Warehouse

Programming Techniques – Second Project

Group: 30424

First\_Name: Daiana

Second\_Name: Spatacean

Teacher: Claudia Pop

Summary:

1**. Introduction**

1.1 Problem specification . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

2. **Description of the project**

2. 1 Problem analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

2. 2 Modeling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

3. **Projection**

3. 1 UML diagrams . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . .

3. 1. 1 Use-case diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . .

3. 1.2 Class diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . .

4**. Implementation** . . . . . . . .. . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . .

5. **Implementation** **and testing** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

6. **Further developments .** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

7. **Conclusions** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

8. **References**  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

1. **Intoduction**

A warehouse is a [commercial building](https://en.wikipedia.org/wiki/Commercial_building) for storage of [goods](https://en.wikipedia.org/wiki/Good_%28economics%29). Warehouses are used by [manufacturers](https://en.wikipedia.org/wiki/Manufacturer), [importers](https://en.wikipedia.org/wiki/Importer), [exporters](https://en.wikipedia.org/wiki/Exporter), [wholesalers](https://en.wikipedia.org/wiki/Wholesaler), [transport](https://en.wikipedia.org/wiki/Transport) businesses, [customs](https://en.wikipedia.org/wiki/Customs), etc. They are usually large plain buildings in industrial areas of cities, towns and villages.

Historically, warehouses were a dominant part of the urban landscape from the start of the [Industrial Revolution](https://en.wikipedia.org/wiki/Industrial_Revolution) through the 19th century and into the twentieth century. The buildings remained when their original usage had changed. There are four identifiable types of warehouses. The cotton industry rose with the development of the warehouse, and all five types were represented in [Manchester](https://en.wikipedia.org/wiki/Manchester) in the United Kingdom. Warehouses of that period in Manchester were often lavishly decorated, but modern warehouses are more functional.

Warehouse allow [transport](https://en.wikipedia.org/wiki/Transport) optimization along the supply chain, and allow companies to work with an optimal inventory ([economic order quantity](https://en.wikipedia.org/wiki/Economic_order_quantity)) regarding service quality. For example, at the terminal point of a transport system it is necessary to stockpile produce until a full load can be transported. Warehouses can also be used to store the unloaded goods from the vessel. In industries whose goods require a period of maturation between production and retail, such as [viniculture](https://en.wikipedia.org/wiki/Viniculture) and [cheesemaking](https://en.wikipedia.org/wiki/Cheesemaking), warehouses can be used to store the goods in large quantities.

**2.Description of the project**

The purpose of this assignment was to make some operations on the products that we can find in a regular warehouse. Frist we can make some changes on the products like the quantity and add new products if it is necessary. Second is about the customer : add a new one or delete an existing one. And at the end we can make some orders regarding the clients and the products that are in stock.

Also this project is made having in the “back” a database in which I have 3 table: Customer, Product and Order. The relations between them are many to many because a customer can order many products. So the database constains 3 tables.

**3.Diagrams**

**3.1 Use-case diagram**

A **use case diagram** at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different [use cases](https://en.wikipedia.org/wiki/Use_case) in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

The user can be any person that wants to use the application and the interface must be pretty friendly.

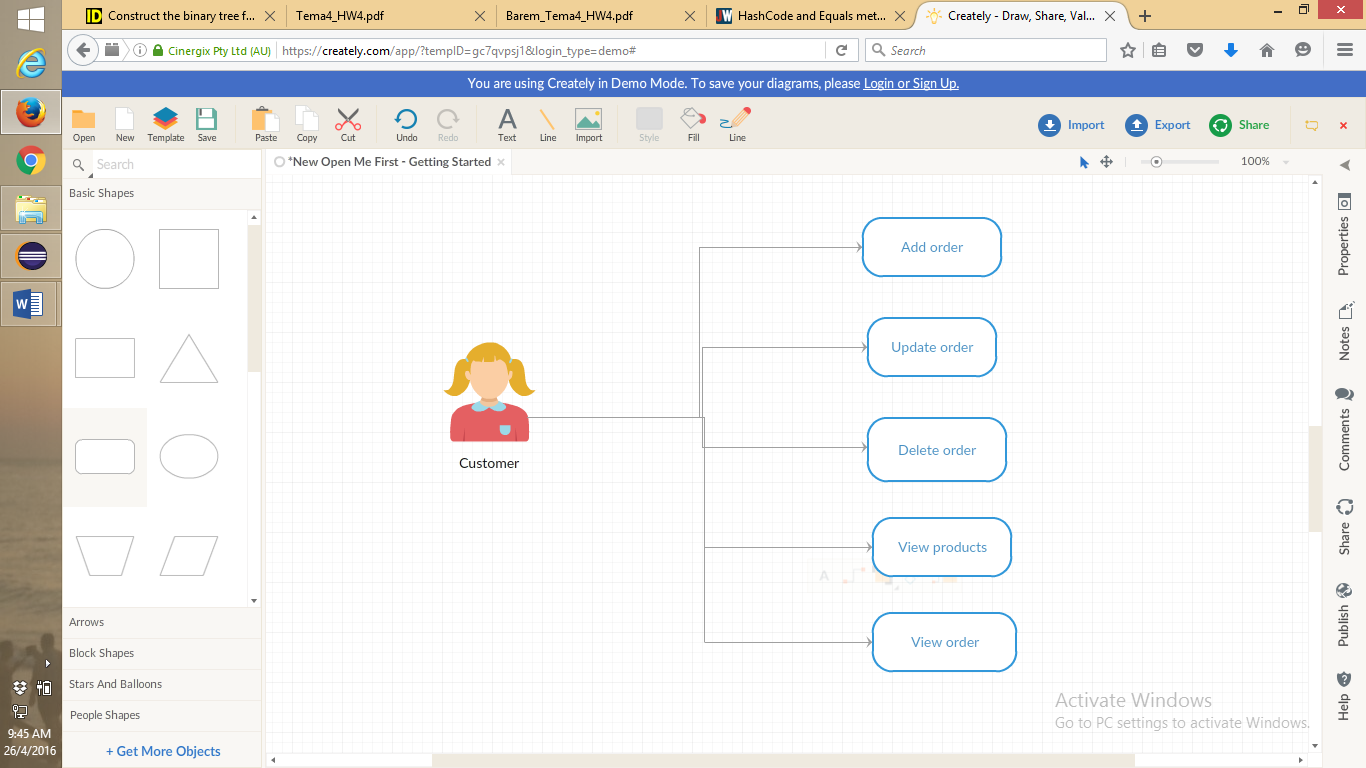
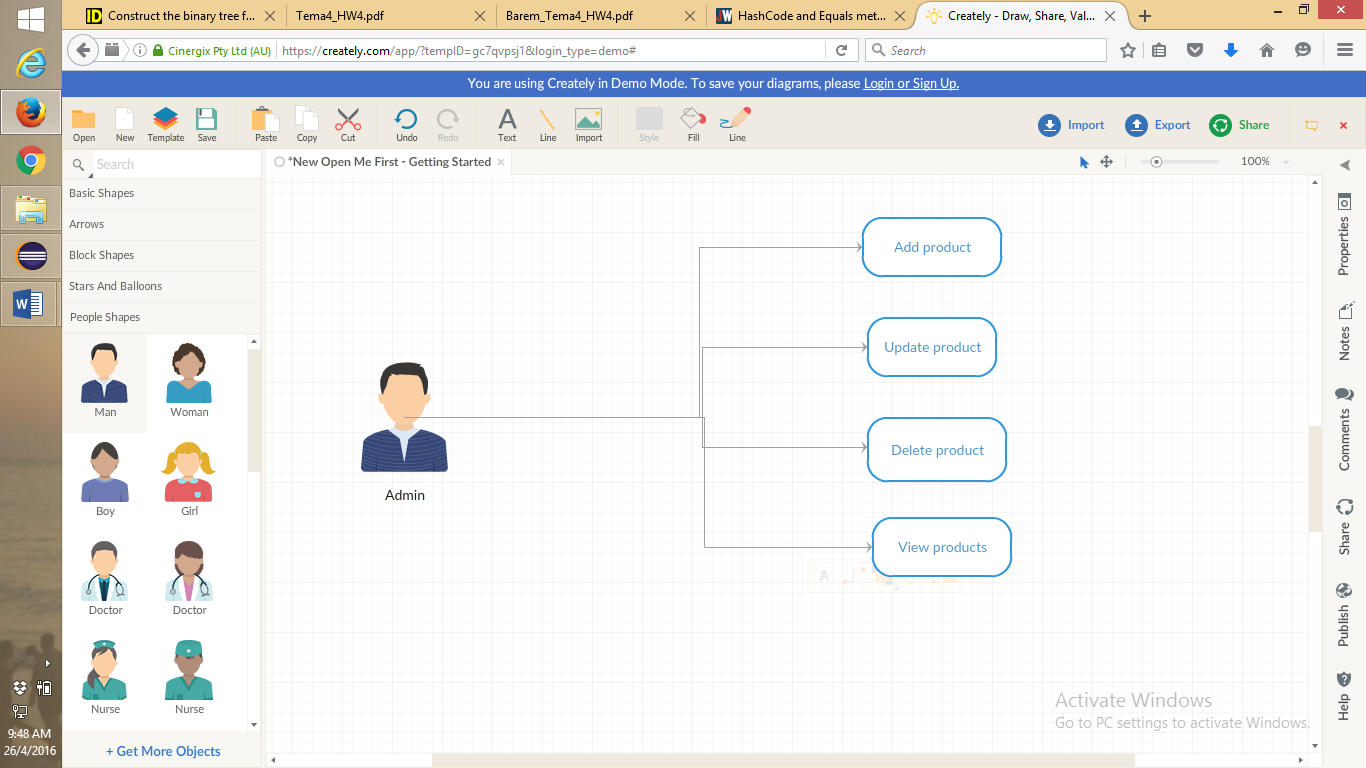
