view.slice(1)}

{/\* Conditional badge showing count \*/}

{view === 'premium' && (

<span style={{ marginLeft: '5px', fontSize: '0.8em' }}>

({blogs.filter(b => b.isPremium).length})

</span>

)}

</button>

))}

</div>

</div>

{/\* 3. Advanced Filtering System \*/}

<div className="conditional-section">

<h3>3. Multi-layered Filtering</h3>

<div className="filters-section">

<div className="search-bar">

<input

type="text"

placeholder="Search blogs, tags, or content..."

value={searchTerm}

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

/>

</div>

<div className="category-filters">

{['all', 'tutorial', 'advanced', 'design', 'trends', 'performance'].map(category => (

<button

key={category}

className={selectedCategory === category ? 'active' : ''}

onClick={() => setSelectedCategory(category)}

>

{category.charAt(0).toUpperCase() + category.slice(1)}

</button>

))}

</div>

</div>

</div>

{/\* 4. Permission-based Rendering \*/}

<div className="conditional-section">

<h3>4. Role-based Content Access</h3>

<div className="example-box">

{(() => {

// Complex permission logic

if (!isLoggedIn) {

return (

<div className="status-message guest">

<h4>🔒 Guest Access</h4>

<p>• View public blog previews</p>

<p>• Read first paragraph only</p>

<p>• Limited to 3 articles per day</p>

</div>

);

}

const permissions = {

reader: [

"Read all published articles",

"Comment on articles",

"Save articles to reading list"

],

author: [

"All reader permissions",

"Create and edit own articles",

"View draft articles",

"Access writing analytics"

],

premium: [

"All reader permissions",

"Access premium content",

"Ad-free reading experience",

"Download articles for offline reading"

],

admin: [

"Full platform access",

"Manage all articles",

"User management",

"Platform analytics",

"Content moderation"

]

};

return (

<div className="status-message member">

<h4>✅ {userRole.toUpperCase()} Permissions</h4>

<ul>

{permissions[userRole]?.map((permission, index) => (

<li key={index}>• {permission}</li>

))}

</ul>

</div>

);

})()}

</div>

</div>

{/\* 5. Complex Blog Grid with Multiple Conditions \*/}

<div className="conditional-section">

<h3>5. Smart Content Display</h3>

{/\* Conditional empty state messages \*/}

{!isLoggedIn && currentView === 'premium' ? (

<div className="example-box">

<p>🔐 Please log in to view premium content</p>

</div>

) : getFilteredBlogs().length === 0 ? (

<div className="example-box">

<p>

{searchTerm ?

`No blogs found matching "${searchTerm}"` :

selectedCategory !== 'all' ?

`No blogs found in "${selectedCategory}" category` :

"No blogs available for your current access level"

}

</p>

</div>

) : (

<div className="blogs-grid">

{getFilteredBlogs().map(blog => (

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

<div className="blog-header">

<div className="blog-icon">

{blog.category === 'tutorial' ? '📚' :

blog.category === 'advanced' ? '🔬' :

blog.category === 'design' ? '🎨' :

blog.category === 'trends' ? '🚀' :

blog.category === 'performance' ? '⚡' : '📝'}

</div>

<div className="blog-meta">

<span className="category">{blog.category}</span>

{blog.isPremium && <span className="premium-badge">Premium</span>}

{!blog.isPublished && userRole === 'admin' && (

<span className="draft-badge">Draft</span>

)}

</div>

</div>

<div className="blog-content">

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

{/\* Conditional excerpt based on user access \*/}

<p className="blog-excerpt">

{!isLoggedIn ?

blog.excerpt.substring(0, 50) + "..." :

blog.excerpt

}

{!isLoggedIn && (

<span style={{ color: '#e74c3c', fontWeight: 'bold' }}>

{" "}[Login to read more]

</span>

)}

</p>

<div className="blog-info">

<span>👤 {blog.author}</span>

<span>📅 {blog.publishDate}</span>

<span>⏱️ {blog.readTime} min read</span>

</div>

{/\* Conditional stats based on user role \*/}

{(userRole === 'admin' || userRole === 'author') && (

<div className="blog-stats">

<span>👁️ {blog.viewCount} views</span>

<span>❤️ {blog.likes} likes</span>

<span>💬 {blog.comments} comments</span>

</div>

)}

<div className="blog-tags">

{blog.tags.map(tag => (

<span key={tag} className="tag">#{tag}</span>

))}

</div>

<div className="blog-actions">

{/\* Complex button logic \*/}

{!isLoggedIn ? (

<button className="read-btn" disabled>

🔒 Login to Read

</button>

) : blog.isPremium && userRole === 'reader' ? (

<button className="read-btn" disabled>

⭐ Premium Only

</button>

) : !blog.isPublished && userRole !== 'admin' ? (

<button className="read-btn" disabled>

📝 Coming Soon

</button>

) : (

<>

<button className="read-btn">

{userRole === 'premium' ? '📖 Read Premium' : '📖 Read Article'}

</button>

{isLoggedIn && (

<button className="comments-btn">

💬 {blog.comments}

</button>

)}

</>

)}

</div>

</div>

</div>

))}

</div>

)}

</div>

{/\* 6. Dynamic Statistics Panel \*/}

<div className="conditional-section">

<h3>6. Dynamic Content Statistics</h3>

<div className="example-box">

{isLoggedIn && (

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(200px, 1fr))', gap: '15px' }}>

<div style={{ textAlign: 'center', padding: '15px', background: '#ecf0f1', borderRadius: '8px' }}>

<h4>📊 Available Content</h4>

<p><strong>{getFilteredBlogs().length}</strong> articles</p>

</div>

{userRole === 'premium' && (

<div style={{ textAlign: 'center', padding: '15px', background: '#f39c12', color: 'white', borderRadius: '8px' }}>

<h4>⭐ Premium Access</h4>

<p><strong>{blogs.filter(b => b.isPremium).length}</strong> exclusive articles</p>

</div>

)}

{(userRole === 'admin' || userRole === 'author') && (

<div style={{ textAlign: 'center', padding: '15px', background: '#e74c3c', color: 'white', borderRadius: '8px' }}>

<h4>📝 Draft Content</h4>

<p><strong>{blogs.filter(b => !b.isPublished).length}</strong> unpublished</p>

</div>

)}

{userRole === 'admin' && (

<div style={{ textAlign: 'center', padding: '15px', background: '#27ae60', color: 'white', borderRadius: '8px' }}>

<h4>👑 Total Platform</h4>

<p><strong>{blogs.length}</strong> total articles</p>

</div>

)}

</div>

)}

{!isLoggedIn && (

<div style={{ textAlign: 'center', padding: '20px', background: '#e74c3c', color: 'white', borderRadius: '8px' }}>

<h4>🔐 Limited Access</h4>

<p>Log in to see detailed statistics and access more content</p>

</div>

)}

</div>

</div>

</div>

);

};

export default BlogDetails;

**//BookDetails.js**

import React, { useState } from 'react';

const BookDetails = () => {

// State for demonstrating different conditional rendering scenarios

const [userType, setUserType] = useState('guest');

const [isLoading, setIsLoading] = useState(false);

const [selectedBook, setSelectedBook] = useState(null);

const [showDetails, setShowDetails] = useState(false);

// Sample book data

const books = [

{

id: 1,

title: "Advanced React Patterns",

author: "John Doe",

genre: "Technology",

price: 45.99,

originalPrice: 59.99,

inStock: true,

stockCount: 15,

rating: 4.8,

isPremium: false,

reviews: 245

},

{

id: 2,

title: "Mastering JavaScript",

author: "Jane Smith",

genre: "Programming",

price: 39.99,

originalPrice: 49.99,

inStock: true,

stockCount: 3,

rating: 4.9,

isPremium: true,

reviews: 189

},

{

id: 3,

title: "Data Structures & Algorithms",

author: "Mike Johnson",

genre: "Computer Science",

price: 55.99,

originalPrice: 55.99,

inStock: false,

stockCount: 0,

rating: 4.7,

isPremium: true,

reviews: 332

},

{

id: 4,

title: "Web Development Fundamentals",

author: "Sarah Wilson",

genre: "Web Development",

price: 29.99,

originalPrice: 39.99,

inStock: true,

stockCount: 25,

rating: 4.6,

isPremium: false,

reviews: 156

}

];

// Simulate loading for demonstration

const handleLoadBooks = () => {

setIsLoading(true);

setTimeout(() => {

setIsLoading(false);

}, 2000);

};

const handleBookSelect = (book) => {

setSelectedBook(book);

setShowDetails(true);

};

return (

<div className="book-details">

<div className="component-header">

<h2>📚 Book Details Component</h2>

<p>Demonstrating Multiple Conditional Rendering Techniques</p>

</div>

{/\* 1. Ternary Operator - User Type Selection \*/}

<div className="conditional-section">

<h3>1. Ternary Operator - User Authentication</h3>

<div className="user-type-selector">

<p>Current User Type: <strong>{userType}</strong></p>

<div className="user-buttons">

<button

className={userType === 'guest' ? 'active' : ''}

onClick={() => setUserType('guest')}

>

Guest User

</button>

<button

className={userType === 'member' ? 'active' : ''}

onClick={() => setUserType('member')}

>

Member

</button>

<button

className={userType === 'premium' ? 'active' : ''}

onClick={() => setUserType('premium')}

>

Premium Member

</button>

</div>

</div>

<div className="example-box">

{userType === 'guest' ? (

<div className="guest-message">

🚫 Please login to view book details and make purchases

</div>

) : userType === 'member' ? (

<div className="member-message">

✅ Welcome Member! You can view all books and purchase non-premium content

</div>

) : (

<div className="premium-badge">

👑 Premium Member - Full Access to All Content!

</div>

)}

</div>

</div>

{/\* 2. Logical AND Operator \*/}

<div className="conditional-section">

<h3>2. Logical AND (&&) Operator</h3>

<div className="example-box">

{userType !== 'guest' && (

<div>

<p>✨ Exclusive member benefits:</p>

<ul>

<li>Free shipping on orders over $30</li>

<li>Early access to new releases</li>

<li>Member-only discounts</li>

</ul>

</div>

)}

{userType === 'premium' && (

<div>

<p>🌟 Premium exclusive features:</p>

<ul>

<li>Access to premium books</li>

<li>Unlimited downloads</li>

<li>Priority customer support</li>

</ul>

</div>

)}

</div>

</div>

{/\* 3. Short-circuit Evaluation with Logical OR \*/}

<div className="conditional-section">

<h3>3. Logical OR (||) Operator</h3>

<div className="example-box">

<p>

Welcome, {userType === 'guest' && 'Guest'}

{userType === 'member' && 'Valued Member'}

{userType === 'premium' && 'Premium Member'}

{!userType && 'User'}!

</p>

<p>

Your status: {(userType === 'premium' && 'VIP Access') ||

(userType === 'member' && 'Standard Access') ||

'Limited Access'}

</p>

</div>

</div>

{/\* 4. Switch Case Simulation using Object Mapping \*/}

<div className="conditional-section">

<h3>4. Switch-Case Style Conditional Rendering</h3>

<div className="example-box">

{(() => {

const userBenefits = {

guest: (

<div style={{color: '#e74c3c'}}>

<h4>Guest Benefits:</h4>

<p>• Browse book catalog</p>

<p>• Read book descriptions</p>

<p>• Create an account</p>

</div>

),

member: (

<div style={{color: '#27ae60'}}>

<h4>Member Benefits:</h4>

<p>• Purchase books</p>

<p>• Access free samples</p>

<p>• Save to wishlist</p>

<p>• Rate and review books</p>

</div>

),

premium: (

<div style={{color: '#f39c12'}}>

<h4>Premium Benefits:</h4>

<p>• All member benefits</p>

<p>• Access premium content</p>

<p>• Unlimited downloads</p>

<p>• Exclusive author events</p>

<p>• Personal recommendations</p>

</div>

)

};

return userBenefits[userType] || <p>Unknown user type</p>;

})()}

</div>

</div>

{/\* 5. Loading State Conditional Rendering \*/}

<div className="conditional-section">

<h3>5. Loading State Management</h3>

<div className="example-box">

<button onClick={handleLoadBooks} disabled={isLoading}>

{isLoading ? 'Loading Books...' : 'Load Books'}

</button>

{isLoading && (

<div className="loading-overlay">

<div className="loading-spinner">

<div className="spinner"></div>

<p>Fetching latest books...</p>

</div>

</div>

)}

</div>

</div>

{/\* 6. Nested Conditional Rendering \*/}

<div className="conditional-section">

<h3>6. Nested Conditionals</h3>

<div className="example-box">

{userType !== 'guest' ? (

<div>

{userType === 'member' ? (

<div>

<p>🎯 Recommended for Members:</p>

{books.filter(book => !book.isPremium).length > 0 ? (

<p>We found {books.filter(book => !book.isPremium).length} books perfect for you!</p>

) : (

<p>No regular books available at the moment.</p>

)}

</div>

) : (

<div>

<p>👑 All Books Available for Premium Members:</p>

{books.length > 0 ? (

<p>Browse our complete collection of {books.length} books!</p>

) : (

<p>Our collection is being updated.</p>

)}

</div>

)}

</div>

) : (

<div>

<p>🔒 Please log in to see personalized recommendations</p>

</div>

)}

</div>

</div>

{/\* 7. Element Variables \*/}

<div className="conditional-section">

<h3>7. Element Variables</h3>

<div className="example-box">

{(() => {

let statusElement;

if (userType === 'guest') {

statusElement = (

<div className="status-message guest">

<h4>Guest Mode Active</h4>

<p>Sign up today to unlock full features!</p>

</div>

);

} else if (userType === 'member') {

statusElement = (

<div className="status-message member">

<h4>Member Dashboard</h4>

<p>Welcome back! Ready to discover new books?</p>

</div>

);

} else {

statusElement = (

<div className="status-message premium">

<h4>Premium Experience</h4>

<p>Enjoy unlimited access to our entire library!</p>

</div>

);

}

return statusElement;

