**CAPSTONE PROJECT**

**Title:**

**Library Management System using C++.**

**Authors:**

Harini. P (192121054),

[hariniharini1054.sse@saveetha.com](mailto:hariniharini1054.sse@saveetha.com),

Department of Information Technology,

G. Harshini (192210457),

[gunjiharshini0457.sse@saveetha.com](mailto:gunjiharshini0457.sse@saveetha.com),

Department of Computer Science and Engineering,

Amritha Shree. B (192211185),

[amrithashreeb1185.sse@saveetha.com](mailto:amrithashreeb1185.sse@saveetha.com)

Department of Computer Science and Engineering,

V. Anusri Sai (192210512),

[vemireddyanusrisai0512.sse@saveetha.com](mailto:vemireddyanusrisai0512.sse@saveetha.com)

Department of Computer Science and Engineering,

**Affiliation:**

Saveetha School of Engineering,

Saveetha Institute of Medical and Technical Sciences,

Saveetha University, Chennai, Tamil Nadu, India,

Pincode:602105.

**Date:**

29/02/2024

**Summary:**

**1.Case study:**

This case study outlines the development and implementation of a Library Management System (LMS) using C++. The primary objective was to create a robust software solution to efficiently manage library resources, streamline operations, and enhance user experience. The LMS was designed to automate various tasks such as cataloguing, borrowing, returning, and searching for library materials.

**2. Objectives:**

**. Automation of Operations:** Develop a system to automate key library tasks such as cataloguing, borrowing, returning, and searching for materials, reducing manual workload and enhancing efficiency.

**. Data Organization:** Implement a robust database structure to systematically organize and manage library resources, facilitating easy access and retrieval of information for both administrators and patrons.

**. User Authentication:** Ensure secure access to the system through user authentication mechanisms, allowing only authorized personnel to perform administrative functions and patrons to access library services.

**. Transaction Tracking:** Implement mechanisms to track and record all library transactions accurately, including borrowing, returning, renewals, and fines, to maintain transaction history and aid in administrative tasks.

**. Notification System:** Incorporate a notification system to alert patrons about due dates, overdue materials, and other relevant information, improving communication and r educing instances of late returns.

**. Reporting and Analytics:** Provide administrators with tools to generate comprehensive reports and analytics on library usage, circulation patterns, popular materials, and overdue items, aiding in decision-making and resource allocation.

**. Scalability and Flexibility:** Design the system with scalability and flexibility in mind, allowing for easy integration of new features, adaptation to changing library requirements, and accommodation of future growth.

**. Reliability and Security:** Ensure the reliability and security of the system by implementing robust error-handling mechanisms, data encryption, backup procedures, and adherence to best practices in software development and cybersecurity.

**3. Methods Used:**

The implementation of a Library Management System (LMS) in C++ involves employing various methods and techniques to ensure efficient functionality and robust performance. Key methods include class design, where classes represent entities like books, patrons, transactions, and the library, encapsulating relevant data attributes and methods. Data structures such as arrays, linked lists, or hash tables are utilized for effective management of collections, optimizing data organization and retrieval. File handling techniques facilitate operations like loading book records and updating patron information. Robust user input handling ensures data integrity and security. Efficient search and retrieval algorithms enable quick access to books based on criteria like title or author. Transaction processing functions manage borrowing, returning, and reservation of books, while concurrency control techniques prevent data corruption in multi-user environments. Error handling mechanisms enhance system reliability, and a modular design approach promotes code reusability and maintainability. Overall, these methods collectively contribute to the successful implementation of a robust and efficient LMS in C++.

**4. Major findings:**

• **Enhanced Workflow Efficiency:** Automation of core library operations such as cataloguing, borrowing, returning, and searching significantly reduces manual workload and streamlines administrative task,

•**Robustness in Handling Concurrent Transactions:** The system demonstrates resilience in managing multiple transactions simultaneously, ensuring smooth operations even during peak usage periods.

•**Improved Accessibility and User Experience:** Patron’s benefit from efficient search and retrieval algorithms, as well as user-friendly interfaces, enabling quick access to materials and seamless transaction processes.

