

**STORE ORDER MANAGEMENT SYSTEM**

**Course**: **Programming 1 - COSC2081**

*Lecturer:* ***Vu Thanh Minh***

*Group 4*

*Members: Nguyen Thi Ha Giang - S3914108*

*Nguyen Tuan Thang - S3877039*

*Trinh Viet Quy - S3915202*

*Tran Duy Phong - S3879069*

*Sep 4th, 2022*

**Table of Contents**

[List of Figures 1](#_Toc114433496)

[I. Introduction 2](#_Toc114433497)

[II. Project Description 4](#_Toc114433498)

[III. Implementations Details 5](#_Toc114433499)

[IV. Project Planning Report 23](#_Toc114433500)

[V. Conclusion 27](#_Toc114433501)

# **List of Figures**

[Figure 1. Class diagram of Store Management System 5](file:///C:\Users\Giang\Desktop\Programming1\Programming1\Group4_report.docx#_Toc114433445)

[Figure 2. All CSV files to store database 7](#_Toc114433446)

[Figure 3. Admin class object 8](#_Toc114433447)

[Figure 4. Member class object 8](file:///C:\Users\Giang\Desktop\Programming1\Programming1\Group4_report.docx#_Toc114433448)

[Figure 5. Category class object 9](file:///C:\Users\Giang\Desktop\Programming1\Programming1\Group4_report.docx#_Toc114433449)

[Figure 6. Product class object 10](#_Toc114433450)

[Figure 7. Cart class object 11](file:///C:\Users\Giang\Desktop\Programming1\Programming1\Group4_report.docx#_Toc114433451)

[Figure 8. Order class object 11](file:///C:\Users\Giang\Desktop\Programming1\Programming1\Group4_report.docx#_Toc114433452)

[Figure 9. Discount abstract class 12](#_Toc114433453)

[Figure 10. SilverDiscount extends class Discount 12](#_Toc114433454)

[Figure 11. GoldDiscount extends class Discount 12](#_Toc114433455)

[Figure 12. Platinum extends class Discount 13](#_Toc114433456)

[Figure 13. Member home page (GUI) 16](#_Toc114433457)

[Figure 14. User register form (GUI) 16](#_Toc114433458)

[Figure 15. User login page (GUI) 17](#_Toc114433459)

[Figure 16. Add product to cart (GUI) 17](#_Toc114433460)

[Figure 17. Cart (user) (GUI) 18](#_Toc114433461)

[Figure 18. Order invoice (GUI) 18](#_Toc114433462)

[Figure 19. Account details (GUI) 19](#_Toc114433463)

[Figure 20. Orders history (GUI) 19](#_Toc114433464)

[Figure 21. Admin login page (GUI) 20](#_Toc114433465)

[Figure 22. Admin home page (List of products is default list) (GUI) 20](#_Toc114433466)

[Figure 23. Add new product form (GUI) 21](#_Toc114433467)

[Figure 24. Add new category form (GUI) 21](#_Toc114433468)

[Figure 25. Update product price form (GUI) 22](#_Toc114433469)

1. **Introduction**

**Background:**

In this project, we designed a store management system, building basic features for both users and admins. This system will make it easy for users to purchase and manage orders, besides, the system will also help admins easily manage all information of products, users, as well as other information of the orders.

Our team consists of 4 people, and each person takes on different roles:

1. Nguyen Thi Ha Giang - *S3914108*: Back-end for user and UI developer.
2. Tran Duy Phong - *S3879069* : Back-end for admin and UI developer.
3. Trinh Viet Quy - *S3915202* : Back-end for admin.
4. Nguyen Tuan Thang - *S3877039* : Back-end for user.

**Scope and objective:**

* Goal: We made this project based on the knowledge provided in the course on java programming. Besides, we will also research external documents to get the most complete application possible. Through this project, we hope to achieve a complete product that meets the basic features of a store management system. Besides, acquiring new knowledge and applying them during project implementation is also one of the important purposes when we started this project.
* Scope:
  + Back-end and database:
    - User:

Our system includes the most basic features needed in a store management system to meet the purchasing needs of users. Users can easily view all products in our system, and customers can register as members to receive benefits from the system. After registering for an account, customers will be able to purchase products, manage order history as well as receive incentives from the store system.

* + - Admin:

As an admin, the system will allow the admin to manage all lists of products, customers as well as orders. Admin has the ability to add products, edit product prices as well as delete products. In addition, the admin can change the status of the order and from there, the system will update the order status for the customer.

