LIBRARY MANAGEMENT SYSTEM USING QBM

1st Dr. Gauri Ghule

gauri.ghule@viit.ac.in

2nd Sai Trisha N

saitrisha.22311719@viit.ac.in

3rd Isha Gandewar

isha.22311728@viit.ac.in

4th Shrinidhi Gundewar

shrinidhi.22311609@viit.ac.in

5th Kunal Desale

kunal.22311612@viit.ac.in

6th Ronit Tambekar

ronit.22311694@viit.ac.in

(Vishwakarma Institute of Information Technology, E&TC dept, Pune)

I. ABSTRACT:

This paper presents a library management systemusing queue (QBM) that efficiently handles book transactions using linked lists and queues. The system maintains a queue of available books and tracks student information for issued books. It uses a queue mechanism to manage students awaiting book availability, enhancing user experience by ensuring no requests are overlooked. Through a user-friendly interface, students can issue and return books while monitoring both current loans and waiting lists. This approach not only simplifies library operations but also introduces an effective method for managing resource allocation in educational settings. The implementation demonstrates the effectiveness of combining data structures to create an effective solution for modern library management challenges.

II. INTRODUCTION:

With the improvement of people's level of knowledge, the library has become an indispensable part of daily life. But the library storage and the business volume are huge, the traditional accounts' management is merely not feasible [2]. Series of difficulties have been occurring in the physical library because of this inefficient use in library management. Mostly human error happens in keeping records, like some manually written files loss and damaged due to inefficiency use [3].

In today's digital age, effective library management is crucial for ensuring that educational institutions can efficiently serve their users. The evolution of library management systems (LMS) has been driven by the need to streamline operations, improve user experience, and adapt to the rapid changes in technology. Traditional manual processes often lead to inefficiencies and errors, hindering the accessibility and availability of resources.

This paper presents a comprehensive library management system that integrates essential functionalities such as book issuing, returning, and management of waiting lists. By using data structures like linked lists and queues, the system optimizes resource allocation and enhances user engagement. The proposed library management system not only allows for real-time tracking of book availability but also facilitates a easy process for students to access and return materials. Through the implementation of this system, libraries can significantly improve operational efficiency, reduce administrative overhead, and provide a user-friendly experience for both staff and patrons.

III. METHODOLOGY:

1. ROLE OF LINKED LISTS

1.1 Dynamic Data Structure: Linked lists allow for dynamic memory allocation, meaning books and student information can be added or removed without needing a fixed size. This flexibility is essential in a library context, where the number of books and students can vary significantly [1].

1.2 Book Management:

1.2.1 The Book class uses a linked list to maintain a list of available books in the library. Each Book object points to the next book, creating a chain that allows for easy traversal and management of the collection.

1.2.2 This structure simplifies operations such as adding new books or displaying the list of available titles.

1.3 Student Information Management:

1.3.1 The StudentInfo class also uses a linked list to keep track of students who have borrowed books. This allows for efficient insertion of new student records as they borrow books and provides a straightforward way to access and display information about currently issued books.

2. ROLE OF QUEUES

- **2.1 First-Come, First-Served Model:**

The waiting list for books that are currently issued to other students is managed using a queue. This data structure follows a first-in-first-out (FIFO) principle, ensuring that students who request a book are served in the order they arrive [3].

2.2 Handling Waiting Students:

2.2.1 When a student requests a book that is unavailable, their request is added to the waitingQueue. When the book is returned, the system checks the queue and notifies the next student in line, ensuring efficient management of book requests.

2.3 Efficient Resource Management:

2.3.1 Using a queue allows the library system to manage requests without the need for complex indexing or searching, which can enhance performance and reduce overhead.