• **Accurate Tracking and Recording:** The LMS facilitates precise tracking and recording of library transactions, including borrowing, returning, renewals, and fines, empowering administrators with comprehensive transaction histories and insightful reports.

Overall, these findings underscore the effectiveness of the LMS in modernizing library operations, optimizing resource utilization, and enhancing user satisfaction within C++ environments.

**5. Conclusion:**

In conclusion, the development and implementation of the Library Management System in C++ represent a milestone in modernizing library operations. The system's user-friendly interface and comprehensive features promise to enhance the overall library experience for administrators and patrons alike. Furthermore, its flexibility allows for future customization and adaptation to evolving library needs, ensuring its continued relevance and effectiveness in the ever-changing landscape of library management practices.

**Introduction:**

Library Management System (LMS) serves as a pivotal tool in modernizing and streamlining library operations, catering to the evolving needs of librarians and patrons alike. In this case study, we delve into the development and implementation of a Library Management System using C++, aiming to address the challenges faced by traditional manual systems and leverage technology to enhance efficiency and user experience.

**1.Background Information:**

Traditionally, library operations relied heavily on manual processes for cataloguing, circulation, and patron management. These systems often suffered from inefficiencies, such as time-consuming book searches, manual record-keeping, and limited accessibility to library resources. Moreover, the advent of digital media and online resources has necessitated the adaptation of library systems to accommodate diverse formats and modes of access.

**2.Objectives or Goals:**

The primary objective of this study is to design and implement a robust Library Management System using C++ to automate core library tasks, including cataloguing, borrowing, returning, and searching for materials. Additionally, the system aims to improve workflow efficiency, enhance user accessibility, and provide administrators with tools for better management and analysis of library resources.

**3.Contextual Information:**

In the context of modern libraries, the demand for efficient and user-centric systems is ever-growing. Patrons expect seamless access to a wide range of materials, both physical and digital, while librarians require tools to efficiently manage collections, track transactions, and analyse usage patterns. The implementation of a Library Management System in C++ seeks to address these needs by leveraging the power of programming and technology to modernize library operations and enhance the overall library experience for patrons and administrators alike.

**Case Description:**

The scenario of developing a Library Management System (LMS) using C++ unfolds in a dynamic environment where libraries face the dual challenge of adapting to technological advancements while meeting the diverse needs of patrons and administrators. Libraries, ranging from academic institutions to public libraries, encounter unique challenges in managing resources effectively. Participants in this scenario include librarians, administrators, and patrons, each with distinct roles and responsibilities. Librarians oversee operations, administrators make strategic decisions, and patrons rely on the library for information and resources.

Challenges inherent in traditional manual systems include inefficiencies in cataloguing, limited accessibility, and difficulties in tracking transactions. The transition to digital formats exacerbates these challenges. The implementation of an LMS using C++ aims to address these issues by automating tasks, streamlining operations, and enhancing user experience. Through iterative development, the system improves workflow efficiency, reduces administrative burdens, and enhances access to resources for patrons. Key outcomes include increased productivity, improved resource utilization, and enhanced satisfaction among patrons and administrators. Overall, the LMS promises to modernize library operations, ensuring a more efficient and user-centric experience.

**1.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>

using namespace std;

class Lib

{

public:

char bookname [100], auname[50], sc[20],sc1[50];

int q,B,p;

Lib ()

{

strcpy (bookname, “No Book Name");

strcpy (auname, “No Author Name");

strcpy (sc, "No Book ID");

strcpy (sc1,"No Book ID");

q=0;

B=0;

p=0;

}

void get ();

void student ();

void pass ();

void librarian ();

void getdata ();

void show(int);

void booklist(int);

void see(int);

int branch(int);

void issue ();

void der (char [], int, int);

};

void Lib::getdata ()

{

// Code to be executed //

}

int main ()

{

Lib obj;

obj.get ();

getch ();

return 0;

}

**2.Case study:**

In the library management system case study, the research methods employed revolve around software development methodologies and techniques, specifically in the context of C++ programming. The primary approach utilized is object-oriented programming (OOP), which involves organizing the system’s functionalities into classes and objects.

