# Library Management System - Mini Project

## Project Overview

Develop a console-based Library Management System that helps librarians manage books, track borrowers, and handle book lending operations using core Java programming concepts.

## Learning Objectives

* Master Object-Oriented Programming principles
* Work with multiple classes and their relationships
* Implement ArrayList and HashMap collections
* Practice string manipulation and date handling
* Apply inheritance and polymorphism concepts
* Develop problem-solving skills through real-world scenarios

## Project Requirements

### Core Features (Must Implement)

1. **Book Class**: Create a Book class with:
   * Book ID (String)
   * Title (String)
   * Author (String)
   * ISBN (String)
   * Publication Year (int)
   * Available copies (int)
   * Total copies (int)
2. **Member Class**: Create a Member class with:
   * Member ID (String)
   * Name (String)
   * Email (String)
   * Phone Number (String)
   * Books borrowed (ArrayList)
   * Maximum books allowed (default: 3)
3. **Library Operations**: Implement the following functionalities:
   * Add new books to the library
   * Register new members
   * Issue books to members
   * Return books from members
   * Search books by title, author, or ISBN
   * Display all available books
   * Display member details and borrowed books
   * Generate overdue book reports

### Advanced Features (Bonus)

* Implement fine calculation for late returns (₹5 per day)
* Book reservation system
* Generate library statistics (most popular books, active members)
* Export member/book data to CSV file
* Implement different member types (Student, Faculty, General) with different borrowing limits

## Technical Requirements

* Use ArrayList to store books and members
* Implement proper getter/setter methods
* Use HashMap for quick book/member lookups
* Include at least 2 constructors per class
* Handle exceptions (book not found, member limit exceeded)
* Use LocalDate for date operations (issue/return dates)

## Sample Interaction

=== LIBRARY MANAGEMENT SYSTEM ===  
1. Add Book  
2. Add Member   
3. Issue Book  
4. Return Book  
5. Search Books  
6. View Member Details  
7. Display Available Books  
8. Library Statistics  
9. Exit  
  
Enter your choice: 3  
  
=== ISSUE BOOK ===  
Enter Member ID: M001  
Enter Book ID: B001  
  
Member: John Smith (john@email.com)  
Book: "The Java Programming Language" by James Gosling  
Issue Date: 2025-08-06  
Due Date: 2025-08-20  
  
Book issued successfully!  
Remaining copies: 2

## Class Structure Suggestion

Library (Main class)  
├── Book  
├── Member   
├── Transaction (Optional)  
└── LibraryManager (Handles operations)

## Data Requirements

**Pre-populate your system with:** - At least 10 books from different genres - 5 sample members - Few sample transactions to demonstrate functionality

## Submission Guidelines

* All source code files (.java)
* Sample data file (if implementing file operations)
* Documentation explaining your design choices
* Test cases demonstrating each feature
* Screenshots of program execution

## Evaluation Rubric (100 points)

| Criteria | Points | Description |
| --- | --- | --- |
| **Object-Oriented Design** | 25 | Proper use of classes, encapsulation, inheritance |
| **Core Functionality** | 35 | All required features working correctly |
| **Data Structures & Collections** | 15 | Effective use of ArrayList, HashMap |
| **Error Handling** | 10 | Proper exception handling and validation |
| **Code Quality** | 10 | Clean code, proper naming, comments |
| **User Interface** | 5 | Clear menu system and user interaction |

## Implementation Timeline

* **Week 1**: Design classes, create Book and Member classes
* **Week 2**: Implement core operations (add, issue, return)
* **Week 3**: Add search functionality and data validation
* **Week 4**: Testing, debugging, and bonus features

## Key Programming Concepts Covered

* **Classes and Objects**: Book, Member, Library classes
* **Encapsulation**: Private fields with public methods
* **Collections**: ArrayList for dynamic storage
* **Exception Handling**: Custom exceptions for library operations
* **String Manipulation**: Book search and formatting
* **Date Operations**: Issue/return date calculations
* **File I/O**: Optional data persistence

## Sample Books to Include

1. “Effective Java” by Joshua Bloch (Programming)
2. “Clean Code” by Robert Martin (Programming)
3. “To Kill a Mockingbird” by Harper Lee (Fiction)
4. “1984” by George Orwell (Fiction)
5. “The Alchemist” by Paulo Coelho (Philosophy)

## Extension Ideas

If you finish early, consider adding: - Book categories and filtering - Member activity history - Automated email notifications (simulation) - Book recommendation system - Multiple library branch management

**Remember**: Focus on implementing clean, working code for core features before attempting bonus functionality. Good luck!