<Your Books Everywhere!>

Analysis and Design Document

Student:Luca-Dan Adrian

**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

In order to implement a book management service I used JAVA objected oriented programming language. The app has two kinds of users: one is the client who can find, lend and return books, and the other one is the staff (or the librarian) who is able to manage the library.

A user is able to create an account, choose a payment plan and log in to search for books in the library. The client can also borrow books (if the books are available and if not he will join a waiting list).

The librarian or the staff can add, delete or update the books from the library and also can search for them by filtering. The books are filtered by the release date, author or title.

# Functional Requirements

The information about books, staff and users are saved in MySql database. The project has a layered architecture which is structured like this: Model, Data Access Object and Presentation.

The model has classes which include particularities and have the exact same fields as the tables from the database.

The data access object or DAO does the bond to the database, here we retrieve information based on some SQL statements.

In the presentation layer we have the user interface classes which helps the users to interact with the library management application.

# Non-functional Requirements

*[Discuss the non-functional requirements for the system]*

2. Use-Case Model

*A picture containing text, map

Description generated with very high confidence*

Use case: Manage library

Level: User- goal level

Primary actor: Staff/Librarian

Main success scenario: The librarian manages to connect to the app using the log in page. Then he is able to manage the library by adding, removing or updating the stock of books and then he can filter the information from the database

Extension: If the librarian will not be able to connect, it will be impossible to access or to filter the information from the database

A close up of a logo

Description generated with very high confidence

Use case: Borrow a book and return it

Level: user-goal level

Primary actor: User/Client

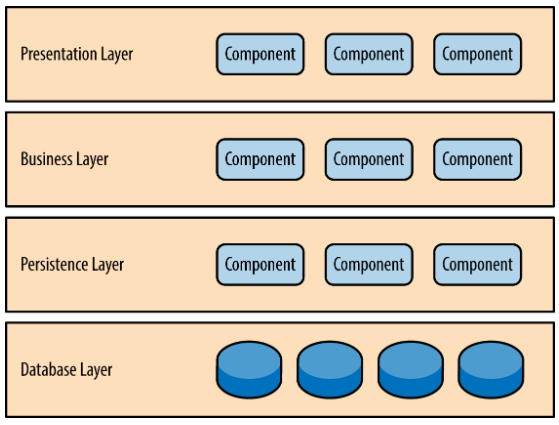
Main success scenario: After the user manages to connect to the application using the log in page, he will be able to see the books which are available and not. The client will be able to see all the title of the books, including authors, genre and price. After then a user can borrow a books and after that to return them.

Extension: If a book is not available at a moment, the user must wait until it is and a message will be displayed.

3. System Architectural Design

**3.1 Architectural Pattern Description**

Layered architecture is organized into horizontal layers, each layer performing a specific role within the application. The layered architecture pattern does not specify the number and types of layers that must exist in the pattern, most layered architectures consist of four standard layers: presentation, business, persistence and database. In some cases, the business layer and persistence layer are combined into a single business layer, particularly when the persistence logic is embedded within the business layer components. Smaller application may have only three layers, whereas larger and more complex business applications may contain five or more layers.



**3.2 Diagrams**

System conceptual architerture:

*A screenshot of a cell phone

Description generated with high confidence*

Package Diagram:

A close up of a logo

Description generated with very high confidence

Component diagram:

A screenshot of a social media post

Description generated with very high confidence

Deployment diagram:

4. UML Sequence Diagrams

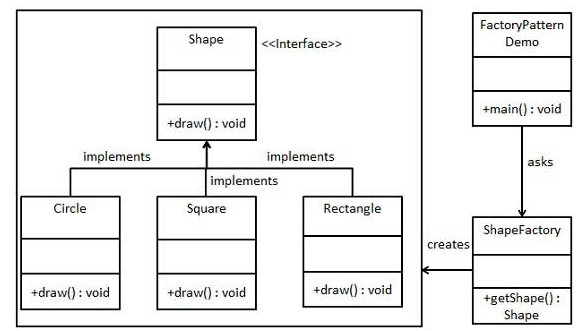
*[Create a sequence diagram for a relevant scenario.]*

5. Class Design

**5.1 Design Patterns Description**

**Factory pattern** is one of the most used design patterns in Java. 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 and refer to newly created object using a common interface.

Example:



**5.2 UML Class Diagram**

*[Create the UML Class Diagram and highlight and motivate how the design patterns are used.]*

6. Data Model

*[Present the data models used in the system’s implementation.]*

7. System Testing

*[Present the used testing strategies (unit testing, integration testing, validation testing) and testing methods (data-flow, partitioning, boundary analysis, etc.).]*

8. Bibliography

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

<https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html>