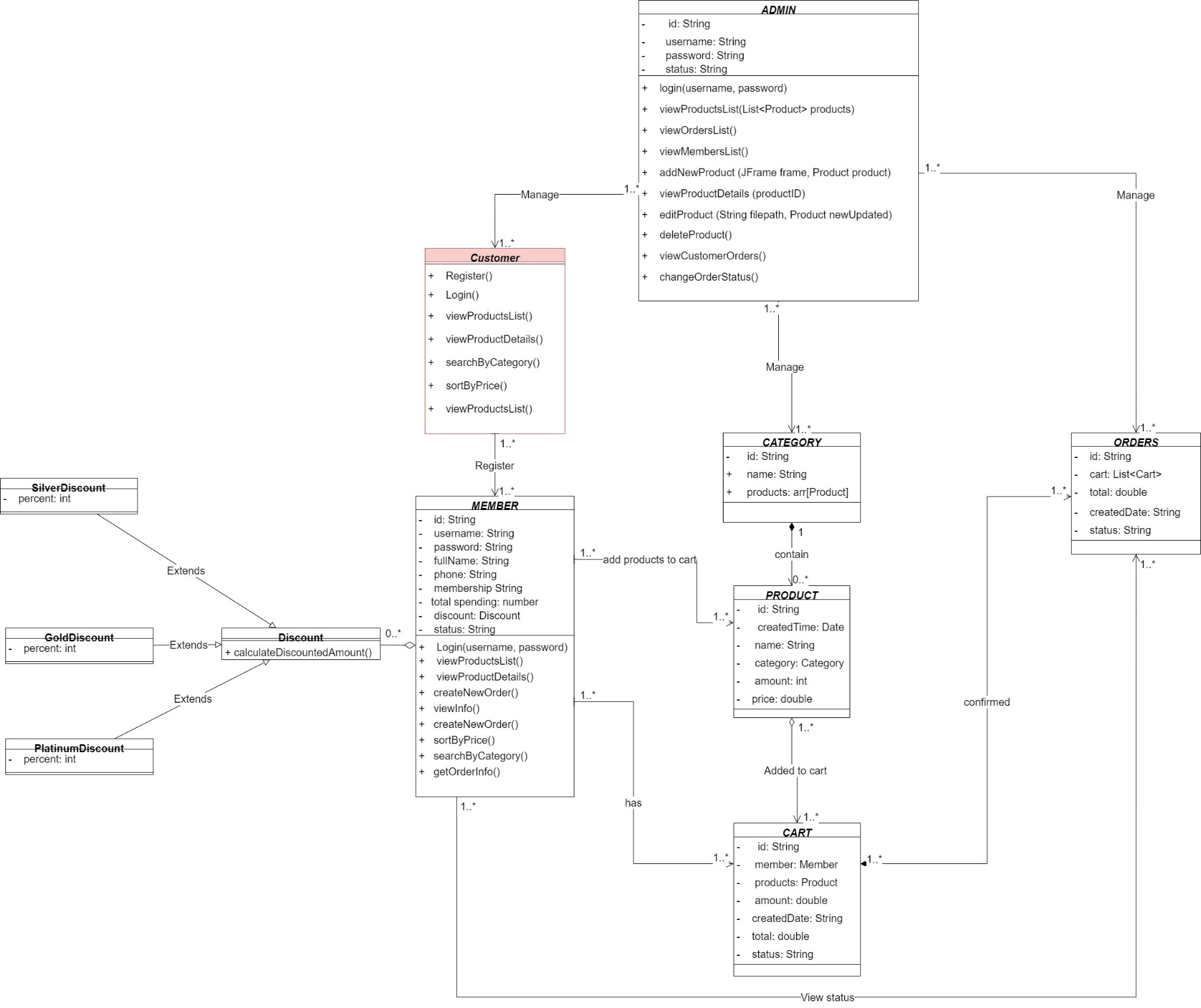
* + GUI:

As for the interface, our system will provide users with the simplest possible interface, which will be easy for users to use and interact with the system. Besides, we have explicitly and tightly linked back-end data and UI, so the information customers make on the interface will always be updated simultaneously in the database.

1. **Project Description**

* First, to design all classes of the system, we decide to sketch a class diagram to manage all class and functions of our application. In class diagram,
* We have used JDK 16 Oracle to run our application. Besides, we have used 4 main library in java to implements our code, including: java.util, java.io, java.text and especially java.swing to design GUI.
  + Java.util: functions that use ArrayList, Scanner, Date, … and so on
  + Java.io: functions that are related to input and output through file system
  + Java.text: this package helps us to format string appearance of messages, numbers, times and dates.
  + Java.swing: this package is used for designing our system UI and provides some basic components for the interface such as Table, Buttons, or Text Box and some other advanced components like JSlider.

1. **Implementations Details**
2. **Class diagram**

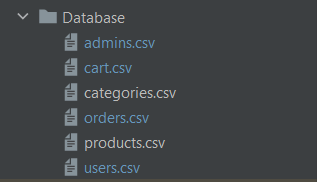
* First of all, to build a complete application, we sketched a class diagram to understand all the components needed to design a store management system. Below is a diagram image of our system:

**Figure 1. Class diagram of Store Management System**

* Base on the class diagram, we can see clearly that in the whole system has 7 main objective classes and 1 abstract class with 3 sub-class extend it.
* Admin class: Admin has a role that management all of system. An admin has 4 attributes: id, username, password, and status. The id and username are unique and the status will be used to show when the admin log in/ log out. Besides, you cannot create admin account, just us (app’s developer/ admins) can import a new admin!
* Customer class: This class is represented for guest. We cannot collect their information. They can view all list of products in system, sort by category and price, but they cannot buy it. Customer need to register to be a member, then they can add product to cart and create new order.
* Member class: After the guest registers successfully, all of their information will be stored in the system, and they will be a member object. As a member object, they have 9 attributes: id, username, password, email, password, totalSpending, membership, discount, status. In that, id and username is unique. We will base all their totalSpending to change their membership automatically. totalSpending is calculated by the sum of total of all their orders. And when the membership changed, the discount, implement class Discount, will be apply into this Member. And the status will be show when member log in/ log out.
* Category class: In this object, it has 2 attributes: name and List of products. name is a String that represented for title of this category. And in each Category object will contain 1 List of Products.
* Product class: A product object has 6 attributes: id, name, category, price, amount, createTime. In that, id and name are unique. category is a object Category that contains this product. An amount is used to calculate the amount of product that a member adds to cart. createTime is the time and date that the admin add product into the system.
* Cart: A cart object has 7 attributes: id, member’s name, product, amount, createDate, total, status. In that, id is unique and automatically generated by the system. Member’s name is the name of a member who add product in to cart (this cart will be owned by this member). Product is the product object that member add to cart. Total is calculated by price of product multiple by amount. The default status is “unpaid” and will be changed to “paid” when member confirm to pay it.
* Order: A order object has 5 attributes: id, List of carts, total, createDate, status. In that, id is unique and automatically generated by the system. List of carts is all of products that member confirm to pay. Total is the sum of total of all carts in list of carts. createDate is the date that member confirm this order. A status will be displayed “confirmed” when member created a new order. And it can be changed manually by an admin.
* Discount: this class contains the function is calculateDiscountAmount. If the sub-class is SilverDiscount, it will discount 5%. If the sub-class is GoldDiscount, it will discount 10%. If the sub-class is PlatinumDiscount, it will discount 15%.

