Week-7  
  
9-REACTJS-HOL  
Components/EvenPlayers.js  
import React from 'react';

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

return (

<div>

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

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

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

</div>

);

}

export default EvenPlayers;  
Components/Listof.js  
import React from 'react';

function Listof({ IndianPlayers }) {

return (

<div>

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

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

))}

</div>

);

}

export default Listof;

Components/ListofPlayers.js  
import React from 'react';

function ListofPlayers({ players }) {

return (

<div>

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

<div key={index}>

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

</div>

))}

</div>

);

}

export default ListofPlayers;

Components/OddPlayers.js  
import React from 'react';

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

return (

<div>

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

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

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

</div>

);

}

export default OddPlayers;

Components/Scorebelow70.js  
import React from 'react';

function Scorebelow70({ players }) {

const players70 = [];

players.map((item) => {

if (item.score < 70) {

players70.push(item);

}

});

return (

<div>

{players70.map((item, index) => (

<div key={index}>

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

</div>

))}

</div>

);

}

export default Scorebelow70;  
  
App.js  
import React from 'react';

import ListofPlayers from './Components/ListofPlayers';

import Scorebelow70 from './Components/Scorebelow70';

import OddPlayers from './Components/OddPlayers';

import ListofIndianPlayers from './Components/Listof';

import EvenPlayers from './Components/EvenPlayers';

const players = [

{ name: 'Jack', score: 90 },

{ name: 'John', score: 65 },

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

{ name: 'Rahul', score: 45 },

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

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

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

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

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

{ name: 'Elon', score: 78 },

{ name: 'Musk', score: 55 },

];

const IndianTeam = ['Sachin', 'Dhoni2', 'Virat', 'Rohit4', 'Yuvaraj5'];

const T20Players = ['First Player', 'Second Player', 'Third Player'];

const RanjiTrophyPlayers = ['Fourth Player', 'Fifth Player', 'Sixth Player'];

const IndianPlayers = [...T20Players, ...RanjiTrophyPlayers];

const flag = true;

function App() {

if (flag === true) {

return (

<div>

<h1>List of Players:</h1>

<ListofPlayers players={players} />

<hr />

<h1>List of Players having Scores Less than 70</h1>

<Scorebelow70 players={players} />

</div>

);

} else {

return (

<div>

<h1>Indian Team</h1>

<h1> Odd Players </h1>

<OddPlayers players={IndianTeam} />

<hr />

<h1> Even Players </h1>

<EvenPlayers players={IndianTeam} />

<hr />

<h1>List of Indian Players Merged:</h1>

<ListofIndianPlayers IndianPlayers={IndianPlayers} />

</div>

);

}

}

export default App;

Index.js  
import React from 'react';

import ReactDOM from 'react-dom/client'; // 👈 Notice the '/client' part

import App from './App';

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

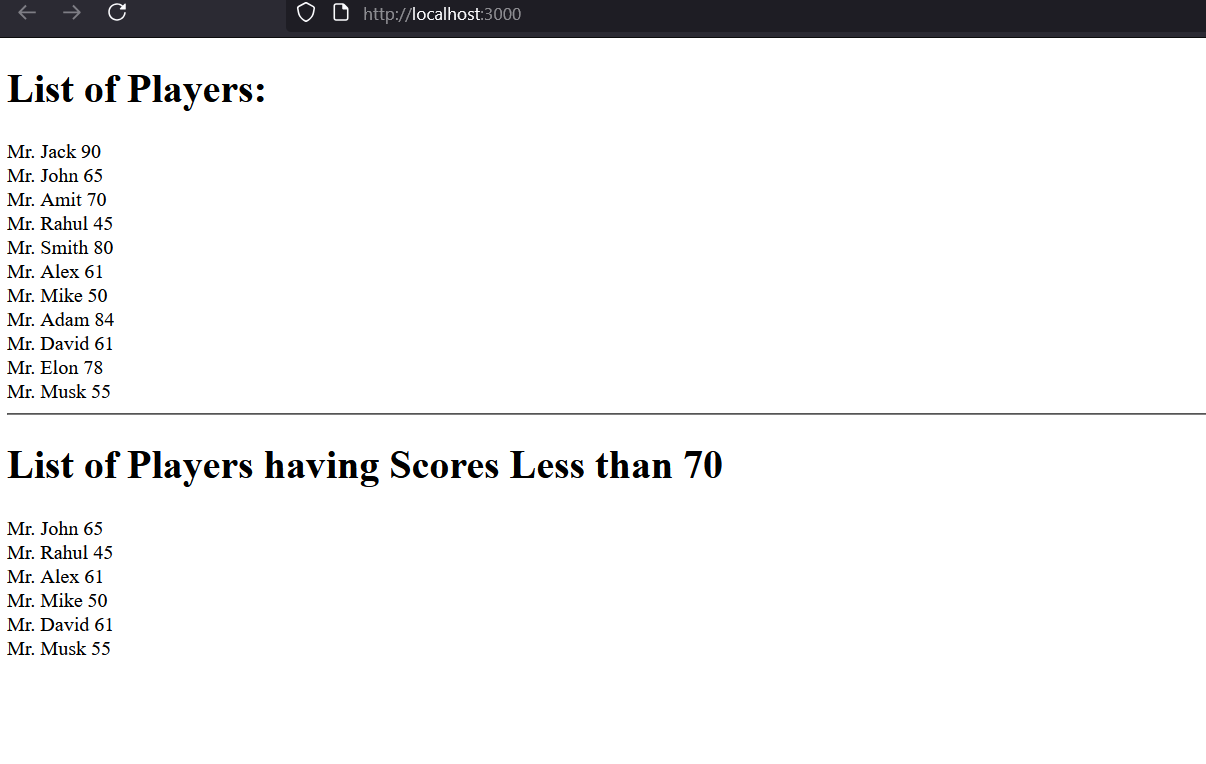
root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

OUTPUT:  
  
  
  
  
  
  
  
  
  
  
  
  
10-ReactJS-HOL  
App.js  
import React from 'react';

// Sample list of office spaces

const officeSpaces = [

{

id: 1,

name: "Indiranagar Workspace",

rent: 45000,

address: "123 MG Road, Bangalore",

image: "<https://images.unsplash.com/photo-1600585154340-be6161a56a0c?auto=format&fit=crop&w=600&q=80>"

},

{

id: 2,

name: "Koramangala Co-working",

rent: 70000,

address: "456 5th Block, Bangalore",

image: "<https://images.unsplash.com/photo-1600585154340-be6161a56a0c?auto=format&fit=crop&w=600&q=80>"

},

{

id: 3,

name: "Whitefield Tech Park",

rent: 58000,

address: "789 ITPL Road, Bangalore",

image: "<https://images.unsplash.com/photo-1600585154340-be6161a56a0c?auto=format&fit=crop&w=600&q=80>"

}

];

function App() {

return (

<div style={{ padding: '20px', fontFamily: 'Arial' }}>

{/\* Page Heading \*/}

<h1>Office Space Rental App</h1>

{/\* Loop through each office space \*/}

{officeSpaces.map((office) => (

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

border: '1px solid #ccc',

borderRadius: '8px',

padding: '16px',

marginBottom: '20px',

maxWidth: '600px'

}}>

{/\* Office Image \*/}

<img

src={office.image}

alt={office.name}

style={{ width: '100%', height: '300px', objectFit: 'cover', borderRadius: '4px' }}

/>

{/\* Office Details \*/}

<h2>{office.name}</h2>

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

<p>

<strong>Rent:</strong>{" "}

<span style={{ color: office.rent < 60000 ? 'red' : 'green' }}>

₹{office.rent.toLocaleString()}

</span>

</p>

</div>

))}

</div>

);

