**- useState  
- useEffect  
- useContext  
  
FAQ: What is Context memory?  
Ans:  It is the memory allocated for a component and accessible to  
other components that run with in the context of current component.  
  
FAQ: What is the purpose of Context memory?  
Ans: It enables sharing of data across components configure in multilevel hierarchy.**

**A higher component can share its data to lower component. [outer to inner]**

**It also enables to share data across multiple requests of same component.  
  
  
 Implementing Context using React Hook  
  
- Context uses Dependency Injection mechanism.  
- DI is a technique that defines how an object get holds of its dependencies.  
- It is about how it locates and injects the dependencies.  
- DI depends on two components  
  
    a) Provider  
    b) Injector  
  
- Provider is responsible for locating the dependencies in memory.  
            [ stores and locates data ]  
  
- Injector is responsible for injecting the data into component.  
- React "useContext"  uses provider and injector  
  
- useContext acts as a "Service"  
  
- Service follows "Single Ton" pattern  
  
- Single Ton pattern creates an instance for first request and uses the same across multiple requests.  
    [only once instance can be defined for a class]  
  
Step-1: You have to create a new context memory for component  
           "React" core library provides a function "createContext()"  
Syntax:  
        var contextName = React.createContext(null);  
  
Step-2: You have to provide value into context  
Syntax:  
           <contextName.Provider  value={any}>  
                    //scope  
                    // all components must be within the context  
                       provider scope  
           </contextName.Provider>  
           
Step-3: You have to inject into any component  
            Inner component have to access the context.  
Syntax:  
            const  ref = useContext(contextName);  
  
Ex:  
*import React, {useContext, useState} from "react"  
  
var userDetailsContext = React.createContext(null);  
  
export default function ContextDemo()  
{  
    const [userDetails, setUserDetails] = useState({  
        UserName: '',  
        Email:''  
    })  
  
    function HandleUser(e){  
        setUserDetails({  
            UserName: e.target.value,  
            Email: userDetails.Email  
        })  
    }  
    function HandleEmail(e){  
        setUserDetails({  
            UserName: userDetails.Email,  
            Email: e.target.value  
        })  
    }  
    function HandleSet(){  
        setUserDetails({  
            UserName: userDetails.UserName,  
            Email: userDetails.Email  
        })  
    }  
  
    return(  
        <userDetailsContext.Provider value={userDetails}>  
        <div className="container-fluid">  
                <h1>Site Index - {userDetails.UserName}</h1>  
                <dl>  
                    <dt>User Name</dt>  
                    <dd><input onChange={HandleUser} type="text"/></dd>  
                    <dt>Email</dt>  
                    <dd><input onChange={HandleEmail} type="email"/></dd>  
                </dl>  
                <button onClick={HandleSet}>Set Data</button>  
                <HeaderComponent />  
        </div>  
        </userDetailsContext.Provider>  
         
    )  
}  
  
function HeaderComponent(){  
    var userdetails = useContext(userDetailsContext);  
    return(  
        <div className="bg-info text-white" style={{height:'150px', padding:'10px'}}>  
            <h2>Home - {userdetails.UserName} </h2>  
            <NavbarComponent />  
        </div>  
    )  
}  
function NavbarComponent() {  
    var userdetails = useContext(userDetailsContext);  
    return(  
        <div className="btn-toolbar bg-dark text-white justify-content-between">  
            <div className="btn-group">  
                <button className="btn btn-dark">Amazon</button>  
            </div>  
            <div className="btn-group">  
                <button className="btn btn-dark">{userdetails.Email}</button>  
            </div>  
        </div>  
    )  
}***