**Hooks**

* Hooks are mainly used for the UI updation.
* If we have many variables and we need to update all these variable at same time then we need hooks.
* There are many types of hooks are available – useState (), useRef (), useEffect (), useContext (), etc.
* **useState () –** Whenever we need a variable to update at many places and that will change over times eg. Counter, user input.
* **useEffect () –** Anytime you want something to happen “after” the component renders eg. API Calls, timers, logging.
* **useContext () –** When you need to share data across multiple components without passing the props though each component.
* **useReducer () –** When we need to update complex state like toggling multiple values or if the state depends on the previous state values.
* **useRef () –** For accessing DOM elements directly or storing values that don’t need to trigger a re-render eg. Timers, previous values.
* **useMemo () –** For performance optimization when a calculation is complex or data-intensive.
* **useCallback() –** It helps to optimize the programme by memorizing the functions. It is used when we have others child and these children will also use that function to do some of their work. While using the useCallback we have to pass the function name and then all the children in an array format. Call back simply means keep the things in the cache which will be using/used.
* **useEffect() –** It is when we have to do some after the component renders, We can use it to handle things that don’t directly involve displaying data, like fetching from API, updating the browser’s title, etc. It takes two values – Call-back and dependencies. If anything happens in the code, run the code again.
* **useRef() –** It is used to hold a reference of a DOM element or to store a value. Basically used for storing the reference. While using the useRef then we have to store it in a variable and then we have to pass that variable in any element whose reference we want to store.