}

export default App;

App.css

.app-container {

padding: 20px;

font-family: 'Segoe UI', sans-serif;

text-align: center;

}

.heading {

margin-bottom: 30px;

font-size: 2em;

color: #333;

}

.card-container {

display: flex;

justify-content: center;

flex-wrap: wrap;

gap: 25px;

}

.card {

width: 280px;

border-radius: 12px;

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.12);

overflow: hidden;

transition: transform 0.3s;

background-color: #fff;

}

.card:hover {

transform: scale(1.05);

}

.office-image {

width: 100%;

height: 160px;

object-fit: cover;

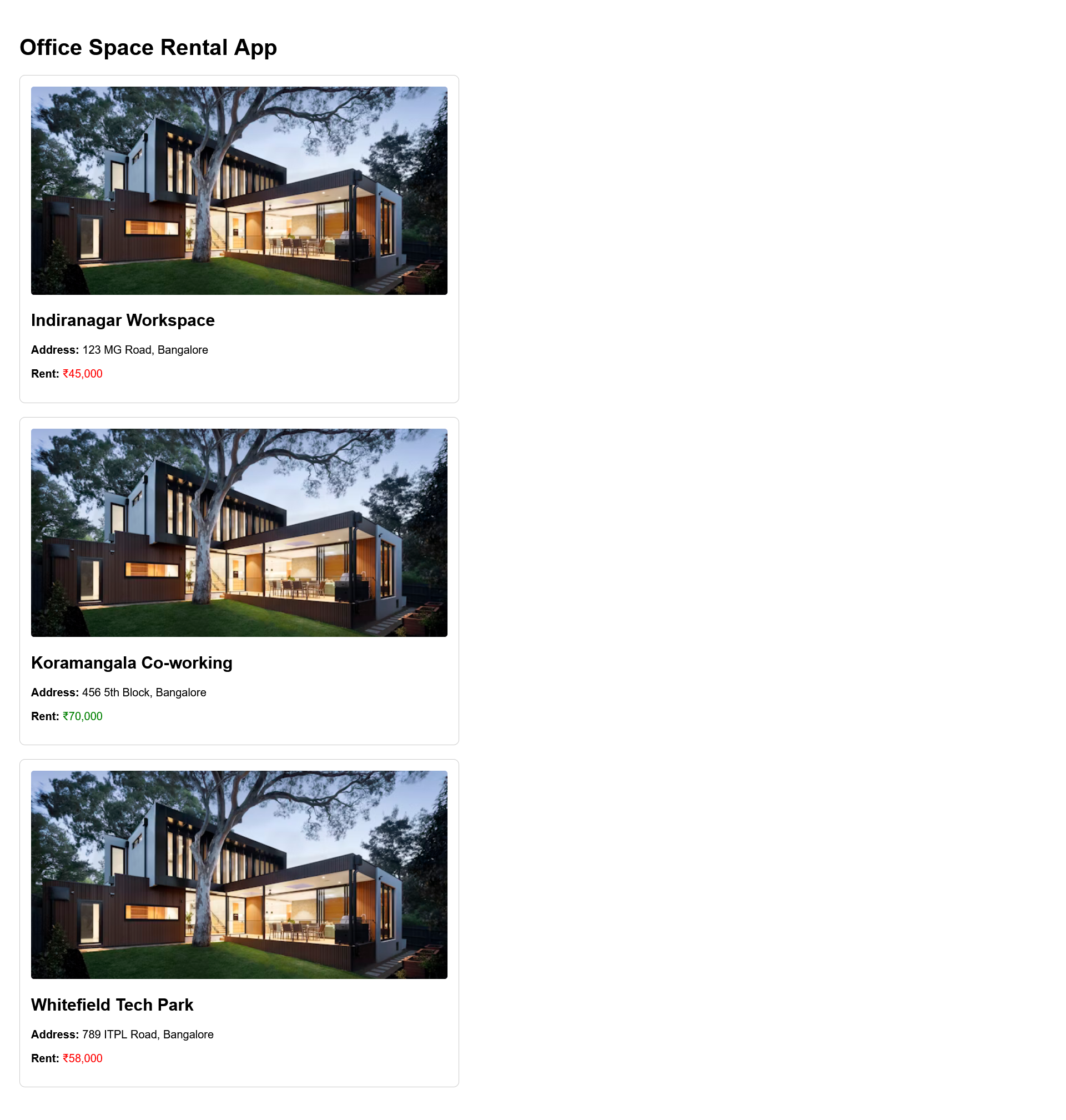
}

.card-content {

padding: 15px;

text-align: left;

}

OUTPUT:  
  
  
  
  
  
  
  
  
  
  
  
  
11-ReactJS-HOL  
  
App.js  
import React, { useState } from 'react';

function EventExamplesApp() {

// State for counter example

const [count, setCount] = useState(0);

// State for currency converter

const [amount, setAmount] = useState('');

const [currency, setCurrency] = useState('EUR');

const [convertedAmount, setConvertedAmount] = useState(null);

// Counter increment handler (handles multiple methods)

const handleIncrement = () => {

incrementValue();

sayHello();

};

const incrementValue = () => {

setCount(prevCount => prevCount + 1);

};

const decrementValue = () => {

setCount(prevCount => prevCount - 1);

};

const sayHello = () => {

alert('Hello! This is a static message.');

};

// Welcome button handler

const handleWelcome = (message) => {

alert(message);

};

// Synthetic event handler

const handleSyntheticClick = (e) => {

// e is the synthetic event

alert('I was clicked (synthetic event)');

console.log(e); // You can inspect the synthetic event in console

};

// Currency converter handler

const handleSubmit = (e) => {

e.preventDefault();

if (!amount || isNaN(amount)) {

alert('Please enter a valid amount');

return;

}

const amountNum = parseFloat(amount);

let result;

if (currency === 'EUR') {

// Convert INR to EUR (approximate rate)

result = amountNum \* 0.011;

setConvertedAmount(`${amount} INR = ${result.toFixed(2)} EUR`);

} else {

// Convert EUR to INR (approximate rate)

result = amountNum \* 89.5;

setConvertedAmount(`${amount} EUR = ${result.toFixed(2)} INR`);

}

};

return (

<div className="app" style={{ padding: '20px', maxWidth: '600px', margin: '0 auto' }}>

<h1>events example!!!</h1>

{/\* Counter Example \*/}

<section style={{ marginBottom: '30px', padding: '20px', border: '1px solid #ccc' }}>

<h2>Counter Example</h2>

<p>Current Count: {count}</p>

<button onClick={handleIncrement} style={{ marginRight: '10px' }}>

Increment (calls multiple methods)

</button>

<button onClick={decrementValue}>

Decrement

</button>

</section>

{/\* Welcome Button Example \*/}

<section style={{ marginBottom: '30px', padding: '20px', border: '1px solid #ccc' }}>

<h2>Welcome Button</h2>

<button onClick={() => handleWelcome('Welcome!')}>

Say Welcome

</button>

</section>

{/\* Synthetic Event Example \*/}

<section style={{ marginBottom: '30px', padding: '20px', border: '1px solid #ccc' }}>

<h2>Synthetic Event</h2>

<button onClick={handleSyntheticClick}>

Click for Synthetic Event

</button>

</section>

{/\* Currency Converter \*/}

<section style={{ padding: '20px', border: '1px solid #ccc' }}>

<h2>Currency Converter</h2>

<form onSubmit={handleSubmit}>

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

<label htmlFor="amount" style={{ display: 'block', marginBottom: '5px' }}>

Amount:

</label>

<input

type="text"

id="amount"

value={amount}

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

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

/>

</div>

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

<label htmlFor="currency" style={{ display: 'block', marginBottom: '5px' }}>

Currency:

</label>

<select

id="currency"

value={currency}

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

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

>

<option value="EUR">INR to EUR</option>

<option value="INR">EUR to INR</option>

</select>

</div>

<button type="submit" style={{ padding: '10px 15px', backgroundColor: '#4CAF50', color: 'white', border: 'none' }}>

Convert

</button>

</form>

{convertedAmount && (

<div style={{ marginTop: '15px', padding: '10px', backgroundColor: '#f8f8f8' }}>

<p>Conversion Result: {convertedAmount}</p>

</div>

)}

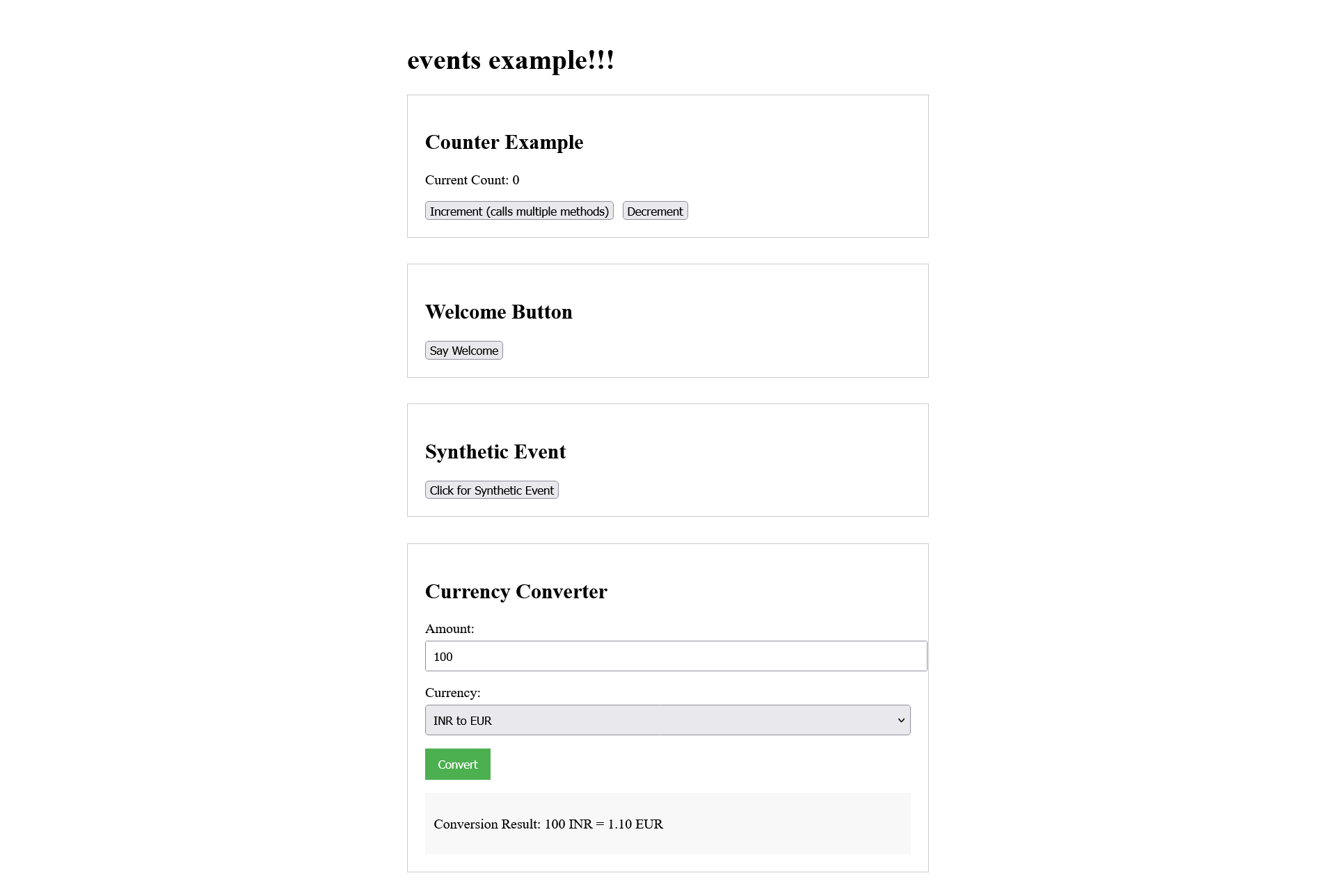
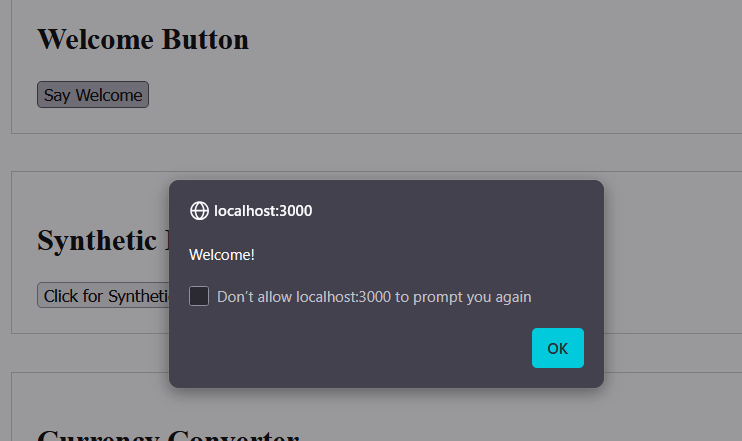
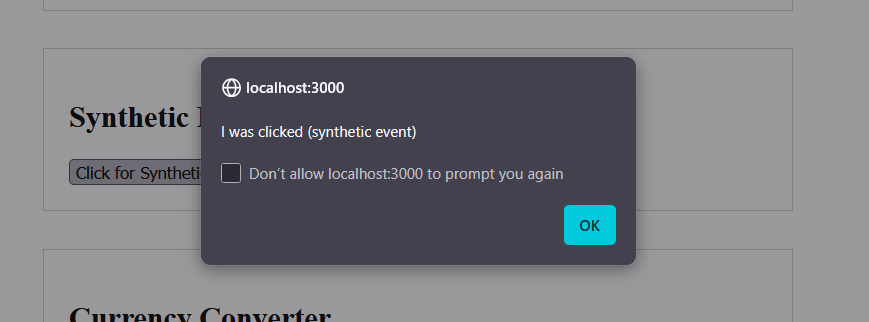
</section>

</div>

);

}

export default EventExamplesApp;

OUTPUT:  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
12-ReactJS-HOL  
App.js  
  
import React, { useState } from 'react';

import './App.css';

function TicketBookingApp() {

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

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

// Sample flight data

const flights = [

{ id: 1, airline: 'Delta', departure: '10:00 AM', arrival: '2:00 PM', price: '$250' },

{ id: 2, airline: 'United', departure: '12:00 PM', arrival: '4:30 PM', price: '$300' },

{ id: 3, airline: 'American', departure: '3:00 PM', arrival: '7:15 PM', price: '$275' },

];

const handleLogin = (e) => {

e.preventDefault();

const formData = new FormData(e.target);

const user = formData.get('username');

if (user) {

setUsername(user);

setIsLoggedIn(true);

}

};

const handleLogout = () => {

setIsLoggedIn(false);

setUsername('');

};