})()}

</div>

</div>

{/\* 8. Conditional Classes and Styles \*/}

<div className="conditional-section">

<h3>8. Conditional CSS Classes</h3>

<div className="example-box">

<div

className={`user-badge ${userType === 'premium' ? 'premium-user' : ''} ${userType === 'member' ? 'member-user' : ''}`}

style={{

backgroundColor: userType === 'premium' ? '#f39c12' : userType === 'member' ? '#27ae60' : '#e74c3c',

color: 'white',

padding: '10px',

borderRadius: '5px',

textAlign: 'center',

fontWeight: 'bold'

}}

>

{userType.toUpperCase()} STATUS

</div>

</div>

</div>

{/\* 9. Array Method Conditional Rendering \*/}

<div className="conditional-section">

<h3>9. Books Grid with Multiple Conditions</h3>

{!isLoading && (

<div className="books-grid">

{books

.filter(book => {

// Filter based on user type and premium access

if (userType === 'guest') return false;

if (userType === 'member' && book.isPremium) return false;

return true;

})

.map(book => (

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

<div className="book-image">

<div className="book-icon">📖</div>

{/\* Conditional availability badge \*/}

<div className={`availability-badge ${

book.inStock

? book.stockCount > 10

? 'in-stock'

: 'limited-stock'

: 'out-of-stock'

}`}>

{book.inStock

? book.stockCount > 10

? 'In Stock'

: `Only ${book.stockCount} left`

: 'Out of Stock'

}

</div>

</div>

<div className="book-info">

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

<p className="author">by {book.author}</p>

<p className="genre">{book.genre}</p>

{/\* Conditional pricing display \*/}

<div className="price-display">

{book.price < book.originalPrice ? (

<div className="discount-pricing">

<span className="original-price">${book.originalPrice}</span>

<span className="discounted-price">${book.price}</span>

<span className="discount-badge">

{Math.round((1 - book.price / book.originalPrice) \* 100)}% OFF

</span>

</div>

) : (

<span className="regular-price">${book.price}</span>

)}

</div>

<p>⭐ {book.rating} ({book.reviews} reviews)</p>

{/\* Conditional action buttons \*/}

{book.inStock ? (

userType === 'member' && book.isPremium ? (

<button className="premium-lock" disabled>

🔒 Premium Only

</button>

) : (

<button

className="view-details-btn"

onClick={() => handleBookSelect(book)}

>

{userType === 'premium' ? 'Download Now' : 'View Details'}

</button>

)

) : (

<button className="out-of-stock-btn" disabled>

Currently Unavailable

</button>

)}

</div>

</div>

))}

</div>

)}

{/\* Conditional empty state \*/}

{!isLoading && books.filter(book => {

if (userType === 'guest') return false;

if (userType === 'member' && book.isPremium) return false;

return true;

}).length === 0 && (

<div className="example-box">

<p>No books available for your current membership level.</p>

{userType === 'member' && (

<p>Upgrade to Premium to access our complete library!</p>

)}

</div>

)}

</div>

{/\* 10. Modal Conditional Rendering \*/}

{showDetails && selectedBook && (

<div className="book-details-modal" onClick={() => setShowDetails(false)}>

<div className="modal-content" onClick={(e) => e.stopPropagation()}>

<div className="modal-header">

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

<button

className="close-btn"

onClick={() => setShowDetails(false)}

>

✕

</button>

</div>

<div className="modal-body">

<p><strong>Author:</strong> {selectedBook.author}</p>

<p><strong>Genre:</strong> {selectedBook.genre}</p>

<p><strong>Price:</strong> ${selectedBook.price}</p>

<p><strong>Rating:</strong> ⭐ {selectedBook.rating}</p>

<p><strong>Reviews:</strong> {selectedBook.reviews}</p>

{selectedBook.isPremium && (

<div className="premium-badge">

👑 Premium Content

</div>

)}

{userType === 'premium' ? (

<button className="view-details-btn">

Add to Library

</button>

) : (

<button className="view-details-btn">

Add to Cart - ${selectedBook.price}

</button>

)}

</div>

</div>

</div>

)}

</div>

);

};

export default BookDetails;

**//CourseDetails.js**

import React, { useState } from 'react';

const CourseDetails = () => {

// State for various conditional scenarios

const [selectedLevel, setSelectedLevel] = useState('all');

const [enrollmentStatus, setEnrollmentStatus] = useState({});

const [viewMode, setViewMode] = useState('grid');

const [sortBy, setSortBy] = useState('popularity');

const [filterBy, setFilterBy] = useState('all');

// Sample course data with rich conditional properties

const courses = [

{

id: 1,

title: "React Fundamentals",

instructor: "Sarah Johnson",

level: "beginner",

price: 0,

originalPrice: 0,

duration: "6 weeks",

students: 15420,

rating: 4.8,

reviews: 2341,

isPopular: true,

category: "frontend",

skills: ["React", "JSX", "Components"],

hasCaption: true,

hasCertificate: true,

language: "English",

lastUpdated: "2024-01-15"

},

{

id: 2,

title: "Advanced JavaScript Patterns",

instructor: "Mike Chen",

level: "advanced",

price: 89.99,

originalPrice: 129.99,

duration: "8 weeks",

students: 8750,

rating: 4.9,

reviews: 1205,

isPopular: false,

category: "javascript",

skills: ["ES6+", "Design Patterns", "Performance"],

hasCaption: true,

hasCertificate: true,

language: "English",

lastUpdated: "2024-01-10"

},

{

id: 3,

title: "Full-Stack Web Development",

instructor: "Emma Wilson",

level: "intermediate",

price: 149.99,

originalPrice: 199.99,

duration: "12 weeks",

students: 12300,

rating: 4.7,

reviews: 1890,

isPopular: true,

category: "fullstack",

skills: ["Node.js", "React", "MongoDB", "Express"],

hasCaption: false,

hasCertificate: true,

language: "English",

lastUpdated: "2024-01-08"

},

{

id: 4,

title: "UI/UX Design Masterclass",

instructor: "David Rodriguez",

level: "beginner",

price: 79.99,

originalPrice: 79.99,

duration: "5 weeks",

students: 6890,

rating: 4.6,

reviews: 945,

isPopular: false,

category: "design",

skills: ["Figma", "Prototyping", "User Research"],

hasCaption: true,

hasCertificate: false,

language: "English",

lastUpdated: "2024-01-05"

},

{

id: 5,

title: "Python for Data Science",

instructor: "Lisa Park",

level: "intermediate",

price: 99.99,

originalPrice: 149.99,

duration: "10 weeks",

students: 9876,

rating: 4.8,

reviews: 1456,

isPopular: true,

category: "data-science",

skills: ["Python", "Pandas", "NumPy", "Matplotlib"],

hasCaption: true,

hasCertificate: true,

language: "English",

lastUpdated: "2024-01-12"

},

{

id: 6,

title: "Mobile App Development with React Native",

instructor: "James Wilson",

level: "advanced",

price: 119.99,

originalPrice: 159.99,

duration: "9 weeks",

students: 5432,

rating: 4.5,

reviews: 678,

isPopular: false,

category: "mobile",

skills: ["React Native", "iOS", "Android", "Redux"],

hasCaption: false,

hasCertificate: true,

language: "English",

lastUpdated: "2024-01-03"

}

];

// Simulate enrollment status

const handleEnrollment = (courseId) => {

setEnrollmentStatus(prev => ({

...prev,

[courseId]: prev[courseId] || 'enrolled'

}));

};

// Get filtered and sorted courses

const getProcessedCourses = () => {

let filteredCourses = courses.filter(course => {

// Level filter

const levelMatch = selectedLevel === 'all' || course.level === selectedLevel;

// Category filter

const categoryMatch = filterBy === 'all' || course.category === filterBy;

return levelMatch && categoryMatch;

});

// Sort courses

return filteredCourses.sort((a, b) => {

switch (sortBy) {

case 'price-low':

return a.price - b.price;

case 'price-high':

return b.price - a.price;

case 'rating':

return b.rating - a.rating;

case 'students':

return b.students - a.students;

case 'newest':

return new Date(b.lastUpdated) - new Date(a.lastUpdated);

default: // popularity

return b.students - a.students;

}

});

};

return (

<div className="course-details">

<div className="component-header">

<h2>🎓 Course Details Component</h2>

<p>Comprehensive Conditional Rendering for E-Learning Platform</p>

</div>

{/\* 1. Level-based Filtering \*/}

<div className="conditional-section">

<h3>1. Skill Level Targeting</h3>

<div className="user-type-selector">

<p>Select your experience level:</p>

<div className="level-buttons">

{['all', 'beginner', 'intermediate', 'advanced'].map(level => (

<button

key={level}

className={selectedLevel === level ? 'active' : ''}

onClick={() => setSelectedLevel(level)}

>

{level === 'all' ? '🌟 All Levels' :

level === 'beginner' ? '🌱 Beginner' :

level === 'intermediate' ? '🌿 Intermediate' : '🌳 Advanced'}

</button>

))}

</div>

</div>

{/\* Conditional recommendations based on level \*/}

<div className="example-box">

{(() => {

const recommendations = {

beginner: {

message: "Perfect for getting started! These courses include:",

features: ["Step-by-step guidance", "No prior experience needed", "Practical projects", "Community support"],

icon: "🌱",

color: "#27ae60"

},

intermediate: {

message: "Ready to level up your skills:",

features: ["Build on existing knowledge", "Real-world projects", "Advanced techniques", "Portfolio development"],

icon: "🌿",

color: "#f39c12"

},

advanced: {

message: "Master complex topics and patterns:",

features: ["Expert-level content", "Industry best practices", "Architecture patterns", "Performance optimization"],

icon: "🌳",

color: "#e74c3c"

},

all: {

message: "Explore courses at every skill level:",

features: ["Beginner to advanced paths", "Flexible learning", "All topics covered", "Complete learning journey"],

icon: "🌟",

color: "#667eea"

}

};

const current = recommendations[selectedLevel];

return (

<div style={{ borderLeft: `4px solid ${current.color}` }}>

<h4 style={{ color: current.color }}>

{current.icon} {current.message}

</h4>

<ul>

{current.features.map((feature, index) => (

<li key={index}>• {feature}</li>

))}

</ul>

</div>

);

})()}

</div>

</div>

{/\* 2. Advanced Filtering and Sorting Controls \*/}

<div className="conditional-section">

<h3>2. Smart Course Discovery</h3>

<div className="controls-section">

<div className="filters">

<div>

<label>Category: </label>

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

<option value="all">All Categories</option>

<option value="frontend">Frontend Development</option>

<option value="javascript">JavaScript</option>

<option value="fullstack">Full-Stack</option>

<option value="design">UI/UX Design</option>

<option value="data-science">Data Science</option>

<option value="mobile">Mobile Development</option>

</select>

</div>

<div>

<label>Sort by: </label>

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

<option value="popularity">Most Popular</option>

<option value="rating">Highest Rated</option>

<option value="price-low">Price: Low to High</option>

<option value="price-high">Price: High to Low</option>

<option value="students">Most Students</option>

<option value="newest">Newest First</option>

</select>

</div>

<div className="view-modes">

<button

className={viewMode === 'grid' ? 'active' : ''}

onClick={() => setViewMode('grid')}

>

📱 Grid

</button>

<button

className={viewMode === 'list' ? 'active' : ''}

onClick={() => setViewMode('list')}

>

📋 List

</button>

</div>

</div>

</div>

</div>

{/\* 3. Dynamic Course Display \*/}

<div className="conditional-section">

<h3>3. Smart Course Catalog</h3>

{/\* Conditional empty state \*/}

{getProcessedCourses().length === 0 ? (

<div className="example-box">

<h4>🔍 No courses found</h4>

<p>Try adjusting your filters or explore different skill levels.</p>

<button

onClick={() => {

setSelectedLevel('all');

setFilterBy('all');

}}

style={{

padding: '10px 20px',

background: '#667eea',

color: 'white',

border: 'none',

borderRadius: '5px',

cursor: 'pointer'

}}

>

Reset Filters

</button>

</div>

) : (

<div className={`courses-container ${viewMode}`}>

{getProcessedCourses().map(course => (

<div key={course.id} className={`course-card ${course.isPopular ? 'popular' : ''}`}>

{/\* Conditional badges \*/}

{course.isPopular && (

<div className="popular-badge">🔥 Popular</div>

)}

{course.price === 0 && (

<div className="free-badge">FREE</div>

)}

<div className="course-image">

<div className="course-icon">

{course.category === 'frontend' ? '🖥️' :

course.category === 'javascript' ? '⚡' :

course.category === 'fullstack' ? '🔗' :

course.category === 'design' ? '🎨' :

course.category === 'data-science' ? '📊' :

course.category === 'mobile' ? '📱' : '💻'}

</div>

<div className={`difficulty-badge ${course.level}`}>

{course.level.toUpperCase()}

</div>

</div>

<div className="course-content">

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

<p className="instructor">👨‍🏫 {course.instructor}</p>

<div className="course-stats">

<span>👥 {course.students.toLocaleString()} students</span>

<span>⭐ {course.rating} ({course.reviews} reviews)</span>

<span>⏰ {course.duration}</span>

</div>

{/\* Conditional pricing display \*/}

<div className="pricing">

{course.price === 0 ? (

<div className="free-price">FREE</div>

) : course.price < course.originalPrice ? (

<div className="discount-pricing">

<span className="original-price">${course.originalPrice}</span>

<span className="current-price">${course.price}</span>

<span className="discount">

{Math.round((1 - course.price / course.originalPrice) \* 100)}% OFF

</span>

</div>

) : (

<div className="current-price">${course.price}</div>

)}

</div>

{/\* Conditional course features \*/}

<div className="course-features">

{course.skills.slice(0, 3).map(skill => (

<span key={skill} className="feature">{skill}</span>

))}

{course.skills.length > 3 && (

<span className="feature">+{course.skills.length - 3} more</span>

)}

</div>

{/\* Additional features with conditional rendering \*/}

<div style={{ margin: '10px 0', fontSize: '0.9rem', color: '#7f8c8d' }}>

{course.hasCaption && <span>📝 Captions • </span>}

{course.hasCertificate && <span>🏆 Certificate • </span>}

<span>🌐 {course.language}</span>

</div>

{/\* Complex enrollment button logic \*/}

{(() => {

const enrollmentState = enrollmentStatus[course.id];

if (enrollmentState === 'enrolled') {

return (

<button className="continue-btn">

📚 Continue Learning

</button>

);

} else if (enrollmentState === 'completed') {

return (

<button className="review-btn">

⭐ Leave a Review

</button>

);

} else {

return (

<button

className="enroll-btn"

onClick={() => handleEnrollment(course.id)}

>

{course.price === 0 ? '🆓 Enroll Free' : `💳 Enroll Now - $${course.price}`}

</button>

);

}

})()}

</div>

</div>

))}

</div>

)}

</div>

{/\* 4. Learning Path Recommendations \*/}

<div className="conditional-section">

<h3>4. Personalized Learning Paths</h3>

<div className="example-box">

{(() => {

// Complex recommendation logic based on selected level

if (selectedLevel === 'beginner') {

const beginnerCourses = courses.filter(c => c.level === 'beginner');

return (

<div>

<h4>🛤️ Recommended Learning Path for Beginners:</h4>

<ol>

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

<li key={course.id} style={{ margin: '8px 0' }}>

<strong>{course.title}</strong> - {course.duration}

{course.price === 0 && <span style={{ color: '#27ae60', marginLeft: '10px' }}>✨ FREE</span>}

</li>

))}

</ol>

{beginnerCourses.length > 0 && (

<p style={{ marginTop: '15px', color: '#7f8c8d' }}>

💡 Tip: Start with the free courses to build your foundation!

