VNUHCM-UNIVERSITY OF SCIENCE

FACULTY OF INFORMATION TECHNOLOGY

CSC10003 – OBJECT-ORIENTED PROGRAMMING

OOP-Lab 5: Assignment 04

Lecturer
Mr. Nguyễn Lê Hoàng Dũng
Mr. Hồ Tuấn Thanh

Class
23CLC08

Students
23127255 - Nguyễn Thọ Tài

November 12nd, 2024

Contents

1 Introduction

This report explains a simple console application in C++ to simulate the following shopping cart in Shopee as the image bellow.
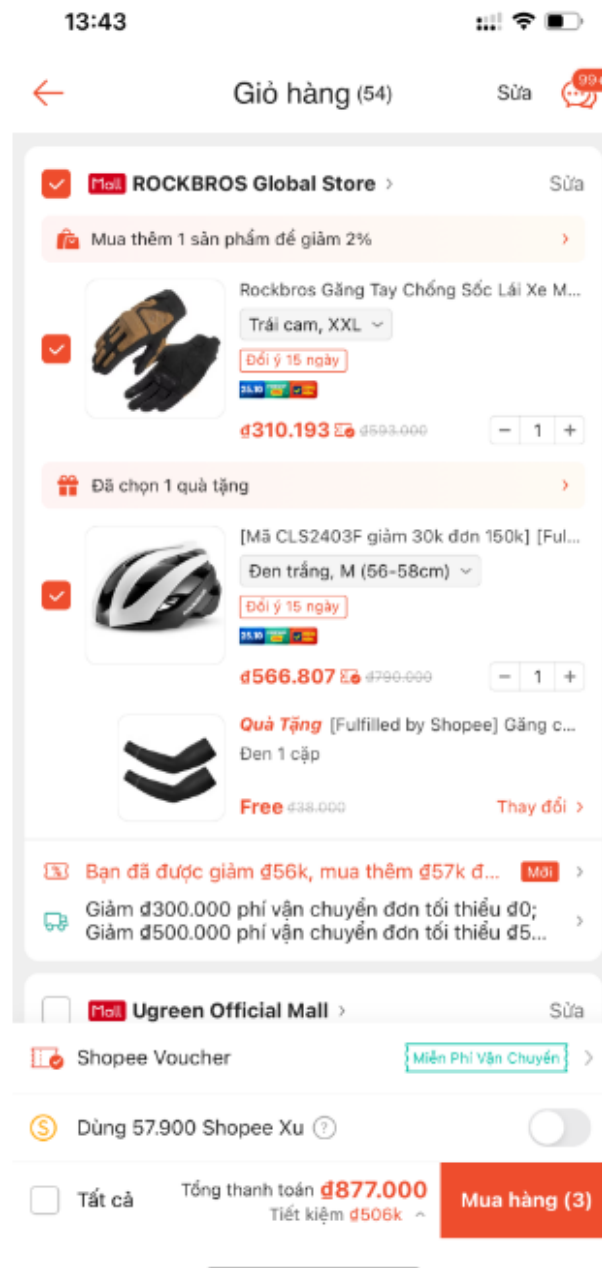

Figure 1: Shoppe's shopping cart

2 The application

2.1 Project structure

The project contains 5 files, with 3 type of class.

- Product (name, size, shop, price)

- Slot (product, quantity)

- Voucher (type, data)

- Inventory (slots, vouchers)

2.1.1 Product class

The Product class contain information about the an item's name, size, shop, and its price as presented here as an UML.

```
                              <<concrete>>
                               Product
─────────────────────────────────────────────────────────────────────
- name: string
- size: string
- shop: string
- price: double
─────────────────────────────────────────────────────────────────────
+ <<constructor>> Product()
+ <<constructor>> Product(name: const string &, shop: const string &, price: double)
+ <<constructor>> Product(name: const string &, size: const string &, shop: const string &, price: double)
+ <<constructor>> Product(other: const Product &)
+ getName(): string
+ getSize(): string
+ getShop(): string
+ getPrice(): double
+ setPrice(price: double): Product &
```

Figure 2: Product UML

Its role mainly is saving the information with no specical functions.

