WE MZEE LIBRARY MANAGEMENT SYSTEM

SOFTWARE REQUIREMENTS SPECIFICATION

By Barasa Michael Murunga

15/10/2023

List of Abbreviations

LMS .....Library Management System
SQL .....Structured Query Language
SRS ..... Software Requirements Specification
URL ..... Uniform Resource Locator

Table of Contents

INTRODUCTION

Purpose

The purpose of this Library Management System SRS document is to define and document the software requirements for the development of We Mzee LMS. This document outlines what the application should do (the "what") and provides a clear understanding of its functionality and constraints. It serves as a guide for developers, designers, testers, and other stakeholders involved in the project.

This document is intended for use by various stakeholders involved in the development and testing of the LMS. These stakeholders include, but are not limited to:

1. **Project Managers**: To understand the scope and objectives of the project.
2. **Business Analysts**: To capture and analyze the functional and non-functional requirements.
3. **Designers**: To ensure the user interface and user experience aligns with the requirements.
4. **Developers**: To implement the features and functionality described in this document.
5. **Testers**: To create test cases and verify that the application meets the specified requirements.
6. **Quality Assurance Teams**: To ensure that the application complies with the industry standards and regulations.
7. **Client or Product Owners**: To have a clear understanding of what the application will deliver.

Scope

This document describes the software requirements of the LMS. It covers all aspects of the system, including user interactions, security, performance, and compatibility with various devices and operating systems.

Additionally, it focuses on the "what" of the LMS, not the "how". It specifies the features, functions, and constraints, but does not delve into the technical implementation details. For example, it might specify that the system should support storage of data in an SQL database but does not detail the specific database to be used.

It also serves as the foundation for the subsequent phases of the project, including the design and implementation of the system. The requirements described here will be used by the development team to create a Software Design Description (SDD) document that outlines the technical architecture and implementation details of the mobile banking application.

Software Artefacts to Be Produced

1. **The LMS Application**: This will be available for download and installation on GitHub and will allow We Mzee Library to manage their library.
2. **User Documentation**: A comprehensive user manual will be developed to guide users on how to use the LMS effectively.
3. **System Documentation**: Detailed documentation of the application's architecture, design, configuration will be produced to support future maintenance and enhancements.

What the Current System Will Do

1. **Account Access**: Users will be able to login securely to their accounts using their credentials.

2. **Borrowing of books**: Rules upheld, borrowers will be able to borrow books from the library.
3. **Returning of books**: Borrowers will be able to return borrowed books to the library.
4. **Management of books and borrowers**: Administrators will be able to add, remove and deactivate borrowers and books alike.

What the Current System Will Not Do

1. **Customer Support**: The application will not offer real-time customer support. Users can access contact information for customer support but will not engage in direct calls or chats within the application.
2. **Sending of Notifications**: The application will not offer notification services to borrowers.
3. **Direct Payment**: The application will not offer direct payment of charges via the application.

Consistency with Other Documents

The scope described in this document aligns with the project plan, which outlines the project's timeline, resources, and milestones. The project plan will ensure that the development of the LMS stays on track and within the scope. The requirements specified here are consistent with the project's objectives and goals, as well as any other documents related to the project.

References

Project Plan (Version 1.0) The project plan, version 2.0, provides comprehensive overview of the project's timeline, resource allocation, and milestones. It serves as a reference to ensure the requirements align with the project's objectives and goals.

Documentation of Workshops Outcomes and Decisions (Workshop Documentation, Revision 3):

The document contains the records of workshops and meetings where stakeholders discussed and finalized requirements and decisions related to the mobile banking application. It is essential for understanding how the specific features and functionalities were determined.

Change Requests (Change Request Log, Version 1.1): The Change Request Log, version 1.1, documents any requested changes or modifications to the project's scope, objectives, or requirements. This document is referred to for tracking and assessing potential alterations to the project.

Design and Development Guidelines (Library System Standards, Version 3.8): The Library System Standards, version 3.8 outlines the design and development guidelines, including security standards and best practices, to be followed during the development of the LMS. This document ensures compliance with industry standards and bank-specific requirements.

Repository URL for Software Artifacts: [Link] This URL provides access to the latest versions of software artifacts, system documentation, and other related materials, ensuring that the development team can access the most up-to-date information for reference.

Overview of Document

This LMS SRS document serves as a comprehensive guide to understanding the requirements, scope, and objectives of the development project. It is structured into the following section, each of which plays a vital role in defining and realizing the project:

SYSTEM OVERVIEW

Project Perspective

The LMS is a replacement for the current manual paper-based system. The current system will be phased out a replaced by the one described in this document. This transition aims to enhance user experience, security, and overall efficiency of the system.

System Context

The LMS is a strategic initiative undertaken to enhance the library experience for borrowers. It aligns with the organization's long-term vision of remaining at the forefront of technological advancements in Academia.

The LMS directly addresses several strategic issues:

1. Efficient management of library materials.
2. Efficient management of borrowers.
3. Spontaneous access to library records and resources

General Constraints

Regulatory Compliance

The LMS must adhere to all relevant industry regulations and data security standards. This constraint affects the design and implementation of security measures, as well as testing procedures to ensure compliance.

Data Privacy and Security

Protecting user data and ensuring the security of financial transactions is paramount. The constraint to uphold stringent data privacy and security requirements will influence the design of encryption mechanisms, authentication protocols and security testing.

Platform Compatibility

The application should be compatible with various desktop platforms including MacOS, Windows and Linux. This requirement requires the design and implementation of platform-agnostic features and extensive testing on different devices and operation systems.

Budget and Resource Limitations

The project is subject to budget and resource constraints, which may affect the level of sophistication, feature development, and the timeline for the implementation. These constraints necessitate the trade-off in design and development choices.

Hardware Limitations

The application needs to perform effectively on a variety of mobile devices with varying hardware capabilities. This affects design decisions to optimize performance and requires comprehensive testing across different hardware configurations.

Maintenance and Support

Post-implementation, the application will require ongoing maintenance and support. This constraint involves establishing procedures to maintaining the software, handling updates, and providing user support.

Assumptions

User Adoption: It is assumed that users will readily adopt and use the LMS as their primary means of conducting library related activities. User adoption rates are critical for the project's success.

Technical Infrastructure: It is assumed that the organization's budget on technical infrastructure and resources is sufficient to support the development of the system without any spillovers.

Regulatory Stability: It is assumed that there will be no significant changes in industry regulations during the project's life cycle that would require major revisions to the software's compliance measures.

Stakeholder Availability: It is assumed that key stakeholders, including business analysts, developers, and testers, will be available as needed to collaborate on the project's various phases.

The Current Situation

A local library needs a simple software system to manage its collection of books. The library contains various books authored by different authors and categorized into different genres. The system should allow library staff to perform various operations, such as adding and removing books, searching for books, and checking out books to borrowers. Additionally, it should maintain records of borrowers and their borrowing history.

Entities are as listed below:

Book

Attributes: Title, Author, Genre, ISBN, Available Copies, Total Copies, and Status (Available, Checked Out).

Methods: Display book information.

Author

Attributes: Name, ID, and Biography.

Methods: Display author information.

Genre

Attributes: Name, ID.

Methods: Display genre information.

Borrower

Attributes: Name, ID, Contact Information.

Methods: Display borrower information.

Library

Attributes: List of books, list of authors, list of genres, list of borrowers.

Methods: Add book, remove book, search book, display book list, add borrower, remove borrower, and more.

Processes are as follows:

Add Book

Collect book information (title, author, genre, ISBN, total copies).

Add the book to the library's collection.

Remove Book

Search for a book by title or ISBN.

Remove the book from the library's collection.

Search Book

Search for books by title, author, genre, or ISBN.

Display a list of matching books.

Check Out Book

Search for a book by title or ISBN.

Check out the book to a borrower, reducing the available copies.

Return Book

Search for a book by title or ISBN.

Check the book in, increasing the available copies.

Add Borrower

Collect borrower information (name, contact information).

Add the borrower to the library's records.

Remove Borrower

Search for a borrower by name or ID.

Remove the borrower from the library's records.

Display Borrowing History

Search for a borrower by name or ID.

Display the books checked out by the borrower.

User Interface: You can create a simple command-line interface where the library staff can input commands and interact with the system. Use menus and prompts to guide users through various operations.

Data Storage: Store book, author, genre, and borrower data in text files or simple databases to persist information between program runs.

Additional Features (Optional)

i. Implement error handling for invalid inputs or operations.
ii. Provide the ability to edit book information.
iii. Implement fine-grained permission levels for library staff (e.g., administrator vs. librarian).
iv. Generate reports on the library's collection and borrowing statistics.

SYSTEM ANALYSIS

Overview of the Current System

Author and Book Registration

In the existing library system, the administrator is responsible for registering authors and their books. This process involves inputting comprehensive information about authors and their published works into the library's database. It ensures that the library's collection is properly organized and includes all relevant details about the books' authors.