</p>

)}

</div>

);

} else if (selectedLevel === 'intermediate') {

return (

<div>

<h4>🚀 Accelerated Path for Intermediate Learners:</h4>

<p>Build upon your existing knowledge with these specialized tracks:</p>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(250px, 1fr))', gap: '15px', marginTop: '15px' }}>

<div style={{ padding: '15px', background: '#ecf0f1', borderRadius: '8px' }}>

<h5>🖥️ Frontend Specialist</h5>

<p>Advanced React + UI/UX Design</p>

</div>

<div style={{ padding: '15px', background: '#ecf0f1', borderRadius: '8px' }}>

<h5>🔗 Full-Stack Developer</h5>

<p>Complete web development stack</p>

</div>

<div style={{ padding: '15px', background: '#ecf0f1', borderRadius: '8px' }}>

<h5>📊 Data-Driven Developer</h5>

<p>Python for Data Science path</p>

</div>

</div>

</div>

);

} else if (selectedLevel === 'advanced') {

const advancedCourses = courses.filter(c => c.level === 'advanced');

return (

<div>

<h4>🎯 Expert-Level Mastery:</h4>

<p>Master complex patterns and architectures:</p>

{advancedCourses.map(course => (

<div key={course.id} style={{

margin: '10px 0',

padding: '10px',

background: '#2c3e50',

color: 'white',

borderRadius: '5px'

}}>

<strong>{course.title}</strong> - Advanced {course.category.replace('-', ' ')}

<br />

<small>⭐ {course.rating} rating • {course.students.toLocaleString()} students</small>

</div>

))}

</div>

);

} else {

// All levels view

const stats = {

beginner: courses.filter(c => c.level === 'beginner').length,

intermediate: courses.filter(c => c.level === 'intermediate').length,

advanced: courses.filter(c => c.level === 'advanced').length,

free: courses.filter(c => c.price === 0).length

};

return (

<div>

<h4>🌈 Complete Learning Ecosystem:</h4>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(150px, 1fr))', gap: '15px' }}>

<div style={{ textAlign: 'center', padding: '15px', background: '#27ae60', color: 'white', borderRadius: '8px' }}>

<h4>{stats.beginner}</h4>

<p>Beginner Courses</p>

</div>

<div style={{ textAlign: 'center', padding: '15px', background: '#f39c12', color: 'white', borderRadius: '8px' }}>

<h4>{stats.intermediate}</h4>

<p>Intermediate Courses</p>

</div>

<div style={{ textAlign: 'center', padding: '15px', background: '#e74c3c', color: 'white', borderRadius: '8px' }}>

<h4>{stats.advanced}</h4>

<p>Advanced Courses</p>

</div>

<div style={{ textAlign: 'center', padding: '15px', background: '#667eea', color: 'white', borderRadius: '8px' }}>

<h4>{stats.free}</h4>

<p>Free Courses</p>

</div>

</div>

</div>

);

}

})()}

</div>

</div>

{/\* 5. Dynamic Progress Tracking \*/}

<div className="conditional-section">

<h3>5. Learning Progress Dashboard</h3>

<div className="example-box">

{Object.keys(enrollmentStatus).length === 0 ? (

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

<h4>📈 Your Learning Journey Starts Here</h4>

<p>Enroll in courses to track your progress and achievements!</p>

</div>

) : (

<div>

<h4>📊 Your Course Progress:</h4>

<div style={{ display: 'grid', gridTemplateColumns: 'repeat(auto-fit, minmax(300px, 1fr))', gap: '15px' }}>

{Object.entries(enrollmentStatus).map(([courseId, status]) => {

const course = courses.find(c => c.id === parseInt(courseId));

if (!course) return null;

return (

<div key={courseId} style={{

padding: '15px',

background: status === 'completed' ? '#27ae60' : '#3498db',

color: 'white',

borderRadius: '8px'

}}>

<h5>{course.title}</h5>

<p>Status: {status === 'enrolled' ? '📚 In Progress' : '✅ Completed'}</p>

{status === 'enrolled' && (

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

<div style={{ background: 'rgba(255,255,255,0.3)', height: '8px', borderRadius: '4px' }}>

<div style={{

background: 'white',

height: '100%',

width: `${Math.floor(Math.random() \* 80) + 20}%`,

borderRadius: '4px'

}}></div>

</div>

</div>

)}

</div>

);

})}

</div>

</div>

)}

</div>

</div>

</div>

);

};

export default CourseDetails;

**//animations.css**

/\* Animation Effects for Conditional Rendering Demo \*/

/\* Fade In Animations \*/

@keyframes fadeIn {

from {

opacity: 0;

transform: translateY(20px);

}

to {

opacity: 1;

transform: translateY(0);

}

}

@keyframes fadeInLeft {

from {

opacity: 0;

transform: translateX(-30px);

}

to {

opacity: 1;

transform: translateX(0);

}

}

@keyframes fadeInRight {

from {

opacity: 0;

transform: translateX(30px);

}

to {

opacity: 1;

transform: translateX(0);

}

}

@keyframes slideUp {

from {

transform: translateY(100%);

opacity: 0;

}

to {

transform: translateY(0);

opacity: 1;

}

}

@keyframes zoomIn {

from {

transform: scale(0.8);

opacity: 0;

}

to {

transform: scale(1);

opacity: 1;

}

}

@keyframes bounceIn {

0% {

transform: scale(0.3);

opacity: 0;

}

50% {

transform: scale(1.1);

}

70% {

transform: scale(0.9);

}

100% {

transform: scale(1);

opacity: 1;

}

}

@keyframes pulse {

0% { transform: scale(1); }

50% { transform: scale(1.05); }

100% { transform: scale(1); }

}

@keyframes glow {

0% { box-shadow: 0 0 10px rgba(102, 126, 234, 0.3); }

50% { box-shadow: 0 0 20px rgba(102, 126, 234, 0.6); }

100% { box-shadow: 0 0 10px rgba(102, 126, 234, 0.3); }

}

@keyframes shimmer {

0% {

background-position: -468px 0;

}

100% {

background-position: 468px 0;

}

}

@keyframes typing {

from { width: 0; }

to { width: 100%; }

}

@keyframes blink {

50% { border-color: transparent; }

}

@keyframes shake {

0%, 100% { transform: translateX(0); }

10%, 30%, 50%, 70%, 90% { transform: translateX(-10px); }

20%, 40%, 60%, 80% { transform: translateX(10px); }

}

@keyframes float {

0%, 100% { transform: translateY(0px); }

50% { transform: translateY(-10px); }

}

@keyframes rotate {

from { transform: rotate(0deg); }

to { transform: rotate(360deg); }

}

@keyframes gradient {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

/\* Apply Animations to Components \*/

.book-card {

animation: fadeIn 0.5s ease-out;

}

.blog-card {

animation: fadeInLeft 0.6s ease-out;

}

.course-card {

animation: fadeInRight 0.7s ease-out;

}

.conditional-section {

animation: slideUp 0.4s ease-out;

}

.premium-badge {

animation: pulse 2s infinite;

}

.popular-badge {

animation: glow 2s infinite;

}

.loading-spinner .spinner {

animation: rotate 1s linear infinite;

}

.welcome-banner {

animation: slideDown 0.5s ease-out;

}

.nav-btn:hover {

animation: float 0.6s ease-in-out;

}

.theme-toggle:hover {

animation: rotate 0.3s ease-in-out;

}

.fab:hover {

animation: bounceIn 0.3s ease-out;

}

/\* Conditional Rendering Animation Classes \*/

.fade-enter {

opacity: 0;

transform: translateY(20px);

}

.fade-enter-active {

opacity: 1;

transform: translateY(0);

transition: opacity 0.3s ease, transform 0.3s ease;

}

.fade-exit {

opacity: 1;

transform: translateY(0);

}

.fade-exit-active {

opacity: 0;

transform: translateY(-20px);

transition: opacity 0.3s ease, transform 0.3s ease;

}

.slide-enter {

transform: translateX(-100%);

}

.slide-enter-active {

transform: translateX(0);

transition: transform 0.4s ease;

}

.slide-exit {

transform: translateX(0);

}

.slide-exit-active {

transform: translateX(100%);

transition: transform 0.4s ease;

}

/\* Skeleton Loading for Conditional Rendering \*/

.skeleton {

background: linear-gradient(90deg, #f0f0f0 25%, #e0e0e0 50%, #f0f0f0 75%);

background-size: 200% 100%;

animation: shimmer 1.5s infinite;

}

.skeleton-text {

height: 20px;

border-radius: 4px;

margin-bottom: 10px;

}

.skeleton-title {

height: 30px;

border-radius: 4px;

margin-bottom: 15px;

}

.skeleton-card {

height: 200px;

border-radius: 15px;

margin-bottom: 20px;

}

/\* Error State Animations \*/

.error-shake {

animation: shake 0.5s ease-in-out;

}

.error-message {

animation: fadeIn 0.3s ease-out;

background: linear-gradient(135deg, #ff6b6b, #ff5252);

color: white;

padding: 15px;

border-radius: 10px;

margin: 10px 0;

}

.success-message {

animation: slideUp 0.4s ease-out;

background: linear-gradient(135deg, #51cf66, #40c057);

color: white;

padding: 15px;

border-radius: 10px;

margin: 10px 0;

}

/\* Tooltip Animations \*/

.tooltip {

position: relative;

}

.tooltip:hover::after {

content: attr(data-tooltip);

position: absolute;

bottom: 100%;

left: 50%;

transform: translateX(-50%);

background: #2c3e50;

color: white;

padding: 8px 12px;

border-radius: 6px;

font-size: 0.8rem;

white-space: nowrap;

animation: fadeIn 0.2s ease-out;

z-index: 1000;

}

/\* Interactive Element Animations \*/

.interactive-hover:hover {

animation: pulse 0.3s ease-in-out;

}

.button-click {

animation: bounceIn 0.2s ease-out;

}

.input-focus {

animation: glow 0.3s ease-out;

}

/\* Staggered Animation for Lists \*/

.stagger-item:nth-child(1) { animation-delay: 0.1s; }

.stagger-item:nth-child(2) { animation-delay: 0.2s; }

.stagger-item:nth-child(3) { animation-delay: 0.3s; }

.stagger-item:nth-child(4) { animation-delay: 0.4s; }

.stagger-item:nth-child(5) { animation-delay: 0.5s; }

.stagger-item:nth-child(6) { animation-delay: 0.6s; }

/\* Responsive Animation Adjustments \*/

@media (prefers-reduced-motion: reduce) {

\*, \*::before, \*::after {

animation-duration: 0.01ms !important;

animation-iteration-count: 1 !important;

transition-duration: 0.01ms !important;

}

}

@media (max-width: 768px) {

.book-card, .blog-card, .course-card {

animation-duration: 0.3s;

}

.welcome-banner {

animation-duration: 0.3s;

}

}

/\* Dark Mode Animation Overrides \*/

.dark-mode .skeleton {

background: linear-gradient(90deg, #34495e 25%, #2c3e50 50%, #34495e 75%);

background-size: 200% 100%;

animation: shimmer 1.5s infinite;

}

.dark-mode .error-message {

background: linear-gradient(135deg, #e74c3c, #c0392b);

}

.dark-mode .success-message {

background: linear-gradient(135deg, #27ae60, #229954);

}

/\* Loading Dots Animation \*/

.loading-dots::after {

content: '';

animation: loading-dots 2s infinite;

}

@keyframes loading-dots {

0%, 20% { content: '.'; }

40% { content: '..'; }

60%, 100% { content: '...'; }

}

/\* Notification Animation \*/

.notification {

animation: slideInRight 0.5s ease-out;

}

@keyframes slideInRight {

from {

transform: translateX(100%);

opacity: 0;

}

to {

transform: translateX(0);

opacity: 1;

}

}

/\* Progress Bar Animation \*/

.progress-bar {

background: linear-gradient(45deg, #667eea, #764ba2);

background-size: 200% 200%;

animation: gradient 3s ease infinite;

}

/\* Typewriter Effect \*/

.typewriter {

overflow: hidden;

border-right: 2px solid #667eea;

white-space: nowrap;

animation: typing 3s steps(40, end), blink 0.75s step-end infinite;

}

**//App.css**

/\* Global Styles and Variables \*/

:root {

--primary-color: #667eea;

--secondary-color: #764ba2;

--success-color: #27ae60;

--warning-color: #f39c12;

--error-color: #e74c3c;

--info-color: #3498db;

--light-bg: #f8f9fa;

--dark-bg: #2c3e50;

--card-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);

--border-radius: 15px;

--transition: all 0.3s ease;

}

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

line-height: 1.6;

}

/\* Main App Container \*/

.App {

min-height: 100vh;

transition: var(--transition);

