SHIFA TAMEER-E-MILLAT UNIVERSITY

Department of Computing

FINAL SEMESTER PROJECT

Name: MUHAMMAD YOOSHA BIN NOON

Roll No. BSAI-24F-0128

Section: BSAI(1A)

NAME: IQRA BATOOL

Roll No. BSAI-24F-0110

Section: BSAI(1B)

Total Marks: 10

Due Date: 3rd/2/2025

Department of Computing
Shifa Tameer-e-Millat University Islamabad
www.stmu.edu.pk

Contents

Acknowledgement

I would like to express my heartfelt gratitude to my professors, Mr. Talha and Professor Tanvir, for their unwavering support and guidance. Their dedication to teaching programming fundamentals was invaluable, and their readiness to assist and their consistent hard work made a significant impact.

I also want to extend my sincere thanks to my incredible team members, Iqra Batool and myself Muhammad Yoosha Bin Noon, for putting in the maximum hours we could and for our relentless problem-solving abilities. Together, we have overcome numerous challenges and made remarkable progress.

1. INTRODUCTION

1.1 Introduction

We are developing a library management system designed to simplify and enhance the way a library operates. This system will implement various membership types to offer tailored benefits and discounts to students, making the library experience more appealing and accessible. By storing comprehensive data on both books and students, the system aims to maintain accurate and up-to-date records.

The system will feature distinct login modules for both staff and students. Staff members will have access to functionalities such as adding new books, managing student data, and viewing all records. Students, on the other hand, will be able to issue and return books based on their membership type, benefiting from the discounts associated with their membership.

1.2 Problem Statement

Managing a library's vast collection of books and student memberships manually is a daunting task. It often leads to errors, inefficiencies, and frustrations for both library staff and users. The lack of a proper system can result in misplaced books, inaccurate records, and difficulties in tracking overdue items. Additionally, managing various membership types and applying appropriate discounts and loan periods manually can be quite challenging.

1.3 Objectives

Our library management system aims to:

1. **Automate Book and Membership Management**: Implement a seamless system to automate the addition, issuance, and return of books, as well as the management of student memberships.

2. **Implement Membership Benefits**: Provide different membership types (e.g., Executive, Premium, Gold) with corresponding benefits such as discounts.
3. **Accurate Record Keeping**: Ensure accurate and up-to-date records of all books and student memberships by reading from and writing to external files.
4. **Streamline Staff and Student Access**: Create distinct login modules for staff and students, allowing each group to access relevant functionalities.
5. **Improve User Experience**: Enhance the overall experience for both library staff and users by making the system user-friendly, efficient, and reliable.

2. LITERATURE REVIEW / THEORY

2.1 Background of Study

The inspiration for developing this library management system came from a fascinating exploration of final year projects displayed on LinkedIn. The innovative solutions and practical applications showcased by others left a lasting impression, motivating us to create something equally useful and impactful. Witnessing how technology can transform traditional systems encouraged us to take on the challenge of modernizing library management.

2.2 Understanding of the Problem

Our current library at Shifa Tameer-e-Millat University is outdated, lacking modern methods for efficient management. There are no proper membership types or discounts available to students, which can make the library experience less engaging and beneficial. This issue is not unique to our university but is prevalent in many libraries across Pakistan. Therefore, our project aims to introduce new strategies to enhance the management system, making it more efficient, user-friendly, and aligned with contemporary needs.

3. METHODOLOGY

3.1 Project Work & Research Methodology

The development of our library management system involves a systematic approach that combines project work and research. Our methodology includes:

- **Literature Review**: Analyzing existing library management systems to understand their strengths and weaknesses. This helped us identify the unique features and improvements we wanted to incorporate into our system.
- **Requirements Gathering**: Collecting requirements from potential users, including students and library staff, to ensure the system meets their needs. This involved surveys, interviews, and observation.
- **Learning Foundations**: The foundational concepts for this project, such as structures for adding books and student data, were learned in our programming classes at Shifa Tameer-e-Millat University. We also covered functions, loops, and basic file handling, which were particularly useful for this project.
- **Project Modules**: Project modules were decided group-wise, with each member responsible for their own module. Iqra Batool worked on Student login calculating

discounts and membership types also storing data for book issuing and returning using complex data structures like functions, structures, loops and conditional statements, while Muhammad Yoosha Bin Noon worked on staff portal saving developing code to create a robust system to keep an updated file of books and student data using structures and functions, loops and conditional statements. We researched and brainstormed module ideas using Google Search, ensuring we included all necessary functionalities for our system.

