RTK - Redux took kit

* Install libraries
  + @reduxjs/toolkit and react-redux
* Build our store
* Connect out store to our app
* Create slice (card slice)
* Dispatch (action)
* Selector

Npm install @reduxjs/toolkit.

Npm i react-redux

app.js

return (

<Provider store={appStore}>

<UserContext.Provider value={{name:userName, setUserName}}>

<Header />

<Outlet />

<Footer />

</UserContext.Provider>

</Provider>

)

appstore,js

import { configureStore } from "@reduxjs/toolkit";

import cartSlice from "./cartSlice";

const appStore = configureStore({

reducer:{

cart: cartSlice

}

});

export default appStore

cartSlicer.js

import { createSlice } from "@reduxjs/toolkit";

const cartSlice = createSlice({

name:"cart",

initialState:{

items:[]

},

reducers:{

addItem:(state, action)=>{

state.item.push(action.payload)

},

removeItem:(state, action)=>{

state.item.pop()

},

clearCard:(state, action)=>{

state.item.length = 0;

}

}

})

export const {addItem, removeItem, clearCard} = cartSlice.actions;

export default cartSlice.reducer;

In older version

In the slice part

We do not allow to mutate the state

But in new version, we have to do it

In older version

Const newState = [...state]

newState.items.push(action.payload)

Return newState;

Return was also mandatory

But in new this is not.

What happened before in the older version is still happening in the background but for developers, they made it easy,

Redux is using the immer library to find the difference between state and update.

Immer is a tiny package that allows you to work with an immutable state in a more convenient way

In new way

Either mutate the state or return a new state.

We can use redex dev tool

In old version of rexu we use middle ware and thungs

In new version we have RTK query