Library Software Requirements

Specification Sheet

Revision History

Date	Revision	Description	Author
mm/dd/yyyy	1.0	Initial Version	Your Name
9/17/2024	1.1	Initial Drafting + 1,2 definitions and acronyms 1.3 use cases 01-03 added	Christian Amoranto, Darrel Heim, Chandler Hui, Quan Luu, Prayusha Praikh
9/24/2024	1.11	Use Case 04 added, small detail edits	Chandler Hui
9/26/2024	1.12	Use Case 05 added, more small edits	Chandler Hui
9/30/2024	1.13	Use Case 06 added, small edits, and added one requirement	Quan Luu
10/1/2024	1.14	Part 3 requirements added	Christian Amoranto
10/2/2024	1.15	Class UML added,2.2 Production Architecture Updated,use case 07 and 08 added, use case 12 amd 13 added	Darrell Heim
10/2/2024	1.16	Use case 09, 10,and 11 added and small edits, added Class + Sequence Diagrams. Non-functional requirements added.	Prayusha Parikh, Chandler Hui, Quan Luu
10/2/2024	1.17	Added 2.1 Perspective and other small edits	Christian Amoranto

Table of Contents

1. Purpose

This document outlines the requirements for the library software.

1.1. Scope

This document will catalog the user, system, and hardware requirements for the library system. It will not, however, document how these requirements will be implemented.

1.2. Definitions, Acronyms, Abbreviations

LMS = Library Management System

Staff = A registered worker at the library with varying degrees of authority
Member = Public user of the library

URS = User Registration system, a way for members and staff to log in

BCS = Book Catalog Search

Checked out = a book that has been taken by a member or staff

Available = a book that isn't currently being loaned and within library

Lost = a book that is neither Loaned or available

Return date = date a loaned book must be returned by

Overdue book = a book that has been loaned to a member or staff past its return date

Late fee = money that must be paid by member or staff for overdue books

1.3. References

Use Case ID: 01

Use Case Name: **Registration System**

Relevant Requirements: Users must either register as a staff member or a public library member.

Primary Actor: Staff or member

Pre-conditions: User Registration System works and accepts new users

Post-conditions: user added to system as a staff or member with appropriate class object created and written to output file

Basic Flow or Main Scenario: {Numbered flow of events: 1 The user initiates an action by entering 0 for both username and password. 2 The system responds by displaying a prompt on screen to enter a username. 3 the user enters their username via keyboard. 4 system responds by checking the input file to see if the username has already been taken. 5 the system then displays to the screen, asking the user for password 6. User enters a password via keyboard. 7 asking if the user is a staff or member. 6 The User enters whether they are a staff member or a member. 8 a new class object created for users either staff or a member depending on input. 9 State of class object saved to outputFile}

Extensions or Alternate Flows: { step 4 if Username.is already taken user asked to pick a different username}

Exceptions: {file not found throw an error message saying input or output file not detected }

Related Use Cases: {login system.is used after}

Use Case ID: 02

Use Case Name: **Login System**

Relevant Requirements: User must either be staff or member with a username and password

Primary Actor: Staff or member

Pre-conditions: Staff or member must be registered

Post-conditions: Staff or member will have access to the menu with options available depending on status of level of authority

Basic Flow or Main Scenario: {Numbered flow of events: 1 The user initiates an action by entering username and password into keyboard. 2 The system responds by using a while loop to check keyboard input against inputfile with names and password written to it. 3 if name and password match one in input file then user is logged in,4 main menu is displayed to screen}

Extensions or Alternate Flows: {If name and or password do match any in the input file then user is asked to re-enter name and password}

Exceptions: In the case of an invalid combination of username/password, the system denies and sends an error to try again.

Related Use Cases: UC-01, the user must have registered first to have a username and password combination.

Use Case ID: 03

Use Case Name: **Administration System**

Relevant Requirements: Be a staff member of the library

Primary Actor: Staff member

Pre-conditions: Must have the authority of a staff member, public members cannot access this system

Post-conditions: Gives more detailed information about both books and public members, such as seeing who has what book and if books are overdue/missing

Basic Flow or Main Scenario: 1) Staff member logs in 2) System identifies staff as different than public member and displays information differently

Extensions or Alternate Flows: The different display will not appear for public members.

