

Shopping cart implementation



September 26, 2024

LAVEEZA BUKHARI

SP22-BSE-002

**Introduction:**

**Objective:** The objective of this assignment is to create a JavaScript program that simulates a mobile shopping cart feature, allowing users to add, remove, and update items in their cart, as well as calculate the total cost and apply discounts.

**Operations Implemented:**

**Add Item to Cart:** Allows users to add products to their cart with a specified quantity and price.

Remove Item from Cart: Enables users to remove products from their cart by product ID.

**Update Item Quantity:** Updates the quantity of a product in the cart.

**Display Cart Summary:** Displays a summary of the products in the cart, including product name, quantity, and total product price.

**Calculate Total Cost:** Calculates the total cost of all products in the cart.

**Apply Discount:** Applies a discount code to the total cost, reducing the total amount payable.

**Code Explanation:**

Here's a breakdown of the logic behind each function implemented in the code:

**1. addItemToCart(product):**

This function takes a product object as an argument, which contains the product's ID, name, quantity, and price.The function adds the product to the cart array using the push() method.

It then shows a message to the console indicating that the product has been added to the cart.

**Logic:** The function simply adds the product to the cart array, without checking for duplicates or validating the product data.

**2. removeItemFromCart(productId):**

This function takes a productId as an argument, which is used to identify the product to be removed from the cart.The function uses the findIndex() method to find the index of the product with the matching ID in the cart array.If the product is found, it is removed from the cart using the splice() method.The function logs a message to the console indicating whether the product was found and removed or not.

**Logic:** The function searches for the product by ID, removes it if found, and logs a message to indicate the outcome.

**3. updateItemQuantity(productId, newQuantity):**

This function takes a productId and a newQuantity as arguments, which are used to update the quantity of the product in the cart.The function uses the find() method to find the product with the matching ID in the cart array.If the product is found, its quantity property is updated to the new value.The function logs a message to the console indicating whether the product was found and updated or not.

**Logic:** The function searches for the product by ID, updates its quantity if found, and logs a message to indicate the outcome.

**4. displayCartSummary():**

This function returns an array of product summaries, which include the product name, quantity, and total product price.The function uses the filter() method to exclude products with a quantity of 0 from the summary.It then uses the map() method to transform each product into a summary object with the required properties.

**Logic:** The function filters out products with zero quantity and transforms the remaining products into summary objects.

**5. calculateTotalCost():**

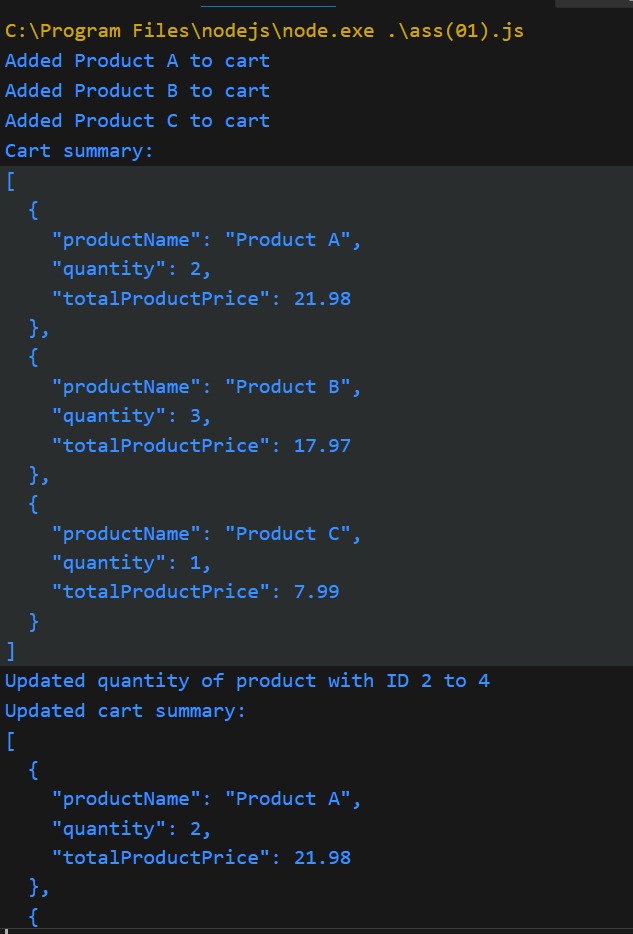
This function returns the total cost of all products in the cart. The function uses the reduce() method to iterate over the cart array and calculate the total cost by summing up the product of each product's price and quantity.