2.1.2 Slot class

Similar to the Product class, its ownly role is to save informations. It likes a bundle which contain the products and how much is the products.
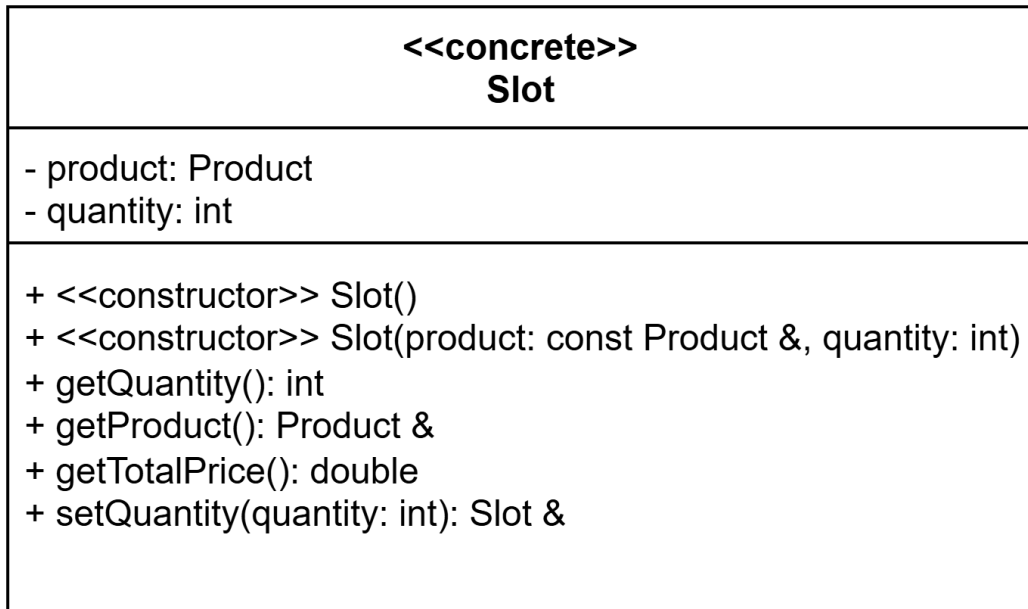
```
┌─────────────────────────────────────────────────────────┐
│                     <<concrete>>                         │
│                        Slot                              │
├─────────────────────────────────────────────────────────┤
│ - product: Product                                       │
│ - quantity: int                                          │
├─────────────────────────────────────────────────────────┤
│ + <<constructor>> Slot()                                 │
│ + <<constructor>> Slot(product: const Product &, quantity: int) │
│ + getQuantity(): int                                     │
│ + getProduct(): Product &                                │
│ + getTotalPrice(): double                                │
│ + setQuantity(quantity: int): Slot &                     │
│                                                          │
└─────────────────────────────────────────────────────────┘
```

Figure 3: Slot UML

2.1.3 Voucher class

```
┌───────────────────────────────────────────────────────────┐
│                       <<concrete>>                         │
│                         Voucher                            │
├───────────────────────────────────────────────────────────┤
│ - type: int                                                │
│ - data: double                                             │
├───────────────────────────────────────────────────────────┤
│ + <<static>> TYPE_PERCENT: const int                       │
│ + <<static>> TYPE_DISCOUNT: const int                      │
│ + <<constructor>> Voucher()                                │
│ + <<constructor>> Voucher(data: double, type: int = TYPE_PERCENT) │
│ + getVoucherType(): int                                    │
│ + getData(): double                                        │
└───────────────────────────────────────────────────────────┘
```
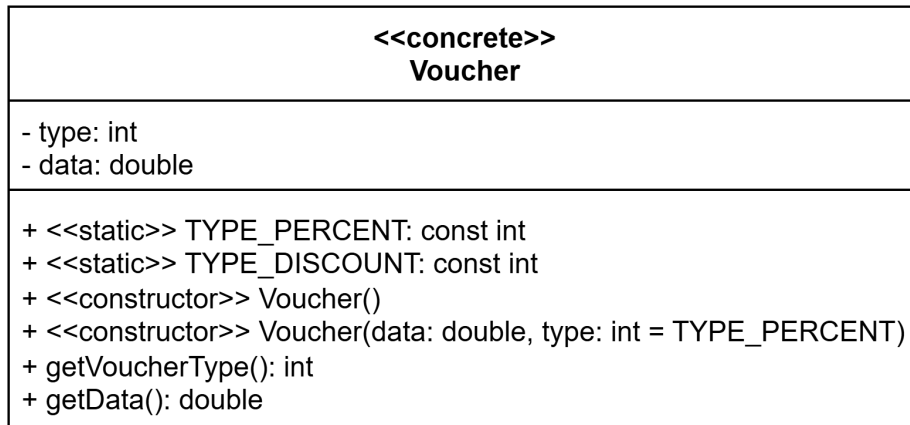
Figure 4: Voucher UML

