Faculty of Natural Science
Department of Computer Science
Kristianstad University

Course: DA562B – Grundläggande programmering I C#

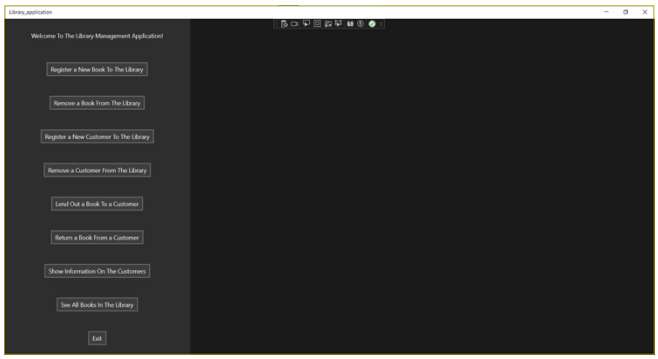
Authors: Anton Ljungkvist

Älta Library

Table of Contents

1. Introduction

This project is a graphical library management system aimed at replicating how libraries track and manage their inventory, users and lending operations. The system allows users to add/remove books, register/remove customers, manage borrow/return/reserve transactions, search records efficiently with filtering by title, author, availability and generating reports with information on all borrowed books and their customers. Developed using C# with a Universal Windows Platform (UWP)-based GUI, the application provides an easy-to-use main menu to navigate between the different library functions.



1.1 Restrictions

- **No Limit on Text Length:** Currently, there's no character limit for customer names, book titles, or authors, which can cause display issues in fixed-size lists. Scroll bars and multi-line wrapping haven't worked well with the current layout.
- **Loose Input Validation for Titles and Authors:** Author names and book titles aren't strictly validated—e.g., names like "J.J.R. Tolkien" and titles like "1984" are accepted. While this hasn't caused functional issues, defining better input rules would improve consistency.

1.2 Possible future improvements

- **Improve Return Book Menu:** Replace manual input with a visual interface: select a customer from a list, then choose which borrowed books to return. Use a popup for confirmation, similar to the customer info screen.
- **Simplify Remove Book/Customer Menus:** Use list views to display removable books and customers for easier selection.
- **Fix UI Layout Issues:** Make the interface responsive so that text, buttons, and fields stay properly aligned when resizing the window.
- **Support Multiple Reservations:** Allow customers to reserve more than one book and cancel reservations if needed.
- **Enable Batch Actions:** Once core menus are improved, add the ability to borrow, return, and reserve multiple books at once.
- **Improve Input Field Labels:** Keep input prompts visible while typing, possibly by using header labels.
- **Use Table View for Lists:** Display items in a structured, table-like format for better readability.
- **Use Custom Exceptions:** Replace system exceptions with custom ones to improve error handling and make debugging easier.
- **Add Database Support:** Implement persistent data storage using a database like SQLite instead of relying on session-based data.

- **Enhance Security Features:** Add password protection for customers, including encryption for better security.
- **Add Customer Search:** Introduce a search function in the customer info menu to quickly find specific users.
- **Enable Book Categories and Search:** Allow categorization of books and let users filter or search by category for easier browsing.

2. Requirements

Table 1 below lists all the requirements for this project.

Table 1 - Requirements

Req. No	Req. Name	Req. Description
1	Add a new book	The application can add a new book.
2	Remove a book	The application can remove a book who is neither borrowed nor reserved.
3	Add new customer	The application can add a new customer.
4	Remove a customer	The application can remove a customer who has no borrowed or reserved books.
5	Lend book to customer	The application can lend books to customers.
6	Return book from customer	A customer who's borrowed a book can return it.
7	Book reservations	A customer can reserve a book that is not available and get exclusive borrowing rights when it's available.
8	Show information on customers	View a list that contains all customers registered in the system with their information. Select a customer to view all their borrowed books.
9	See all books	View a list that contains all books in the library with their information displayed.
10	Search and filter shown books	In Req.No 9; The list can be filtered to find what you are searching for. Filter by title, author and availability.
11	Book report	In Req.No 9; Generate a report that shows all borrowed books, the borrowing customer and the reserving customer if there is one.
12	Late return fees	If a customer borrows a book for longer than the allowed time, a fee will be given upon returning the book. The fee amount is based on the overdue time.

3. Design and Implementation

3.1 Book

The book class is responsible for modeling a single book in the library. It contains attributes such as the title, author, ISBN number, availability status, reserved status and the time when it was borrowed. Methods and properties in this class allow the system to change its status when borrowed, returned or reserved as well as get its details. This class is primarily used by the LibraryLogic class to manage inventory and track borrowing and reserving activities.

3.2 Customer
The customer class is responsible for modeling a single customer in the library. It contains attributes such as the name, ID number, a list of borrowed books and the ISBN number of their reserved book. Methods and properties in this class allow the system to update a customer's borrowed books, book reservation and retrieve customer information. This class is primarily used by the LibraryLogic class to manage library customers and track borrowing and reserving activities.

3.3 LibraryLogic
The library logic class is responsible for managing all logistical actions of the library. It contains attributes such as a list of all registered books and a list of all registered customers in the library. Methods in this class allow the system to manage, change, edit and update the books and customers of the library in different ways. This class is primarily used by the MainPage class to execute any action the user wants to make regarding the back-end logical side of the library.

3.4 MainPage
The main page class is responsible for managing and interpreting all user input for the application. This class is directly connected to the front-end GUI of the application and contains all the methods used by the graphical interface. As the class responsible for managing and interpreting all user input, it works alongside the LibraryLogic class as its interpreter that translates the inputs to usable data. This class is also responsible for the applications exception handling to simplify the communication of wrong inputs or invalid actions to the user.

3.5 UML-Diagram
The UML-Diagram Is located as a file in the folder.

4. Summary and Conclusion

This project is a graphical library management system built with C# and UWP, designed to handle core library tasks like managing books, customers, lending, returning, and reserving. It meets all key requirements and offers an intuitive interface supported by a well-structured backend.
Core classes (Book, Customer, LibraryLogic, and MainPage) ensure clear separation between logic and UI, making the system easy to maintain and extend.
While the system works well, improvements like better input validation, database integration, enhanced UI, and added security features would increase its reliability and user experience.
Overall, the project meets its goals and provides a strong base for future development.