# Redux-toolkit

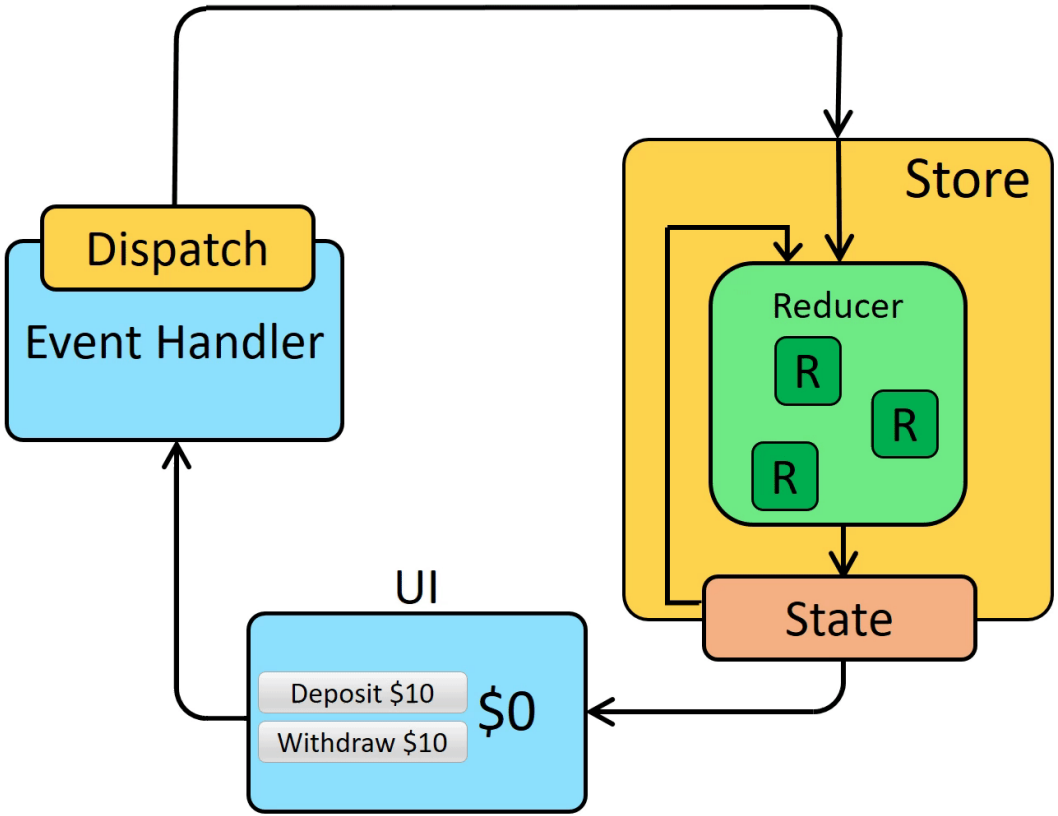
# Redux-toolkit

- Redux Toolkit là một thư viện được cung cấp bởi Redux để giúp việc sử dụng Redux dễ dàng hơn, đặc biệt là khi quản lý trạng thái trong các ứng dụng JavaScript, nhất là với React.

# Redux-toolkit

- Trước đây, Redux yêu cầu viết nhiều mã boilerplate (mã lặp lại không cần thiết) và có cấu trúc hơi phức tạp, nhưng với Redux Toolkit, việc thiết lập và quản lý trở nên đơn giản hơn rất nhiều.

# Redux-toolkit



# Redux-toolkit

- Redux Toolkit cung cấp một số tính năng chính:
  1. configureStore: Giúp thiết lập store với những cài đặt mặc định (middleware và enhancers) mà không cần phải thêm thủ công.

2. createSlice: Tạo ra reducer và các action creators dựa trên một object mô tả các hành động và trạng thái.
3. createAsyncThunk: Giúp dễ dàng xử lý các hành động bất đồng bộ như gọi API mà không cần phải viết nhiều mã phức tạp.

4. combineReducers: Kết hợp các reducers thành một reducer duy nhất để quản lý trạng thái tổng thể.
5. immer: Cho phép bạn thay đổi trực tiếp state bên trong reducer mà không vi phạm nguyên tắc bất biến của Redux.