Borrower Registration

The library accommodates both self-registered and administrator-registered borrowers. When a person wishes to borrow books from the library, they can either register themselves or seek assistance from the administrator. The registration process involves gathering essential personal details and confirming that the potential borrower meets certain eligibility criteria.

Borrowing Limits

To maintain a fair and balanced borrowing system, borrowers are assigned specific limits on the number of books they can have checked out simultaneously. This borrowing limit ensures that borrowers cannot take out additional books until they have returned at least one of their borrowed items, promoting fair use of the library's resources.

Book Return Process

When a borrower returns a book to the library, the administrator plays a vital role. They inspect the condition of the returned book and check whether it has been returned within the stipulated

due date. If the book is found to be in good condition and returned within the expected timeframe, it is officially marked as returned, and no additional charges are imposed.

Fines for Poor Condition or Overdue Books

The library enforces fines for books that are returned in poor condition. These fines are designed to cover the costs of repair or replacement. Additionally, if a book is returned after the due date, the administrator calculates fines based on the number of days overdue and the predetermined charges per day. It's important to note that these fines and charges may be revised periodically to account for changing circumstances.

Administrator Actions

The administrator has the authority to execute a range of actions in the library system. They can delete a book from the library's database, effectively removing it from the collection. They can also deactivate a book temporarily, making it unavailable for borrowing. Furthermore, they have the power to deactivate or suspend a borrower's account, temporarily restricting their borrowing privileges. Finally, the administrator can also remove an author's information from the library's database, ensuring that the database remains up to date and well-organized.

Challenges Faced by the Current System

Manual Data Entry and Storage

The current paper-based system requires manual data entry and storage, resulting in time-consuming processes and potential errors.

Proposed Solution

The new terminal-based system will automate data entry and storage, reducing manual effort and minimizing errors.

Inefficient Search and Retrieval

In the current system, searching for specific books, authors, genres, and borrowers is labor-intensive and time-consuming.

Proposed Solution

The terminal-based system will offer efficient search functionality, allowing staff to quickly locate information through user-friendly text-based menus and prompts.

Data Management and Updates

The current system struggles with data inconsistencies and updates, impacting data accuracy.

Proposed Solution

The new system will streamline data management, enabling staff to easily add, remove, and edit book, author, genre, and borrower information while implementing validation and error handling.

Record Keeping

Maintaining borrowing history and generating reports is a challenge in the current paper-based system.

Proposed Solution

The terminal-based system will automate the recording of borrowing history and offer reporting features to provide staff with insights into the library's collection and borrowing statistics.

User Guidance

Lack of structured user guidance and menus in the current system has led to operational challenges.

Proposed Solution

The new system will feature an intuitive command-line interface with clear menus and prompts to guide users, ensuring a more user-friendly experience.

Data Security

The paper-based records in the current system require manual guarding which is inefficient.

Proposed Solution

The terminal-based system will implement basic security measures, including user authentication and access control, to protect sensitive data and ensure a secure environment.

Functional Requirements

Book Management

- ✓ Add Book: Allow library staff to add new books to the system, including details like title, author, genre, ISBN, total copies, and status.
- ✓ Remove Book: Provide the capability to search for and remove books from the library's collection based on title or ISBN.

Search and Retrieval

- ✓ Search Book: Enable staff to search for books by title, author, genre, or ISBN and display a list of matching books.

- ✓ Display Book List: Allow staff to view a comprehensive list of all books in the library.

Borrowing and Returning

- ✓ Check Out Book: Enable staff to search for a book by title or ISBN, check it out to a borrower, and update available copies.
- ✓ Return Book: Provide the ability to search for a book by title or ISBN, check it in, and update available copies.

Borrower Management

- ✓ Add Borrower: Allow staff to add new borrowers to the system, capturing their name and contact information.
- ✓ Remove Borrower: Provide the capability to search for and remove borrowers from the library's records based on name or ID.

Data Recording and History

- ✓ Display Borrowing History: Enable staff to search for a borrower by name or ID and display the books checked out by the borrower.
- ✓ Data Validation: Implement validation and error handling to maintain data accuracy.

User Guidance

- ✓ User-Friendly Interface: Create an intuitive command-line interface with clear menus and prompts to guide users through various operations.
- ✓ Error Messages: Include informative error messages to assist staff in performing operations accurately.

Data Security

- ✓ User Authentication: Implement user authentication mechanisms to ensure data security.
- ✓ Access Control: Apply access controls to protect sensitive data.

