Student:Ilies Alina Denisa

**Group:30233**

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 JAVA/C# API to design and implement an application for the front desk employees of a bank. The application should have two types of users (a regular user represented by the front desk employee 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 client information (name, identity card number, personal numerical code, address, etc.).
* Create/update/delete/view client account (account information: identification number, type, amount of money, date of creation).
* Transfer money between accounts.
* Process utilities bills.

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 system should have access to a database in order to stock informations and it should use a domain logic pattern or a data source hybrid pattern and a data source pure pattern.

2. Use-Case Model

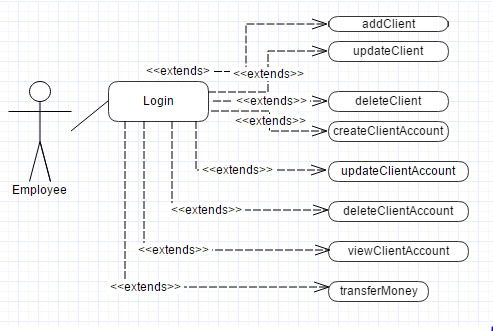
Use case: Insert a new account

Level: user-goal level

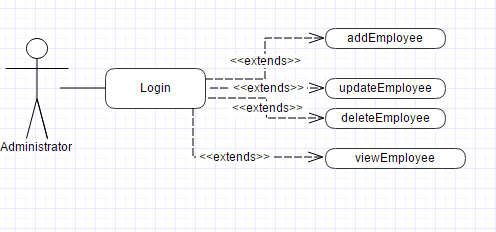
Primary actor: regular employee

Main success scenario: The user is successfully added in database.

Use case for Employee



Use case for Admin



3. System Architectural Design

**3.1 Architectural Pattern Description**

I used a Layered Architectural Pattern with 4 main layers :

-presentation: responsible for user interfaces

-dataaccess: responsible for accessing the database

-business: responsible for the logic of application

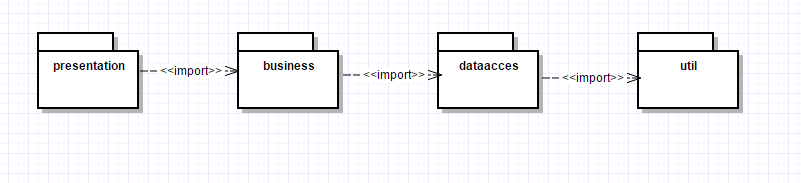
-util: responsible for access the database

In my case dataaccess layer have another 2 packages : model (in which I use a class for each table from the database ) and dao ( in which I put the operations with database).

In the same way, the business layer is formed by : logic package in which I call the methods from dao and controller package which make the connection between model and user interface.

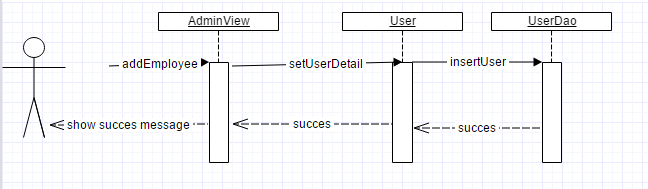
**3.2 Diagrams**

Package diagram :



4. UML Sequence Diagrams

Sequence Diagram for adding a new User by Administrator



5. Class Design

**5.1 Design Patterns Description**

Design Patterns used in this application:

-MVC in order to connect different parts of application like model, view and controller.

The model represents Java objects, the view represents the way in which data can be visualize and controller control the data flow and update the view when parts of the model are changed.

-Data Source Table Module in order to access every table from the database

-Data Source Table Data Gateway in order to access database and make operations on it.

**5.2 UML Class Diagram**

*[Create the UML Class Diagram and highlight and motivate how the design patterns are used.]*

6. Data Model

Data is represented in multiple tables:

-client which contains: id, name, cardNo, cnp

-account which contains: id, type, amount,creationDate,idClient

In this table, the last column idClient is a foreign key to the column id from table client

-employee which contains: id, username, password, role

Column role can have integer values, and depends on them the role for every user is established.

7. System Testing

All the operations which can be done in application were tested. I introduce valid data for every text field in order to complete operations. In case of success, an appropriate message is show. Also, for the login in Login View, the username and password are verified. If combination of username and password is not stored in database, the login will fail. In case of success, the type of user is tested. If the user who logged in is an employee user, an Employee View appear, and if the user is an admin, will appear

8. Bibliography

1. <http://docs.oracle.com/javase/tutorial/uiswing/>

2.<http://docs.oracle.com/javase/tutorial/jdbc/basics/index.html>