1. `import { createSlice } from '@reduxjs/toolkit';`

🔹 **Kya karta hai:**
Redux Toolkit se `createSlice` function import karta hai.

🔹 **Kyoon chahiye:**
Redux Toolkit me hum `createSlice()` use karke state + actions + reducers ek saath define karte hain. Purane tarike se `switch` case waale reducers likhne ki zarurat nahi.

🔹 **Kaha use hoga:**
Ye cart ke liye ek slice (module) banane me use hoga — `cartSlice`.

🔶 2. `const initialState = { items: [], };`

🔹 **Kya karta hai:**
Ye cart slice ka **initial/default state** define karta hai.

🔹 **Kyoon chahiye:**
Redux store me cart initially empty hota hai, isliye `items` array khaali hai.

🔹 **Project me kya hoga:**
User jab tak "Add to Cart" nahi karega, cart empty rahega.
Example:

js
CopyEdit

```
state.cart.items = []
```

🔹 **UI par:**
Cart page par `Your cart is empty!` dikhai dega.

🔶 3. `const cartSlice = createSlice({ ... })`

🔹 **Kya karta hai:**
Redux slice banata hai jisme:

- name (slice name),
- initialState,
- reducers (jo actions handle karte hain)

🔷 **Kyoon chahiye:**
Ek hi jagah pe state + logic rakhne ke liye.

🔷 **Project me kaam:**
Ye slice define karega `addToCart`, `removeFromCart`, etc. jo Redux actions ke roop me kaam karenge.

🔽 REDUCERS Start (Real Life Actions)

🔷 `addToCart: (state, action) => { ... }`

1️⃣ **`const newItem = action.payload;`**

🔷 **Kya karta hai:**
Jab tu product add karta hai cart me, to wo product ka data aata hai `action.payload` me.
Example:

js
CopyEdit

```js
{
  id: 1,
  title: "Nike Shoes",
  price: 2500,
  img: "nike.jpg"
}
```

🔷 **UI:**
User ne "Add to Cart" dabaya toh product ka data yaha aata hai.

2️⃣ **`const existingItem = state.items.find(item => item.id === newItem.id);`**

🔷 **Kya karta hai:**
Check karta hai ki kya ye product pehle se cart me hai ya nahi.

🔷 **Project me:**
Agar same product dobara add ho, to quantity badhani hai — naya item nahi banana.

3️⃣ **`if (existingItem) { existingItem.quantity += 1; }`**

🔷 **Kya karega:**
Agar item already cart me hai, toh uski quantity +1 kar dega.

🔷 **Example:**
Nike Shoes quantity 1 thi → Ab 2 ho jaayegi.

4️⃣ `else { state.items.push({ ...newItem, quantity: 1 }); }`

🔷 **Kya karega:**
Agar item pehli baar cart me add ho raha hai, to `items[]` array me push karega with quantity = 1.

🔷 **UI Output:**
Cart page me naye product ki image, title, price, quantity sab dikhai denge.

🔷 `removeFromCart: (state, action) => { ... }`

1️⃣ `const idToRemove = action.payload;`

🔷 **Kya karega:**
Ye line fetch karti hai us item ka `id` jise delete karna hai.

2️⃣ `state.items = state.items.filter(item => item.id !== idToRemove);`

🔷 **Kya karega:**
Cart me se wo item hata dega jiska id match karta hai.

🔷 **UI Output:**
"Remove" button dabane ke baad wo item cart se gayab ho jaayega.
Agar cart empty ho gaya toh `Your cart is empty!` dikh jaayega.

🔷 `increaseQuantity: (state, action) => { ... }`

1️⃣ `const id = action.payload;`

🔷 **Kya karega:**
Ye line button click se aaye `id` ko store karti hai.

2️⃣ `const item = state.items.find(item => item.id === id);`

🔷 **Kya karega:**
Us item ko dhoondhta hai jiska id match karta hai.

3️⃣ `item.quantity += 1;`

🔷 **Kya karega:**
Uski quantity +1 kar deta hai.

🔷 **UI Output:**
`+` button dabane pe quantity number badh jaayega, aur uska **total price bhi badhega.**

🔷 `decreaseQuantity: (state, action) => { ... }`

1️⃣ `const id = action.payload;`

🔷 **Same as above** — button click se mila id

2️⃣ `const item = state.items.find(item => item.id === id);`

🔷 **Same** — item dhoondhna

3️⃣ `if (item.quantity > 1) { item.quantity -= 1; }`

🔷 **Kya karega:**
Quantity 1 se zyada hai toh -1 kar dega

4️⃣ `else { state.items = state.items.filter(item => item.id !== id); }`

🔷 **Kya karega:**
Agar quantity 1 thi, aur tu `-` dabata hai → item remove ho jaayega.

🔷 **UI Output:**

- Quantity kam ho jaayegi.
- Agar 0 hui toh item cart se chala jaayega.

🔷 Exports:

js
CopyEdit

```
export const { addToCart, removeFromCart, increaseQuantity,
decreaseQuantity } = cartSlice.actions;
```

```
export default cartSlice.reducer;
```

🔷 **Kya karega:**

- Ye actions ko bahar export karta hai taaki tu unhe use kar sake `dispatch()` ke saath.
- Reducer ko export karta hai taaki tu `store.js` me add kar sake.

