React- Day -5: React Hooks:

What is the hook:

a "hook" is a special function that allows functional components to use **state and other React features**. Hooks were introduced in **React 16.8** to provide a way to use stateful logic **in functional components**, which were **previously stateless.**

Some commonly used React hooks include:

- useState: Allows functional components to **manage state.**
- useEffect: Enables **side effects** in functional components (e.g., data fetching, subscriptions).
- useContext: avoid **props drilling** and **global state management**. context.
- useReducer: managing complex **global state management**
- useCallback and useMemo: Memoize values and functions to optimize performance. **Avoiding unnecessary re-render**

Lifecycle of Hook:

Functional Component Lifecycle with Hooks:

Mounting Phase: useState(), useEffect()
Updating Phase : useState(), useEffect()
(useMemo() and useCallback() **AVOIDING UNNECESSARY RE RENDERING THE COMPONENTS**)
Unmounting Phase : useEffect():

useState, useEffect:

UseState :

hook is one of the fundamental React hooks that allows you to add state to functional components.

The useState hook takes an initial value as its argument and returns an array containing two elements: the **current state value** and a **function to update that value.**

const [stateValue, setStateFunction] = useState(initialValue);

UseEffect & dependency Array :

that allows you to **perform side effects in functional components**. Side effects include actions like data fetching, DOM manipulation.

Dependency array : Mounting , Updating, Unmounting

[] – empty array -> render once

[props,state] - > update

Return -> clear

Note: Flipkart or amazon opening screen

Props drilling :

It refers to the process of passing props through multiple layers of React components in order to provide data to **deeply nested components.**

Disadvantages:

Maintenance Complexity: If you need to add or modify a prop at the bottom of the component hierarchy, **you have to update every intermediate component.**

Breaking Abstraction: Components in the middle of the hierarchy might receive and pass down props that **they don't use**

Passing data from child to parent component - practical example