Project Report: Library Management System

Introduction:

The **Library Management System (LMS)** is designed to automate the library's operations. It helps in managing the collection of books, keeping track of issued books, adding new books, modifying details of existing books, and maintaining the records of borrowed books. This system is implemented in **C++**, utilizing **File Handling** for data storage and retrieval. The project supports functionalities for both **students** and **librarians**.

The LMS allows students to view available books, search for them, issue books, return books, and check fines. Librarians can add new books, modify or delete existing books, issue books to students, and track borrowed books.

Project Objectives:

The primary objectives of the Library Management System project are:

- To create an easy-to-use software system for managing library books.
- To provide functionalities for students to search for and issue books.
- To allow librarians to add, modify, or delete books in the system.
- To maintain records of all issued books, including due dates and fines.
- To automate the process of managing student and book data, improving overall efficiency.

Features:

The Library Management System comes with the following features:

For Students:

1. **View Available Books**: Students can view the list of books available in the library.
2. **Search Books**: Students can search books either by name or by Book ID.
3. **Issue Books**: Students can request to issue books by providing their details.
4. **Return Books**: Students can return books, and if overdue, pay fines.
5. **Reissue Books**: If a student requires an extension on the due date, they can request a reissue.

For Librarians:

1. **Add New Books**: Librarians can add new books to the library system.
2. **Modify Existing Books**: Librarians can update details such as book name, author, price, quantity, etc.
3. **Delete Books**: Librarians can delete books that are no longer needed.
4. **Issue Books to Students**: Librarians can issue books to students by verifying their information.
5. **Track Issued Books**: Librarians can view the list of books issued, including details like student name, book name, and issue date.

Additional Functionalities:

- **Fines**: If books are returned late, the system calculates the fine based on the days overdue.
- **Branch Selection**: The system supports multiple branches for students to issue books from specific categories (e.g., Class 12th, CS, EC, etc.).

Tools and Technologies Used:

- **Programming Language**: C++
- **Libraries**:
  - <iostream>: For input and output operations.
  - <fstream>: For file handling operations (reading/writing data to files).
  - <cstring>: For string manipulation.
  - <conio.h>: For user input and control functions (like getch()).
- **Development Environment**: Visual Studio Code (VS Code)

System Design:

The system follows a simple structure where different modules interact with each other:

1. **Main Module**: The main module acts as the interface between the student/librarian and the system. It invokes different functionalities such as issuing books, viewing available books, etc.
2. **Book Handling Module**: Responsible for maintaining book-related data such as name, author, ID, price, quantity, etc.
3. **Student Module**: Manages student-related information such as name, ID, and issued books.

4. **Fine Calculation**: A module that calculates the fine based on overdue days when a student returns a book late.

The system is designed with a menu-driven interface. Users can select the desired option and the corresponding function is executed.

Implementation:

Class Structure:

The **Lib** class is the core of the Library Management System. It has several functions to perform the operations of book management, student management, and issuing books:

- **Data Members**:
  - bookname[]: Stores the name of the book.
  - auname[]: Stores the author's name.
  - sc[]: Stores the book's ID.
  - sc1[]: Stores the publication name.
  - q: Quantity of the book in the library.
  - B, p: Book price and quantity available.
- **Member Functions**:
  - getdata(): Accepts book details from the librarian.
  - show(): Displays the details of the books.
  - modify(): Allows the librarian to modify or delete book details.
  - issue(): Allows librarians to issue books to students.
  - see(): Enables search functionality by book name or ID.
  - fine(): Calculates fines based on overdue days.
  - der(): Decreases or increases the quantity of a book after issuance or return.

The system interacts with files like Booksdata.txt and student.txt to store and retrieve book and student records.

Main Functions Flow:

- The program starts by asking the user (student or librarian) to log in.
- The system then displays the corresponding menu based on user type.

- Librarians can perform CRUD operations (Create, Read, Update, Delete) on book records.
- Students can search for books, issue them, and return books.

File Handling:

The system uses file handling to manage books and student records:

- **Booksdata.txt**: Stores details of books available in the library. Each book is represented by a record containing its name, author, ID, price, and quantity.
- **Student.txt**: Stores the details of books issued to students, along with their personal information and the date of issue.

File Operations:

- **Reading from Files**: The system reads book and student data to perform various functions like searching and displaying information.
- **Writing to Files**: New records (books and students) are written to their respective files when actions like issuing books or adding new books are performed.

Challenges and Solutions:

Challenges:

1. **Handling Multiple Users**: The system needs to handle requests from both students and librarians simultaneously.
    - **Solution**: The system provides different menus and functionalities based on user roles.
2. **File Management**: The system uses file handling, which can be prone to errors if files are not managed correctly.
    - **Solution**: Proper error handling is implemented for file operations (e.g., checking if a file exists before performing read/write operations).
3. **Late Fine Calculation**: Implementing a robust system to calculate fines based on the difference between the return date and due date.
    - **Solution**: A function fine() was implemented that calculates the total fine based on the number of days overdue.

Conclusion:

The Library Management System project provides an efficient and user-friendly way of managing library operations. It automates tedious manual tasks such as issuing books, maintaining records, and calculating fines. The system is modular and can be easily extended or modified to accommodate more features. Through this project, I gained valuable experience in file handling, object-oriented programming, and system design.

Future Scope:

- **Online Integration**: Integrating the system with an online portal so that users can access it remotely.
- **Database Management**: Moving from file handling to using an actual database (e.g., MySQL or SQLite) for more efficient data management.
- **User Authentication**: Implementing password-based authentication for both librarians and students to improve security.