<Your Books Everywhere>

Analysis and Design Document

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 una web, 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 poate efectua urmatoarele:

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

Clientii care doresc o anumita carte, iar aceasta nu este in stoc vor si introdusi intr-o lista de asteptare, iar in momentul in care cartea revine in stoc vor fi notificati pe email . Se va folosi design patternul Observer pentru notificarea clientilor.

# Functional Requirements

Datele vor fi stocate intr-o baza de date. Se va folosi pattern-ul architectural MVC la proiectarea si organizarea aplicatiei. Parolele vor fi criptate si salvate in baza de date.

# Non-functional Requirements

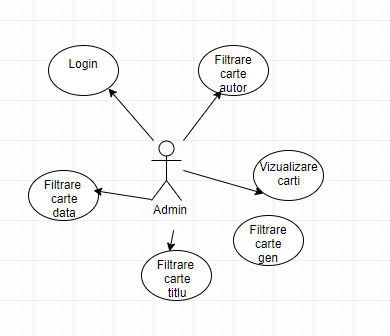
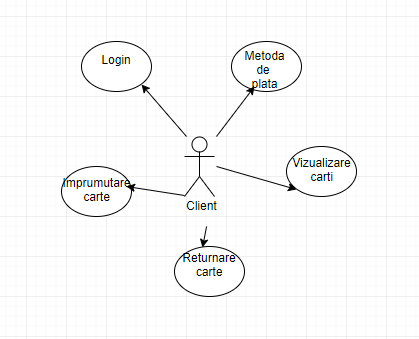
Aplicatia este una web, iar aceasta nu are nevoie de conexiune la internet pentru a comunica cu baza de date, aceasta fiind locala.

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.

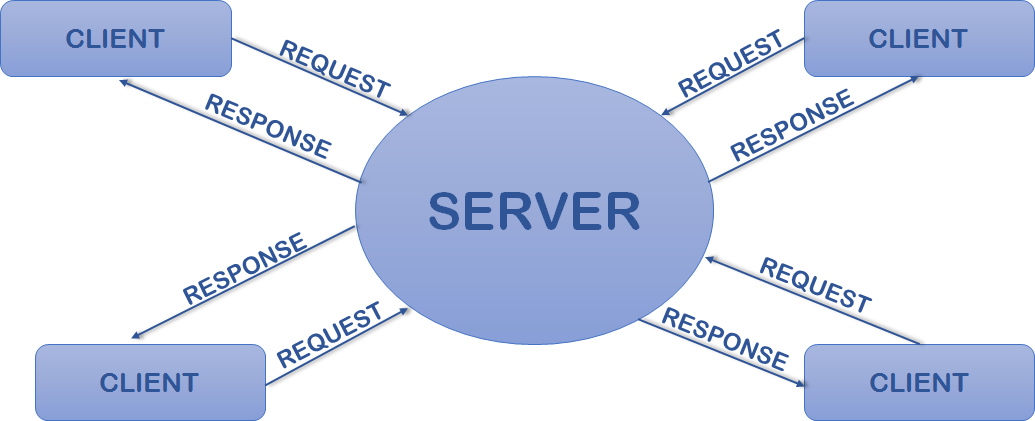
2. Use-Case Model



3. System Architectural Design

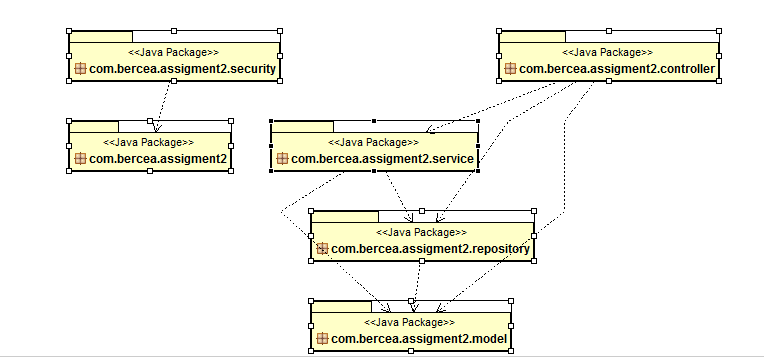
**3.1 Client-Server Architecture**

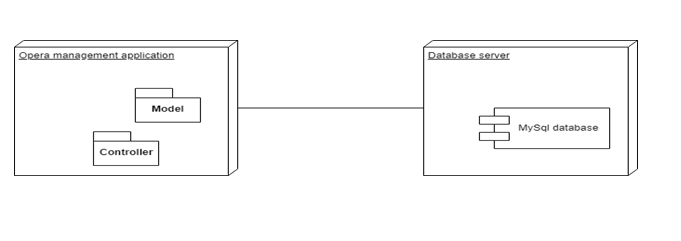
Arhitectura client-server este o arhitectura producator-consumator unde serverul se comporta ca un producator si clientul ca un consumator. Clientul trimite resurse sau procese (request-uri) catre server printr-o conexiune la internet, aceasta este procesata si trimisa inapoi catre client. Un server poate lucre cu mai multi clienti simultan. Internetul se bazeaza si el pe o arhitectura client-server unde serverele web servesc mai multor client in acelasi timp ce folosesc site-urile web.



