Programming techniques

Homework 2: Simulating an application for processing customer orders

Student: Sporis Ioan Catalin

Group: 30421

Contents:

I. Project specification

The task of this homework is surprised in the next specifications : Implement and test an application OrderManagement for proessing customer orders. It is necessary to have the classes Order, OPDept (OrderProcessing Department), Customer, Product and Warehouse.

Using this application we are able to see the products that are under-stock or overstock, to display totals, to manage the insertion, deletion and uptading both of products, and of customers by the admin user. Moreover, the application offers the possibility of creating new clients accounts, accounts athat are further saved in a database. Also, we are able to make searches on the products list, in order to decide what to buy. The application is designed to deliver the user a bill which informs him about the details of the transaction.

II. Analysing problem, modeling application, scenarios, using cases

a. Analyzing and modeling the problem

Taking into account the problem's specificcation have decided to implement an application that realizes the simulation of a warehouse. Also I have decied to implement a system that is widely found. More exactly I followed the model of online shops, where the customer can add an ordr or can see the products that are in stock of some warehous, or to search for a product , and the admin user can add new products or to see the customer orders, or see products that are under a certain amount – uderstock – or over a certain amount – overstock – having unlimited possibilities.

The principal problem, that is found, is finding a balance between the privileges that a user has and an administrator.It can we noticed from the problem's specification that the principal actors are the customers that perform the orders and the manager that executes the orders, adds, deletes or modifies products from the store.

The user is able to se the product and also the number of products available in the market. Based on this he can add an order taking into account his wish. Also the buyer is able to see the price for every product.

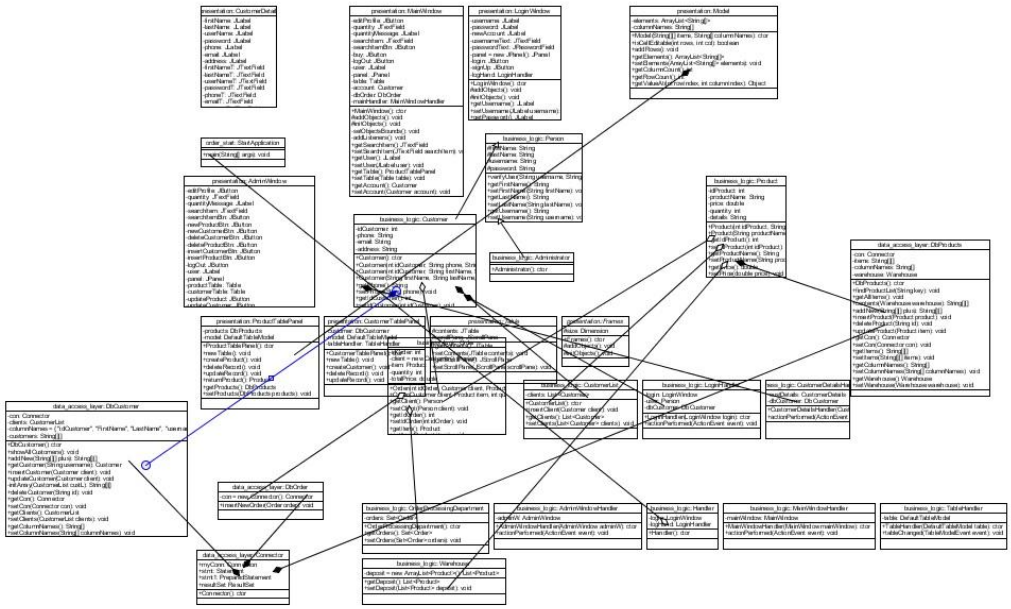
For developing this application I have used a database design which holds information needed for well working in some tables named accordingly ("customer", "product", "order"); for linking my java app to the database I have used some king of driver, by means of which I am abe to move info from one place to another. The driver is the one managing the connection, by the field "connection", and by means of SQL language, it allows creating different statements like querries, insertions, delitions, or uptation.

At the first reading of the problem we notice that our application should contain at least the next classes: Order, OPDept (Order Processing Department), Customer, Product, and Warehouse.

At the first analysis we make distinction between Product Class and Customer class. Product class contains methods like getters and setters of the attributes that a product owns: id, name, quatity and price.

Also the Customer class contains methods like getters and setters of the attributes: id, firstname, lastname ,username, password, phone email adress.

As long as the interaction between user and application is concerned, this is realized using a graphical user interface that should not create problems for usage. The user must know where is the starting button, where he will receive the result, and where are the panels from inserting information. Also the introduction of data must be user friendly. The application that I have developed has a graphical user interface , that allows the user to insert data in the text fields and also buttons for validating an event.



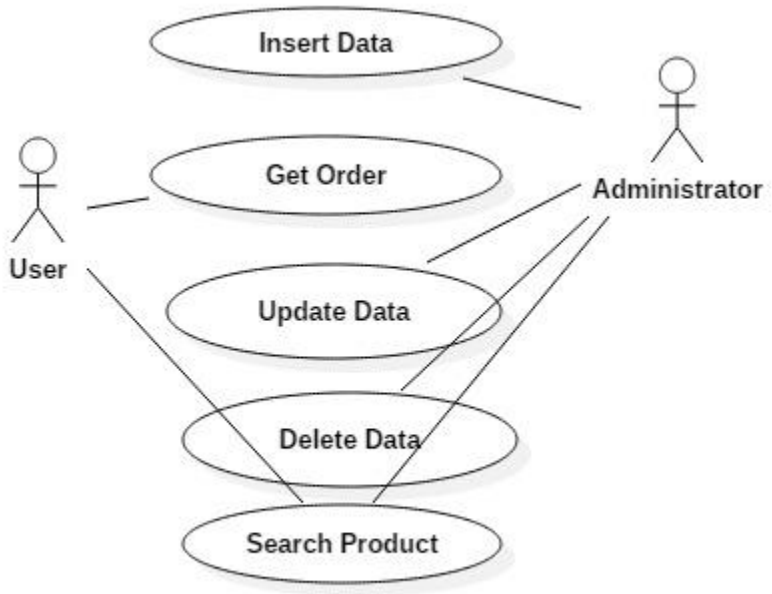
b. Scenarios, using cases

After a preliminary analysis regarding the implementation of the problem, a using case of the application would be:

- ✓ Launching the application
- ✓ Selecting the logging type: user or admin

- ✓ If the user mode was selected then the name of the user and his password should be inserted. Operations like: displaying products from the warehouse, displaying the own orders, making a new order, see products with a price less than a given value
- ✓ If the admin mode is selected then the operations that can be performed are: add products, delete products,modifying products attributes, computing the total value of the products from the warehouse.

In case the format of entering data is not respected than different box messages will be displayed. There are tested cases like the actor introduces letters instead of numbers, or the text fields are empty, giving him the possibility to introduce the data again, correctly.



III. Design

The programming language used for building this application is Java, and the developing platform is Eclipse. The interaction with the user is realized in an interactive way using a graphical user interface.

Package : **order_startapplication**: has just one class, „**Start Application”,** which has the major role, as its name says, to start my java application by instantianing a new „Handler” object.

StartApplication :

Method: public static void main (String [] args);

„data_access_layer” – contains classes that are the major players in the exchange of informatin between Eclipse, the java compiler, and Workbench, the system that holds the database;

Connector :

Fields: - private Connection myConn : sets connection between the 2 softwares:

(myConn=DriverManager.*getConnection*("jdbc:mysql://localhost:3306/ordermanagement", "root", null);

DbCustomer : retrieves information from customer table, instatiate objects of type ” **Customer “ ,** and insert in the database info for “ **Customer** ” table.

Fields:

- **Private Connector con ;**
- **private String [ ] columnNames** - array of strings which holds the names of customer table's columns
- **private String [ ] [ ] customers** – two dimension array whose contents are the records of customer table from atabase

Methods :

- **public void showAllCustomers() –** retrieves all record fields of customers
- **public String [ ] [ ] addNew(String [ ] [ ] plus)** – increseas the length of the customer array
- **public Customer getCustomer (String username)** - returns an object of type Customer
- **public void insertCustomer (Customer client)** - inserts in Customer table a new record with information about a new Customer
- **private String [ ] [ ] initArray (CustomerList custL)**
- **public void deleteCustomer (String id)** - deletes a record of table Customer, by a key of its id, if exists

DbProducts :

Fields:

- **private Connector con**
- **private String [ ] [ ] items** – two dimension array containing information about products
- **private String [ ] columnNames** – keeps the names of product columns, used in the JTable which displays information
- **private Warehouse warehouse;**

Methods :

- **public void findProductList (String key) –** finds products by the name;
- **public void getAllItems ( ) -** creates the two dimension array of products
- **public String [ ] [ ] contents (Warehouse warehouse)**
- **public String [ ] [ ] addNew (String [ ] [ ] plus)** – increseses items length
- **public void insertProduct (Product product)** - inserts a new product record
- **public void deleteProduct (Stirng id)** – deletes product form database after the given id

bussiness_logic packge

Administrator: inheritant from Person

Fields : Just the one inherited from the super class

Custommer : extends Person

- private int idCustomer
- private String phone
- private Sring email
- private String address

CustomerList: -list of customers

Handler : handler for login window; implements ActionListener

Fields :

- **private LoginWindow login**
- **private LoginHandler logHand**

LoginHandler implements ActionListener

Fields :

- **private LoginWindow login**
- **private Person user**
- **private DbCustomer dbCustomer**

MainWindow handler : implements ActionListner

Order

Fields :

- **private int idOrder**

- **private Person client**
- **private Product item**
- **private String orderDate**
- **private int quantity**
- **private int totalPrice**

OrderProccessingDepartment – List of orders

Person

Fields :

- **protected String firstName**
- **protected String lastName**
- **protected String username**
- **protected String password**

Methods: - public Boolean verifyUser(String username , String Password)

Product implements Comparable

Fields :

- **private int idProduct**
- **private String productName**
- **private double price**

- **private int quantity**
- **private String details**

presentation package:

AdminWindow – is the graphical user interface wen admin user is introduced

Const – an enum of string constants

CustomerDetails

CustomerDetailsHandler

CustomerTablePanel

Frames – superclass of all frames

Login window

MainWindow

Model [ ]ProductTablePanel

Table

IV.

V. Results

The results have a designated area in the graphical user interface. By testing the right functioning of the application on various types of data, we can see that these are well performed. So the simulator has a good performance. Also the result can be immediately observed for the most of the operations. For example in case a new product is added or an

existing product is deleted or modified the results are available quickly. Also this is happening too when placing a new order or filtering the data after price.

VI. Conclusions, further developments

To sum up, realizing a program which is able to simulate orders is not as hard as I thought at the very beginning. This application is easy to use because of the graphical user interface. Realising this homework I learned how to make a JTable and print data in a table.

The program could be developed by creating filters, to display products having certain attributes, like name and quantity, see the date when they were added.

VII. Bibliografy

http://stackoverflow.com/

Mainly google.ro