# Digital Library Management System

## Problem Statement

The main objective of this project is to provide a complete automated Library by digitizing every functionality: book-keeping, issuing/returning books, fine generation, advance booking, and report generation. The system will have an Admin module (full control) and a User module (limited privileges). The final delivery is a user-friendly web-based system with a backend database; this document contains a simplified Java console implementation for demonstration.

## Features

* Admin: Add / Remove / Update books and members, view reports.
* User: Browse books, search by title/author/ISBN, issue and return books.
* Issue/Return operations with availability checks and simple fine calculation.
* Data persistence using simple file serialization (for demonstration).
* Clear menu-driven console interface with input validation.

## Algorithm (High-level)

1. Start the application and load persisted data (if any).

2. Show main menu: 1) Admin Login 2) User Menu 3) Exit

3. Admin Login: prompt password; if correct show admin menu to manage books/members and view reports.

4. User Menu: search/display books, issue book (if available), return book (calculate fines), view borrowed list.

5. Persist data on exit (serialize to file).

6. End program.

## Classes and Responsibilities

Book - holds title, author, ISBN, availability.

Member - holds member id, name, borrowed books list.

Library - manages collections, issue/return logic, search.

LibraryManagementSystem - contains main method and UI loop.

## Java Implementation (Console version)

// Full Java code (save as LibraryManagementSystem.java)  
import java.io.\*;  
import java.util.\*;  
  
class Book implements Serializable {  
 String title;  
 String author;  
 String ISBN;  
 boolean available = true;  
  
 public Book(String title, String author, String ISBN) {  
 this.title = title;  
 this.author = author;  
 this.ISBN = ISBN;  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title: %s | Author: %s | ISBN: %s | Available: %s", title, author, ISBN, available);  
 }  
}  
  
class Member implements Serializable {  
 String memberId;  
 String name;  
 ArrayList<String> borrowedISBNs = new ArrayList<>(); // store ISBNs  
  
 public Member(String memberId, String name) {  
 this.memberId = memberId;  
 this.name = name;  
 }  
}  
  
class Library implements Serializable {  
 ArrayList<Book> books = new ArrayList<>();  
 ArrayList<Member> members = new ArrayList<>();  
 // Simple fine policy: 10 units per day late  
  
 public void addBook(Book b) { books.add(b); }  
 public void removeBook(String isbn) {  
 books.removeIf(b -> b.ISBN.equals(isbn));  
 }  
 public Book findByISBN(String isbn) {  
 for (Book b: books) if (b.ISBN.equals(isbn)) return b;  
 return null;  
 }  
 public ArrayList<Book> searchByTitle(String keyword) {  
 ArrayList<Book> res = new ArrayList<>();  
 for (Book b: books) if (b.title.toLowerCase().contains(keyword.toLowerCase())) res.add(b);  
 return res;  
 }  
 public void displayAllBooks() {  
 if (books.isEmpty()) { System.out.println("No books in library."); return; }  
 for (Book b: books) System.out.println(b);  
 }  
  
 public void addMember(Member m) { members.add(m); }  
 public Member findMember(String memberId) {  
 for (Member m: members) if (m.memberId.equals(memberId)) return m;  
 return null;  
 }  
  
 public boolean issueBook(String memberId, String isbn) {  
 Member m = findMember(memberId);  
 Book b = findByISBN(isbn);  
 if (m==null) { System.out.println("Member not found."); return false; }  
 if (b==null) { System.out.println("Book not found."); return false; }  
 if (!b.available) { System.out.println("Book is currently issued."); return false; }  
 b.available = false;  
 m.borrowedISBNs.add(isbn);  
 System.out.println("Book issued successfully.");  
 return true;  
 }  
  
 public double returnBook(String memberId, String isbn, int daysKept) {  
 Member m = findMember(memberId);  
 Book b = findByISBN(isbn);  
 if (m==null || b==null) { System.out.println("Invalid return operation."); return 0; }  
 if (!m.borrowedISBNs.remove(isbn)) { System.out.println("This member did not borrow this ISBN."); return 0; }  
 b.available = true;  
 int allowedDays = 14;  
 double fine = 0;  
 if (daysKept > allowedDays) fine = (daysKept - allowedDays) \* 10.0;  
 System.out.println("Book returned. Fine: " + fine);  
 return fine;  
 }  
}  
  
public class LibraryManagementSystem {  
 static final String DATA\_FILE = "library\_data.ser";  
 public static void main(String[] args) throws Exception {  
 Scanner sc = new Scanner(System.in);  
 Library lib = loadLibrary();  
  
