UseEffect hook pollyfill :

useEffect hook will take two arguments : callback function and dependency array

Create custom use effect component with two argument for callback function and dependency array. And then use logics to achieve useEffect cases

3 cases in useEffect :

1. No dependency array🡪 will render the useEffect each time when every state changes
2. Empty dependency array 🡪 it will render the useEffect function for the 1st time when application render
3. Dependency array 🡪 when specific state in the dependency array changes, that time only it will call the useEffect

export const CustomUseEffect = (callBackFun,dep) => {

    const firstRender=useRef(true)  //don't forget to use .current property while accessing the useRef

    const preDependency= useRef([])

    if(firstRender.current){

        firstRender.current=false;

        callBackFun();

    }

   const depChanged = dep? JSON.stringify(dep)!== JSON.stringify(preDependency.current) : true;

if(depChanged){

    callBackFun();

}

preDependency.current=dep||[];

clean up function :

clean up functions will be called in two scenarios

1. When the component unmounts
2. When Dependency array changes

In the 1st render useEffect returns clean up function and based on some condition that will be called

 const cleanup= callBackFun();

        if(cleanup && typeof cleanup ==='function'){

            cleanup();

if(depChanged){

    const cleanup= callBackFun();

        if(cleanup && typeof cleanup ==='function' && dep){

            cleanup();

        }

}