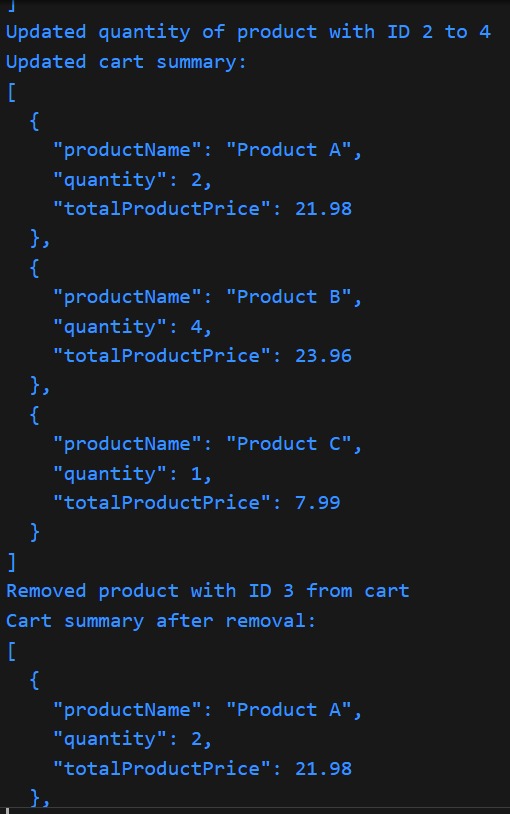
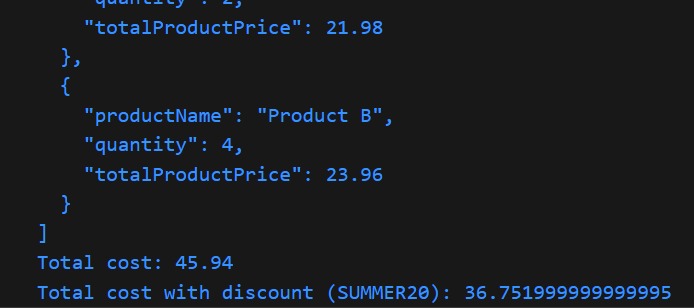
**Logic:** The function iterates over the cart array, calculates the total cost for each product, and return. Here's a breakdown of the logic behind each function implemented in the code:

**6. applyDiscount(discountCode):**

This function takes a discountCode as an argument, which is used to apply a discount to the total cost.The function calculates the total cost using the calculateTotalCost() function.It then applies the discount based on the discount code, using a switch statement to determine the discount amount.The function returns the total cost with the discount applied.

**Logic:** The function calculates the total cost, applies the discount based on the code, and returns the discounted total cost.

**Screenshots:**

****

**Conclusion:**

**Reflection on the Assignment:**

Through this assignment, I learned how to implement a JavaScript-based mobile shopping cart feature that utilizes ES6 arrow functions, array methods (map, filter, reduce), and object manipulation to manage items in a shopping cart. The assignment helped me understand the importance of breaking down complex tasks into smaller, manageable functions, each with a specific responsibility.

**Key Takeaways:**

* I learned to divide the code into separate functions, each responsible for a specific task, such as adding items to the cart, removing items, updating quantities, calculating the total cost, and displaying the cart summary. This modular approach made the code more organized, readable, and maintainable.
* I gained a deeper understanding of how to effectively use array methods like map, filter, and reduce to manipulate and transform data in the cart. These methods allowed me to perform complex operations in a concise and efficient manner.
* I practiced working with objects and their properties, including creating new objects, updating existing ones, and accessing their properties using dot notation or bracket notation.
* I became more comfortable using ES6 arrow functions, which provided a concise way to define functions and improved code readability.
* I learned the importance of organizing code in a logical and structured way, using comments to explain the logic and functionality of each function, and making the code easy to understand and maintain.

**Challenges Faced:**

One of the biggest challenges was managing the complexity of the assignment, which involved multiple functions and interactions between them. I had to ensure that each function worked correctly and that the overall system functioned as expected. Debugging was another challenge, as I had to identify and fix errors in the code. I used console logs and the browser's developer tools to debug the code and ensure that it worked correctly.

**Conclusion:**

This assignment helped me develop my skills in JavaScript, particularly in using ES6 arrow functions, array methods, and object manipulation to build a complex system like a mobile shopping cart feature. I overcame challenges related to managing complexity, debugging, and balancing code readability and conciseness. The experience has prepared me to tackle more complex assignments and projects in the future.