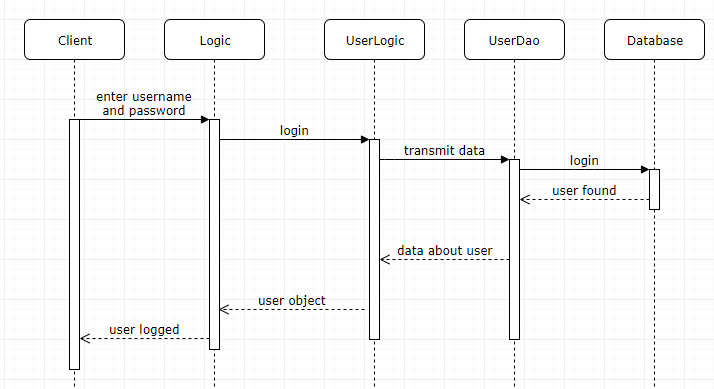
**3.2 Diagrams**

**Diagrama de pachete:**





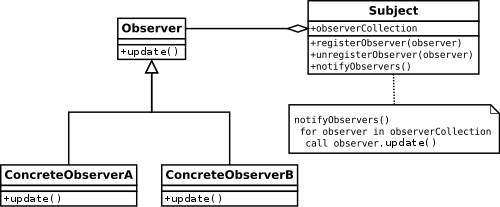
4. UML Sequence Diagrams



5. Class Design

**5.1 Design Patterns Description**

**Observer pattern este un design pattern in care un obiect numit subiect, mentine o lista de observatori pe care ii notifica automat pentru fiecare schimbare care se produce.**



**5.2 UML Class Diagram**

Diagrama de clase pentru pachetul service

:

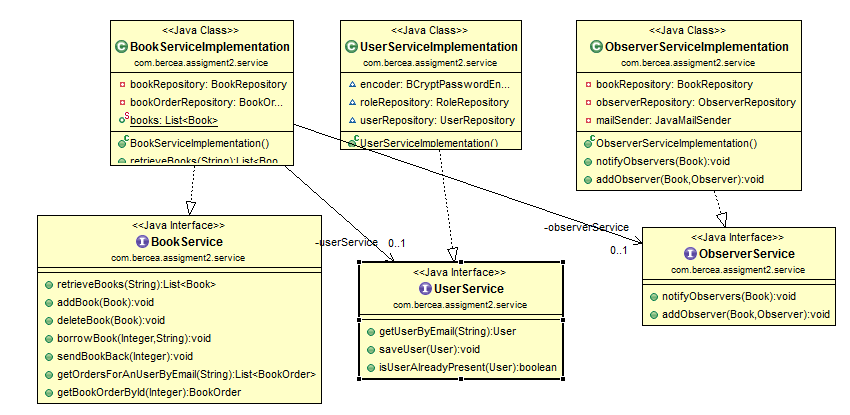


Diagrama de clase pentru pachetul model:

.

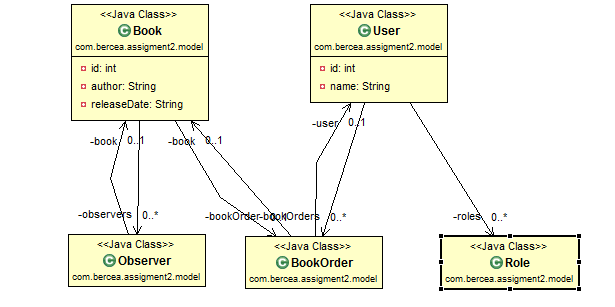
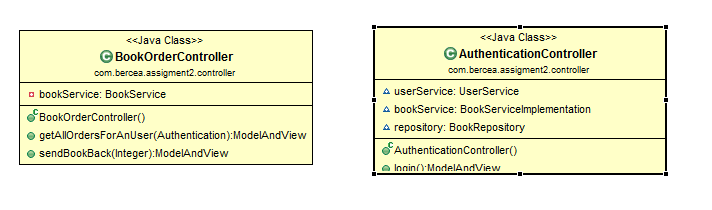


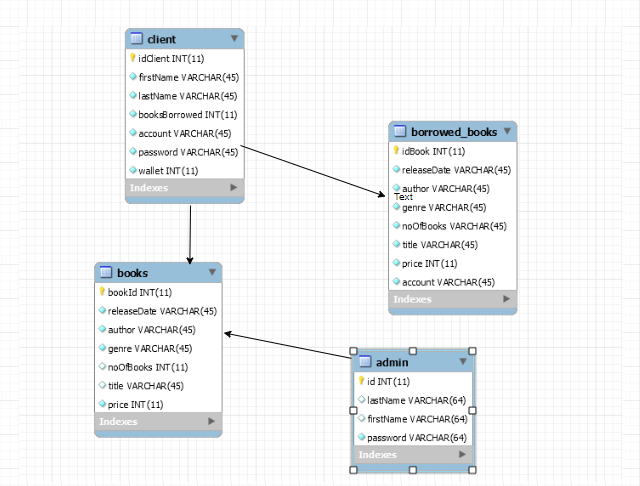
Diagrama Controller



6. Data Model

Clasele din pachetul service sunt responsabile pentru interogarile care au loc cu baza de date.

Clasele din Controller 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/