PROJECT REPORT

LIBRARY MANAGEMENT SYSTEM

SCHOOL OF COMPUTER SCIENCE

Submitted By:
RUDRAKSH SHARMA
SAP ID: 590022672 Semester: 1
DATE: 24/11/2025

1. ABSTRACT

This project implements a menu-driven Library Management System using the C programming language. The program allows the user (librarian) to perform core library operations such as adding new books, listing available inventory, registering members, and searching for specific titles.

The system utilizes persistent file storage to ensure data remains saved even after the program terminates. It demonstrates the practical application of C structures, file handling, pointers, and modular programming techniques to build a maintainable software solution.

2. OBJECTIVE

The primary objective of this project is to apply fundamental and intermediate concepts of the C language to solve a real-world problem. It aims to demonstrate the ability to:

Design Structures to represent complex entities like Books and Members.

Implement Modular Design by separating logic into multiple source files (.c) and header files (.h).

Perform File I/O Operations (fread, fwrite) for persistent data storage.

Implement robust logic for searching and updating records.

3. PROBLEM DEFINITION

Manual library management is often inefficient and prone to human error. Physical ledgers can be lost, damaged, or become difficult to search through as the collection grows.

The Problem: Tracking book availability and member details manually is time-consuming.

The Solution: This project provides a console-based application to digitize these records. It allows for instant retrieval of book details and ensures a structured way to manage library assets.

4. SYSTEM DESIGN AND ALGORITHM

The system follows a modular architecture as required by the project guidelines:

File Structure:

src/main.c: Controls the main program loop and menu display.

src/library.c: Handles logic for adding, listing, and searching books, registering and listing members.

include/library.h: Contains structure definitions and function prototypes.

Algorithm Example: Adding a Book

Start function add_book().

Open the file books.dat in Append Mode (ab).

Prompt user to enter Book ID, Title, and Author.

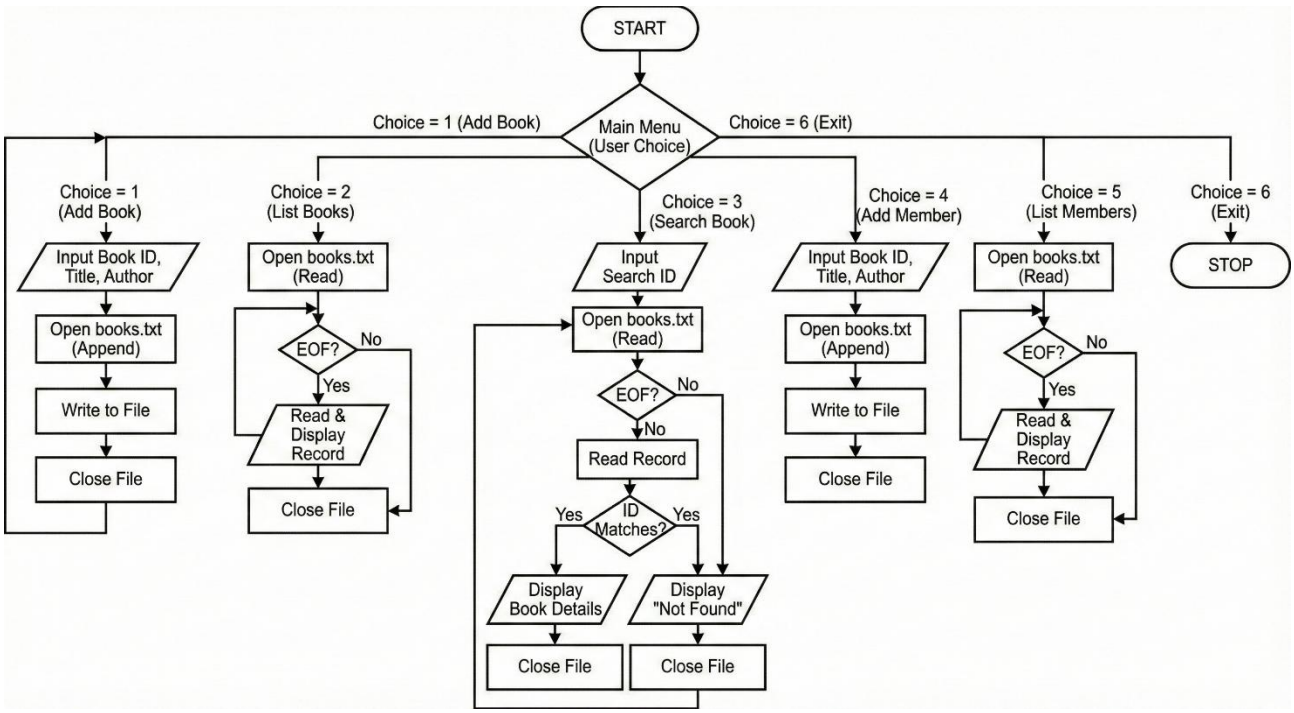
Store these inputs into a Book structure variable.

