# 📌 Complete Example: Lists and Keys in React

Rendering lists efficiently is crucial for **dynamic UIs** like product catalogs, user comments, notifications, and task lists. React optimizes list rendering using **keys**, which help React identify which items have changed, been added, or removed.

---

## ✔️ Features in This Example

✅ **Using `.map()` to render lists**
✅ **Assigning unique `key` props for performance**
✅ **Handling dynamic list updates (adding & deleting items)**
✅ **Rendering nested lists**
✅ **Rendering lists from an API (Fetching JSON Data)**

---

## 📝 `ListExample.js` (Functional Component with Dynamic List Updates)

This example demonstrates:

- **Rendering a list of users**
- **Adding and removing items dynamically**
- **Using keys for optimized re-renders**

jsx
CopyEdit

```
import React, { useState } from "react";

const ListExample = () => {
  const [users, setUsers] = useState([
    { id: 1, name: "Alice" },
    { id: 2, name: "Bob" },
    { id: 3, name: "Charlie" },
  ]);

  const [newUser, setNewUser] = useState("");

  // Add a new user to the list
  const addUser = () => {
    if (newUser.trim() === "") return;
```

```
  const newEntry = { id: users.length + 1, name: newUser };
  setUsers([...users, newEntry]);
  setNewUser("");
 };

 // Remove user by filtering out the clicked user
 const removeUser = (id) => {
  setUsers(users.filter(user => user.id !== id));
 };

 return (
  <div style={styles.container}>
   <h2>User List</h2>

   {/* Input and Button to add new users */}
   <input
    type="text"
    value={newUser}
    onChange={(e) => setNewUser(e.target.value)}
    placeholder="Enter user name"
    style={styles.input}
   />
   <button style={styles.button} onClick={addUser}>Add User</button>

   {/* Rendering the list with unique keys */}
   <ul style={styles.list}>
    {users.map((user) => (
     <li key={user.id} style={styles.listItem}>
      {user.name}
      <button onClick={() => removeUser(user.id)} style={styles.deleteButton}> ✕ </button>
     </li>
    ))}
   </ul>
  </div>
 );
};

// Inline styles for better visualization
const styles = {
 container: {
  textAlign: "center",
  padding: "20px",
  border: "1px solid #ddd",
  borderRadius: "8px",
  width: "350px",
  margin: "20px auto",
  backgroundColor: "#f9f9f9",
 },
 input: {
  padding: "8px",
  fontSize: "16px",
  marginRight: "8px",
```

```
  },
  button: {
    padding: "8px 12px",
    fontSize: "16px",
    cursor: "pointer",
  },
  list: {
    listStyle: "none",
    padding: "0",
  },
  listItem: {
    padding: "8px",
    margin: "5px 0",
    backgroundColor: "#e0e0e0",
    display: "flex",
    justifyContent: "space-between",
    alignItems: "center",
  },
  deleteButton: {
    background: "red",
    color: "white",
    border: "none",
    cursor: "pointer",
    padding: "4px 8px",
  },
};

export default ListExample;
```

---

##  **ListExampleClass.js** (Class Component Example)

This example achieves the same functionality **using class components**.

jsx
CopyEdit

```
import React, { Component } from "react";

class ListExampleClass extends Component {
  constructor(props) {
    super(props);
    this.state = {
      users: [
        { id: 1, name: "Alice" },
        { id: 2, name: "Bob" },
        { id: 3, name: "Charlie" },
      ],
      newUser: "",
    };
  }
```

```
  addUser = () => {
    const { newUser, users } = this.state;
    if (newUser.trim() === "") return;
    const newEntry = { id: users.length + 1, name: newUser };
    this.setState({ users: [...users, newEntry], newUser: "" });
  };

  removeUser = (id) => {
    this.setState({ users: this.state.users.filter(user => user.id !== id) });
  };

  handleInputChange = (event) => {
    this.setState({ newUser: event.target.value });
  };

  render() {
    return (
      <div style={styles.container}>
        <h2>User List (Class Component)</h2>

        <input
          type="text"
          value={this.state.newUser}
          onChange={this.handleInputChange}
          placeholder="Enter user name"
          style={styles.input}
        />
        <button style={styles.button} onClick={this.addUser}>Add User</button>

        <ul style={styles.list}>
          {this.state.users.map((user) => (
            <li key={user.id} style={styles.listItem}>
              {user.name}
              <button onClick={() => this.removeUser(user.id)} style={styles.deleteButton}> ✕ </button>
            </li>
          ))}
        </ul>
      </div>
    );
  }
}

export default ListExampleClass;
```

---

## FetchingListExample.js (Fetching Data from API)

This example demonstrates **fetching data dynamically** from an external API and rendering a list.

jsx
CopyEdit

```
import React, { useState, useEffect } from "react";

const FetchingListExample = () => {
  const [users, setUsers] = useState([]);
  const [loading, setLoading] = useState(true);

  useEffect(() => {
    fetch("https://jsonplaceholder.typicode.com/users")
      .then(response => response.json())
      .then(data => {
        setUsers(data);
        setLoading(false);
      });
  }, []);

  return (
    <div style={styles.container}>
      <h2>Fetched User List</h2>
      {loading ? <p>Loading...</p> : (
        <ul style={styles.list}>
          {users.map(user => (
            <li key={user.id} style={styles.listItem}>
              {user.name} - {user.email}
            </li>
          ))}
        </ul>
      )}
    </div>
  );
};

export default FetchingListExample;
```

---

## App.js (Rendering All Examples)

jsx
CopyEdit

```
import React from "react";
import ListExample from "./ListExample"; // Functional Component
import ListExampleClass from "./ListExampleClass"; // Class Component
import FetchingListExample from "./FetchingListExample"; // API Fetching Component

const App = () => {
  return (
    <div className="App">
      <h1>React Lists and Keys</h1>
      <ListExample />
      <ListExampleClass />
```

```
    <FetchingListExample />
  </div>
 );
};

export default App;
```

---

## ✅ Understanding `.map()` and `key` Usage

| Concept | Example |
|---|---|
| **Basic List Rendering** | {items.map(item => <li key={item.id}>{item.name}</li>)} |
| **Dynamic List with State** | setItems([...items, newItem]) |
| **Fetching List from API** | useEffect(() => { fetchData(); }, []) |

### 🚀 Real-World Use Cases

- **Product Catalogs** (E-commerce)
- **User Comments & Reviews** (Social Media)
- **Notifications List**
- **To-Do Apps & Task Managers**
- **Navigation Menus** (Sidebar or Dropdowns)