The figure below shows the flow chart of the Library Management System

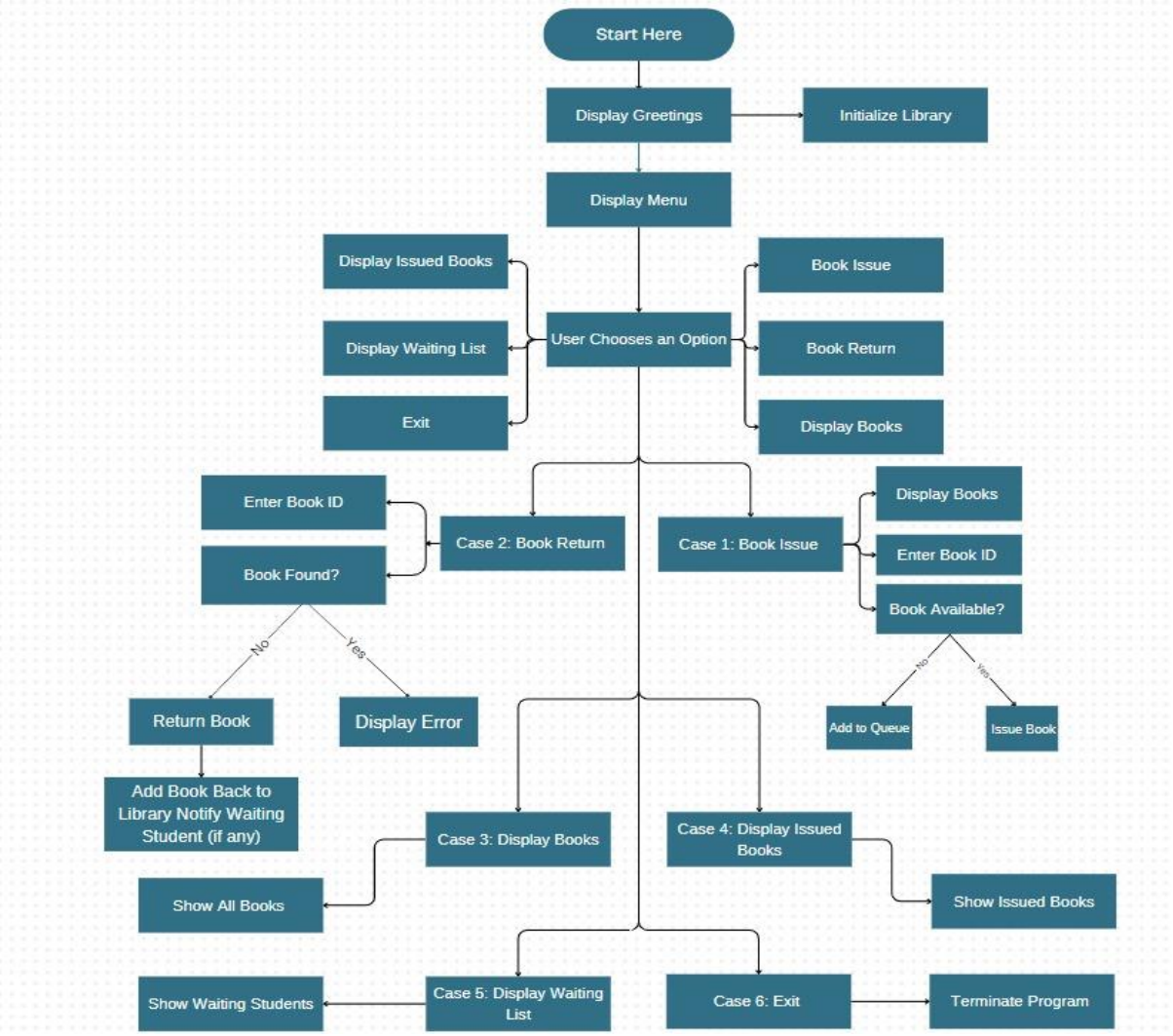

Fig: Flow Diagram Of Library Management System

3. CLASSES USED:

3.1 **Book Class**

Attributes:

1) name: Stores the book's title.
2) author: Stores the book's author.
3) id: Stores the book's ID.
4) next: Points to the next book in the linked list.

Constructor:

1) Initializes a new book with its name, author, and ID, and sets next to nullptr.

3.2 **StudentInfo Class**

Attributes:

1) name: Stores the student's name.
2) email: Stores the student's email.
3) book: Stores the name of the issued book.
4) author: Stores the author of the issued book.
5) id: Stores the ID of the issued book.
6) next: Points to the next student in the linked list.

