**SIMATS SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**CHENNAI-602105**

**Library Management System**

**A CAPSTONE PROJECT REPORT**

*Submitted in the partial fulfilment for the award of the degree of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

**K. Abitha Reddy (192210112)**

**Saliya Sameer patankar (192221145)**

**Under the Supervision of**

**Dr. yuvaraj**

**Jan-2025**

**DECLARATION**

We, **K.Abitha Reddy**, **Saliya Sameer Patankar**, students of **Bachelor of Engineering in Information Technology**, Department of Computer Science and Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, hereby declare that the work presented in this Capstone Project Work entitled **Library Management System** is the outcome of our own Bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics.

(K.Abitha Reddy-192210115)

(Saliya Sammer Patankar-192221145)

Date: 4-01-2025

Place: Chennai

**CERTIFICATE**

This is to certify that the project entitled **“Library Management System”** submitted by **K. Abitha Reddy , Saliya Sameer Patankar** has been carried out under my supervision. The project has been submitted as per the requirements in the current semester of B. Tech Informat ion Technology.

Teacher-in-charge

Dr. Yuvaraj

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| S.No | Title | Page. No. |
| 1 | ABSTRACT | 3 |
| 2 | INTRODUCTION | 4 |
| 3 | Objectives and Case Description | 5-7 |
| 4 | Pseudocode and Explanation | 7-9 |
| 5 | Results & Discussion,  Future Scope | 9-14 |
| 6 | Conclusion | 14 |
| 7 | REFERENCES | 14-15 |

**ABSTRACT**

In today's digital age, efficient management of library resources is paramount for educational institutions and public libraries. This project presents a comprehensive Library Management System developed using C++. The system aims to streamline library operations, enhance user experience, and facilitate effective resource management. This project not only addresses the challenges faced by traditional library management but also offers a scalable solution that can be adapted to the specific needs of different libraries. The implementation of this Library Management System demonstrates the potential of C++ in developing robust and efficient software solutions for real-world applications.

The Library Management System (LMS) developed using C++ is a software solution aimed at streamlining and automating the processes involved in managing a library's resources. The primary objective of this system is to provide a user-friendly platform that facilitates efficient inventory management, and user interaction. This system is designed to handle a variety of tasks, including book issuance and returns, catalog maintenance, user registration, and fines calculation, among others.

The LMS is built with an object-oriented approach, leveraging the principles of encapsulation, inheritance, and polymorphism to create a robust and scalable architecture. The LMS is designed with a user-friendly interface, ensuring ease of use for both library staff and members. It aims to reduce manual effort, minimize errors, and enhance the overall efficiency of library management. The project serves as an educational tool to demonstrate the practical application of C++ programming concepts in developing real-world solutions.

Library management systems are essential for the efficient organization and operation of libraries, ensuring that resources such as books, journals, and digital media are easily accessible to users. Implementing a library management system in C++ offers a robust and efficient way to handle the myriad tasks involved in library administration. C++'s object-oriented nature allows for the creation of classes to represent different entities within the library, such as books, patrons, and transactions. These classes can encapsulate relevant data and provide methods for performing operations like adding new books, searching for titles, issuing books to patrons, and tracking returns. Utilizing C++'s powerful standard libraries and data structures, such as vectors, lists, and maps, ensures that the system can efficiently handle large volumes of data and complex queries, providing a smooth and responsive user experience

**INTRODUCTION**

The Library Management System (LMS) is a vital instrument for the modernization and optimization of library operations, meeting the changing requirements of both users and librarians. In this case study, we explore the design and execution of a C++ Library Management System with the goal of resolving issues with manual systems and utilizing technology to improve productivity and user experience.

The Library Management System encompasses several core functionalities, including book inventory management, user registration, book lending and return processes, and overdue fine calculations. The system is designed with a user-friendly interface that allows librarians and patrons to interact with the system effortlessly.