Staff Training and Support

- ✓ Staff Training: Offer comprehensive training and documentation to assist staff during the transition and adaptation to the new terminal-based system.
- ✓ Ongoing Support: Provide support and assistance to address staff concerns during the transition period.

Automation

- ✓ Data Management Automation: Automate data entry and storage processes to reduce manual work and minimize errors.
- ✓ Fine Calculation: Implement fine calculation and due date tracking to automate the fine-related processes.

Reporting

- ✓ Generate Reports: Develop reporting functions that allow staff to generate reports on the library's collection and borrowing statistics.

Non-Functional Requirements

Performance

- ✓ Response Time: The system should respond promptly to user commands, ensuring quick retrieval and updates of data.
- ✓ Scalability: The system should be able to accommodate a growing number of books, authors, borrowers, and transactions without significant degradation in performance.

Usability

- ✓ User-Friendly Interface: The terminal-based interface should be intuitive, making it easy for staff to learn and use the system effectively.
- ✓ Accessibility: The system should be accessible to staff with varying levels of technical expertise.

Reliability

- ✓ System Availability: The system should be available during regular library hours, with minimal downtime for maintenance.
- ✓ Data Integrity: Data should be consistently accurate and reliable. All operations should ensure the integrity of the data.

Security

- ✓ Data Security: Sensitive information, such as borrower details, should be securely stored and transmitted.
- ✓ User Authentication: Ensure that only authorized library staff can access and modify data.
- ✓ Audit Trail: Maintain a log of system activities to monitor and track any unauthorized access or changes.

Data Storage and Management

- ✓ Data Backup: Implement regular data backup procedures to prevent data loss.
- ✓ Data Storage Efficiency: Optimize data storage to minimize the system's storage requirements.

Compliance

- ✓ Regulatory Compliance: Ensure that the system complies with relevant data protection and privacy regulations.

Error Handling

- ✓ Robust Error Handling: The system should provide clear and informative error messages to guide staff in the event of incorrect inputs or system failures.

Interoperability

- ✓ Integration: The system should be capable of integrating with other library systems or external databases for data sharing if necessary.

Staff Training

- ✓ Training Requirements: The system should have minimal training requirements, and the training should be efficient and effective for staff.

Documentation

- ✓ Comprehensive Documentation: Provide comprehensive system documentation for administrators, including user guides and technical documentation.

Transition Period

- ✓ Smooth Transition: During the transition from the paper-based system to the terminal-based system, there should be minimal disruption to library operations.

Resource Utilization

- ✓ Efficient Resource Usage: The system should utilize hardware and software resources efficiently to minimize costs and maximize performance.

Backup and Recovery

- ✓ Disaster Recovery: Plan for disaster recovery to ensure data is not permanently lost in case of system failures.