Constructor:

1) Initializes a new student record with their name, email, and book details, and sets next to nullptr.

3.3 **Library Class**

Attributes:

1) start_lib: Pointer to the start of the linked list of books.
2) start_student: Pointer to the start of the linked list of students who have issued books.
3) waitingQueue: A queue to store students waiting for a specific book.

Methods:

1) initialize_lib(): Initializes the library with some predefined books.
2) add_book(): Adds a new book to the library by appending it to the end of the linked list.
3) display_books(): Displays all available books in the library.
4) book_issue(): Issues a book to a student. If the book exists, it's issued and removed from the library, with the student added to the student list. If the book doesn't exist, the student is added to the waiting queue.
5) book_return(): Returns a book to the library, re-adds it to the available list, and notifies waiting students, if any.
6) display_students(): Displays all students who have issued books along with the book details.
7) display_waiting_list(): Displays the list of students waiting for specific books.
8) delete_book(): Removes a book from the library when issued.

4. EXECUTION FLOW:

4.1 **Main Function**
The program starts by creating an instance of the Library class, which initializes the library with predefined books.
The greetings() function displays a welcome message.
The menu() function provides options for the user to issue or return books, display available or issued books, view the waiting list, and exit. Based on the user's selection, the respective action is performed.

5. MENU FUNCTIONALITY:

5.1 **Book Issue**
The user is prompted to enter a book ID. If the book exists, it is issued, and the book is removed from the library. If the book does not exist, the student is added to the waiting queue for that book.

5.2 **Book Return**
The user enters the book ID to return. If the student had issued the book, it is returned to the library. If another student is waiting for that specific book, they are notified and served next.

5.3 **Display Books**
Displays all available books in the library.

5.4 **Display Issued Books**
Shows the list of issued books along with student details.

5.5 **Display Waiting List**
Displays students in the queue waiting for a specific book.

5.6 **Exit**
The program terminates when the user selects this option.

6. DATA STRUCTURES USED:

6.1 **Linked List**
Books and student information are stored as linked lists, where each node points to the next.

6.2 **Queue**
The waiting list is implemented using a queue to serve students in a first-come, first-served manner.

7. KEY ALGORITHMS:

7.1 **Adding Books and Students**
Books and students are appended to the end of their respective linked lists.

7.2 **Deleting Books**
Books are removed from the library's linked list when issued to a student.

7.3 **Queue Operations**
Students waiting for books are added to the queue, and when a book is returned, the next student in the queue is notified.

VI. CODE OUTPUT:

Figure 1 illustrates the welcome page of the student library system. The menu has four choices Book issue, Book return, Displaying Books , Displaying issued books , Displaying Waiting list and exit.

When choice 1 is entered by the user , a list of available books is shown followed by questions asking about the book ID , student name and student email.

```
* WELCOME TO STUDENT LIBRARY *



MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice: 1

Available Books:
Book Title: Let Us C, Author: Yashavant Kanetkar, ID: 101
Book Title: Object-Oriented Programming with C++, Author: E. Balagurusamy, ID: 102
Book Title: Java: The Complete Reference, Author: Herbert Schildt, ID: 103
Book Title: A Byte of Python, Author: Swaroop C H, ID: 104
Enter the Book ID to issue: 101
Enter Student Name: Isha
Enter Student Email: isha@gmail.com
Book 'Let Us C' issued successfully!
```

FIGURE 1: OUTPUT DISPLAYING BOOK TITLES AND MENU WITH CHOICE 1.

When choice 2 is entered by the user the program asks for the ID of the book to be returned. If the ID is present, a success message is displayed as shown in figure 2.

```
MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice:1

2
Enter the Book ID to return: 101
Book 'Let Us C' returned successfully!
```

FIGURE 2: OUTPUT FOR CHOICE 2.

When there is no user in the waiting list, a blank waiting list is displayed as shown in figure 3 .

```
Book Title: Java: The Complete Reference, Author: Herbert Schildt,
    ID: 103

MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice: 5

Waiting List:

MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
```

FIGURE 3: OUTPUT DISPLAYING WAITING LIST WITH NO PRIOR QUEUE.

