**UseEffect**:-

1. If we do not put dependency inside the useEffect hook then it will render every

useEffect(()=>{

console.log(“use effect runs”)

})

1. If we put dependency array as empty inside the useEffect then it will render only when component mount.

useEffect(()=>{

}.[])

3.If we want to control the render of useEffect that it will runs only the some condition .

useEffect()=>{

console.log(“useeffect runs”)

},[count])

4.if we want render mounting and unmounting inside the same useEffect hook

useEffect(()=>{

console.log(“runs on mounting”)

const interval=setTimeout(()=>{

console.log(“intervall”)

},1000)

// this is the cleanup function runs on the unmounting only

return ()=>{

console.log(“component unmounted”)

}

},[])

5.What is controlled and uncontrolled components in react js

Input form elements such as <input/> and <textarea> values are controlled by react state in functional components we uses usestate or with class base components we uses this.state .to   
usestate and this.state to store input value and update it with onchange

Controlled =>

import { useState } from 'react';

function ControlledForm() {

const [name, setName] = useState('');

return (

<input

type="text"

value={name}

onChange={(e) => setName(e.target.value)}

/>

);

}

Uncontrolled:- A form input which manages its own state internally (like traditional html manipulating dom). In react functional components we use the useRef hook to access it value by referencing its DOM id  
  
example :-

import { useRef } from 'react';

function UncontrolledForm() {

const nameRef = useRef();

const handleSubmit = () => {

alert(nameRef.current.value);

};

return (

<>

<input type="text" ref={nameRef} />

<button onClick={handleSubmit}>Submit</button>

</>

);

}