Library Management System Report

Project Title: Library Management System

Developed by:

1. Yash Gatkal
2. Hrishikesh Gavai
3. Atharva Ghayal

Course: Object-Oriented Programming and Computer Graphics (OOPCG)

Abstract

Introduction: The **Library Management System** is designed to address the essential needs of a library by digitizing core operations, such as book and student management, using C++. This program models an effective library interface through the command line, enabling users to perform fundamental library operations in a simplified, user-friendly manner.

Motivation: A traditional library system is often burdened by manual processes, time-consuming record-keeping, and the risk of data mismanagement. Motivated to create a streamlined solution, this project aims to automate library tasks and improve data accuracy, offering librarians an efficient and accessible tool for managing library resources.

Outcome: This project demonstrates the implementation of Object-Oriented Programming (OOP) principles in a real-world context. By using C++ constructs such as classes, file handling, and modular design, we created a functional, user-centric library management system that is both effective and extendable for future enhancements. The outcome is a reliable, command-line-based application that simplifies library management and enhances user experience.

Objectives

1. To design a manageable and efficient library system using Object-Oriented Programming.
2. To allow users to perform core library management tasks while ensuring data persistence.
3. To provide an easy-to-navigate, text-based administrative interface.
4. To showcase modular programming, encapsulation, and file management in C++.

Hardware and Software Requirements

Hardware Requirements:

- **Processor**: Intel Core i3 or equivalent (minimum)
- **RAM**: 4 GB (minimum)
- **Storage**: 100 MB of free disk space
- **Display**: Standard monitor with at least 800x600 resolution

Software Requirements:

- **Operating System**: Windows, macOS, or Linux
- **Compiler**: Any C++ compiler (e.g., GCC, Clang, or Microsoft Visual C++)
- **Development Environment**: Recommended to use Visual Studio Code, Code::Blocks, or any other IDE with C++ support

---

System Overview

Features and Functionalities:

- **Book and Student Management**: Creation, modification, display, and deletion of records.
- **Book Issuing and Depositing**: Tracking transactions and calculating overdue fines.
- **Data Persistence**: Permanent storage of data with file handling.
- **Access Control**: Admin-level access to key management features.

Theoretical Concepts

This Library Management System demonstrates several foundational C++ concepts, including Object-Oriented Programming (OOP), file handling, modular design, and data validation, each outlined below.

---

1. Object-Oriented Programming (OOP)

Classes and Objects

- Two main classes:
    - **Book Class**: Contains information on book number, title, and author.

    - **`Student` Class**: Holds student details like admission number, name, issued book number, and token status.

Encapsulation and Data Hiding

- Class attributes (e.g., `bno` for books and `admno` for students) are managed through public member functions, encapsulating data to prevent unauthorized access.

Member Functions

- **Book Management**: Functions like `createBook`, `showBook`, and `modifyBook` handle book records.
- **Student Management**: Functions like `createStudent`, `showStudent`, and `modifyStudent` manage student details.
- **Accessor Functions**: Methods such as `retBNo` and `retAdmNo` allow controlled access to specific class data.

---

2. File Handling and Data Persistence

File Stream Operations

- **Persistent Storage**: Files `book.dat` and `student.dat` store book and student records in binary format using `fstream`.
- **Record Management**:
    - **Addition**: New records are appended to their respective files.
    - **Modification**: `seekp` is used for direct access and updating records.
    - **Deletion**: Unwanted records are removed by rewriting remaining records to a new file.

Data Retrieval and Display

- Functions `displayAllBooks` and `displayAllStudents` read and list all stored records, while individual display functions allow for specific record retrieval.

---

3. Modular Function Design

The system is divided into functional modules, which improves readability, ease of maintenance, and code reuse:

- **Creation Functions**:
    - `writeBook()` and `writeStudent()` add new book and student entries, respectively, saving data to files.
- **Display Functions**:

    - `displayAllBooks()` and `displayAllStudents()` display records; `displayBook()` and `displayStudent()` handle single-record queries.
- **Transaction Management**:
    - `bookIssue()` verifies a student's eligibility to issue a book and updates records accordingly.
    - `bookDeposit()` calculates overdue fees and resets the student's token status.

---

4. Control Structures and Data Validation

Menu Navigation

- **Switch-Case Statements**: A main menu loop uses switch-case for structured navigation, enabling easy access to program functions.

