<Assignment A1>

Analysis and Design Document

Student:Oltean Mihai

**Group:30238**

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

**Availability**

Is dependable only by the server,which in this case is running on a localhost.

Scenario

Source of stimulus : Connecting to the server

Stimulus : No connection

Artifact : Application

Envicorment : The day by day user

Response : No connection

Response measure : Ture or False

Tactics : Show a message that announce the user that there is no connection to the server

**Security**

The security of the application is dependable by the user login information

Scenario

Source of stimulus : Login action

Stimulus : Invalid login

Artifact : Application

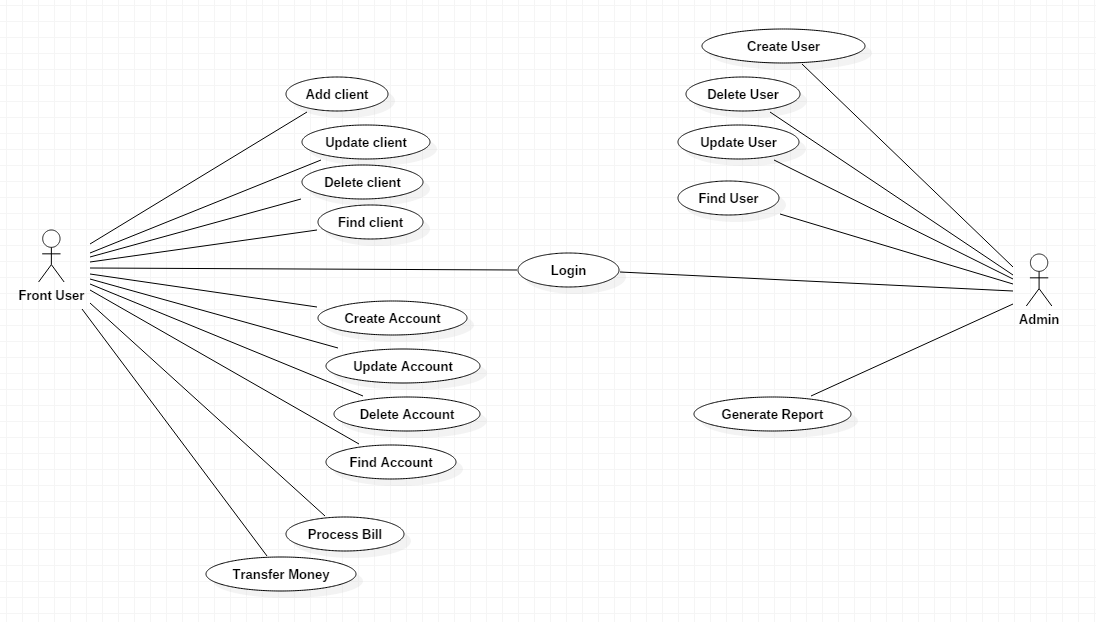
Enviroment : Admin / User

Response : Error

Response measure : instant

Tactics : Show a message that announce the user that the user id or/and the password is wrong

2. Use-Case Model



Use case: Login

Level: user-goal level

Primary actor: Front User

Main success scenario: The Front User form will pop up, and will be ready to use

Extensions: If the login credentials are wrong then a error message will pop up.