SYSTEM DESIGN

Class Diagram

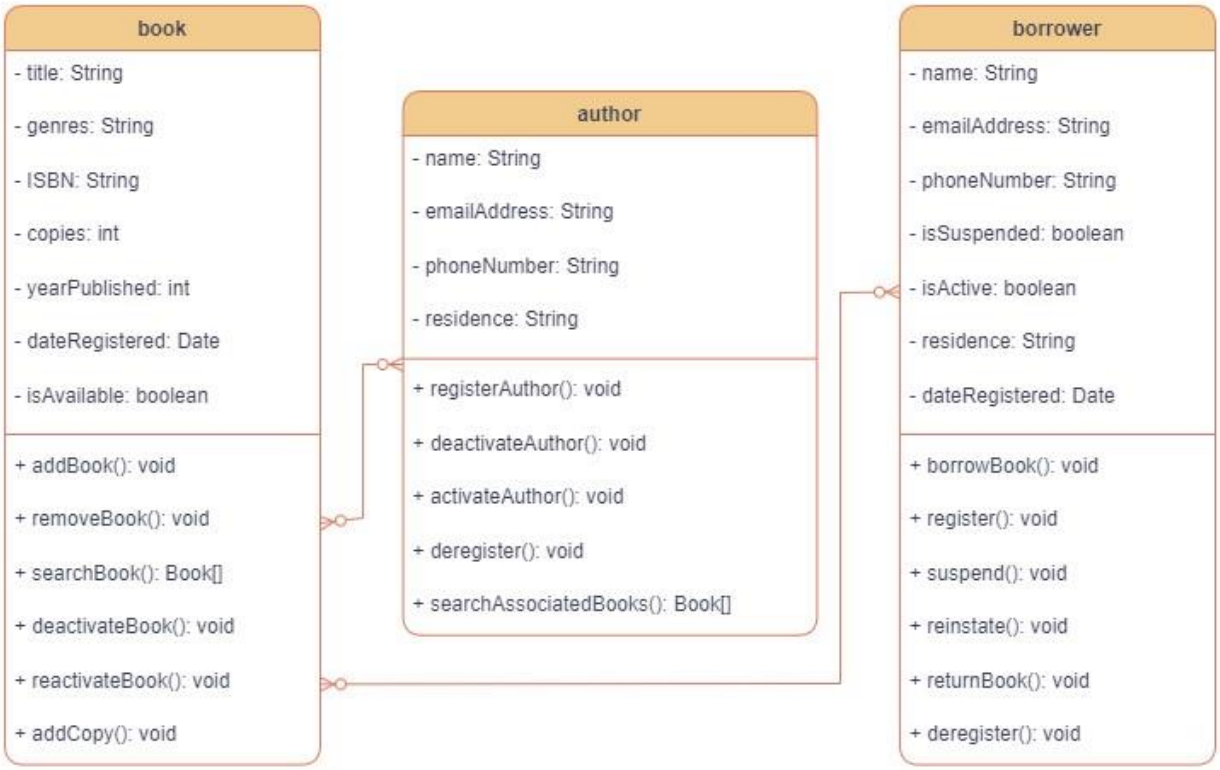


Figure 1: Class Diagram

Entity Relationship Diagram

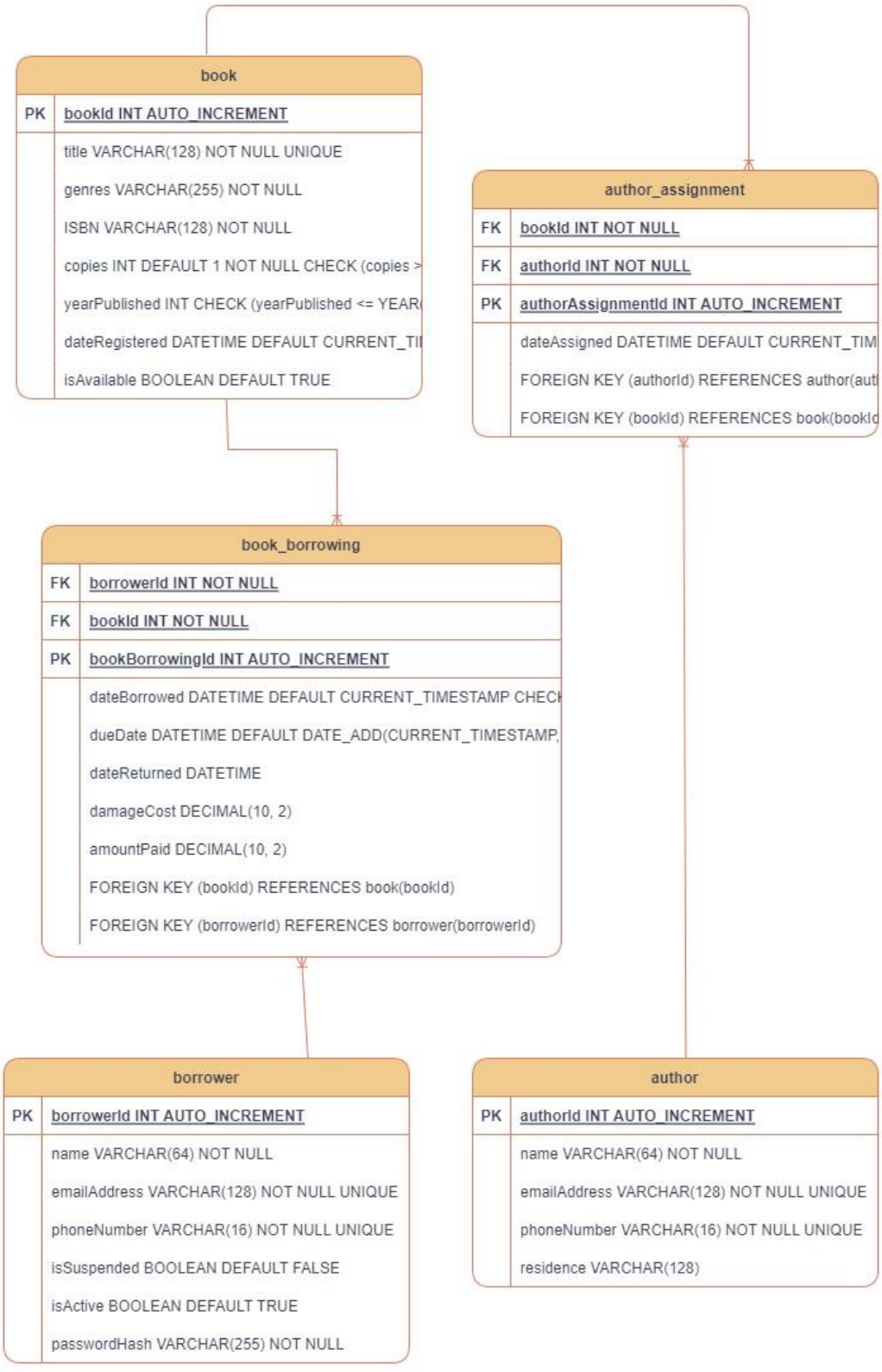

book
PK
bookId INT AUTO_INCREMENT
title VARCHAR(128) NOT NULL UNIQUE
genres VARCHAR(255) NOT NULL
ISBN VARCHAR(128) NOT NULL
copies INT DEFAULT 1 NOT NULL CHECK (copies >
yearPublished INT CHECK (yearPublished <= YEAR(
dateRegistered DATETIME DEFAULT CURRENT_TI
isAvailable BOOLEAN DEFAULT TRUE
author_assignment
FK
bookId INT NOT NULL
FK
authorId INT NOT NULL
PK
authorAssignmentId INT AUTO_INCREMENT
dateAssigned DATETIME DEFAULT CURRENT_TIM
FOREIGN KEY (authorId) REFERENCES author(aut
FOREIGN KEY (bookId) REFERENCES book(bookI
book_borrowing
FK
borrowerId INT NOT NULL
FK
bookId INT NOT NULL
PK
bookBorrowingId INT AUTO_INCREMENT
dateBorrowed DATETIME DEFAULT CURRENT_TIMESTAMP CHEC
dueDate DATETIME DEFAULT DATE_ADD(CURRENT_TIMESTAMP,
dateReturned DATETIME
damageCost DECIMAL(10, 2)
amountPaid DECIMAL(10, 2)
FOREIGN KEY (bookId) REFERENCES book(bookId)
FOREIGN KEY (borrowerId) REFERENCES borrower(borrowerId)
borrower
PK
borrowerId INT AUTO_INCREMENT
name VARCHAR(64) NOT NULL
emailAddress VARCHAR(128) NOT NULL UNIQUE
phoneNumber VARCHAR(16) NOT NULL UNIQUE
isSuspended BOOLEAN DEFAULT FALSE
isActive BOOLEAN DEFAULT TRUE
passwordHash VARCHAR(255) NOT NULL
author
PK
authorId INT AUTO_INCREMENT
name VARCHAR(64) NOT NULL
emailAddress VARCHAR(128) NOT NULL UNIQUE
phoneNumber VARCHAR(16) NOT NULL UNIQUE
residence VARCHAR(128)

