Student:Vid Alexandru

**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

Use Swing/C# API to design and implement an application for the order managers of a furniture manufacturer. The application should have two types of users (a regular user represented by the  order manager and an administrator user) which have to provide a username and a password in order to use the application.

# Functional Requirements

The regular user can perform the following operations:

* Add/update/view order information (customer, shipping address, identification number, delivery date, status.).
* Create/update/delete/view product information (title, description, color, size, price, stock etc).
* Add products to order and update order value and stock accordingly.

The administrator user can perform the following operations:

* CRUD on employees’ information.
* Generate reports for a particular period containing the activities performed by an employee.

# Non-functional Requirements

The data is stored in a database. Layers architectural pattern is used to organize the application. Domain logic and a data source pure pattern are also used for the application. The login is performed in a secured manner.

2. Use-Case Model

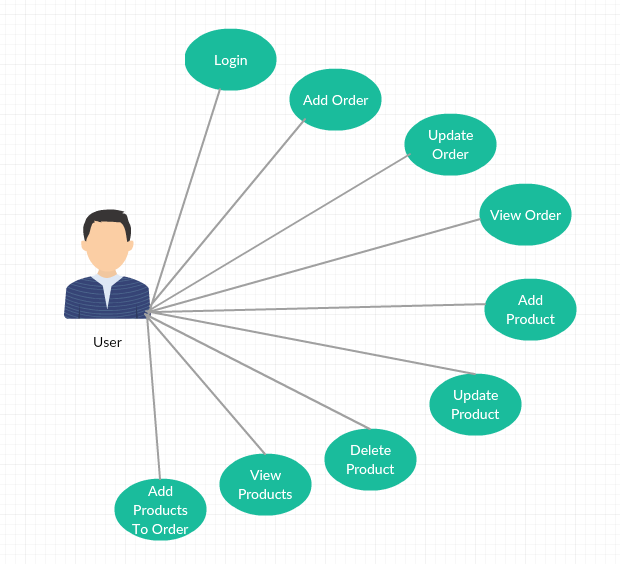
Use case: adding a product

Level: user-goal level

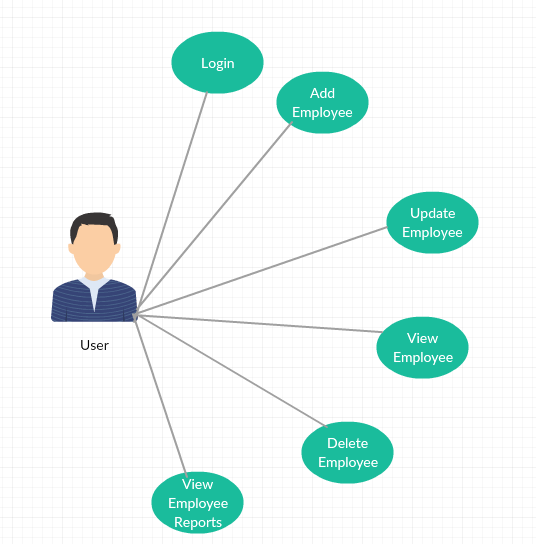
Primary actor: regular user

Main success scenario: login (username and password required)-go to products tab – fill in the information about the product – click “Add Product” button

Extensions: failure : the product already exists in the database



[Use case diagram for a regular user]



[Use case diagram for an administrator user]

3. System Architectural Design

**3.1 Architectural Pattern Description**

Three-tier architecture is a client-server architecture in which the functional process logic, data access, computer data storage and user interface are developed and maintained as independent modules on separate platforms. Three-tier architecture is a software design pattern and a well-established software architecture.

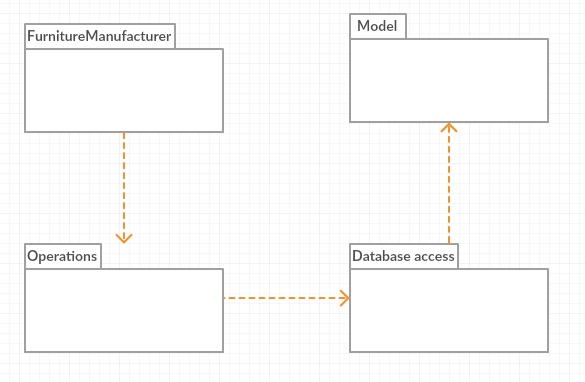
**3.2 Diagrams**

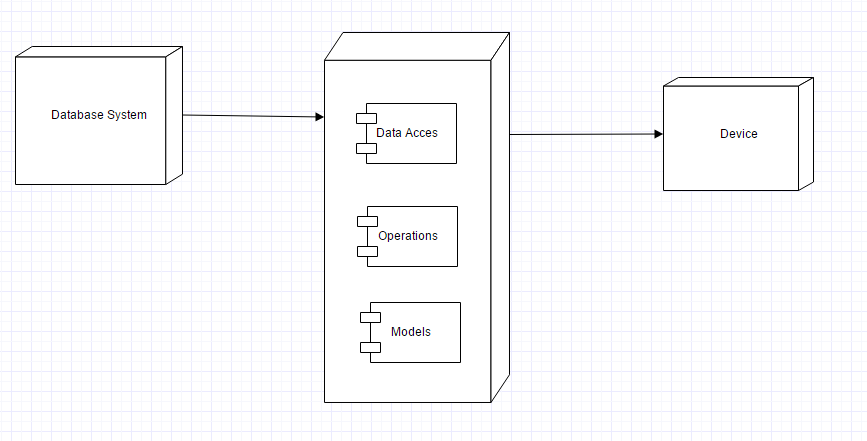
**3.2.1. Conceptual Diagram**



[3-Layer architecture conceptual diagram]

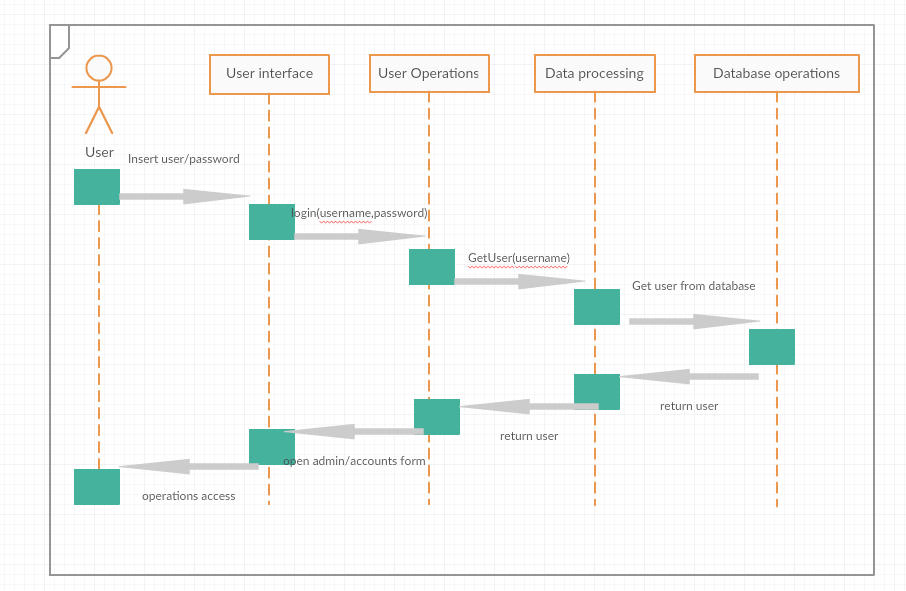
**3.2.2 Package Diagram**





[Deployment Diagram]