Data Validation

- Conditions check for:
    - Existing student records during issuing and depositing.
    - Validity of books for issuance.
    - Student eligibility to prevent multiple book issuance.

Program Structure

Main Function

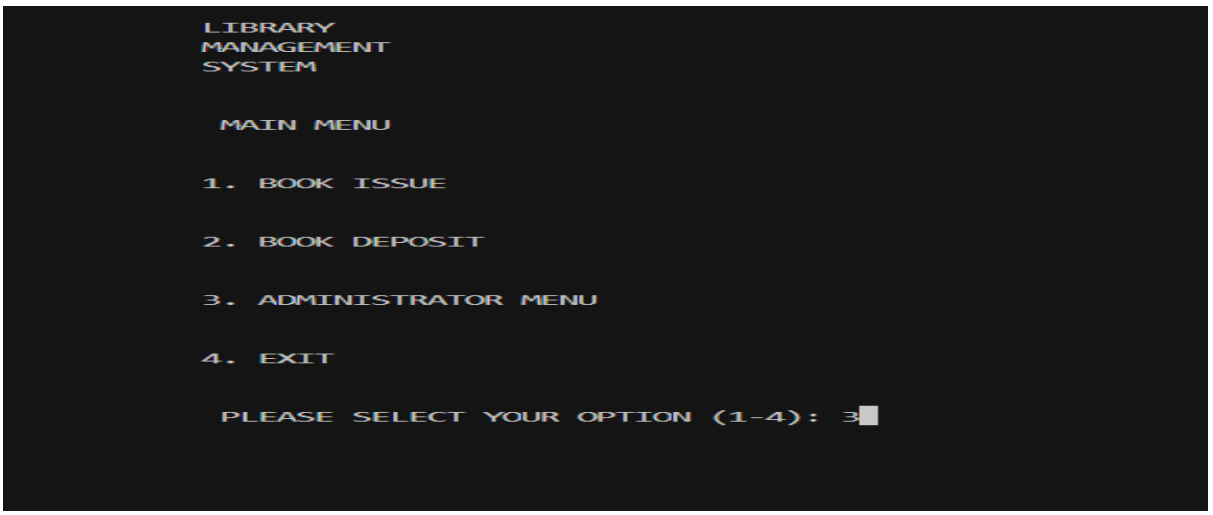
- The program's entry point is the `main` function, which launches a **welcome message** (`start()`) and opens the **main menu** loop. Depending on user input, control is passed to either the **student menu** or **admin menu** for further options.