Figure 2: ERD Diagram

Use Case Diagram

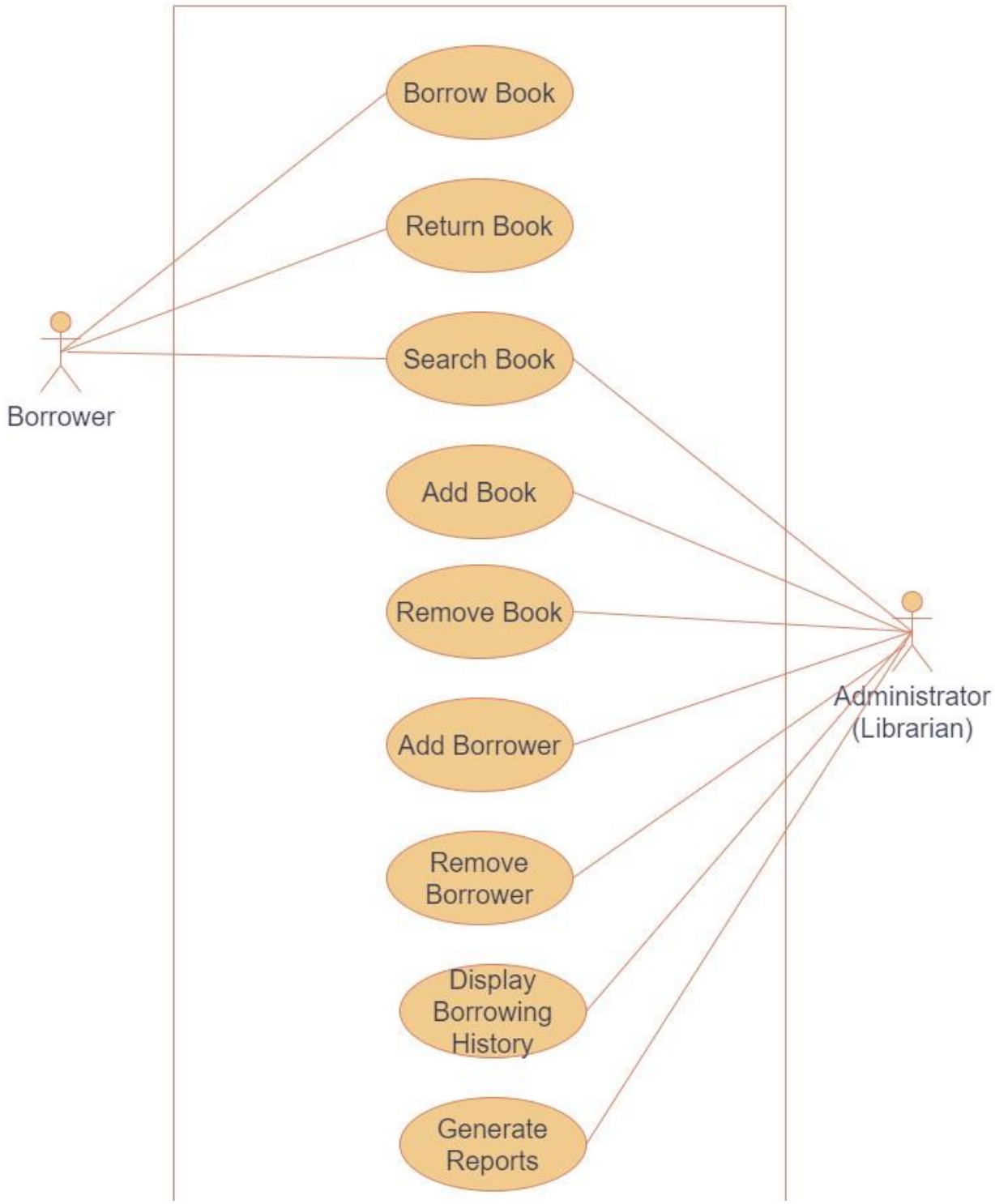
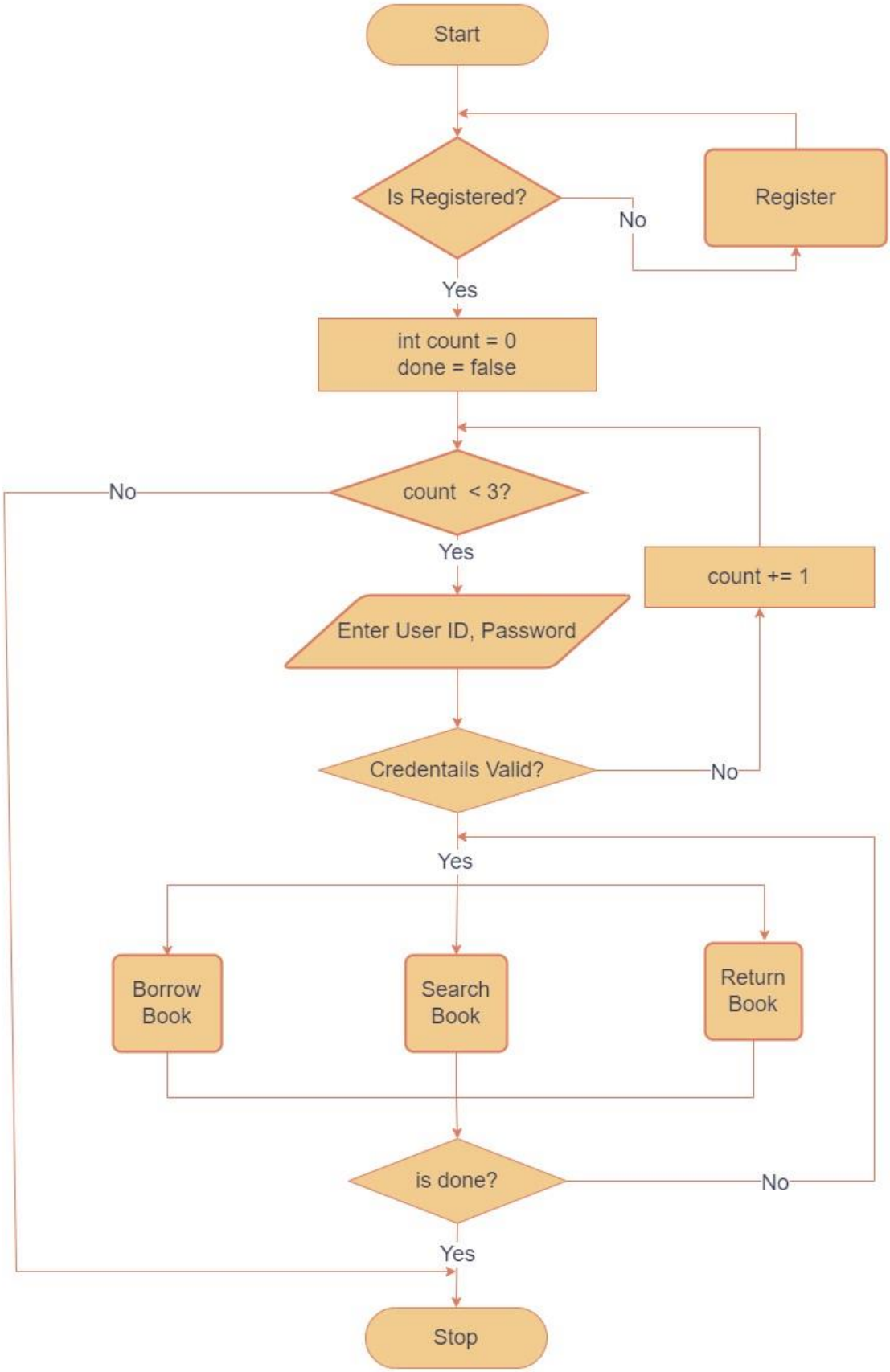


