Your Books Everywhere

Student: Borsa Iulian-Laurentiu

**Group: 30238**

Table of Contents

Your Books Everywhere 1

Analysis and Design Document 1

Student: Borsa Iulian-Laurentiu 1

Table of Contents 2

1. Requirements Analysis 4

1.1 Assignment Specification 4

1.2 Functional Requirements 4

1.3 Non-functional Requirements 4

2. Use-Case Model 4

3. System Architectural Design 5

4. UML Sequence Diagrams 7

5. Class Design 7

6. Data Model 9

9

7. System Testing 9

8. Bibliography 9

1. Requirements Analysis

# Assignment Specification

Design and implementation of an application for a book management service using OOP language Java. It will represent a library management with two types of users: normal user and staff user.

A user should be able to create an account, choose a payment plan and login to search the book library. Payments can be done via a cash only policy and need to be validated by library staff. The library is managed by staff.

If a book is available a user can add it to library. If not the user can join a waiting list. Once a book has been read by a user it can be returned via the online library return function.

The service also provides users with dynamic recommendations based on latest trends (popular borrowed books) or user defined interests by genre or topic.

# Functional Requirements

* The data will be stored in a relationship database.
* The application should have layered architecture
* A part of inputs will be validated against invalid data
* It contains a factory method to send recommendations
* Payment can be done via a cash only policy and need to be validated by staff
* Deals are managed by staff

# Non-functional Requirements

* Accessibility – the application is for all kind of users
* Security and privacy – the application is just for registered users
* Ease of development – The application is not complex to implement

2. Use-Case Model

Use case goal: return a book to library

Primary actor: an actor named “user”

Main success scenario:

1. User hasn’t got an account, so User is creating a new account

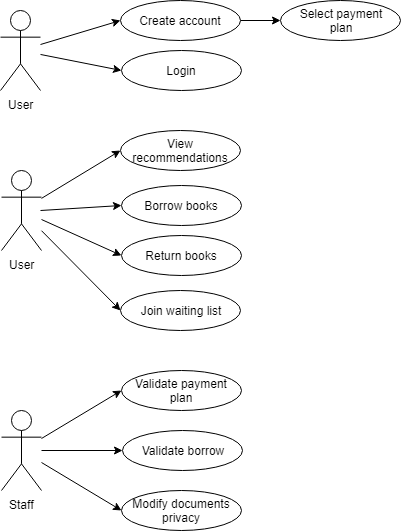
2. User is waiting to be validated by Staff

3. User login

4. User choose a book which is available

5. User borrow the book

6. User read the book and return it



3. System Architectural Design

**3.1 Architectural Pattern Description**

delving into the details of interface specification. I chose a layered architecture pattern because it is easy to testing components, it’s a common pattern used for general desktop applications.

There are 3 big layers:

* **Presentation Layer** doesn’t need to know or worry about how to get customer data

it is used just to display information on a Graphical User Interface

* **Business Logic Layer** doesn’t need to be concerned about how to format customer data for display on a screen or even where the customer data is coming from

it is used just to get the data from the persistence layer, perform business logic against the data and pass that information up to the presentation layer

