■ Library Management System – Documentation

1. Introduction

The Library Management System (LMS) is a console-based Java application that helps manage books and users in a library. It provides basic features such as adding books, searching for books, borrowing/returning books, and managing library users. This project demonstrates the use of Object-Oriented Programming (OOP) concepts like encapsulation, inheritance, and polymorphism, along with design principles like DAO (Data Access Object) and Service Layer architecture.

2. Objectives

• To create a digital platform for managing library resources.
• To enable users to borrow and return books efficiently.
• To allow administrators to add and manage books and users.
• To demonstrate the layered architecture (DAO → Service → UI).

3. System Architecture

The system follows a 3-layered architecture:

1. DAO Layer (Data Access Object): Handles storage and retrieval of data (Books, Users).
2. Service Layer: Contains business logic (borrowing, returning, searching).
3. UI Layer (Menu Layer): Console-based menu for interacting with users.

4. Features

Book Management:
- Add new books.
- List all available books.
- Search books by title, author, or ISBN.

Borrow/Return:
- Borrow books (if copies are available).
- Return borrowed books.

User Management:
- Add users (Admin/Member).
- List all registered users.

5. Technologies Used

• Programming Language: Java
• Collections Used: HashMap, ArrayList
• Concepts Used: OOP, Encapsulation, Layered Design, Scanner Input

6. Classes and Responsibilities

1. Main.java → Entry point of the program.
2. Book.java → Represents a book object.
3. User.java → Represents a user object.
4. