### 1. What are the different phases of React component's lifecycle?

The different phases of React component's lifecycle are:

**Initial Phase:** It is the birth phase of the React lifecycle when the component starts its journey on a way to the DOM. In this phase, a component contains the default Props and initial State. These default properties are done in the constructor of a component.

**Mounting Phase:** In this phase, the instance of a component is created and added into the DOM.

**Updating Phase:** It is the next phase of the React lifecycle. In this phase, we get new Props and change State. This phase can potentially update and re-render only when a prop or state change occurs. The main aim of this phase is to ensure that the component is displaying the latest version of itself. This phase repeats again and again.

**Unmounting Phase:** It is the final phase of the React lifecycle, where the component instance is destroyed and unmounted(removed) from the DOM.

For More Information

2. Explain the lifecycle methods of React components in detail.

The important React lifecycle methods are:

* **getInitialState():** It is used to specify the default value of this.state. It is executed before the creation of the component.
* **componentWillMount():** It is executed before a component gets rendered into the DOM.
* **componentDidMount():** It is executed when the component gets rendered and placed on the DOM. Now, you can do any DOM querying operations.
* **componentWillReceiveProps():** It is invoked when a component receives new props from the parent class and before another render is called. If you want to update the State in response to prop changes, you should compare this.props and nextProps to perform State transition by using this.setState() method.
* **shouldComponentUpdate():** It is invoked when a component decides any changes/updation to the DOM and returns true or false value based on certain conditions. If this method returns true, the component will update. Otherwise, the component will skip the updating.
* **componentWillUpdate():** It is invoked before rendering takes place in the DOM. Here, you can't change the component State by invoking this.setState() method. It will not be called, if shouldComponentUpdate() returns false.
* **componentDidUpdate():** It is invoked immediately after rendering takes place. In this method, you can put any code inside this which you want to execute once the updating occurs.
* **componentWillUnmount():** It is invoked immediately before a component is destroyed and unmounted permanently. It is used to clear up the memory spaces such as invalidating timers, event listener, canceling network requests, or cleaning up DOM elements. If a component instance is unmounted, you cannot mount it again.

3.What are Pure Components?

* Pure components introduced in React 15.3 version. The React.Component and React.PureComponent differ in the shouldComponentUpdate() React lifecycle method. This method decides the re-rendering of the component by returning a boolean value (true or false). In React.Component, shouldComponentUpdate() method returns true by default. But in React.PureComponent, it compares the changes in state or props to re-render the component. The pure component enhances the simplicity of the code and performance of the application.

4. What are Higher Order Components(HOC)?

* In React, Higher Order Component is an advanced technique for reusing component logic. It is a function that takes a component and returns a new component. In other words, it is a function which accepts another function as an argument. According to the official website, it is not the feature(part) in React API, but a pattern that emerges from React's compositional nature.

### 5. What can you do with HOC?

You can do many tasks with HOC, some of them are given below:

* Code Reusability
* Props manipulation
* State manipulation
* Render highjacking

### 6. How to write comments in React?

In React, we can write comments as we write comments in JavaScript. It can be in two ways:

**1. Single Line Comments:** We can write comments as /\* Block Comments \*/ with curly braces:

{/\* Single Line comment \*/}

**2. Multiline Comments:** If we want to comment more that one line, we can do this as

{ /\*

   Multi

   line

   comment

\*/ }

### 7.Why is it necessary to start component names with a capital letter?

### In React, it is necessary to start component names with a capital letter. If we start the component name with lower case, it will throw an error as an unrecognized tag. It is because, in JSX, lower case tag names are considered as HTML tags.