

ONLINE LIBRARY MANAGEMENT SYSTEM

ANKIT SINGH

CSE 3A II Year

2200290100025

Class RNO: 25



PROBLEM STATEMENT •



Inefficient Book Tracking: Current manual library systems lack a streamlined process for tracking books, leading to issues in locating and managing resources.

Limited Accessibility: Physical access restrictions to the library hinder users from accessing resources remotely, impacting the overall efficiency of the system.

Time-Consuming Transactions: Manual check-in and check-out processes contribute to delays and inefficiencies in library transactions.

Ineffective Record-Keeping: Traditional record-keeping methods are prone to errors, leading to inaccuracies in user histories, overdue notices, and inventory management.

Lack of Analytics: The absence of analytical tools hampers the library's ability to derive insights from user behavior, book popularity, and overall system performance.



- 1/18/2024

PURPOSE

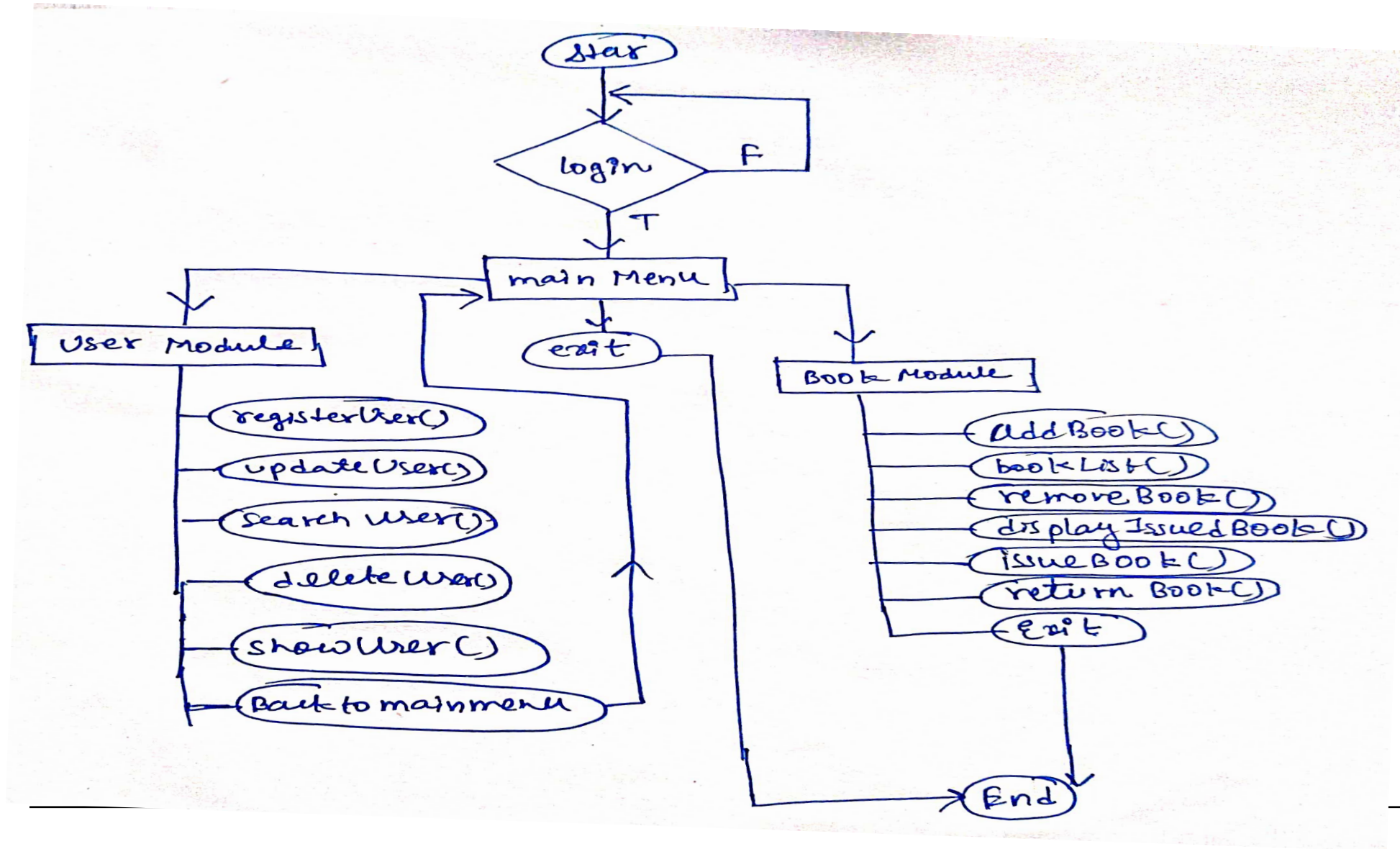
.

