<Assignment 2 - Bookstore>

Analysis and Design Document

Student:Cozma Andreea

**Group:30233**

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

La acest proiect se urmareste intelegerea pattern-ului architectural Model-View-Controller si a pattern-ului de design Factory.

# Functional Requirements

Aplicatia va avea doua tipuri de utilizatori, unul reprezinta angajatul unui magazin care vinde carti, iar celalat este administratorul. Cei doi vor putea avea acces la aplicatie cu daca se logheaza cu un username si o parola.

Cerintele pentru angajat sunt:

* Cautarea cartiilor dupa gen, titlu, autor
* Vinderea cartilor

Cerintele pentru utilizatorul de tip administrator sunt:

* Operatii CRUD asupra cartilor (acestea vor avea informatii despre titlu,autor,genul cartii, cantitatea si pretul)
* Operatii CRUD pe informatiile legate de angajati
* Generarea a doua tipuri de rapoarte, unul intr-un fisier de tip PDF si unul CSV care sa contina cartile care nu mai sunt pe stoc

# Non-functional Requirements

Datele vor fi stocate in fisiere XML ( am ales doua fisiere, unul pentru utilizatori, altul pentru carti si cu vanzarile aferente lor).

Se va folosi pattern-ul Factory pentru s spori flexibilitatea si pentru o usoara mentenanta a aplicatiei. In cazul in care se doreste adaugarea unei noi functionalitati legata de generarea rapoartelor, se va putea adauga usor un nou tip de raport pe princiul Open-Closed Principle, deoarece nu va necesita nicio modificare, ci doar extindere.

Datele vor fi validate inainte ca ele sa fie stocate persistent.

Tot din considerente de flexibilitate, datorita necesitatii filtrarii dupa diferite criterii, s-a folosit pattern-ul de design Strategy care contine o clasa abstracta si mai multe implementari specific in functie de metoda de filtrare posibila. Contextul in aceasta situatie este reprezentat de o clasa ce contine o referinta de tip Strategy.

2. Use-Case Model

Use case: vanzarea unei carti

Level: user-goal level

Primary actor: angajatul librariei

Main success scenario: exista un numar sufficient de carti pentru a se putea realiza vanzarea (astfel se scade stocul existent)

Extensions: s-ar putea extinde prin generarea unei facturi cu pretul aferent

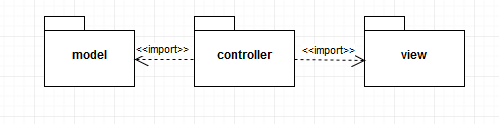
3. System Architectural Design

**3.1 Architectural Pattern Description**

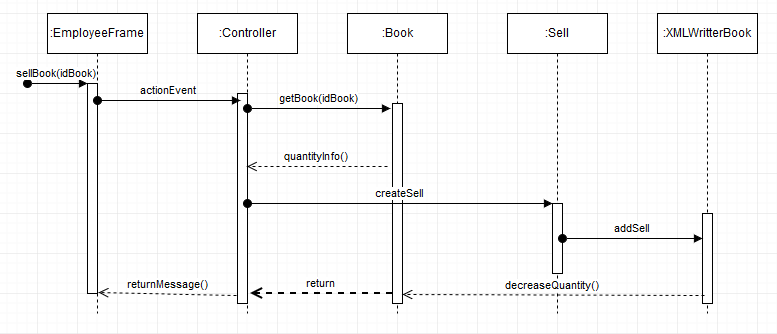
Pattern-ul architectural folosit este Model-View-Controller, cel din urma decupland modelul de interfata si trateaza evenimentele aparute.

**3.2 Diagrams**

Arhitecture Diagram



4. UML Sequence Diagrams

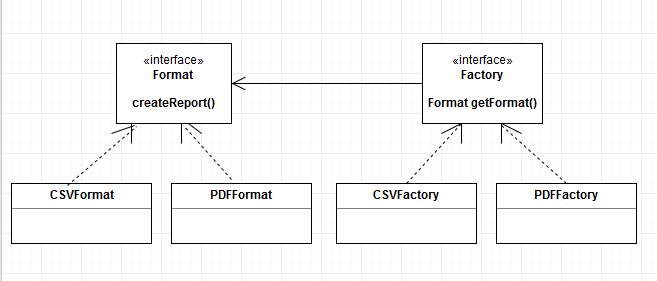


5. Class Design

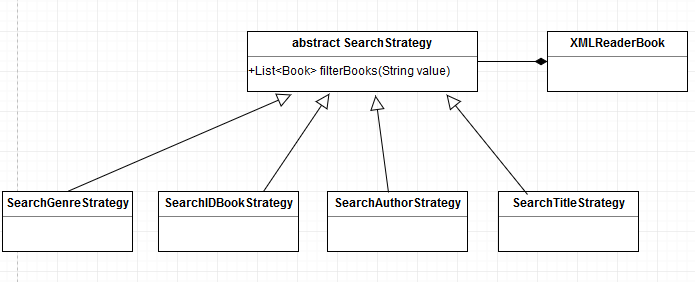
**5.1 Design Patterns Description**

S-au ales urmatoarele doua pattern-uri de design:

Factory Pattern

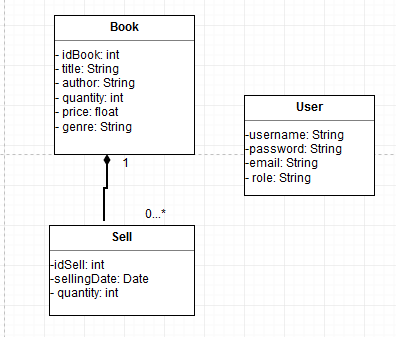
**

Strategy Pattern



6. Data Model

Datele sunt stocate in doua fisiere XML, “Book.xml” pentru cartile cu vanzarile aferente, iar “User.xml” este folosit pentru a retine utilizatorii. Modelul de date este surprins in diagrama urmatoare si indica relatiia care exista, o carte poate avea mai mutle referinte de vanzari.



7. System Testing

Fiecare functionalitate nou integrata a fost testate dupa finalizarea acesteia.

8. Bibliography

* <https://www.w3schools.com/xml/xpath_nodes.asp>
* <https://docs.oracle.com/javase/tutorial/jaxp/xslt/xpath.html>
* <https://www.tutorialspoint.com/java_xml/java_xpath_parse_document.htm>