Analysis and Design Document

Student: Tuluc Adelina

**Group: 30643**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

# Proiectul implementat este o aplicaţie desktop pentru angajaţii unei bănci. Aplicaţia are 2 tipuri de utilizatori: angajat sau administrator. Pentru ca aceştia să utilizeze aplicaţia trebuie să introducă numele şi parola corespunzătoare.

Un angajat poate să efectueze următoarele operaţii:

* Adăugare / modificare / vizualizare informaţii despre clienţi;
* Creare / modificare / ştergere / vizualizare informaţii despre conturi;
* Transferarea unei sume de bani ȋntre conturi;
* Procesarea de facturi;

Administratorul poate efectua următoarele operaţii:

* Operaţiile CRUD asupra informaţiilor despre angajaţi;

# Elaboration – Iteration 1.1

# Domain Model

*[Define the domain model and create the conceptual class diagrams]*

Proiectul este structurat, folosind pattern-ul arhitectural Layers. Astfel, am creat 4 pachete, şi anume: BusinessLogic, DataLayer, Model şi Presentation. Pachetul BusinessLogic, care se ocupă de funcţionalitatea aplicaţiei, conţine clasa TransferTransactionScript si AccountBusinessLogic, ClientBusinessLogic, EmloyeeBusinessLogic. Pachetul DataLayer, care se ocupă de lucrul cu baza de date (conectare şi interogări), conţine clasele AccountGateway, ClientGateway şi EmployeeGateway. Pachetul Model conţine clasele Account, Client şi Employee. Pachetul Presentation, care reprezintă de fapt interfaţa grafică a aplicaţiei, conţine clasele AccountFrame, AdminFrame, Decision, EmployeeFrame şi Login.

# Architectural Design

## Conceptual Architecture

*[Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]*

Am folosit următoarele pattern-uri: TransactionScript şi Table Data Gateway. Ȋn clasa TransferTransactionScript am implementat operaţiile care se pot executa asupra sumei de bani dintr-un anumit cont. Pentru fiecare tabela din baza de date (am implementat 3 tabele: Client, Account şi Employee) am implementat câte o clasă care conţine operaţiile care se pot efectua, utilizând baza de date.

## Package Design

*[Create a package diagram]*

Am creat aceasta diagram de pachete mai jos care arata si interactiunile dintre clase si flow-ul aplicatiei.

## Component and Deployment Diagrams

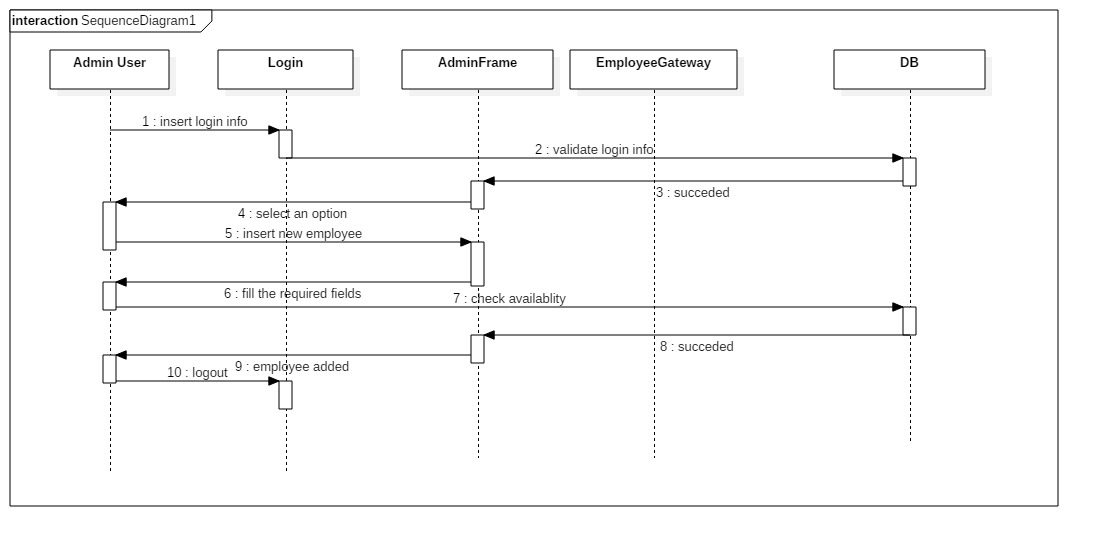
*[Create the component and deployment diagrams.]*

# Elaboration – Iteration 1.2

# Design Model

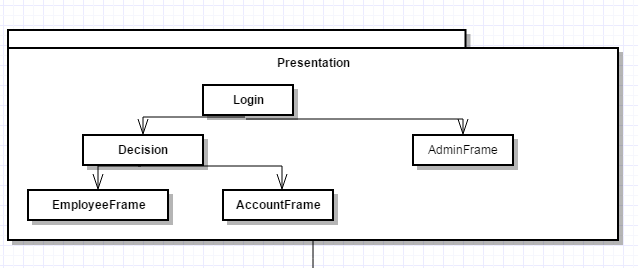
## Dynamic Behavior

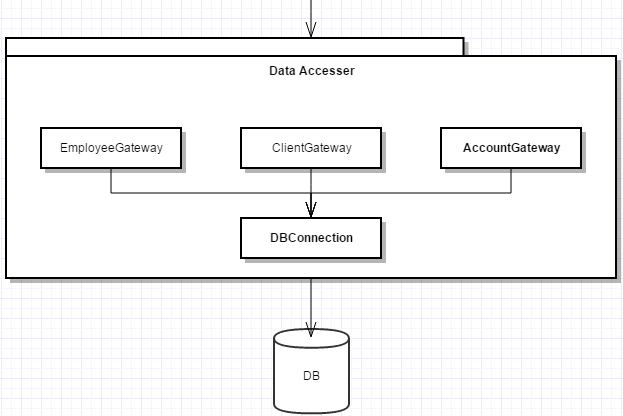
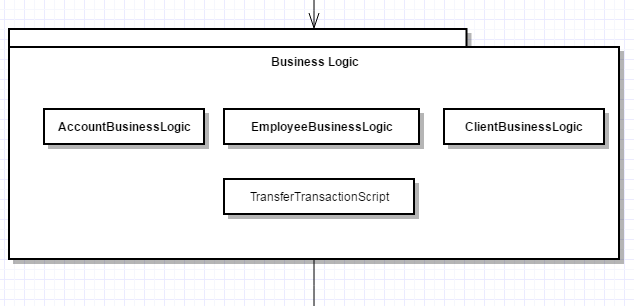
*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

**

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*





# Data Model

*[Create the data model for the system.]*

Pentru realizarea bazei de date, am utilizat tool-ul MySQL Workbench. Am creat 4 tabele, şi anume: client, account, employee şi operations. Tabela client este folosită pentru a reţine informaţiile despre clienţi:id, nume, id card, cnp, adresă şi parolă. Tabela account este folosită pentru reţinerea informaţiilor despre un cont: id, tipul contului, suma de bani şi data la care s-a creat contul. Tabela employee reţine datele despre angajaţi: nume şi parolă.

**

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

Arhitectura folosită pentru implementarea aplicaţiei este Layers. La capitolul 2.1 am eplicat ȋn detaliu cum a fost structurat proiectul.

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

Creator. Pachetul de dataAccesser nu stie de modelul folosit de catre aplicatie, paramentrii lui fiind transmisi fiecare nu ca de tip obiect. La fel pachetul de presenation nu e constient de cum sunt salvate in baza de date. De aici rezulta incapsularea si reusability. Daca s-ar crea si intefete intre aceste pachete atunci se vor putea implementa diferite tratari si metode in functie de baza de date sau de tabelele din baza de date, sau de afisarea lor in partea de prezentare.

# Construction and Transition

# System Testing

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

Bineȋnţeles, această aplicaţie poate fi ȋmbunătăţită ȋn multe moduri. Una dintre ȋmbunătăţirile posibile constă ȋn implementarea interfeţei şi pentru clienţii băncii.

# Bibliography

<http://stackoverflow.com/>

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

<http://martinfowler.com/eaaCatalog/tableDataGateway.html>