**Group 5 – Library System**

Design Document Specification

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 3/25/2025 | 1.0 | Initial Version | Joshua Charlesworth |
| 3/30/2025 | 1.1 | Document Layout Adjustments | Joshua Charlesworth |
| 4/05/2025 | 1.2 | Updated UML Class Diagrams | Joshua Charlesworth |
| 4/07/2025 | 1.3 | Updated UML Use Case Diagrams | Wai Ip |
| 4/07/2025 | 1.4 | Separated Sequence Diagrams | Wai Ip |
| 4/07/2025 | 1.5 | Added new Network and Communication Module | Arsen Dzhuraev |
| 4/07/2025 | 1.6 | Refined UML Class Diagrams | Joshua Charlesworth |
| 4/07/2025 | 1.7 | Design Descriptions Added | Jordan Walker |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** **Purpose 4**

1.1. Scope 4

1.2. Definitions, Acronyms, Abbreviations 4

1.3. References 4

1.4. Overview 4

**2.** **Design Description 5**

2.1. Design Perspective 5

2.2. Design Architecture 5

2.3. Product Functionality/Features 5

**3.** **Modules 6**

3.1. Inventory Management Modules 6

3.2. Account Management Modules 6

3.3. Location Management Modules 7

3.4. Item Attainment Modules 7

3.5. Network Communication Modules 7

**4.** **UML Use Case Diagram 8**

4.1. Use Case Diagrams 8

**5.** **UML Class Diagrams 9**

5.1. Class Diagrams 9

**6.** **UML Sequence Diagram 10**

6.1. Sequence Diagrams 10

# Purpose

This document outlines the requirements for the County Library System (CLS).  
This document should be used in conjunction with the *Software Requirements Specification (SRS)* document.

**Project found on GitHub at: https://github.com/C0rp0r4l/CS401LibraryProject**

## Scope

This document will catalog the design of user, system, and hardware requirements for the CLS system.

## Definitions, Acronyms, Abbreviations

ID: identification

GUI: Graphical User Interface

Items: Books, CDs, Laptops, Computers

## References

Use Case Specification Document – Pages 9-20

UML Use Case Diagrams Document – page 21

Class Diagrams – Page 22

Sequence Diagrams – Page 23

## Overview

The Multi-Library System is meant to provide a software package to handle all the necessary functions to run a library. Handling functions such as managing library ID’s, managing site inventory, as well as providing the flexibility to add or remove new library sites to the system.

# Design Description

## Design Perspective

The way we implement this system is heavily dependent on the intended use cases and interactions we have devised with how we wish for the software to be used. For instance, almost all the interactions will take place via a staff member interacting with the GUI. The only time a regular member (cardholder) will be directly interacting with the software is to search up items at various locations. This system was designed with the ability for staff to be able to check themselves out since they are already trusted members of the system. The server will be acting as a backend by fetching and updating data as requested.

## Design Architecture

The system will be organized into 5 major modules: the Inventory Management module, the Account Management module, the Item Attainment module, the Location Management module, and the Network Communications module. The inventory management module is included in order to accurately and efficiently manage inventory such as location, genre, and what is being currently rented out for example. The Account Management module is included in order to make sure that accounts are accurately kept up with and in order to create checks and balances between locations and other staff members. The Item Attainment module is included in order to have smooth processing of items and in order to ensure that each member has fair access to items based on their staff status. The Location Management module exists in order to ensure proper bookkeeping of locations; which items and staff are at each location, and what items/staff might be transferred between locations. Finally, the Network Communications module is included in order to specify the regulations of data transfer between the Staff Interface and the backend. This includes specifications such as headers, maximum data size, and the fashion in which the database will be updated.

## Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

Librarians should be able to, using a member’s library ID, check out various types of items using the Multi-LIbrary system. Librarians should be able to create or delete member accounts, check the status of an account, and change the status of an account. Librarians should also be able to check the status of a type of item as well as change the status of the item, add to a waitlist, and create/delete items.

# Modules

## Inventory Management Module

3.1.1.1 Users and Staff should be allowed to log in using their issued id and pin, both of which are alphanumeric strings between 6 and 20 characters in length.

## Account Management Modules

3.2.1 The system must provide an interface to the Library database in order for items to be reliably looked up based on location, genre, type of item, year, title, author, and id.

3.2.2 The interface must be a Java based GUI allowing the users to borrow, return books from the library.

3.2.3 The system should communicate between the user and the server over TCP/IP.

3.2.4 The system should only allow staff to check in/out items, create/delete accounts, add/delete locations, and add staff.

3.2.5 The external interface must not give users even the option to do the same actions as staff. Members should not have a button/option to do privileged actions.

## Location Management Modules

3.3.1 The system will store items by layers of distinction. Layer 1 is Location, Layer 2 is type of item (ex: CD, book, etc.), Layer 3 is genre or year.

3.3.2 The system must process new accounts and items by storing them in correct distinction layers and ensuring duplicates only when valid.

3.3.3 Each account, item, and location will have a unique alphanumeric identifier of 10 characters that can be used to search for a specific id number as well as to track transactions.

## Item Attainment Modules

3.4.1 The system will store items by layers of distinction. Layer 1 is Location, Layer 2 is type of item (ex: CD, book, etc.), Layer 3 is genre or year.

3.4.2 The system must process new accounts and items by storing them in correct distinction layers and ensuring duplicates only when valid.

3.4.3 Each account, item, and location will have a unique alphanumeric identifier of 10 characters that can be used to search for a specific id number as well as to track transactions.

## Network and Communication Modules

3.5.1 The Network Communication module must handle all the backend communication between the server and client interface making sure that all the operations (login, book search, borrowing, returning, staff operations) are all handled securely over a TCP/IP connection.

3.5.2 The system must have a secure TCP/IP connection between the server and the client interface for stable communication.

3.5.3 The module must be able to send requests from the GUI to the server and receive updates, and a **ClientHandler** class to receive for incoming process requests, and return appropriate responses.

3.5.4 The system must synchronize all the communication data across server/client through the GUI

3.5.5 The module must use a shared **Message** class that interacts with the GUI to display system messages such as feedback, errors, confirmations, and alerts after receiving feedback from the server.

3.5.6 The module must delegate different requests to other modules such as the Inventory Management, Account management, Location Management based on message type headers. These headers are related to the modules, which include a subsection of use cases.

# UML Use Case Diagrams

# Class Diagrams

## 

# Sequence Diagrams

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 









