CA 3: Experiential Learning

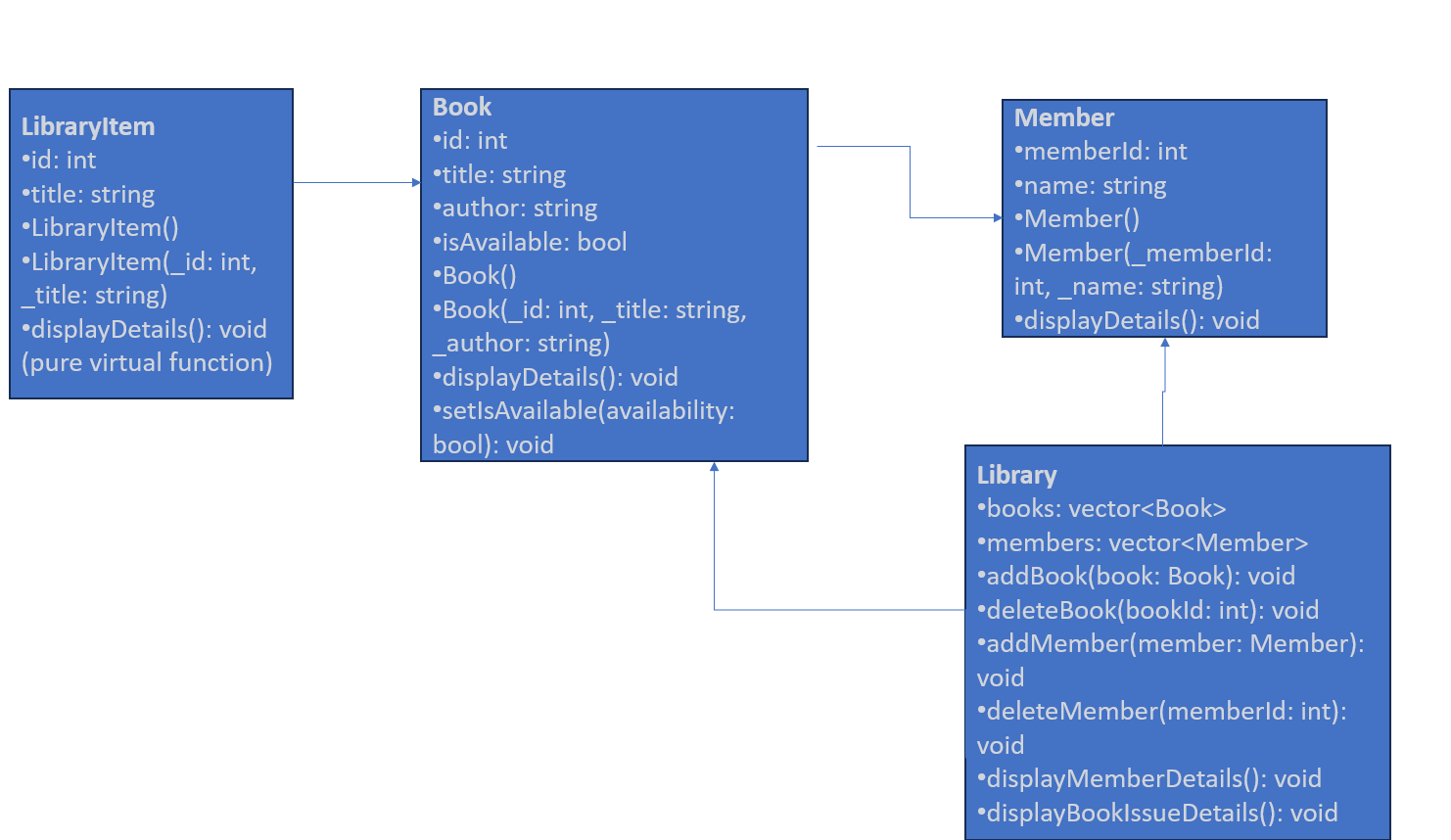
Group Members:

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | PRN | Name of Student | Mail id |
| 1 | 22070122185 | SARTHAK SAHU | sahu.sarthak.btech2022@sitpune.edu.in |
| 2 | 22070122215 | SONU GARG | sonu.garg.btech2022@sitpune.edu.in |
| 3 | 22070122223 | SUJAY SAKHARE | sujay.sakhare.btech2022@sitpune.edu.in |

Problem Statement: The existing library management system lacks efficiency and organization in book management, user registration, and book borrowing. The system heavily relies on manual processes, leading to errors in book availability and delays in user requests. Additionally, it faces issues related to user accessibility, data analysis.

