# Software Development Plan

## Library Management System (LMS)

## 09/17/2023

Carlos Lopez Padilla

Software Development I CEN 3024C

CRN: 14835

# Introduction

The Library Management System (LMS) will be a tool that will allow a library's staff to manage the collection of books. The faculty will have the ability to add, remove, check out, and check in books, as well as list/display all books currently in the collection.

# Functional Requirements

The library staff have expressed their needs for the LMS tool as follows:

- The program must run on the console.
- To add a book(s) to the collection, the program must read a text file, which will list each book entry in a new line. Each line will be formatted with a unique barcode number, the title, the author's name, and the genre delimited by a comma ("461098, It Ends With Us, Colleen Hoover, Romance").
- To remove a book from the collection, the library staff will enter the book's barcode number or the title in the console. The LMS must find the book and delete it.
- To check out a book from the collection, the library staff will enter the book's title in the console. The LMS must find the book and change its status to "CHECKED OUT" and its due date to 4 weeks from the current date. If the book has already been checked out, the program must display an error message.
- To check a book back into the collection, the library staff will enter the book's title in the console. The LMS must find the book and change its status to "CHECKED IN" and its due date to null.
- The list functionality must display all items currently in the collection.

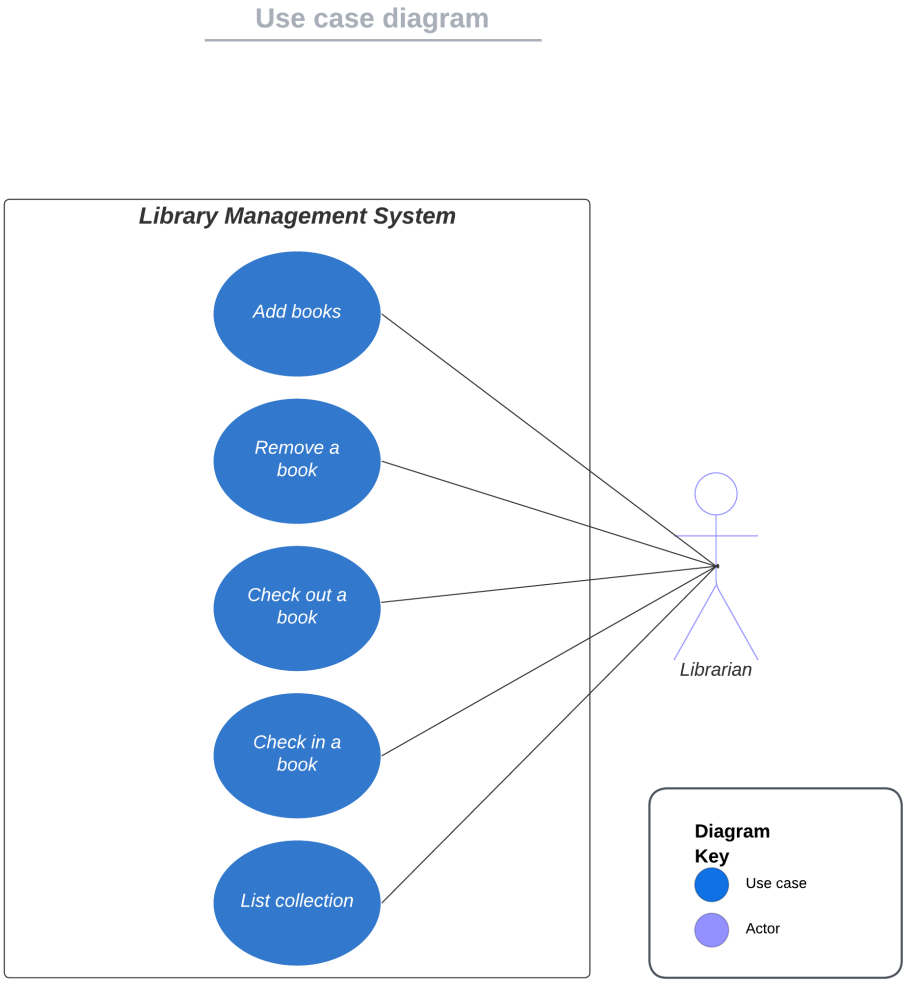
# Non-functional Requirements

- The program must greet the user when it first launches.
- The program must display error messages when invalid input is entered by the user, as well as when it fails to read the text file.
- The program must store the book collection in an ArrayList.
- The program must use a merge sort to traverse through the ArrayList.
- The program must use a switch statement to determine the user's input for the main menu actions.

# Use Cases and User Stories

- As a librarian, I can add books to the library's collection so that more are available to visitors.

- In order to not have unwanted/banned books, I can remove books from the library's collection as a librarian.
- As a librarian, I can check out books from the library collection so visitors can read them.
- In order to return books to the library collection, I can check them back in as a librarian.
- As a librarian, I can list the library collection to display all the books in it.




## Data Requirements

The system must hold the title, author, barcode number, genre, status, and due date as data points for each book in the library's collection.

## User-Interface Design

The LMS will have a main menu with several buttons – “Add books”, “Remove a book”, “Check out a book”, “Check in a book”, “List collection”, and “Exit”.

Every time user input is required the LMS will bring up a text box for the user to enter the necessary data.

Error messages will pop up as text fields.

## Constraints

Currently, the Library Management System’s functionality could be impacted by either invalid input from the user or failure to read the text file. However, this has been considered and a series of actions have been taken to prevent crashes during run time, for example, error handling statements.

* This section is subject to change as further progress in development and testing is made.

## Implementation Plan

The program will be implemented as a console-based Java program. When first run, the program will greet the user and present it with the main menu – “Add books”, “Remove a book”, “List collection”, and “Exit”. Each of these functionalities will have its method, *addBooks*, *removeBook*, *listCollection*, and *exit*, respectively.

addBooks – This method will scan a .txt file containing the details of the book(s) to be added to the collection and insert each entry into the collection utilizing the merge sort algorithm. The contents of said file will be formatted as described previously in the user requirements.

removeBook – This method will prompt the user to type the book ID of the item to be removed and will search for the item using the merge sort algorithm. Once found, the book will be deleted from the collection.

listCollection – This method will simply display to the console all books in the collection.

exit – This method stops the program and exits the console.

Once any of these functionalities are performed, except for “Exit”, the program will return the user to the main menu for further actions.

* This plan is subject to change as further progress in development and testing is made.

## Testing Plan

Each functionality will be tested thoroughly and incrementally as development progresses to guarantee an error-free experience. Test cases varying in size and format will be used to account for invalid inputs.