// Guest view - can only browse flights

const guestView = (

<div className="guest-view">

<h2>Welcome Guest</h2>

<p>Please sign up or login to book tickets.</p>

<div className="login-form">

<h3>Login</h3>

<form onSubmit={handleLogin}>

<div className="form-group">

<label htmlFor="username">Username:</label>

<input

type="text"

id="username"

name="username"

required

/>

</div>

<div className="form-group">

<label htmlFor="password">Password:</label>

<input

type="password"

id="password"

name="password"

required

/>

</div>

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

</form>

</div>

<div className="flight-list">

<h3>Available Flights</h3>

<ul>

{flights.map(flight => (

<li key={flight.id}>

<strong>{flight.airline}</strong> - Departs: {flight.departure}, Arrives: {flight.arrival} ({flight.price})

</li>

))}

</ul>

</div>

</div>

);

// User view - can book tickets

const userView = (

<div className="user-view">

<h2>Welcome back, {username}!</h2>

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

<div className="flight-list">

<h3>Available Flights - Book Now!</h3>

<ul>

{flights.map(flight => (

<li key={flight.id}>

<div className="flight-info">

<strong>{flight.airline}</strong> - Departs: {flight.departure}, Arrives: {flight.arrival} ({flight.price})

</div>

<button className="book-btn">Book This Flight</button>

</li>

))}

</ul>

</div>

</div>

);

return (

<div className="app">

<header>

<h1>Flight Ticket Booking</h1>

<div className="auth-status">

{isLoggedIn ? (

<p>Logged in as {username}</p>

) : (

<p>Please sign up.</p>

)}

</div>

</header>

<main>

{/\* Conditional rendering based on login status \*/}

{isLoggedIn ? userView : guestView}

</main>

</div>

);

}

export default TicketBookingApp;

App.css  
body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #f5f5f5;

}

.app {

max-width: 800px;

margin: 0 auto;

padding: 20px;

background-color: white;

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

}

header {

text-align: center;

margin-bottom: 30px;

padding-bottom: 20px;

border-bottom: 1px solid #eee;

}

.auth-status {

margin-top: 10px;

color: #666;

}

.guest-view, .user-view {

padding: 20px;

border-radius: 5px;

background-color: #f9f9f9;

}

.login-form {

max-width: 400px;

margin: 20px auto;

padding: 20px;

background-color: white;

border: 1px solid #ddd;

border-radius: 5px;

}

.form-group {

margin-bottom: 15px;

}

.form-group label {

display: block;

margin-bottom: 5px;

font-weight: bold;

}

.form-group input {

width: 100%;

padding: 8px;

box-sizing: border-box;

border: 1px solid #ddd;

border-radius: 4px;

}

button {

background-color: #4CAF50;

color: white;

border: none;

padding: 10px 15px;

border-radius: 4px;

cursor: pointer;

font-size: 16px;

}

button:hover {

background-color: #45a049;

}

.logout-btn {

background-color: #f44336;

margin-bottom: 20px;

}

.logout-btn:hover {

background-color: #d32f2f;

}

.flight-list {

margin-top: 30px;

}

.flight-list ul {

list-style: none;

padding: 0;

}

.flight-list li {

padding: 15px;

margin-bottom: 10px;

background-color: white;

border: 1px solid #ddd;

border-radius: 5px;

display: flex;

justify-content: space-between;

align-items: center;

}

.book-btn {

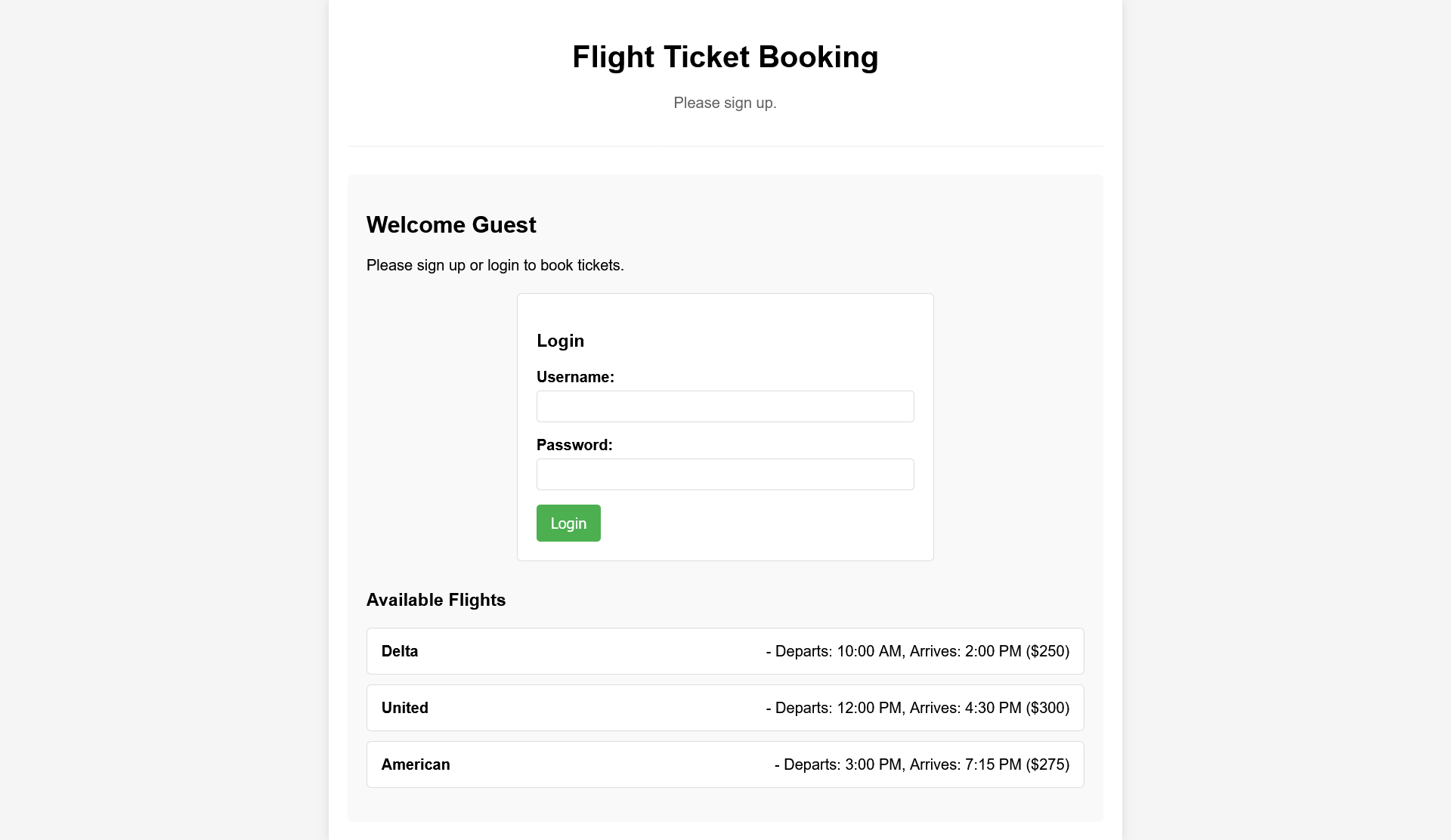
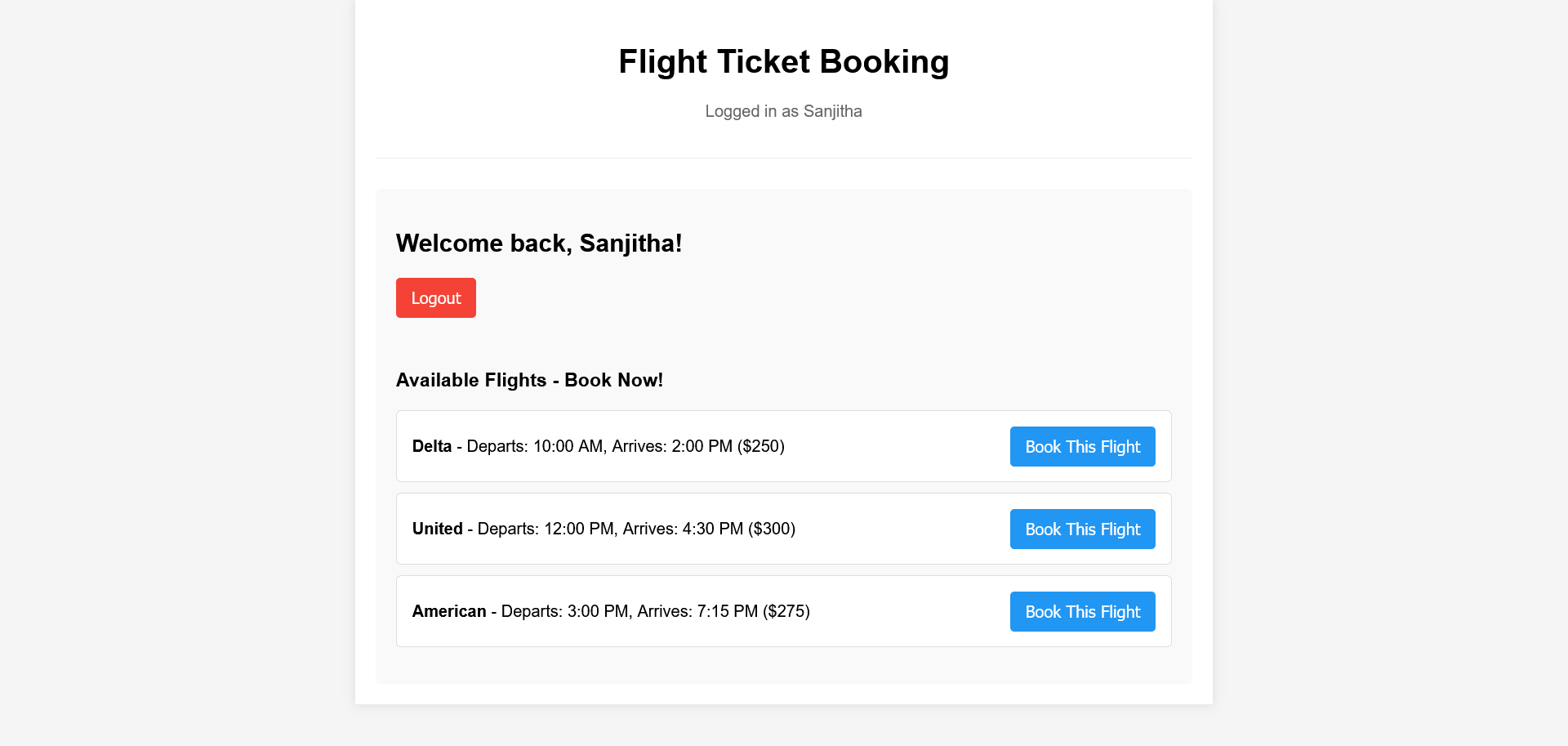
background-color: #2196F3;

