**Exercise 13**

**Objectives:**

1. **Explain various ways of conditional rendering**

Conditional rendering in React controls which UI elements are displayed. Common methods include **if-else statements**, **ternary operators**, and **logical &&**. Example: {isLoggedIn ? <Dashboard /> : <LoginPage />}. You can use switch statements or element variables for cleaner logic. Returning null from a component is also a way to prevent rendering.

1. **Explain how to render multiple components**

Multiple components can be rendered by placing them inside a parent component. Wrap them in a container like a <div>, <React.Fragment>, or <> fragment. Example: <Header /> <Content /> <Footer />. Components can be rendered conditionally or inside loops (.map()). This makes React apps modular and easy to maintain.

1. **Define list component**

A list component displays repeating elements from an array using .map().It helps dynamically render structured data like users, products, or posts.Example: items.map(item => <li>{item}</li>).Each list item should have a unique key to improve performance.You can extract list items into child components for reusability.

1. **Explain about keys in React applications**

**Keys** are special attributes used to uniquely identify elements in lists. They help React efficiently update, reorder, or remove components. The key should be a **stable and unique identifier** (e.g., id, not index if data changes). React uses keys to compare virtual DOM elements during re-rendering. Without proper keys, React may show unexpected behavior or warnings.

1. **Explain how to extract components with keys**

You can **extract a list item** into a child component for clarity and reusability. Pass the key to the top-level element inside the mapped child component. Example: <ListItem key={item.id} data={item} />. This keeps logic modular and clean in large applications. The key should remain with the component **at the point of list mapping**.

1. **Explain React Map, map() function**

The .map() function is a JavaScript method used to **iterate over arrays**. In React, it’s commonly used to dynamically render lists of components. Example: items.map(item => <li>{item}</li>). Each rendered element should include a key to optimize rendering. It’s an essential pattern for building scalable, data-driven UIs.

**Hands On Practice**

1. **Create a React App named “bloggerapp”**

npx create-react-app bloggerapp

1. **Create 3 components named:** 
   1. **Book Details**
   2. **Blog Details**
   3. **Course Details**

**BookDetails.js**

import React from 'react';

function BookDetails() {

  const books = [

    { id: 1, title: "React Explained", author: "Zac Gordon" },

    { id: 2, title: "Eloquent JavaScript", author: "Marijn Haverbeke" },

    { id: 3, title: "JavaScript: The Good Parts", author: "Douglas Crockford" }

  ];

  return (

    <div>

      <h3>Book Details</h3>

      <ul>

        {books.map(book => (

          <li key={book.id}>

            <strong>{book.title}</strong> by {book.author}

          </li>

        ))}

      </ul>

    </div>

  );

}

export default BookDetails;

**BlogDetails.js**

import React from 'react';

function BlogDetails() {

  const blogs = [

    { id: 101, title: "React vs Angular", author: "John" },

    { id: 102, title: "JSX Deep Dive", author: "Sara" }

  ];

  return (

    <div>

      <h3>Blog Details</h3>

      <ul>

        {blogs.map(blog => (

          <li key={blog.id}>

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

          </li>

        ))}

      </ul>

    </div>

  );

}

export default BlogDetails;

**CourseDetails.js**

import React from 'react';

function CourseDetails() {

  const courses = [

    { id: 'C1', name: "Full Stack Web Dev", duration: "3 months" },

    { id: 'C2', name: "React Bootcamp", duration: "6 weeks" }

  ];

  return (

    <div>

      <h3>Course Details</h3>

      <ul>

        {courses.map(course => (

          <li key={course.id}>

            {course.name} – {course.duration}

          </li>

        ))}

      </ul>

    </div>

  );

}

export default CourseDetails;

**App.js**

import React, { useState } from 'react';

import BookDetails from './BookDetails';

import BlogDetails from './BlogDetails';

import CourseDetails from './CourseDetails';

function App() {

  const [selected, setSelected] = useState("books");

  let content;

  if (selected === "books") {

    content = <BookDetails />;

  } else if (selected === "blogs") {

    content = <BlogDetails />;

  } else {

    content = <CourseDetails />;

  }

  return (

    <div style={{ padding: '20px' }}>

      <h2>BloggerApp</h2>

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

        <button onClick={() => setSelected("books")}>Show Books</button>

        <button onClick={() => setSelected("blogs")} style={{ marginLeft: '10px' }}>Show Blogs</button>

        <button onClick={() => setSelected("courses")} style={{ marginLeft: '10px' }}>Show Courses</button>

      </div>

      {content}

      <hr />

      <p>

        {selected === "books"

          ? "You are viewing book list."

          : selected === "blogs"

          ? "You are viewing blog posts."

          : "You are viewing courses."}

      </p>

      {selected === "courses" && <p>Courses are hot right now!</p>}

    </div>

  );

}

export default App;

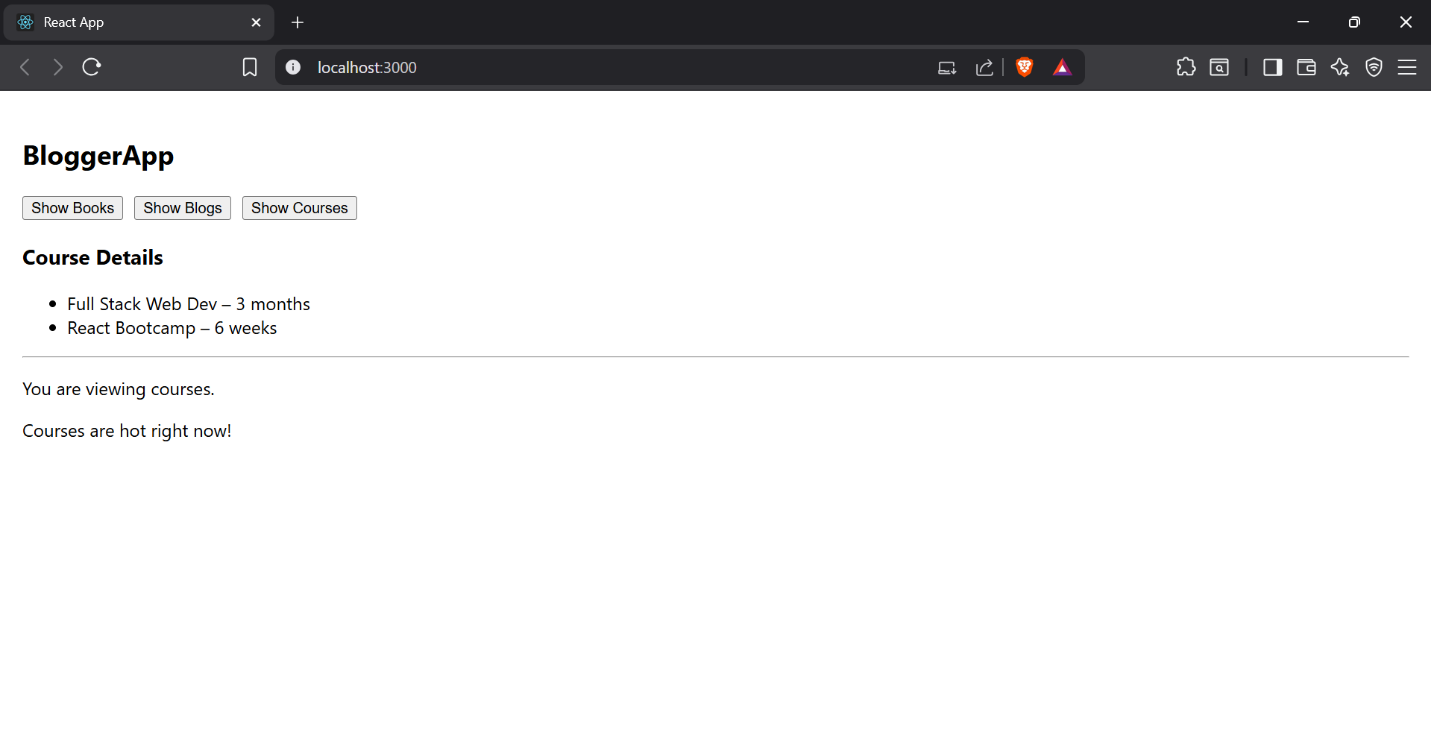
**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

****