**LIBRARY MANAGEMENT**

**SYSTEM**

**(DATA STRUCTURE COURSE PROJECT)**

**Submitted by:**

**Abel Jeevan**

**ASI23CA001**

**Abhijith U**

**ASI23CA002**

**Adwaidh AS**

**ASI23CA004**

**Ajwin KS**

**ASI23CA005**

**Akshay T Biju**

**ASI23CA006**

**Albin Paul Pathadan**

**ASI23CA007**

**PROJECT OVERVIEW:**

KEY FEATURES:

1. Book Management:

* Add New Books**:**
  + Admin can add books by providing details like book ID, title, and author.
  + The availability of the book is automatically set to "Available" upon addition.
* Display All Books**:**
  + The system lists all books in the library, showing details such as the book ID, title, author, and whether the book is available or issued.

2. Search Functionality:

* Search Book by Title:
  + Users can search for a book by providing its title.
  + The system performs a case-insensitive search and displays details like book ID, title, author, and availability status.

3. Book Circulation Management:

* Issue a Book**:**
  + A student can borrow a book by entering their student ID and the book ID.
  + The system checks if the book is available before issuing it. If the book is available, it gets marked as "Issued" (unavailable), and the student’s details (ID and name) are recorded.
* Return a Book**:**
  + A student can return a book by providing their student ID.
  + The system looks up the student and the corresponding book that they borrowed, then updates the book's status to "Available."

4. Availability Tracking**:**

* Each book in the system has an availability status:
  + Available**:** The book is present and can be issued to a student.
  + Issued: The book is currently borrowed by a student, and it cannot be issued until returned.

5.Basic User Management:

* Add and Manage Students:
  + Student details (ID and name) are recorded when a book is issued. This allows the system to track who has borrowed which book.

6.Borrowed Book Tracking:

* + The system keeps track of which student has borrowed which book, ensuring that a student can only return a book that they have borrowed.

**DATA STRUCTURES USED:**

 **Structures:** struct Book and struct Student represent entities with multiple attributes.

 **Arrays:** Arrays of these structures (books[100] and students[100]) store multiple instances of books and students.

 **Primitive Data Types:** Integer variables (bookCount, studentCount, etc.) and character arrays (title, author, etc.) handle basic data operations.

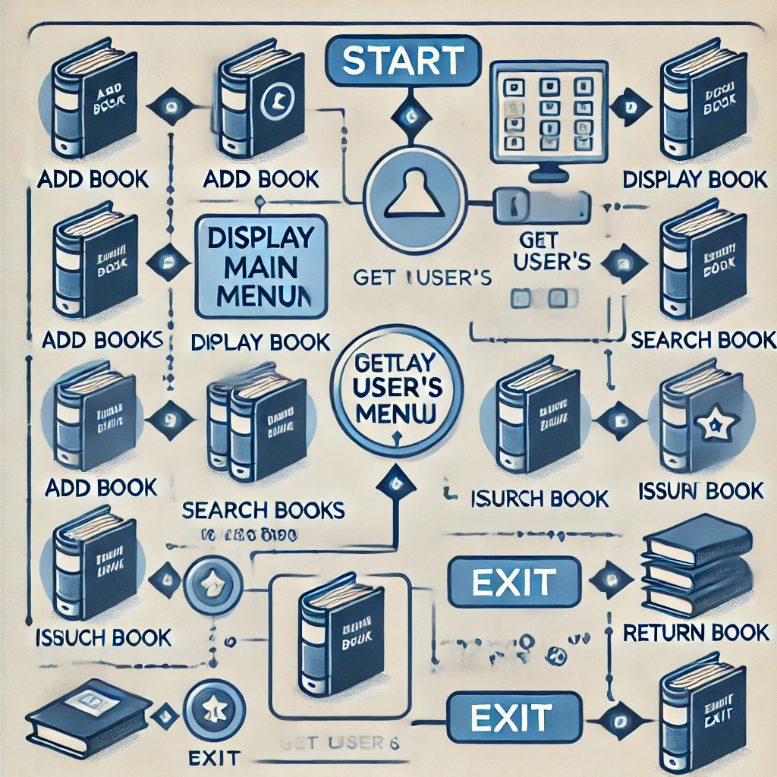
**ALGORITHMS USED**:

**Summary of Algorithms and Their Time Complexities:**

| **Functionality** | **Algorithm** | **Time Complexity** |
| --- | --- | --- |
| Add Book | Insertion | **O(1)** |
| Search Book by Title | Linear Search | **O(n)** |
| Issue Book | Linear Search + Update | **O(n)** |
| Return Book | Linear Search + Update | **O(m + n)** |
| Display All Books | Iteration | **O(n)** |

* **n**: Number of books in the library.
* **m**: Number of students who have borrowed books.

**SYSTEM DESIGN**



**ALGORITHM:**

1. Initialization

* Define structures for Book and Student.
* Initialize arrays for storing books and students.
* Set counters for books and students to zero.

2. Main Function

1. Start the program.
2. Display the main menu options:
   * Add Book
   * Display Books
   * Search Book
   * Issue Book
   * Return Book
   * Exit
3. Get user input for choice.
4. Repeat steps until the user chooses to exit.

