Estrutura do projeto

```
src
├── components
│   ├── Cart.js
│   ├── CartItem.js
├── store
│   ├── cart
│   │   ├── actions.js
│   │   ├── reducer.js
│   └── index.js
├── App.js
└── index.js
```

1. `store/cart/actions.js`

```
export const ADD_ITEM = 'cart/addItem';
export const REMOVE_ITEM = 'cart/removeItem';

export const addItem = (item) => ({
  type: ADD_ITEM,
  payload: item,
});

export const removeItem = (id) => ({
  type: REMOVE_ITEM,
  payload: id,
});
```

2. `store/cart/reducer.js`

```
import { ADD_ITEM, REMOVE_ITEM } from './actions';

const initialState = {
  items: [],
  totalQuantity: 0,
};

const cartReducer = (state = initialState, action) => {
  switch (action.type) {
    case ADD_ITEM:
      const newItem = action.payload;
      const existingItem = state.items.find(item => item.id === newItem.id);

      if (existingItem) {
        return {
          ...state,
          items: state.items.map(item =>
            item.id === newItem.id ? { ...item, quantity: item.quantity + 1 }
: item
          ),
          totalQuantity: state.totalQuantity + 1,
        };
```

```
      } else {
        return {
          ...state,
          items: [...state.items, { ...newItem, quantity: 1 }],
          totalQuantity: state.totalQuantity + 1,
        };
      }

    case REMOVE_ITEM:
      const id = action.payload;
      const itemToRemove = state.items.find(item => item.id === id);

      if (itemToRemove) {
        if (itemToRemove.quantity === 1) {
          return {
            ...state,
            items: state.items.filter(item => item.id !== id),
            totalQuantity: state.totalQuantity - 1,
          };
        } else {
          return {
            ...state,
            items: state.items.map(item =>
              item.id === id ? { ...item, quantity: item.quantity - 1 } : item
            ),
            totalQuantity: state.totalQuantity - 1,
          };
        }
      }
      return state;

    default:
      return state;
  }
};

export default cartReducer;
```

3. store/index.js

```
import { configureStore } from '@reduxjs/toolkit';
import cartReducer from './cart/reducer';

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

export default store;
```

4. components/Cart.js

```
import React from 'react';
import { useSelector } from 'react-redux';
import CartItem from './CartItem';
```

```
const Cart = () => {
  const cartItems = useSelector(state => state.cart.items);
  const totalQuantity = useSelector(state => state.cart.totalQuantity);

  return (
    <div>
      <h2>Carrinho de Compras</h2>
      <p>Total de Itens: {totalQuantity}</p>
      <ul>
        {cartItems.map(item => (
          <CartItem key={item.id} item={item} />
        ))}
      </ul>
    </div>
  );
};

export default Cart;
```

5. components/CartItem.js

```
import React from 'react';
import { useDispatch } from 'react-redux';
import { addItem, removeItem } from '../store/cart/actions';

const CartItem = ({ item }) => {
  const dispatch = useDispatch();

  const handleAddItem = () => {
    dispatch(addItem(item));
  };

  const handleRemoveItem = () => {
    dispatch(removeItem(item.id));
  };

  return (
    <li>
      {item.name} - Quantidade: {item.quantity}
      <button onClick={handleAddItem}>Adicionar</button>
      <button onClick={handleRemoveItem}>Remover</button>
    </li>
  );
};

export default CartItem;
```

6. App.js

```
import React from 'react';
import Cart from './components/Cart';

const App = () => {
  return (
```

```
    <div>
      <h1>Meu E-commerce</h1>
      <Cart />
    </div>
  );
};

export default App;
```

7. index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import { Provider } from 'react-redux';
import store from './store';
import App from './App';

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