Key features of the Library Management System include:

* **Book Inventory Management**: Efficient tracking and cataloging of books, including information such as title, author, genre, and availability status.
* **User Management**: Registration and management of library members, including maintaining records of borrowed books and transaction history.
* **Book Lending and Returns**: Streamlined processes for issuing and returning books, with automated updates to the inventory.
* **Fine Calculation**: Automated calculation of overdue fines to ensure timely returns and adherence to library policies.
* **Search Functionality**: Advanced search options for books based on various criteria, improving the ease of locating resources.

The system employs object-oriented programming principles, ensuring modularity, scalability, and maintainability. Data persistence is achieved through the use of file handling mechanisms in C++, enabling reliable storage and retrieval of library records.

**OBJECTIVES**

**1.Automate Library Operations:**

* Develop a system to automate the core operations of a library, including book, member registration, book lending, and return processes**.**

**2. Enhance Resource Management:**

* Implement efficient tracking and management of library resources to ensure accurate inventory records and availability status of books.

**3. Improve User Experience:**

* Design a user-friendly interface that facilitates easy interaction for both librarians and patrons, making library operations smoother and more accessible.

**4.Ensure Data Integrity and Security:**

* Utilize file handling mechanisms in C++ to ensure reliable data storage, retrieval, and security of library records.

**5.Streamline Book Lending and Return Processes:**

* Simplify and automate the processes of issuing and returning books, reducing manual effort and minimizing errors.

**6.Automate Fine Calculation:**

* Implement automated fine calculation for overdue books to enforce timely returns and maintain library policies.

**7. Provide Advanced Search Functionality:**

* Develop advanced search features to allow users to easily locate books based on various criteria such as title, author, genre, and availability.

**8. Implement Object-Oriented Design Principles:**

* Apply object-oriented programming principles to ensure the system is modular, scalable, and maintainable.

**9. Adapt to Different Library Needs:**

* Create a scalable solution that can be customized to meet the specific requirements of different types and sizes of libraries.

**10.Demonstrate the Potential of C++:**

* Showcase the capabilities of C++ in developing robust and efficient software solutions for real-world applications.
* **User Management:** The system supports different user roles, such as administrators, librarians, and members, each with distinct access levels and permissions. This hierarchical access control ensures data security and efficient workflow management.
* **Catalog Management:** The system provides comprehensive functionalities for managing book records, including adding, updating, and deleting entries. Each book record includes details like title, author, ISBN, publication date, and genre, facilitating easy search and retrieval.
* **Transaction Handling:** The LMS automates the process of issuing and returning books, updating inventory in real-time. It also manages due dates and fine calculation for late returns, thus ensuring the smooth operation of library services.
* **Search and Retrieval:** Users can search for books using various criteria such as title, author, genre, or ISBN. The system's efficient search algorithms enable quick access to desired books, enhancing user experience.
* **Reporting and Analytics:** The system includes reporting features that provide insights into various aspects of library operations, such as book circulation statistics, user activity logs, and inventory status. These reports aid in decision-making and strategic planning.

**Case Description:**

The scenario of creating a C++ library management system (LMS) takes place in a dynamic setting where libraries must simultaneously accommodate the varied needs of administrators and patrons while embracing technological improvements. Effective resource management presents particular issues for libraries of all kinds, including academic institutions and public libraries. Administrators, patrons, and librarians are all involved in this scenario, and each has certain tasks and obligations. Patrons rely on the library for resources and information, managers make strategic decisions, and librarians oversee daily operations.

Traditional manual systems have a number of drawbacks, such as inefficient cataloging, restricted accessibility, and challenging transaction tracking. The shift to digital media makes these problems worse. The goal of implementing a C++ LMS is to solve these problems by improving user experience, automating jobs, and optimizing operations. The system's iterative development process has improved workflow efficiency, decreased administrative burdens, and improved patron access to resources. Important results include better resource use, more productivity, and happier administrators and customers. All things considered, the LMS guarantees a more effective and user-focused experience by modernizing library operations.

**Methods:**

• We have created separate logins for students and the librarian, in which the librarian is password protected.

• In this project, the librarian can add, update, delete and create books and can also assign the books to the students.

• The students can also view the list of the books available in the entire library database.