}

.book-btn:hover {

background-color: #0b7dda;

}

OUTPUT:  
  
  


13-ReactJS-HOL  
  
App.js  
import React, { useState } from 'react';

import './App.css';

// Course Details Component

const CourseDetails = ({ isVisible, courses }) => {

// Method 1: Early return (if statement)

if (!isVisible) return null;

return (

<div className="card course-card">

<h2 className="card-header">Course Details</h2>

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

<div key={`course-${index}`} className="item-container course-item">

<h3 className="item-title course-title">{course.name}</h3>

<p className="item-date">{course.date}</p>

{/\* Method 2: Logical AND operator (&&) \*/}

{course.description && (

<p className="item-description">{course.description}</p>

)}

{/\* Method 3: Multiple conditions with logical operators \*/}

{course.level && course.duration && (

<div className="course-meta">

<span className="meta-item">Level: {course.level}</span>

<span className="meta-item">Duration: {course.duration}</span>

</div>

)}

</div>

))}

{/\* Method 4: Array length conditional \*/}

{courses.length === 0 && (

<p className="no-data">No courses available</p>

)}

</div>

);

};

// Book Details Component

const BookDetails = ({ isVisible, books }) => {

// Method 5: Function-based conditional rendering

const renderBookLevel = (pages) => {

if (pages > 700) return 'Advanced';

if (pages > 500) return 'Intermediate';

return 'Beginner';

};

const renderBookBadge = (pages) => {

let badgeClass = 'book-badge ';

if (pages > 700) badgeClass += 'badge-advanced';

else if (pages > 500) badgeClass += 'badge-intermediate';

else badgeClass += 'badge-beginner';

return (

<span className={badgeClass}>

{renderBookLevel(pages)}

</span>

);

};

return (

<div className="card book-card">

<h2 className="card-header">Book Details</h2>

{/\* Method 6: Ternary operator with complex expression \*/}

{isVisible ? (

<div>

{books.length > 0 ? (

books.map((book) => (

<div key={`book-${book.id}`} className="item-container book-item">

<h3 className="item-title book-title">{book.title}</h3>

<p className="item-pages">Pages: {book.pages}</p>

{/\* Method 7: Inline conditional with ternary \*/}

<p className="book-category">

Category: {book.pages > 600 ? 'Technical Reference' : 'Learning Guide'}

</p>

{renderBookBadge(book.pages)}

{/\* Method 8: Nested ternary operators \*/}

<div className={`book-difficulty ${

book.pages > 700 ? 'difficulty-hard' :

book.pages > 500 ? 'difficulty-medium' : 'difficulty-easy'

}`}>

{book.pages > 700 ? 'Comprehensive Study Required' :

book.pages > 500 ? 'Moderate Reading Time' : 'Quick Read'}

</div>

</div>

))

) : (

<p className="no-data">No books available</p>

)}

</div>

) : (

<div className="hidden-message">

<p>📚 Book details are currently hidden</p>

<p className="subtitle">Toggle the switch to view books</p>

</div>

)}

</div>

);

};

// Blog Details Component

const BlogDetails = ({ isVisible, blogs }) => {

// Method 9: Switch-case like conditional rendering using object mapping

const getCategoryInfo = (category) => {

const categoryMap = {

tutorial: { class: 'category-tutorial', icon: '📖', label: 'Tutorial' },

news: { class: 'category-news', icon: '📰', label: 'News' },

review: { class: 'category-review', icon: '⭐', label: 'Review' },

general: { class: 'category-general', icon: '📝', label: 'General' }

};

return categoryMap[category] || categoryMap.general;

};

// Method 10: Complex function with multiple conditions

const renderBlogContent = () => {

if (!blogs || blogs.length === 0) {

return <p className="no-data">No blogs available at the moment</p>;

}

return blogs.map((blog) => {

const categoryInfo = getCategoryInfo(blog.category);

return (

<div key={`blog-${blog.id}`} className="item-container blog-item">

<h3 className="item-title blog-title">{blog.title}</h3>

<p className="blog-author">By: {blog.author}</p>

<p className="item-description blog-content">{blog.content}</p>

{/\* Method 11: IIFE (Immediately Invoked Function Expression) for complex logic \*/}

{(() => {

switch (blog.category) {

case 'tutorial':

return (

<div className="blog-meta tutorial-meta">

<span className={categoryInfo.class}>

{categoryInfo.icon} {categoryInfo.label}

</span>

<span className="reading-time">⏱️ 5 min read</span>

</div>

);

case 'news':

return (

<div className="blog-meta news-meta">

<span className={categoryInfo.class}>

{categoryInfo.icon} {categoryInfo.label}

</span>

<span className="publish-date">📅 Latest</span>

</div>

);

case 'review':

return (

<div className="blog-meta review-meta">

<span className={categoryInfo.class}>

{categoryInfo.icon} {categoryInfo.label}

</span>

<span className="rating">⭐⭐⭐⭐⭐</span>

</div>

);

default:

return (

<span className={categoryInfo.class}>

{categoryInfo.icon} {categoryInfo.label}

</span>

);