🧠 BONUS: Project me Ye Lines Kab Call Hongi?

Line in cartSlice	Kab Call Hogi	Kis Component se
`addToCart(payload)`	ProductList ya ProductDetail me "Add to Cart" dabane par	`ProductCard.js`, `ProductPage.js`
`removeFromCart(id)`	CartPage me "Remove" button dabane par	`CartPage.js`
`increaseQuantity(id)`	CartPage me + button dabane par	`CartPage.js`
`decreaseQuantity(id)`	CartPage me – button dabane par	`CartPage.js`

✅ CartPage UI Example:

text

CopyEdit

```
🛍️ Your Cart

[Nike Shoes] ₹2500
[ - ] 2 [ + ]
Total: ₹5000
[Remove]

Total Cart Value: ₹5000
[Buy Now]
```

Bhai ab batao:
 Kya main is tarah se `ProductList` ka bhi

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

const initialState = {
  items: [],
};

const cartSlice = createSlice({
  name: 'cart',
  initialState,
  reducers: {
    addToCart: (state, action) => {
      const newItem = action.payload;
      // Jab tu product add karta hai cart me, to wo product ka data aata hai action.payload me.

      const existingItem = state.items.find(item => item.id === newItem.id);


// Check karta hai ki kya ye product pehle se cart me hai ya nahi.

// Agar same product dobara add ho, to quantity badhani hai — naya item nahi banana.

//
      if (existingItem) {
        existingItem.quantity += 1;
```

```
         //  Kya karega:
// Agar item already cart me hai, toh uski
quantity +1 kar dega.


// Nike Shoes quantity 1 thi → Ab 2 ho jaayegi.

      } else {
        state.items.push({ ...newItem, quantity: 1
});
      }
     },

    //  Kya karega:
// Agar item pehli baar cart me add ho raha hai,
to items[] array me push karega with quantity = 1.

// 🔹 UI Output:
// Cart page me naye product ki image, title,
price, quantity sab dikhai denge.




    removeFromCart: (state, action) => {
      const idToRemove = action.payload;
      // Ye line fetch karti hai us item ka id
jise delete karna hai.
```

```
      state.items = state.items.filter(item => item.id !== idToRemove);

      // Cart me se wo item hata dega jiska id match karta hai.

// 🔷 UI Output:
// "Remove" button dabane ke baad wo item cart se gayab ho jaayega.
// Agar cart empty ho gaya toh Your cart is empty! dikh jaayega.


    },
    increaseQuantity: (state, action) => {
      const id = action.payload;
      // Ye line button click se aaye id ko store karti hai.


      const item = state.items.find(item => item.id === id);
      // Us item ko dhoondhta hai jiska id match karta hai.


      if (item) {
        item.quantity += 1;
      }
     },
    //  Us item ko dhoondhta hai jiska id match
karta hai.

// + button dabane pe quantity number badh
jaayega, aur uska total price bhi badhega.


    decreaseQuantity: (state, action) => {
      const id = action.payload;

      // const id = action.payload;
// 🔹 Same as above — button click se mila id
      const item = state.items.find(item =>
item.id === id);
      // 🔹 Same — item dhoondhna


      if (item) {
        if (item.quantity > 1) {
          item.quantity -= 1;
          // Quantity 1 se zyada hai toh -1 kar
dega



        } else {
          state.items = state.items.filter(item =>
item.id !== id);
          // Agar quantity 1 thi, aur tu - dabata
hai → item remove ho jaayega.
// 🔷 UI Output:

// Quantity kam ho jaayegi.

// Agar 0 hui toh item cart se chala jaayega.


        }
      }
    },
  },
});

export const { addToCart, removeFromCart,
increaseQuantity, decreaseQuantity } =
cartSlice.actions;
export default cartSlice.reducer;
//

/*
🔷 Kya karta hai:
Redux Toolkit se createSlice function import karta
hai.

🔷 Kyoon chahiye:
Redux Toolkit me hum createSlice() use karke state
+ actions + reducers ek saath define karte hain.
Purane tarike se switch case waale reducers likhne
ki zarurat nahi.
```

🔹 Kaha use hoga:
Ye cart ke liye ek slice (module) banane me use hoga — cartSlice.

🔶 2. const initialState = { items: [], };
🔹 Kya karta hai:
Ye cart slice ka initial/default state define karta hai.

🔹 Kyoon chahiye:
Redux store me cart initially empty hota hai, isliye items array khaali hai.

🔹 Project me kya hoga:
User jab tak "Add to Cart" nahi karega, cart empty rahega.
Example:

js
Copy
Edit
state.cart.items = []
🔹 UI par:
Cart page par Your cart is empty! dikhai dega.

🔶 3. const cartSlice = createSlice({ ... })
🔹 Kya karta hai:
Redux slice banata hai jisme:

name (slice name),

```
initialState,

reducers (jo actions handle karte hain)

🔷 Kyoon chahiye:
Ek hi jagah pe state + logic rakhne ke liye.*/

// 🔷 Project me kaam:
// Ye slice define karega addToCart,
removeFromCart, etc. jo Redux actions ke roop me
kaam karenge
```