

**visual programming project report**

**Topic: Shopping Cart**



**bscs 3rd semester**

malaika Batool

fatima rani

owaif amir

**Report on Visual Programming Project: Shopping Cart System**

**1. Introduction**

The Shopping Cart System is a console-based application developed using C# to simulate the functionality of a typical online shopping cart. The system allows users to browse available products, add items to their cart, manage cart contents, view the cart total, and complete the checkout process. It also incorporates features like sales tax calculation, discounts, and cart expiration.

**Purpose:**

The primary goal of this project is to demonstrate an understanding of object-oriented programming (OOP) principles such as encapsulation, inheritance, and modular programming by building a functional shopping cart system.

**2. System Overview**

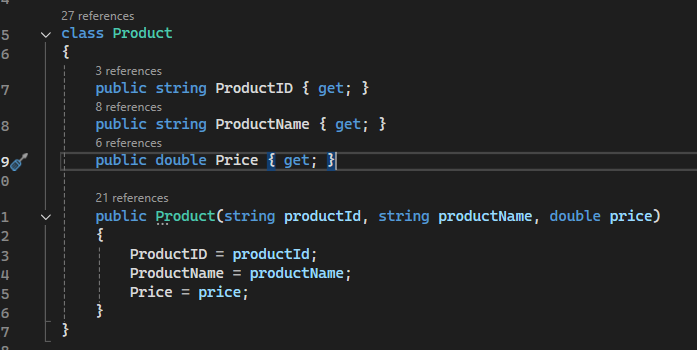
**Features:**

1. **Product Management:**
   * *Displays a list of available products.*
   * *Allows adding products to the cart.*
2. **Cart Management:**
   * *View items in the cart, with real-time calculations of subtotal, discounts, sales tax, and grand total.*
   * *Modify or remove items from the cart.*
   * *Clear cart upon checkout or expiration.*
3. **Expiration Handling:**
   * *Automatically clears the cart if no activity is detected within 30 minutes.*
4. **Financial Calculations:**
   * ***Sales Tax:*** *A tax of 8% is applied to all purchases.*
   * ***Discount:*** *A 10% discount is applied to the subtotal before tax.*

**3. Classes and Architecture**

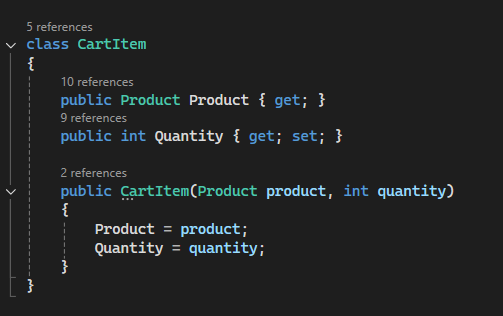
The application is designed around two core classes: Product and CartItem, with the ShoppingCart class acting as the main controller of the system.

**Product Class**

****

* **Attributes:**
  + *ProductID: Uniquely identifies the product.*
  + *ProductName: Stores the name of the product.*
  + *Price: Stores the price of the product.*

**CartItem Class**

****

* **Attributes:**
  + *Product: Holds the product object being added to the cart.*
  + *Quantity: Stores the number of units of that product.*

**ShoppingCart Class**

This class contains the business logic and manages user interactions with the system.

