The God We Trust

Nuroz project report

Mohammad Ghaderi
<401 222 104>

Introduction:

- In this assignment, we will build an online shop simulator App providing e-commerce services, similar to Digikala or Amazon. The application should allow users to simulate a simplified version of the online shopping experience through the command line interface.
- A Major goal of this project for you, is to exercise your creativity and implement your own unique approach to building the application. Another goal of this project is to implement OOP concepts effectively.

If I want to give you a general view of the approach that I've used and the whole project, is that I thought and said to myself that in general if we want to build an online shop where a seller should be able to present a product and any user should be able to see that and if he\she wants that product cloud be able to add that product to his\her shopping cart and buy it.
Well I said if a seller wants to present a product so we need to declare a concept like categories and subcategories so that we cloud add that product to a subcategory and give to all the products a systematic form,
so that seller cloud add a new product to his available products easier and better,
Therefore I implement a class named "Product" and give it the primary properties of a product (like it's name, brand, price,…).
After this I implement nine subcategories that can consist almost all products in a real online shop (except sport products and some beauty products,…).
When I finished creating subcategories then I started to implement a functionality that gives the seller this ability to add a product to it's presentation and sell it.

Design and Implementation:

- Here is the UML diagrams of all the existing classes in the project:

Product class
Attributes : String name, String brand, double price, int number of available products, String color, String ID, String company name that is representing this product
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer)
Override "To.String()" method.

Abstract Clothe class extends Product class
Attributes : String material, String genre,
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Shoe class extends Abstract Clothe class
Attributes : String is it shoelace(Seller respond to this question with 'yes' or 'no')
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Sock class extends Abstract Clothe class
Attributes : String size(Seller respond to this question with "small- big" algorithm)
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Electronic devices class extends Abstract product class
Attributes : String processor, int storage
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Laptop class extends Abstract Electronic devices class
Attributes : int RAM, int weight
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Smartphone class extends Abstract Electronic devices class
Attributes : int camera quality , int number of cameras
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Electronic tools class extends abstract Product class
Attributes: double weight , int power consumption
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Car wash class extends Abstract Electronic tools class
Attributes: int maximum water temperature , int hose length
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Electric engine class extends Abstract Electronic tools class
Attributes: String fuel type, int Engine power
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Health tools class extends abstract Product class
Attributes: String Supply power, String used
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Blood Sugar test machine class extends Abstract Health tools class
Attributes: double minimum blood required, int number of test strips
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Sphygmomanometer class extends Abstract Health tools class
Attributes: String suitable arm size , String display type
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Health tools class extends abstract Product class
Attributes: String Supply power, String used
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Blood Sugar test machine class extends Abstract Health tools class
Attributes: double minimum blood required, int number of test strips
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Sphygmomanometer class extends Abstract Health tools class
Attributes: String suitable arm size , String display type
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Kitchen devices class extends abstract Product class
Attributes: double volume , String energy consumption chart
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Dish washer class extends Abstract Kitchen devices class
Attributes: int capacity , int number of baskets
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Vacuum cleaner class extends Abstract Kitchen devices class
Attributes: int engine power , int power cable length
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Musical instruments class extends abstract Product class
Attributes: double weight, String material
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Guitar class extends Abstract Musical instruments class
Attributes: int wires number, int bowl thickness
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Violin class extends Abstract Musical instruments class
Attributes: String chain material, String top plate material
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Non electronic tools class extends abstract Product class
Attributes: String material, double weight
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Non electronic tools class extends abstract Product class
Attributes: String material, double weight
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Saw class extends Abstract Non electronic tools class
Attributes: String used, int iron blade length
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Stationary class extends abstract Product class
Attributes: String material, String suitable for
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Backpack class extends Abstract Stationary class
Attributes: int number of external pockets, int number of internal pockets
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Study light class extends Abstract Stationary class
Attributes: int bubble diameter, int number of usable lamps
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Abstract Vehicle class extends abstract Product class
Attributes: int horse power, int tank volume
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Car class extends Abstract Vehicle class
Attributes: int number of available seats
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Truck class extends Abstract Vehicle class
Attributes: int number of wheels , String dose have bed
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

In the pages before you can see the "UML" diagrams of the categories and subcategories.
If you check them carefully you will get that they have implemented as " abstract class" because we don't have any product that be an object of them. In fact all the products are a part of it not itself.

User base class
Attributes: String username, String password
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

User class extends User base class
Attributes: String email address, String phone number, String address, double wallet
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

Seller class extends User base class
Attributes: double wallet, Array list<Laptop>, Array list<Smartphone>, Array list <Shoe> Array list <Sock>, Array list <Car Wash>, Array list <Electric engine>, Array list< Blood Sugar test machine> , Array list <Sphygmomanometer>, Array list<Dish Washer>, Array list<Vacuum cleaner>, Array list <Guitar>, Array list<Violin>, Array list<Hammer>, Array list<Saw>, Array list<Backpack>, Array list<Study light>, Array list<Car>, Array list<Truck>
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class + a function for all the array lists separately to add a new one to their array list.
Override "To.String()" method.

Admin class extends User base class
Attributes: String email address
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Override "To.String()" method.

In the previous page you can see the "UML" diagrams about users account available in the shop and their attributes.
The reason that I've used "Inheritance" concept for them was because all of the accounts have some attributes such as "username", "password" in common.
The reason why seller have array lists of all the product that can exist in the shop is that if a seller wants to add a product to the shop, this product in addition that get added to the shop, it should added to the lists of available products of the seller either. (That is the reason we implement functions that can add a new product to this lists to).

Shopping cart class
Attributes: String name, String brand, String category, String ID, double price, String Seller company name, String number of available items,
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Implement two functions that one of them should add a product to the shopping cart list and another one should search in the shopping cart list by the ID of the each product and return it's result.
Override the "To.String()" method.

According to the doc each user should have a shopping cart just in case he\she wants to add a product to his\her shopping cart and buy it in the future if he\she wants to.
Shopping cart have some attributes such as ("name", "brand", category",…) that gives some information about the product and seller and the user .
if user wants to buy the products that are in his\her shopping cart first user should confirm he\she wants to buy and after this a request will be make and send to the admins to check and respond to it.

Buy request class
Attributes: User user ,Seller seller, double product price, double user wallet, Boolean is checked, int number of available products in the stack, int number of ordered products, Shop shop, Shopping cart shopping cart, String ID of the product
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Implement some functions to do thing that should be done after order is confirmed. Increase the seller wallet, increase the shop wallet, decrease the user wallet, add the order to the list of purchased products of the user, decrease the number of available products according to the purchase.
Override the "To.String()" method.

in the previous page you can see the "Buy Request" class.
This class has some attributes such as ("user", "seller", "product price", "user wallet", …) that admin can see it.
If admin approve this request then some functions will be implemented, that should
1- increase the seller wallet about 90% of the product price.
2- increase the shop wallet about 10% of the product price.
3- decrease the user wallet about the product price.
4- decrease the number of available products of the product about the number of confirmed ordered products.
5- add the shopping cart to the list of the purchased product of the user.

Wallet request class
Attributes: User user , int amount of the money user wants to increase, Boolean is checked.
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class
Implement a function to increase the wallet if the admin confirm the request.

If the user wants to pay for the order and he\she didn't have enough money in his\her wallet then what should he\she do?
The answer is that he\she should increase his\her wallet. For doing this he\she should make a request for increasing his\her wallet(user can find it in their menu) about the money that they he\she wants to increase and wait for the admin respond. If admin confirm the request then the money will be added to the user's wallet.

Shop class
Attributes: String name, String web address, String support number, Array list<Admin>, Array list<Seller>, Array list<User>, Array list<Shopping cart>, Array list<Wallet request>, Array list<Buy request>, Array list<Laptop>, Array list<Smartphone>, Array list <Shoe> Array list <Sock>, Array list <Car Wash>, Array list <Electric engine>, Array list< Blood Sugar test machine> , Array list <Sphygmomanometer>, Array list<Dish Washer>, Array list<Vacuum cleaner>, Array list <Guitar>, Array list<Violin>, Array list<Hammer>, Array list<Saw>, Array list<Backpack>, Array list<Study light>, Array list<Car>, Array list<Truck> , double total profit of the shop
Implementation the setter-getter methods of the attributes (Because our attributes declared private for encapsulation concept and make the program safer) + the constructor of the class f
Functions for creating account (User \ seller), Functions to check if a "user" or "seller" or "admin" exists, also some functions for log out, and some functions to show their profile.
Implement functions to add new products to the subcategories (These array lists that have been declared in the top) because if a seller add a product to represent this product must add to the shop either. Also implement some functions for searching in the list of the subcategories(Products array lists) by their "ID" and "name".
Implement a function to show all the products available and if the stack is empty print s meaningful message . Also a function to return the result of the search by name in general by the user to only watch the products.
Declaring some functions to add a request and remove a request of "wallet request" and "buy request".

And finally you can see the Shop "UML"
Shop contains almost all the important functionalities and save everything in itself as you can see in the attribute part of the "Shop UML". Such as all the products that exists in the shop a list of all users, admins and sellers and also a list of all request whether it’s a buy request or wallet request.
There is exist functions like search product by it’s name and by it’s ID, creating an account or login in accounts and deleting an account.

Done

Thank you for reading .