The Online Shopping Cart System

Eric Vara

The University of Arizona Global Campus

CST 301: Programming in C++

Professor Amjad Alkilani

November 07, 2023

**The Online Shopping Cart System**

In the Online Shopping Cart System developed in C++ using OOP principles, several classes are central to its functionality, encapsulating data and operations specific to the system's requirements. Each class represents a different aspect of the shopping experience:

**ItemToPurchase Class:** This class represents items that a customer can purchase. It includes attributes such as itemName, itemPrice, itemQuantity, and discountPercentage to store the item's details. Methods like GetName(), GetPrice(), and SetDiscountPercentage() allow for manipulating these attributes, ensuring that each item's information is maintained accurately and can be modified as needed.

**Inventory Class:** The Inventory class serves as a collection of ItemToPurchase objects, effectively simulating a store's inventory. It provides methods to addItem and deductItem, enabling the addition and removal of items from the inventory. It also includes a method to retrieve a specific item, which is crucial for processing purchases and applying discounts.

**ShoppingCart Class:** This class represents a user's shopping cart and contains a list of ItemToPurchase objects added by the user. It includes a method additem to include new purchases, and applyCoupon to apply discounts using a Coupon object. Additionally, calculateTotal() computes the cart's total cost, considering any discounts applied, and displayCart() outputs the cart's contents to the user.

**Coupon Class:** The Coupon class encapsulates the concept of a coupon, which can be applied to items in the shopping cart for discounts. It holds information like the code for coupon identification and discountPercentage to determine the discount's value. Methods like GetCode() and GetDiscountPercentage() allow for the retrieval of this information when applying the coupon to the shopping cart.

Together, these classes create a system that mimics an online shopping environment, where items can be browsed, added to a cart, and purchased with applicable discounts. The structure of the program ensures each class has a single responsibility and interacts with the others in a coherent manner, which is in line with OOP best practices. This modularity also facilitates future expansions, such as adding new features or integrating with other systems like payment gateways or inventory management systems.

**The Testing Processes**

The development process of this system included rigorous testing to ensure reliability and robustness. Boundary tests were implemented to check the system's responses to extreme input values, such as the highest and lowest possible prices, quantities, and diverse item names. Unit tests focused on individual class methods, ensuring that each function, like AddItem or SetDiscountPercentage, performed correctly across a range of inputs. Integration tests then verified the seamless interaction between these classes, confirming that, for example, items added to the inventory were accurately reflected in the shopping cart.

**Importance of C++ Programming**

C++'s versatility in handling low-level operations and high-performance computations makes it an ideal choice for diverse fields such as game engines in game development, complex algorithms in finance, and large-scale applications in software development. In scientific computing, its speed and efficiency facilitate extensive data analysis and simulations. Best practices in C++ emphasize the importance of clean, readable, and maintainable code. Variables should have descriptive names, and comments should elucidate the purpose of code blocks. Adherence to coding standards, such as those set by the ISO C++ Committee, and the use of tools like Git for version control are pivotal for collaboration and maintaining code integrity.

**Importance of C++ Programming**

C++ programming has a wide range of industrial and application uses, including game development, scientific computing, software development, and finance, among others. For example, C++ is frequently used in game development to create game engines and manage low-level programming tasks. In software development, C++ is utilized to develop high-performance applications and operating systems. C++ is also used in scientific computing for numerical simulations and data analysis tools, and in finance for trading algorithms and risk management systems.

**Conclusion**

The Online Shopping Cart System serves as a practical demonstration of C++'s capabilities in creating an interactive and user-friendly application. By leveraging OOP principles, the program achieves modularity and scalability, which are crucial for today's dynamic software development landscape. The project also highlights the need for comprehensive testing to build a reliable system. C++ remains a powerful tool in the programmer's arsenal, able to meet the demands of various industries while fostering best coding practices and keeping pace with technological advancements.

**Program Directory**

**A screenshot of a computer

Description automatically generated**

**Main Function**

**A screen shot of a computer program

Description automatically generated**

A screenshot of a computer screen

Description automatically generated

**ItemToPurchase.h**

**A screen shot of a computer program

Description automatically generated**

**ItemToPurchase.cpp**

**A screen shot of a computer program

Description automatically generated**

**Inventory.h**

**A screen shot of a computer program

Description automatically generated**

**Inventory.cpp**

**A screen shot of a computer program

Description automatically generated**

**ShoppingCart.h**

**A screen shot of a computer program

Description automatically generated**

**ShoppingCart.cpp**

**A screen shot of a computer program

Description automatically generated**

**Coupon.h**

**A screen shot of a computer program

Description automatically generated**

**Coupon.cpp**

**A screen shot of a computer code

Description automatically generated**

**Output**

**A screen shot of a computer

Description automatically generated**

**Work Cited**

Kolodiy, S. (2015, June 18). Unit testing and coding: Best Practices for Unit tests: Toptal. Toptal Engineering Blog. Retrieved March 18, 2023, from https://www.toptal.com/qa/how-to-write-testable-code-and-why-it-matters#:~:text=A%20typical%20unit%20test%20contains,it%20observes%20the%20resulting%20behavior.