}

.App.light-mode {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: #2c3e50;

}

.App.dark-mode {

background: linear-gradient(135deg, #2c3e50 0%, #34495e 100%);

color: #ecf0f1;

}

/\* Welcome Banner \*/

.welcome-banner {

background: linear-gradient(135deg, #f093fb 0%, #f5576c 100%);

color: white;

padding: 30px 20px;

text-align: center;

animation: slideDown 0.5s ease;

}

@keyframes slideDown {

from { transform: translateY(-100%); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

.welcome-content h1 {

font-size: 2.5rem;

margin-bottom: 15px;

}

.welcome-content p {

font-size: 1.2rem;

margin-bottom: 20px;

}

.close-welcome {

background: rgba(255, 255, 255, 0.2);

border: 2px solid white;

color: white;

padding: 8px 16px;

border-radius: 20px;

cursor: pointer;

transition: var(--transition);

}

.close-welcome:hover {

background: white;

color: #f5576c;

}

/\* Header \*/

.app-header {

background: white;

box-shadow: var(--card-shadow);

position: sticky;

top: 0;

z-index: 100;

}

.dark-mode .app-header {

background: #34495e;

}

.header-content {

max-width: 1400px;

margin: 0 auto;

padding: 20px;

display: flex;

justify-content: space-between;

align-items: center;

flex-wrap: wrap;

gap: 20px;

}

.logo-section h2 {

color: var(--primary-color);

font-size: 1.8rem;

margin-bottom: 5px;

}

.subtitle {

color: #7f8c8d;

font-size: 0.9rem;

}

.main-navigation {

display: flex;

gap: 10px;

flex-wrap: wrap;

}

.nav-btn {

padding: 12px 20px;

border: 2px solid transparent;

border-radius: 25px;

background: linear-gradient(135deg, var(--light-bg), #fff);

color: #2c3e50;

cursor: pointer;

transition: var(--transition);

font-weight: 500;

position: relative;

display: flex;

align-items: center;

gap: 8px;

}

.nav-btn:hover {

transform: translateY(-2px);

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);

}

.nav-btn.active {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border-color: var(--primary-color);

}

.active-indicator {

color: #f1c40f;

font-size: 0.8rem;

}

.theme-toggle {

padding: 12px;

border: none;

border-radius: 50%;

background: linear-gradient(135deg, #ffeaa7, #fab1a0);

cursor: pointer;

font-size: 1.5rem;

transition: var(--transition);

}

.theme-toggle:hover {

transform: scale(1.1) rotate(15deg);

}

/\* Breadcrumb \*/

.breadcrumb {

max-width: 1400px;

margin: 0 auto;

padding: 15px 20px;

display: flex;

align-items: center;

gap: 10px;

background: rgba(255, 255, 255, 0.1);

backdrop-filter: blur(10px);

border-radius: 0 0 15px 15px;

}

.breadcrumb-current {

font-weight: bold;

color: #f1c40f;

}

/\* Main Content \*/

.main-content {

max-width: 1400px;

margin: 0 auto;

padding: 30px 20px;

display: grid;

grid-template-columns: 1fr 300px;

gap: 30px;

}

.component-container {

background: white;

border-radius: var(--border-radius);

box-shadow: var(--card-shadow);

overflow: hidden;

}

.dark-mode .component-container {

background: #34495e;

}

/\* Component Headers \*/

.component-header {

text-align: center;

padding: 30px;

background: linear-gradient(135deg, var(--light-bg), #fff);

border-bottom: 3px solid var(--primary-color);

}

.dark-mode .component-header {

background: linear-gradient(135deg, #2c3e50, #34495e);

color: white;

}

.component-header h2 {

font-size: 2.2rem;

margin-bottom: 10px;

color: var(--primary-color);

}

.dark-mode .component-header h2 {

color: #74b9ff;

}

/\* Book Details Styles \*/

.book-details {

padding: 30px;

}

.user-type-selector {

margin-bottom: 30px;

text-align: center;

}

.user-buttons {

display: flex;

justify-content: center;

gap: 15px;

margin-top: 15px;

flex-wrap: wrap;

}

.user-buttons button {

padding: 10px 20px;

border: 2px solid #ddd;

border-radius: 20px;

background: white;

cursor: pointer;

transition: var(--transition);

}

.user-buttons button.active {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border-color: var(--primary-color);

}

.conditional-section {

margin: 30px 0;

}

.example-box {

background: var(--light-bg);

padding: 20px;

border-radius: 10px;

margin: 15px 0;

border-left: 4px solid var(--primary-color);

}

.dark-mode .example-box {

background: rgba(255, 255, 255, 0.1);

}

.guest-message, .member-message {

padding: 15px;

border-radius: 8px;

text-align: center;

font-weight: bold;

}

.guest-message {

background: #ffe6e6;

color: var(--error-color);

border: 2px solid var(--error-color);

}

.member-message {

background: #e8f5e8;

color: var(--success-color);

border: 2px solid var(--success-color);

}

.premium-badge {

background: linear-gradient(135deg, #f093fb, #f5576c);

color: white;

padding: 10px 20px;

border-radius: 20px;

text-align: center;

font-weight: bold;

}

.books-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(350px, 1fr));

gap: 25px;

margin-top: 30px;

}

.book-card {

background: white;

border-radius: var(--border-radius);

box-shadow: var(--card-shadow);

overflow: hidden;

transition: var(--transition);

border: 2px solid transparent;

}

.book-card:hover {

transform: translateY(-5px);

box-shadow: 0 15px 40px rgba(0, 0, 0, 0.2);

border-color: var(--primary-color);

}

.book-image {

position: relative;

padding: 20px;

text-align: center;

background: linear-gradient(135deg, var(--light-bg), #fff);

}

.book-icon {

font-size: 3rem;

}

.availability-badge {

position: absolute;

top: 10px;

right: 10px;

padding: 5px 10px;

border-radius: 15px;

font-size: 0.8rem;

font-weight: bold;

}

.availability-badge.in-stock {

background: var(--success-color);

color: white;

}

.availability-badge.limited-stock {

background: var(--warning-color);

color: white;

}

.availability-badge.out-of-stock {

background: var(--error-color);

color: white;

}

.book-info {

padding: 20px;

}

.book-info h4 {

color: #2c3e50;

font-size: 1.3rem;

margin-bottom: 8px;

}

.author {

color: #7f8c8d;

font-style: italic;

margin-bottom: 5px;

}

.genre {

color: var(--primary-color);

font-weight: bold;

margin-bottom: 15px;

}

.price-display {

margin: 15px 0;

}

.original-price {

text-decoration: line-through;

color: #7f8c8d;

margin-right: 10px;

}

.discounted-price {

font-weight: bold;

color: var(--success-color);

font-size: 1.2rem;

}

.discount-badge {

background: var(--error-color);

color: white;

padding: 2px 8px;

border-radius: 10px;

font-size: 0.8rem;

margin-left: 10px;

}

.view-details-btn {

width: 100%;

padding: 12px;

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border: none;

border-radius: 8px;

cursor: pointer;

transition: var(--transition);

font-weight: bold;

}

.view-details-btn:hover {

transform: translateY(-2px);

box-shadow: 0 5px 15px rgba(102, 126, 234, 0.4);

}

.premium-lock, .out-of-stock-btn {

width: 100%;

padding: 12px;

background: #bdc3c7;

color: #7f8c8d;

border: none;

border-radius: 8px;

cursor: not-allowed;

}

/\* Loading States \*/

.loading-overlay {

position: fixed;

top: 0;

left: 0;

right: 0;

bottom: 0;

background: rgba(0, 0, 0, 0.7);

display: flex;

justify-content: center;

align-items: center;

z-index: 1000;

}

.loading-spinner {

background: white;

padding: 30px;

border-radius: 15px;

text-align: center;

}

.spinner {

width: 40px;

height: 40px;

border: 4px solid #f3f3f3;

border-top: 4px solid var(--primary-color);

border-radius: 50%;

animation: spin 1s linear infinite;

margin: 0 auto 15px;

}

@keyframes spin {

0% { transform: rotate(0deg); }

100% { transform: rotate(360deg); }

}

/\* Modal Styles \*/

.book-details-modal {

position: fixed;

top: 0;

left: 0;

right: 0;

bottom: 0;

background: rgba(0, 0, 0, 0.8);

display: flex;

justify-content: center;

align-items: center;

z-index: 1000;

padding: 20px;

}

.modal-content {

background: white;

border-radius: var(--border-radius);

max-width: 600px;

width: 100%;

max-height: 90vh;

overflow-y: auto;

}

.modal-header {

display: flex;

justify-content: space-between;

align-items: center;

padding: 20px;

border-bottom: 2px solid var(--light-bg);

}

.close-btn {

background: var(--error-color);

color: white;

border: none;

border-radius: 50%;

width: 35px;

height: 35px;

cursor: pointer;

font-size: 1rem;

}

.modal-body {

padding: 20px;

}

/\* Blog Details Styles \*/

.blog-details {

padding: 30px;

}

.user-controls {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 30px;

padding: 20px;

background: var(--light-bg);

border-radius: 10px;

flex-wrap: wrap;

gap: 20px;

}

.login-btn, .logout-btn {

padding: 10px 20px;

border: none;

border-radius: 20px;

cursor: pointer;

font-weight: bold;

transition: var(--transition);

}

.login-btn {

background: linear-gradient(135deg, var(--success-color), #2ecc71);

color: white;

}

.logout-btn {

background: linear-gradient(135deg, var(--error-color), #e74c3c);

color: white;

}

.role-selector {

display: flex;

align-items: center;

gap: 10px;

}

.role-selector select {

padding: 8px 12px;

border: 2px solid #ddd;

border-radius: 5px;

font-size: 1rem;

}

.view-navigation {

display: flex;

justify-content: center;

gap: 15px;

margin-bottom: 30px;

flex-wrap: wrap;

}

.view-navigation button {

padding: 12px 24px;

border: 2px solid #ddd;

border-radius: 25px;

background: white;

cursor: pointer;

transition: var(--transition);

font-weight: 500;

}

.view-navigation button.active {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border-color: var(--primary-color);

}

.filters-section {

margin-bottom: 30px;

padding: 20px;

background: var(--light-bg);

border-radius: 10px;

}

.search-bar input {

width: 100%;

padding: 12px 20px;

border: 2px solid #ddd;

border-radius: 25px;

font-size: 1rem;

margin-bottom: 15px;

}

.category-filters {

display: flex;

gap: 10px;

flex-wrap: wrap;

}

.category-filters button {

padding: 8px 16px;

border: 2px solid #ddd;

border-radius: 20px;

background: white;

cursor: pointer;

transition: var(--transition);

}

.category-filters button.active {

background: var(--primary-color);

color: white;

border-color: var(--primary-color);

}

.blogs-grid {

display: grid;

gap: 25px;

}

.blog-card {

background: white;

border-radius: var(--border-radius);

box-shadow: var(--card-shadow);

overflow: hidden;

transition: var(--transition);

border: 2px solid transparent;

}

.blog-card:hover {

transform: translateY(-3px);

border-color: var(--primary-color);

}

.blog-header {

display: flex;

justify-content: space-between;

align-items: center;

padding: 20px 20px 0;

}

.blog-icon {

font-size: 2rem;

}

.blog-meta {

display: flex;

gap: 10px;

}

.category {

background: var(--primary-color);

color: white;

padding: 4px 12px;

border-radius: 15px;

font-size: 0.8rem;

}

.premium-badge {

background: linear-gradient(135deg, #f093fb, #f5576c);

color: white;

padding: 4px 12px;

border-radius: 15px;

font-size: 0.8rem;

}

.draft-badge {

background: var(--warning-color);

color: white;

padding: 4px 12px;

border-radius: 15px;

font-size: 0.8rem;

}

.blog-content {

padding: 20px;

}

.blog-content h3 {

color: #2c3e50;

margin-bottom: 10px;

font-size: 1.4rem;

}

.blog-excerpt {

color: #7f8c8d;

margin-bottom: 15px;

line-height: 1.6;

}

.blog-info {

display: flex;

gap: 15px;

margin-bottom: 15px;

font-size: 0.9rem;

color: #95a5a6;

flex-wrap: wrap;

}

.blog-stats {

display: flex;

gap: 15px;

margin-bottom: 15px;

}

.blog-tags {

display: flex;

gap: 8px;

flex-wrap: wrap;

margin-bottom: 15px;

}

.tag {

background: #ecf0f1;

color: #2c3e50;

padding: 4px 8px;

border-radius: 10px;

font-size: 0.8rem;

}

.blog-actions {

display: flex;

gap: 10px;

flex-wrap: wrap;

}

.read-btn, .comments-btn {

padding: 10px 15px;

border: none;

border-radius: 20px;

cursor: pointer;

font-weight: bold;

transition: var(--transition);

}

.read-btn {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

flex: 1;

}

.comments-btn {

background: var(--info-color);

color: white;

}

/\* Course Details Styles \*/

.course-details {

padding: 30px;

}

.level-buttons {

display: flex;

justify-content: center;

gap: 15px;

margin-top: 15px;

flex-wrap: wrap;

}

.level-buttons button {

padding: 10px 20px;

border: 2px solid #ddd;

border-radius: 20px;

background: white;

cursor: pointer;

transition: var(--transition);

}

.level-buttons button.active {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border-color: var(--primary-color);

}

.controls-section {

margin: 30px 0;

padding: 20px;

background: var(--light-bg);

border-radius: 10px;

}

.filters {

display: flex;

gap: 20px;

align-items: center;

flex-wrap: wrap;

}

.filters select {

padding: 8px 12px;

border: 2px solid #ddd;

border-radius: 5px;

font-size: 1rem;

}

.view-modes {

display: flex;

gap: 10px;

}

.view-modes button {

padding: 8px 16px;

border: 2px solid #ddd;

border-radius: 15px;

background: white;

cursor: pointer;

transition: var(--transition);

}

.view-modes button.active {

background: var(--primary-color);

color: white;

border-color: var(--primary-color);

}

.courses-container.grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(350px, 1fr));

gap: 25px;

}

.courses-container.list {

display: flex;

flex-direction: column;

gap: 20px;