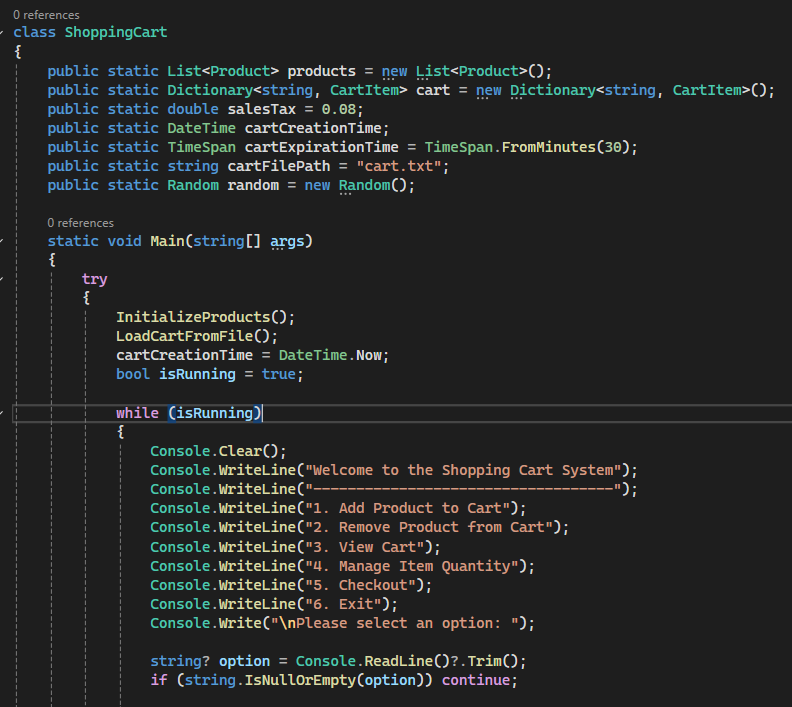
**Key Variables:**

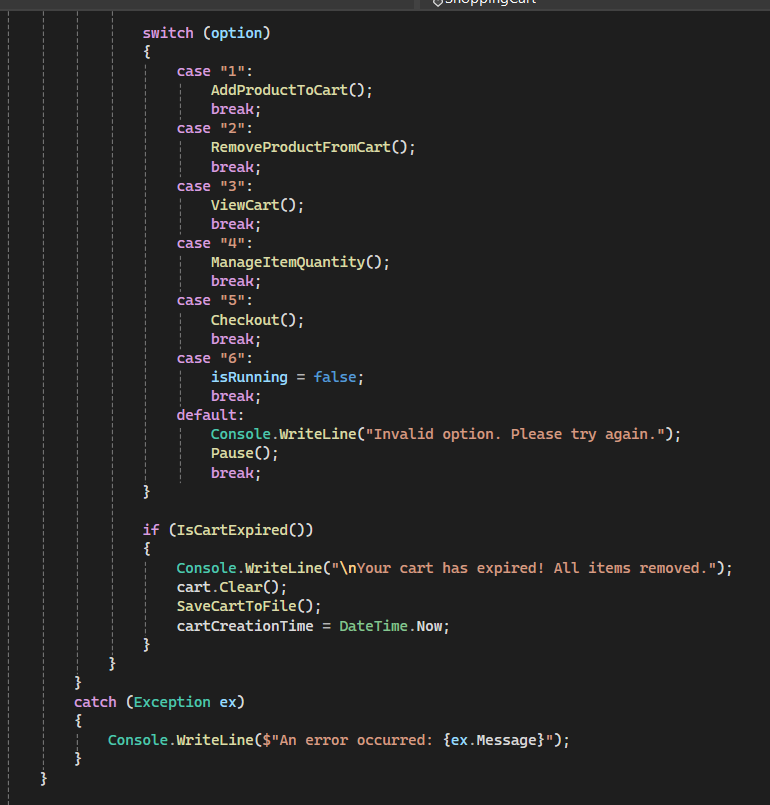
* products: A static list of available products.
* cart: A dictionary storing items in the cart, where the product ID acts as the key.
* salesTax and discount: Static variables representing the sales tax and discount rates.
* cartCreationTime: Stores the timestamp when the cart is created.
* cartExpirationTime: A static TimeSpan representing the expiration period of 30 minutes.