• The entire rights are given to the librarian to adding books, issuing books, and modify the book.

• This project uses file handling to store the data of books in a project.

• Reissuing and returning the books are the main features of this project.

**•** The student can also be able to see which student has already borrowed the book.

**Modules of Library Management System:**

* Add Book.
* Modify Book.
* Delete Book.
* Search Book.
* Issue Book.
* Return Book.

**Pseudocode:**

**#include <iostream>**

**#include <vector>**

**#include <string>**

**#include <iomanip> // For formatting**

**#include <ctime> // For date handling**

**using namespace std;**

**// Structure for Member**

**struct Member {**

**int id;**

**string name;**

**string type; // Staff or Student**

**};**

**// Structure for Book**

**struct Book {**

**int id;**

**string title;**

**string author;**

**bool isBorrowed;**

**int borrowDate; // Day of the year when the book was borrowed**

**};**

**// Global vectors to store members and books**

**vector<Member> members;**

**vector<Book> books;**

**// Function prototypes**

**void staffLogin();**

**void studentLogin();**

**void addMember();**

**void addBook();**

**void viewBooks();**

**void borrowBook();**

**void returnBook();**

**void mainMenu();**

**// Utility function to get the current date**

**int getCurrentDate() {**

**time\_t t = time(0); // Get current time**

**tm \*now = localtime(&t); // Convert to local time structure**

**return now->tm\_yday; // Return day of the year (1-365)**

**}**

**// Constants**

**const string STAFF\_PASSWORD = "1234";**

**const string STUDENT\_PASSWORD = "5678";**

**const int FINE\_PER\_DAY = 5; // Fine amount per overdue day**

**int main() {**

**mainMenu();**

**return 0;**

**}**

**void mainMenu() {**

**int choice;**

**while (true) {**

**cout << "\n--- Library Management System ---\n";**

**cout << "1. Staff Login\n";**

**cout << "2. Student Login\n";**

**cout << "3. Exit\n";**

**cout << "Enter your choice: ";**

**cin >> choice;**

**switch (choice) {**

**case 1:**

**staffLogin();**

**break;**

**case 2:**

**studentLogin();**

**break;**

**case 3:**

**cout << "Exiting the system. Goodbye!\n";**

**return;**

**default:**

**cout << "Invalid choice. Please try again.\n";**

**}**

**}**

**}**

**void staffLogin() {**

**string password;**

**cout << "Enter Staff Password: ";**

**cin >> password;**

**if (password == STAFF\_PASSWORD) {**

**int choice;**

**while (true) {**

**cout << "\n--- Staff Menu ---\n";**

**cout << "1. Add Member\n";**

**cout << "2. Add Book\n";**

**cout << "3. View Books\n";**

**cout << "4. Logout\n";**

**cout << "Enter your choice: ";**

**cin >> choice;**

**switch (choice) {**

**case 1:**

**addMember();**

**break;**

**case 2:**

**addBook();**

**break;**

**case 3:**

**viewBooks();**

**break;**

**case 4:**

**cout << "Logging out...