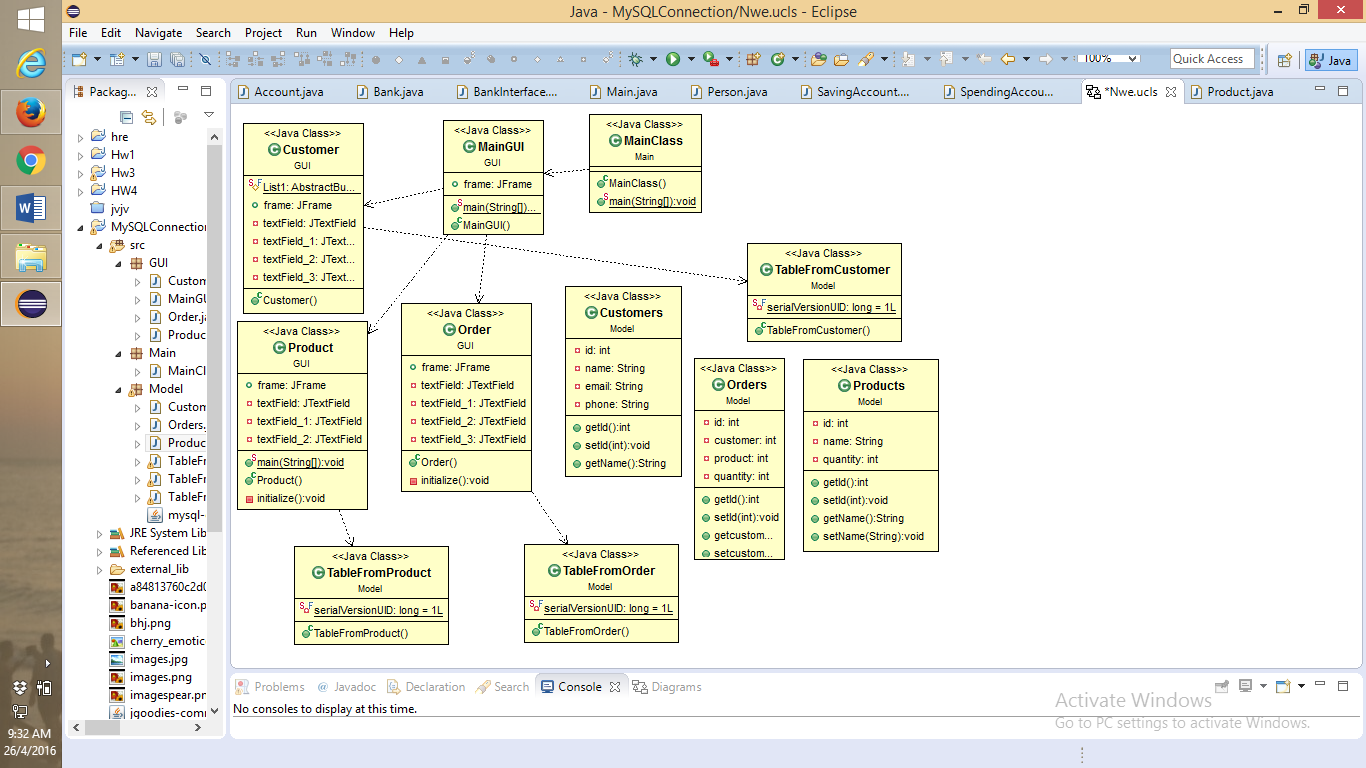


Fig. User- diagram



**3.2 Class Diagram**



**4.Implementation**

This project is made following the OOP structure (classes, methods) and principles:

- **inheritance** when one object acquires all the properties and behaviours of parent object i.e. known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism

- **polymorphism** when one task is performed by different ways i.e. known as polymorphism. For example: to convert the customer differently, to draw something e.g. shape or rectangle etc.

- **abstraction** hiding internal details and showing functionality is known as abstraction. For example: phone call, we don't know the internal processing

- **encapsulation** binding (or wrapping) code and data together into a single unit is known as encapsulation. For example: capsule, it is wrapped with different medicines.

Starting with the database:

* The table Customer contains the id of the customer which is a unique key, the name, the email address and the phone number
* The second table is called Product which also contains the id, name and quantity
* And the last one is about the Order

1. **Model package**

* Class **CUSTOMERS**
* Contains some private variables which are unique for each customer

**public** **class** Customers {

**private** **int** id ;

**private** String name ;

**private** String email ;

**private** String phone ;

* Class **PRODUCTS**
* Contains some private variables which are unique for each product

**public** **class** Customers {

**private** **int** id ;

**private** String name ;

**private** int quantity ;

* Class **ORDERS**
* Contains some private variables which are unique for each ORDER

**public** **class** Customers {

**private** **int** id ;

**private** int customer;

**private** int product;

**private** int quantity ;

* Class **TableFromCustomer**
* Here is a class for printing the jtable for the customers
* Class **TableFromProduct**
* Here is a class for printing the jtable for the products
* Class **TableFromOrders**
* Here is a class for printing the jtable for the orders

**2.GUI package**

* In this one is implemented the interface for the project
* It contains windows for order customer and product
* Also here is a main window for the GUI

**public** **class** Customers {

**public** **class** Products {

**public** **class** Orders {

**public** **class** MainGUI {

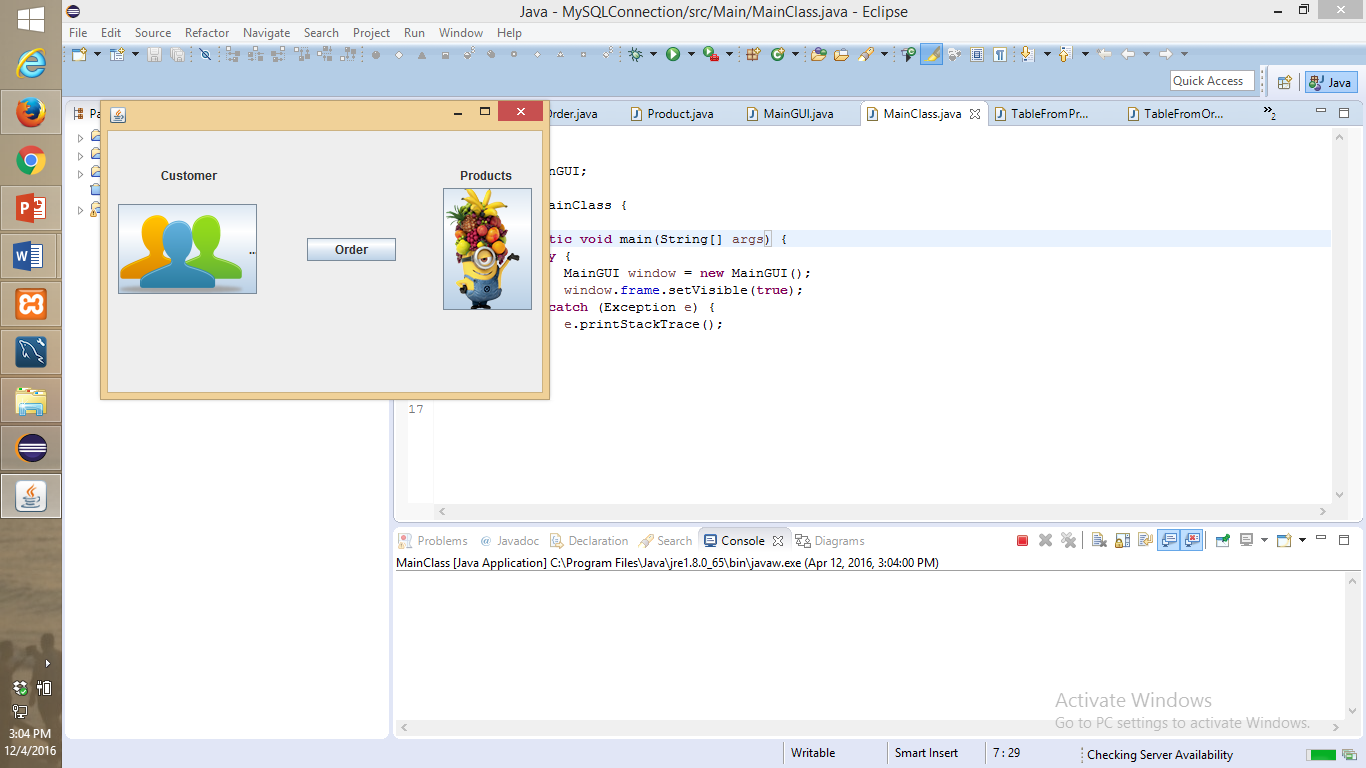
**3.Main package**

* here is only one class in which all the ” magic ” happens

**public** **class** MainClass {

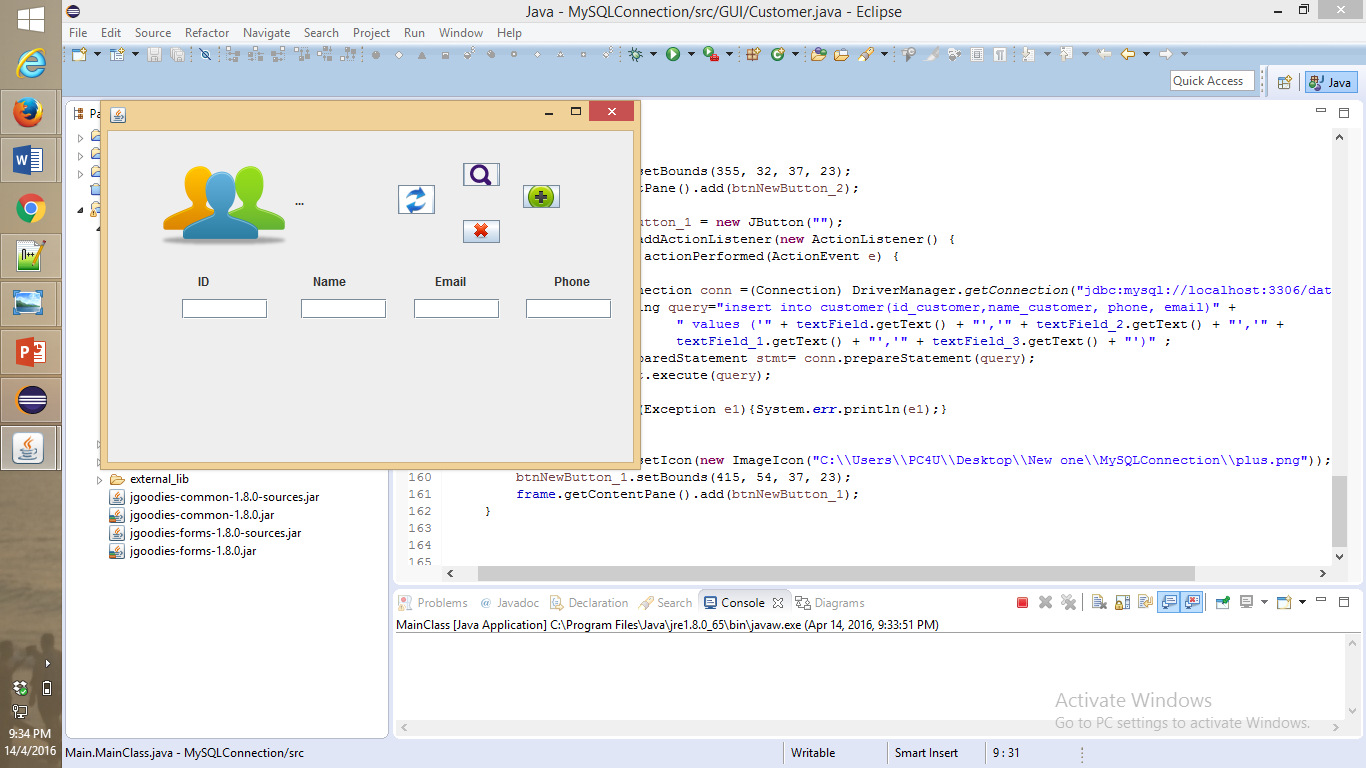
5. **Implementation** **and testing**

This is how the interface looks like:



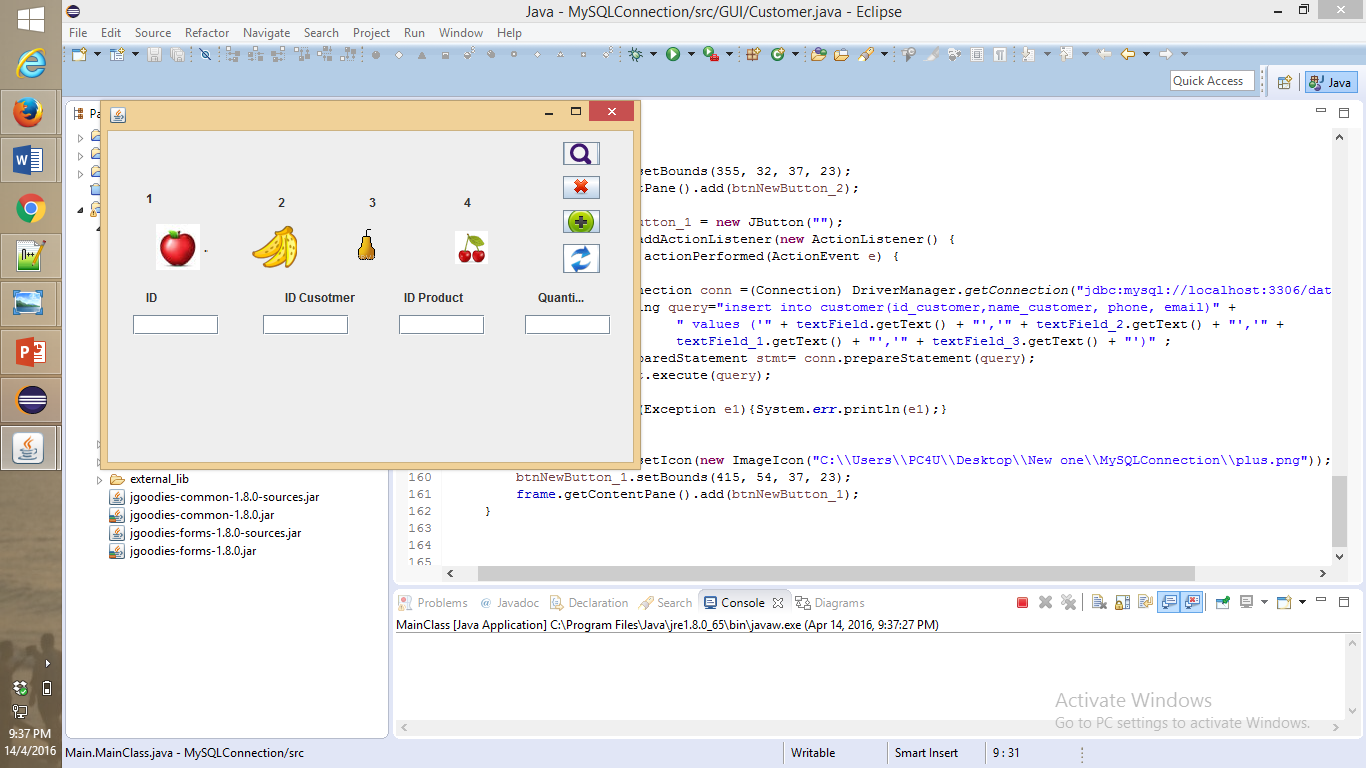
* The **user** should:
* Enter the id, name and so on for each option that he can choose: order, customer, product
* Then if the user chooses to press the button for printing the table and what is in the table he can see the changes that were made in the tables
* After that the user can make another change in another table and so on

If the user selects the first button which is **customer** a new window will be opened and it will look like this:

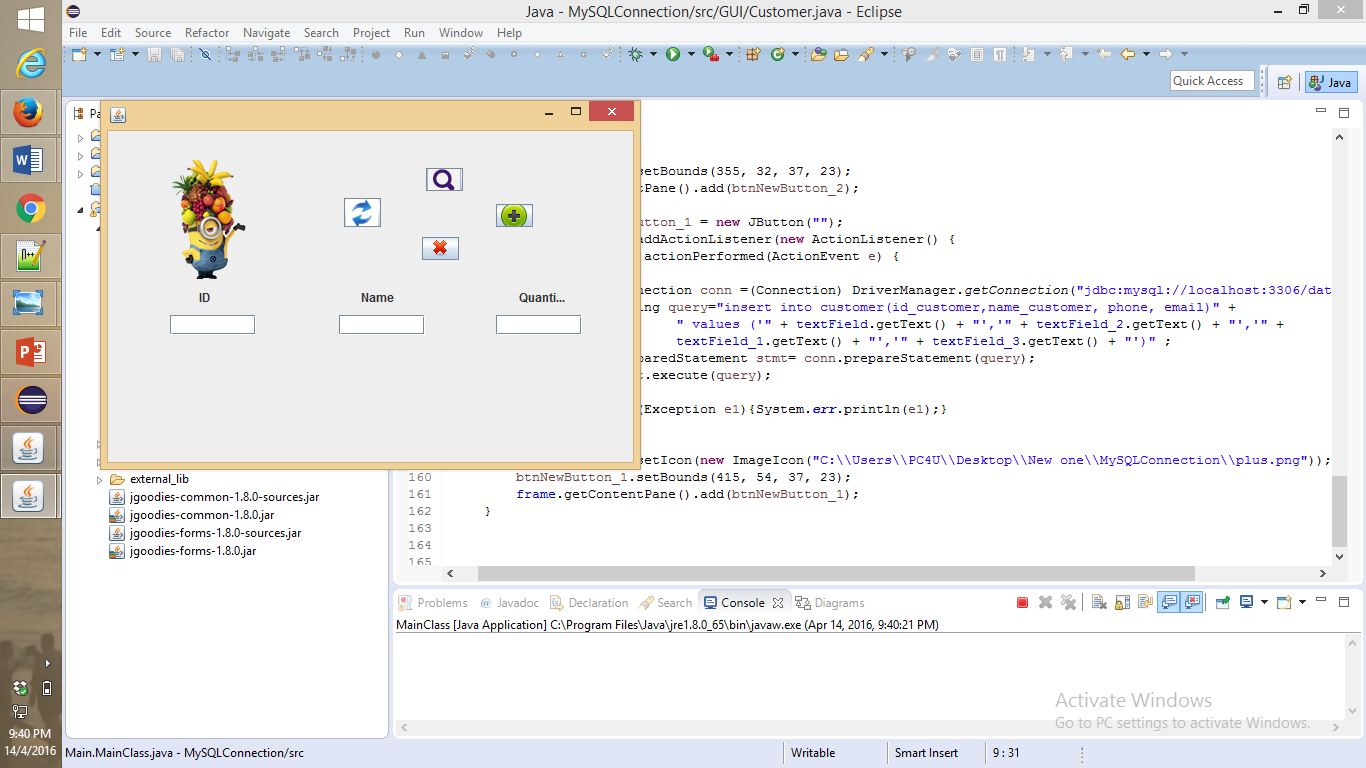


In the text fields the user will input what he want to see in the database. If the first button is clicked a window with the table from the database will appear. And also there are buttons for update a customer delete and add a new one.

For the **order** the window is like that in the next picture. The id for ordering apples is 1 and so on. This will make easier the order command.



And finally the **product** window in which the user can input products that will be sold in the warehouse.



Regarding the implementation process, I used as program Eclipse IDE. During the implementation of the project I made a lot of changes. First my program was into one package and then I split it. The classes had many methods. I also start the project not based on OOP principles. Before the interface was made I print the results in the console to see if they are correct and to know if I continue in that manner or I had to make some changes.

5. **Results**

I am pretty satisfied with my work and I think that the project is very easy to understand and the most important thing is that any type of user is able to use the interface and will have no problem with it. And in the end I think that I implemented all the petitions of the “client”.

6. **Further developments**

I think that I can make a very long list with the things that I want to improve related to my project with the polynomial calculator:

* Starting from the “back” of the project: I like to make more operations like: if the order is made then when it is delivered a message will appear and then the option for the customer to be able to modify the order even if it was delivered
* Also the product window can be modified because it doesn’t look so nice
* I want to make the database much complex, this means to add more tables and make more connections between them
* I also want to make reflection on the tables
* More OOP with more classes and method with less lines of code and not so many duplicate code in some
* I want to have a more structured code
* And for the interface I want to make it much pretty
* For the buttons if I will make them with colors and with another type of writing and for the toolbar I want to implement help and menu and so on
* It will be nice if the size of the window can be modified from the user

7. **Conclusions**

To conclude, I can say that this project meant hard work, a lot of new things learned, focusing, development and creativity. Even if I encountered a lot of problems, I was able to fix them after all, by searching on the internet or asking a colleague for advice. I think that my application satisfies the requirements and the users will have at their disposal all its functionalities. And in the end I found out that is very interesting to work with database in java because you make the connection and every thing that you modify in the database in any table it can be seen from the java program that you use.

8. **References**

* <https://en.wikipedia.org>
* <http://stackoverflow.com>
* <http://www.oracle.com/technetwork/articles/java/index-137868.html>
* <http://www.tutorialspoint.com/java/java_data_structures.htm>

And others!

.