React Component Lifecycle (with Hooks)

1. Components Design (Modular & Reusable)

React mein har component ko modular aur reusable banayein. Ek component sirf ek kaam kare aur use reusability ke liye design karein.

Isse aapka code clean aur maintainable rahega. Components ko small parts mein divide karein jise aap multiple places pe use kar sakein.

2. State Management

State ko manage karne ke liye React hooks, jaise ke useState aur useReducer ka use karein. Agar aapka app bara ho, to state management libraries jaise Redux ya Context API ka use karein.

State ko directly manipulate karne se bachain aur state ko properly manage karne ke liye functional updates ka use karein.

3. useEffect Optimization

useEffect hook ko sahi tareeqe se optimize karein taake unnecessary renders se bach sakein. Jab aap useEffect ka use kar rahe hain, to dependency array ko dhang se define karein.

Agar kisi specific prop ya state ka effect hai, to usi ko dependency array mein daalain.

4. Code Splitting (Efficient Loading)

Code splitting se aap apne React app ko optimize kar sakte hain. React lazy loading aur Suspense ka use kar ke components ko dynamically load kar sakte hain, taake app jaldi load ho.

Iska fayda yeh hota hai ke initial loading time kam hota hai, aur users ko fast experience milta hai.

5. Error Boundaries (Error Handling)

Error boundaries ka use React app mein error handling ke liye karein. Yeh aapko app mein runtime errors ko gracefully handle karne mein madad deti hain.

Aap componentDidCatch aur static getDerivedStateFromError ka use karke error ko manage kar sakte hain.

6. Summary

React ki best practices follow karna aapko ek efficient, maintainable, aur scalable app banane mein madad dega. Components ko modular rakhein, state ko manage karte hue code ko clean aur efficient banayein.