Assignment # 01
Mobile Application Development
Sir Kamran
Muhammad Moaaz Safdar
Sp22-bse-053
22 September, 2024

Code

```
class ShoppingCart {
constructor() {
this.cart = [];
}
// 1. Add Items to the Cart//

addItem(productId, productName, quantity, price) {
const product = { productId, productName, quantity, price };
this.cart.push(product);
}

// 2. Remove and Update Items//

removeItem(productId) {
const index = this.cart.findIndex((product) =>
product.productId === productId);
if (index !== -1) {
this.cart.splice(index, 1);
}
}
updateQuantity(productId, newQuantity) {
const product = this.cart.find((product) => product.productId
=== productId);
if (product) {
product.quantity = newQuantity;
}
}

// 3. Calculate Total Cost//

calculateTotalCost() {return this.cart.reduce((acc, product) => acc
+ (product.price
* product.quantity), 0);
```

```
}

// 4. Display Cart Summary//

getCartSummary() {
const summary = this.cart
.filter((product) => product.quantity > 0)
.map((product) => ({
productName: product.productName,
quantity: product.quantity,
totalPrice: product.price * product.quantity,
}));
return summary;
}

// 5. Bonus (Optional): Apply Discount Code//

applyDiscountCode(discountCode) {
const discountAmount =
this.getDiscountAmount(discountCode);
return this.calculateTotalCost() - discountAmount;
}
getDiscountAmount(discountCode) {
// implement logic to retrieve discount amount based on
discount code
// for example:
const discounts = {
"SUMMER10": 10,
"WINTER20": 20,
};
return discounts[discountCode] || 0;
}}
```

// Create a new shopping cart instance//

```
const cart = new ShoppingCart();
```

// Add items to the cart//

```
cart.addItem(1, "Product A", 2, 10.99);
cart.addItem(2, "Product B", 3, 9.99);
cart.addItem(3, "Product C", 1, 19.99);
```

// Display cart summary//

```
console.log("Initial Cart Summary:");
console.log(cart.getCartSummary());
```

// Update quantity of an item//

```
cart.updateQuantity(2, 2);
```

// Display updated cart summary//

```
console.log("Updated Cart Summary:");
console.log(cart.getCartSummary());
```

// Calculate total cost//

```
console.log("Total Cost: $" + cart.calculateTotalCost());
```

// Apply discount code//

```
console.log("Total Cost with Discount (SUMMER10): $" +
cart.applyDiscountCode("SUMMER10"));
```

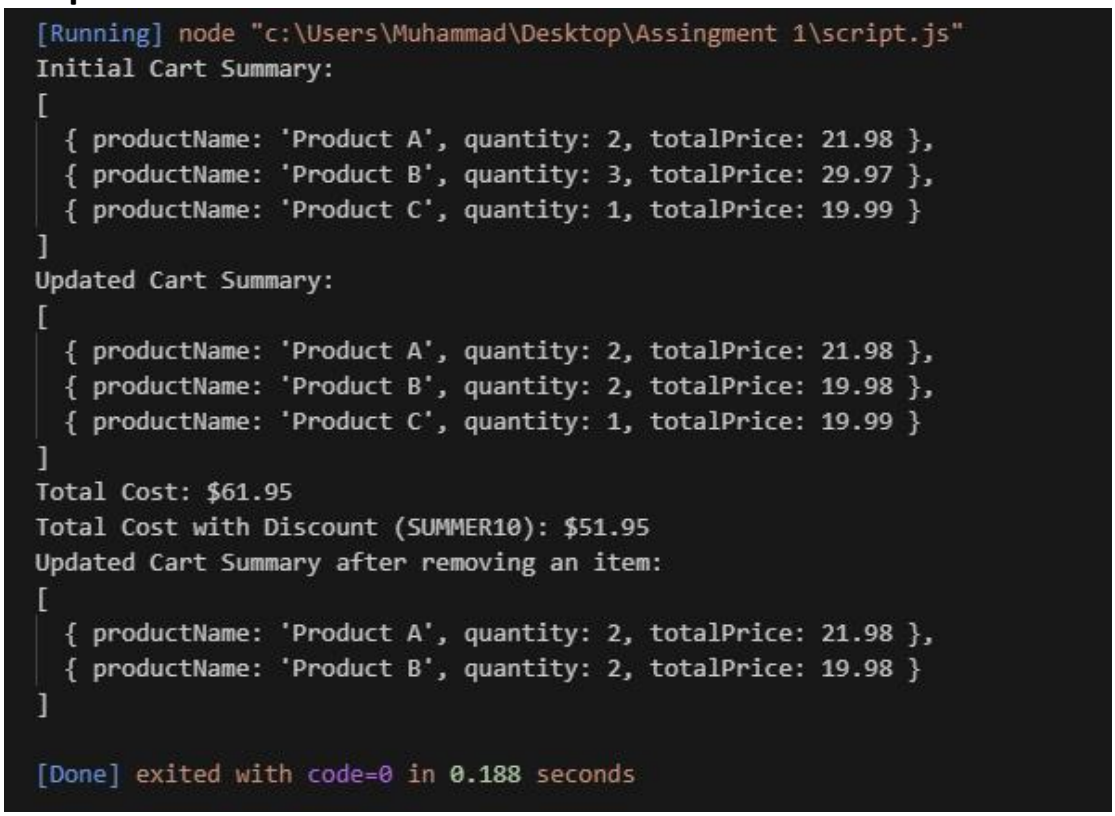
// Remove an item from the cart//

```
cart.removeItem(3);
```