2.1.4 Inventory class

The inventory is on other-hand, not only saving all the Slots but also do the calculation and output. It also validate the slot and manage the products and slots inside it.

In this week, the inventory also capable of saving the order to file, loading products from files and having vouchers implemented!

Its design is a singleton, which mean there is only one of it for all instances. This would help easier to manage the inventory on everywhere without the need of having to save a reference to the inventory.
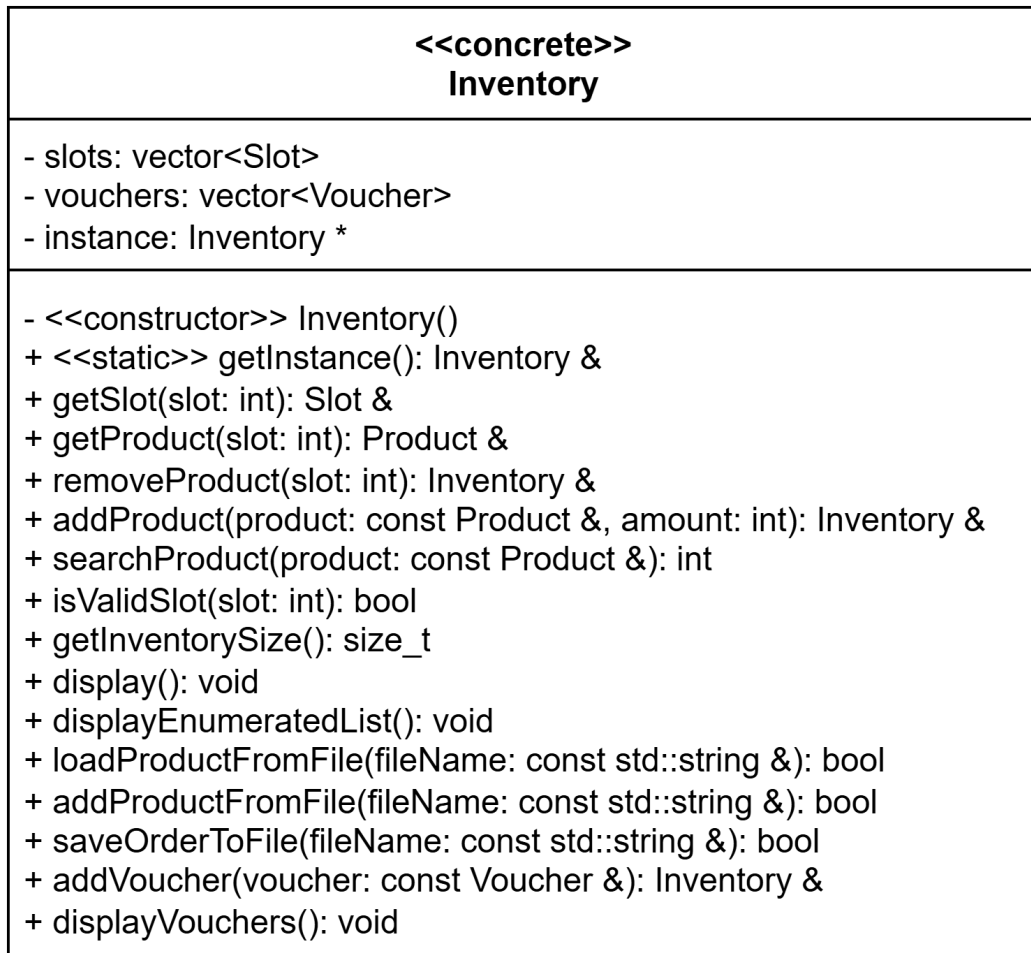
<<concrete>> **Inventory**
- slots: vector<Slot> - vouchers: vector<Voucher> - instance: Inventory *
- <<constructor>> Inventory() + <<static>> getInstance(): Inventory & + getSlot(slot: int): Slot & + getProduct(slot: int): Product & + removeProduct(slot: int): Inventory & + addProduct(product: const Product &, amount: int): Inventory & + searchProduct(product: const Product &): int + isValidSlot(slot: int): bool + getInventorySize(): size_t + display(): void + displayEnumeratedList(): void + loadProductFromFile(fileName: const std::string &): bool + addProductFromFile(fileName: const std::string &): bool + saveOrderToFile(fileName: const std::string &): bool + addVoucher(voucher: const Voucher &): Inventory & + displayVouchers(): void