}

.course-card {

background: white;

border-radius: var(--border-radius);

box-shadow: var(--card-shadow);

overflow: hidden;

transition: var(--transition);

border: 2px solid transparent;

position: relative;

}

.course-card:hover {

transform: translateY(-5px);

border-color: var(--primary-color);

}

.course-card.popular {

border-color: #f39c12;

}

.popular-badge {

position: absolute;

top: 10px;

left: 10px;

background: linear-gradient(135deg, #f39c12, #e67e22);

color: white;

padding: 5px 10px;

border-radius: 15px;

font-size: 0.8rem;

font-weight: bold;

z-index: 1;

}

.free-badge {

position: absolute;

top: 10px;

right: 10px;

background: linear-gradient(135deg, var(--success-color), #2ecc71);

color: white;

padding: 5px 10px;

border-radius: 15px;

font-size: 0.8rem;

font-weight: bold;

z-index: 1;

}

.course-image {

position: relative;

padding: 30px;

text-align: center;

background: linear-gradient(135deg, var(--light-bg), #fff);

}

.course-icon {

font-size: 3rem;

}

.difficulty-badge {

position: absolute;

bottom: 10px;

right: 10px;

padding: 5px 10px;

border-radius: 15px;

font-size: 0.8rem;

font-weight: bold;

background: white;

border: 2px solid;

}

.difficulty-badge.beginner {

border-color: var(--success-color);

color: var(--success-color);

}

.difficulty-badge.intermediate {

border-color: var(--warning-color);

color: var(--warning-color);

}

.difficulty-badge.advanced {

border-color: var(--error-color);

color: var(--error-color);

}

.course-content {

padding: 25px;

}

.course-content h3 {

color: #2c3e50;

margin-bottom: 10px;

font-size: 1.4rem;

}

.instructor {

color: #7f8c8d;

margin-bottom: 5px;

}

.course-stats {

display: flex;

gap: 15px;

margin: 15px 0;

font-size: 0.9rem;

flex-wrap: wrap;

}

.pricing {

margin: 15px 0;

}

.free-price {

font-size: 1.5rem;

font-weight: bold;

color: var(--success-color);

}

.discount-pricing {

display: flex;

align-items: center;

gap: 10px;

flex-wrap: wrap;

}

.original-price {

text-decoration: line-through;

color: #7f8c8d;

}

.current-price {

font-size: 1.3rem;

font-weight: bold;

color: var(--success-color);

}

.discount {

background: var(--error-color);

color: white;

padding: 2px 8px;

border-radius: 10px;

font-size: 0.8rem;

}

.course-features {

display: flex;

gap: 8px;

margin: 15px 0;

flex-wrap: wrap;

}

.feature {

background: var(--light-bg);

padding: 4px 8px;

border-radius: 10px;

font-size: 0.8rem;

color: #2c3e50;

}

.enroll-btn, .continue-btn, .review-btn {

width: 100%;

padding: 12px;

border: none;

border-radius: 8px;

cursor: pointer;

font-weight: bold;

transition: var(--transition);

margin-top: 10px;

}

.enroll-btn {

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

}

.continue-btn {

background: linear-gradient(135deg, var(--success-color), #2ecc71);

color: white;

}

.review-btn {

background: linear-gradient(135deg, var(--warning-color), #e67e22);

color: white;

}

/\* Info Panel \*/

.info-panel {

background: white;

border-radius: var(--border-radius);

box-shadow: var(--card-shadow);

padding: 25px;

height: fit-content;

position: sticky;

top: 100px;

}

.dark-mode .info-panel {

background: #34495e;

}

.info-panel h3 {

color: var(--primary-color);

margin-bottom: 20px;

text-align: center;

}

.techniques-list ul {

list-style: none;

padding-left: 0;

}

.techniques-list li {

padding: 5px 0;

color: #2c3e50;

font-size: 0.9rem;

}

.dark-mode .techniques-list li {

color: #ecf0f1;

}

.overall-stats {

margin-top: 20px;

padding: 15px;

background: linear-gradient(135deg, var(--primary-color), var(--secondary-color));

color: white;

border-radius: 10px;

text-align: center;

}

/\* Footer \*/

.app-footer {

background: white;

padding: 30px 20px;

margin-top: 50px;

box-shadow: 0 -4px 20px rgba(0, 0, 0, 0.1);

}

.dark-mode .app-footer {

background: #34495e;

}

.footer-content {

max-width: 1400px;

margin: 0 auto;

display: flex;

justify-content: space-between;

align-items: center;

flex-wrap: wrap;

gap: 20px;

}

.footer-stats {

display: flex;

gap: 20px;

font-size: 0.9rem;

color: #7f8c8d;

}

/\* Floating Action Buttons \*/

.floating-actions {

position: fixed;

bottom: 30px;

right: 30px;

display: flex;

flex-direction: column;

gap: 15px;

z-index: 50;

}

.fab {

width: 60px;

height: 60px;

border-radius: 50%;

border: none;

cursor: pointer;

font-size: 1.5rem;

box-shadow: var(--card-shadow);

transition: var(--transition);

}

.fab:hover {

transform: scale(1.1);

}

.help-btn {

background: linear-gradient(135deg, var(--info-color), #2980b9);

color: white;

}

.next-btn {

background: linear-gradient(135deg, var(--success-color), #2ecc71);

color: white;

}

/\* Status Messages \*/

.status-message {

text-align: center;

padding: 15px;

border-radius: 25px;

margin: 20px 0;

font-weight: bold;

}

.status-message.guest {

background: #ffe6e6;

color: var(--error-color);

border: 2px solid var(--error-color);

}

.status-message.member {

background: #e8f5e8;

color: var(--success-color);

border: 2px solid var(--success-color);

}

.status-message.premium {

background: linear-gradient(135deg, #f093fb, #f5576c);

color: white;

border: 2px solid #f5576c;

}

/\* Responsive Design \*/

@media (max-width: 1200px) {

.main-content {

grid-template-columns: 1fr;

}

.info-panel {

position: static;

}

}

@media (max-width: 768px) {

.header-content {

flex-direction: column;

text-align: center;

}

.main-navigation {

justify-content: center;

}

.courses-container.grid {

grid-template-columns: 1fr;

}

.books-grid {

grid-template-columns: 1fr;

}

.user-controls {

flex-direction: column;

text-align: center;

}

.filters {

flex-direction: column;

align-items: stretch;

}

.floating-actions {

bottom: 20px;

right: 20px;

}

.fab {

width: 50px;

height: 50px;

font-size: 1.2rem;

}

.footer-content {

flex-direction: column;

text-align: center;

}

.welcome-content h1 {

font-size: 1.8rem;

}

}

@media (max-width: 480px) {

.main-content {

padding: 15px 10px;

}

.book-details, .blog-details, .course-details {

padding: 15px;

}

.component-header {

padding: 20px 15px;

}

.component-header h2 {

font-size: 1.8rem;

}

.nav-btn {

padding: 10px 15px;

font-size: 0.9rem;

}

}

**//App.js**

import React, { useState } from 'react';

import './App.css';

import './styles/animations.css';

import BookDetails from './components/BookDetails';

import BlogDetails from './components/BlogDetails';

import CourseDetails from './components/CourseDetails';

function App() {

const [activeComponent, setActiveComponent] = useState('books');

const [isDarkMode, setIsDarkMode] = useState(false);

const [showWelcome, setShowWelcome] = useState(true);

// Conditional rendering helper

const renderActiveComponent = () => {

switch(activeComponent) {

case 'books':

return <BookDetails />;

case 'blogs':

return <BlogDetails />;

case 'courses':

return <CourseDetails />;

default:

return <div className="error-view">❌ Component not found</div>;

}

};

return (

<div className={`App ${isDarkMode ? 'dark-mode' : 'light-mode'}`}>

{/\* Conditional Welcome Banner \*/}

{showWelcome && (

<div className="welcome-banner">

<div className="welcome-content">

<h1>🎯 Blogger App - Conditional Rendering Showcase</h1>

<p>Explore different conditional rendering techniques across 3 components</p>

<button

onClick={() => setShowWelcome(false)}

className="close-welcome"

>

❌ Close

</button>

</div>

</div>

)}

{/\* Header with Navigation \*/}

<header className="app-header">

<div className="header-content">

<div className="logo-section">

<h2>📚 Blogger App</h2>

<span className="subtitle">Conditional Rendering Demo</span>

</div>

{/\* Navigation with Conditional Active States \*/}

<nav className="main-navigation">

<button

onClick={() => setActiveComponent('books')}

className={`nav-btn ${activeComponent === 'books' ? 'active' : ''}`}

>

📚 Book Details

{activeComponent === 'books' && <span className="active-indicator">●</span>}

</button>

<button

onClick={() => setActiveComponent('blogs')}

className={`nav-btn ${activeComponent === 'blogs' ? 'active' : ''}`}

>

📝 Blog Details

{activeComponent === 'blogs' && <span className="active-indicator">●</span>}

</button>

<button

onClick={() => setActiveComponent('courses')}

className={`nav-btn ${activeComponent === 'courses' ? 'active' : ''}`}

>

🎓 Course Details

{activeComponent === 'courses' && <span className="active-indicator">●</span>}

</button>

</nav>

{/\* Theme Toggle with Conditional Icon \*/}

<div className="theme-controls">

<button

onClick={() => setIsDarkMode(!isDarkMode)}

className="theme-toggle"

title={isDarkMode ? 'Switch to Light Mode' : 'Switch to Dark Mode'}

>

{isDarkMode ? '☀️' : '🌙'}

</button>

</div>

</div>

</header>

{/\* Breadcrumb with Conditional Rendering \*/}

<div className="breadcrumb">

<span className="breadcrumb-home">🏠 Home</span>

<span className="breadcrumb-separator">→</span>

<span className="breadcrumb-current">

{activeComponent === 'books' ? '📚 Books' :

activeComponent === 'blogs' ? '📝 Blogs' :

activeComponent === 'courses' ? '🎓 Courses' : 'Unknown'}

</span>

</div>

{/\* Main Content Area \*/}

<main className="main-content">

{/\* Conditional Loading Simulation \*/}

{(() => {

// Simulate different loading states

const loadingStates = {

books: false,

blogs: false,

courses: false

};

if (loadingStates[activeComponent]) {

return (

<div className="loading-container">

<div className="loading-spinner"></div>

<p>Loading {activeComponent}...</p>

</div>

);

}

return (

<div className="component-container">

{renderActiveComponent()}

</div>

);

})()}

</main>

{/\* Conditional Rendering Examples Info Panel \*/}

<aside className="info-panel">

<h3>🔍 Conditional Rendering Techniques Demonstrated:</h3>

<div className="techniques-list">

{/\* Conditional list based on active component \*/}

{activeComponent === 'books' && (

<div className="techniques-for-books">

<h4>📚 In Book Details:</h4>

<ul>

<li>✅ Ternary Operator (? :)</li>

<li>✅ Logical AND (&&)</li>

<li>✅ Nested Conditional Rendering</li>

<li>✅ Conditional CSS Classes</li>

<li>✅ Switch Case using Objects</li>

<li>✅ Multiple Condition Checks</li>

<li>✅ Loading State Management</li>

<li>✅ Complex Modal Rendering</li>

<li>✅ Render Props Pattern</li>

<li>✅ Element Variables</li>

</ul>

</div>

)}

{activeComponent === 'blogs' && (

<div className="techniques-for-blogs">

<h4>📝 In Blog Details:</h4>

<ul>

<li>✅ Switch-Case Rendering</li>

<li>✅ Complex Filtering Logic</li>

<li>✅ Role-Based Conditional Rendering</li>

<li>✅ Dynamic Component Rendering</li>

<li>✅ Conditional Navigation</li>

<li>✅ State-Based UI Changes</li>

<li>✅ Multi-Level Conditionals</li>

<li>✅ Permission-Based Rendering</li>

<li>✅ Search Result Conditionals</li>

<li>✅ Status Bar Rendering</li>

</ul>

</div>

)}

{activeComponent === 'courses' && (

<div className="techniques-for-courses">

<h4>🎓 In Course Details:</h4>

<ul>

<li>✅ Level-Based Access Control</li>

<li>✅ Enrollment Status Rendering</li>

<li>✅ Progress Tracking Display</li>

<li>✅ Dynamic Filtering & Sorting</li>

<li>✅ View Mode Switching</li>

<li>✅ Conditional Recommendations</li>

<li>✅ Achievement Celebrations</li>

<li>✅ Pricing Display Logic</li>

<li>✅ Feature Availability Checks</li>

<li>✅ Learning Path Guidance</li>

</ul>

</div>

)}

</div>

{/\* Overall Statistics \*/}

<div className="overall-stats">

<h4>📊 Total Techniques: 30+</h4>

<p>Each component demonstrates 10+ different conditional rendering approaches!</p>

</div>

</aside>

{/\* Footer with Conditional Content \*/}

<footer className="app-footer">

<div className="footer-content">

<div className="footer-info">

<p>© 2025 Blogger App - Conditional Rendering Showcase</p>

<p>

Currently viewing:

<strong>

{activeComponent === 'books' ? ' 📚 Book Details Component' :

activeComponent === 'blogs' ? ' 📝 Blog Details Component' :

activeComponent === 'courses' ? ' 🎓 Course Details Component' : ' Unknown Component'}

</strong>

</p>

</div>

<div className="footer-stats">

<span>Theme: {isDarkMode ? '🌙 Dark' : '☀️ Light'}</span>

<span>Active: {activeComponent}</span>

<span>Welcome: {showWelcome ? 'Shown' : 'Hidden'}</span>

</div>

</div>

</footer>

{/\* Floating Action Button with Conditional Functionality \*/}

<div className="floating-actions">

{!showWelcome && (

<button

onClick={() => setShowWelcome(true)}

className="fab help-btn"

title="Show Welcome Message"

>

❓

</button>

)}

<button

onClick={() => {

const components = ['books', 'blogs', 'courses'];

const currentIndex = components.indexOf(activeComponent);

const nextIndex = (currentIndex + 1) % components.length;

setActiveComponent(components[nextIndex]);