Explanation: There are significant issues with the current library management system. The system is inefficient and disorganized in managing various aspects, such as book resources, user registration, and the book borrowing process. These inefficiencies are largely due to the heavy reliance on manual procedures, which result in errors and delays in providing books to users. Moreover, the system faces challenges in terms of user accessibility, data analysis, security, and privacy concerns. In summary, the library is grappling with a range of operational problems that hinder its ability to provide a seamless and efficient experience to both staff and library users. Hence, we need a computerized system for efficient library management

Class Diagram:



Code snippets:

#include <iostream>

#include <vector>

using namespace std;

class LibraryItem {

public:

int id;

string title;

LibraryItem() {}

LibraryItem(int \_id, string \_title) : id(\_id), title(\_title) {}

virtual void displayDetails() = 0;

};

class Book : public LibraryItem {

private:

string author;

bool isAvailable;

public:

Book() {}

Book(int \_id, string \_title, string \_author) : LibraryItem(\_id, \_title), author(\_author), isAvailable(true) {}

void displayDetails() {

cout << "Book ID: " << id << endl;

cout << "Title: " << title << endl;

cout << "Author: " << author << endl;

cout << "Availability: " << (isAvailable ? "Available" : "Not available") << endl;

}

void setIsAvailable(bool availability) {

isAvailable = availability;

}

};

class Member {

public:

int memberId;

string name;

Member() {}

Member(int \_memberId, string \_name) : memberId(\_memberId), name(\_name) {}

void displayDetails() {

cout << "Member ID: " << memberId << endl;

cout << "Name: " << name << endl;

}

};

class Library {

private:

vector<Book> books;

vector<Member> members;

public:

void addBook(Book book) {

books.push\_back(book);

}

void deleteBook(int bookId) {

for (int i = 0; i<books.size(); i++) {

if (books[i].id == bookId) {

books.erase(books.begin() + i);

break;

}

}

}

void addMember(Member member) {

members.push\_back(member);

}

void deleteMember(int memberId) {

for (int i = 0; i<members.size(); i++) {

if (members[i].memberId == memberId) {

members.erase(members.begin() + i);

break;

}

}

}

void displayMemberDetails() {

for (int i = 0; i<members.size(); i++) {

members[i].displayDetails();

cout << "-------------------------" << endl;

}

}

void displayBookIssueDetails() {

for (int i = 0; i<books.size(); i++) {

books[i].displayDetails();

cout << "-------------------------" << endl;

}

}

};

void issueBook(Book& book, Member& member) {

book.setIsAvailable(false);

cout << "Book has been issued to member: " << member.memberId << endl;

}

int main() {

Library library;

while (true) {

cout << "Library Management System" << endl;

cout << "1. Add Book" << endl;

cout << "2. Add Member" << endl;

cout << "3. Issue Book" << endl;

cout << "4. Delete Book" << endl;

cout << "5. Display Member Details" << endl;

cout << "6. Display Book Issue Details" << endl;

cout << "7. Exit" << endl;

cout << "Enter your choice: ";

int choice;

cin >> choice;

switch (choice) {

case 1: {

int id;

string title, author;

cout << "Enter Book ID: ";

cin >> id;

cin.ignore();

cout << "Enter Book Title: ";

getline(cin, title);

cout << "Enter Author: ";

getline(cin, author);

Book newBook(id, title, author);

library.addBook(newBook);

break;

}

case 2: {

int memberId;

string name;

cout << "Enter Member ID: ";

cin >> memberId;

cin.ignore();

cout << "Enter Member Name: ";

getline(cin, name);

Member newMember(memberId, name);

library.addMember(newMember);

break;

}

case 3: {

int bookId, memberId;

cout << "Enter Book ID to issue: ";

cin >> bookId;

cout << "Enter Member ID: ";

cin >> memberId;

break;

}

case 4: {

int bookId;

cout << "Enter Book ID to delete: ";

cin >> bookId;

library.deleteBook(bookId);

break;

}

case 5:

library.displayMemberDetails();

break;

case 6:

library.displayBookIssueDetails();

break;

case 7:

return 0;

default:

cout << "Invalid choice. Please try again." << endl;

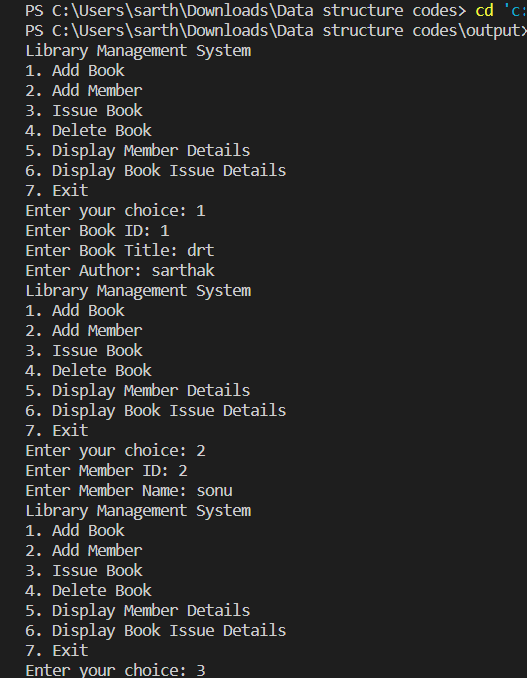
}

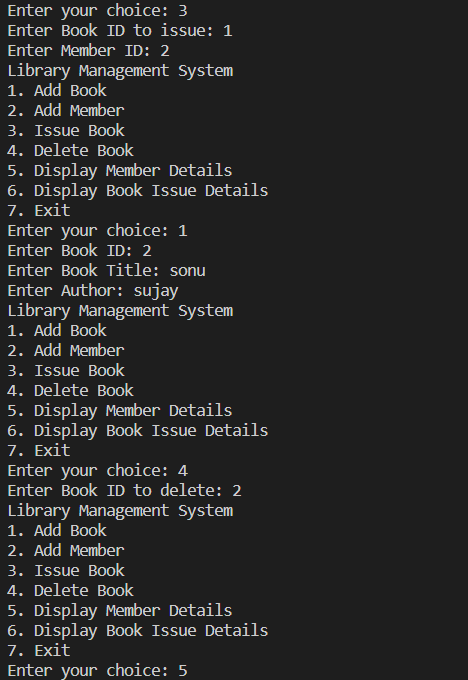
}

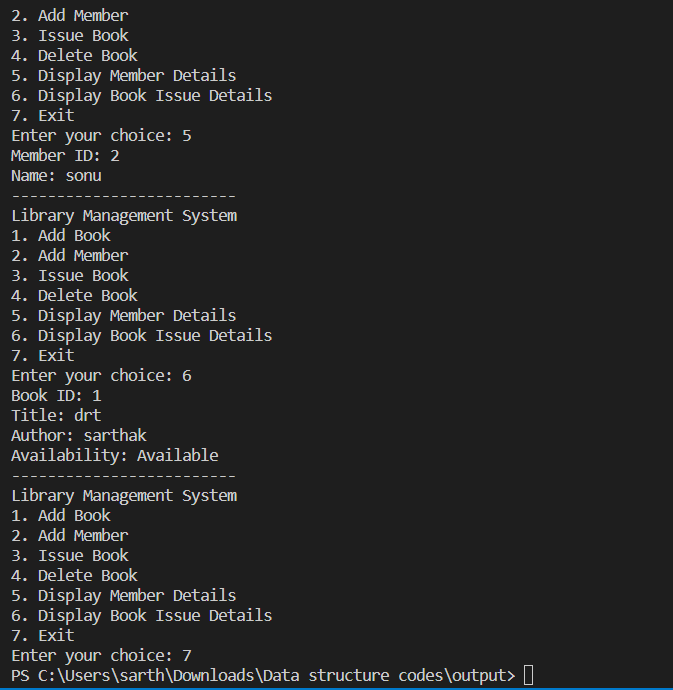
return 0;

}

Input/Output:







Github repository link:

**https://github.com/S1aERT/Library-management-system/blob/main/Library%20management%20system.txt**