}

})()}

{/\* Method 12: Conditional rendering based on content length \*/}

{blog.content.length > 50 && (

<div className="content-indicator">

<span className="long-content-badge">Detailed Article</span>

</div>

)}

</div>

);

});

};

return (

<div className="card blog-card">

<h2 className="card-header">Blog Details</h2>

{/\* Method 13: Complex conditional with multiple logical operators \*/}

{isVisible && blogs && blogs.length > 0 ? (

renderBlogContent()

) : !isVisible ? (

<div className="hidden-message">

<p>✍️ Blog details are currently hidden</p>

<p className="subtitle">Enable blogs to see the latest posts</p>

</div>

) : (

<div className="empty-state">

<p className="no-data">📝 No blog posts found</p>

<p className="subtitle">Check back later for new content!</p>

</div>

)}

</div>

);

};

// Main App Component

const App = () => {

const [activeSection, setActiveSection] = useState('all');

const [showCourses, setShowCourses] = useState(true);

const [showBooks, setShowBooks] = useState(true);

const [showBlogs, setShowBlogs] = useState(true);

const [viewMode, setViewMode] = useState('grid'); // New state for demo

// Sample data with more comprehensive information

const courses = [

{

name: 'Angular',

date: '4/5/2021',

description: 'Learn Angular framework from basics to advanced concepts including TypeScript, RxJS, and NgRx',

level: 'Intermediate',

duration: '8 weeks'

},

{

name: 'React',

date: '6/3/2021',

description: 'Master React with hooks, context, and modern patterns. Build scalable applications.',

level: 'Beginner to Advanced',

duration: '10 weeks'

},

{

name: 'Vue.js',

date: '8/15/2021',

description: 'Progressive JavaScript framework for building user interfaces',

level: 'Beginner',

duration: '6 weeks'

}

];

const books = [

{

id: 1,

title: 'Master React',

pages: 670

},

{

id: 2,

title: 'Deep Dive into Angular 11',

pages: 800

},

{

id: 3,

title: 'Mongo Essentials',

pages: 450

},

{

id: 4,

title: 'Advanced JavaScript Patterns',

pages: 520

}

];

const blogs = [

{

id: 1,

title: 'React Learning',

author: 'Stephen Biz',

content: 'Welcome to learning React! This comprehensive guide will take you through all the essential concepts.',

category: 'tutorial'

},

{

id: 2,

title: 'Installation',

author: 'Schewzdenier',

content: 'You can install React from npm. Here\'s a step-by-step guide to get started.',

category: 'tutorial'

},

{

id: 3,

title: 'React vs Angular 2024',

author: 'Tech Reviewer',

content: 'Comparing modern frontend frameworks and their ecosystem in 2024.',

category: 'review'

},

{

id: 4,

title: 'Latest React 18 Features',

author: 'News Reporter',

content: 'Breaking: New features announced in React 18 release.',

category: 'news'

}

];

// Method 14: Object-based conditional rendering with functions

const sectionRenderers = {

courses: () => <CourseDetails isVisible={showCourses} courses={courses} />,

books: () => <BookDetails isVisible={showBooks} books={books} />,

blogs: () => <BlogDetails isVisible={showBlogs} blogs={blogs} />,

all: () => (

<div className={`content-grid ${viewMode === 'list' ? 'list-view' : 'grid-view'}`}>

<CourseDetails isVisible={showCourses} courses={courses} />

<BookDetails isVisible={showBooks} books={books} />

<BlogDetails isVisible={showBlogs} blogs={blogs} />

</div>

)

};

// Method 15: Array-based conditional rendering with filtering

const getActiveSections = () => {

return [

showCourses && 'Courses',

showBooks && 'Books',

showBlogs && 'Blogs'

].filter(Boolean);

};

const handleSectionChange = (section) => {

setActiveSection(section);

};

return (

<div className="app-container">

<div className="main-wrapper">

<header className="app-header">

<h1 className="app-title">📚 Blogger App</h1>

<p className="app-subtitle">Conditional Rendering Demonstration</p>

</header>

{/\* Control Panel \*/}

<div className="control-panel">

<h2 className="control-title">🎛️ Control Panel</h2>

{/\* Section Selection \*/}

<div className="control-group">

<label className="control-label">Select Section:</label>

<div className="button-group">

{['all', 'courses', 'books', 'blogs'].map((section) => (

<button

key={section}

onClick={() => handleSectionChange(section)}

className={`section-btn ${activeSection === section ? 'active' : ''}`}

>

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

</button>

))}

</div>

</div>

{/\* View Mode Toggle \*/}

<div className="control-group">

<label className="control-label">View Mode:</label>

<div className="toggle-group">

<button

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

className={`toggle-btn ${viewMode === 'grid' ? 'active' : ''}`}

>

🔳 Grid

</button>

<button

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

className={`toggle-btn ${viewMode === 'list' ? 'active' : ''}`}

>

📋 List

</button>

</div>

</div>

{/\* Individual toggles with Method 16: State-based conditional styling \*/}

<div className="control-group">

<label className="control-label">Toggle Sections:</label>

<div className="checkbox-group">

{[

{ key: 'courses', state: showCourses, setter: setShowCourses, label: 'Show Courses', icon: '🎓' },

{ key: 'books', state: showBooks, setter: setShowBooks, label: 'Show Books', icon: '📚' },

{ key: 'blogs', state: showBlogs, setter: setShowBlogs, label: 'Show Blogs', icon: '✍️' }

].map(({ key, state, setter, label, icon }) => (

<label key={key} className={`checkbox-item ${state ? 'enabled' : 'disabled'}`}>

<input

type="checkbox"

checked={state}

onChange={(e) => setter(e.target.checked)}

className="checkbox-input"

/>

<span className="checkbox-label">

{icon} {label}

</span>

</label>

))}

</div>

</div>

</div>

{/\* Techniques Information Panel \*/}

<div className="info-panel">

<h2 className="info-title">🔧 Conditional Rendering Techniques Demonstrated</h2>

<div className="techniques-grid">

<div className="technique-item">

<span className="technique-number">1-3</span>

<span className="technique-name">Basic Methods: Early Return, Logical AND, Ternary</span>

</div>

<div className="technique-item">

<span className="technique-number">4-6</span>

<span className="technique-name">Function-based & Array Conditional Rendering</span>

</div>

<div className="technique-item">

<span className="technique-number">7-9</span>

<span className="technique-name">Complex Ternary & Object Mapping</span>

</div>

<div className="technique-item">

<span className="technique-number">10-12</span>

<span className="technique-name">IIFE, Switch-like & Content-based Conditions</span>

</div>

<div className="technique-item">

<span className="technique-number">13-16</span>

<span className="technique-name">State-based, Array Filtering & Dynamic Styling</span>

