<Your Books Everywhere>

Student:Bercea Ionut-Alexandru

**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

Aplicatia este proiectata in limbajul Java. Aceasta are doua tipuri de utilizatori (clientul si adminul) care trebuie sa se autentifice folosind un cont si o parola.

Clientul poate efectua urmatoarele :

* Creare cont nou
* Achitare comanda
* Autentificare
* Imprumutare/Returnare carte

Adminul paote efectua urmatoarele:

* Filtrare carti dupa: data lansarii, autor, titlu si gen.

# Functional Requirements

Datele vor fi stocate intr-o baza de date, iar pentru gruparea claselor am folosit un modelul de design Layerd architecture.

# Non-functional Requirements

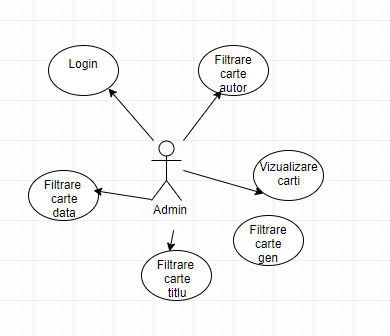
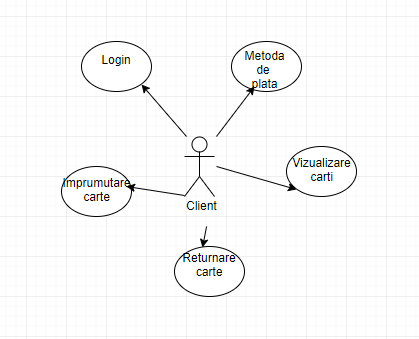
Aplicatia este una desktop, iar aceasta nu are nevoie de conexiune la internet pentru a comunica cu baza de date, aceasta fiind locala.

Performanta: Aplicatia neavand nevoie de resurse crescute pentru a fi executata are un timp redus de raspuns la comenzi.

Mentenanta: Aplicatia este foarte usor de intretinut fiind structurata bine, accesul la componente fiind rapid.

Securitate: Aplicatia foloseste un system de autentificare astfel doar persoanele care se afla in baza de date o pot utiliza.

2. Use-Case Model



3. System Architectural Design

**3.1 Architectural Pattern Description**

Arhitectura pe straturi are ca scop gruparea functionalitatilor aplicatiei in diferite straturi, care sunt asamblate unul peste calalalt. Functionalitatea fiecarui nivel este legata de un rol sau responsabilitate comuna . Comunicarea intre straturi este conectata explicit. Aceasta arhitectura ajuta la separarea componentelor aplicatiei ajutand la o mai buna flexibilitate si mentenabilitate.

Stilul architectural stratificat a fost descries ca o piramida inversata de reutilizare, in fiecare strat agrega responsabilitatile si abstractizarile stratului direct sub el.



**3.2 Diagrams**

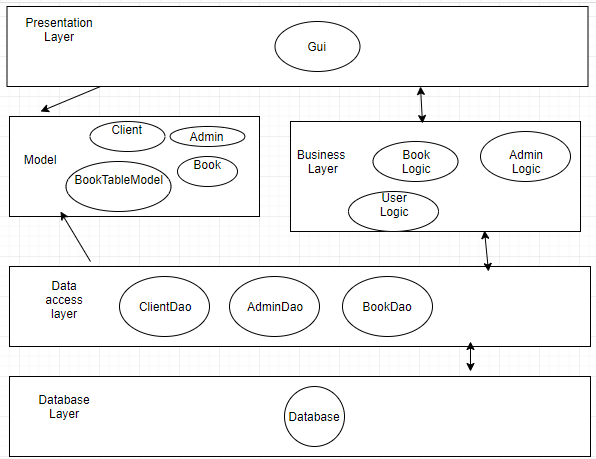
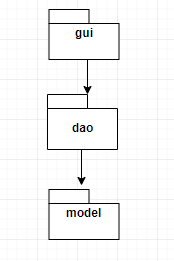
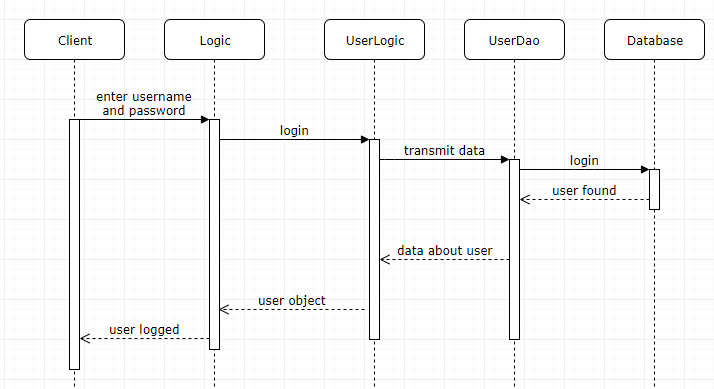


Diagrama de pachete:



4. UML Sequence Diagrams



5. Class Design

**5.1 Design Patterns Description**

**5.2 UML Class Diagram**

Diagrama de clase pentru pachetul dao:

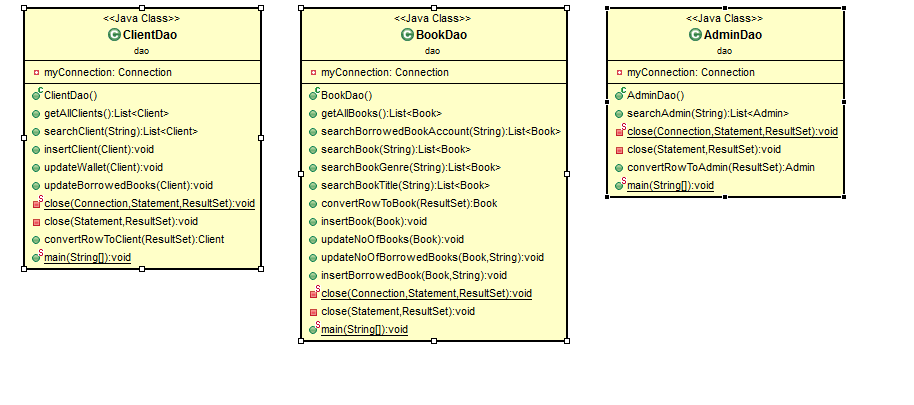


Diagrama de clase pentru pachetul model:

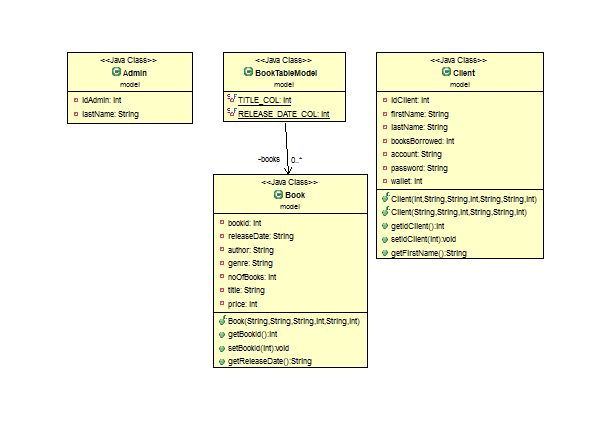
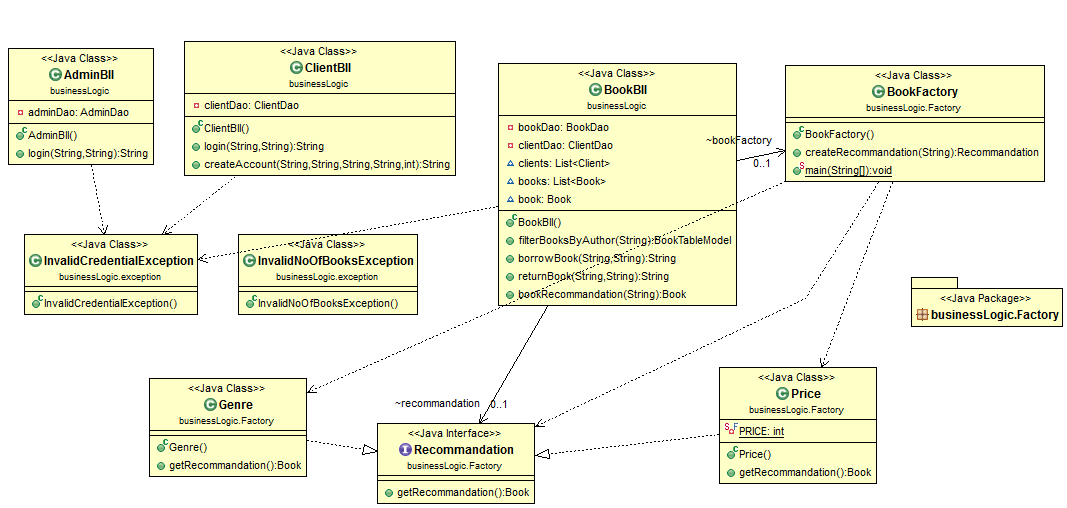


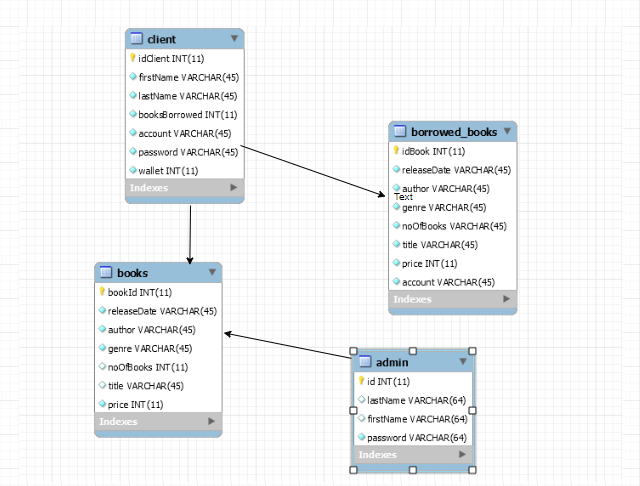
Diagrama Business Logic



6. Data Model

Clasele Dao sunt responsabile pentru conexiunea dintre baza de date si pentru accesul la datele din baza de date.

Clasele din BLL sunt responsabile pentru partea de logica a aplicatiei, modul in care datele preluate din interfata sunt apoi transmise mai departe pana la baza de date unde se preiau datele necesare. Pentru fiecare operatiune a utilizatorilor, adminilor respective cartilor avem clase dedicate pentru aceaste.



7. System Testing

Pentru testarea sistemului am creat mai multe teste care executa anumite parti din aplicatie pentru a-i verifica buna functionalitate, astfel rezultatele produse de teste sunt comparate cu cele corecte.

8. Bibliography

Informare si intelegere:

* https://www.tutorialspoint.com
* <https://en.wikipedia.org>
* <https://www.w3schools.com/>

Diagramele din documentatie au fost create cu ajutorul:

* https://www.draw.io/