- **Design and Implementation**: Using the gathered requirements, we designed the system architecture and developed the code. The system is written in C++ and involves modules for book and student management, membership types, and discount calculations.
- **Testing and Debugging**: Ensuring the system works as intended through rigorous testing and debugging processes.

3.2 Tools

To develop and implement our library management system, we used the following tools:

- **Programming Language**: C++
- **Development Environment**: Integrated Development Environment (IDE) such as Dev c++.
- **File Handling**: If stream and OFstream for reading from and writing to external files
- **Data Structures**: Structs for managing book and student data.

3.3 Activities/Gantt Chart and Milestone

Activity Duration Start Date End Date Milestone

Activity	**Duration**	**Start Date**	**End Date**	**Milestone**
Requirements Gathering	2 weeks	01/01/2025	14/01/2025	Requirements Document
Basic Login for Student and Staff	1 week	14/01/2025	20/01/2025	Basic
Structures for Book and Student Data	1 week	21/01/2025	27/01/2025	Structures for Data Created
Membership Types and Discounts	1 week	28/01/2025	02/02/2025	Discounts and Membership Types Implemented
File Handling for Book Data	1 week	28/01/2025	02/02/2025	File Handling Implemented

System Design	2 weeks	28/01/2025	02/02/2025	System Architecture Design
Coding and Implementation	2 weeks	28/01/2025	02/02/2025	Initial System Prototype
Testing and Debugging	2 weeks	28/01/2025	02/02/2025	Bug-Free System
Documentation	1 week	28/01/2025	02/02/2025	User and Developer Docs
Final Review and Submission	1 week	28/01/2025	02/02/2025	Completed Project Submission

4. RESULT & DISCUSSION

4.1 STEP 1: Data Structures and Their Implementation

In our library management system, we utilized specific data structures to efficiently manage books and student records. The key data structures include:

- **Book Structure**: This structure is designed to store essential details about each book, including its name, author, and ISBN number.
- **Student Structure**: This structure holds the details of each student, such as their name, father's name, membership type, and student ID. Both structures are implemented using arrays to handle multiple entries, ensuring scalability and efficient data management.

4.3 STEP 3: Language Used in the Project

The library management system is developed using C++, leveraging its robust features for system-level programming and efficient memory management. Key aspects of the code include:

- **Functions and Loops**: Utilized for various operations like issuing and returning books, calculating discounts, and managing membership types.
- **File Handling**: Implemented for reading from and writing to external files to maintain persistent records of books and students.

4.2 STEP 2: Architecture and Working of the Project

Our library management system is designed with two distinct login modules: one for staff and one for students. Each module directs the user to specific functionalities based on their role.

Staff Login

When a staff member logs in, the system performs the following steps:

1. **Login Authentication**:
    - The staff member is prompted to enter a password. The system verifies this password against a predefined value.
    - If the password is correct, the staff member is granted access. Otherwise, they are given multiple attempts to log in.
2. **Menu Options**:
    - Once authenticated, the staff member is presented with a menu of options:
        - Add books to the shelf
        - Add student data
        - View all students
        - View all books
3. **Book Management**:
    - **Adding Books**: Staff can add new books by entering the book name, author name, and ISBN number. This data is stored in the `books` array and written to an external file using file handling techniques.
    - **Viewing Books**: Staff can view the list of all books by reading from the external file and displaying the data.
4. **Student Management**:
    - **Adding Students**: Staff can add new students by entering the student's name, father's name, student ID, and membership type. This data is stored in the students array and written to an external file using file handling techniques.
    - **Viewing Students**: Staff can view the list of all students by reading from the external file and displaying the data.
5. **Loop and Function Utilization**:
    - The system uses loops to iterate through the arrays of books and students, ensuring efficient data management.
    - Functions are utilized to encapsulate specific tasks, such as reading and writing data, and calculating discounts based on membership types.

ADVANTAGES:

1. CAN STORE DATA OF STUDENTS.
2. CAN STORE DATA OF BOOKS.
3. CAN GIVE DISCOUNTS ON TYPES OF MEMBERSHIPS THUS PRESENTING A NEW WAY IN THE LIBRARY MANAGEMENT SYSTEM.
4. HAVE SEPARATE LOGIN FOR STAFF AND STUDENTS.