# Redux-toolkit

- Các bước sử dụng redux – toolkit
  1. Cài đặt
  2. Tạo slice
  3. Cấu hình store
  4. Sử dụng
  5. Cung cấp store cho ứng dụng

# Redux-toolkit

- Bước 1: Sử dụng lệnh: npm install @reduxjs/toolkit react-redux để cài đặt redux toolkit
- Bước 2 tạo slice
  - Tên của  slice: name
  - Khởi tạo giá trị ban đầu
  - Tạo các action
  - Export các action và reducer

# Redux-toolkit

- Bước 3: Cấu hình store, khai báo các reducer.
- Bước 4: Sử dụng useSelector, useDispatch từ react-redux để lấy giá trị và thực hiện các hành động
- Bước 5: Sử dụng Provider từ react-redux và cung cấp store cho Provider

- Ví dụ tăng số lượng trong giỏ hang
- Tạo file cartSlice.js

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


const cartSlice = createSlice({
    name: 'cart',
    initialState: {
        items: [], // tất cả sản phẩm trong giỏ hàng.
        // {id, image, title, price, quantity, totalItem}
        total: 0 //Tổng giá trị trong giỏ hàng
    },
```

# Redux-toolkit

```
reducers:{
    //Thêm và cập sản phẩm trong giỏ hàng
    addUpdateProduct: (state, action) => {
        let item = action.payload; //Lấy thông tin được gửi vào từ action
        let existProduct = state.items.find( i => i.id === item.id);//Tìm sản phẩm trong giỏi hàng
        //Chưa có sản phẩm trong giỏ hàng
        if(!existProduct){
            let money = Math.round(item.price * item.quantity);
            state.items.push({...item, totalItem: money});
            state.total += money;
        }else{
            //Sản phẩm đã có trong giỏ hàng
            if(item.quantity > 0){
                let sLuong = Number(item.quantity - existProduct.quantity)
                let money = Math.round(item.price * sLuong);

                existProduct.quantity +=  sLuong;//số lượng
                existProduct.totalItem +=  money;//Thành tiền
                state.total += money;//Tổng tiền
            }else{
                //item chỉ chứa id
                let item = action.payload; //Lấy thông tin được gửi vào từ action
                //Tìm vị trí sản phẩm trong giỏ hàng
                let indexItem = state.items.findIndex(i => i.id === item.id);
                state.total -= state.items[indexItem].totalItem;
                //Xóa sản phẩm trong giỏ hàng
                state.items.splice(indexItem,1);
            }

        }
    },
```

```
        removeProduct: (state, action) => {
            //item chỉ chứa id
            let item = action.payload; //Lấy thông tin được gửi vào từ action
            //Tìm vị trí sản phẩm trong giỏ hàng
            let indexItem = state.items.findIndex(i => i.id === item.id);
            state.total -= state.items[indexItem].totalItem;
            //Xóa sản phẩm trong giỏ hàng
            state.items.splice(indexItem,1);
        }
    }
})

export const {addUpdateProduct, removeProduct} = cartSlice.actions;
export default cartSlice.reducer;
```

- Tạo store.js

```
import { configureStore } from "@reduxjs/toolkit";
import cartReduce from "./Reducer/cartSlice.js";

const store = configureStore({
    reducer: {
        cart: cartReduce,
    }
})

export default store;
```

- Cài đặt store trong file chạy chính.

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import App from './App.jsx'
import { Provider } from 'react-redux'
import store from './Component/store.js'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <Provider store={store}>
      <App/>
    </Provider>
  </StrictMode>,
)
```

# Sử dụng trong component giỏ hàng

```
import { useDispatch } from "react-redux";
import {addUpdateProduct, removeProduct} from "../../Component/Reducer/cartSlice.js"

let ItemCart = ({data}) => {

    const dispatch = useDispatch();

    let xoaSanPham = (event) => {
        event.preventDefault();
        dispatch(removeProduct({id: data.id}));
    }

    let capNhatSoLuong = (event) => {
        ///{id: samPham.id, price: samPham.price, quantity: 1}
        event.preventDefault();
        let soHienTai =  event.target.value;//số lượng tăng
        dispatch(addUpdateProduct({id: data.id, price: data.price, quantity: soHienTai}));
    }
```

- Sử dụng trong component thêm vào giỏ hàng

```
import { useDispatch } from "react-redux"
import { addUpdateProduct } from "../../Component/Reducer/cartSlice.js";

let ItemProduct = ({samPham}) => {

    const dispatch = useDispatch();

    let themSanPhamVaoGioHang = (event) => {
        event.preventDefault();
        //{id, image, title, price, quantity}
        //Thêm sản phẩm
        dispatch(addUpdateProduct({id: samPham.id, image: samPham.images[0],
            title: samPham.title, price: samPham.price, quantity: 1}))
        alert('Đã thêm sản phẩm vào giỏ hàng');
    }
```