**Methods:**

1. **Main Menu:**
   * Displays options for the user to interact with the cart.
2. **InitializeProducts():**
   * Populates the product list with five predefined products.
3. **AddProductToCart():**
   * Adds a selected product to the user's cart after verifying the product ID and quantity.
4. **RemoveProductFromCart():**
   * Removes a specific item from the cart based on the product ID.
5. **ViewCart():**
   * Displays all the items in the cart, along with a breakdown of subtotal, discount, sales tax, and grand total.
6. **ManageItemQuantity():**
   * Allows the user to update the quantity of a product in the cart.
7. **Checkout():**
   * Displays the cart total and prompts the user to proceed with checkout.
8. **IsCartExpired():**
   * Checks if the current time exceeds the cart's expiration time (30 minutes), and clears the cart if it has expired.
9. **Pause():**
   * Pauses the console output and waits for user input to proceed.

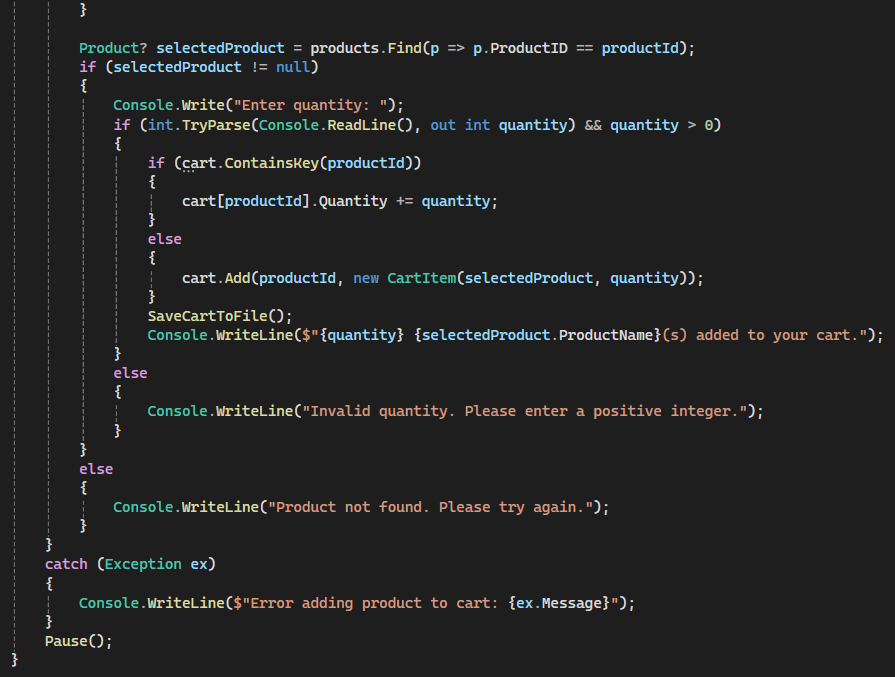
**C++ CODE:**

****

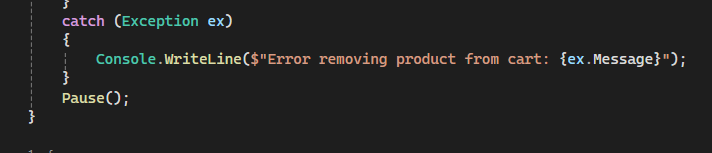
****

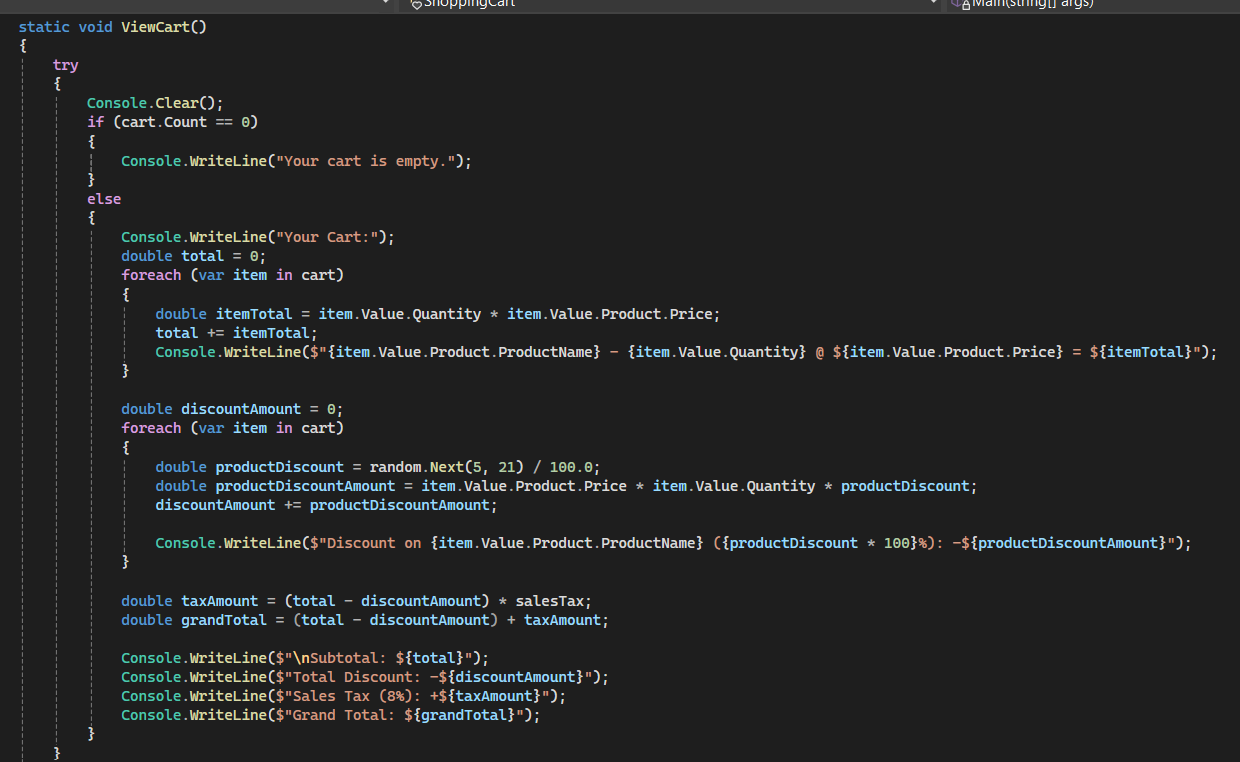
****

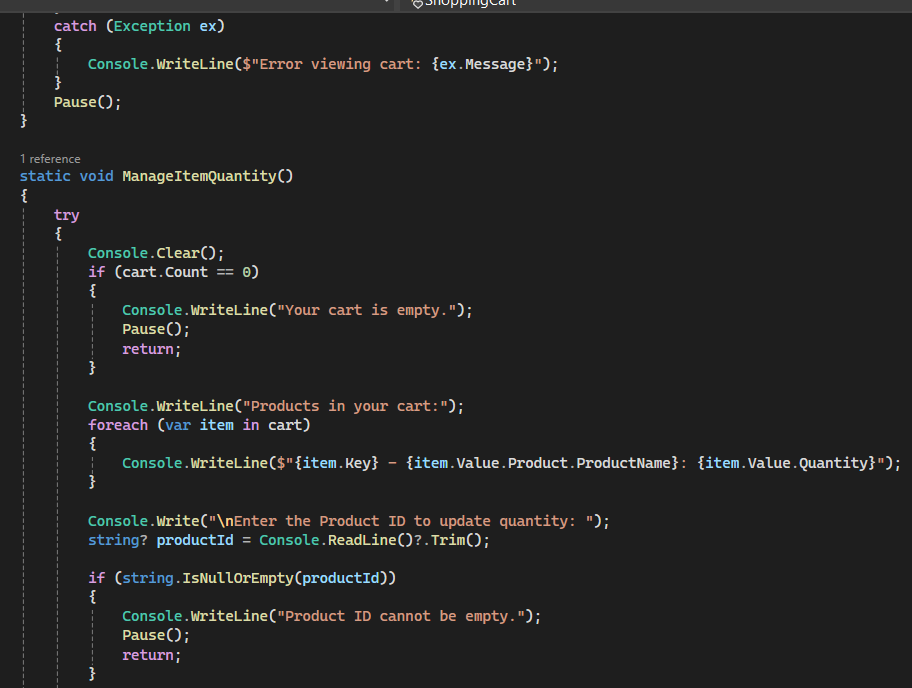
****

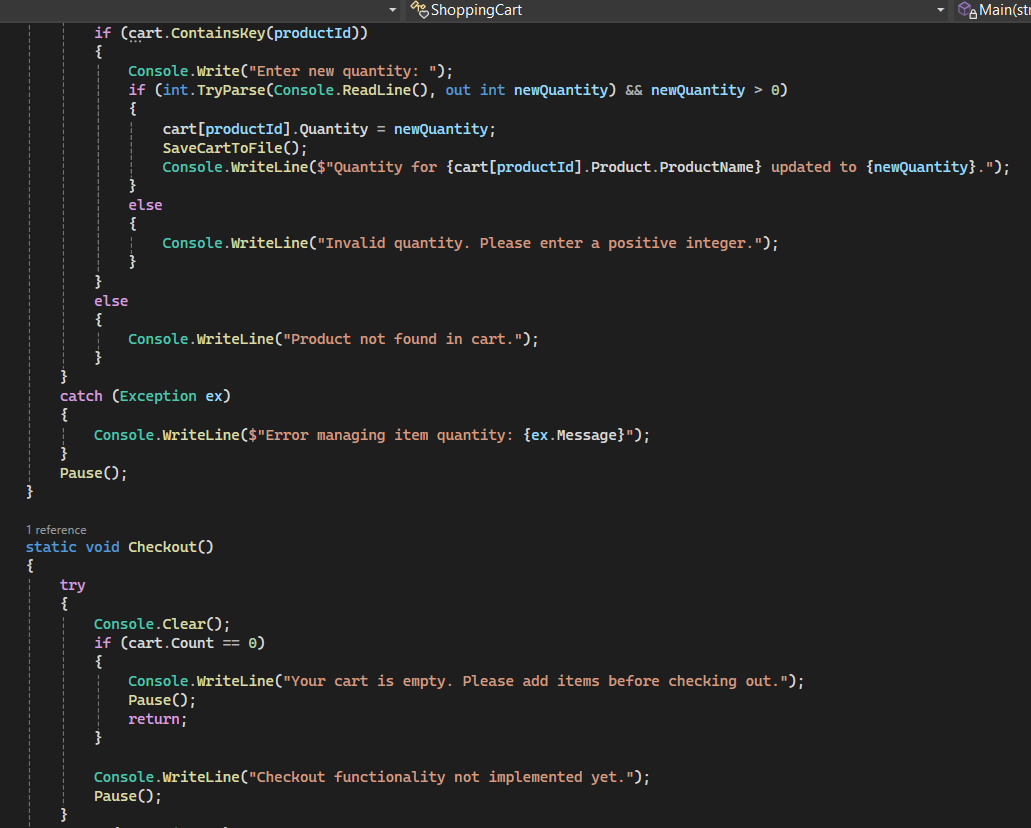
****

****

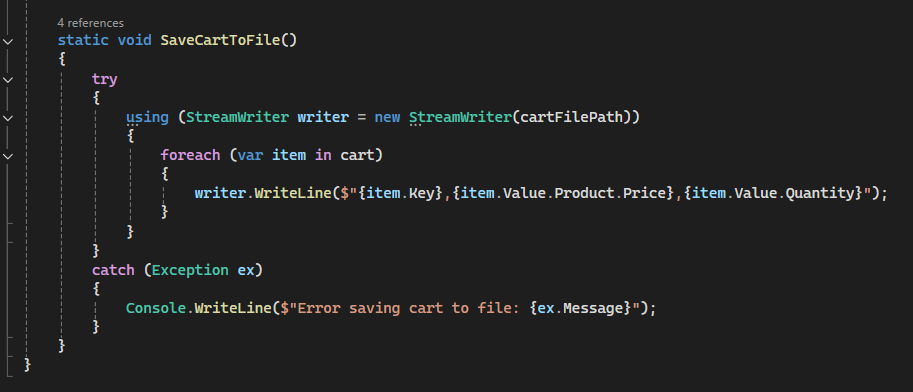
****

****

****

****

****

****

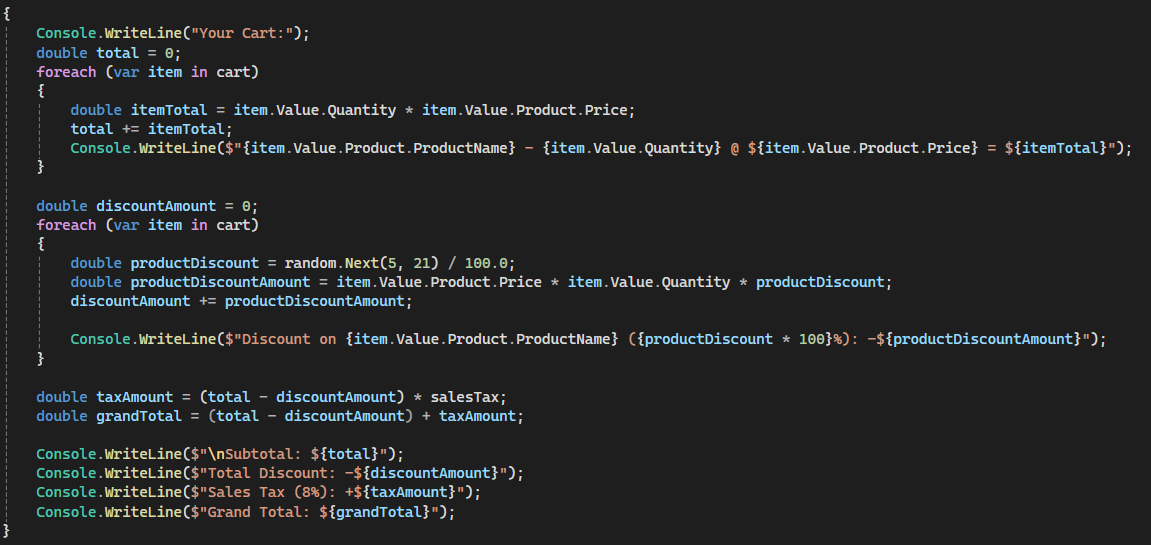
**4. User Interaction Flow**

1. **Start the Application:**
   * The application begins by displaying the main menu, where users can add products, remove products, view the cart, manage item quantities, or proceed to checkout.
2. **Adding Products:**
   * Users select a product from the displayed list of products, specify the quantity, and the system adds the item to the cart.
3. **Managing Cart:**
   * Users can view the items in the cart, modify the quantity, or remove items.
   * The system provides a real-time display of the subtotal, discount, sales tax, and grand total.
4. **Cart Expiration:**
   * The system checks if the cart has expired after each action. If it has, all items are removed, and the cart creation time is reset.
5. **Checkout:**
   * Upon checking out, users confirm the total and complete their transaction, which clears the cart.

**5. Financial Calculation**

The system performs financial calculations dynamically based on the current items in the cart:

* **Subtotal:** The sum of the product of quantity and price for all items in the cart.
* **Discount:** Calculated as 10% of the subtotal.
* **Sales Tax:** Calculated as 8% of the subtotal minus the discount.
* **Grand Total:** The final amount payable after applying the discount and tax.

****

**6. Conclusion**

The Shopping Cart System successfully demonstrates key object-oriented programming concepts in C#, such as encapsulation, modularization, and the use of static and instance members. The system is designed to be extendable, allowing additional features like payment processing or more sophisticated product management.

***END***