3. Add Book

1. If the user chooses "Add Book":
   * Create a new Book instance.
   * Prompt the user to enter the book ID, title, and author.
   * Set the available status to true (1).
   * Store the book in the books array.
   * Increment the book counter.
   * Print a confirmation message.

4. Display Books

1. If the user chooses "Display Books":
   * Iterate through the books array.
   * For each book, print its ID, title, author, and availability status.

5. Search Book

1. If the user chooses "Search Book":
   * Prompt the user to enter the book title.
   * Iterate through the books array.
   * For each book, compare its title with the input.
   * If found, print the book details (ID, title, author, availability).
   * If not found, print a message indicating that the book was not found.

6. Issue Book

1. If the user chooses "Issue Book":
   * Prompt the user to enter the book ID.
   * Check if the book is available by searching the books array.
   * If available:
     + Prompt for student ID and name.
     + Store student details in the students array.
     + Mark the book as issued (set available to false).
     + Increment the student counter.
     + Print a confirmation message.
   * If not available, print a message indicating that the book cannot be issued.

7. Return Book

1. If the user chooses "Return Book":
   * Prompt the user to enter the book ID.
   * Check the books array for the corresponding book.
   * If found:
     + Mark the book as available (set available to true).
     + Clear the student’s borrow record.
     + Print a confirmation message.
   * If not found, print a message indicating that the book is not in the issued records.

8. Exit

1. If the user chooses "Exit":
   * Print a message indicating that the program is terminating.
   * End the program.

**PROGRAM CODE:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

// Structure for storing book details

struct Book {

int id;

char title[50];

char author[50];

int available;

};

// Structure for storing student details

struct Student {

int id;

char name[50];

int borrowedBookId;

};

// Function prototypes

void addBook();

void displayBooks();

void searchBook();

void issueBook();

void returnBook();

// Variables to store books and students

struct Book books[100];

struct Student students[100];

int bookCount = 0;

int studentCount = 0;

int main() {

int choice;

while(1) {

printf("\n====== Library Management System ======\n");

printf("1. Add Book\n");

printf("2. Display All Books\n");

printf("3. Search Book by Title\n");

printf("4. Issue Book\n");

printf("5. Return Book\n");

printf("6. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch(choice) {

case 1:

addBook();

break;

case 2:

displayBooks();

break;

case 3:

searchBook();

break;

case 4:

issueBook();

break;

case 5:

returnBook();

break;

case 6:

exit(0);

default:

printf("Invalid choice! Please try again.\n");

}

}

return 0;

}

// Function to add a new book to the library

void addBook() {

struct Book newBook;

printf("Enter Book ID: ");

scanf("%d", &newBook.id);

printf("Enter Book Title: ");

scanf(" %[^\n]%\*c", newBook.title);

printf("Enter Author Name: ");

scanf(" %[^\n]%\*c", newBook.author);

newBook.available = 1; // Book is available by default

books[bookCount] = newBook;

bookCount++;

printf("Book added successfully!\n");

}

// Function to display all available books

void displayBooks() {

printf("\n--- List of Available Books ---\n");

for(int i = 0; i < bookCount; i++) {

printf("Book ID: %d\n", books[i].id);

printf("Title: %s\n", books[i].title);

printf("Author: %s\n", books[i].author);

printf("Availability: %s\n", books[i].available ? "Available" : "Issued");

printf("---------------------------\n");

}

}

// Function to search for a book by title

void searchBook() {

char searchTitle[50];

printf("Enter book title to search: ");

scanf(" %[^\n]%\*c", searchTitle);

for(int i = 0; i < bookCount; i++) {

if (strcasecmp(books[i].title, searchTitle) == 0) {

printf("Book Found!\n");

printf("Book ID: %d\n", books[i].id);

printf("Title: %s\n", books[i].title);

printf("Author: %s\n", books[i].author);

printf("Availability: %s\n", books[i].available ? "Available" : "Issued");

return;

}

}

printf("Book not found.\n");

}

// Function to issue a book to a student

void issueBook() {

int bookId, studentId;

printf("Enter Student ID: ");

scanf("%d", &studentId);

printf("Enter Book ID: ");

scanf("%d", &bookId);

// Check if the book exists and is available

for(int i = 0; i < bookCount; i++) {

if(books[i].id == bookId) {

if (books[i].available) {

books[i].available = 0; // Mark the book as issued

// Add student details

struct Student newStudent;

newStudent.id = studentId;

printf("Enter Student Name: ");

scanf(" %[^\n]%\*c", newStudent.name);

newStudent.borrowedBookId = bookId;

students[studentCount] = newStudent;

studentCount++;

printf("Book issued to student successfully!\n");

return;

} else {

printf("Book is already issued.\n");

return;

}

}

}

printf("Book not found.\n");

}

// Function to return a book

void returnBook() {

int studentId;

printf("Enter Student ID: ");

scanf("%d", &studentId);

for(int i = 0; i < studentCount; i++) {

if (students[i].id == studentId) {

int bookId = students[i].borrowedBookId;

// Mark the book as available again

for (int j = 0; j < bookCount; j++) {

if (books[j].id == bookId) {

books[j].available = 1;

printf("Book returned successfully!\n");

return;

}

}

}

}

printf("Student or borrowed book not found.\n");

}

**USE CASE DIAGRAM**

