BS2202 - Object Oriented Software Development

Stock Management System: Library Loan System

**Word Count: /2000**

**Jake Clapson**

Table of Contents

[Project Scope and Client Requirements 3](#_Toc65510483)

[Introduction 3](#_Toc65510484)

[User Requirements 3](#_Toc65510485)

[Core Product List 4](#_Toc65510486)

[Key System Features 4](#_Toc65510487)

[User Interface Diagrams 4](#_Toc65510488)

[Wireframes 4](#_Toc65510489)

[Accessibility 4](#_Toc65510490)

[Usability 4](#_Toc65510491)

[Class Diagrams 5](#_Toc65510492)

[Unified Modelling Languages (UML) 5](#_Toc65510493)

[Class Diagrams 5](#_Toc65510494)

[Use Case Diagram 5](#_Toc65510495)

[Database Design Diagram 5](#_Toc65510496)

[Design Patterns 5](#_Toc65510497)

[Testing Documentation 5](#_Toc65510498)

[Testing Plan 5](#_Toc65510499)

[Testing Execution 5](#_Toc65510500)

[References 5](#_Toc65510501)

Table of Figures

Table of Tables

[**Table 1 - Table of Requirments** 3](#_Toc71881401)

[**Table 2 - Core Product List** 4](#_Toc71881402)

[**Table 3 - Wireframes** 5](#_Toc71881403)

[**Table 4 - Functional Testing Table 1** 8](#_Toc71881404)

[**Table 5 - Functional Testing Tabel 2** 9](#_Toc71881405)

Library Loan System

# Project Scope and Client Requirements

## Introduction

For my summative assignment, I am tasked to develop and program a project which is a library loan system, and this portfolio will contain all the project documentation associated for the system. The scenario for my system will be public library based in Basingstoke. The library loan systems intended users are only for the library employees and not for the public. A library loan system is effectively a stock management system for the library’s items/stock which can be loaned out to members of the public. Which records numerous pieces of information such as availability of items which can be loaned and dates for returning items.

Programming concepts which will be included in the system will be using Java as my programming language and employing Object Oriented Programming (OOP) for the programming paradigm. In addition to this, I am also going to incorporate inheritance within my code for code efficiency, have a database to store information on items, a Graphical User Interface (GUI) to make the system user friendly, and include pictures of each item to provide some visual aspects for the system.

This portfolio covers the user requirements and system components for the system, user interface diagrams to picture how the system interface will look, class diagrams and other Unified Modelling Language (UML) diagrams to help explain the structure of the code and system, and lastly conducting testing on the system.

## User Requirements

My users for the library loan system are for the librarians only. Users will supply requirements for what functionality they desire for their system. This helps to guide us into constructing the system and what features it needs to ensure the end users are satisfied. This is important for all businesses since, if the customer is not satisfied then it can have financial and reputational effects, with the business going into jeopardy. Requirements table is found below.

**Table 1 - Table of Requirments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Functional Requirements** | **Requirement ID** | **Non-functional Requirements**  (Mairiza et al, 2010) |
| FReq.1 | More than one type of item can be loaned to one student | NReq.1 | Performance: When in use abnd the user enters input, the system must respond instantaneously. |
| FReq.2 | More than one option is available for each item | NReq.2 | Usabilitiy: I aim to ensure my library system will be easy to use, and anyone without prior knowledge could be introduced to the system and learn it very quickly. |
| FReq.3 | Ability to check the current number of each item which is available for loan, and already loaned | NReq.3 | Reliability: The systems functionality must be reliable and consistent thoughout. Furthermore, the system must keep the integrity of the information by not corrupting. |
| FReq.4 | Ability to process a loaned item | NReq.4 | Accessibility: During designing the interface, I must ensure the accessibility is open to everyone to use irrespective if they are at a disadvantage. |
| FReq.5 | Ability to process a returned item |  |  |
| FReq.6 | Ability to print information to a text file about a single item, information to print could be basic details such as book title, author, song name, artists and song length, the number of copies available and on loan, and return date |  |  |
| FReq.7 | Having a search function to search for items |  |  |
| FReq.8 | A login system for different user accounts, a standard and admin account. |  |  |
| FReq.9 | Storing customer details and contact information |  |  |
| FReq.10 | Manage and promote users |  |  |

**How to gather requirements in an ideal environment:**

(Shen et al, 2004) & (Young, 2002)

* Identify and analyse a problem so there is an issue to focus on which requires a solution.
* Put a business team together to solve this problem.
* Research into the stakeholders, market, and any other relevant entities to gather background information on the identified problem.
* Conduct research activities such as questionnaires and surveys to the target audience to collect their insights.
* Analyse the results from the previous two steps and group the research data into appropriate groups, which are the requirements.
* Based on the requirements, narrow down the final solution ideas.