\n";**

**return;**

**default:**

**cout << "Invalid choice. Please try again.\n";**

**}**

**}**

**} else {**

**cout << "Incorrect password. Access denied.\n";**

**}**

**}**

**void studentLogin() {**

**string password;**

**cout << "Enter Student Password: ";**

**cin >> password;**

**if (password == STUDENT\_PASSWORD) {**

**int choice;**

**while (true) {**

**cout << "\n--- Student Menu ---\n";**

**cout << "1. View Books\n";**

**cout << "2. Borrow Book\n";**

**cout << "3. Return Book\n";**

**cout << "4. Logout\n";**

**cout << "Enter your choice: ";**

**cin >> choice;**

**switch (choice) {**

**case 1:**

**viewBooks();**

**break;**

**case 2:**

**borrowBook();**

**break;**

**case 3:**

**returnBook();**

**break;**

**case 4:**

**cout << "Logging out...\n";**

**return;**

**default:**

**cout << "Invalid choice. Please try again.\n";**

**}**

**}**

**} else {**

**cout << "Incorrect password. Access denied.\n";**

**}**

**}**

**void addMember() {**

**Member newMember;**

**cout << "Enter Member ID: ";**

**cin >> newMember.id;**

**cout << "Enter Member Name: ";**

**cin.ignore();**

**getline(cin, newMember.name);**

**cout << "Enter Member Type (Staff/Student): ";**

**getline(cin, newMember.type);**

**members.push\_back(newMember);**

**cout << "Member added successfully!\n";**

**}**

**void addBook() {**

**Book newBook;**

**cout << "Enter Book ID: ";**

**cin >> newBook.id;**

**cout << "Enter Book Title: ";**

**cin.ignore();**

**getline(cin, newBook.title);**

**cout << "Enter Book Author: ";**

**getline(cin, newBook.author);**

**newBook.isBorrowed = false;**

**newBook.borrowDate = -1; // Default value for not borrowed**

**books.push\_back(newBook);**

**cout << "Book added successfully!\n";**

**}**

**void viewBooks() {**

**cout << "\n--- Book Catalog ---\n";**

**cout << left << setw(10) << "Book ID" << setw(30) << "Title" << setw(20) << "Author" << "Status\n";**

**cout << string(60, '-') << "\n";**

**for (const auto &book : books) {**

**cout << left << setw(10) << book.id**

**<< setw(30) << book.title**

**<< setw(20) << book.author**

**<< (book.isBorrowed ? "Borrowed" : "Available") << "\n";**

**}**

**}**

**void borrowBook() {**

**int bookId;**

**cout << "Enter Book ID to borrow: ";**

**cin >> bookId;**

**for (auto &book : books) {**

**if (book.id == bookId) {**

**if (book.isBorrowed) {**

**cout << "Book is already borrowed.\n";**

**} else {**

**book.isBorrowed = true;**

**book.borrowDate = getCurrentDate(); // Record the borrow date**

**cout << "Book borrowed successfully!\n";**

**}**

**return;**

**}**

**}**

**cout << "Book ID not found.\n";**

**}**

**void returnBook() {**

**int bookId;**

**cout << "Enter Book ID to return: ";**

**cin >> bookId;**

**for (auto &book : books) {**

**if (book.id == bookId) {**

**if (!book.isBorrowed) {**

**cout << "Book is not borrowed.