LIBRARY MANAGEMENT PROJECT

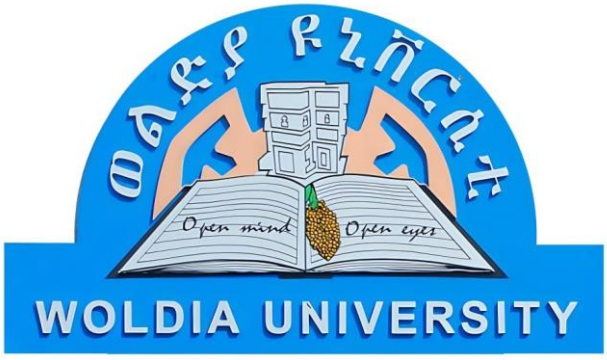
2025

Prepared by:

2ND YEAR IT STUDWENTS

Submitted to:

MR Nega A.



UNIVERSITY OF WOLDIYA INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

COMPUTER PROGRAMMING

GROUP 6

NAME	ID
1. Paulos Aschenaki	171293
2. Tadios Getaneh	171495
3. Sador Kebede	171322
4. Atsede Aschalew	170221
5. Eyerus Adino	170597

BEFORE ALL CHECK THE LINK HERE

Link: click here

Project 2: Library Management System

Create a library management system that allows users to manage books, including checking them in and out. This project should use structures for book attributes and a class to handle book management functionalities. Requirements 1. Define a structure for Book with attributes:

- int bookID
- std::string title
- std::string author
- bool isBorrowed

2. Create a class LibraryManager with methods to:

- Add a new book
- Check out a book (mark as borrowed)
- Return a book (mark as available)
- View all books

3. Implement a menu in the main function for user interactions.

The code:

```
#include <iostream>
#include <vector>
#include <string>
#include <iomanip>

using namespace std;

// Book structure
struct Book {
    int bookID;
    string title;
    string author;
    bool isBorrowed;
    string borrowerName;
};

// Library Manager class
class LibraryManager {
private:
    vector<Book> books;

public:
    void addBook() {
        Book b;
        cout << "\nADD NEW BOOK\n";
        cout << "----------------------------\n";
        cout << "Book ID: ";
        cin >> b.bookID;
        cin.ignore();
```

```
    cout << "Title: ";
    getline(cin, b.title);
    cout << "Author: ";
    getline(cin, b.author);

    b.isBorrowed = false;
    b.borrowerName = "None";
    books.push_back(b);

    cout << "Book successfully added.\n";
}

void borrowBook() {
    int id;
    cout << "\nBORROW BOOK\n";
    cout << "----------------------------\n";
    cout << "Enter Book ID: ";
    cin >> id;
    cin.ignore();

    for (int i = 0; i < books.size(); i++) {
        if (books[i].bookID == id) {
            if (!books[i].isBorrowed) {
                cout << "Borrower Name: ";
                getline(cin, books[i].borrowerName);
                books[i].isBorrowed = true;
                cout << "Book borrowed successfully.\n";
            } else {
                cout << "Book already borrowed by "
                     << books[i].borrowerName << ".\n";
            }
```

```
            return;
        }
    }
    cout << "Book not found.\n";
}

void returnBook() {
    int id;
    cout << "\nRETURN BOOK\n";
    cout << "----------------------------\n";
    cout << "Enter Book ID: ";
    cin >> id;

    for (int i = 0; i < books.size(); i++) {
        if (books[i].bookID == id) {
            if (books[i].isBorrowed) {
                books[i].isBorrowed = false;
                books[i].borrowerName = "None";
                cout << "Book returned successfully.\n";
            } else {
                cout << "Book was not borrowed.\n";
            }
            return;
        }
    }
    cout << "Book not found.\n";
}

void viewBooks() {
    cout << "\nLIBRARY BOOK LIST\n";
    cout << "-------------------------------------------------------------------------\n";
```

```
        cout << left << setw(8) << "ID"
             << setw(22) << "Title"
             << setw(18) << "Author"
             << setw(12) << "Status"
             << setw(15) << "Borrower" << endl;
        cout << "--------------------------------------------------------------------------\n";

        if (books.size() == 0) {
            cout << "No books available in the library.\n";
            return;
        }

        for (int i = 0; i < books.size(); i++) {
            cout << left << setw(8) << books[i].bookID
                 << setw(22) << books[i].title
                 << setw(18) << books[i].author
                 << setw(12) << (books[i].isBorrowed ? "Borrowed" : "Available")
                 << setw(15) << books[i].borrowerName << endl;
        }
    }
};

int main() {
    LibraryManager library;
    int choice;

    do {
        cout << "\n==========================================\n";
        cout << "      SMART LIBRARY MANAGEMENT SYSTEM\n";
        cout << "==========================================\n";
        cout << "1. Add New Book\n";
```

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

        switch (choice) {
            case 1: library.addBook(); break;
            case 2: library.borrowBook(); break;
            case 3: library.returnBook(); break;
            case 4: library.viewBooks(); break;
            case 5: cout << "\nThank you for using the library system.\n"; break;
            default: cout << "Invalid choice. Please try again.\n";
        }
    } while (choice != 5);

    return 0;
}
```

Explanation Of Functionality

1. Book Registration (Add New Book)

The system allows the user to add a new book record by entering the book ID, title, and author. When this option is selected from the menu, the system collects the required input and stores the book information in a vector. Each newly added book is marked as available and not borrowed. This function ensures that book data is properly recorded for library management.

2. Book Borrowing (Borrow Book)

The system enables a user to borrow a book by providing the book ID. If the book exists and is currently available, the system records the borrower's name and marks the book as borrowed. If the book has already been borrowed, the system displays the name of the current borrower. This function helps manage book lending efficiently.

3. Book Return (Return Book)

The system allows users to return a borrowed book by entering the book ID. If the book is found and is currently borrowed, the system updates its status to available and clears the borrower information. If the book was not borrowed, the system notifies the user. This function ensures accurate tracking of book availability.

4. Book Record Display (View All Books)

The system provides functionality to display all books stored in the library. When selected, it shows each book's ID, title, author, borrowing status, and borrower name in a well-formatted table. This function helps users easily view and verify library records.

5. System Termination (Exit Program)

The system provides an option to exit the program safely. When this option is selected, the program terminates execution properly. This functionality allows users to close the application after completing their tasks without errors.

Project Summary

The **Smart Library Management System** is a C++ console-based application developed as a **class project** at **Woldiya University, Institute of Technology, Department of Information Technology**. The system is designed to manage library books efficiently by allowing users to add books, borrow and return books, and view all book records. It stores essential book information such as book ID, title, author, and borrowing status using a simple menu-driven interface.

THANK YOU