## Core Product List

In order to develop and program the system, hardware and software components are required as a part of the system setup. Here, I will outline the hardware and software components.

**Table 2 - Core Product List**

|  |  |
| --- | --- |
| **Component** | **My Component Type Choice** |
| Programming Language | Java |
| Programming Paradigm | Object Oriented |
| Database | MySQL |
| Integrated Development Environment | Apache Netbeans IDE |
| Operating System |  |

## Key System Features

The key system features are derived from the requirements, however below is the core functionality my library loan system must perform:

* Must be able to process the action of loaning and returning items with the database and item availability numbers updating accordingly.
* Must have a GUI whereby the navigation must be useable easily.
* Database up and operating, storing the information of the items in my library system.
* Have different user accounts, administrator and standard,, with the ability to regioster new accounts.

# User Interface Diagrams

## Wireframes

Below is a table demonstrating my wireframes for my proposed application. A wireframe is a skeletal outline of my interface designs showing the basic components of how they will look (Sharief, 2020). Wireframes don’t show any graphics.

**Table 3 - Wireframes**

|  |  |  |
| --- | --- | --- |
| Wireframe number | Wireframe | Page Purpose |
| 1 |  | Login Page: Enables users to login and access the application. |
| 2 |  | Settings Page: Where users can access different setting functions |
| 3 |  | Home Page: Where users can navigate to different functionalities within the application. Users can also logout here. |
| 4 |  | Loan Items Page: Users have a choice of three items to loan. |
| 5 |  | Loan Book Page: This is where books are loaned from. |
| 6 |  | Loan Movie Page: This is where movies are loaned from. |
| 7 |  | Loaned TV Shows Page: This is where TV Shows are loaned from. |
| 8 |  | Checkout Page: This is where users checkout the item they want to loan. |
| 9 |  | Return Page: This is where uses go to return an item which has been loaned. |

## Accessibility

The aim is to ensure my library system is accessible to all users, even those at a disadvantage. Therefore, in the real envrionment, I would ensure users with visual impairments, motor skills, or cognitive disabilities (such as dyslexia) would have equal accessibility to user who don’t have these disabilities. This is achieved by ensuring the system has screen readers, speech recognition software, braille keyboards and mouses, and other technologies to help these users.

This links to the Sustainbvale Development Goal 10 - Reducing Inequalities. As I am ensuring all types of users can and will access my system with ease regardless if they have any disabilites, which reduces inequality.

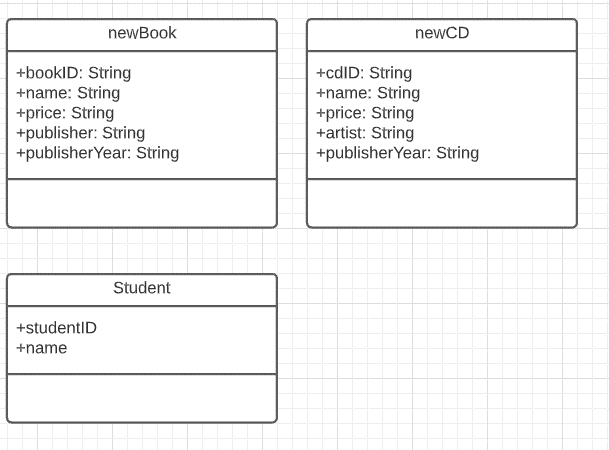
## Usability

Based upon my wireframes, the usability for my interface is going to be basic and simple. This ensures that the application will be easy to use and interact with. Furthermore, this influences their user experience, and my intentions is to maximse the user experience.