\n";**

**} else {**

**// Calculate fine**

**int currentDay = getCurrentDate();**

**int overdueDays = currentDay - book.borrowDate - 14; // Assuming 14 days borrowing period**

**if (overdueDays > 0) {**

**int fine = overdueDays \* FINE\_PER\_DAY;**

**cout << "Late submission! You have a fine of " << fine << " units.\n";**

**} else {**

**cout << "Book returned on time.\n";**

**}**

**book.isBorrowed = false; // Mark as returned**

**book.borrowDate = -1; // Reset borrow date**

**cout << "Book returned successfully!\n";**

**}**

**return;**

**}**

**}**

**cout << "Book ID not found.\n";**

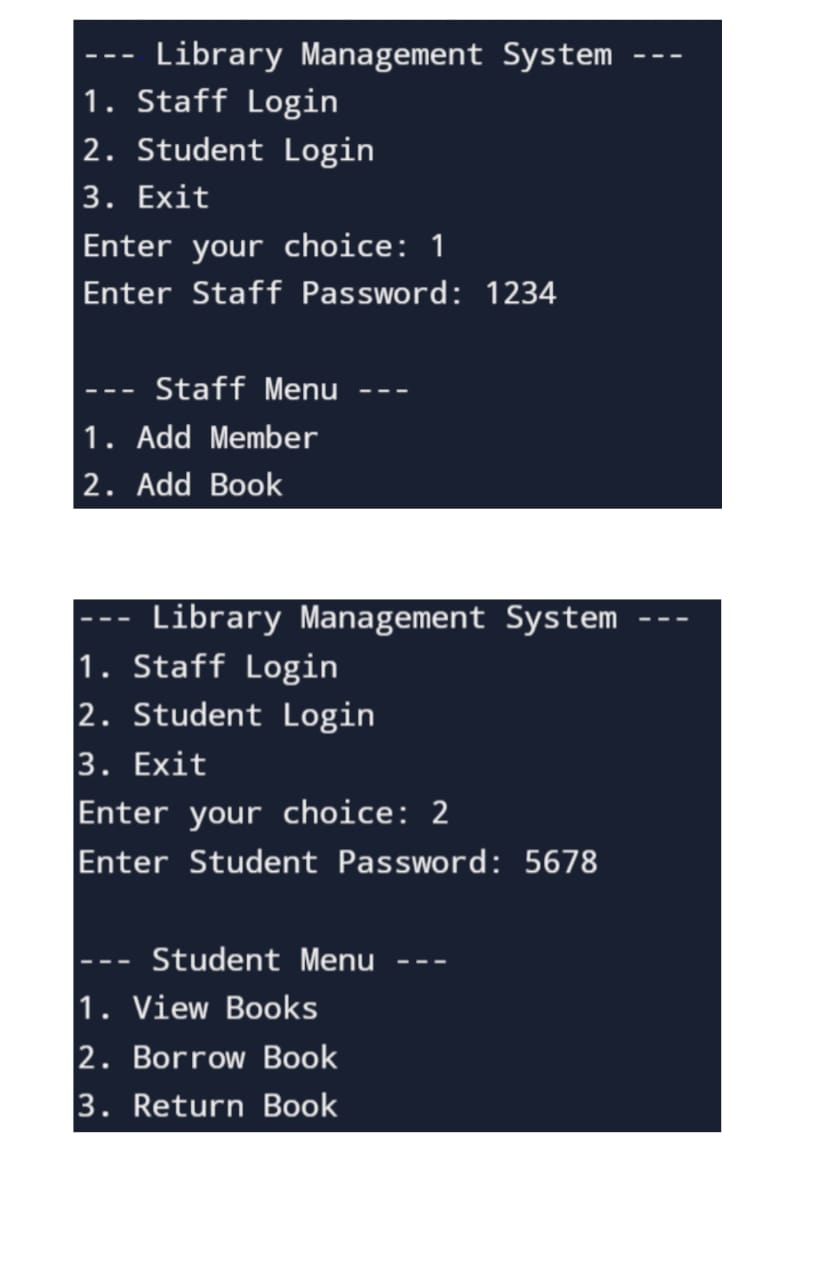
**}**

**Explanation:**

* The library management system (LMS) was developed in C++ through a methodical use of object-oriented programming techniques and software engineering concepts. A key factor in encouraging modular development and improving the structure, maintainability, and extensibility of the system was object-oriented design.
* In order to manage and store library-related data effectively, file handling techniques were used during data collection, which involved user interactions via a console-based interface.
* In data analysis, user input was processed and various functionalities were carried out using control structures and error handling mechanisms.
* Although limitations including dependency on console-based interfaces and scalability constraints were acknowledged, the chosen methods—including object-oriented programming and file handling—were justified for their usefulness in accomplishing project objectives.
* In general, these methods of inquiry assisted in the creation of a working C++ library management system that satisfied project requirements while taking its goals and limitations into account.

**Result:**

When you run the project from any compiler or directly clicking on the executable .exe file you’ll see the following screen shown in the picture



• **View Booklist:**

Same as students view booklist, librarians will also able to see the books available in the library database.

• **Search For a Book:**

The Librarian can search book either by book name or by book id. Both the options are available in the project.

• **Modify/Add Book:**

In this menu option Librarian can do three main operations i.e. Adding a Book, deleting a Book and Modifying the existing Book.

As we are using the file handling methods in this project, Every time new file is generated to store the details of the books. i.e. **Booksdata.txt**.

• **Issue Book:**

Due to this option 70% of the work is been reduced. In this option Librarian can do the following operations:

* Issue a Book.
* View Issued Books.
* He can also search the students who issued the books.
* Librarian can also reissue the book to the same student.
* Return the Book.