Figure 5: Inventory UML

Methods detail:

- getInstance(): return the instance of Inventory using Singleton pattern design.

- getSlot(): return a reference to the slot inside the inventory

5

- getProduct(): return a reference to the product at a specific slot inside the inventory.

- addProduct(): add a product to the inventory

- removeProduct(): remove the product at slot position

- searchProduct(): search for the product if it exist in the inv. If yes, then return the position of the product in the inventory. If not return -1.

- display(): display the shopping cart with total price calculated. If any vouchers exist, it will also calculating the discount and display the final price of the orders.

- displayEnumeratedList(): display the inventory as a list for minimal views and without the price calculating.

- loadProductFromFile(): load products from files, delete all existing products

- addProductFromFile(): add products from files, don't delete existed products

- saveOrderToFile(): save orders to file

- addVouchers(): add vouchers

- displayVouchers(): display all vouchers in list

2.2 Project runtime

2.2.1 Main menu

```
Menu:
1. Add product
2. Remove product by search
3. Remove product by slot's ID
4. Display Inventory/Price
5. Display Inventory/Price (List)
6. Change slot quantity
7. Load products from file (delete existed products)
8. Add products from file
9. Save orders to file
10. Add promotions (vouchers)
11. Display promotions (vouchers)
12. Exit
0. Clear screen
Enter your choice:
```

Figure 6: Main menu

2.2.2 Add product

Figure 7: Add product

When you re-add a same product, it will add up the quantity but not add the new product as its separated slot.

2.2.3 Remove product using infomation

User will input product's name, size, shop to find the product in the inventory to remove the product.

2.2.4 Remove product using IDx

Figure 8: Remove product using Idx

2.2.5 Display inventory and prices

If any vouchers is in the inventory, it will auto-select the best voucher to apply for the discount!

```
D:\clion\OOP\csc10003-lab\2312725
Enter your choice: 4
SHOP: LocalBranch
Name:        Shirt
Size:        M
Unit Price: 250.23
Quantity:    3
Price:        750.69

Name:        Bra
Size:        M
Unit Price: 41.21
Quantity:    3
Price:        123.63

Name:        Shirt
Size:        L
Unit Price: 257.23
Quantity:    3
Price:        771.69

        -=-
SHOP: MALL LiTU
Name:        Dress
Size:        L
Unit Price: 311.21
Quantity:    3
Price:        933.63

        -=-
SHOP: Trung Nguyen
Name:        Coffee
Size:        10 packs
Unit Price: 111.21
Quantity:    3
Price:        333.63

        -=-

--------------------
TOTAL PRICE: 2913.27
Vouchers type: percent
Voucher percentage: 2
Voucher discount: 58.2654
FINAL PRICE: 2855
```

8

Figure 9: View inventory, total prices and final prices

2.2.6 Display list of products

Figure 10: List of products

2.2.7 Change slot quantity

You can change a slot quantity

2.2.8 Load products from files

You can load products from files, as well add products from files using choice 7 and 8 in the menu!

Figure 11: Load products from files

2.2.9 Save order to file

You can save order to files by using choice 9 in the menu!

Figure 12: Save orders to file

2.2.10 Add vouchers

You can add vouchers to the inventory, when displaying the price... the best voucher will be auto-selected for the best discount!

You can add vouchers by using choice 10. There are two type of vouchers, discount vouchers and percentage vouchers.

Discount voucher do discount the same amount of moneys it stated, but the percentage vouchers do discount the percentage amount of total prices.

Figure 13: Add vouchers

2.2.11 Display vouchers

Figure 14: Voucher display

11