DISADVANTAGES:

1. UNABLE TO VERIFY WHETHER THE STUDENT WHO IS LOGGIN IN IS PRESENT IN THE FILE OR NOT.
2. DOESN'T HAVE UNIQUE LOGIN PASSWORDS FOR STUDENTS AND FILE CANNOT BE ACCESSED FOR THIS FUNCTION.
3. THE PROGRAM DOESN'T HAVE A PROPER WAY TO CALCULATE FINES IF BOOKS ARE RETURNED LATE.
4. IF A BOOK HAS BEEN ISSUED THERE ISN'T A WAY TO REMOVE IT FROM THE TEXT FILE.
5. THUS MULTIPLE PEOPLE CAN BUY THE SAME BOOK.
6. DOESN'T CHECK IF THE BOOK IS AVAILABLE IN THE TEXT FILE.

Student Login

When a student logs in, the system performs the following steps:

1. **Login Authentication**:
    - The student is prompted to enter a password. The system verifies this password against a predefined value.
    - If the password is correct, the student is granted access. Otherwise, they are given multiple attempts to log in.
2. **Menu Options**:
    - Once authenticated, the student is presented with a menu of options:
        - Issue a book
        - Return a book
3. **Book Issuance**:
    - **Issuing Books**: Students can issue books based on their membership type. The system calculates the maximum issue days and applicable discounts, and prompts the student to enter their data (Name,Father Name, Student Id and Membership type) also book data (Book name, Book Author Name and ISB number) and saves and displays the data in the file and output screen.
    - **Calculating Discounts**: Based on the membership type, the system calculates the discount and final price for issuing a book.
4. **Book Return**:
    - **Returning Books**: Students can return books by entering the return date. The program takes data from the user like book name, author name,isb number then the student who is returning the book their data and then that data is stored in a file.
5. **Loop and Function Utilization**:
    - The system uses loops to iterate through the process of issuing and returning books, ensuring that all steps are followed correctly.
    - Functions are used to handle specific tasks, such as calculating discounts, determining maximum issue days, and managing overdue fines.

By separating the functionalities for staff and students, the system ensures that each user group has access to relevant features, enhancing the overall efficiency and user experience of the library management system.

4.4 STEP 4: Modules of the Project

The project consists of several modules, each responsible for a specific functionality:

- **Login Module**: Facilitates secure login for staff and students, ensuring access control.
- **Book Management Module**: Handles operations related to adding, updating, and retrieving book data.
- **Student Management Module**: Manages student records, including membership details and student IDs.
- **Membership Module**: Implements different membership types, calculates discounts, and manages loan periods.
- **File Handling Module**: Ensures data persistence by managing the reading and writing of book and student data to external files.

5. CONCLUSION & FUTURE WORK

5.1 Conclusion

In conclusion, while our library management system presents a significant step forward in automating and enhancing library operations, there are several areas for future improvement. Currently, the system faces certain limitations, such as the inability to verify if the student logging in is present in the file, the lack of unique login passwords for students, and the challenge of calculating fines for late book returns. Additionally, once a book has been issued, there is no mechanism to remove it from the text file, leading to potential issues with multiple people issuing the same book. The system also doesn't check if the book is available in the text file before issuing. However, these limitations highlight opportunities for future development. By addressing these issues, we aim to create a more robust and efficient library management system. Future enhancements will include implementing verification mechanisms for student logins, developing unique login passwords, creating a proper fine calculation method, and ensuring that issued books are correctly tracked and unavailable books are identified. This will ultimately lead to a more reliable and user-friendly system.

6. REFERENCES

The project was completely built on our own without using any outside source.

The idea was taken from these two posts:

https://www.linkedin.com/feed/update/urn:li:activity:7285818016603201537?updateEntityUrn=urn%3Ali%3Afs_updateV2%3A%28urn%3Ali%3Aactivity%3A7285818016603201537%2CFEED_DETAIL%2CEMPTY%2CDEFAULT%2Cfalse%29

https://www.linkedin.com/feed/update/urn:li:activity:7280651279545122817?updateEntityUrn=urn%3Ali%3Afs_updateV2%3A%28urn%3Ali%3Aactivity%3A7280651279545122817%2CFEED_DETAIL%2CEMPTY%2CDEFAULT%2Cfalse%29

Debugging was done through group effort.

7. Code

```
#include <iostream>

#include <fstream>

#include <string>

using namespace std;


int maxissuedays;

int price = 100;

int returndate;

int issuedate;

int discount;


struct Book {

    string bookname;

    string authorname;

    int isbnumber;

};


struct Student {

    string studentname;

    string fathername;

    string membershipType;
    int studentid;
};

Book books[100];
Student students[100];
int bookCount = 0;
int studentCount = 0;

void readBookData() {
    ifstream fin("books1.txt");
    if (fin.is_open()) {
        bookCount = 0;
        while (fin >> books[bookCount].isbnumber) {
            fin.ignore();
            getline(fin, books[bookCount].bookname, ',');
            getline(fin, books[bookCount].authorname);
            bookCount++;
        }
        fin.close();

        // Display books data
        for (int i = 0; i < bookCount; i++) {
            cout << "Book " << i + 1 << ": " << endl;
            cout << "ISBN: " << books[i].isbnumber << endl;
            cout << "Book Name: " << books[i].bookname << endl;
            cout << "Author Name: " << books[i].authorname << endl;
            cout << "--------------------------" << endl;
        }
    }
}

void writeBookData() {
    ofstream fout("books1.txt", ios::app); // Open file in append mode
    fout << books[bookCount-1].isbnumber << "," << books[bookCount-
1].bookname << "," << books[bookCount-1].authorname << endl;
    fout.close();
}

void readStudentData() {
    ifstream fin("students1.txt");
    if (fin.is_open()) {
        studentCount = 0;
        while (fin >> students[studentCount].studentid) {
            fin.ignore();
            getline(fin, students[studentCount].studentname, ',');
            getline(fin, students[studentCount].fathername, ',');
            getline(fin, students[studentCount].membershipType);
            studentCount++;
        }
        fin.close();
        // Display students data
        for (int i = 0; i < studentCount; i++) {
            cout << "Student " << i + 1 << ": " << endl;
            cout << "Student ID: " << students[i].studentid << endl;
            cout << "Student Name: " << students[i].studentname << endl;
            cout << "Father's Name: " << students[i].fathername << endl;
            cout << "Membership Type: " << students[i].membershipType << endl;
            cout << "--------------------------" << endl;
        }
    }
}

void writeStudentData() {
    ofstream fout("students1.txt", ios::app); // Open file in append mode
    fout << students[studentCount-1].studentid << "," << students[studentCount-
1].studentname << "," << students[studentCount-1].fathername << "," <<
students[studentCount-1].membershipType << endl;
    fout.close();
}

int calculateDiscount(string membershipType) {
    if (membershipType == "Executive") {
        return price * 0.10;
    } else if (membershipType == "Premium") {
        return price * 0.20;
    } else if (membershipType == "Gold") {
        return price * 0.30;
    } else {
        return 0;
    }
}

void Executive() {
    maxissuedays = 3;
    discount = price - calculateDiscount("Executive");
    cout << "The final price after discount: " << discount << endl;
}

void Premium() {
    maxissuedays = 6;
    discount = price - calculateDiscount("Premium");
    cout << "Final price after discount is: " << discount << endl;
}

void Gold() {
    maxissuedays = 9;
    discount = price - calculateDiscount("Gold");
    cout << "Final price after discount is: " << discount << endl;
}
void writeIssuedBookData(Student s, Book b, int finalPrice);
void writeReturnedBookData(Student s, Book b);
void issueBook() {
    Student s;
    Book b;

    cout << "Enter student name: ";
    getline(cin, s.studentname);
    cout << "Enter father's name: ";
    getline(cin, s.fathername);
    cout << "Enter student ID: ";
    cin >> s.studentid;
    cin.ignore();
    cout << "Enter membership type: ";
    getline(cin, s.membershipType);

    cout << "Enter book name: ";
    getline(cin, b.bookname);
    cout << "Enter author name: ";
    getline(cin, b.authorname);
    cout << "Enter ISBN number: ";
    cin >> b.isbnumber;
    cin.ignore();

    int discount = calculateDiscount(s.membershipType);
    int finalPrice = price - discount;
    cout << "Book '" << b.bookname << "' issued to " << s.studentname
         << " (ID: " << s.studentid << ") with membership " << s.membershipType
         << ". Final price after discount: " << finalPrice << endl;



    writeIssuedBookData(s, b, finalPrice);
}

void returnBook() {
    Student s;
    Book b;

    cout << "Enter student name: ";
    getline(cin, s.studentname);
    cout << "Enter father's name: ";