 // seed with sample data if empty  
 if (lib.books.isEmpty()) {  
 lib.addBook(new Book("Clean Code","Robert C. Martin","9780132350884"));  
 lib.addBook(new Book("Introduction to Algorithms","Cormen et al","9780262033848"));  
 lib.addBook(new Book("Effective Java","Joshua Bloch","9780134685991"));  
 }  
 if (lib.members.isEmpty()) {  
 lib.addMember(new Member("M001","Alice"));  
 lib.addMember(new Member("M002","Bob"));  
 }  
  
 while (true) {  
 System.out.println(\"\\n--- Digital Library ---\");  
 System.out.println(\"1. Admin Login\");  
 System.out.println(\"2. User Menu\");  
 System.out.println(\"3. Save & Exit\");  
 System.out.print(\"Choose: \");  
 int ch = sc.nextInt(); sc.nextLine();  
 if (ch==1) {  
 System.out.print(\"Enter admin password: \");  
 String pwd = sc.nextLine();  
 if (!pwd.equals(\"admin123\")) { System.out.println(\"Wrong password.\"); continue; }  
 adminMenu(sc, lib);  
 } else if (ch==2) {  
 userMenu(sc, lib);  
 } else if (ch==3) {  
 saveLibrary(lib);  
 System.out.println(\"Data saved. Exiting.\"); break;  
 }  
 }  
 sc.close();  
 }  
  
 static void adminMenu(Scanner sc, Library lib) {  
 while (true) {  
 System.out.println(\"\\n--- Admin Menu ---\");  
 System.out.println(\"1. Add Book\");  
 System.out.println(\"2. Remove Book\");  
 System.out.println(\"3. Add Member\");  
 System.out.println(\"4. View All Books\");  
 System.out.println(\"5. View All Members\");  
 System.out.println(\"6. Back\");  
 System.out.print(\"Choose: \"); int ch = sc.nextInt(); sc.nextLine();  
 if (ch==1) {  
 System.out.print(\"Title: \"); String t = sc.nextLine();  
 System.out.print(\"Author: \"); String a = sc.nextLine();  
 System.out.print(\"ISBN: \"); String i = sc.nextLine();  
 lib.addBook(new Book(t,a,i));  
 System.out.println(\"Book added.\");   
 } else if (ch==2) {  
 System.out.print(\"ISBN to remove: \"); String i = sc.nextLine();  
 lib.removeBook(i); System.out.println(\"Book removed (if existed).\");   
 } else if (ch==3) {  
 System.out.print(\"Member Id: \"); String id = sc.nextLine();  
 System.out.print(\"Name: \"); String name = sc.nextLine();  
 lib.addMember(new Member(id,name)); System.out.println(\"Member added.\");   
 } else if (ch==4) {  
 lib.displayAllBooks();  
 } else if (ch==5) {  
 for (Member m: lib.members) {  
 System.out.println(\"Id:\"+m.memberId+\" Name:\"+m.name+\" Borrowed:\"+m.borrowedISBNs);  
 }  
 } else break;  
 }  
 }  
  
 static void userMenu(Scanner sc, Library lib) {  
 while (true) {  
 System.out.println(\"\\n--- User Menu ---\");  
 System.out.println(\"1. Search by Title\");  
 System.out.println(\"2. Display All Books\");  
 System.out.println(\"3. Issue Book\");  
 System.out.println(\"4. Return Book\");  
 System.out.println(\"5. Back\");  
 System.out.print(\"Choose: \"); int ch = sc.nextInt(); sc.nextLine();  
 if (ch==1) {  
 System.out.print(\"Keyword: \"); String k = sc.nextLine();  
 var res = lib.searchByTitle(k);  
 for (Book b: res) System.out.println(b);  
 } else if (ch==2) {  
 lib.displayAllBooks();  
 } else if (ch==3) {  
 System.out.print(\"Member Id: \"); String mid = sc.nextLine();  
 System.out.print(\"ISBN: \"); String isbn = sc.nextLine();  
 lib.issueBook(mid,isbn);  
 } else if (ch==4) {  
 System.out.print(\"Member Id: \"); String mid = sc.nextLine();  
 System.out.print(\"ISBN: \"); String isbn = sc.nextLine();  
 System.out.print(\"Days kept: \"); int days = sc.nextInt(); sc.nextLine();  
 lib.returnBook(mid,isbn,days);  
 } else break;  
 }  
 }  
  
 static Library loadLibrary() {  
 try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(DATA\_FILE))) {  
 return (Library) ois.readObject();  
 } catch (Exception e) {  
 return new Library();  
 }  
 }  
  
 static void saveLibrary(Library lib) {  
 try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(DATA\_FILE))) {  
 oos.writeObject(lib);  
 } catch (Exception e) {  
 System.out.println(\"Failed to save data: \" + e.getMessage());  
 }  
 }  
}

## Sample Output (one run)

--- Digital Library ---  
1. Admin Login  
2. User Menu  
3. Save & Exit  
Choose: 2  
  
--- User Menu ---  
1. Search by Title  
2. Display All Books  
3. Issue Book  
4. Return Book  
5. Back  
Choose: 2  
Title: Clean Code | Author: Robert C. Martin | ISBN: 9780132350884 | Available: true  
Title: Introduction to Algorithms | Author: Cormen et al | ISBN: 9780262033848 | Available: true  
Title: Effective Java | Author: Joshua Bloch | ISBN: 9780134685991 | Available: true  
  
(Then user issues a book and returns after 16 days: fine computed = 20.0)

## Conclusion

This console implementation demonstrates core library functions and can be extended into a web application using a backend database (MySQL/Postgres) and a frontend (React/Angular).