# Exercise: Conditional Rendering, Lists, and Keys in React – Blogger App

## 1. Objective

To implement various conditional rendering techniques in React and demonstrate:  
- Different ways of rendering components conditionally.  
- Rendering multiple components at once.  
- Defining and rendering list components.  
- Understanding and using keys in lists.  
- Extracting components with keys.  
- Using the map() function for rendering lists dynamically.

## 2. Problem Statement / Scenario

We need to create a React application named 'bloggerapp' with three components:  
- BookDetails – Displays a list of books.  
- BlogDetails – Displays a list of blogs.  
- CourseDetails – Displays a list of courses.  
  
The app must:  
1. Implement conditional rendering using:  
 - if statements  
 - Ternary operators (condition ? A : B)  
 - Logical AND (condition && <Component />)  
 - Element variables  
 - Inline conditional expressions  
2. Render multiple components together.  
3. Demonstrate list rendering using map() and keys.  
4. Extract list item components for better reusability.

## 3. Implementation

### Step 1: Create React Application

npx create-react-app bloggerapp  
cd bloggerapp  
npm start

### Step 2: App.js

import React, { useState } from "react";  
import BookDetails from "./BookDetails";  
import BlogDetails from "./BlogDetails";  
import CourseDetails from "./CourseDetails";  
  
function App() {  
 const [view, setView] = useState("books");  
  
 let content;  
 if (view === "books") {  
 content = <BookDetails />;  
 } else if (view === "blogs") {  
 content = <BlogDetails />;  
 } else {  
 content = <CourseDetails />;  
 }  
  
 return (  
 <div>  
 <h1>📚 Blogger App</h1>  
 <button onClick={() => setView("books")}>Books</button>  
 <button onClick={() => setView("blogs")}>Blogs</button>  
 <button onClick={() => setView("courses")}>Courses</button>  
  
 <hr />  
  
 <h2>1. Element Variable:</h2>  
 {content}  
  
 <h2>2. Ternary Operator:</h2>  
 {view === "books" ? <BookDetails /> : <BlogDetails />}  
  
 <h2>3. Logical AND:</h2>  
 {view === "courses" && <CourseDetails />}  
  
 <h2>4. Inline Conditional:</h2>  
 {(view === "blogs" && <BlogDetails />) || <BookDetails />}  
 </div>  
 );  
}  
  
export default App;

### Step 3: BookDetails.js

import React from "react";  
  
function BookDetails() {  
 const books = [  
 { id: 1, title: "React Basics", author: "Dan Abramov" },  
 { id: 2, title: "JavaScript Essentials", author: "Kyle Simpson" }  
 ];  
  
 return (  
 <div>  
 <h3>Book List</h3>  
 <ul>  
 {books.map(book => (  
 <BookItem key={book.id} title={book.title} author={book.author} />  
 ))}  
 </ul>  
 </div>  
 );  
}  
  
function BookItem({ title, author }) {  
 return (  
 <li>  
 {title} – <em>{author}</em>  
 </li>  
 );  
}  
  
export default BookDetails;

### Step 4: BlogDetails.js

import React from "react";  
  
function BlogDetails() {  
 const blogs = [  
 { id: 1, title: "React Hooks Overview" },  
 { id: 2, title: "Advanced JavaScript Patterns" }  
 ];  
  
 return (  
 <div>  
 <h3>Blog List</h3>  
 <ul>  
 {blogs.map(blog => (  
 <li key={blog.id}>{blog.title}</li>  
 ))}  
 </ul>  
 </div>  
 );  
}  
  
export default BlogDetails;

### Step 5: CourseDetails.js

import React from "react";  
  
function CourseDetails() {  
 const courses = [  
 { id: 1, name: "Full Stack Development" },  
 { id: 2, name: "Data Science Bootcamp" }  
 ];  
  
 return (  
 <div>  
 <h3>Course List</h3>  
 <ul>  
 {courses.map(course => (  
 <li key={course.id}>{course.name}</li>  
 ))}  
 </ul>  
 </div>  
 );  
}  
  
export default CourseDetails;

## 4. Output

Books View – Shows book list with authors.

Blogs View – Shows blog titles.

Courses View – Shows course names.

Buttons switch between views using different conditional rendering techniques.

## 5. Conclusion

This exercise demonstrates multiple ways of conditional rendering in React, rendering multiple components, using keys in lists, extracting list items into separate components, and using the map() function for dynamic rendering.