# REACT STATE MANAGEMENT

@JovonneCameron

## WHAT IS STATE MANAGEMENT?

State management is how you handle and organize data that changes in your React application. Think of State as your app's memory. It remembers things like user input, which tab is active, or data from an API

## TYPES OF STATE

**LOCAL STATE:** For single values or simple objects in a component

**GLOBAL STATE:** Shared across multiple components.

**SERVER STATE:** Comes from external sources (APIs, databases)

**URL STATE:** Data in the URL (search params, pathname)

## RESOURCES

- React State and Lifecycle
- React useState
- React Context API
- React Query Docs
- Redux Fundamental Tutorial
- Redux Toolkit Docs

| WHEN YOU WANT TO | USE THIS | EXAMPLE USE CASE |
|---|---|---|
| Store data in one component | useState | Form inputs, toggles |
| Share data between a few components | useContext | Theme settings, user preferences |
| Manage form data | React Hook Form | Login forms, registration forms |
| Handle API data | React Query or SWR | User profiles, product listings |
| Handle Global data | Redux Toolkit or Zustand | Shopping carts, multi-step workflows |

## COMMON PITFALLS

- Placing state too high in your component tree
- Putting everything in global state when it's only needed locally
- Direct mutation instead of creating new objects
- Storing derived data that could be calculated from existing state
- Not splitting state by domain or purpose

## STATE MANAGEMENT CHECKLIST

### FOR COMPONENT STATE:

- Could this state be local to the component?
- Am I updating state based on previous state?
- Am I avoiding direct state mutation? (Use spread operator instead)

### FOR SHARED STATE:

- How many components need this data?
- How often does this data change?
- Does this data come from a server?