</div>

</div>

</div>

{/\* Active Sections Display - Method 17: Dynamic list rendering \*/}

{getActiveSections().length > 0 && (

<div className="active-sections">

<h3 className="active-title">🟢 Currently Active Sections:</h3>

<div className="active-list">

{getActiveSections().map((section, index) => (

<span key={index} className="active-badge">

{section}

</span>

))}

</div>

</div>

)}

{/\* Main Content Area \*/}

<main className="main-content">

{/\* Method 18: Function execution with fallback \*/}

{sectionRenderers[activeSection] ? sectionRenderers[activeSection]() : (

<div className="error-state">

<h2>❌ Section Not Found</h2>

<p>The requested section "{activeSection}" is not available.</p>

<button onClick={() => setActiveSection('all')} className="reset-btn">

🔄 Show All Sections

</button>

</div>

)}

</main>

{/\* Footer with conditional summary \*/}

<footer className="app-footer">

{/\* Method 19: Mathematical conditional rendering \*/}

<p>

Showing {getActiveSections().length} of 3 sections

{getActiveSections().length === 3 && ' (All Active! 🎉)'}

{getActiveSections().length === 0 && ' (None Active 😴)'}

</p>

{/\* Method 20: Time-based conditional (demo) \*/}

<p className="footer-note">

{new Date().getHours() < 12 ? '🌅 Good Morning!' :

new Date().getHours() < 18 ? '☀️ Good Afternoon!' :

'🌙 Good Evening!'}

Welcome to the Conditional Rendering Demo

</p>

</footer>

</div>

</div>

);

};

export default App;

App.css  
  
/\* App.css - Custom styles for BloggerApp without Tailwind \*/

/\* Reset and Global Styles \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

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;

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

min-height: 100vh;

color: #333;

line-height: 1.6;

}

/\* Main App Container \*/

.app-container {

min-height: 100vh;

padding: 2rem 0;

background: linear-gradient(135deg, #f5f7fa 0%, #c3cfe2 100%);

}

.main-wrapper {

max-width: 1200px;

margin: 0 auto;

padding: 0 1rem;

}

/\* Header Styles \*/

.app-header {

text-align: center;

margin-bottom: 2rem;

}

.app-title {

font-size: 3rem;

font-weight: 800;

color: #2d3748;

margin-bottom: 0.5rem;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.1);

}

.app-subtitle {

font-size: 1.2rem;

color: #4a5568;

font-weight: 300;

}

/\* Card Styles \*/

.card {

background: white;

padding: 2rem;

border-radius: 12px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);

margin-bottom: 2rem;

border: 1px solid #e2e8f0;

transition: transform 0.2s ease, box-shadow 0.2s ease;

}

.card:hover {

transform: translateY(-2px);

box-shadow: 0 15px 35px rgba(0, 0, 0, 0.15);

}

.card-header {

font-size: 1.75rem;

font-weight: 700;

margin-bottom: 1.5rem;

color: #2d3748;

border-bottom: 2px solid #edf2f7;

padding-bottom: 0.5rem;

}

/\* Course Card Specific \*/

.course-card {

border-left: 5px solid #48bb78;

}

.course-card .card-header {

color: #38a169;

}

/\* Book Card Specific \*/

.book-card {

border-left: 5px solid #4299e1;

}

.book-card .card-header {

color: #3182ce;

}

/\* Blog Card Specific \*/

.blog-card {

border-left: 5px solid #9f7aea;

}

.blog-card .card-header {

color: #805ad5;

}

/\* Item Container Styles \*/

.item-container {

background: #f7fafc;

border: 1px solid #e2e8f0;

border-radius: 8px;

padding: 1.5rem;

margin-bottom: 1.5rem;

transition: all 0.2s ease;

}

.item-container:hover {

background: #edf2f7;

border-color: #cbd5e0;

transform: translateX(5px);

}

.item-container:last-child {

margin-bottom: 0;

}

.item-title {

font-size: 1.4rem;

font-weight: 600;

margin-bottom: 0.5rem;

}

.course-title {

color: #38a169;

}

.book-title {

color: #3182ce;

}

.blog-title {

color: #805ad5;

}

.item-date {

color: #718096;

font-size: 0.95rem;

margin-bottom: 0.75rem;

font-weight: 500;

}

.item-description {

color: #4a5568;

margin-top: 0.75rem;

line-height: 1.7;

}

.item-pages {

color: #718096;

font-weight: 500;

margin-bottom: 0.5rem;

}

/\* Course Specific Styles \*/

.course-meta {

margin-top: 1rem;

display: flex;

gap: 1rem;

flex-wrap: wrap;

}

.meta-item {

background: #e6fffa;

color: #234e52;

padding: 0.25rem 0.75rem;

border-radius: 20px;

font-size: 0.85rem;

font-weight: 500;

border: 1px solid #81e6d9;

}

/\* Book Specific Styles \*/

.book-category {

color: #2b6cb0;

font-weight: 500;

margin: 0.5rem 0;

}

.book-badge {

display: inline-block;

padding: 0.4rem 1rem;

border-radius: 20px;

font-size: 0.85rem;

font-weight: 600;

margin: 0.5rem 0;

}

.badge-beginner {

background: #c6f6d5;

color: #22543d;

border: 1px solid #9ae6b4;

}

.badge-intermediate {

background: #fefcbf;

color: #744210;

border: 1px solid #f6e05e;

}

.badge-advanced {

background: #fed7d7;

color: #742a2a;

border: 1px solid #fc8181;

}

.book-difficulty {

margin-top: 1rem;

padding: 0.75rem;

border-radius: 8px;

font-size: 0.9rem;

font-weight: 500;

}

.difficulty-easy {

background: #f0fff4;

color: #22543d;

border: 1px solid #c6f6d5;

}

.difficulty-medium {

background: #fffbeb;

color: #744210;

border: 1px solid #fefcbf;

}

.difficulty-hard {

background: #fef5e7;

color: #c53030;

border: 1px solid #fed7d7;

}

/\* Blog Specific Styles \*/

.blog-author {

color: #805ad5;

font-weight: 500;

margin-bottom: 0.5rem;

font-size: 0.95rem;

}

.blog-content {

color: #4a5568;

line-height: 1.7;

margin-bottom: 1rem;

}

.blog-meta {

display: flex;

align-items: center;

gap: 1rem;

flex-wrap: wrap;

margin-top: 1rem;

}

.category-tutorial {

background: #bee3f8;

color: #2a69ac;

padding: 0.3rem 0.75rem;

border-radius: 15px;

font-size: 0.8rem;

font-weight: 600;

}

.category-news {

background: #c6f6d5;

color: #22543d;

padding: 0.3rem 0.75rem;

border-radius: 15px;

font-size: 0.8rem;

font-weight: 600;

}

.category-review {

background: #fefcbf;

color: #744210;

padding: 0.3rem 0.75rem;

border-radius: 15px;

font-size: 0.8rem;

font-weight: 600;

}

.category-general {

background: #e2e8f0;

color: #4a5568;

padding: 0.3rem 0.75rem;

border-radius: 15px;

font-size: 0.8rem;

font-weight: 600;

}

.reading-time,

.publish-date,

.rating {

font-size: 0.8rem;

color: #718096;

background: #f7fafc;

padding: 0.3rem 0.6rem;

border-radius: 12px;

border: 1px solid #e2e8f0;

}

.content-indicator {

margin-top: 1rem;

}

.long-content-badge {

background: #d6f5d6;

color: #2f855a;

padding: 0.25rem 0.75rem;

border-radius: 12px;

font-size: 0.75rem;

font-weight: 600;

border: 1px solid #9ae6b4;

}

/\* Control Panel Styles \*/

.control-panel {

background: white;

padding: 2rem;

border-radius: 12px;

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

margin-bottom: 2rem;

border: 1px solid #e2e8f0;

}

.control-title {

font-size: 1.5rem;

font-weight: 700;

color: #2d3748;

margin-bottom: 1.5rem;

text-align: center;

}

.control-group {

margin-bottom: 2rem;

}

.control-group:last-child {

margin-bottom: 0;

}

.control-label {

display: block;

font-weight: 600;

color: #4a5568;

margin-bottom: 0.75rem;

font-size: 1.1rem;

}

.button-group {

display: flex;

gap: 0.5rem;

flex-wrap: wrap;

