React Hooks & State

UNDERSTANDING STATE AND IMPORTANT HOOKS WITH EXAMPLES

What is State in React?

- State is data that changes over time in a component.
- When state changes, React re-renders the component.
- Managed using the useState hook in function components.
- Example: Counter, form inputs, toggles.

useState Hook

- useState lets you add state to a function component.
- Syntax: const [state, setState] = useState(initialValue)
- Example: Counter
- const [count, setCount] = useState(0);
- Button onClick={() => setCount(count + 1)}

useEffect Hook

- Used for side effects in components.
- Examples: fetching data, setting timers, updating DOM.
- Runs after render by default.
- Example:
- useEffect(() => { fetchData(); }, []);

useContext Hook

- Provides a way to share data across components without props.
- Example: Theme, Authentication.
- Example: const value = useContext(MyContext);

useRef Hook

- Gives direct access to DOM elements or stores mutable values.
- Does not cause re-renders when updated.
- Example: const inputRef = useRef(); inputRef.current.focus();

useMemo Hook

- Memoizes values to avoid expensive recalculations.
- Runs when dependencies change.
- Example: const result = useMemo(() => compute(a, b), [a, b]);

useCallback Hook

- Memoizes functions to prevent unnecessary re-creation.
- Useful when passing functions as props.
- Example: const handleClick = useCallback(() => doSomething(), []);

useReducer Hook

- Alternative to useState for complex state logic.
- Takes a reducer function and initial state.
- Example: const [state, dispatch] = useReducer(reducer, initialState);

React 18+ Hooks

- useId – generates unique IDs for accessibility.
- useTransition – allows UI updates without blocking.
- useDeferredValue – defers re-rendering for performance.
- useSyncExternalStore – subscribe to external stores.
- useInsertionEffect – injects styles before DOM mutations.

Custom Hooks

- Custom hooks let you reuse stateful logic.
- Must start with 'use'.
- Example: useFetch, useAuth, useLocalStorage.

Key Takeaways

- State is the heart of React components.
- useState and useEffect are the most commonly used hooks.
- Hooks simplify state management and side effects.
- React 18 introduced performance-oriented hooks.
- Custom hooks let you abstract and reuse logic.