Class Component Lifecycle:

Mounting Phase:

constructor(props):

This is the first method called when a class component is created. It's used for initializing state and binding event handlers.

render():

This method returns the JSX representation of the component and is called whenever the component needs to be rendered.

componentDidMount():

This method is called after the component has been rendered in the DOM. It's often used for making API calls, setting up subscriptions, or performing other side effects.

Updating Phase:

shouldComponentUpdate(nextProps, nextState):

This method allows you to optimize performance by deciding whether the component should re-render. It's called before rendering when new props or state are received.

render():

Again, this method is called to re-render the component.

componentDidUpdate(prevProps, prevState):

This is called after the component has been updated in the DOM. It's often used for side effects related to state changes.

Unmounting Phase:

componentWillUnmount():

This method is called just before the component is removed from the DOM. It's used for cleanup tasks like canceling timers or unsubscribing from subscriptions.

Functional Component with Hooks Lifecycle:

Functional components with hooks have a more simplified lifecycle:

Mounting and Updating Phase:

useState, useEffect, and other hooks: In functional components, you use hooks like useState to manage state and useEffect for side effects. Hooks are called on every render, so you can think of each render as encompassing both the mounting and updating phases of a class component.

Unmounting Phase:

There's no specific hook for the unmounting phase. However, you can return a cleanup function from useEffect to perform cleanup tasks when the component is unmounted. This is similar to the componentWillUnmount method in class components.