    getline(cin, s.fathername);
    cout << "Enter student ID: ";
    cin >> s.studentid;
    cin.ignore();
    cout << "Enter membership type: ";
    getline(cin, s.membershipType);

    cout << "Enter book name: ";
    getline(cin, b.bookname);
    cout << "Enter author name: ";
    getline(cin, b.authorname);
    cout << "Enter ISBN number: ";
    cin >> b.isbnumber;
    cin.ignore();

    cout << "Book '" << b.bookname << "' returned by " << s.studentname
        << " (ID: " << s.studentid << ") with membership " << s.membershipType
<< "." << endl;

    writeReturnedBookData(s, b);
}

void writeIssuedBookData(Student s, Book b, int finalPrice) {
    ofstream fout("issued_books.txt", ios::app);
    fout << s.studentid << "," << s.studentname << "," << s.fathername << ","
        << s.membershipType << "," << b.bookname << "," << b.authorname << ","
        << b.isbnumber << "," << finalPrice << endl;
    fout.close();
}

void writeReturnedBookData(Student s, Book b) {
    ofstream fout("returned_books.txt", ios::app);
    fout << s.studentid << "," << s.studentname << "," << s.fathername << ","
        << s.membershipType << "," << b.bookname << "," << b.authorname << ","
        << b.isbnumber << endl;
    fout.close();
```

```
}





int main() {
        cout<<"                              ---------------------------------------------"<<endl;
        cout<<"                                 Welcome to Our library Managment
Sytem"<<endl;
   cout<<"                              ---------------------------------------------"<<endl;
   cout<<"\n \n ";



   char login;
   cout << "Enter S IF YOU ARE LOGGING IN AS A STAFF ELSE PRESS A
FOR STUDENT: ";
   cin >> login;
   int passwordSta;
   int passwordStu;
   string membershipType;
   int choice;
   int attempts = 0;
   int soft;

   if (login == 'S' || login == 's') {
      while (attempts < 5) {
        cout << "Enter password: ";
        cin >> passwordSta;
        if (passwordSta == 1234) {
            cout << "How many times do you want to run the software? " << endl;
            cin >> soft;
            for (int i = 0; i < soft; i++) {
                cout << "1. Add books to the shelf\n2. Add student data\n3. View all
students\n4. View all books\nEnter your choice: ";
                cin >> choice;
                cin.ignore();
                if (choice == 1) {
                    cout << "Enter book name: ";
                    getline(cin, books[bookCount].bookname);
                    cout << "Enter author name: ";
                    getline(cin, books[bookCount].authorname);
                    cout << "Enter ISB number: ";
                    cin >> books[bookCount].isbnumber;
                    bookCount++;
                    writeBookData();
                } else if (choice == 2) {
                    cout << "Enter student name: ";
                    getline(cin, students[studentCount].studentname);
                    cout << "Enter father's name: ";
                    getline(cin, students[studentCount].fathername);
                    cout << "Enter student ID: ";
                    cin >> students[studentCount].studentid;
                    cin.ignore();
                    cout << "Enter membership type: ";
                    getline(cin, students[studentCount].membershipType);
                    studentCount++;
                    writeStudentData();
                } else if (choice == 3) {
                    readStudentData();
                } else if (choice == 4) {
                    readBookData();
                }
            }
            return 0;
        } else {
            cout << "Wrong Password. Try again." << endl;
            attempts++;
        }
      }
    } else  if (login == 'A' || login == 'a') {
      while (attempts < 5) {
        cout << "Enter password: ";
        cin >> passwordStu;
        if (passwordStu == 786) {
            cout << "How many times do you want to run the software? " << endl;
            cin >> soft;
            cin.ignore();
```

```
                for (int i = 0; i < soft; i++) {
                    cout << "1. Issue a book\n2. Return a book\nEnter your choice: ";
                    cin >> choice;
                    cin.ignore();
                    if (choice == 1) {
                        issueBook();
                    } else if (choice == 2) {
                        returnBook();
                    }
                }
                return 0;
            } else {
                cout << "Wrong Password. Try again." << endl;
                attempts++;
            }
        }
    }
    cout << "Too many attempts." << endl;

    return 0;
}
```