

NAME	Ambuj Shukla
UID	23BCS10884
CLASS	622-A

➤ Full Stack PRACTICE 7.2

- CODE

cartSlice.js :::

```
// src/cartSlice.js
import { createSlice } from '@reduxjs/toolkit';

const cartSlice = createSlice({
  name: 'cart',
  initialState: {
    items: [], // [{ id, name, price, quantity }]
  },
  reducers: {
    // Action to add an item or increase quantity
    addItem: (state, action) => {
      const newItem = action.payload; // { id, name, price }
      const existingItem = state.items.find(item => item.id === newItem.id);

      if (existingItem) {
        existingItem.quantity += 1;
      } else {
        state.items.push({ ...newItem, quantity: 1 });
      }
    },

    // Action to remove an item entirely from the cart
```

```

removeItem: (state, action) => {
  const itemId = action.payload;
  state.items = state.items.filter(item => item.id !== itemId);
},

// Action to update quantity (used for input field changes)
updateQuantity: (state, action) => {
  const { id, quantity } = action.payload;
  const itemToUpdate = state.items.find(item => item.id === id);

  if (itemToUpdate) {
    const newQuantity = parseInt(quantity);
    if (newQuantity > 0) {
      itemToUpdate.quantity = newQuantity;
    } else {
      // If quantity is set to 0 or less, remove the item
      state.items = state.items.filter(item => item.id !== id);
    }
  }
},
});

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

// Selector to get the entire cart state
export const selectCartItems = (state) => state.cart.items;

export default cartSlice.reducer;

```

Store.js :::

```

// src/store.js
import { configureStore } from '@reduxjs/toolkit';
import cartReducer from './cartSlice';

export const store = configureStore({
  reducer: {
    cart: cartReducer, // The key 'cart' is what you use in useSelector
                      (state.cart)
  },
});

```

index.js :::

```
// src/index.js (or equivalent entry file)
import React from 'react';
import ReactDOM from 'react-dom/client';
import { Provider } from 'react-redux';
import { store } from './store';
import App from './App';
import './index.css';

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

App.js:

```
// src/App.js
import React from 'react';
import ProductList from './ProductList';
import ShoppingCart from './ShoppingCart';
import './App.css'; // For basic styling

const PRODUCTS = [
  { id: 1, name: 'Laptop', price: 1200 },
  { id: 2, name: 'Mouse', price: 25 },
  { id: 3, name: 'Keyboard', price: 45 },
];

function App() {
  return (
    <div className="container">
      <h1 className="main-title">My Shop</h1>

      <h2>Products</h2>
      <ProductList products={PRODUCTS} />

      <h2>Shopping Cart</h2>
      <ShoppingCart />
    </div>
  );
};
```

```
}  
  
export default App;
```

## ProductList.js :::

```
// src/ProductList.js  
import React from 'react';  
import { useDispatch } from 'react-redux';  
import { addItem } from './cartSlice';  
  
const ProductCard = ({ product }) => {  
  const dispatch = useDispatch();  
  
  const handleAddToCart = () => {  
    // Dispatch the addItem action with the product details  
    dispatch(addItem({  
      id: product.id,  
      name: product.name,  
      price: product.price,  
    }));  
  };  
  
  return (  
    <div className="product-card">  
      <h3>{product.name}</h3>  
      <p>${product.price}</p>  
      <button onClick={handleAddToCart}>Add to Cart</button>  
    </div>  
  );  
};  
  
const ProductList = ({ products }) => {  
  return (  
    <div className="product-grid">  
      {products.map(product => (  
        <ProductCard key={product.id} product={product} />  
      ))}  
    </div>  
  );  
};  
  
export default ProductList;
```

## ShoppingCart.js :::

```
// src/ShoppingCart.js
import React from 'react';
import { useSelector, useDispatch } from 'react-redux';
import { selectCartItems, updateQuantity, removeItem } from './cartSlice';

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

  const handleQuantityChange = (e) => {
    const quantity = e.target.value;
    // Dispatch updateQuantity action
    dispatch(updateQuantity({ id: item.id, quantity }));
  };

  const handleRemove = () => {
    // Dispatch removeItem action
    dispatch(removeItem(item.id));
  };

  return (
    <div className="cart-item">
      <span className="item-details">
        {item.name} (${item.price})
      </span>
      <input
        type="number"
        value={item.quantity}
        min="1"
        onChange={handleQuantityChange}
        className="quantity-input"
      />
      <button onClick={handleRemove} className="remove-button">
        Remove
      </button>
    </div>
  );
};

const ShoppingCart = () => {
  // Use useSelector to subscribe to the cart state
  const cartItems = useSelector(selectCartItems);

  if (cartItems.length === 0) {
```

```

    return <p className="empty-cart-message">Your cart is empty.</p>;
  }

  return (
    <div className="cart-list">
      {cartItems.map(item => (
        <CartItem key={item.id} item={item} />
      ))}
      {/* Optional: Add a total calculation here */}
      <div className="cart-total">
        Total Items: {cartItems.reduce((acc, item) => acc + item.quantity, 0)}
      </div>
    </div>
  );
};

export default ShoppingCart;

```

App.css :::

```

/* src/App.css */
.container {
  max-width: 800px;
  margin: 0 auto;
  padding: 20px;
  font-family: sans-serif;
  text-align: center;
}

.main-title, h2 {
  margin-bottom: 20px;
}

/* Product Grid */
.product-grid {
  display: flex;
  justify-content: center;
  gap: 20px;
  margin-bottom: 40px;
}

/* Product Card */
.product-card {
  border: 1px solid #ccc;

```

```
border-radius: 8px;
padding: 20px;
width: 150px;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
text-align: center;
display: flex;
flex-direction: column;
justify-content: space-between;
align-items: center;
}
```

```
.product-card button {
padding: 8px 15px;
cursor: pointer;
border: 1px solid #000;
background-color: #fff;
border-radius: 4px;
margin-top: 10px;
}
```

```
/* Shopping Cart */
```

```
.cart-list {
display: flex;
flex-direction: column;
gap: 15px;
align-items: center;
}
```

```
.cart-item {
display: flex;
align-items: center;
gap: 10px;
width: 90%;
max-width: 500px;
justify-content: space-between;
padding: 5px 0;
}
```

```
.item-details {
flex-grow: 1;
text-align: left;
}
```

```
.quantity-input {
width: 40px;
```

```

padding: 5px;
text-align: center;
border: 1px solid #ccc;
border-radius: 4px;
}

.remove-button {
padding: 5px 10px;
background-color: #f4f4f4;
border: 1px solid #ccc;
cursor: pointer;
border-radius: 4px;
}

.empty-cart-message {
color: #888;
}

.cart-total {
margin-top: 20px;
font-weight: bold;
border-top: 1px solid #eee;
padding-top: 10px;
}

```

OUTPUT:

# My Shop

## Products

**Laptop**

\$1200

Add to Cart

**Mouse**

\$25

Add to Cart

**Keyboard**

\$45

Add to Cart

## Shopping Cart

Laptop (\$1200)  Remove

Mouse (\$25)  Remove



