



COMSATS UNIVERSITY ISLAMABAD, ATTOCK CAMPUS

PREPARED BY:

Muhammad Sufyan (SP22-BSE-045)

SUBJECT:

Mobile Application Development

SUBMITTED TO:

MR Muhammad Kamran

DATE:

26th September, 2024

ASSIGNMENT #01

Introduction

The objective of this assignment is to create a simple shopping cart system that allows users to add, remove, and update items in their cart, as well as calculate the total cost of the items. The system consists of five main operations:

1. Add items to the cart
2. Remove items from the cart
3. Update the quantity of items in the cart
4. Calculate the total cost of the items in the cart
5. Display a summary of the cart contents

These operations are implemented using JavaScript functions which are explained

Code Explanation

1. Add Items to the Cart:

Four parameters are required by the `addItem` function: `productId`, `productName`, `quantity`, and `price`. Using these parameters, a new product object is created and added to the `cart` array with the `push` method. To verify that the item has been put to the cart, it finally sends a message to the console.

2. Remove Items from the Cart:

The `removeItem` function takes one parameter: **`productId`**. It uses the **`findIndex`** method to find the index of the product with the specified **`productId`** in the **`cart`** array. If the product is found, it removes it from the array using the **`splice`** method and logs a message to the console to confirm that the item has been removed. If the product is not found, it sends a message to the console indicating that the product was not found.

3. Update the Quantity of Items in the Cart:

The `updateItemQuantity` function takes two parameters: `productId` and `newQuantity`. It uses the `find` method to find the product with the specified `productId` in the `cart` array. If the product is found, it updates its quantity to the specified `newQuantity` and logs a message to the console to confirm that the quantity has been updated. If the product is not found, it logs a message to the console indicating that the product was not found.

4. Calculate the Total Cost of the Items in the Cart:

The `calculateTotalCost` function uses the `reduce` method to calculate the total cost of the items in the `cart` array. It initializes the accumulator to 0 and then iterates over each product in the array, adding the product's quantity multiplied by its price to the accumulator. Finally, it returns the total cost.

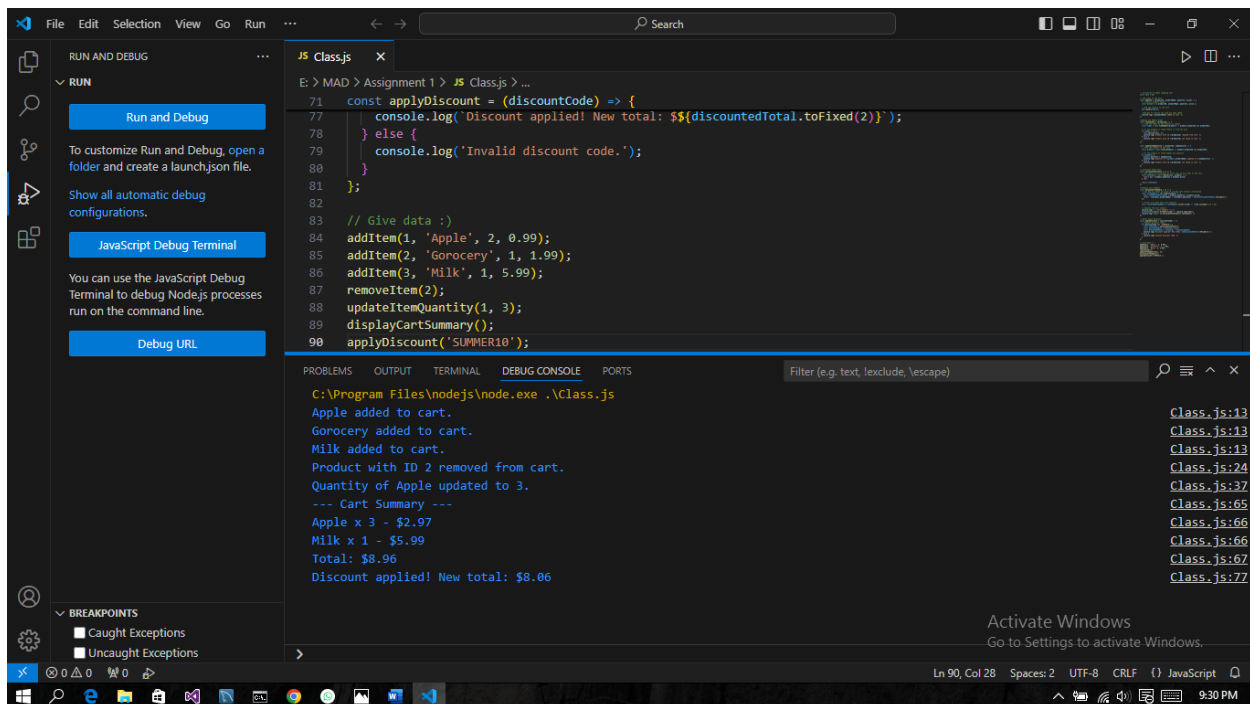
5. Display a Summary of the Cart Contents:

The `displayCartSummary` function uses the `map` method to create an array of strings representing the cart summary. Each string includes the product name, quantity, and total price for the product. It then uses the `filter` method to remove any items with a quantity of 0. Finally, it logs the cart summary to the console, including the total cost.

6. Bonus: Apply Discount:

The `applyDiscount` function takes one parameter: `discountCode`. If the discount code is 'SUMMER10', it calculates the total cost of the items in the cart, applies a 10% discount, and logs a message to the console with the new total cost. If the discount code is invalid, it logs a message to the console indicating that the discount code is invalid.

Output Screenshot:



The screenshot shows the Visual Studio Code editor with the `Class.js` file open. The code defines several functions for managing a shopping cart, including `addItem`, `removeItem`, `updateItemQuantity`, `displayCartSummary`, and `applyDiscount`. The `applyDiscount` function is highlighted, showing its logic for applying a 10% discount for the 'SUMMER10' code.

The Debug Console at the bottom shows the output of the code execution:

```
C:\Program Files\nodejs\node.exe .\Class.js
Apple added to cart.
Grocery added to cart.
Milk added to cart.
Product with ID 2 removed from cart.
Quantity of Apple updated to 3.
--- Cart Summary ---
Apple x 3 - $2.97
Milk x 1 - $5.99
Total: $8.96
Discount applied! New total: $8.06
```

Conclusion:

in this assignment, I gained hands-on experience with JavaScript, specifically with array methods and functions. I learned how to:

1. **Create and manage a shopping cart:** I implemented functions to add, remove, and update items in the cart, which helped me understand how to manipulate arrays in JavaScript.
2. **Use array methods:** I applied various array methods, such as `push`, `splice`, `map`, `findIndex`, and `filter`, to perform different operations on the cart array.

3. **Implement a discount system:** I created a function to apply a discount to the total price, which introduced me to conditional statements and basic arithmetic operations in JavaScript.
4. **Organize code:** I structured the code into separate functions, each with a specific responsibility, which improved the code's readability and maintainability.

Challenges Faced:

1. **Understanding array methods:** Initially, I struggled to grasp the differences between various array methods, such as `map` and `forEach`. However, through practice and experimentation, I gained a deeper understanding of their usage and applications.
2. **Managing state:** I encountered issues with updating the cart state correctly, particularly when removing or updating items. I had to revisit the code and ensure that I was updating the correct references to the cart array.
3. **Debugging:** I encountered errors due to incorrect indexing or out-of-bounds access. I learned to use the browser's console and debugging tools to identify and resolve these issues.
4. **Code organization:** As the codebase grew, I had to reorganize the functions and variables to maintain a clean and readable structure. This helped me develop a better understanding of code organization and modularization.