Episode 12 - Let's Build Our Store

Managing the data efficiently

When the app is small you don't have to manage the data that efficiently its not crucial but for large production apps you need to manage the data its necessary.

Why do we need Redux ?

We need Redux to manage & handle data layer. We need context to avoid prop drilling. Instead of creating many contexts, we can create one store of Redux.

Cons of Redux

It is hard to set up.
It has a huge learning curve

But now it has come up with Redux Toolkit which is much more easier than Redux.

Difference between Redux & Redux Toolkit

Problems with Redux that Redux Toolkit resolves

1. Configuring a Redux store is too complicated
2. I have to add a lot of packages to get Redux to do anything useful
3. Redux requires too much boiler plate code

Architecture of Redux Toolkit

What is Redux Store ?

It's like a big object where we store data which all the component can access. There will be only one store in one app we will have logic seperations in a store means slices like User Slice, Authentication Slice, Theme Slice, Cart Slice

What is a Slice ?

A slice is a smaller sections of a store when can be used for different purposes

Our components cannot directly modify the store. We have to dispatch and action

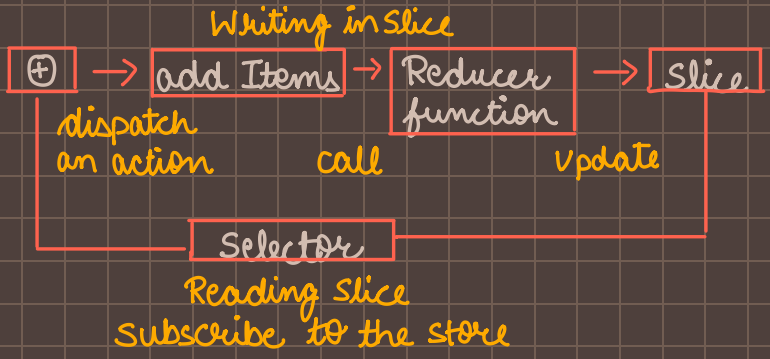
⇒ On clicking a button it will dispatch an action like addItems which will call a function. Then this function will modify the Cart Slice

** The function is reducer

Why?
We need to keep a track of everything. We don't want random components to modify our cart

+button → Dispatches action → Calls reducer function → Update Slice of Redux store

How to read the Slice?
Selector is to be called in order to read the slice



What is Selector?
Selecting the portion of that Slice. It is used to read the data of Slice. Selector is a hook which is a function at the end of the day.

Subscribe to the Store
Means reading from the store. In sync with the store when store will modify the UI will automatically modify

Redux in Project
Install Redux toolkit in project

→ npm i @reduxjs/toolkit Core of redux

→ npm i react-redux Bridge between React & Redux

Creating Store
In Store.js

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

const store = configureStore({})

export default store
```

Providing store to whole app

```
import {Provider} from "react-redux"
import store from "./utils/store";

return <Provider store={store}>
          <App/>
       </Provider>
```

To provider store to our app. It is a bridge between react and redux so it comes from react-redux

Actions are like small apis to communicate with redux store for add, delete, update states

Creating a Slices

In cartSlice.js,

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

const cartSlice = createSlice({
    name: "cart",
    initialState: {
        items: ["Banana", "Apple"]
    },
    reducers: {
        addItem: (state, action) => {
            state.items.push(action.payload);
        },
        removeItem: (state, action) => {
            state.items.pop();
        },
        clearCart: (state) => {
            state.items = [];
        }
    }
})

export const {addItems, removeItem, clearCart} = cartSlice.actions;

export default cartSlice.reducer;
```

```
cartSlice = {
    actions: {addItems, removeItems, clearCart},
    reducer: reducer
}
```

Adding Slice to the store

```
import {configureStore} from "@reduxjs/toolkit"
import cartSlice from "./cartslice";

const store = configureStore ({
        reducer: {
                cart : cartSlice
            }
    })

export default store;
```

Note Revision

-> Create Store
- configureStore () - RTC

-> Provide my store to app
- <Provider store = {store}> - import from react-redux

-> Slice
- RTK - createSlice ({
        name : " "
        initialState : ,
        reducers: {
            addItems : (state, action) => {
                    state = action.payload;
                }
        }
    })

export const {addItem} = cartSlice.actions;
export default cartSlice.reducer;

-> Put that Slice into Store
- {
    reducer : {
            cart : cartSlice,
            user : userSlice
        }
    }

Subscribe to Slice

```
import {useSelector} from "react-redux"

const cartItems = useSelector ( store.cart.items);
```

Update the cart on click

```
import {addItems} from "./utils/cartSlice"
import {useDispatch} from "react-redux"

const dispatch = useDispatch()

const handleAddItem = () => {
      dispatch(addItem("Grapes"));
    };

<button onClick={() => handleAddItem()}>
    Add to Cart
</button>
```

→ Steps
- Install @reduxjs/toolkit and react-redux
- Build our store
- Connect our store to our app
- Slice (cartSlice)
- Dispatch (action)
- Selector