* **Persistence Layer** communicates with Database

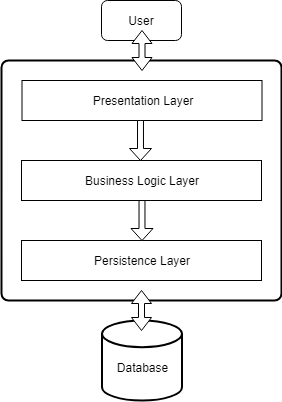


Figure 3.1 - Architectural Pattern

* 1. **Diagrams**

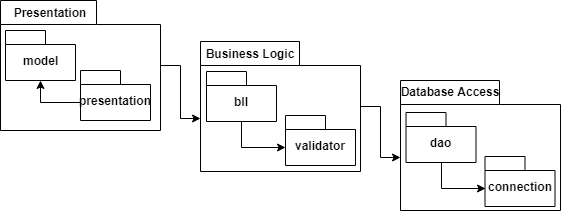
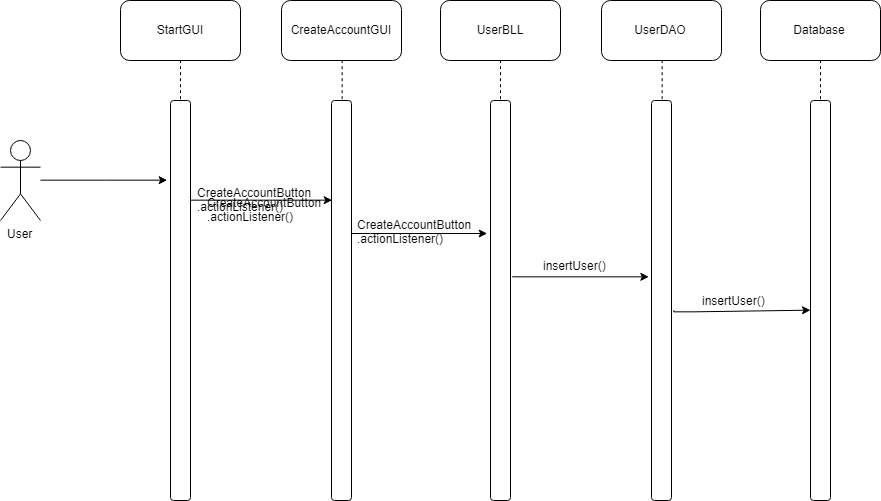


Figure 3.2 - Package Diagram

4. UML Sequence Diagrams



5. Class Design

**5.1 Design Patterns Description**

Factory pattern is one of the used patterns, specifically into the persistence layer. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object. In Factory pattern, we create object without exposing the creation logic to the client.

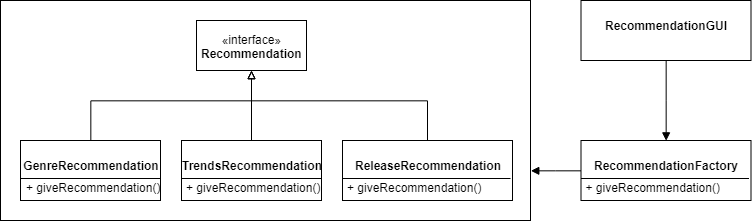


Figure 5.1 – Factory Design Pattern

**5.2 UML Class Diagram**

Bellow is a fragment from Class Diagram, just for User.

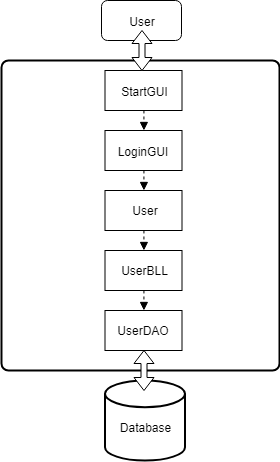
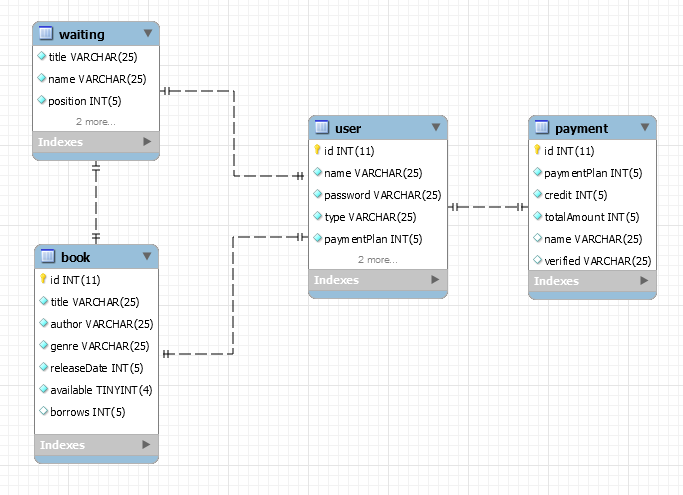


Figure 5.2 – Fragment from Class Diagram

6. Data Model



7. System Testing

I’m using **Graphical User Interface Testing**, who is the process of testing a product graphical interface. This is done through the use of a set of test case scenarios. To generate a set of test cases, I will try to cover all the functionality of the system and verify entire GUI system, first as normal user, and than as guest.

8. Bibliography

<http://www.bredemeyer.com/ArchitectingProcess/ConceptualArchitecture.htm>

<https://www.tutorialspoint.com/design_pattern/factory_pattern.htm>