Project Title
Library Management System

Introduction to Problem Solving (CSE1021)

Name : Sai Kumar
Reg n.o : 25MIB10027
Submitted to : Dr. Dheresh soni
Slot :A14+D11+D12

1.Introduction

The Library Management System is a Python-based application designed to handle essential library operations such as adding books, borrowing books, returning books, and searching for books. The system uses lists and dictionaries to store book data and offers a clean, menu-driven interface for easy use.
It is ideal for beginners learning Python and basic software development concepts.

2. Problem Statement

Managing books manually in a library is time-consuming and leads to errors such as misplaced records, difficulty in searching, and wrong availability status.
The goal is to build a simple digital solution to automate basic library tasks.

3. Functional Requirements

These are features the system must perform:

1. Add new books with title and author.

2. View all books with their current availability status.

3. Borrow a book (mark as not available).

4. Return a book (mark as available).

5. Search books by title or author keyword.

6. Exit the application safely.

4. Non-Functional Requirements

1. Usability – The system should be simple and easy to use for beginners.

2. Performance – The response time should be fast for all operations

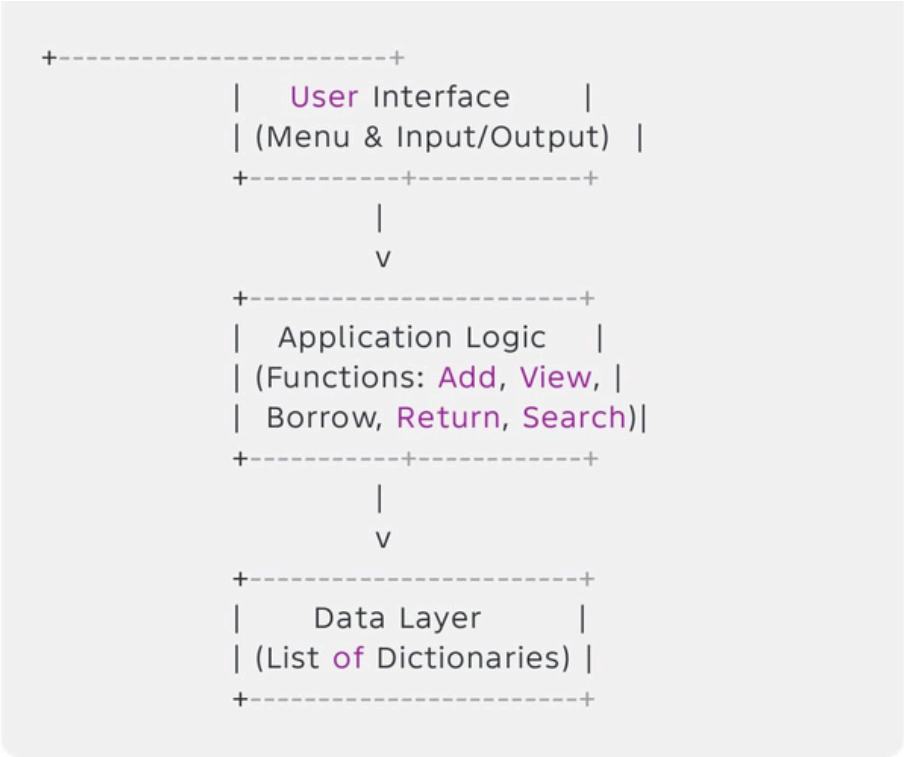
3. Reliability – Must correctly update the availability status of books.

4. Scalability – Should allow unlimited number of books.

5. Portability – Works on Windows, Mac, or Linux with Python 3.

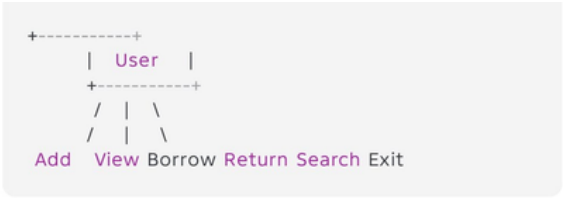
6. Maintainability – Code is modular and easy to extend.