Exceptions: {This section describes all error conditions that can arise in the use case.}

Related Use Cases: UC-02, the staff member must be logged in to access this system

Use Case ID: 04

Use Case Name: **BCS (Book Catalog Search)**

Relevant Requirements: Be a staff or public member of the library

Primary Actor: Staff or public library member

Pre-conditions: Must be a registered user

Post-conditions: Gives either a basic overview of books and search function to public members, and more complicated actions like sorting, managing, marking book statuses ect for staff members.

Basic Flow or Main Scenario: 1) Staff or public member logs in 2) System identifies either a staff or public member 3) Public members can perform a basic search and see if a book's status is available or already borrowed, staff members have more options like changing book statuses and information.

Extensions or Alternate Flows: Could include a more intricate UI for staff specifically.

Exceptions: A book is unavailable or has missing information

Related Use Cases: UC-01 and UC-03, it must know the status of who is accessing the catalog and grant higher privileges to staff members

Use Case ID: 05

Use Case Name: **Checking Out Books Online**

Relevant Requirements: Have a selection of books to display for members to check out

Primary Actor: Public library member

Pre-conditions: Must be a registered user to be able to check out books, and the book must be available

Post-conditions: Adds selected book to user's list, also giving an option to just check it out by itself

Basic Flow or Main Scenario: 1) Public library member logs in, which gives them access to check out books after viewing them or adding to a cart 2) User will confirm books they want to check out 3) Library system will flag those books as unavailable 4) User will be able to go to a library circulation desk and take the books out

5)check out date date variable on book updated

Extensions or Alternate Flows: There could be a way for the user to manually scan books they want to take out, and the computer terminal will read from that scanner, if book status anything but available display"book unavailable" return to main menu

Exceptions: User is not logged in, and thus cannot check out books. Possible inventory error could display a

book as available when it isn’t.

Related Use Cases: UC-04, user could search up books, add them to a cart-style list, and then check them out

Use Case ID: 06

Use Case Name: **Returning Book(s)**

Relevant Requirements: The book must be checked out by the user

Primary Actor: Public library member

Pre-conditions: Must be a registered user and the checked-out book must be flagged under their ID

Post-conditions: The book is flagged as available in the system, and the book is no longer checked out under the returned user’s ID

Basic Flow or Main Scenario: 1. Public library member returns it either in a designated area or goes to the front desk. 2. Staff scans the book. 3. The system flags the book as available and the book is no longer under the member’s checkedOut array

Extensions or Alternate Flows: if the due date has been passed calculate the late fee and charge to balance, if the book is marked reserved don't update the book status.

Exceptions: N/A

Related Use Cases: UC-05, the book must be checked out

Use Case ID: 07

Use Case Name: **Pay Fees**

Relevant Requirements: Must be a registered user

Primary Actor: Staff and members

Pre-conditions: Have an non zero balance on user account

Post-conditions: subtract payment from balance and display balance to screen

Basic Flow or Main Scenario: 1) Public library member logs in, which gives them access to menu 2) User select pay balance 3) user balance is displayed to screen 4) User inputs the amount they are paying 5) user pays that amount 6) balance subtracts payment and is updated 7) balance displayed to screen

Extensions or Alternate Flows: if payment exceeds balance change will be dispersed

Exceptions:n/a.

Related Use Cases: n/a

Use Case ID: 08

Use Case Name: **Add Book**

Relevant Requirements: Must be a registered staff

Primary Actor: Staff

Pre-conditions: staff must be logged in

Post-conditions: new book added to library array and library txtfile updated

Basic Flow or Main Scenario: 1) Public library staff logs in, which gives them access to menu 2) staff selects add book 3) system responds by checking the users staff bool 4) User inputs the book title,Author,and genre 5) system responds by creating a book obj 6) book obj is added to Library’s book array and book txt file updated

Extensions or Alternate Flows: if book already exists in system display “book already added no replicas”

Exceptions:if book txt file doesn't open Throw message”file error” and return to main menu

Related Use Cases: n/a

Use Case ID: 09

Use Case Name: **Book Reservation**

Relevant Requirements: Must be a registered user

Primary Actor: Public Library Member

Pre-conditions:The book must be available for reservation, and the user must be logged in.

Post-conditions:The book is flagged as reserved under the user's ID, and the system prevents others from checking it out until the reservation period ends.

Basic Flow or Main Scenario: 1)A public library member logs in.2)A user searches for a book.3)The user selects to reserve a book. 4)The system confirms the reservation and sets book status to reserved.