4. UML Sequence Diagrams

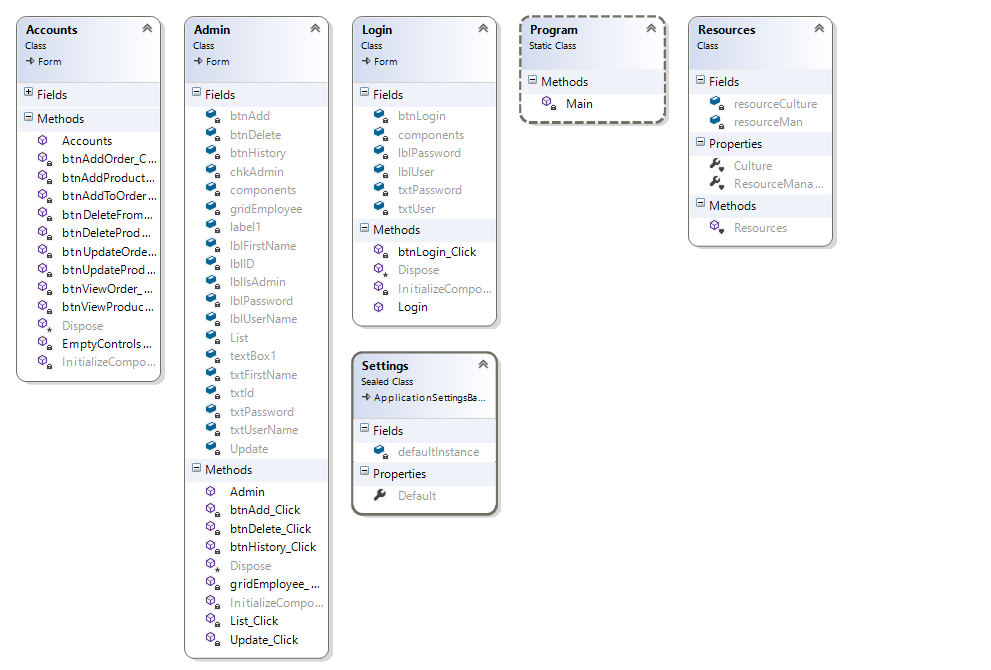


5. Class Design

**5.1 Design Patterns Description**

The three-tier architecture model, which is the fundamental framework for the [logical design model](https://msdn.microsoft.com/en-us/library/windows/desktop/ms682280(v=vs.85).aspx), segments an application's components into three tiers of services. These tiers do not necessarily correspond to physical locations on various computers on a network, but rather to logical layers of the application. How the pieces of an application are distributed in a physical topology can change, depending on the system requirements.

**5.2 UML Class Diagram**

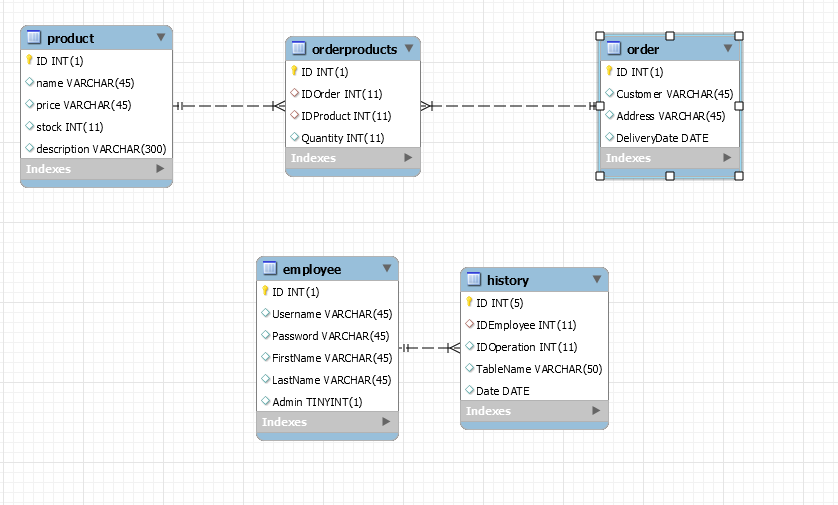


[UML Class Diagram]

6. Data Model

As data models I used in my implementation the following:

* Product
* Order
* Employee
* Order products
* History



8. Bibliography

<https://www.techopedia.com/definition/24649/three-tier-architecture>

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

<http://exponential.io/blog/2015/03/05/3-layer-architecture-in-detail/>