}}

className="fab next-btn"

title="Next Component"

>

⏭️

</button>

</div>

</div>

);

}

export default App;

**//App.test.js**

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

render(<App />);

const linkElement = screen.getByText(/learn react/i);

expect(linkElement).toBeInTheDocument();

});

**//BlogDetails.js**

import React, { useState, useEffect } from 'react';

function BlogDetails() {

const [currentView, setCurrentView] = useState('list'); // list, create, edit, view

const [selectedCategory, setSelectedCategory] = useState('all');

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

const [userRole, setUserRole] = useState('reader'); // reader, author, admin

const [searchTerm, setSearchTerm] = useState('');

const [showComments, setShowComments] = useState({});

const [selectedBlog, setSelectedBlog] = useState(null);

const blogs = [

{

id: 1,

title: "Getting Started with React Hooks",

category: "Technology",

author: "John Doe",

date: "2025-01-15",

excerpt: "Learn the fundamentals of React Hooks and how they can simplify your component logic.",

content: "React Hooks revolutionized how we write React components. In this comprehensive guide, we'll explore useState, useEffect, and custom hooks...",

image: "⚛️",

likes: 245,

comments: 23,

readTime: "5 min read",

status: "published",

isPremium: false,

tags: ["React", "JavaScript", "Frontend"]

},

{

id: 2,

title: "The Future of Artificial Intelligence",

category: "AI",

author: "Jane Smith",

date: "2025-01-10",

excerpt: "Exploring the potential impact of AI on various industries and everyday life.",

content: "Artificial Intelligence is rapidly evolving and transforming industries. From healthcare to transportation, AI is creating new possibilities...",

image: "🤖",

likes: 189,

comments: 45,

readTime: "8 min read",

status: "published",

isPremium: true,

tags: ["AI", "Technology", "Future"]

},

{

id: 3,

title: "Sustainable Living in 2025",

category: "Lifestyle",

author: "Mike Johnson",

date: "2025-01-08",

excerpt: "Practical tips for reducing your carbon footprint and living more sustainably.",

content: "Sustainability is more important than ever. Here are practical steps you can take to live more sustainably in 2025...",

image: "🌱",

likes: 156,

comments: 18,

readTime: "6 min read",

status: "published",

isPremium: false,

tags: ["Sustainability", "Environment", "Lifestyle"]

},

{

id: 4,

title: "Digital Marketing Trends",

category: "Business",

author: "Sarah Wilson",

date: "2025-01-05",

excerpt: "The latest trends shaping digital marketing strategies in the modern world.",

content: "Digital marketing continues to evolve. Stay ahead with these emerging trends and strategies...",

image: "📊",

likes: 201,

comments: 31,

readTime: "7 min read",

status: "draft",

isPremium: false,

tags: ["Marketing", "Business", "Digital"]

}

];

const categories = ['all', 'Technology', 'AI', 'Lifestyle', 'Business'];

// Filter blogs based on multiple conditions

const filteredBlogs = blogs.filter(blog => {

const matchesCategory = selectedCategory === 'all' || blog.category === selectedCategory;

const matchesSearch = blog.title.toLowerCase().includes(searchTerm.toLowerCase()) ||

blog.excerpt.toLowerCase().includes(searchTerm.toLowerCase());

const hasAccess = !blog.isPremium || isLoggedIn;

const isVisible = blog.status === 'published' || (userRole === 'admin' || userRole === 'author');

return matchesCategory && matchesSearch && hasAccess && isVisible;

});

const toggleComments = (blogId) => {

setShowComments(prev => ({

...prev,

[blogId]: !prev[blogId]

}));

};

// Conditional rendering helper functions

const renderBlogCard = (blog) => (

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

<div className="blog-header">

<span className="blog-icon">{blog.image}</span>

<div className="blog-meta">

<span className="category">{blog.category}</span>

{/\* Conditional badge rendering \*/}

{blog.isPremium && <span className="premium-badge">💎 Premium</span>}

{blog.status === 'draft' && <span className="draft-badge">📝 Draft</span>}

</div>

</div>

<div className="blog-content">

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

<p className="blog-excerpt">{blog.excerpt}</p>

<div className="blog-info">

<span className="author">By {blog.author}</span>

<span className="date">{blog.date}</span>

<span className="read-time">{blog.readTime}</span>

</div>

<div className="blog-stats">

<span className="likes">❤️ {blog.likes}</span>

<span className="comments">💬 {blog.comments}</span>

</div>

<div className="blog-tags">

{blog.tags.map(tag => (

<span key={tag} className="tag">#{tag}</span>

))}

</div>

</div>

<div className="blog-actions">

{/\* Complex conditional rendering for actions \*/}

{!isLoggedIn ? (

<button className="login-required" disabled>

🔐 Login to Read

</button>

) : blog.isPremium && userRole === 'reader' ? (

<button className="upgrade-required" disabled>

💎 Premium Required

</button>

) : (

<>

<button

onClick={() => {setSelectedBlog(blog); setCurrentView('view');}}

className="read-btn"

>

📖 Read Full Article

</button>

<button

onClick={() => toggleComments(blog.id)}

className="comments-btn"

>

{showComments[blog.id] ? '🔼 Hide Comments' : '🔽 Show Comments'}

</button>

</>

)}

{/\* Author/Admin specific actions \*/}

{(userRole === 'admin' || (userRole === 'author' && blog.author === 'Current User')) && (

<div className="admin-actions">

<button

onClick={() => {setSelectedBlog(blog); setCurrentView('edit');}}

className="edit-btn"

>

✏️ Edit

</button>

{userRole === 'admin' && (

<button className="delete-btn">

🗑️ Delete

</button>

)}

</div>

)}

</div>

{/\* Conditional comments section \*/}

{showComments[blog.id] && isLoggedIn && (

<div className="comments-section">

<h4>Comments:</h4>

{blog.comments > 0 ? (

<div className="comments-list">

{/\* Sample comments \*/}

<div className="comment">

<strong>User1:</strong> Great article! Very informative.

</div>

<div className="comment">

<strong>User2:</strong> Thanks for sharing these insights.

</div>

</div>

) : (

<p className="no-comments">No comments yet. Be the first to comment!</p>

)}

<div className="add-comment">

<textarea placeholder="Add your comment..." rows="3"></textarea>

<button className="submit-comment">💬 Post Comment</button>

</div>

</div>

)}

</div>

);

return (

<div className="blog-details">

<div className="component-header">

<h2>📝 Blog Details Component</h2>

<p>Advanced Conditional Rendering Techniques</p>

</div>

{/\* User Status and Role Selector \*/}

<div className="user-controls">

<div className="login-section">

<h3>User Status:</h3>

<button

onClick={() => setIsLoggedIn(!isLoggedIn)}

className={isLoggedIn ? 'logout-btn' : 'login-btn'}

>

{isLoggedIn ? '🚪 Logout' : '🔑 Login'}

</button>

{/\* Conditional role selector - only show when logged in \*/}

{isLoggedIn && (

<div className="role-selector">

<label>Role:</label>

<select

value={userRole}

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

>

<option value="reader">👤 Reader</option>

<option value="author">✍️ Author</option>

<option value="admin">👑 Admin</option>

</select>

</div>

)}

</div>

</div>

{/\* View Navigation \*/}

<div className="view-navigation">

<button

onClick={() => setCurrentView('list')}

className={currentView === 'list' ? 'active' : ''}

>

📋 Blog List

</button>

{/\* Conditional navigation based on user permissions \*/}

{isLoggedIn && (userRole === 'author' || userRole === 'admin') && (

<button

onClick={() => setCurrentView('create')}

className={currentView === 'create' ? 'active' : ''}

>

➕ Create Blog

</button>

)}

{userRole === 'admin' && (

<button

onClick={() => setCurrentView('analytics')}

className={currentView === 'analytics' ? 'active' : ''}

>

📊 Analytics

</button>

)}

</div>

{/\* Main Content Area with Switch-Case like rendering \*/}

<div className="main-content">

{(() => {

switch(currentView) {

case 'list':

return (

<div className="blog-list-view">

{/\* Filters Section \*/}

<div className="filters-section">

<div className="search-bar">

<input

type="text"

placeholder="🔍 Search blogs..."

value={searchTerm}

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

/>

</div>

<div className="category-filters">

{categories.map(category => (

<button

key={category}

onClick={() => setSelectedCategory(category)}

className={selectedCategory === category ? 'active' : ''}

>

{category === 'all' ? '🌐 All' : `📂 ${category}`}

</button>

))}

</div>

</div>

{/\* Results Count with Conditional Messages \*/}

<div className="results-info">

{filteredBlogs.length === 0 ? (

<div className="no-results">

{!isLoggedIn ? (

<p>🔐 Login to access more blogs</p>

) : searchTerm ? (

<p>🔍 No blogs found matching "{searchTerm}"</p>

) : (

<p>📭 No blogs available in this category</p>

)}

</div>

) : (

<p>📊 Showing {filteredBlogs.length} of {blogs.length} blogs</p>

)}

</div>

{/\* Blog Cards \*/}

<div className="blogs-grid">

{filteredBlogs.map(renderBlogCard)}

</div>

</div>

);

case 'create':

return (

<div className="create-blog-view">

<h3>✍️ Create New Blog Post</h3>

<form className="blog-form">

<input type="text" placeholder="Blog Title" />

<select>

<option value="">Select Category</option>

{categories.slice(1).map(cat => (

<option key={cat} value={cat}>{cat}</option>

))}

</select>

<textarea placeholder="Write your blog content..." rows="10"></textarea>

<div className="form-actions">

<button type="button" className="save-draft">📝 Save as Draft</button>

<button type="submit" className="publish">🚀 Publish</button>

</div>

</form>

</div>

);

case 'edit':

return (

<div className="edit-blog-view">

<h3>✏️ Edit Blog Post</h3>

{selectedBlog && (

<form className="blog-form">

<input type="text" defaultValue={selectedBlog.title} />

<select defaultValue={selectedBlog.category}>

{categories.slice(1).map(cat => (

<option key={cat} value={cat}>{cat}</option>

))}

</select>

<textarea defaultValue={selectedBlog.content} rows="10"></textarea>

<div className="form-actions">

<button type="button" onClick={() => setCurrentView('list')}>❌ Cancel</button>

<button type="submit" className="update">💾 Update</button>

</div>

</form>

)}

</div>

);

case 'view':

return (

<div className="view-blog">

{selectedBlog && (

<div className="full-blog">

<button

onClick={() => setCurrentView('list')}

className="back-btn"

>

← Back to List

</button>

<div className="blog-full-content">

<h1>{selectedBlog.title}</h1>

<div className="blog-meta-full">

<span>By {selectedBlog.author}</span>

<span>{selectedBlog.date}</span>

<span>{selectedBlog.readTime}</span>

</div>

<div className="blog-body">

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

</div>

</div>

</div>

)}

</div>

);

case 'analytics':

return (

<div className="analytics-view">

<h3>📊 Blog Analytics (Admin Only)</h3>

<div className="analytics-cards">

<div className="stat-card">

<h4>Total Blogs</h4>

<p className="stat-number">{blogs.length}</p>

</div>

<div className="stat-card">

<h4>Published</h4>

<p className="stat-number">{blogs.filter(b => b.status === 'published').length}</p>

</div>

<div className="stat-card">

<h4>Drafts</h4>

<p className="stat-number">{blogs.filter(b => b.status === 'draft').length}</p>

</div>

<div className="stat-card">

<h4>Premium Content</h4>

<p className="stat-number">{blogs.filter(b => b.isPremium).length}</p>

</div>

</div>

</div>

);

default:

return <div>🚫 Invalid view</div>;

}

})()}

</div>

{/\* Status Bar with Multiple Conditions \*/}

<div className="status-bar">

{isLoggedIn ? (

<div className="logged-in-status">

<span>✅ Logged in as {userRole}</span>

{userRole === 'admin' && <span>| 👑 Full access granted</span>}

{userRole === 'author' && <span>| ✍️ Can create and edit blogs</span>}

{userRole === 'reader' && <span>| 👁️ Read-only access</span>}

</div>

) : (

<div className="guest-status">

<span>👥 Browsing as guest | Limited access</span>

</div>

)}

</div>

</div>

);

}

export default BlogDetails;

**//BookDetails.js**

import React, { useState } from 'react';

function BookDetails() {

const [selectedBook, setSelectedBook] = useState(null);

const [showDetails, setShowDetails] = useState(false);

const [userType, setUserType] = useState('guest'); // guest, member, premium

const [isLoading, setIsLoading] = useState(false);

const books = [

{

id: 1,

title: "The Great Gatsby",

author: "F. Scott Fitzgerald",

genre: "Classic Fiction",

price: 299,

rating: 4.5,

pages: 180,

published: "1925",

description: "A classic American novel exploring themes of wealth, love, and the American Dream in the Jazz Age.",

image: "📚",

availability: "In Stock",

reviews: 1250,

isPremium: false

},

{

id: 2,

title: "To Kill a Mockingbird",

author: "Harper Lee",

genre: "Drama",

price: 350,

rating: 4.8,

pages: 281,

published: "1960",

description: "A powerful story of racial injustice and childhood innocence in the American South.",

image: "📖",

availability: "In Stock",

reviews: 2100,

isPremium: false

},

{

id: 3,

title: "1984",

author: "George Orwell",

genre: "Dystopian Fiction",

price: 450,

rating: 4.7,

pages: 328,

published: "1949",

description: "A dystopian social science fiction novel about totalitarian control and surveillance.",

image: "📕",

availability: "Limited Stock",

reviews: 1890,

isPremium: true

},

{

id: 4,

title: "Pride and Prejudice",

author: "Jane Austen",

genre: "Romance",

price: 320,

rating: 4.6,

pages: 432,

published: "1813",

description: "A romantic novel about manners, upbringing, morality, education, and marriage.",

image: "💕",

availability: "Out of Stock",

reviews: 1650,

isPremium: false

}

];

const handleBookSelect = (book) => {

setIsLoading(true);

setTimeout(() => {

setSelectedBook(book);

setShowDetails(true);

setIsLoading(false);

}, 1000);

};

const renderStars = (rating) => {

const stars = [];

const fullStars = Math.floor(rating);

const hasHalfStar = rating % 1 !== 0;

for (let i = 0; i < fullStars; i++) {

stars.push('⭐');

}

if (hasHalfStar) {

stars.push('⭐');

}

return stars.join('');

};

return (

<div className="book-details">

<div className="component-header">

<h2>📚 Book Details Component</h2>

<p>Demonstrating Various Conditional Rendering Techniques</p>

</div>

{/\* User Type Selector \*/}

<div className="user-type-selector">

<h3>Select User Type:</h3>

<div className="user-buttons">

<button

onClick={() => setUserType('guest')}

className={userType === 'guest' ? 'active' : ''}

>

👥 Guest

</button>

<button

onClick={() => setUserType('member')}

className={userType === 'member' ? 'active' : ''}

>

🎯 Member

</button>

<button

onClick={() => setUserType('premium')}

className={userType === 'premium' ? 'active' : ''}

>

💎 Premium

</button>

</div>

</div>

{/\* Conditional Rendering Example 1: Ternary Operator \*/}

<div className="conditional-section">

<h3>🔄 Conditional Rendering Examples:</h3>

{/\* 1. Simple Ternary Operator \*/}

<div className="example-box">

<h4>1. Ternary Operator:</h4>

{userType === 'guest' ? (

<div className="guest-message">

🚫 Limited access - Please register to view all books

</div>

) : (

<div className="member-message">

✅ Welcome {userType}! You have full access

</div>

)}

</div>

{/\* 2. Logical AND (&&) Operator \*/}

<div className="example-box">

<h4>2. Logical AND (&&) Operator:</h4>

{userType === 'premium' && (

<div className="premium-badge">

💎 Premium Member - Access to exclusive books!

