Bank

Student:Ursachi Ionut Gabriel

**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

Acest assignment consta in proiectarea si implementarea unei aplicatii care simuleaza o banca. Aplicatia trebuie sa aiba doua tipuri de utilizatori: angajati si administrator, pentru care vor fi afisate diferite interfete si care vor putea realiza diferite operatii. Ambele categorii de utilizatori trebuie sa introduca un username si o parola pentru a putea folosi aplicatia.

Aplicatia are doua componente importante: baza de date, unde stocam informatiile despre conturi, utilizatori, tranzactii si aplicatia, care face conexiunea cu baza de date si ofera metode de a acesa informatiile din aceasta.

# Functional Requirements

Pentru regular-useri aplicatia ofera urmatoarele operatii:

-afisarea tuturor datelor clientilor existenti, cum ar fi: nume, cont, suma de bani din cont, transferurile intre conturi;

-modificarea datelor clientilor existenti: adaugare clienti, modificare date clienti, trensfer de bani intre conturi, sau chiar plata anumitor facturi pentru clienti;

Pentru administrator aplicatia ofera control asupra regula-userilor(angajati). Acestia pot sa adauge un angajat sa modifice informatiile din baza de date pentru angajati sa stearga angajati, sau sa afiseze o lista cu toti angajatii. Acestia mai pot creea rapoarte cu privire la activitatile unui angajat pentru o anumita perioada de timp.

# Non-functional Requirements

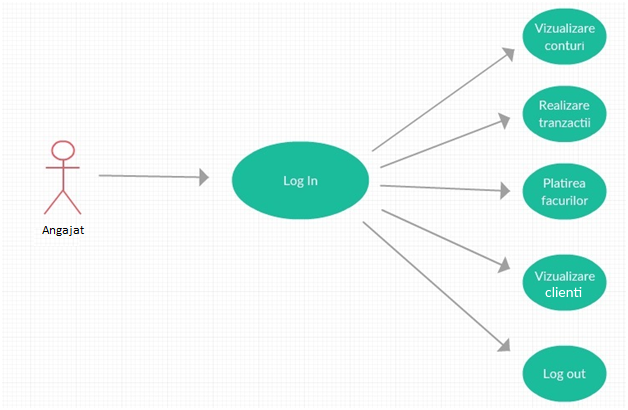
- Availability: sistemul este disponibil pentru orice calculator ce ruleaza aplicatia si se conecteaza cu succes la baza de date aferenta.

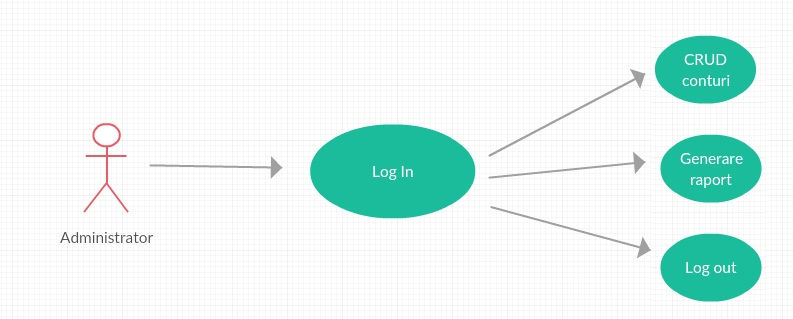
- Usability: datarita sistemului swing aplicatia are o interfata usor de utilizat.

- Response time: In mod normal orice sistem poate oferi un timp de raspuns foarte mic.

- Security: pentru conectarea la baza de date, indiferent de tipul de utilizator este necesara un user si o parola.

2. Use-Case Model





3. System Architectural Design

**3.1 Architectural Pattern Description**

Pentru aplicatia Bank am folosit Architectural Layer Pattern.

Am ales aceasta arhitectura deoarece consider ca ma va ajuta sa implementez mai usor operatiile din cadrul aplicatiei. Aplicatia va fi structurata in 4 mari pachete:

-bussinesLayer

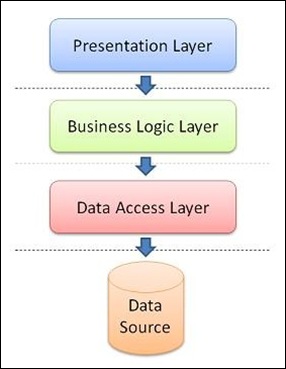
-dataAccesLayer

-modelLayer

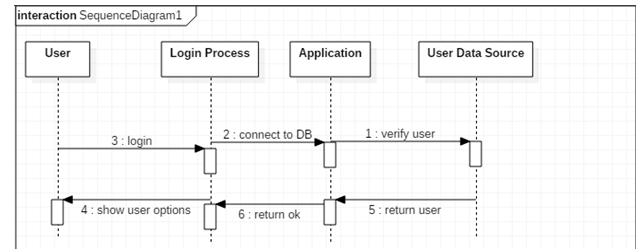
-presentationLayer

**3.2 Diagrams**

Diagrama sistemului:



4. UML Sequence Diagrams



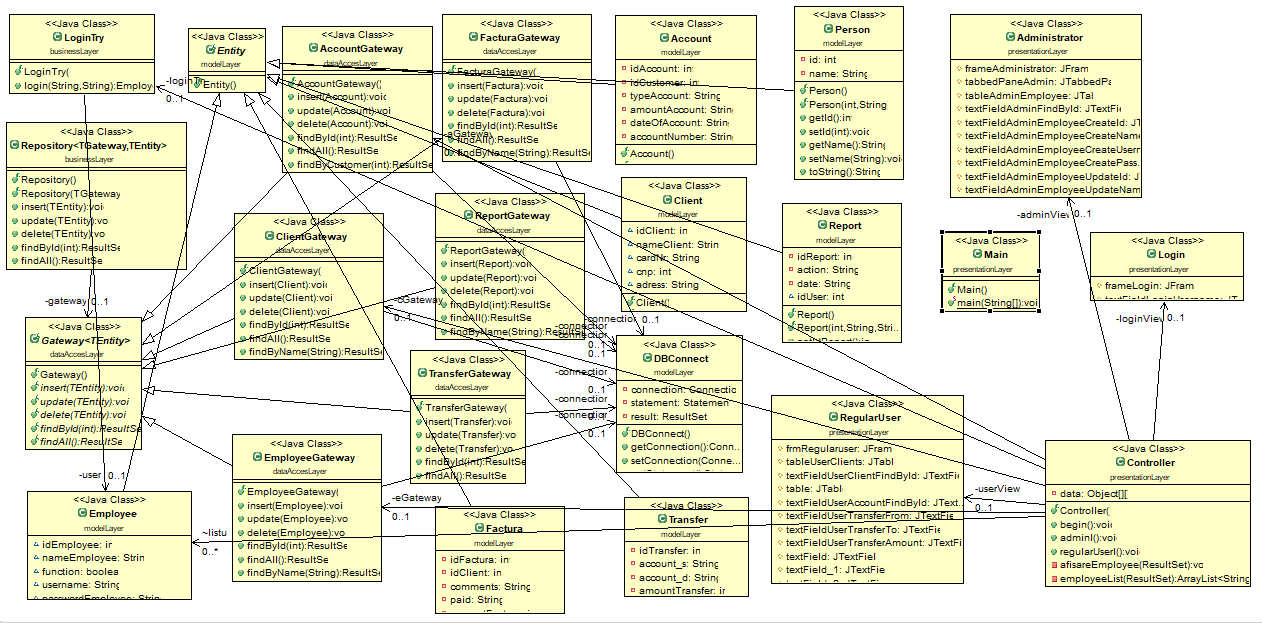
5. Class Design

**5.1 Design Patterns Description**

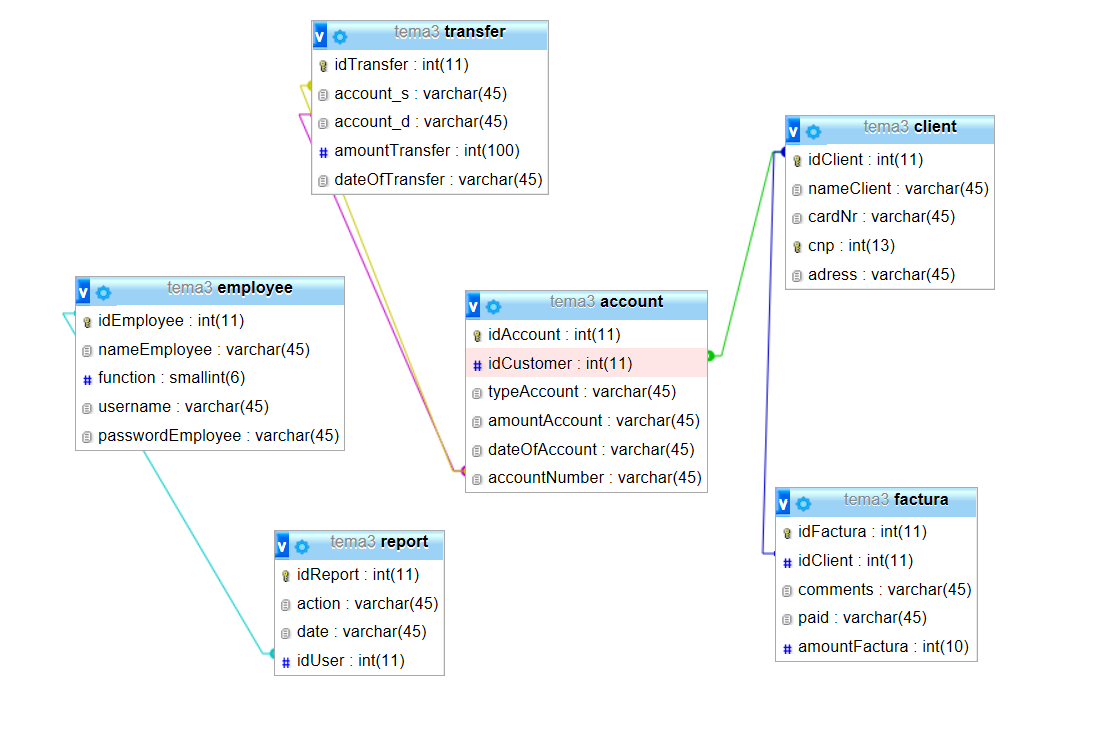
Arhitectura Layers ajuta la structurarea aplicatiilor care pot fi impartite in grupuri mai mici de task-uri, fiecare dintre ele avand un nivel diferit de abstractizare.

Operatiile high-level se bazeaza pe cele low-level, deci avem de-a face cu o structura vertical a dependetelor din cadrul aplicatiei, iar accesul se face doar intre layerele vecine. Nu se poate face o legatura directa daca distanta dintre doua layere este mai mare de 1.

**5.2 UML Class Diagram**



6. Data Model



7. System Testing

Ca si mod de testare am folosit interfata.

8. Bibliography

<https://www.youtube.com/watch?v=6cNYUc2PIag>

<https://www.youtube.com/watch?v=hDIXPbAODmE>