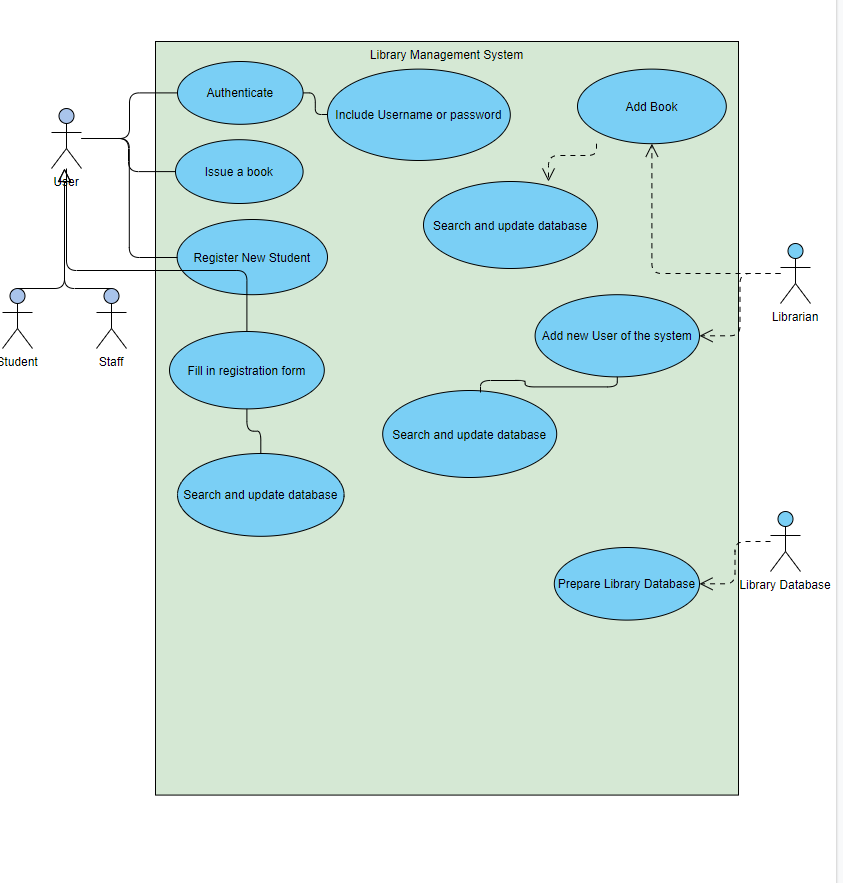
I have achieved good usability by having clear navigation around the application, good size text relative to the screen size, and a logical layout of the buttons. This links to Sustainble Development Goal 10 - Reducing Inequalities, as my interface will reduce inequalities by enabling users at a disadvantage to still use the application with ease. For example, users who struggle to read quickly or have colour blindness, my interface will be designed to have short, readable, and clear text on screen and the colours used will be constrasting one another making it clear where the buttons are, where texts boxes are etc.

# Class Diagrams

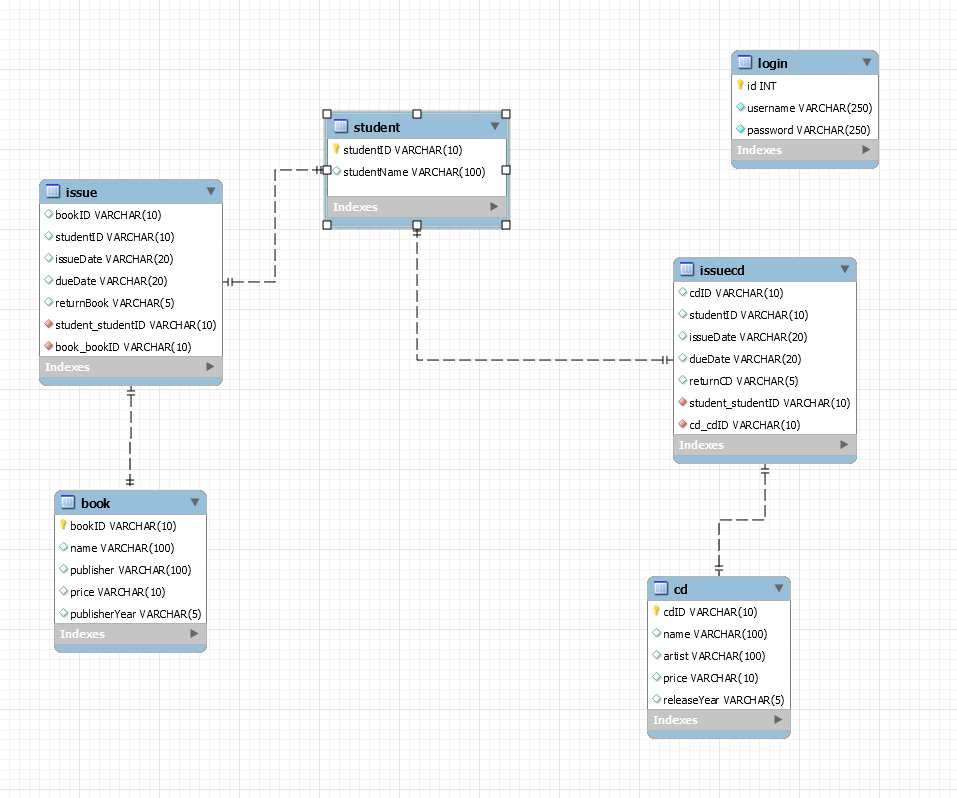
## Unified Modelling Languages (UML) Class Diagram



