**Explain Life cycle in Class Component and functional component with Hooks**

**Component Lifecycle**

**Class Components**

* **Mounting:**

First create a component and inserted into the DOM

The following lifecycle methods are called in this phase:

* + - constructor(): Used to initialize the component's state
    - componentDidMount(): The ideal place to fetch data from external sources or perform actions that rely on the DOM being fully available
* **Updating:**
  + When the component's props or state changes, the component needs to re-render to reflect these updates in the UI.
  + The following lifecycle method is used in this phase:
    - componentDidUpdate(prevProps, prevState): Invoked after the component has re-rendered due to changes in props or state
* **Unmounting:**
  + When a component is no longer needed, it's removed from the DOM.
  + The following lifecycle method is called in this phase:
    - componentWillUnmount(): Used for cleanup tasks

**Functional Components with Hooks**

* **Functional components are simpler** as they don't have lifecycle methods built in. Hooks allow you to "hook into" the lifecycle and manage state
* **Hooks for Lifecycle Management:**
  + useState: This hook is used to initialize and manage state within functional components. It provides a way to track data that can change and trigger re-renders when necessary.
  + useEffect: This hook is a versatile tool for handling side effects like data fetching, subscriptions, and running code after every render