Student: Sergiu Redeca

**Group: 30234**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

*The application is used to manage the orders of a furniture manufacturer. It has two types of users: the regular user (the order manager) and the administrator.*

*The regular user can:*

* *Add/update/view order information (customer, shipping, address, identification number, delivery date, status)*
* *Create/update/delete/view product information (name, description, color, size, price, stock)*
* *Add products to order (the price should be updated accordingly)*

*The administrator can:*

* *CRUD (create, read, update, delete) employees’ information*
* *Generate reports for a particular period containng activities performed by the employees*

# Functional Requirements

*The users should perform the previously mentioned opperations.*

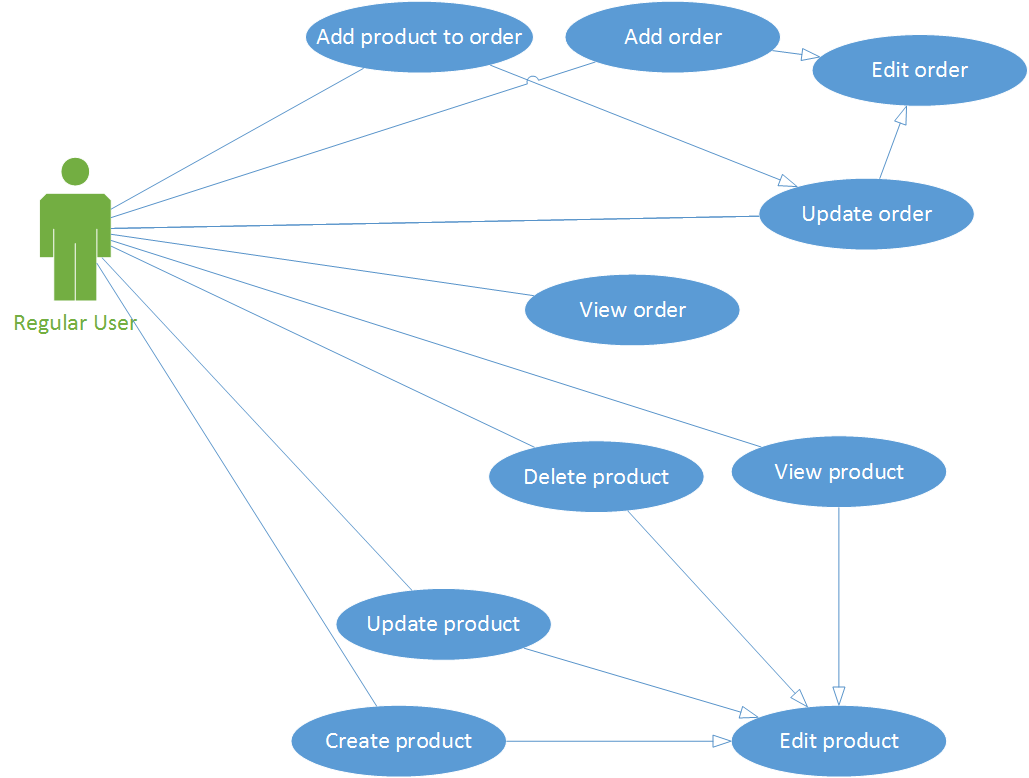
*The administrators can also generate reports for user actions.*

# Non-functional Requirements

*The software should be easy-to-use.*

*Also, the data will be stored in a DB. The login should be secured.*

2. Use-Case Model



*Use case: normal user operations*

*Level: user-goal level*

*Primary actor: normal user*

*Main success scenario: <the steps of the main success scenario from trigger to goal delivery>*

*Extensions: <alternate scenarios of success or failure>*

*Use case: administrator operations*

*Level: user-goal level*

*Primary actor: administrator*

*Main success scenario: <the steps of the main success scenario from trigger to goal delivery>*

*Extensions: <alternate scenarios of success or failure>*

3. System Architectural Design

**3.1 Architectural Pattern Description**

*For this assignment the MVP (Model- View- Presenter) architecture was used.*

*We have the following parts:*

* *Presenter*
* *Model*
* *View*

*Model–view–presenter (MVP) is a derivation of the model–view–controller (MVC) architectural pattern, and is used mostly for building user interfaces.*

*MVP is a user interface architectural pattern engineered to facilitate automated unit testing and improve the separation of concerns in presentation logic:*

*The model is an interface defining the data to be displayed or otherwise acted upon in the user interface.*

*The presenter acts upon the model and the view. It retrieves data from repositories (the model), and formats it for display in the view.*

*The view is a passive interface that displays data (the model) and routes user commands (events) to the presenter to act upon that data.*

**3.2 Diagrams**

*MVP architecture*



4. UML Sequence Diagrams

*Sequence diagram for the order manager. In this scenario the user wants to add a product.*