## Use Case Diagram



## Database Design Diagram



## Design Patterns

# Testing Documentation

## Testing Plan

The test plan is to execute functional testing on my Library System to ensure it fulfils all the requirements and functions it is supposed to perform. Functional testing simply tests the functional aspects of the system such as, buttons operate correctly, I can login as different accounts, I can loan items and more.

## Testing Execution

### Functional Testing Table 1

Functional testing table 1 outlines the test case and scenario, what my input data is and the expected and actual result.

**Table 4 - Functional Testing Table 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Scenario** | **Test Case** | **Input Data** | **Expected Result** | **Actual Result** |
| 1 | Can the admin login | Login in the system as admin | Admin username and password | “Successfully logged in as admin” | “Successfully logged in as admin” |
| 2 | Can the standard user Login | Login as standard user | username and password | “Successfully logged in as standard user” | “Successfully logged in as standard user” |
| 3 | Can all the buttons be pressed on the home page | Press all the buttons to see which ones don’t work | Press | all the buttons work | all the buttons work |
| 4 | Adding a new student to the database | Adding a new student to the database | Student ID and Student Name | “student ID” and “Student Name” get added to the database | “student ID” and “Student Name” get added to the database |
| 5 | Adding a new book to the database | Adding a new book to the database | BookID, Name, Publisher, Price and Publisher Year | BookID, Name, Publisher, Price and Publisher Year all get added to the book database | BookID, Name, Publisher, Price and Publisher Year all get added to the book database |
| 6 | Adding a new CD to the database | Adding a new CD to the database | cdID, Name, Artist, Price and Release Year | cdID, Name, Artist, Price and Release Year all get added to the cd database | cdID, Name, Artist, Price and Release Year all get added to the cd database |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### Functional Testing Table 2

Functional Testing Table 2 provides the screenshots of the testes executed in table 1.

**Table 5 - Functional Testing Tabel 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Screenshot before test** | **Screeenshot after test** | **Test successful? (Y/N)** |
| **1** |  |  | **Y** |
| **2** |  |  | **Y** |
| **3** |  |  | **Y** |
| **4** |  |  | **Y** |
| **5** |  |  | **Y** |
| **6** |  |  |  |
| **7** |  |  |  |
| **8** |  |  |  |
| **9** |  |  |  |
| **10** |  |  |  |
| **11** |  |  |  |
| **12** |  |  |  |
| **13** |  |  |  |
| **14** |  |  |  |
| **15** |  |  |  |
| **16** |  |  |  |
| **17** |  |  |  |

# References

Mairiza, D., Zowghi, D. and Nurmuliani, N., 2010, March. An investigation into the notion of non-functional requirements. In *Proceedings of the 2010 ACM Symposium on Applied Computing* (pp. 311-317).

Shen, H., Wall, B., Zaremba, M., Chen, Y. and Browne, J., 2004. Integration of business modelling methods for enterprise information system analysis and user requirements gathering. *Computers in Industry*, *54*(3), pp.307-323.

Young, R.R., 2002. Recommended requirements gathering practices. *CrossTalk*, *15*(4), pp.9-12.

Sharief, K., 2020. *What is Wireframe? - Definition, Uses, Examples, And More*. [online] Computer Tech Reviews. Available at: <https://www.computertechreviews.com/definition/wireframe/> [Accessed 14 May 2021].