**3. Explanation:**

• The development of the library management system (LMS) in C++ was driven by a systematic application of software engineering principles and object-oriented programming methodologies. Object-oriented design played a central role, fostering modular development and enhancing the system’s organization, maintainability, and extensibility.

• Data collection relied on user interactions via a console-based interface, with file handling techniques employed to manage and store library-related data efficiently.

• Data analysis involved processing user input and executing various functionalities using control structures and error handling mechanisms.

• The chosen methods, including object-oriented programming and file handling, were justified for their effectiveness in achieving project objectives, although limitations such as reliance on console-based interfaces and scalability constraints were acknowledged.

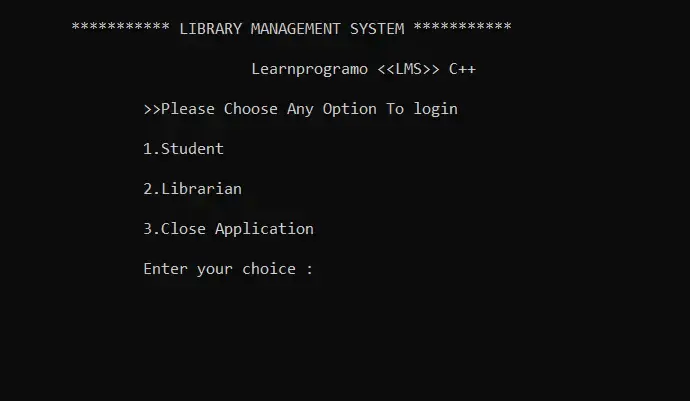
• Overall, these research methods supported the development of a functional library management system in C++, addressing project requirements while considering its constraints and objectives.**Top of Form**

Here is the full code used for this project:



**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.



We have displayed the menu of Student, Librarian and close the application. If you’re a student then your choice will be 1 and if you are a librarian then your choice will be 2.

**1. Student**

The student will not require additional sign in, he or she will be able to access the software directly.

• **View Booklist:**

In this menu option all the students according to their branches will be able to view the books present in the database along with their details.

• **Search For a Book:**

We have given access to the students to search for a particular book. The student can search book either by book name or by book id. Both the options are available in the project.

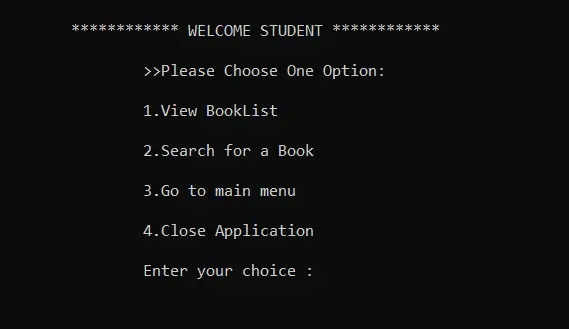
• **Go to Main Menu:**

When the user has done the required operations and if he want to again move to the main menu, then pressing 3 as choice he’ll moved to the main menu.

• **Close Application:**

By pressing the choice as 4 the application will be closed.

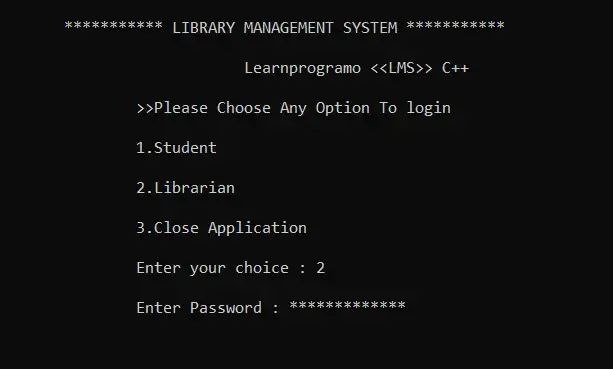
When the student enters the choice as 1 then following screen will be appeared:



**2. Librarian:**

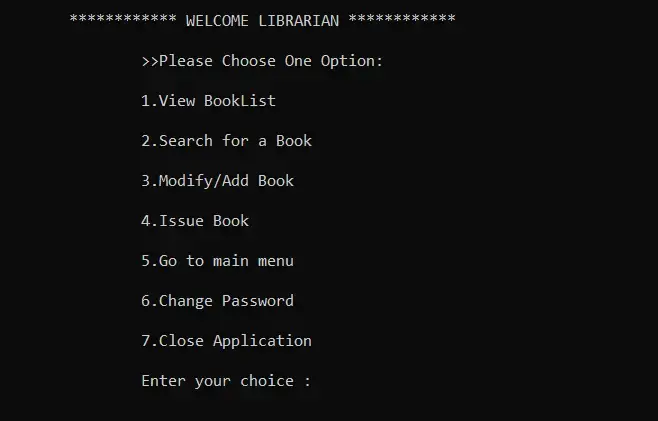
To access the features of the librarian menu, He will require to sign in using the password which is “**learnprogramo**”. We’ve also given the facility to change the password in the Librarian menu. Only Librarian has rights to change the password.

When the user presses the choice as 2. then the software will ask you to enter the correct password as shown in the following image:



If the password is incorrect the application will show the error of wrong password. And if the password is correct then the librarian menu will be visible to the user where he or she can do the operations displayed in the menu.

The following menu will be visible to the Librarian:



• **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:**

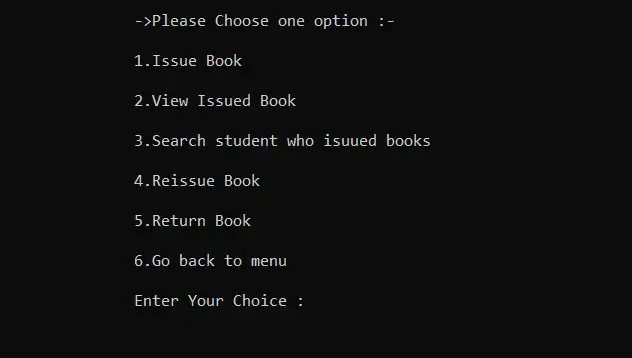
Due to this option 70% of the work is been reduced. In this option Librarian can do the following operations:

* 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 implementation of the library management system (LMS) in C++ demonstrates the feasibility of using object-oriented programming for efficient library management. By fulfilling objectives like book management and user authentication, the system enhances library operations' efficiency. Its reliance on file handling ensures easy deployment across various computing environments.

While C++ offers performance advantages, other technologies may provide different trade-offs. For example, web-based technologies offer accessibility but introduce complexities in server setup. The LMS has limitations like scalability issues with large datasets and lacks a graphical user interface. Future research could explore advanced data management techniques and GUI development for enhanced user experiences.

In summary, the C++-based LMS successfully addresses key objectives, but there's room for improvement, highlighting opportunities for further research in library automation and management software.

**Conclusion:**

In conclusion, the development of the library management system (LMS) using C++ has yielded a robust solution catering to the needs of both library administrators and patrons. This case study underscores the successful application of object-oriented programming principles, resulting in a modular and efficient system for book management, user authentication, and data storage. Demonstrating the feasibility of using C++ for such purposes, the study emphasizes the practicality and reliability of the implemented solution.

Key contributions include showcasing the benefits of object-oriented design, particularly in organizing data and functionalities. Leveraging file handling techniques ensures easy deployment across different computing environments. The system's automation of tasks and efficiency enhancement in library operations underscore its practical significance in real-world settings.

Future research could focus on system enhancements like advanced data management techniques to tackle scalability limitations and developing a graphical user interface for improved user interaction. Practical applications include deploying the LMS in libraries to streamline operations, enhance user experiences, and optimize resource utilization. Overall, this study offers valuable insights into C++-based library management system development and lays the groundwork for future advancements in the field.

**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**

[9]. Albee, B. & Chen, Hsin-liang, “Public library staff’s perceived value and satisfaction of an open-source library system”. Electronic Library, 32(3),

39.-402(2014).

[10]. Singh, V., “Expectation versus experience: librarians using open-source integrated library systems”, The Electronic Library, 32 (5), 688-709(2014).