Bank Application

Student: Mureșan Marius-Sebastian

**Group: CSC I**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 4

1.3 Non-functional Requirements 4

2. Use-Case Model 5

3. System Architectural Design 5

4. UML Sequence Diagrams 7

5. Class Design 8

6. Data Model 10

7. System Testing 10

8. Bibliography 11

1. Requirements Analysis

# Assignment Specification

Design and application for a bank that has a user and administrator. Both the user and administrator have a login and both of them have different privileges.

the admin has all the privileges that the user has plus some privileges that defines the admin, such as: create a client, modify a client, create or modify an account etc..

**1. Objective**

The main objective of this application is to have a user friendly environment that can help the user check his accounts, transfer money, exchange money and create accounts. It should also help the admin manipulate the data more easily.

**2. Application Description**

The application is designed with a login frame, where the user or the admin has to login and after that judging by the status of the login another frame is opened, a user frame or an admin frame.

The admin frame has two radio buttons: CNP and ID. these buttons specify why is taken into consideration when the search takes place. When you can search a client or an account by ID or CNP. You can specify what table are you trying to search in by selecting the drop menu on the left . The buttons on the left are based on the text field in which the admin enters the ID or the CNP, they can either open another frame or have just the popup menu.

The user frame is Just designed with the clients info and some privileges that the user has like: transfer money, create accounts (only up to 5), edit login info and delete account.

**3. Application Constraints**

The application is based on a database which means that the only way the application runs is if the database created for this application is running in the background with the same credentials As when the application was developed and/or the database exists in a local server and the computer where the application runs.

The application is a maven project which means it has to download all the dependencies to run the project.

**4. Requirements**

All the requirements for this application are: download all the dependencies for the project and download the database.

**5. Deliverables**

The application takes the input that the user gives , it processes this information and it tries to return the desired output by the user, either is the information about an account or about the a client. In doing so the application should update database instantly.

# Functional Requirements

The program will create clients in a bank, each client can have up to 5 accounts and 1 username for the login account. He should be able to transfer to foreign accounts and manage money between his accounts.

The login table is divided in 2:

* Basic User – Cannot change information on other accounts (the user’s account is the only area of influence)
* Admin User – Can change information on other accounts

The user can transfer money only if he has that amount of money in his/hers account, otherwise he will get a warning message.

When a client creates an account no other client will see his credentials only the admin . The admin can have access to all client’s information and can modify . The only time when a client can see another’s client information is only when he transfers money (But only the name). the application will ask if that is the person he’s trying to send the money to.

# Non-functional Requirements

The whole system, being based on a logic layer architectural pattern, should have access to a data base from which it will extract and insert information instantly.

The layers are:

* Data
* Business
* Presentation

Usually every input from the user, any changes or money exchanges, should take place the moment the user gives confirmation on the activity. Money transfers and information updates should be instant.

2. Use-Case Model

A picture containing text, map

Description automatically generated

Fig 1. Use-Case Model

3. System Architectural Design

**3.1 Architectural Pattern Description**

this application uses the 3 layers architectural design. This design has 3 packages:

* presentation layer
  + this package handles the user interface
  + every class that has to deal with the user interface is stored in this package
* Business layer
  + this package communicate with the presentation layer as well as with the data layer
  + it takes the information from the presentation layer (the input from the user ) and give this information to the data layer, where it is processed into the database
* Data Layer
  + This layer contains all the queries that manipulate the database
  + This layer also contains the database connection

**3.2 Diagrams**

To pattern used in this application is layer architectural pattern.

this architectural pattern is divided into 3 layers.

The data layer contains all the classes that manipulate the database these classes are:

- account operations

- client operations

- login operations

these three classes, Each one of them manipulates one table:

- the account operations class manipulates did account table

- the client operations class manipulate the client table

- but the logging operations class manipulate the login table

These manipulations are in fact the: update or delete, insert or select.

The business layer contains the classes with the application’s user’s status. Such as:

- admin operations

- user operations

The user operations handles all the methods that concern only the user.

The admin operations handles all methods that concern all the users, the admin can update the info of an user. This class has access to all tables, it can update, create, delete, select in each table. It has full authority.

The presentation layer contains all the classes with the user interface, every class Helps the application user to easily see the information he wants. There is an interface for the admin and there is a interface for the user.

The interface for the admin has a table where all the information he wants to see about a client is displayed. On the left side of the panel there is a button menu from where he can update, delete, insert or select in any table he had selected on the drop menu.

After everything was in place, all 3 packages were codependent. For the presentation layer to be able to show the user the information he wants, This layer had to be connected to the business layer, which specifies which user interface frame should be opened , but the business layer cannot know if the user is an admin or not , that's why it needed to be connected to the data layer. The data layer has all the information about the user and so all the information from the user flew from the presentation layer to the business layer and then to the data layer. After the data layer procured the information about the application user, it can give the information to the business layer , and so the business layer tells the presentation layer which window it should open.

4. UML Sequence Diagrams

A close up of text on a white background

Description automatically generated

Fig 2. Login Sequence UML Diagram

5. Class Design

**5.1 Design Patterns Description**

**Structural Patterns**

These design patterns concern class and object composition. Concept of inheritance is used to compose interfaces and define ways to compose objects to obtain new functionalities.

**State Pattern**

In State pattern a class behavior changes based on its state. This type of design pattern comes under behavior pattern.

In State pattern, we create objects which represent various states and a context object whose behavior varies as its state object changes.1

**5.2 UML Class Diagram**

A close up of text on a white background

Description automatically generated

Fig 3. Class Diagram UML

6. Data Model

*A picture containing screenshot

Description automatically generated*

Fig 4. Data Model

7. System Testing

Login Test

A screenshot of a cell phone

Description automatically generated

Fig 5. Login Window

A screenshot of a cell phone

Description automatically generated

Fig 6. User Login Ok

A screenshot of a cell phone

Description automatically generated

Fig 7. Wrong Login

A screenshot of a social media post

Description automatically generated

Fig 8. Admin Login Ok

Delete Account Test

A screenshot of a cell phone

Description automatically generated

Fig 9. Delete Button

A screenshot of a social media post

Description automatically generated

Fig 10. Select Account to Delete

A screenshot of a social media post

Description automatically generated

Fig 11. Add remaining money to another available account

A screenshot of a cell phone

Description automatically generated

Fig 12. Confirmation

Transfer Money Test

A screenshot of a cell phone

Description automatically generated

Fig 13. Transfer Button

A screenshot of a cell phone

Description automatically generated

Fig 14. Insert Sender Account IBAN (ID)

A screenshot of a cell phone

Description automatically generated

Fig 15. Insert Amount to transfer

A screenshot of a cell phone

Description automatically generated

Fig 16. Insert Receiver Account IBAN (ID)

A screenshot of a cell phone

Description automatically generated

Fig 17. Confirm Client

A screenshot of a cell phone

Description automatically generated

Fig 18. Receiver doesn’t exist

A screenshot of a cell phone

Description automatically generated

Fig 19. Confirmation Message

A screenshot of a cell phone

Description automatically generated

Fig 20. Insufficient Funds Message

8. Bibliography

1 - https://www.tutorialspoint.com/design\_pattern/state\_pattern.htm