}

.section-btn {

background: #edf2f7;

color: #4a5568;

border: 2px solid #cbd5e0;

padding: 0.75rem 1.5rem;

border-radius: 8px;

font-weight: 600;

cursor: pointer;

transition: all 0.2s ease;

font-size: 0.95rem;

}

.section-btn:hover {

background: #e2e8f0;

border-color: #a0aec0;

transform: translateY(-1px);

}

.section-btn.active {

background: #4299e1;

color: white;

border-color: #3182ce;

box-shadow: 0 4px 12px rgba(66, 153, 225, 0.3);

}

.toggle-group {

display: flex;

gap: 0.5rem;

}

.toggle-btn {

background: #f7fafc;

color: #4a5568;

border: 2px solid #e2e8f0;

padding: 0.5rem 1rem;

border-radius: 6px;

cursor: pointer;

transition: all 0.2s ease;

font-size: 0.9rem;

font-weight: 500;

}

.toggle-btn:hover {

background: #edf2f7;

border-color: #cbd5e0;

}

.toggle-btn.active {

background: #805ad5;

color: white;

border-color: #6b46c1;

}

.checkbox-group {

display: flex;

gap: 1.5rem;

flex-wrap: wrap;

}

.checkbox-item {

display: flex;

align-items: center;

cursor: pointer;

padding: 0.75rem;

border-radius: 8px;

transition: all 0.2s ease;

border: 2px solid transparent;

}

.checkbox-item.enabled {

background: #f0fff4;

border-color: #c6f6d5;

}

.checkbox-item.disabled {

background: #fed7d7;

border-color: #fbb6ce;

}

.checkbox-item:hover {

transform: scale(1.02);

}

.checkbox-input {

margin-right: 0.75rem;

width: 1.2rem;

height: 1.2rem;

accent-color: #4299e1;

cursor: pointer;

}

.checkbox-label {

font-weight: 500;

color: #4a5568;

font-size: 0.95rem;

}

/\* Info Panel Styles \*/

.info-panel {

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

color: white;

padding: 2rem;

border-radius: 12px;

margin-bottom: 2rem;

box-shadow: 0 8px 25px rgba(0, 0, 0, 0.15);

}

.info-title {

font-size: 1.5rem;

font-weight: 700;

margin-bottom: 1.5rem;

text-align: center;

text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.3);

}

.techniques-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));

gap: 1rem;

}

.technique-item {

background: rgba(255, 255, 255, 0.15);

padding: 1rem;

border-radius: 8px;

display: flex;

align-items: center;

gap: 1rem;

backdrop-filter: blur(10px);

border: 1px solid rgba(255, 255, 255, 0.2);

transition: transform 0.2s ease;

}

.technique-item:hover {

transform: translateY(-2px);

background: rgba(255, 255, 255, 0.2);

}

.technique-number {

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

color: #4a5568;

font-weight: 700;

padding: 0.5rem;

border-radius: 50%;

min-width: 2.5rem;

height: 2.5rem;

display: flex;

align-items: center;

justify-content: center;

font-size: 0.85rem;

}

.technique-name {

font-weight: 500;

line-height: 1.4;

}

/\* Active Sections Display \*/

.active-sections {

background: #f0fff4;

border: 2px solid #c6f6d5;

padding: 1.5rem;

border-radius: 12px;

margin-bottom: 2rem;

}

.active-title {

font-size: 1.2rem;

font-weight: 600;

color: #22543d;

margin-bottom: 1rem;

}

.active-list {

display: flex;

gap: 0.75rem;

flex-wrap: wrap;

}

.active-badge {

background: #38a169;

color: white;

padding: 0.5rem 1rem;

border-radius: 20px;

font-size: 0.9rem;

font-weight: 600;

box-shadow: 0 2px 8px rgba(56, 161, 105, 0.3);

}

/\* Content Layout \*/

.content-grid {

display: grid;

gap: 2rem;

}

.content-grid.grid-view {

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

}

.content-grid.list-view {

grid-template-columns: 1fr;

}

.main-content {

margin-bottom: 3rem;

}

/\* Hidden/Empty States \*/

.hidden-message {

text-align: center;

padding: 3rem 2rem;

color: #718096;

}

.hidden-message p:first-child {

font-size: 1.5rem;

font-weight: 600;

margin-bottom: 0.5rem;

}

.subtitle {

font-size: 1rem;

color: #a0aec0;

font-style: italic;

}

.no-data {

color: #a0aec0;

font-style: italic;

text-align: center;

padding: 2rem;

background: #f7fafc;

border-radius: 8px;

border: 2px dashed #cbd5e0;

}

.empty-state {

text-align: center;

padding: 3rem 2rem;

}

.empty-state .no-data {

font-size: 1.3rem;

color: #4a5568;

margin-bottom: 0.5rem;

}

.error-state {

text-align: center;

padding: 3rem;

background: #fed7d7;

border: 2px solid #fc8181;

border-radius: 12px;

color: #742a2a;

}

.error-state h2 {

font-size: 2rem;

margin-bottom: 1rem;

}

.reset-btn {

background: #4299e1;

color: white;

border: none;

padding: 0.75rem 1.5rem;

border-radius: 8px;

font-weight: 600;

cursor: pointer;

margin-top: 1rem;

transition: all 0.2s ease;

}

.reset-btn:hover {

background: #3182ce;

transform: translateY(-1px);

box-shadow: 0 4px 12px rgba(66, 153, 225, 0.4);

}

/\* Footer Styles \*/

.app-footer {

background: white;

padding: 2rem;

border-radius: 12px;

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

text-align: center;

margin-top: 3rem;

border-top: 4px solid #4299e1;

}

.app-footer p {

margin-bottom: 0.5rem;

color: #4a5568;

font-weight: 500;

}

.footer-note {

color: #718096;

font-size: 0.95rem;

font-style: italic;

}

/\* Responsive Design \*/

@media (max-width: 768px) {

.app-title {

font-size: 2.5rem;

}

.main-wrapper {

padding: 0 0.5rem;

}

.card {

padding: 1.5rem;

}

.control-panel {

padding: 1.5rem;

}

.button-group {

justify-content: center;

}

.checkbox-group {

flex-direction: column;

gap: 0.75rem;

}

.techniques-grid {

grid-template-columns: 1fr;

}

.content-grid.grid-view {

grid-template-columns: 1fr;

}

.active-list {

justify-content: center;

}

}

@media (max-width: 480px) {

.app-container {

padding: 1rem 0;

}

.app-title {

font-size: 2rem;

}

.card {

padding: 1rem;

margin-bottom: 1rem;

}

.item-container {

padding: 1rem;

}

.control-panel {

padding: 1rem;

}

.section-btn {

padding: 0.5rem 1rem;

font-size: 0.85rem;

}

.blog-meta,

.course-meta {

flex-direction: column;

align-items: flex-start;

gap: 0.5rem;

}

}

/\* Animation Classes \*/

@keyframes fadeIn {

from {

opacity: 0;

transform: translateY(20px);

}

to {

opacity: 1;

transform: translateY(0);

}

}

.card {

animation: fadeIn 0.5s ease-out;

}

.item-container {

animation: fadeIn 0.3s ease-out;

}

/\* Focus States for Accessibility \*/

.section-btn:focus,

.toggle-btn:focus,

.reset-btn:focus {

outline: 3px solid #4299e1;

outline-offset: 2px;

}

.checkbox-input:focus {

outline: 2px solid #4299e1;

outline-offset: 1px;

}

/\* Print Styles \*/

@media print {

.control-panel,

.info-panel,

.app-footer {

display: none;

}

.card {

box-shadow: none;

border: 1px solid #000;

break-inside: avoid;

margin-bottom: 1rem;

}

.app-container {

background: white;

}

}

OUTPUT:  
