**Module:10 List and Hooks**

**Q:-** **Explain Life cycle in Class Component and functional component with Hooks.**

**=>**

* **Mounting :** When a Class Component is created and instead into the DOM for the first time, it goes through these phases :
* **Constructor** : This is like the creature’s birthplace, where we set its initial characterstics and state.
* **Render** : Here, we create the creature’s appearance (UI) based on its current state and props.
* **componentDidMount** : After the creature is born and shown to the world, this is where we can make it do things like fetching data or setting up timers.
* **Growth and Updates (Updating)**: As the creature grows and experiences changes, it goes through these phases:
* **shouldComponentUpdate**: This is like the creature thinking about whether it should change based on new props or state.
* **render**: If it decides to change, it updates its appearance (UI) in this step.
* **componentDidUpdate**: After the change, it might want to do something special, like saving its current state.
* **Aging and Farewell (Unmounting)**: When the creature is no longer needed and removed from the DOM, it goes through:
* **componentWillUnmount**: This is like the creature preparing for its farewell party, where it can clean up resources before it's gone.

**Lifecycle in Functional Components with Hooks :**

* **Birth and Growth (Mounting and Updating)**: Functional components with hooks have two main phases:
* **useState**: This is where you set up the creature's initial characteristics and state.
* **useEffect**: Similar to componentDidMount and componentDidUpdate in class components, this is where you can make your creature do things when it's born and whenever it changes. It's like a combination of those lifecycle methods.
* **Aging and Farewell (Unmounting)**: When the creature is no longer needed, you can use:
* **useEffect cleanup**: This is where you can clean up resources, similar to componentWillUnmount in class components.