**STATE AND LIFECYCLE**

Components can be made usable and encapsulated by adding state. State is only accessible to the component that owns and sets it. State although similar to props, is fully controlled by the component. In class components, props can be transformed to states by replacing the props with states in the code, adding a class constructor with the initial “this.state” and removing the prop from the component. Lifecycle method is the stages components in applications go through when they are rendered(mounted), updated and eventually deleted(unmounted). State can be used correctly if the developer does not modify state directly, does not depend solely on synchronous updates and acknowledge state updates are merged.

**USING THE EFFECT HOOK**

The Effect hook is used in function components and allows the developer to perform side effects such as fetching data. The effect hook renders all three lifecycle methods (mount, update and unmount) of the component. Effects with cleanup (e.g. setting up subscription) and without cleanup (e.g. logging) are the two most common kinds of side effect in React components. Effect hook allows the developer to express various side effects after rendering a component. A function is returned when the effect requires a clean up where as effects that do not require clean ups don’t return anything. It should be noted that the useEffect hook just like the useState hook can be used more than once in a component.