University Institute of Computing

Chandigarh University

Gharuan , Mohali(Punjab)

COMPUTING APTITUDE MINI PROJECT

ON

Library Management System

Submitted by:

Jagpreet Singh & Keshav katoch

UID- 23BCA10376 & 23BCA10016

Submitted to:

Ms. Priyanka Arora

Signature:

Abstract

• A C++ console-based Library Management System.
• Automates library tasks: add, search, issue, and return books.
• Maintains records using file I/O for persistence.
• Demonstrates OOP, file handling, and modular design.

Introduction

• Manual record-keeping in libraries is slow and error-prone.
• The system simplifies and automates core operations.
• Practical mini project demonstrating C++ programming skills.
• Includes classes, file handling, and structured logic.

Problem statement & Objectives

Problem: Managing library books and members manually is inefficient.

Objectives:
• Manage books and members efficiently.
• Automate issuing and returning of books.
• Store and retrieve data from text files.
• Demonstrate object-oriented and modular design.

System Design / Approach

Architecture Overview:
• Data Layer: File-based storage (books.txt, members.txt).
• Domain Layer: Classes – Book, Member, Library.
• Presentation Layer: Menu-driven console UI.

Flow:
Start → Load Data → Menu → Action → Update Data → Save → Exit

Class Diagram (Simplified)

Library 1 --- * Book
Library 1 --- * Member

Book: id, title, author, total, available
Member: id, name, issued_books

Key Classes: Book, Member, Library

Flowchart

Start
↓
Load data from files
↓
Display Menu
↓
Perform selected operation
↓
Update records
↓
Save changes
↓
Exit

Implementation

Modules:
• book.h / book.cpp — Book class and methods.
• member.h / member.cpp — Member data and operations.
• library.h / library.cpp — Manages operations, handles file I/O.
• main.cpp — Menu-driven interface.

Concepts Used:
• Classes and Objects

• File Handling
• Vectors and Maps (STL)

Key Code Snippet

```
bool issueBook(int bookId, int memberId) {
   if (!books.count(bookId) || !members.count(memberId)) return false;
   Book &b = books[bookId];
   Member &m = members[memberId];
   if (b.available <= 0) return false;
   b.available--; m.issued.push_back(bookId);
   cout << "Book issued"; return true;
}
```

Highlights:
• Ensures availability before issuing.
• Updates both book and member data.

Results

C++ code of library management system:

```
#include <iostream>

#include <string>

using namespace std;


struct Book {

   int id;

   string title;

   string author;

   bool isIssued;

};


class Library {

   Book books[100];
```

```
    int count;

public:
    Library() {
        count = 0;
    }

    void addBook() {
        cout << "\nEnter Book ID: ";
        cin >> books[count].id;
        cin.ignore();
        cout << "Enter Book Title: ";
        getline(cin, books[count].title);
        cout << "Enter Author Name: ";
        getline(cin, books[count].author);
        books[count].isIssued = false;
        count++;
        cout << "Book added successfully!\n";
    }

    void showAllBooks() {
        if (count == 0) {
            cout << "\nNo books in library!\n";
            return;
        }
        cout << "\n--- Library Books ---\n";
```

```
    for (int i = 0; i < count; i++) {
        cout << "ID: " << books[i].id
             << " | Title: " << books[i].title
             << " | Author: " << books[i].author
             << " | Status: " << (books[i].isIssued ? "Issued" : "Available") << endl;
    }
}

void issueBook() {
    int id;
    cout << "\nEnter Book ID to issue: ";
    cin >> id;
    for (int i = 0; i < count; i++) {
        if (books[i].id == id) {
            if (!books[i].isIssued) {
                books[i].isIssued = true;
                cout << "Book issued successfully!\n";
            } else {
                cout << "Book already issued!\n";
            }
            return;
        }
    }
    cout << "Book not found!\n";
}
```

```
    void returnBook() {
        int id;
        cout << "\nEnter Book ID to return: ";
        cin >> id;
        for (int i = 0; i < count; i++) {
            if (books[i].id == id) {
                if (books[i].isIssued) {
                    books[i].isIssued = false;
                    cout << "Book returned successfully!\n";
                } else {
                    cout << "Book was not issued!\n";
                }
                return;
            }
        }
        cout << "Book not found!\n";
    }
};

int main() {
    Library lib;
    int choice;

    while (true) {
        cout << "\n--- LIBRARY MANAGEMENT SYSTEM ---\n";
        cout << "1. Add Book\n";
```

```
        cout << "2. Show All Books\n";
        cout << "3. Issue Book\n";
        cout << "4. Return Book\n";
        cout << "5. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1: lib.addBook(); break;
            case 2: lib.showAllBooks(); break;
            case 3: lib.issueBook(); break;
            case 4: lib.returnBook(); break;
            case 5: cout << "Exiting..."; return 0;
            default: cout << "Invalid choice!\n";
        }
    }
}
```

Output:

```
--- LIBRARY MANAGEMENT SYSTEM ---
1. Add Book
2. Show All Books
3. Issue Book
4. Return Book
5. Exit
Enter your choice: 1

Enter Book ID: 2910
Enter Book Title: Mohenjadaro
Enter Author Name: Jagpreet
Book added successfully!

--- LIBRARY MANAGEMENT SYSTEM ---
1. Add Book
2. Show All Books
3. Issue Book
4. Return Book
5. Exit
Enter your choice: 2

--- Library Books ---
ID: 2910 | Title: Mohenjadaro | Author: Jagpreet | Status: Available
```

```
--- LIBRARY MANAGEMENT SYSTEM ---
1. Add Book
2. Show All Books
3. Issue Book
4. Return Book
5. Exit
Enter your choice: 3

Enter Book ID to issue: 2910
Book issued successfully!

--- LIBRARY MANAGEMENT SYSTEM ---
1. Add Book
2. Show All Books
3. Issue Book
4. Return Book
5. Exit
Enter your choice: 4

Enter Book ID to return: 2910
Book returned successfully!
```

```
--- LIBRARY MANAGEMENT SYSTEM ---
1. Add Book
2. Show All Books
3. Issue Book
4. Return Book
5. Exit
Enter your choice: 5
Exiting...
```

Conclusion

• Functional Library Management System implemented in C++.
• Demonstrated OOP, file handling, and persistence.
• Enhances programming and logical design skills.

Future Enhancements:
• GUI interface
• Database integration (MySQL/SQLite)
• Add fine management and due dates

References

1. Bjarne Stroustrup, The C++ Programming Language
2. https://en.cppreference.com/
3. Online tutorials and documentation

GITHUB LINK:

https://github.com/JagpreetSingh752/Library_management