
1️⃣ Conditional Rendering in React

Conditional Rendering = React में components या elements को **condition के आधार पर render करना**।

Methods:

1. If-Else Statement

```
if(isLoggedIn){
  return <Dashboard />;
}else{
  return <Login />;
}
```

2. Ternary Operator

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

3. Logical && Operator

```
{messages.length > 0 && <p>You have {messages.length} new messages</p>}
```

4. JSX Variables

```
let message;
if(isLoggedIn){
  message = <Dashboard />;
}else{
  message = <Login />;
}
return <div>{message}</div>;
```

2️⃣Complete Interactive React Example

2.1 App.js (Parent Component)

```
import React, { useState } from "react";
import Login from "./Login";
import Dashboard from "./Dashboard";

function App(){
  const [isLoggedIn,setIsLoggedIn] = useState(false);
  return(
    <div style={{padding:'20px'}}>
      {isLoggedIn ? <Dashboard handleLogout={()=>setIsLoggedIn(false)} /> :
<Login handleLogin={()=>setIsLoggedIn(true)} />}
    </div>
  );
}
export default App;
```

2.2 Login.js

```
import React from "react";
function Login({handleLogin}){
  return(<div>
    <h2>Please Login</h2>
    <button onClick={handleLogin}>Login</button>
  </div>);
}
export default Login;
```

2.3 Dashboard.js

```
import React,{useState} from "react";
import AddUserForm from './AddUserForm';
import UserCard from './UserCard';

function Dashboard({handleLogout}){
  const [users,setUsers] = useState([
    {id:1,name:'Sunil',age:25},
    {id:2,name:'Aman',age:30},
    {id:3,name:'Rakesh',age:28}
  ]);

  const addUser = user => setUsers([...users,{...user,id:Date.now()}]);
  const removeUser = id => setUsers(users.filter(user=>user.id!==id));
  const editUser = (id,newName,newAge) =>
```

```
setUsers(users.map(user=>user.id===id ? {...user,name:newName,age:newAge} :
user));

  return(<div>
    <h1>Dashboard</h1>
    <button onClick={handleLogout}>Logout</button>
    <AddUserForm addUser={addUser} />
    {users.length===0 ? <p>No users available</p> :
users.map(user=>(<UserCard key={user.id} user={user} removeUser={removeUser}
editUser={editUser} />))}
  </div>);
}
export default Dashboard;
```

2.4 AddUserForm.js

```
import React,{useState} from "react";
function AddUserForm({addUser}){
  const [name,setName]=useState('');
  const [age,setAge]=useState('');
  const handleSubmit = e =>{
    e.preventDefault();
    if(!name || !age) return alert('Please fill all fields');
    addUser({name,age:parseInt(age)});
    setName(''); setAge('');
  };
  return(<form onSubmit={handleSubmit} style={{marginBottom:'20px'}}>
    <input type='text' placeholder='Name' value={name}
onChange={e=>setName(e.target.value)} style={{marginRight:'10px'}} />
    <input type='number' placeholder='Age' value={age}
onChange={e=>setAge(e.target.value)} style={{marginRight:'10px'}} />
    <button type='submit'>Add User</button>
  </form>);
}
export default AddUserForm;
```

2.5 UserCard.js

```
import React,{useState} from "react";
function UserCard({user,removeUser,editUser}){
  const [isEditing,setIsEditing]=useState(false);
  const [name,setName]=useState(user.name);
  const [age,setAge]=useState(user.age);
  const
handleSave=()=>{editUser(user.id,name,parseInt(age));setIsEditing(false);};

  return(<div style={{border:'1px solid
gray',borderRadius:'5px',padding:'10px',marginBottom:'10px',width:'300px'}}>
```

```
    {isEditing ? (<div>
      <input type='text' value={name} onChange={e=>setName(e.target.value)}
style={{marginRight:'10px'}} />
      <input type='number' value={age} onChange={e=>setAge(e.target.value)}
style={{marginRight:'10px'}} />
      <button onClick={handleSave}>Save</button>
    </div>) : (<div>
      <h2>{user.name}</h2>
      <p>Age: {user.age}</p>
      <button onClick={()=>setIsEditing(true)}>Edit</button>
      <button onClick={()=>removeUser(user.id)} style={{marginLeft:'10px'}}
>Remove</button>
    </div>)}
  </div>);
}
export default UserCard;
```

✅Features Covered

1. Conditional Rendering → Login / Logout view
2. Dynamic List Rendering → Map + Keys
3. CRUD → Add, Edit, Remove Users
4. Props → Parent → Child communication
5. State → useState for dynamic content

Conclusion

- **Conditional Rendering** → Dynamic UI based on conditions
- **List + Map + Keys** → Efficient dynamic elements rendering
- **Props & State** → Component communication & interactivity
- **CRUD** → Real-life applications like User Management, Todo App

यह **Complete Interactive React Example** आपको React के सभी basic और advanced concepts **practically समझने और practice करने** में मदद करेगा।