</div>

)}

{books.some(book => book.isPremium) && userType !== 'premium' && (

<div className="upgrade-message">

🔒 Some books require Premium membership

</div>

)}

</div>

{/\* 3. Nested Conditional Rendering \*/}

<div className="example-box">

<h4>3. Nested Conditional Rendering:</h4>

{userType !== 'guest' ? (

userType === 'premium' ? (

<div className="premium-features">

<h5>Premium Features:</h5>

<ul>

<li>✅ Access to all books</li>

<li>✅ Exclusive premium titles</li>

<li>✅ Free shipping</li>

<li>✅ Early access to new releases</li>

</ul>

</div>

) : (

<div className="member-features">

<h5>Member Features:</h5>

<ul>

<li>✅ Access to most books</li>

<li>✅ Member discounts</li>

<li>❌ Premium titles locked</li>

</ul>

</div>

)

) : (

<div className="guest-info">

<p>📋 Register now to unlock member benefits!</p>

</div>

)}

</div>

</div>

{/\* Book List with Multiple Conditional Rendering \*/}

<div className="books-grid">

{books.map(book => (

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

<div className="book-image">

<span className="book-icon">{book.image}</span>

{/\* 4. Conditional CSS Classes \*/}

<div className={`availability-badge ${

book.availability === 'In Stock' ? 'in-stock' :

book.availability === 'Limited Stock' ? 'limited-stock' : 'out-of-stock'

}`}>

{book.availability}

</div>

</div>

<div className="book-info">

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

<p className="author">by {book.author}</p>

<p className="genre">{book.genre}</p>

{/\* 5. Switch Case using Objects \*/}

<div className="price-display">

{(() => {

switch(userType) {

case 'guest':

return <span className="price">₹{book.price}</span>;

case 'member':

return (

<div>

<span className="original-price">₹{book.price}</span>

<span className="discounted-price">₹{Math.round(book.price \* 0.9)}</span>

<span className="discount-badge">10% OFF</span>

</div>

);

case 'premium':

return (

<div>

<span className="original-price">₹{book.price}</span>

<span className="discounted-price">₹{Math.round(book.price \* 0.8)}</span>

<span className="discount-badge">20% OFF</span>

</div>

);

default:

return <span className="price">₹{book.price}</span>;

}

})()}

</div>

<div className="rating">

{renderStars(book.rating)} ({book.reviews} reviews)

</div>

{/\* 6. Multiple Condition Check \*/}

{book.isPremium && userType !== 'premium' ? (

<button className="premium-lock" disabled>

🔒 Premium Only

</button>

) : book.availability === 'Out of Stock' ? (

<button className="out-of-stock-btn" disabled>

📦 Out of Stock

</button>

) : (

<button

onClick={() => handleBookSelect(book)}

className="view-details-btn"

disabled={isLoading}

>

{isLoading ? '⏳ Loading...' : '👁️ View Details'}

</button>

)}

</div>

</div>

))}

</div>

{/\* 7. Conditional Rendering with Loading State \*/}

{isLoading && (

<div className="loading-overlay">

<div className="loading-spinner">

<div className="spinner"></div>

<p>Loading book details...</p>

</div>

</div>

)}

{/\* 8. Complex Conditional Rendering for Selected Book \*/}

{showDetails && selectedBook && !isLoading && (

<div className="book-details-modal">

<div className="modal-content">

<div className="modal-header">

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

<button

onClick={() => {setShowDetails(false); setSelectedBook(null);}}

className="close-btn"

>

❌

</button>

</div>

<div className="modal-body">

<div className="book-meta">

<p><strong>Author:</strong> {selectedBook.author}</p>

<p><strong>Genre:</strong> {selectedBook.genre}</p>

<p><strong>Pages:</strong> {selectedBook.pages}</p>

<p><strong>Published:</strong> {selectedBook.published}</p>

<p><strong>Rating:</strong> {renderStars(selectedBook.rating)} ({selectedBook.reviews} reviews)</p>

</div>

<div className="book-description">

<h4>Description:</h4>

<p>{selectedBook.description}</p>

</div>

{/\* 9. Render Props Pattern Simulation \*/}

<div className="action-buttons">

{userType === 'guest' ? (

<button className="register-btn">

📝 Register to Purchase

</button>

) : selectedBook.availability === 'Out of Stock' ? (

<button className="notify-btn">

🔔 Notify When Available

</button>

) : (

<div className="purchase-actions">

<button className="add-to-cart-btn">

🛒 Add to Cart

</button>

<button className="buy-now-btn">

⚡ Buy Now

</button>

</div>

)}

</div>

</div>

</div>

</div>

)}

{/\* 10. Element Variables \*/}

{(() => {

let statusMessage;

if (userType === 'guest') {

statusMessage = <div className="status-message guest">🚪 Browsing as Guest</div>;

} else if (userType === 'member') {

statusMessage = <div className="status-message member">👤 Logged in as Member</div>;

} else {

statusMessage = <div className="status-message premium">💎 Premium Member Active</div>;

}

return statusMessage;

})()}

</div>

);

}

export default BookDetails;

**//CourseDetails.js**

import React, { useState, useEffect } from 'react';

function CourseDetails() {

const [enrollmentStatus, setEnrollmentStatus] = useState('none'); // none, enrolled, completed

const [userLevel, setUserLevel] = useState('beginner'); // beginner, intermediate, advanced

const [selectedCourse, setSelectedCourse] = useState(null);

const [viewMode, setViewMode] = useState('grid'); // grid, list, detailed

const [filterDifficulty, setFilterDifficulty] = useState('all');

const [sortBy, setSortBy] = useState('popularity');

const [showPrerequisites, setShowPrerequisites] = useState({});

const [completedLessons, setCompletedLessons] = useState({});

const courses = [

{

id: 1,

title: "React Fundamentals",

instructor: "Sarah Johnson",

difficulty: "beginner",

duration: "8 weeks",

price: 2999,

originalPrice: 4999,

rating: 4.8,

students: 12450,

image: "⚛️",

category: "Frontend Development",

description: "Master the fundamentals of React including components, state, and props.",

prerequisites: ["HTML", "CSS", "Basic JavaScript"],

skills: ["React Components", "JSX", "State Management", "Event Handling"],

lessons: 45,

projects: 5,

certificate: true,

language: "English",

lastUpdated: "2025-01-01",

isFree: false,

isPopular: true,

hasVideoContent: true,

downloadableResources: true

},

{

id: 2,

title: "Advanced JavaScript",

instructor: "Mike Chen",

difficulty: "advanced",

duration: "12 weeks",

price: 0,

originalPrice: 0,

rating: 4.9,

students: 8790,

image: "🚀",

category: "Programming",

description: "Deep dive into advanced JavaScript concepts, async programming, and modern ES6+ features.",

prerequisites: ["JavaScript Basics", "ES6", "DOM Manipulation"],

skills: ["Async/Await", "Promises", "Closures", "Prototypes"],

lessons: 60,

projects: 8,

certificate: true,

language: "English",

lastUpdated: "2024-12-15",

isFree: true,

isPopular: false,

hasVideoContent: true,

downloadableResources: false

},

{

id: 3,

title: "Python for Beginners",

instructor: "Dr. Emily Watson",

difficulty: "beginner",

duration: "6 weeks",

price: 1999,

originalPrice: 2999,

rating: 4.6,

students: 15670,

image: "🐍",

category: "Programming",

description: "Start your programming journey with Python. Learn syntax, data structures, and basic programming concepts.",

prerequisites: ["None"],

skills: ["Python Syntax", "Data Types", "Functions", "OOP"],

lessons: 35,

projects: 4,

certificate: true,

language: "English",

lastUpdated: "2025-01-10",

isFree: false,

isPopular: true,

hasVideoContent: true,

downloadableResources: true

},

{

id: 4,

title: "Data Science with R",

instructor: "Prof. David Kumar",

difficulty: "intermediate",

duration: "10 weeks",

price: 3999,

originalPrice: 5999,

rating: 4.7,

students: 5420,

image: "📊",

category: "Data Science",

description: "Learn statistical analysis and data visualization using R programming language.",

prerequisites: ["Statistics Basics", "Mathematics"],

skills: ["R Programming", "Data Analysis", "Statistics", "Visualization"],

lessons: 55,

projects: 6,

certificate: true,

language: "English",

lastUpdated: "2024-11-20",

isFree: false,

isPopular: false,

hasVideoContent: false,

downloadableResources: true

}

];

// Complex filtering logic

const filteredCourses = courses.filter(course => {

const matchesDifficulty = filterDifficulty === 'all' || course.difficulty === filterDifficulty;

const matchesLevel = userLevel === 'beginner' ?

course.difficulty !== 'advanced' :

userLevel === 'intermediate' ?

course.difficulty !== 'beginner' :

true;

return matchesDifficulty && matchesLevel;

});

// Sorting logic with conditional rendering

const sortedCourses = [...filteredCourses].sort((a, b) => {

switch(sortBy) {

case 'popularity':

return b.students - a.students;

case 'rating':

return b.rating - a.rating;

case 'price':

return a.price - b.price;

case 'duration':

return parseInt(a.duration) - parseInt(b.duration);

default:

return 0;

}

});

const togglePrerequisites = (courseId) => {

setShowPrerequisites(prev => ({

...prev,

[courseId]: !prev[courseId]

}));

};

const markLessonComplete = (courseId, lessonIndex) => {

setCompletedLessons(prev => ({

...prev,

[`${courseId}\_${lessonIndex}`]: !prev[`${courseId}\_${lessonIndex}`]

}));

};

const renderStars = (rating) => {

return '⭐'.repeat(Math.floor(rating)) + (rating % 1 ? '⭐' : '');

};

// Component for course card rendering

const CourseCard = ({ course, viewMode }) => (

<div className={`course-card ${viewMode} ${course.isPopular ? 'popular' : ''}`}>

{/\* Conditional popularity badge \*/}

{course.isPopular && <div className="popular-badge">🔥 Popular</div>}

{/\* Conditional free badge \*/}

{course.isFree && <div className="free-badge">🆓 Free</div>}

<div className="course-image">

<span className="course-icon">{course.image}</span>

{/\* Conditional difficulty indicator \*/}

<div className={`difficulty-badge ${course.difficulty}`}>

{course.difficulty === 'beginner' ? '🟢' :

course.difficulty === 'intermediate' ? '🟡' : '🔴'}

{course.difficulty}

</div>

</div>

<div className="course-content">

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

<p className="instructor">👨‍🏫 {course.instructor}</p>

<p className="category">📂 {course.category}</p>

{/\* Conditional description based on view mode \*/}

{viewMode !== 'grid' && (

<p className="description">{course.description}</p>

)}

<div className="course-stats">

<span className="rating">

{renderStars(course.rating)} ({course.rating})

</span>

<span className="students">👥 {course.students.toLocaleString()}</span>

<span className="duration">⏱️ {course.duration}</span>

</div>

{/\* Conditional pricing display \*/}

<div className="pricing">

{course.isFree ? (

<span className="free-price">FREE</span>

) : course.originalPrice > course.price ? (

<div className="discount-pricing">

<span className="original-price">₹{course.originalPrice}</span>

<span className="current-price">₹{course.price}</span>

<span className="discount">

{Math.round(((course.originalPrice - course.price) / course.originalPrice) \* 100)}% OFF

</span>

</div>

) : (

<span className="regular-price">₹{course.price}</span>

)}

</div>

{/\* Conditional features display \*/}

<div className="course-features">

{course.hasVideoContent && <span className="feature">🎥 Video</span>}

{course.downloadableResources && <span className="feature">📁 Resources</span>}

{course.certificate && <span className="feature">🏆 Certificate</span>}

</div>

{/\* Conditional prerequisites \*/}

{course.prerequisites.length > 0 && course.prerequisites[0] !== "None" && (

<div className="prerequisites-section">

<button

onClick={() => togglePrerequisites(course.id)}

className="prerequisites-toggle"

>

{showPrerequisites[course.id] ? '🔼' : '🔽'} Prerequisites

</button>

{showPrerequisites[course.id] && (

<div className="prerequisites-list">

{course.prerequisites.map((prereq, index) => (

<span key={index} className="prerequisite">{prereq}</span>

))}

</div>

)}

</div>

)}

{/\* Action buttons with conditional rendering \*/}

<div className="course-actions">

{enrollmentStatus === 'none' ? (

<>

<button

onClick={() => {setSelectedCourse(course); setEnrollmentStatus('enrolled');}}

className="enroll-btn"