// Display updated cart summary//

```
console.log("Updated Cart Summary after removing an item:");
console.log(cart.getCartSummary());
```

Output



```
[Running] node "c:\Users\Muhammad\Desktop\Assingment 1\script.js"
Initial Cart Summary:
[
  { productName: 'Product A', quantity: 2, totalPrice: 21.98 },
  { productName: 'Product B', quantity: 3, totalPrice: 29.97 },
  { productName: 'Product C', quantity: 1, totalPrice: 19.99 }
]
Updated Cart Summary:
[
  { productName: 'Product A', quantity: 2, totalPrice: 21.98 },
  { productName: 'Product B', quantity: 2, totalPrice: 19.98 },
  { productName: 'Product C', quantity: 1, totalPrice: 19.99 }
]
Total Cost: $61.95
Total Cost with Discount (SUMMER10): $51.95
Updated Cart Summary after removing an item:
[
  { productName: 'Product A', quantity: 2, totalPrice: 21.98 },
  { productName: 'Product B', quantity: 2, totalPrice: 19.98 }
]

[Done] exited with code=0 in 0.188 seconds
```

Objective of the Code

The primary objective of this ShoppingCart class is to simulate an online shopping cart. It allows users to perform essential cart operations such as adding items, updating quantities, removing items, calculating the total cost, and applying discount codes.

Operations Implemented

1. addItem(productId, productName, quantity, price):

This function adds a new product to the cart by creating a product object with relevant details and pushing it into the cart array.
Logic: Each product is represented as an object that includes a unique product ID, name, quantity, and price. It is then stored in the cart array.

2. removeItem(productId):

This function removes an item from the cart by finding the product based on its productId and removing it from the cart array.
Logic: It uses findIndex() to locate the product by its productId and removes the item using splice() if found.

3. updateQuantity(productId, newQuantity):

This function updates the quantity of a specific product in the cart.
Logic: It searches for the product by productId using find() and updates its quantity if the product exists.

4. calculateTotalCost():

This function calculates the total cost of all items in the cart.
Logic: It uses reduce() to sum up the product of price * quantity for each product in the cart array.

5. getCartSummary():

This function returns a summary of the cart, including each product's name, quantity, and total price.
Logic: It filters out products with a quantity of 0 and maps the remaining products into a simplified structure for the summary.

6. applyDiscountCode(discountCode):

This function applies a discount to the total cost based on a discount code.
Logic: It first calculates the total cost and subtracts a discount amount retrieved by the getDiscountAmount() method, which checks if the code is valid.

7. getDiscountAmount(discountCode):

This helper function retrieves a discount amount based on the provided discount code.
Logic: It checks against a predefined list of discount codes (e.g., SUMMER10, WINTER20) and returns the corresponding discount value. If the code is not valid, it returns 0.

Conclusion
Through this assignment, I learned how to implement key shopping cart functionalities using JavaScript. I deepened my understanding of object manipulation, array methods (e.g., find(), findIndex(), reduce()), and how to structure logic for handling different operations.

Challenges faced:

Managing the state of the cart and ensuring that updates to quantities and prices worked seamlessly