Figure 3: Use Case Diagram


Start
Is Registered?
No
Register
Yes
int count = 0
done = false
count < 3?
No
Yes
count += 1
Enter User ID, Password
Credentails Valid?
No
Yes
Borrow Book
Search Book
Return Book
is done?
No
Yes
Stop

Functional Design

Add Book

Before adding a book, validate that the input data is correct (e.g., ISBN format, non-negative total copies).

Handle exceptions for invalid inputs and provide appropriate error messages.

Input: Title, Genre, ISBN,..., Total Copies

Validate the ISBN format.

Validate that Total Copies is non-negative.

If validation fails, display an error message and do not add the book.

Otherwise:

Create a new Book object with the input information.

Add the new Book object to the database.

Remove Book

Check if the book exists before removing it.

Handle exceptions for book not found and provide appropriate error messages.

Input: Book ID

Search for the book by title or ISBN or Id.

If found, remove the book from the database using the Book ID.

If not found, display an error message.

Search Book

Ensure that the search query is not empty.

Handle exceptions for empty search queries and provide appropriate error messages.

Input: Search Value

Check if the search query is empty.

If it's empty, display an error message.

Otherwise:

Initialize an empty list to store matching books.

Store returned result set into the empty list

Display the list of matching books.

Check Out Book

Ensure the book exists and is available before checking it out.

Ensure the borrower has not exceeded the maximum number of borrowed books

Handle exceptions for unavailable books and provide appropriate error messages.

Input: Borrower ID, Book ID

Search for the book by title or ISBN.

If the book is available and current borrowed books < maximum borrowed books:

Check it out to the borrower.

Update the book's status and available copies.

If the book is not available or maximum borrowed books is reached, display an error message.

Return Book

Ensure the book exists and is checked out before checking it in.

Handle exceptions for books that cannot be checked in and provide appropriate error messages.

Input: Title or ISBN

Search for the book by title or ISBN.

If the book is checked out:

If book is not in good condition:

Charge user maintenance/repair fee

If book is overdue:

Charge user overdue fee

Check it in, update the book's status and available copies.

If the book is not checked out, display an error message.

Add Borrower

Validate the input data for the borrower's name and contact information.

Handle exceptions for invalid inputs and provide appropriate error messages.

Input: Name, Email Address, ..., Phone Number

Validate the name and contact information.

If validation fails, display an error message and do not add the borrower.

Otherwise:

Create a new Borrower object with the input information.

Add the new Borrower object to the database.

Remove Borrower

Check if the borrower exists before removing them.

Handle exceptions for borrower not found and provide appropriate error messages.

Input: Borrower ID

Search for the borrower by name or ID.

If found, remove the borrower from the database.

If not found, display an error message.