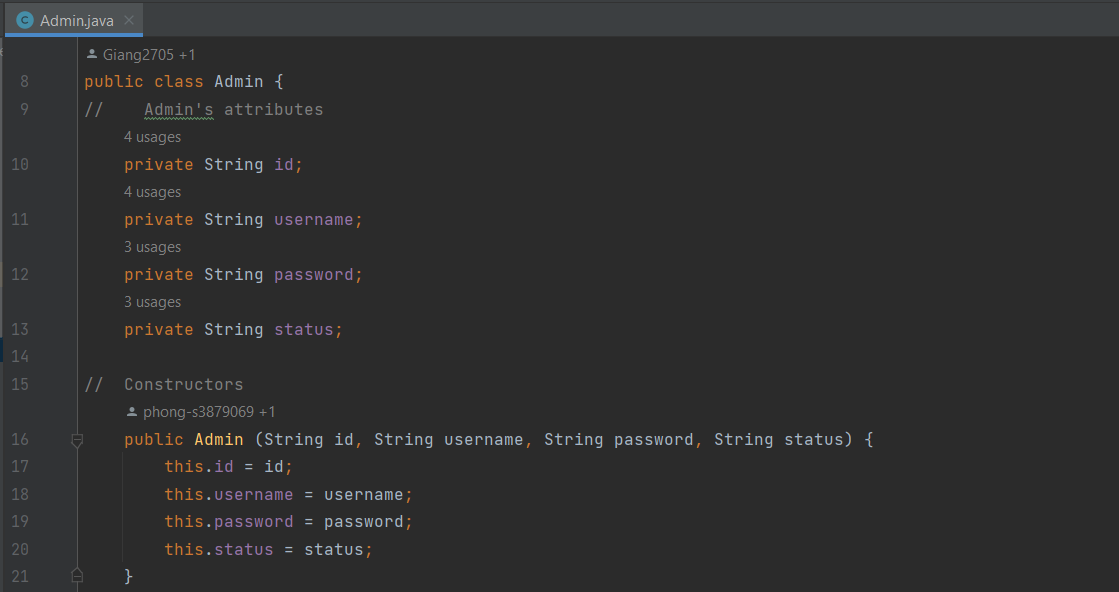
1. **Code details**
2. **Back-end and database**
   * + Database:

* Regarding data storage and management, we use CSV files for this. The data when stored in the CSV file will be clearly divided between attributes by commas ( , ) , so we can easily manage and retrieve data from the database.
* In our entire system, there are a total of 6 CSV files to store data of each different object, including admins, users, categories, products, carts, and orders.