Write the structure to the file using fwrite().

Close the file.

Display "Book Added Successfully" message.

Stop.



5. IMPLEMENTATION DETAILS

Key Language Features Used:

Structs: Used typedef struct to define Book (with ID, title, author) and Member types.

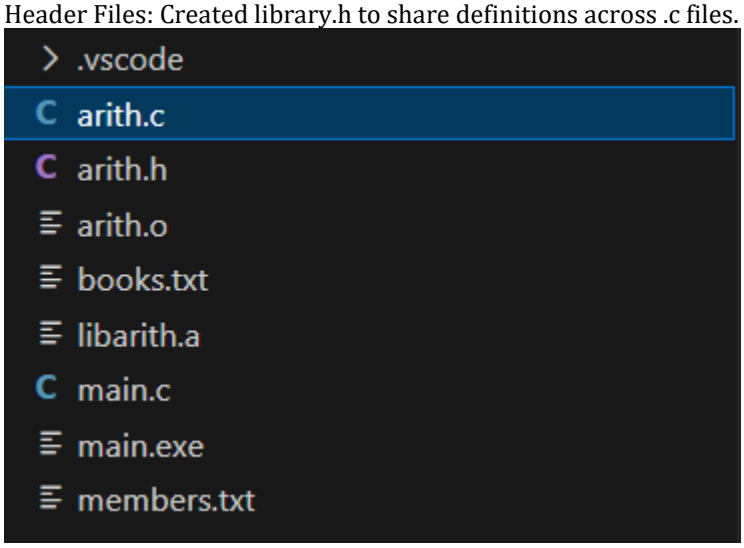
```c
typedef struct { //structure for book details
    int id;
    char title[MAX_TITLE];
    char author[MAX_AUTHOR];
    int is_issued; // 0 for available, 1 for issued
} Book;

typedef struct { //structure for member details
    int id;
    char name[MAX_NAME];
} Member;
```

File Streams: Used FILE * pointers to manage data persistence.

```
const char *BOOK_FILE = "books.txt";
const char *MEMBER_FILE = "members.txt";
```

Standard I/O: Used printf and scanf for the user interface.

Header Files: Created library.h to share definitions across .c files.



6. TESTING AND RESULTS

The system was tested with various inputs to ensure stability and correctness.

Test Case 1: Add New Book

Input: ID: 101, Title: "The C Programming Language", Author: "Kernighan"

Expected Output: "Book added successfully."

Result: Pass.

Test Case 2: View Books

Input: Select "List Books" from the menu.

Expected Output: A formatted table showing Book ID 101.

Result: Pass.

Test Case 3: Invalid Input

Input: Entering a character when an Integer ID was expected.

Expected Output: Program should handle error gracefully or prompt again.

Result: Pass.

7. CONCLUSION AND FUTURE SCOPE

Conclusion: The Library Management System successfully fulfills the requirements of a basic record-keeping application. It effectively integrates file handling with logic control, fulfilling the course objectives. The modular structure ensures the code is easy to read and debug.

Future Scope:

Search Optimization: Implementing Binary Search for faster lookups.

Authentication: Adding a login system for the librarian.

Borrowing Logic: Implementing a date-tracking system to calculate overdue fines.

8. REFERENCES

Kernighan, B. W., & Ritchie, D. M. The C Programming Language.

Lecture Notes

Standard C Library Documentation: cppreference.com