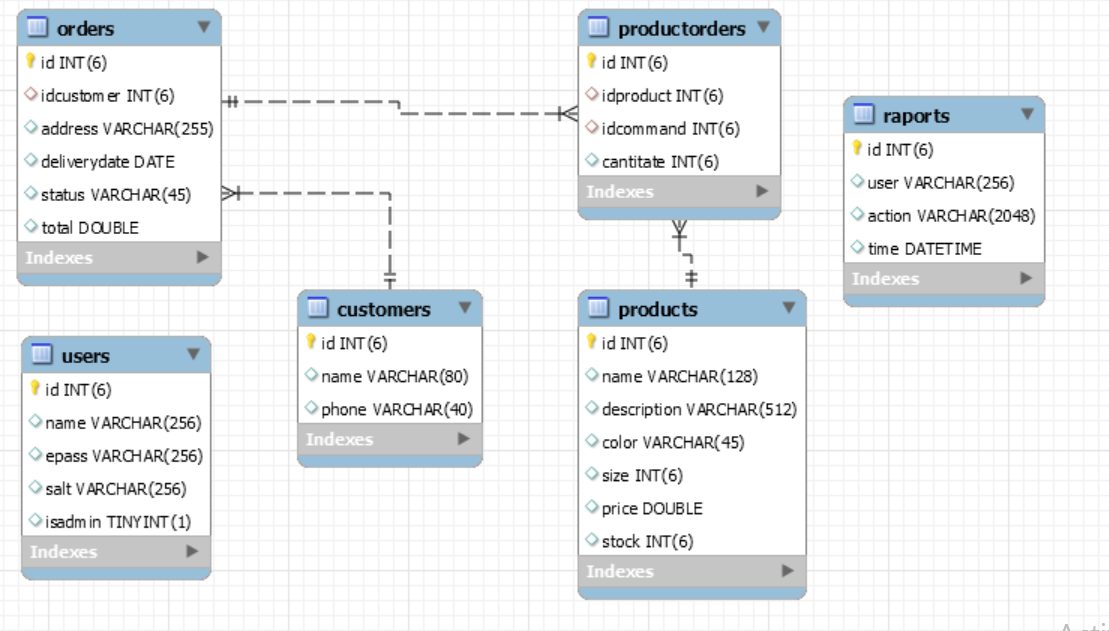


5. Class Design

**5.1 Design Patterns Description**

*The Factory design pattern was used to generate the xml and the txt file report.* *In class-based programming, the factory method pattern is a creational pattern that uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created. This is done by creating objects by calling a factory method—either specified in an interface and implemented by child classes, or implemented in a base class and optionally overridden by derived classes—rather than by calling a constructor.*

6. Data Model

**

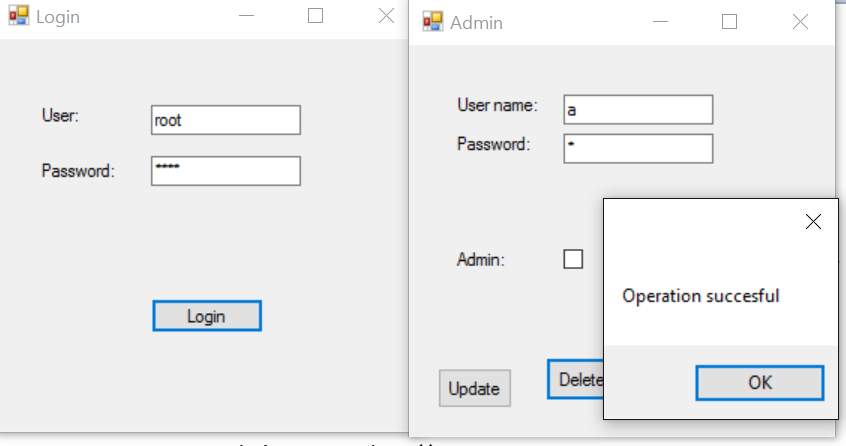
*Beside the SQL description, the data model is also described in the C# program.*

7. System Testing

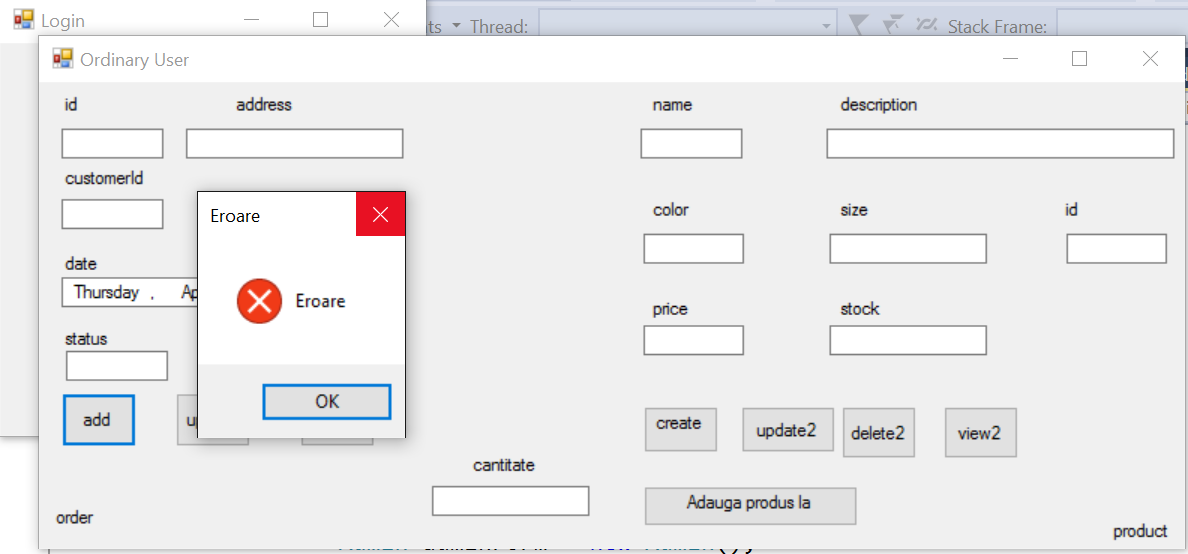
*The system testing have been made using the .NET program and also using MySQL Workbench.*

*In the program there are displayed several error messages if the user makes an error and also there are some messages displayed in case of success.*

*In the image below the administrator have sucessfully added a new user, so he/she sees a message to confirm that.*

**

*Here on the other side, the user tried to add an order without specifiying the details. He/she got an error message.*

**

8. Bibliography

<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter>

<https://en.wikipedia.org/wiki/Factory_method_pattern>