**FAQ: When context ends?  
Ans: On Component Unmount.  
  
                                useCookies  
- Cookie is a simple text document.  
- It can store client authentication details.  
    a) Temporary  
    b) Persistent  
- If cookie is temporary then it is called as "In-memory-cookie".  
- It is deleted automatically when browser is closed.  
- If cookie is permanent then it is called as "Persistent-Cookie".  
- It can store in your HDD for a duration of specified time interval.  
- Cookies are used by web application to store client authentication details and use across multiple requests and components until removed.  
- Cookies can be infected with virus.  
- Cookies can be disabled by browser.  
  
FAQ: How a developer known the status of cookies on client browser?  
Ans: By using JavaScript   "navigator" object  
  
            navigator.cookieEnabled  = true / false  
            navigator.appName  
            navigator.plugins[]  
             
- React allows developers to use various 3rd party cookie libraries  
                "react-cookie"  
  
    > npm install  react-cookie --save  
  
- Cookie is provided as a service by "react-cookie" library  
  
- The provider is  
  
        <CookieProvider>  
  
        </CookieProvider>  
- The components that have to use cookie must be with in the scope of cookie provider.  
  
        <CookieProvider>  
            <YourComponent />  
        </CookieProvider>  
  
- The cookies are created by "useCookies" hook  
- "useCookies" hook is responsible for  
        a) Creating  
        b) Reading  
        c) Removing  
  
Syntax: Configuring Cookie  
       const [cookies, setCookie, removeCookie] = useCookies(['cookiename'])  
  
Syntax: Creating Cookie  
  
        setCookie('cookiename','cookievalue', { cookieOptions });  
  
        Cookie Options:  
        a) path        : It defines the virtual path.  "/"  
        b) expires    : It defines lifespan  
  
Note: If expires is not defined then it is temporary cookie.  
        If expires is defined then it is store in HDD.  
  
        setCookie('cookiename','cookievalue', { path:"/", expires: new Date("2022-07-10 20:22") });  
  
Syntax: Reading Cookie  
  
        cookies.cookieName  
  
Syntax: Check Cookie Availablity  
  
        cookies.cookieName==undefined = true  [not available]  
         
Syntax: Remove Cookie  
  
        removeCookie('cookiename');  
  
  
  
Ex:  
import { useState, useEffect } from "react";  
import { useCookies } from "react-cookie";  
  
export default function UserLogin(){  
    const [cookies, setCookie, removeCookie] = useCookies(['username']);  
    const [userDetails, setUserDetails] = useState({UserName:'', Password:''});  
  
    function handleUserName(e){  
        setUserDetails({  
            UserName: e.target.value,  
            Password: userDetails.Password  
        })  
    }  
    function handlePassword(e){  
        setUserDetails({  
            UserName: userDetails.UserName,  
            Password: e.target.value  
        })  
    }  
    function handleLogin(){  
        setCookie("username",userDetails.UserName,{path:"/", expires:new Date("2022-07-02")});  
        alert("Login Success..");  
    }  
    function handleSignout(){  
        removeCookie("username");  
        alert("Signed out Successfully..");  
    }  
    useEffect(()=>{  
        if(cookies.username==undefined) {  
            alert("Please Login");  
        }  
    },[]);  
  
    return(  
        <div className="container-fluid">  
            <h2>User Login</h2>  
            <dl>  
                <dt>User Name</dt>  
                <dd><input onChange={handleUserName} type="text"/></dd>  
                <dt>Password</dt>  
                <dd><input onChange={handlePassword} type="password"/></dd>  
            </dl>  
            <button onClick={handleLogin}>Login</button>  
            <hr/>  
            <div>  
            <h3>Home  <button onClick={handleSignout} className="btn btn-link">Signout</button> </h3>  
            Hello ! {cookies.username}  
            </div>  
        </div>  
    )  
}**