*]*

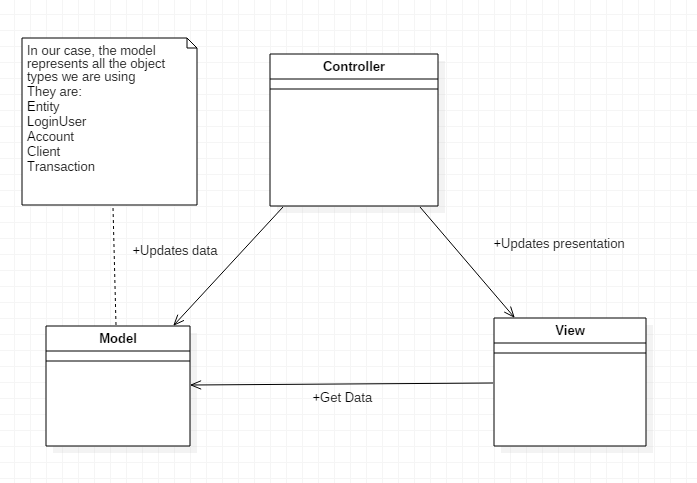
3. System Architectural Design

**3.1 Architectural Pattern Description**

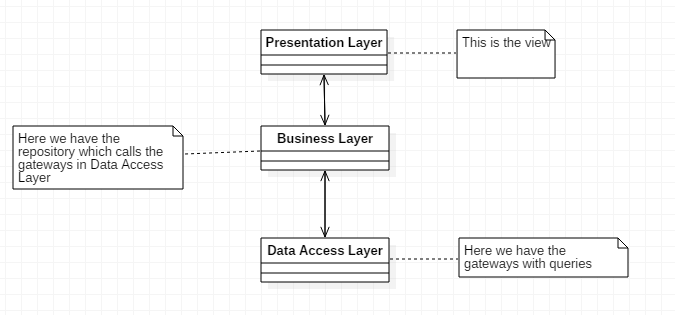
The architectural patterns used in designing this application are MVC(Model – View – Controller) and the Multitier architecture mostly known as Three – Tier Arhitecture(Presentation tier – Logic tier – Data tier).

**3.2 Diagrams**

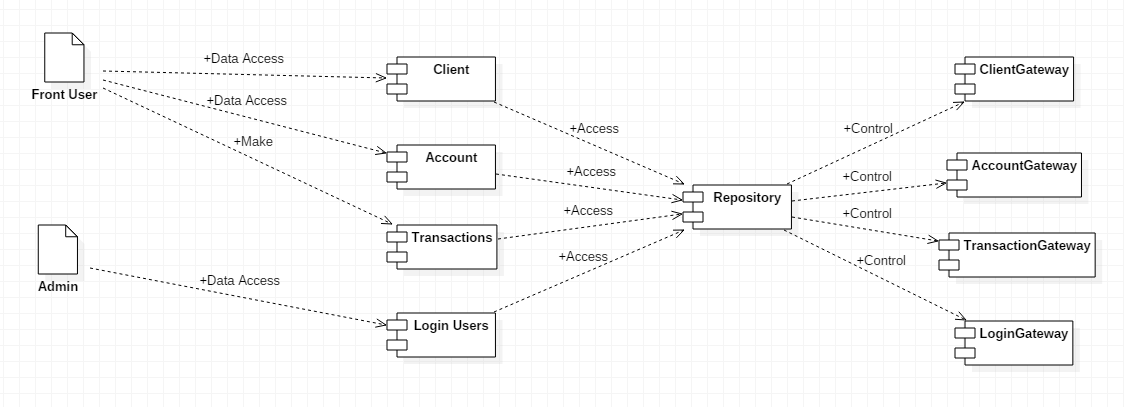
*MVC*

**

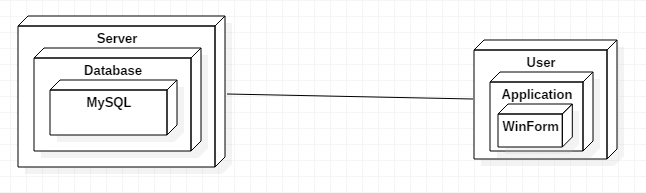
*Three – Tier Arhitecture*

****

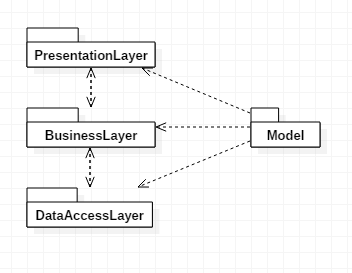
Component Diagram



**Deployment Diagram**

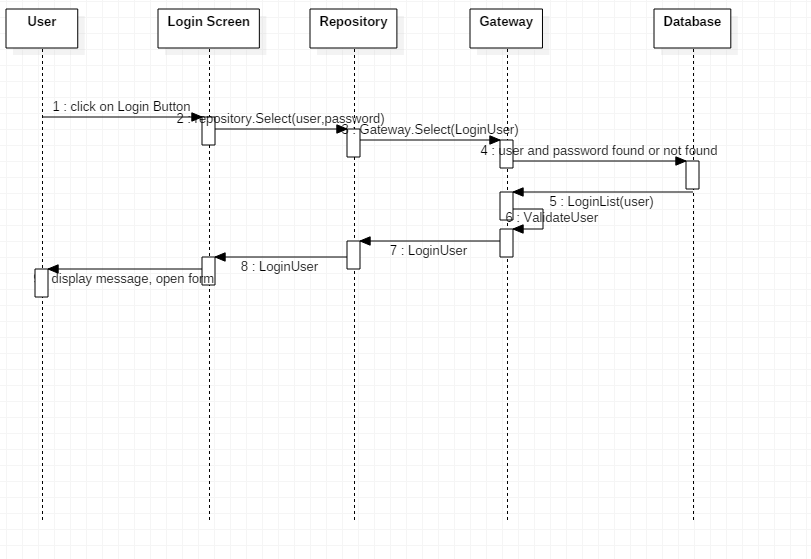


**Package Diagram**



4. UML Sequence Diagrams

**Scenario – Login Sequence**



5. Class Design

**5.1 Design Patterns Description**

The design pattern used are:

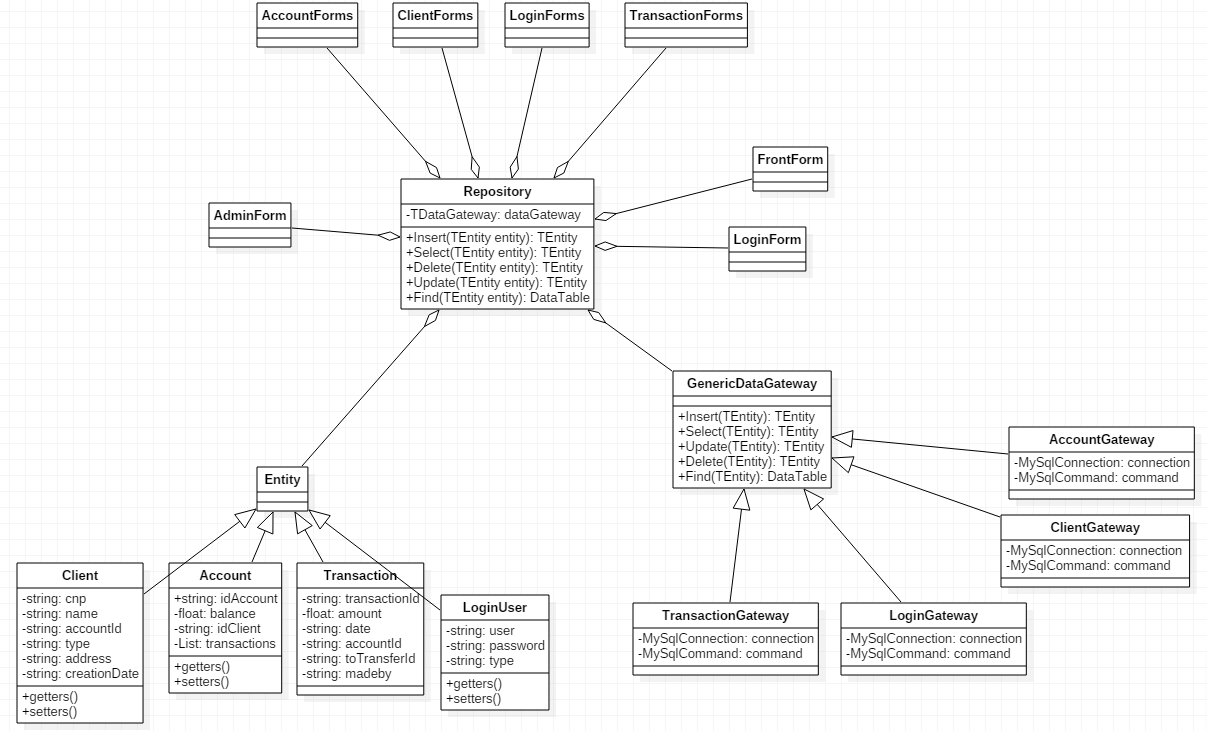
Table Data Gateway

This design pattern helps the developer to hold all the SQL for accessing a single table or view by holding all the operations (selects,inserts,updates,deletes)

Reposiotry Pattern

Helps respecting the three tier architecture by doing the business logic

**5.2 UML Class Diagram**



6. Data Model

The data model used by this application is following:

Entity – A generic model which helps implementing the Generic Repository

Client – Serves as the customers of the bank. The attributes are the same as in the database

Account – Serves as the bank account that customers have

Transaction – Operation between accounts

LoginUser – This keeps the model of the FrontEnd user or the Administrator.

7. System Testing

*[Present the used testing strategies (unit testing, integration testing, validation testing) and testing methods (data-flow, partitioning, boundary analysis, etc.).]*

8. Bibliography