Bookstore

Analysis and Design Document

Student: Coman Vasile

**Group: 30231**

Table of Content

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 Chira2018

# Assignment Specification

Use Java/C# API to design and implement an application for the employees of a book store. The application should have two types of users (a regular user represented by the book store employee and an administrator user) which have to provide a username and a password in order to use the application.

# Functional Requirements

*The regular user can perform the following operations:*

*- Search books by genre, title, author, or any combination of multiple filters.*

*- Sell books.*

*The administrator can perform the following operations:*

*- CRUD on books (book information: title, author, genre, quantity, and price).*

*- CRUD on regular users’ information.*

*- Generate two types of reports files, one in txt format and one in csv format or other, with the books out of stock.*

# Non-functional Requirements

2. Use-Case Model

*Use case: Create user*

*Level: user-goal level*

*Primary actor: admin*

*Main success scenario: - after the authentication is successfully the admin can add/view all/update/delete employees*

*- after he completing the fields he can add a new employees as admin or regular user*

*- if there are no error messages the new user will be added to database*

*Extensions: then he can choose other operations like see all employees, add books or generate some reports*

*Use case: Process sale*

*Level: user-goal level*

*Primary actor: regular user*

*Main success scenario: - after the authentication is successfully the user can search for a book*

*-he will choose the book and the quantity and will process the sale;*

*-if the operation it successful the sale will appear in sales table*

*Extensions: he can also see all the incomings*

*Use case: Generate reports*

*Level: user-goal level*

*Primary actor: admin*

*Main success scenario: - after the authentication is successfully the admin can add/view all/update/delete employees and books or generate reports*

*- the admin can choose type of report CSV or TXT*

*- a CSV file or TXT will be generated with all the books out of stock*

*Extensions: then he can choose other operations like see all employees or generate another report*

3. System Architectural Design

**3.1 Architectural Pattern Description**

*In this project we have 2 architectural design patterns:*

*-3-tier architecture and MVC architecture (model-view-controller).*

*A 3-tier architecture is a type of software architecture which is composed of three “tiers” or “layers” of logical computing.*

*Presentation Tier- The presentation tier is the frontend layer in the 3-tier system and consists of the user interface.*

*Application Tier- The application tier contains the functional business logic which drives an application’s core capabilities.*

*Data Tier- The data tier comprises of the database/data storage system and data access layer.*

*Model–View–Controller (usually known as MVC) is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts.*

*Model*

*The central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application.*

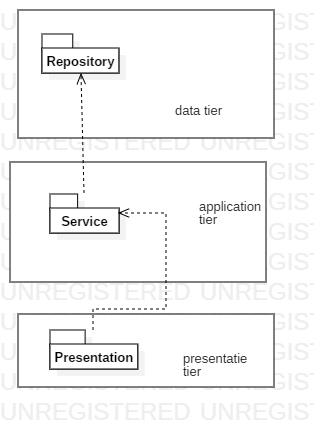
*View*

*Any representation of information such as a chart, diagram or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.*

*Controller*

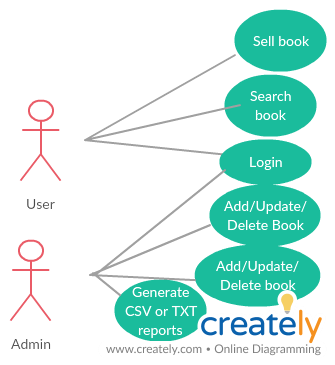
*Accepts input and converts it to commands for the model or view.*

**3.2 Diagrams**





4. UML Sequence Diagrams

**

5. Class Design

**5.1 Design Patterns Description**

*We used Factory Method design patter for generate the reports.*

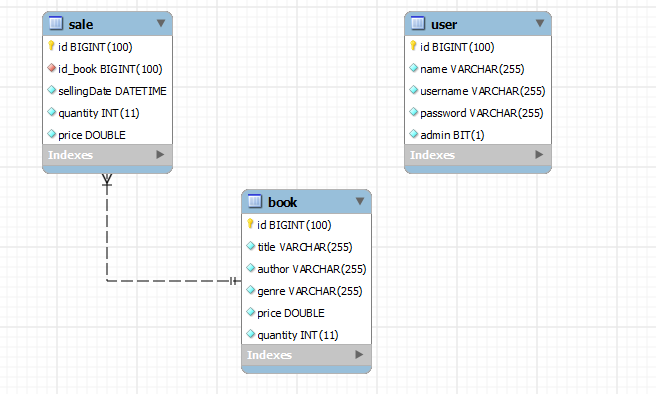
*In class-based programming, the factory method pattern is a creational pattern that uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created. This is done by creating objects by calling a factory method—either specified in an interface and implemented by child classes, or implemented in a base class and optionally overridden by derived classes—rather than by calling a constructor.*

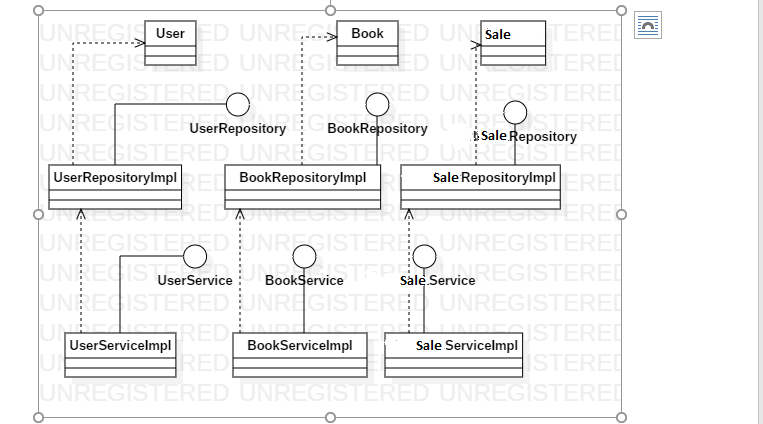
*And we used Observer for generating a message dialog when a sale is made*

*The observer pattern is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods.*

**5.2 UML Class Diagram**







6. Data Model

*Class User contains information about the employees of the bookstore. They can be admin or regular user. They have a username and password so they can log in to application.*

*Class Book contains information about the book like title, genre, author, price.*

*Class Sale contains information about a sold book like selling date, quantity and total price.*

7. System Testing

*System was tested using the graphic interface by entry different sets of data.*

8. Bibliography

*Labs*

[*https://stackoverflow.com*](https://stackoverflow.com)

[*https://www.w3schools.com*](https://www.w3schools.com)

[*https://docs.oracle.com*](https://docs.oracle.com)