- 1. It provides “better and efficient” service to members.*
- 2. Reduce the workload of librarian.*
- 3. Faster retrieval of information about the desired book.*
- 4. Provide facility for proper monitoring reduce paper work and provide data security.*
- 5. All details will be available on a click for both user and librarian.*

SCOPE OF PROJECT

- Objective: Develop a user-friendly library management system to streamline book cataloging, circulation, and inventory control.
Features: Implement a robust database for efficient book tracking, user management, and transaction recording.
Accessibility: Ensure cross-platform accessibility, allowing users to manage library resources seamlessly.
Reports: Generate comprehensive reports on book availability, user history, and overdue items for effective decision-making.
Scalability: Design the system to accommodate future expansion and integration of additional features, ensuring long-term viability and adaptability.

FLOWCHART OF THE ONLINE LIBRARY MANAGEMENT SYSTEM



STRUCTURE USED

1)

```
struct User {  
  
    char username[50];  
    char password[50];  
};
```

2)

```
struct Book {  
    int book_id;  
    char book_name[100];  
    char book_auth[100];  
    char book_pub[100];  
    int quantity;  
    char date[12];  
};
```

3)

```
struct Student {  
    int student_id;  
    char student_name[100];  
    char branch[50];  
    int semester;  
    char section;  
    char email[50];  
    char phone[15];  
    int book_id;  
    char book_name[100];  
    char book_auth[100];  
    char book_pub[100];  
    char date[12];  
    char dueDate[12];  
};
```



FILE USED

Users.txt:

Purpose: This file is used to store information about user accounts.

Format: Each line in this file represents a user account. The format for each line is "username password"

Books.txt:

Purpose: This file is used to store information about books in the library.

Format: Each line in this file represents a book. The format for each line is "book_id book_name book_auth book_pub quantity date".

issuedBooks.txt:

Purpose: This file is used to store information about books that have been issued to students.

Format: Each line in this file represents an issued book. The format for each line is "student_id student_name branch semester section phone email book_id book_name book_auth book_pub date dueDate".

FUNCTION USED

```
each: function(e, t, n) {
  var r, i = 0,
      o = e.length,
      a = N(e);
  if (n) {
    if (a) {
      for (; o > i; i++)
        if (r = t.apply(e[i], n), r === !1) break
    } else
      for (i in e)
        if (r = t.apply(e[i], n), r === !1) break
  } else if (a) {
    for (; o > i; i++)
      if (r = t.call(e[i], i, e[i]), r === !1) break
  } else
    for (i in e)
      if (r = t.call(e[i], i, e[i]), r === !1) break;
  return e
},
trim: b && !b.call("\uffff\u00a0") ? function(e) {
  return null == e ? "" : b.call(e)
} : function(e) {
  return null == e ? "" : (e + "").replace(C, "")
},
makeArray: function(e, t) {
  var n = t || [];
  return null != e && (M(Object(e)) ? x.merge(n, "string" == typeof e ? [e] : e) : h.call(n, e)), n
},
isArray: function(e, t, n) {
  var r;
  if (t) {
    if (n) return a.call(t, e, n);
    for (r = t.length, n = n ? 0 > n ? Math.max(0, r + n) : 0 : 0; r > n; n++)
      if (n in t && t[n] === e) return n
  }
}
```

In user module:

```
void userModel();
void registerUser();
void updateUser();
void searchUser();
void deleteUser();
void showUserList();
```

In book module:

```
void bookModule();
void addBook();
void bookList();
void removeBook();
void updateBook();
void issueBook();
void displayIssuedBooks()9
void returnBook();
```

PROGRAM FLOW



The main() function invokes the login process.



After successful login, users are presented with a main menu.



Users navigate between the User Module and Book Module.



Subsequently, users can perform various operations such as adding books, issuing books, updating user details, etc.



The program keeps running until the user chooses to exit

SUMMARY OF PROJECT

- The Library Management System (LibMS) is a text-based console program designed to manage user accounts, book information, and book transactions within a library. The program utilizes text files to store and retrieve data, including user credentials, book details, and information about issued books. Here is a summary of the key features and functionalities of the Library Management System:

User Management:

Users can register new accounts with unique usernames and passwords.

User information is stored in the "users.txt" file.

Existing user accounts can be updated, searched, and deleted.

Book Management:

Books can be added to the library system with details such as book ID, name, author, publisher, quantity, and date.

Book information is stored in the "books.txt" file.

The library's collection can be listed, updated, and removed.

Book Transactions:

Books can be issued to students, with relevant details recorded in the "issuedBooks.txt" file.

Students are identified by unique IDs and provide information such as name, branch, semester, contact details, and the book they want to borrow.


Asante Gracias

धन्यवाद



अन्यस्योक्तिः

Köszö

Suksema Dankie 

Ευχαριστώ Danke Ahéhee

Thank Dik Tak Dank

Jai-rruh-jef
kasih

kasih

Grazie

Merci

Спасибо

Ngiyabonga

Obrigado Misaotra

ข้อ ๑