Extensions or Alternate Flows: Staff may override the reservation if necessary.

Exceptions: The book has already been reserved or is currently unavailable.

Related use cases include UC-04 (Book Catalog Search) and UC-05 (Checking Out Books).

Extensions or Alternate Flows:Staff can override the reservation if needed.

Exceptions:The book is already reserved or unavailable.

Related Use Cases: UC-04 (Book Catalog Search), UC-05 (Checking Out Books Online).

Use Case ID: 10

Use Case Name: **Manage Reservation**

Relevant Requirements: Must be a Staff Member

Primary Actor: Public Library Staff

Pre-conditions: Must have administrative privileges to manage reservations

Post-conditions: The system updates the status of reserved books based on staff input, such as extending or canceling reservations.

Basic Flow or Main Scenario: Staff member logs in,System recognizes the user authority level,Staff accesses the reservation management system,Staff can view, extend, or cancel reservations as needed.
Extensions or Alternate Flows: Staff can contact users to confirm or notify them about reservations.

Exceptions:Book reservation has expired or cannot be updated due to system error.

Related Use Cases: UC-09 (Book Reservation), UC-02 (Login System).

Use Case ID: 11

Use Case Name: **View Loan History**
Relevant Requirements: Must be a registered user
Primary Actor: Public library member or staff
Pre-conditions: User must be logged in.
Post-conditions: The system displays a detailed history of loaned and returned books under the user's ID.
Basic Flow or Main Scenario: User logs in,User selects "View Loan History" from the menu,System retrieves and displays the user's loan history.
Extensions or Alternate Flows: Staff can view the loan history of any user.
Exceptions: User has no loan history to display.

Related Use Cases: UC-06 (Returning Books), UC-05 (Checking Out Books).

Use Case ID: 12

Use Case Name: **Remove Book**

Relevant Requirements: Must be a registered staff

Primary Actor: Staff

Pre-conditions: staff must be logged in

Post-conditions: a book removed from library array and library txt file updated

Basic Flow or Main Scenario: 1) Public library staff logs in, which gives them access to menu 2) staff selects remove book 3) system responds by checking the users staff bool 4) User inputs the book title,Author,and genre 5) system responds by searching array for matching book obj 6) book obj is removed from Library's book array and book txt file updated

Extensions or Alternate Flows: if book already exists in system display "book already added no replicas"

Exceptions:if book txt file doesn't open Throw message"file error" and return to main menu

Related Use Cases: n/a

Use Case ID: 13

Use Case Name: **Library Search**

Relevant Requirements: Must be a registered staff or member

Primary Actor: Staff

Pre-conditions: user must be logged in

Post-conditions: library name and address displayed to screen

Basic Flow or Main Scenario: 1) Public library user logs in, which gives them access to menu 2) staff selects "search for library" 3) system responds by prompting user to enter library name 4) User inputs the library name 5) system respond by searching array of libraries for name 6) if library is in array display library name and address to screen

