🧱 1. Component State – trạng thái cục bộ trong component

👉 Đặc điểm:

- Chỉ dùng trong **một component cụ thể**.
- Dùng useState hoặc useReducer.
- Có thể truyền xuống child component qua props nếu cần.

🔧 Dùng khi:

- Form input
- Toggle (bật/tắt UI)
- Counter, dropdown mở/đóng,...

✅ Hook dùng:

- useState() → đơn giản, gọn
- useReducer() → nhiều biến hoặc logic phức tạp (giống Redux mini)

🧠 Ví dụ:

tsx

```
function Counter() {
  const [count, setCount] = useState(0);
  return <button onClick={() => setCount(count + 1)}>{count}</button>;
}
```

🌍 2. Application State – trạng thái toàn cục (global)

👉 Đặc điểm:

- Dùng chung bởi nhiều component khác nhau.
- Quản lý **modal mở/đóng**, **theme**, **authentication**, **notifications**,...

✅ **Các giải pháp:**

Giải pháp	Đặc điểm
Context + Hooks	Dễ dùng, đủ cho project nhỏ-trung
Redux Toolkit	Quy mô lớn, cấu trúc rõ, hỗ trợ tốt devtool
Zustand	Nhẹ, dễ dùng, hiện đại
Jotai, MobX, XState	Tùy nhu cầu ứng dụng phức tạp

🧠 **Ví dụ dùng Context:**

tsx

```
const ThemeContext = createContext();

function App() {
  const [dark, setDark] = useState(false);
  return (
    <ThemeContext.Provider value={{ dark, setDark }}>
      <Page />
    </ThemeContext.Provider>
  );
}

function Page() {
  const { dark } = useContext(ThemeContext);
  return <div>{dark ? '🌙 Dark' : '☀️ Light'}</div>;
}
```

🔁 **3. Server Cache State – trạng thái lấy từ server & lưu cache tạm thời**

👉 **Đặc điểm:**

- Dữ liệu fetch từ backend (REST API, GraphQL,...)
- Được quản lý bằng cache để tránh fetch lại không cần thiết

✅ **Dùng thư viện chuyên biệt:**

Thư viện	Giao thức hỗ trợ

React Query	REST + GraphQL
SWR	REST + GraphQL
Apollo Client	GraphQL
URQL	GraphQL
RTK Query	REST (Redux Toolkit)

🧠 **Ví dụ dùng React Query:**

tsx

```
function useUsers() {
  return useQuery(['users'], () => axios.get('/api/users').then(res => res.data));
}
```

> ❗ **Không nên** lưu server data vào Redux hay Context nếu chỉ là fetch và hiển thị → dùng cache libs sẽ hiệu quả hơn.

📝 **4. Form State – trạng thái của form (nhập liệu, validation)**

👉 **Đặc điểm:**

- Gồm giá trị input, error message, touched, validation,...
- Có thể **controlled** hoặc **uncontrolled**

✅ **Thư viện phổ biến:**

Tên	Đặc điểm
React Hook Form	Nhẹ, nhanh, tích hợp Zod/Yup
Formik	Mạnh mẽ, dễ hiểu
React Final Form	Rất tối ưu hiệu năng
+ Validation libs: Zod, Yup	

🧠 **Ví dụ:**

tsx

```
import { useForm } from 'react-hook-form';

function ContactForm() {
  const { register, handleSubmit } = useForm();
  return (
    <form onSubmit={handleSubmit((data) => console.log(data))}>
      <input {...register('email')} />
      <button type="submit">Gửi</button>
    </form>
  );
}
```

🌐 5. URL State – trạng thái gắn với URL

👉 Đặc điểm:

- Là dữ liệu nằm trong đường dẫn (/users/123) hoặc query (?page=2)
- Có thể đọc/ghi URL để điều khiển nội dung trang, phân trang, filter...

✅ Công cụ:

- react-router-dom → điều hướng và đọc useParams, useSearchParams

🧠 Ví dụ:

tsx

```
import { useParams, useSearchParams } from 'react-router-dom';

function ProductPage() {
  const { id } = useParams(); // /products/:id
  const [searchParams] = useSearchParams(); // ?page=2

  return <div>ID sản phẩm: {id}, Trang: {searchParams.get('page')}</div>;
}
```

✅ Tổng kết: Các loại state và cách chọn

Loại State	Dùng khi nào

Component State	Dữ liệu chỉ dùng trong một component
Application State	Dùng chung nhiều nơi (modal, theme, auth)
Server Cache State	Dữ liệu từ API, cần cache + xử lý loading/error
Form State	Quản lý form input, validation, errors
URL State	Phụ thuộc URL, ví dụ: filter, paging, id từ route