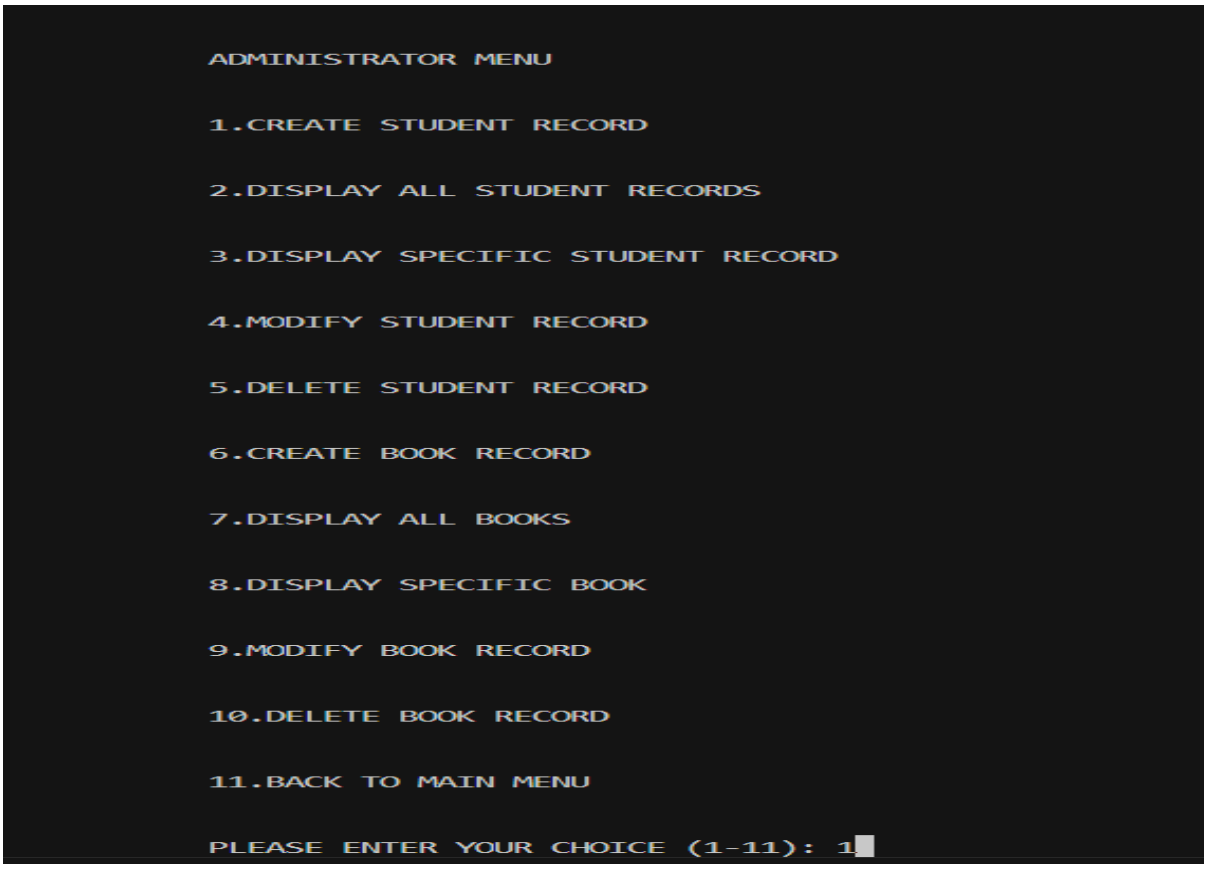
Admin Menu

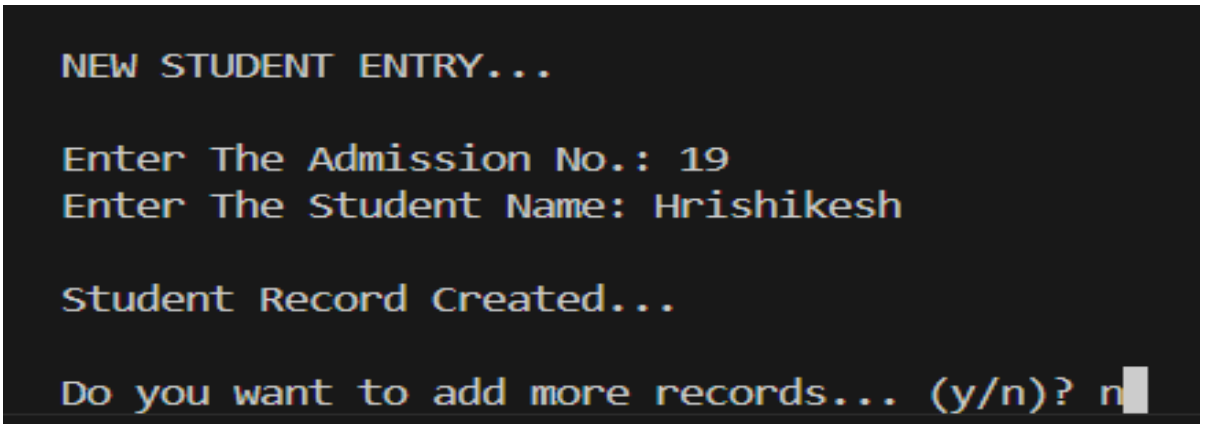
- The `adminMenu()` function centralizes library management activities like adding, modifying, deleting, and viewing records for books and students, thus supporting comprehensive library administration in a user-friendly format.

Output:

Below are some screenshots showcasing the functionality and output of our Library Management System.