If a book is issued by the user then the choice 3 displays all the books present in the linked list excluding the books issued by the user as seen in figure 4 and 5.

```
Book Title: Java: The Complete Reference, Author: Herbert Schildt,
    ID: 103
Enter the Book ID to issue: 101
Enter Student Name: sai
Enter Student Email: sai@gmail.com
Book 'Let Us C' issued successfully!

MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice: 3

Available Books:
Book Title: Object-Oriented Programming with C++, Author: E.
    Balagurusamy, ID: 102
```

FIGURE 4: OUTPUT FOR ISSUING BOOK OF ID 101, AND DISPLAYING AVAILABLE BOOKS.

When a book is issued by the one user and a second user needs the same book , then that particular user is added to the queue. When the option 5 is entered , the waiting list is displayed which displays the book ID present in the queue as seen in figure 5.

```
Book Title: Java: The Complete Reference, Author: Herbert Schildt,
    ID: 103
Enter the Book ID to issue: 101
Invalid Book ID! Adding to waiting list...

MENU:
1. Book Issue
2. Book Return
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice: 5

Waiting List:
Book ID: 101

MENU:
1. Book Issue
2. Book Return
```

FIGURE 5: OUTPUT OF MENU WITH CHOICE 5 GIVING A WAITING LIST OF ID:101.

On entering choice 6 the user can quit the application.

```
3. Display Books
4. Display Issued Books
5. Display Waiting List
6. Exit
Enter your choice:
=== Session Ended. Please Run the code again ===
```

FIGURE 6: OUTPUT ON ENTERING CHOICE 6.

VII. CONCLUSION AND FUTURE SCOPE:

The implemented library management system effectively showcases how linked lists and queues can enhance the organization and management of books and student records. The use of linked lists allows for dynamic and flexible handling of book and student data, enabling efficient additions and deletions. Meanwhile, the queue structure ensures a fair and orderly process for managing waitlists for books, improving user experience. Overall, the system facilitates seamless interactions between library staff and students, providing a foundation for effective library operations.

To further improve and expand the system's functionality, integrating a **database** such as MySQL or SQLite can enhance data storage and retrieval efficiency, especially for large-scale libraries. Database integration would also allow for better query capabilities, supporting features such as advanced book searches or detailed student borrowing histories.

Incorporating **user authentication features** would provide an additional layer of security, ensuring that only authorized personnel and registered students can access and modify sensitive information. This

could also allow for role-based access control, where different users (e.g., administrators, librarians, students) have varying levels of system privileges.

An **enhanced user interface** would improve the overall user experience. A graphical interface with interactive elements could replace the text-based command-line structure, making it more accessible to a broader audience. This could be further extended to mobile applications for greater flexibility in accessing the system.

Introducing **notification and alert mechanisms** could automate reminders for due dates, overdue fines, or book availability for users on the waitlist. These notifications could be sent via email or integrated into a mobile app, ensuring that students and staff are always kept up-to-date.

Moreover, additional features like **data analytics** could be integrated to track popular books, borrowing patterns, and student engagement, helping the library make data-driven decisions about future acquisitions and resource management. **Book recommendations** based on borrowing history could further enhance user satisfaction.

VIII. REFERENCES:

1. P. Shanmugam, A. Ramalakshmi, G. Sasthri, and S. Baalachandran, "Library Management System," Journal of Xi'an University of Architecture & Technology.
2. S. Perumal, S. Jan, A. Sharma, R. Kumawat, and S. Sharma, "Library Management System," International Journal of Advance Research and Innovative Ideas in Education, vol. 9, no. 3, pp. 1683-20354, 2023.
3. T. W. Araya and A. Mengsteab, "Designing Web-based Library Management System," International Journal of Engineering Research & Technology (IJERT), vol. 9, no. 10, pp. 1-6, Oct. 2020. [Online].
4. N. S. Naga Prasanna and B. N. Srinivasa Gupta, "Online Library Management System," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 7, no. 5, pp. 117-120, May 2020. [Online]. Available: