Recoil tutorial: 02 - Code a simple cart 🎉

Agenda:

1. Setup project
2. Setup RecoilRoot
3. Render product list
4. Handle add to cart
5. Render cart info

1. Setup project

- Setup a new project using create-react-app
- Install `recoil` package

```
npm i --save recoil
```

> Recoil official docs: https://recoiljs.org/docs/introduction/getting-started

2. Setup RecoilRoot

Edit `src/index.js` to add `RecoilRoot` component, so that you can use `recoil hooks` anywhere in your app.

```
// src/index.js
import React from 'react';
import ReactDOM from 'react-dom';
import { RecoilRoot } from 'recoil';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <RecoilRoot>
      <App />
    </RecoilRoot>
  </React.StrictMode>,
  document.getElementById('root')
);
```

3. Render product list

Define a `product list state` using `atom`

```
// features/cart/productState.js
import { atom } from "recoil";

export const productListState = atom({
  key: 'productList',
  default: [
    { id: 1, price: 150000, title: 'Áo thun nam' },
    { id: 2, price: 250000, title: 'Áo sơ mi nữ' },
    { id: 3, price: 300000, title: 'Áo khoắc thời trang' },
  ],
});
```

Render `product list state` in component `ProductList`

```
// features/cart/component/ProductList.jsx
function ProductList() {
  const productList = useRecoilValue(productListState);

  return (
    <div>
      <h2>Product List</h2>

      <ul className="product-list">
        {productList.map(product => (
          <li key={product.id}>
            {product.title}
          </li>
        ))}
      </ul>
    </div>
  );
}

export default ProductList;
```

4. Handle add to cart

Define cart state using `atom`

```
// features/cart/cartState.js
import { atom } from 'recoil';

export const cartState = atom({
  key: 'cart',
  // each item in list has 3 keys: id, product and quantity
  default: [],
});
```

Implement `addToCart()` function that take current state and newItem and return a new state.

```
// features/cart/cartState.js
export const addToCart = (cart, item) => {
  const newCart = [...cart];
  const foundIndex = cart.findIndex(x => x.id === item.id);

  // Increase quantity if existing
  if (foundIndex >= 0) {
    newCart[foundIndex] = {
      ...cart[foundIndex],
      quantity: cart[foundIndex].quantity + 1,
    };
    return newCart;
  }

  // Add new item
  newCart.push({
    id: item.id,
    product: item,
    quantity: 1,
  });
  return newCart;
};
```

Add button `Add to cart` to product list

```
// features/cart/component/ProductList.jsx
function ProductList() {
  const productList = useRecoilValue(productListState);

  // 1. Add this handler
  const handleAddToCart = (product) => {}

  return (
    <div>
      <h2>Product List</h2>

      <ul className="product-list">
        {productList.map(product => (
          <li key={product.id}>
            {product.title}

            {/* 2. ADD THIS BUTTON */}
            <button
              style={{ marginLeft: '1rem' }}
              onClick={() => handleAddToCart(product)}
            >
              Add to cart
          </button>
          </li>
        ))}
      </ul>
    </div>
  );
}
```

Implement add to cart handler

```
// features/cart/component/ProductList.jsx
function ProductList() {
  const productList = useRecoilValue(productListState);
  const [cart, setCart] = useRecoilState(cartState); // 1. Get recoil
state

  const handleAddToCart = (product) => {
    const newCart = addToCart(cart, product); // 2. Use helper to create a
new state
    setCart(newCart); // 3. Update recoil state
  }
  // return (...);
}
```

5. Render cart info

Render current cart items in `CartInfo` component

```
// features/cart/components/CartInfo.jsx
function CartInfo(props) {
  const cart = useRecoilValue(cartState);

  return (
    <div>
      <h2>Cart info:</h2>

      <ul className="cart-items">
        {cart.map(item => (
          <li key={item.id}>{item.product.title}: {item.quantity}</li>
        ))}
      </ul>
    </div>
  );
}
```

Define a `selector` that calculate `cart total` base on `cart state`

```
// features/cart/cartState.js
import { atom, selector } from 'recoil';

export const cartState = atom({
  key: 'cart',
  // each item in list has 3 keys: id, product and quantity
  default: [],
});

// NEW CODE HERE
export const cartTotal = selector({
  key: 'cartTotal',
  get: ({ get }) => {
    const cart = get(cartState);

    return cart.reduce((total, item) => {
      return total + (item.product.price * item.quantity);
    }, 0);
  }
});
```

Update CartInfo component to also render cart total

```
function CartInfo(props) {
  const cart = useRecoilValue(cartState);
  const total = useRecoilValue(cartTotal); // 1. Read selector value

  return (
    <div>
      <h2>Cart info:</h2>

      <ul className="cart-items">
        {cart.map(item => (
          <li key={item.id}>{item.product.title}: {item.quantity}</li>
        ))}
      </ul>

      {/* 2. Render it! Sweeeeet! 😍 */}
      <h4>Total: {total} VND</h4>
    </div>
  );
}
```

Cảm ơn các bạn đã xem video của mình!
Nhớ like, share và subscribe để cho bạn bè cùng xem nhen 😉

❤️ Ủng hộ mình làm videos thì đóng góp tại đây nhé:

- Ủng hộ tôi: https://unghotoi.com/easyfrontend
- MoMo/ZaloPay: 0901 309 729

Kết nối với mình:

- 💻 Kênh Easy Frontend: https://youtube.com/easyfrontend
- 🎉 Fan page Facebook: https://www.facebook.com/learn.easyfrontend
- 📝 Nhóm trao đổi Facebook: https://www.facebook.com/groups/easyfrontend