6. System Architecture

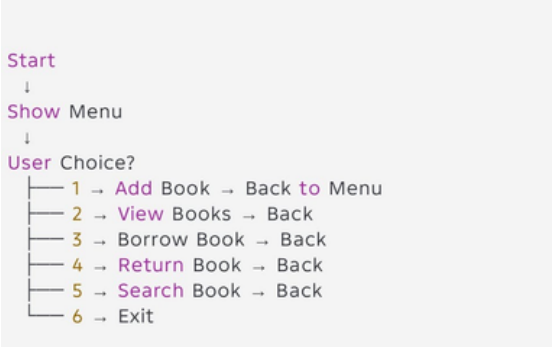


7. Design Diagrams

a) Use Case Diagram



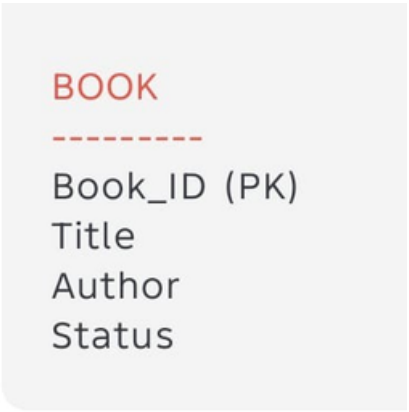
b) Workflow Diagram



c) Sequence Diagram

d) Class/Component Diagram

e) ER Diagram (if storage used)



8. Design Decisions & Rationale

1**. List & Dictionary chosen** — easiest structure for beginners.

2. **Menu-based interface** — reduces complexity.

3. **No database** — makes it simple and portable.

4 **.separate function** — improves readability and debugging.

9. Implementation Details

Programming Language: Python 3

Data Structure: List of dictionaries

Modules Used: None (only built-in features)

User interacts using simple CLI prompts (input())

Each book stored as:

{"title": ..., "author": ..., "available": True}

9. Implementation Details

Programming Language: Python 3

Data Structure: List of dictionaries

Modules Used: None (only built-in features)

User interacts using simple CLI prompts (input())

Each book stored as:

{"title": ..., "author": ..., "available": True}

10. Screenshots / Results

```
----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 1
Enter book title: Harry potter
Enter author name: J.K. Rowling
Book added successfully!

----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 1
Enter book title: Don Quixote
Enter author name: Miguel de Cervantes
Book added successfully!

----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 2

----- All Books -----
1. Harry potter by J.K. Rowling - Available
2. Don Quixote by Miguel de Cervantes - Available
```

```
----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 3

----- All Books -----
1. Harry potter by J.K. Rowling - Available
2. Don Quixote by Miguel de Cervantes - Available
Enter book number to borrow: 1
Book borrowed successfully!

----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 2

----- All Books -----
1. Harry potter by J.K. Rowling - Not Available
2. Don Quixote by Miguel de Cervantes - Available

----- LIBRARY MENU -----
1. Add a new book
2. View all books
3. Borrow a book
4. Return a book
5. Search book
6. Exit
Enter your choice (1-6): 5
Enter title or author to search: Harry potter

Search Results:
Harry potter by J.K. Rowling - Not Available
```

11. Testing Approach

Types of Testing Performed:

Functional Testing (each option tested)

Boundary Testing (empty list, invalid book number)

Input Validation (incorrect input)

Integration Testing (functions working together)

Test Cases:

Test Case Input Expected Output

Add book Title + Author Book added
Borrow invalid number 999 Error message
Search “Harry” Keyword Matching books

12. Challenges Faced

Handling invalid user inputs (like wrong numbers).

Maintaining book status (available/not available).

Ensuring the menu loop runs properly without crashing.

13. Learnings & Key Takeaways

Learned how to use lists and dictionaries effectively.

Gained understanding of modular programming using functions.

Understood basic software architecture and diagrams.

Learned how console-based applications work.

14. Future Enhancements

Add permanent file storage (CSV/JSON).

Add book IDs and user ID system.

Add login system for admin & users.

Implement GUI using Tkinter.

Add due date + fine calculation.

Convert into a full web app (Flask/Django).

15. References

Python Documentation – https://docs.python.org

Online Tutorials – W3Schools, GeeksForGeeks

Class Notes & Materials