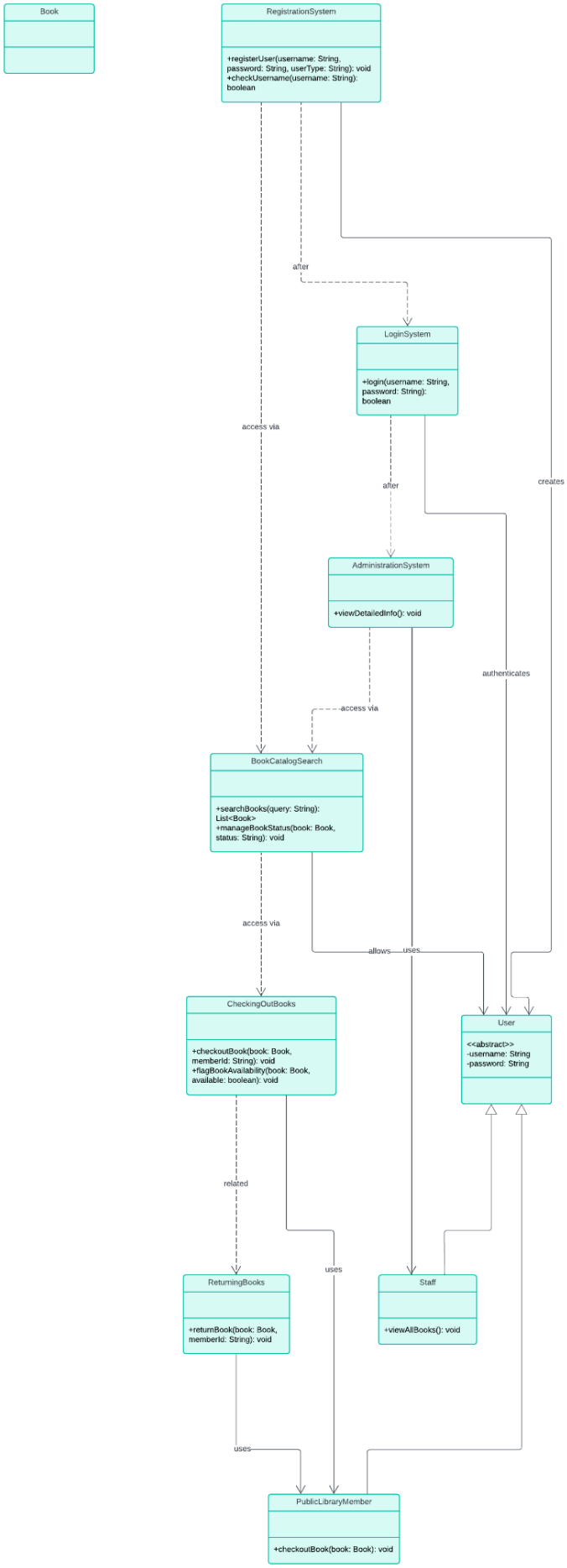
Extensions or Alternate Flows: if library not in array display"library not found" to screen

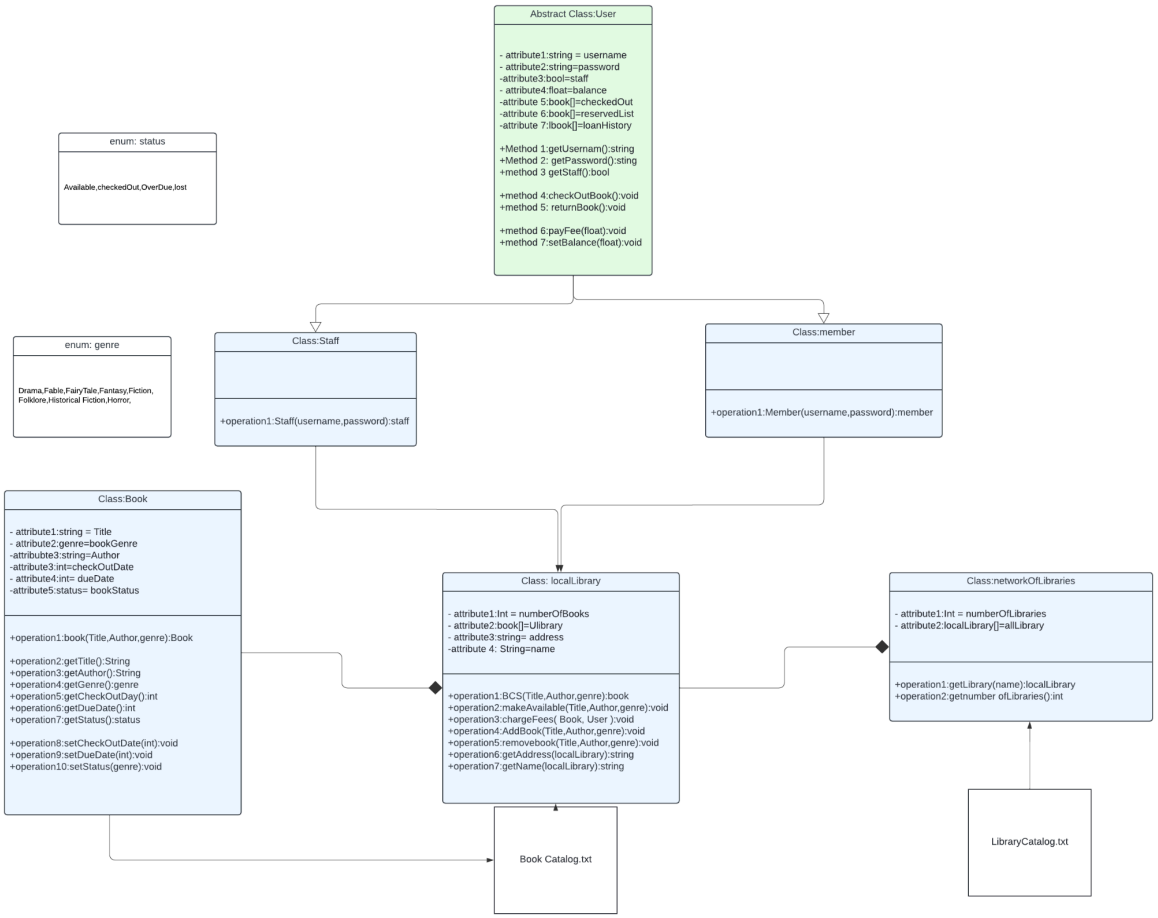
Exceptions:n/a

Related Use Cases: n/a

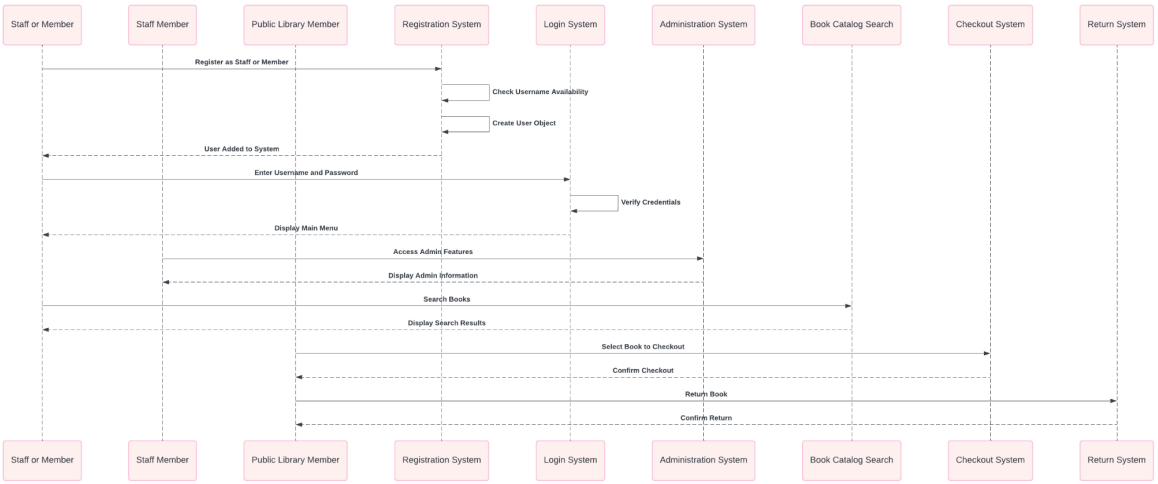
UML Use Case Diagrams Document – Step 3 in assignment description

Class Diagrams – Step 5 in assignment description


Book
RegistrationSystem
+registerUser(username: String, password: String, userType: String): void
+checkUsername(username: String): boolean
after
LoginSystem
+login(username: String, password: String): boolean
access via
after
creates
AdministrationSystem
+viewDetailedInfo(): void
authenticates
access via
BookCatalogSearch
+searchBooks(query: String): List<Book>
+manageBookStatus(book: Book, status: String): void
access via
allows
uses
CheckingOutBooks
+checkoutBook(book: Book, memberId: String): void
+flagBookAvailability(book: Book, available: boolean): void
User
<<abstract>>
-username: String
-password: String
related
uses
ReturningBooks
+returnBook(book: Book, memberId: String): void
Staff
+viewAllBooks(): void
uses
PublicLibraryMember
+checkoutBook(book: Book): void



Sequence Diagrams – Step 6 in assignment description



1.4. Overview

The library software is designed to keep track of books within the library and provide users and staff with ways to search for books and their availability. The system will also have data on these books to provide a better search function.

2. Overall Description

2.1. Product Perspective

The Library Management System (LMS) is designed to support the daily operations of a network of public libraries. Its provided functionalities include: user registration, book catalog management, loan/return tracking, fine calculation, and a renewal and reservation system. The LMS aims to improve the efficiency of library operations and provide a user-friendly experience for public library members.

2.2. Product Architecture

The system will be organized into 6 major modules: the User Authentication Module, the Book Catalog Module, the Loan/Return Tracking module, the Book Renewal Module, the Book Reservation Module, and the Fine Calculation Module.

2.3. 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):

2.4. Constraints

The system must be able to work with and connect multiple library networks.

Since public library computers will be running this, the system must not take many resources to execute.

An easy to understand interface and experience must be kept in mind due to the wide variety of people who may use this, i.e. children and elderly.

2.5. Assumptions and Dependencies

We will assume that a variety of users will be using the system, so it should be easy to understand for everyone.

The amount of users will be assumed to be numerous as it is for a public library.

There are other library networks we must make this compatible with, not just for one library specifically.

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1 Users are assigned a unique ID for checkouts and returns.

3.1.1.2 The system should be able to differentiate between a public member and staff.

3.1.1.3. Tracks all book transactions and maintains transaction logs.

3.1.1.4. Ensures that each module performs its respective operations in an efficient manner.

3.1.2. User Authentication Module Requirements:

3.1.2.1. Allows users to register/login as library staff or a member with a unique username and password.

3.1.2.2. Grants administrative permissions to staff and restricts members from accessing these permissions.

3.1.2.3. Restricts password combinations to have a minimum of 8 letters and include uppercase letters, lowercase letters, numbers, and special characters.

3.1.3. Book Catalog Module Requirements:

3.1.3.1. Allows staff to add/update books including their titles, authors, genres, ISBNs, availability status, and reservation status.

3.1.3.2. Allows staff to remove books, provided that the book is available and/or not reserved.

3.1.3.3. Allows users to search the catalog for books by title, author, genre, or ISBN.

3.1.4. Loan/Return Tracking Module Requirements:

3.1.4.1. Allows members to borrow a maximum of 3 books at a time, setting a due date of two weeks from the loan date.

3.1.4.2. Allows members to return books.

3.1.4.3. Records the date and member details of every book transaction and updates the availability/reservation status of each book.

3.1.5. Book Renewal Module

3.1.5.1. Allows members to request renewals on books as long as the request is submitted before the due date.

3.1.5.2. Extends the loan period on renewed books by another 2 weeks from the original due date, updating the due date in the loan records

3.1.5.3. Denies renewal requests if:

- The book is overdue
- The member has outstanding fines
- The book is already reserved by another member

3.1.6. Book Reservation Module Requirements:

3.1.6.1. Allows members to place a reservation on books that are already checked out, adding them to a reservation queue for the book and updating the reservation status of said book.

3.1.6.2. Notifies the next member in queue when the book is available.

3.1.6.3. Allows members to cancel their reservation.

3.1.6.4. Prevents the book from being loaned to other members outside of the reservation queue.

3.1.7. Fine Calculation Module Requirements:

3.1.7.1. Automatically applies a 5$ flat fee plus $0.25 for each day past the due date a book is late and calculates the total fine when the book is returned.

3.1.7.2. Allows staff to update book status as ‘lost’ or ‘damaged’.

3.1.7.3. Notifies members of any incurred fines and displays the outstanding fines on the member’s account.

3.1.7.4. Provides options for members to pay fines, updating the member’s account status upon payment.

3.2. External Interface Requirements

3.2.1. Provides a user-friendly interface that allows staff and members to perform all required functions.

3.2.2. Provides notifications on member dashboards for overdue books and reservation availability.

3.2.3. Integrates a third-party payment processor to allow members to pay fines online.

3.3. Internal Interface Requirements

3.3.1. The User Authentication Module must interface with all other modules to enforce role-based permissions.

3.3.2. The Book Catalog Module must interface with the Loan/Return Tracking Module in order to update books’ availability and reservation status after a transaction takes place.

3.3.3. The Fine Calculation Module must interface with the Loan/Return Tracking Module to calculate late fees and document them on the member’s account.

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.1.1 The system must have a username and password system for members to log in with.
4.1.2 The system should have a check to differentiate whether the user is staff or member.
4.1.3 Only staff members should be able to view more detailed information about users to ensure privacy.

4.2. Environmental Requirements

4.2.1 System cannot require that any software other than a web browser be installed on user computers.
4.2.2 System must be able to be used on existing library hardware without error.

4.3. Performance Requirements

4.3.1 System must render all UI pages in no more than 9 seconds for dynamic pages. Static pages (HTML-only) must be rendered in less than 3 seconds.
4.3.2 System must be able to run smoothly on low-budget library hardware.