>

{course.isFree ? '📚 Enroll Free' : '💳 Enroll Now'}

</button>

<button

onClick={() => setSelectedCourse(course)}

className="preview-btn"

>

👁️ Preview

</button>

</>

) : enrollmentStatus === 'enrolled' && selectedCourse?.id === course.id ? (

<div className="enrolled-actions">

<button

onClick={() => setViewMode('detailed')}

className="continue-btn"

>

▶️ Continue Learning

</button>

<span className="enrollment-status">✅ Enrolled</span>

</div>

) : enrollmentStatus === 'completed' && selectedCourse?.id === course.id ? (

<div className="completed-actions">

<button className="review-btn">🔄 Review Course</button>

<button className="certificate-btn">🏆 Download Certificate</button>

<span className="completion-status">🎉 Completed</span>

</div>

) : (

<button

onClick={() => {setSelectedCourse(course); setEnrollmentStatus('enrolled');}}

className="enroll-btn"

>

{course.isFree ? '📚 Enroll Free' : '💳 Enroll Now'}

</button>

)}

</div>

</div>

</div>

);

return (

<div className="course-details">

<div className="component-header">

<h2>🎓 Course Details Component</h2>

<p>Comprehensive Conditional Rendering Examples</p>

</div>

{/\* User Level and Status Controls \*/}

<div className="user-controls">

<div className="level-selector">

<h3>Your Level:</h3>

<div className="level-buttons">

{['beginner', 'intermediate', 'advanced'].map(level => (

<button

key={level}

onClick={() => setUserLevel(level)}

className={userLevel === level ? 'active' : ''}

>

{level === 'beginner' ? '🟢 Beginner' :

level === 'intermediate' ? '🟡 Intermediate' : '🔴 Advanced'}

</button>

))}

</div>

</div>

<div className="enrollment-simulator">

<h3>Enrollment Status:</h3>

<select

value={enrollmentStatus}

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

>

<option value="none">Not Enrolled</option>

<option value="enrolled">Enrolled</option>

<option value="completed">Completed</option>

</select>

</div>

</div>

{/\* Conditional Welcome Message \*/}

<div className="welcome-section">

{(() => {

if (userLevel === 'beginner') {

return (

<div className="beginner-welcome">

<h3>🌟 Welcome to Learning!</h3>

<p>Perfect courses for beginners are highlighted. Start your journey!</p>

</div>

);

} else if (userLevel === 'intermediate') {

return (

<div className="intermediate-welcome">

<h3>🚀 Ready for the Next Level!</h3>

<p>Great! You can access beginner and intermediate courses.</p>

</div>

);

} else {

return (

<div className="advanced-welcome">

<h3>🎯 Expert Path</h3>

<p>Access all courses including advanced level content.</p>

</div>

);

}

})()}

</div>

{/\* Filters and Controls \*/}

<div className="controls-section">

<div className="filters">

<div className="difficulty-filter">

<label>Filter by Difficulty:</label>

<select

value={filterDifficulty}

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

>

<option value="all">All Levels</option>

<option value="beginner">Beginner</option>

<option value="intermediate">Intermediate</option>

<option value="advanced">Advanced</option>

</select>

</div>

<div className="sort-options">

<label>Sort by:</label>

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

<option value="popularity">Popularity</option>

<option value="rating">Rating</option>

<option value="price">Price</option>

<option value="duration">Duration</option>

</select>

</div>

<div className="view-modes">

<button

onClick={() => setViewMode('grid')}

className={viewMode === 'grid' ? 'active' : ''}

>

🔲 Grid

</button>

<button

onClick={() => setViewMode('list')}

className={viewMode === 'list' ? 'active' : ''}

>

📋 List

</button>

</div>

</div>

</div>

{/\* Results Summary with Conditional Messages \*/}

<div className="results-summary">

{sortedCourses.length === 0 ? (

<div className="no-results">

<p>😔 No courses match your current level and filters.</p>

<button onClick={() => {setFilterDifficulty('all'); setUserLevel('advanced');}}>

🔓 Show All Courses

</button>

</div>

) : (

<div className="results-info">

<p>

📊 Showing {sortedCourses.length} courses

{filterDifficulty !== 'all' && ` (${filterDifficulty} level)`}

{userLevel !== 'advanced' && ` suitable for ${userLevel} level`}

</p>

{/\* Conditional recommendations \*/}

{userLevel === 'beginner' && sortedCourses.some(c => c.difficulty === 'advanced') && (

<p className="level-warning">

⚠️ Some advanced courses are hidden. Level up to access them!

</p>

)}

</div>

)}

</div>

{/\* Main Content Area \*/}

{viewMode !== 'detailed' ? (

<div className={`courses-container ${viewMode}`}>

{sortedCourses.map(course => (

<CourseCard key={course.id} course={course} viewMode={viewMode} />

))}

</div>

) : selectedCourse ? (

/\* Detailed Course View \*/

<div className="detailed-course-view">

<button

onClick={() => setViewMode('grid')}

className="back-to-courses"

>

← Back to Courses

</button>

<div className="course-detail-header">

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

<div className="course-progress">

{enrollmentStatus === 'enrolled' && (

<div className="progress-section">

<h4>📚 Course Progress</h4>

<div className="lessons-list">

{Array.from({length: selectedCourse.lessons}, (\_, i) => (

<div key={i} className="lesson-item">

<input

type="checkbox"

checked={completedLessons[`${selectedCourse.id}\_${i}`] || false}

onChange={() => markLessonComplete(selectedCourse.id, i)}

/>

<span className={completedLessons[`${selectedCourse.id}\_${i}`] ? 'completed' : ''}>

Lesson {i + 1}: Sample Lesson Title

</span>

</div>

))}

</div>

{Object.keys(completedLessons).filter(key =>

key.startsWith(`${selectedCourse.id}\_`) && completedLessons[key]

).length === selectedCourse.lessons && (

<div className="completion-celebration">

<h3>🎉 Congratulations! Course Completed!</h3>

<button

onClick={() => setEnrollmentStatus('completed')}

className="mark-complete-btn"

>

🏆 Claim Certificate

</button>

</div>

)}

</div>

)}

</div>

</div>

</div>

) : null}

{/\* Statistics Section with Conditional Data \*/}

<div className="statistics-section">

<h3>📈 Learning Statistics</h3>

<div className="stats-grid">

<div className="stat-item">

<h4>Available Courses</h4>

<p>{sortedCourses.length}</p>

</div>

<div className="stat-item">

<h4>Free Courses</h4>

<p>{sortedCourses.filter(c => c.isFree).length}</p>

</div>

<div className="stat-item">

<h4>Your Level Access</h4>

<p>{userLevel === 'beginner' ? 'Basic' : userLevel === 'intermediate' ? 'Standard' : 'Full'}</p>

</div>

<div className="stat-item">

<h4>Enrollment Status</h4>

<p>{enrollmentStatus === 'none' ? 'Browsing' : enrollmentStatus}</p>

</div>

</div>

</div>

{/\* Conditional Footer Messages \*/}

<div className="footer-messages">

{enrollmentStatus === 'none' && (

<p className="cta-message">🚀 Enroll in a course to start your learning journey!</p>

)}

{enrollmentStatus === 'enrolled' && selectedCourse && (

<p className="encouragement">💪 Keep going! You're enrolled in {selectedCourse.title}</p>

)}

{enrollmentStatus === 'completed' && (

<p className="completion-message">🎓 Amazing! You've completed a course. Ready for the next challenge?</p>

)}

</div>

</div>

);

}

export default CourseDetails;

**//index.css**

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

**//index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**//reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

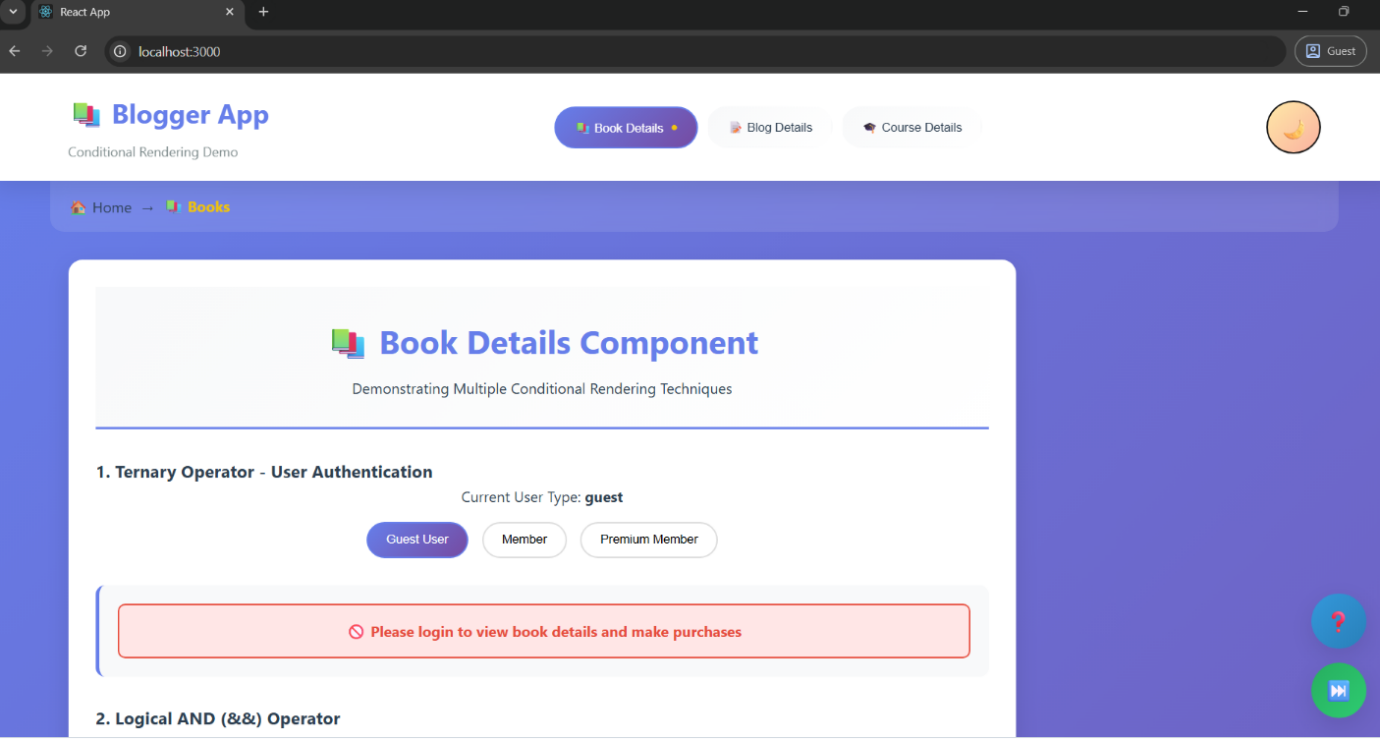
getTTFB(onPerfEntry);

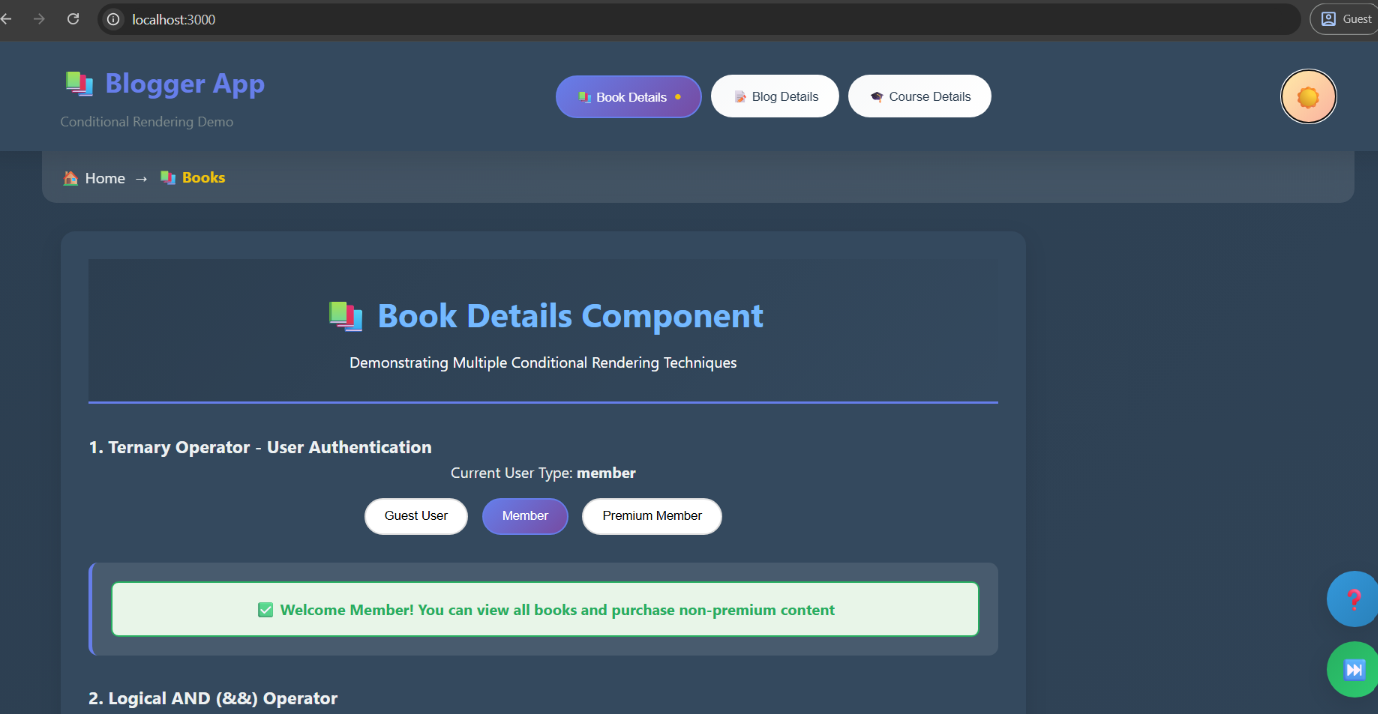
});

}

};

export default reportWebVitals;

**OUTPUT:**

****