As the name suggest the Library Management System is a software which handles the entire data of library. It makes the work of librarian very easy instead of writing data in a notebook. In past the librarians were using notebooks to write the data of books along with student’s name who borrowed that book. So, it was very difficult to keep track on each and every book.

If librarian wants to search for a particular book, then that task was very time consuming. So, to make this task easy the programming languages were developed and C++ language is one of them.

To store the student details the separate file name **Student.txt** is been created.

In this way, we’ve created and executed the program, this is how are program runs and gives output

**Discussion:**

The viability of utilizing object-oriented programming for effective library administration is demonstrated by the C++ implementation of the library management system (LMS). Through achieving goals such as user authentication and book management, the system improves the efficiency of library operations. Because it depends on file handling, it can be easily deployed in a variety of computing settings. Although C++ has performance benefits, there can be distinct trade-offs with other technologies. Web-based solutions, for instance, provide accessibility but increase server setup complexity. The LMS has drawbacks, such as a graphical user interface deficiency and scalability problems with huge datasets. In order to improve user experiences, future study might examine sophisticated data management strategies and GUI development.

In conclusion, the C++-based LMS effectively accomplishes the main goals, although there is still opportunity for development, indicating areas that warrant additional study in the field of library automation and management software.

**Future Scope:**

The future scope of the Library Management System using C++ includes several enhancements and expansions to further improve its functionality and adaptability. Integration with online databases and digital libraries could provide access to a broader range of resources, including e-books and academic journals. Implementing a mobile application interface would offer greater accessibility and convenience for users on the go. Advanced features like predictive analytics for book recommendations, automated notifications for overdue books, and integration with RFID technology for efficient tracking and security could further streamline library operations. Additionally, incorporating multilingual support and customization options would make the system more versatile and inclusive for diverse user bases. Continuous updates and the inclusion of emerging technologies will ensure that the Library Management System remains a cutting-edge solution for modern library management**.**

**Conclusion:**

The Library Management System developed using C++ effectively addresses the challenges faced by traditional library management. By automating core library operations such as book user registration, lending, and returns, the system significantly enhances resource management and user experience. The implementation of object-oriented design principles ensures the system is modular, scalable, and maintainable, providing a robust solution adaptable to various library needs. Moreover, advanced search functionalities and automated fine calculations streamline processes and enforce policies efficiently. This project exemplifies the potential of C++ in creating efficient and reliable software solutions for real-world applications, offering a comprehensive and scalable approach to modern library management.

**References:**

[1]. HonghaiKan, Zhimin Yang, Yue Wang, Nana Qi, “Research on Library

Management System for CDs Attached to Books Based on Cloud Computing”, in Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design 2010.

[2]. Bao Sun, JiangweiFeng and Ling Liu, “A Study on How to Construct the Prediction Model of Library Lending of University Library”, International Conference on Information Science and Technology March 26-28,2011 Nanjing, Jiangsu, China.

[3]. Erxiang Chen,Minghui Liu,“Research and Design on Library Management System Based on Struts and Hibernate Framework”, WASE International Conference on Information Engineering2009.

[4]. Jianzhong, YunqingFeng, Yun Zhao, “A Unified Modelling Language-Based

Design and Application for a Library Management Information System”, incybernetics and information technologies.

[5]. Michael Hitchens, Andrew Firmage, “The Design of a Flexible Class Library

Management System”, in IEEE conference 1998.

[6]. WeihongYang, “Design and Implementation of Library Management System”, International Conference on Management Science and Innovative Education (MSIE 2015).

[7]. Bretthauer, D. “Open-source software in libraries. Library Hi Tech News, 18 (5), 8-9(2001).

[8]. Barve, S., &Dahibhate, N. B., “Open-source software for library services”, DESIDOC Journal of Library & Information Technology, 32(5) (2012). Journal of Xi'an University of Architecture & Technology Volume XII, Issue XI, 2020ISSN No: 1006-7930Page No: 752

[8]. Ching-Yu Huang and Patricia A., “MorrealeA Web-based, Self-Controlled Mechanism to Support Students Learning SQL” IEEE Integrated STEM Education Conference (ISEC)2011

**Top of Form**