LIBRARY
MANAGEMENT
SYSTEM
MAIN MENU
1. BOOK ISSUE
2. BOOK DEPOSIT
3. ADMINISTRATOR MENU
4. EXIT
PLEASE SELECT YOUR OPTION (1-4): 3

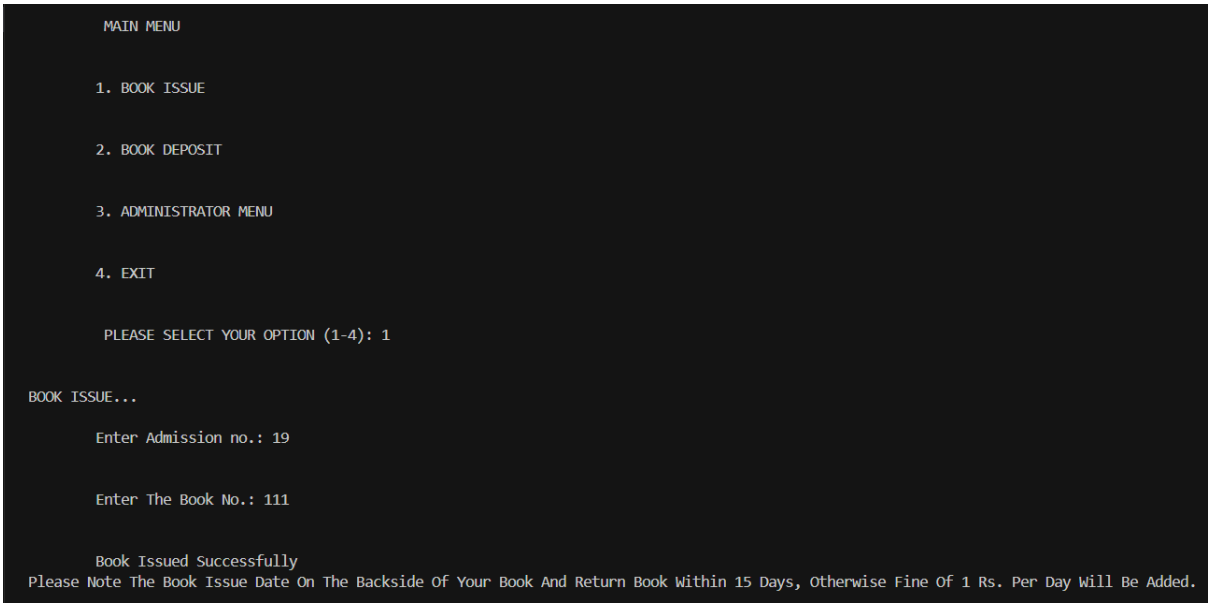

ADMINISTRATOR MENU
1.CREATE STUDENT RECORD
2.DISPLAY ALL STUDENT RECORDS
3.DISPLAY SPECIFIC STUDENT RECORD
4.MODIFY STUDENT RECORD
5.DELETE STUDENT RECORD
6.CREATE BOOK RECORD
7.DISPLAY ALL BOOKS
8.DISPLAY SPECIFIC BOOK
9.MODIFY BOOK RECORD
10.DELETE BOOK RECORD
11.BACK TO MAIN MENU
PLEASE ENTER YOUR CHOICE (1-11): 1


NEW STUDENT ENTRY...
Enter The Admission No.: 19
Enter The Student Name: Hrishikesh
Student Record Created...
Do you want to add more records... (y/n)? n

```
        PLEASE ENTER YOUR CHOICE (1-11): 6

NEW BOOK ENTRY...

ENTER BOOK NO.: 111

ENTER BOOK NAME: Digital Logic and Computer Design

ENTER AUTHOR NAME: M. Morris Mano


Book Created..

Do you want to add more records... (y/n)? n
```

```
        PLEASE ENTER YOUR CHOICE (1-11): 2


                Student List

========================================================================
        Admission No.       Name        Books Issued
========================================================================
        19          Hrishikesh          0
```

```
        PLEASE ENTER YOUR CHOICE (1-11): 7


                Book List

========================================================================
        Book No.            Book Name               Book Author
========================================================================
111Digital Logic and Computer Design                M. Morris Mano
```



Conclusion

The Library Management System project is an effective demonstration of OOP principles applied to a real-world problem, with data persistence achieved through file handling in C++. This project has enhanced our understanding of object-oriented design, modular programming, and practical implementation of C++ concepts.

The outcome is a basic but extendable library management system that can easily be enhanced with more complex functionalities or a graphical interface. This project provides a strong foundation for future exploration in programming and software development, reinforcing both theoretical knowledge and practical skills.

End of Report…