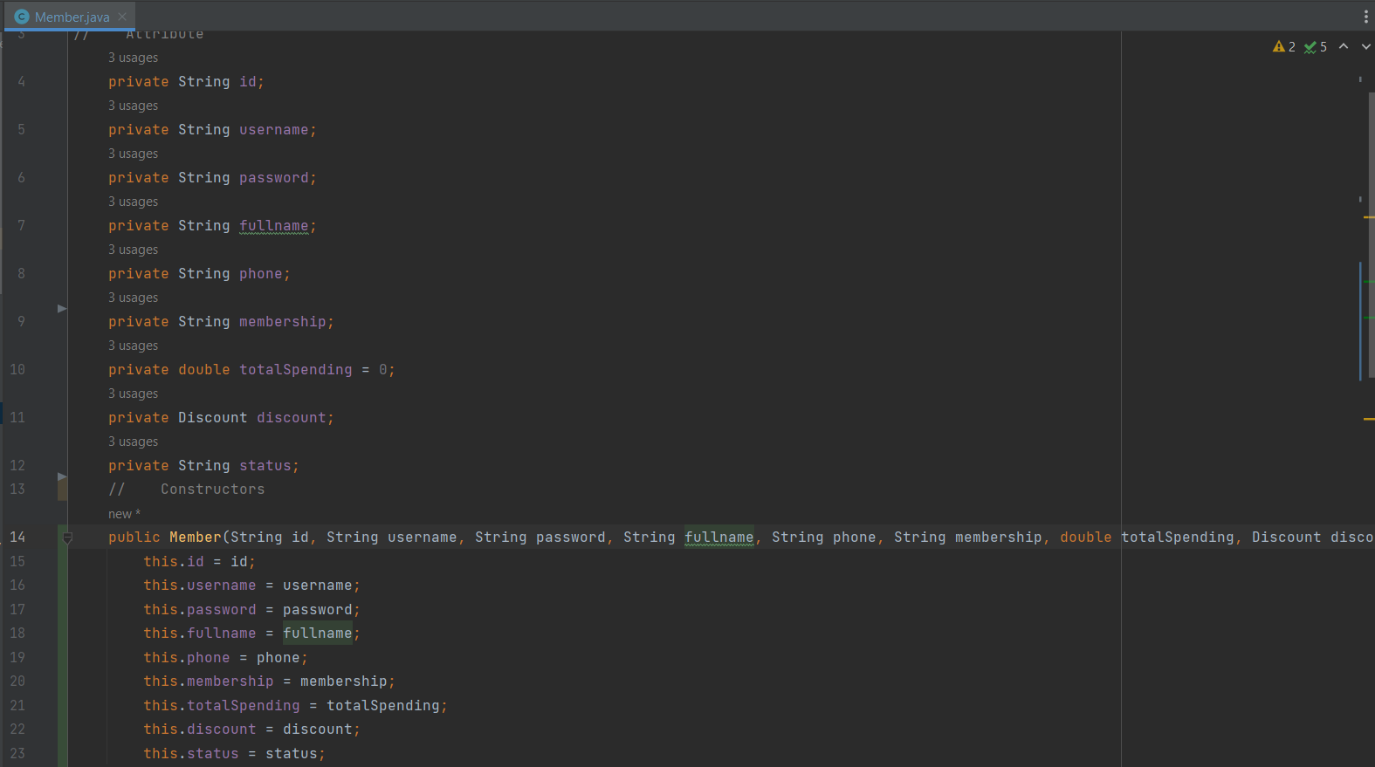


**Figure 2. All CSV files to store database**

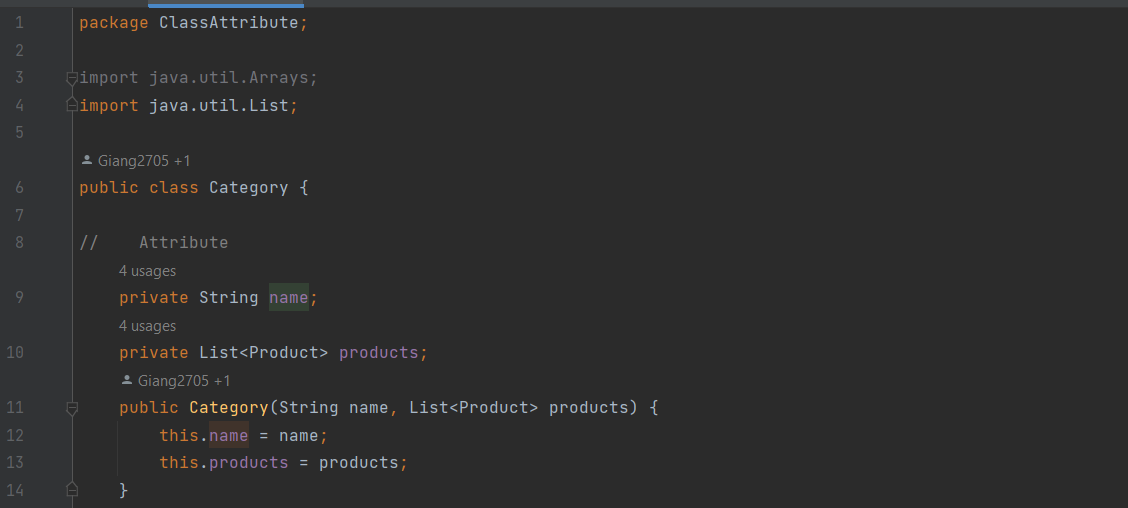
* + - Classes: We have built the whole system including 6 main object classes: Admin, Member, Category, Product, Cart, and Order. In addition, there is an abstract class that is Discount with 3 inherited subclasses, SilverDiscount, GoldDiscount and PlatinumDiscount.
* We will put all of figures that represent for each class in our system and the explanation about some main functions will be given below:
  + Class Admin:



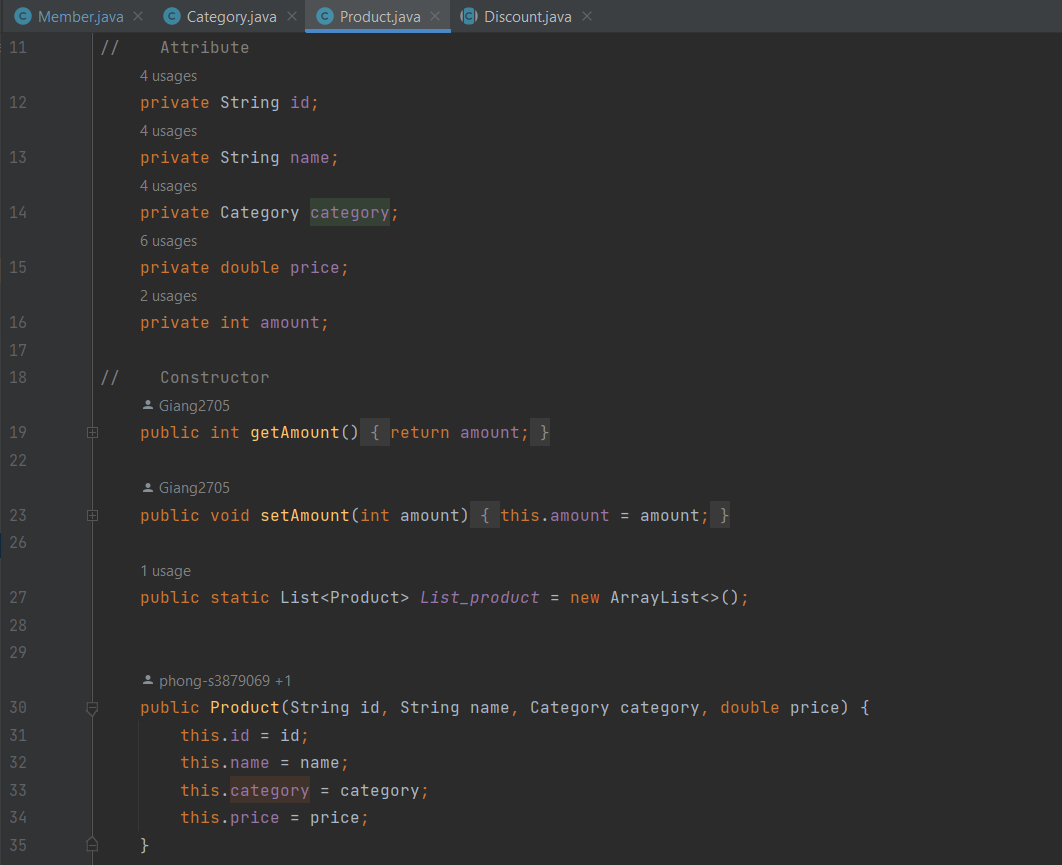
**Figure 3. Admin class object**

* + Class Member:

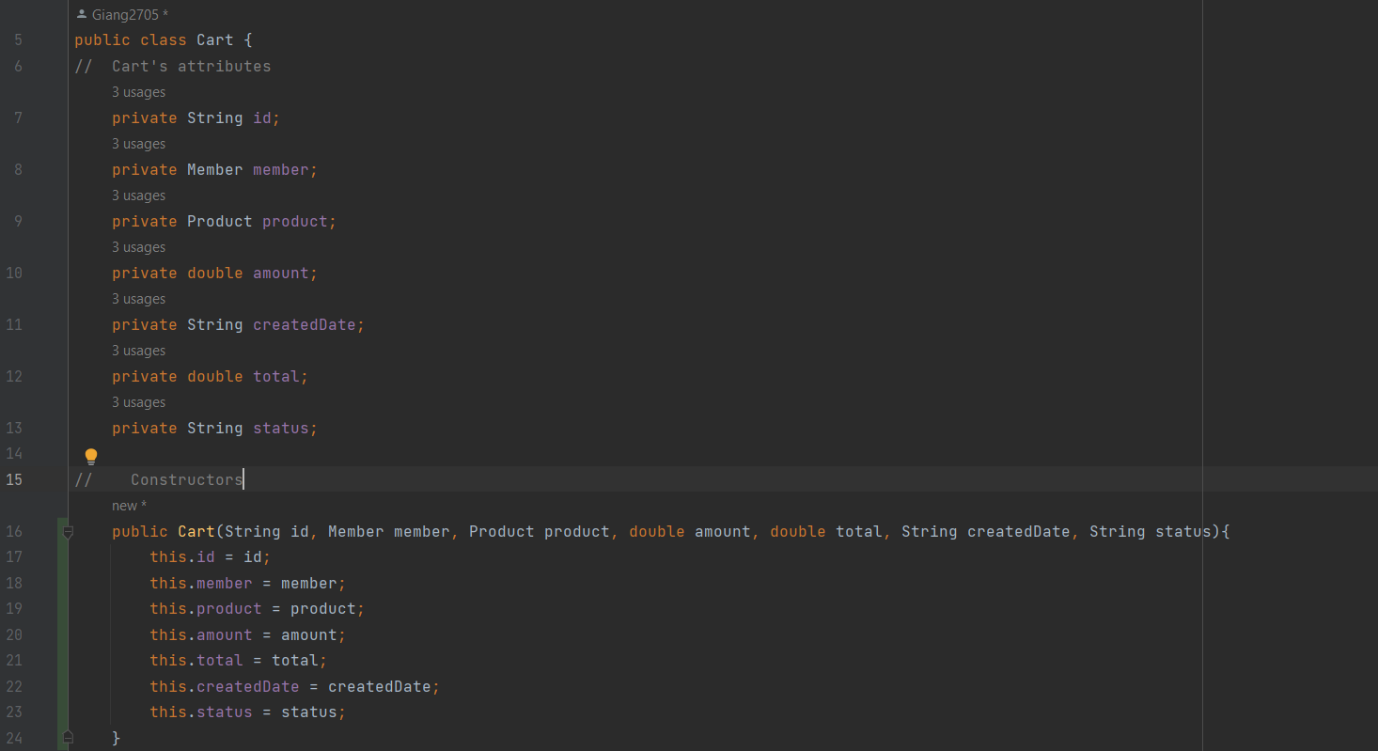
**Figure 4. Member class object**

* + Class Category:

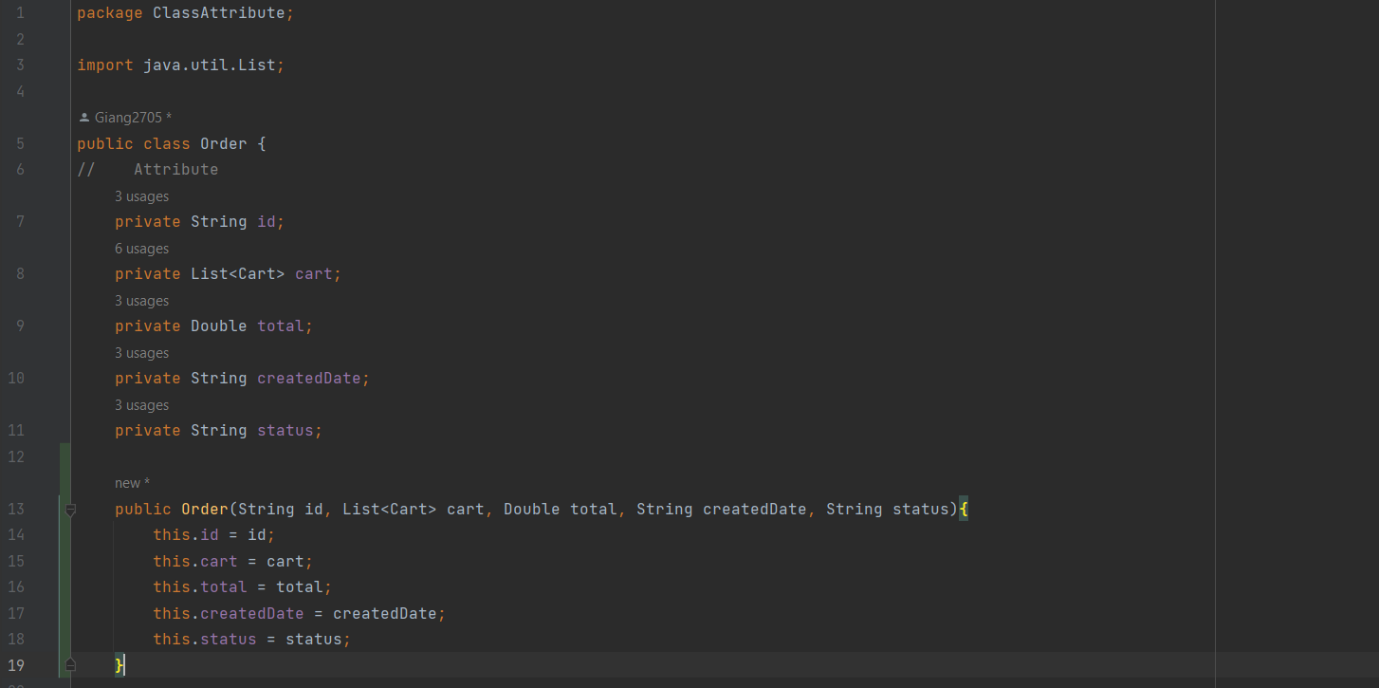
**Figure 5. Category class object**

* + Class Product:

**Figure 6. Product class object**

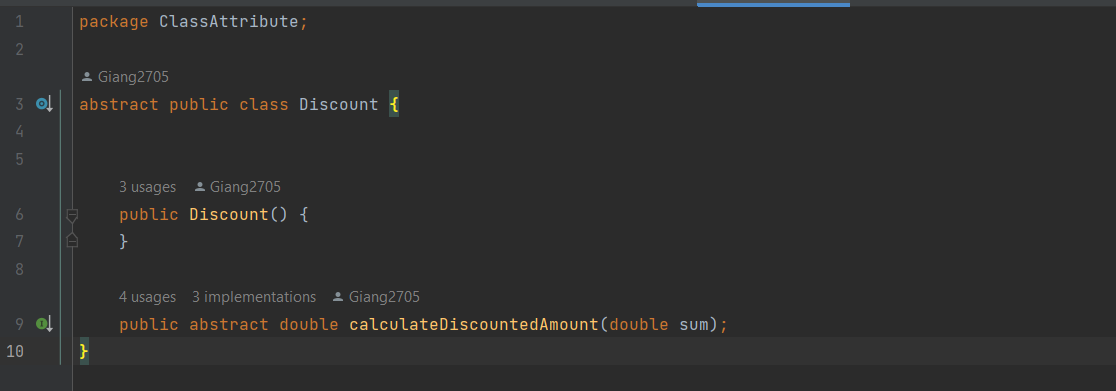
* + Class Cart:

**Figure 7. Cart class object**

* + Class Order:

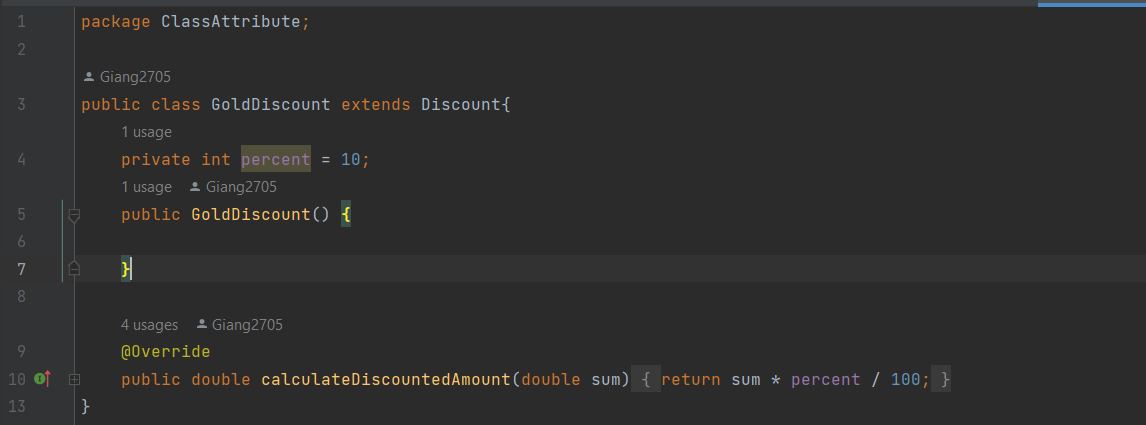
**Figure 8. Order class object**

* + Class Discount:



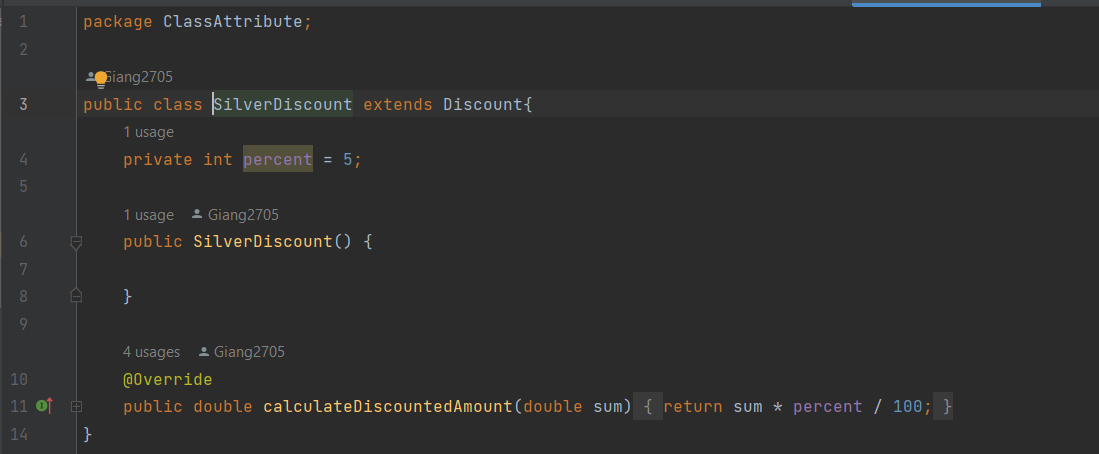
**Figure 9. Discount abstract class**

* + Class SilverDiscount:



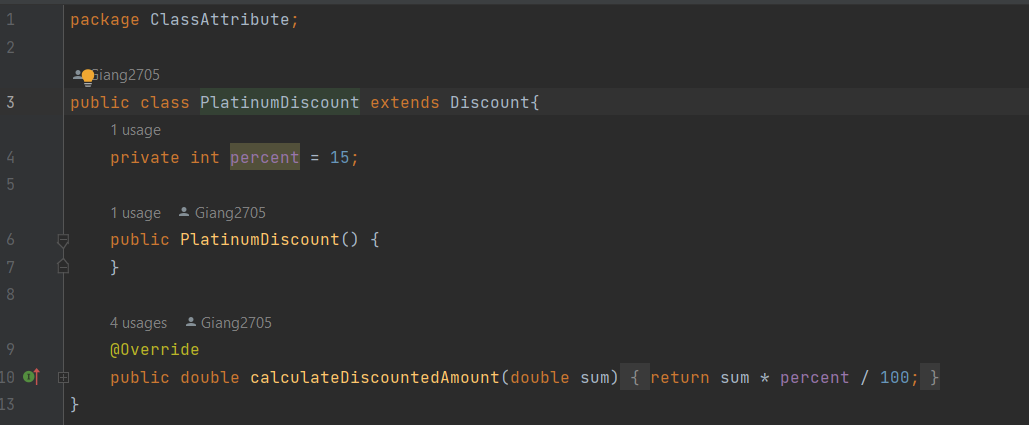
**Figure 10. SilverDiscount extends class Discount**

* + Class GoldDiscount:



**Figure 11. GoldDiscount extends class Discount**

* + Class PlatinumDiscount:



**Figure 12. Platinum extends class Discount**

* + - Functions:
* GenerateID:
  + In this function, we have use StringBuilder to generate automatically ID for Member, Product, and Order. This is a string include n elements (n is a integer number which input by developer, in our system, we choose n = 3). This string will be unique and includes random lower as well as upper character and number.
* GetDate:
  + To get date at the time that product or order created, we have used method Date and System.currentTimeMillis() to get exactly the time. Then we used SimpleDateFormat() to format for this date.
* storeDatabase:
  + This is one of the most important function in our system. We use methods in this class to create Database for our system. In the class, we have some sub-functions which are used to create new data file. It includes:
    - createFile: create users.csv to store member’s information;
    - createCategoryFile: create categories.csv to store category’s information;
    - createProductFile: create products.csv to store product’s information;
    - createCartsFile: create carts.csv to store cart’s information;
    - createOrdersFile: create orders.csv to store order’s information;
  + Besides, some functions to create new element also be implemented in this class, such as: register (for user), addNewCategory (admin add category), addNewProduct (admin add product), addNewOrder (user create new order), etc. .
* readDatabase:
  + This is also one of the most important function in the system. Functions in this class to fetch data from database into a List (ArrayList) to use for calling another functions. In this class have 6 sub-functions:
    - readUserFile: fetch users.csv into a List of all users
    - readAdminFile: fetch admins.csv into a List of all admins
    - readCategoryFile: fetch categories.csv into a List of all categories
    - readProductFile: fetch products.csv into a List of all products
    - readCartFile: fetch carts.csv into a List of all carts
    - readOrderFile: fetch orders.csv into a List of all orders
* Login:
  + This function depends on readDatabase function to take the information of user (by readUserFile and readAdminFile) in the database and compare the information with inputting information from the user.
* SearchOrder:
  + Based on the readDatabase function, this function uses readOrderFile sub-function to take order data which is ID from the database and compare with the input from user to output.
* SortByCategory:
  + Based on readProductFile of readDatabase, this function gets the name of the products and compare with what category is to output.
* SortByPrice:
  + This function takes the price of products from the order database by readProductFile of readDatabase and compare the price of all products. Then, sorting depends on the choice of the user.
* RemoveProductFromCart:
  + This function will be applied when member choose delete product in their cart. All of the product’s information will be removed from cart database and the cart database will be reloaded.
* ChangeOrderStatus:
  + When admin change status, the system will be call readOrderFile to access list of all orders. Then it will check if the id the admin chose equal the id in orders list, the system will take this order, and then this order’s status will be change to the string that admin input.

***These are SOME IMPORTANT functions in our systems, remaining functions have been implemented and explained directly in source code.***

1. **GUI**
   * + User:

* MemberHomePage:

**Graphical user interface, application

Description automatically generated**

**Figure 13. Member home page (GUI)**

* UserRegisterForm:

**Graphical user interface, application

Description automatically generated**

**Figure 14. User register form (GUI)**

* UserLoginPage:

**Graphical user interface, application

Description automatically generated**

**Figure 15. User login page (GUI)**

* modalAddToCart:

**Graphical user interface, application

Description automatically generated**

**Figure 16. Add product to cart (GUI)**

* CartScreen:

**Graphical user interface, text, application

Description automatically generated**

**Figure 17. Cart (user) (GUI)**

* ConfirmedOrderModal:

**Text

Description automatically generated with low confidence**

**Figure 18. Order invoice (GUI)**

* AccountPage:

**Graphical user interface, text, application, email

Description automatically generated**

**Figure 19. Account details (GUI)**

* OrderHistoryScreen:

**Table

Description automatically generated**

**Figure 20. Orders history (GUI)**

* + - Admin:
* AdminLoginPage:

**Graphical user interface, application

Description automatically generated**

**Figure 21. Admin login page (GUI)**

* AdminHomePage:

**Table

Description automatically generated with low confidence**

**Figure 22. Admin home page (List of products is default list) (GUI)**

* AddNewProductForm:

**Graphical user interface, application

Description automatically generated**

**Figure 23. Add new product form (GUI)**

* AddNewCategoryForm:

**Graphical user interface, text, application

Description automatically generated**

**Figure 24. Add new category form (GUI)**

* UpdateProductForm:

**Graphical user interface, application

Description automatically generated**

**Figure 25. Update product price form (GUI)**

1. **Project Planning Report**

* We have planned and managed our works on Trello Board as well as we created a Meeting Record to manage timeline. We will share our Trello link and GitHub repo in which the Meeting Record is pushed.
  + **GitHub repo:** <https://github.com/Giang2705/Programming1.git>
  + **Trello Board:** <https://trello.com/invite/b/SWmTXgRt/cc096ec7a981e674711da8d168e95c1b/programming-1>
* We will summarize our works through each week (from 25/07/20022 to 18/09/2022)
* **Week 1 (25/07/2022 – 01/08/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Design UX-UI for the app + Implement code for Welcome Screen | 80% |
| 1. Nguyen Tuan Thang | * Create class Admin and class Order | 80% |
| 1. Trinh Viet Quy | * Create class Product and Category | 80% |
| 1. Tran Duy Phong | * Create class Customer and Member | 80% |

* **Week 2 (01/08/2022 – 08/08/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Add products to cart (User) (database + GUI) | 100% |
| 1. Nguyen Tuan Thang | * Register + Login + Logout (User) (database + GUI) | 100% |
| 1. Trinh Viet Quy | * CRUD Products (Admin - GUI) | 100% |
| 1. Tran Duy Phong | * CRUD Products (Admin) | 100% |

* **Week 3 (08/08/2022 – 15/08/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Confirm (Create order) + delete product in cart (database + GUI) | 100% |
| 1. Nguyen Tuan Thang | * List all order history (User) (database + GUI table) | 100% |
| 1. Trinh Viet Quy | * Search order by member’s ID (Admin) | 100% |
| 1. Tran Duy Phong | * List all products, members, orders + design Admin Homepage (GUI + fix database) | 100% |

* **Week 4 (15/08/2022 – 22/08/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Sort products by price (User) (database + GUI) + Change order’s status (Admin) (database + GUI) | 100% |
| 1. Nguyen Tuan Thang | * Sort products by price + Search order by order’s ID (User) (database + GUI) | 100% |
| 1. Trinh Viet Quy | * Change membership (User) (database) | 100% |
| 1. Tran Duy Phong | * Sort products by category (user) (database + GUI) + Change log in/ log out status (database) | 100% |

* **Week 5 (22/08/2022 – 29/08/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Test app for the first time + fix code | 100% |
| 1. Nguyen Tuan Thang | * Test app for the first time + fix code | 100% |
| 1. Trinh Viet Quy | * Test app for the first time + fix code | 100% |
| 1. Tran Duy Phong | * Test app for the first time + fix code | 100% |

* **Week 6 (29/08/2022 – 04/09/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Test final app + fix for clean code | 100% |
| 1. Nguyen Tuan Thang | * Test final app + fix for clean code | 100% |
| 1. Trinh Viet Quy | * Test final app + fix for clean code | 100% |
| 1. Tran Duy Phong | * Test final app + fix for clean code | 100% |

* **Week 7 (04/09/2022 – 11/09/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | * Write report | 100% |
| 1. Nguyen Tuan Thang | * Fix class diagram + check all functions in code | 100% |
| 1. Trinh Viet Quy | * Fix class diagram + check all functions in code | 100% |
| 1. Tran Duy Phong | * Fix class diagram + check all functions in code | 100% |

* **Week 8 (11/09/2022 – 18/09/2022)**

|  |  |  |
| --- | --- | --- |
| All Team Members | Role and Task Given | Individual Contribution (%) |
| 1. Nguyen Thi Ha Giang | Record video demonstration:   * User view orders history * Admin search orders by member’s id * Admin change order’s status | 100% |
| 1. Nguyen Tuan Thang | Record video demonstration:   * User sort products by category * User sort products by price * User add products to cart * User delete products in cart * User create new order | 100% |
| 1. Trinh Viet Quy | Record video demonstration:   * Customer view list of products * User registration * User login * User view list of products | 100% |
| 1. Tran Duy Phong | Record video demonstration:   * Admin view all information of products, users, and orders * Admin add new product * Admin edit product’s price * Admin delete products | 100% |

1. **Conclusion**

In general, our system is complete and runs the basic features stably. However, there are still a lot of aspects we need to improve to make our system better.

First, let's talk about the code. As can be seen, we still have a lot of errors in the code and system programming. There are still duplicated and unnecessary functions. This may result in lower performance of the application (the app will run slower due to heavier capacity). Therefore, the first thing we need to improve in the future is to fix the code to make it more transparent and efficient.

Besides, security will also be a necessary issue for our system to become more logical and secure. Currently, we still don't have a way to completely secure the database, so anyone who owns the source code can edit the database easily. This will greatly affect the system if we want to deploy this project to continue. Therefore, we are also and will be researching methods to store data as securely as possible.

And finally, we look forward to developing more features for our system. For example, the user can cancel the order, or edit the order and other features.

Although there are still quite a few shortcomings in this project, however, we will try to continue to improve the project in the best way to have a product in the most complete java language.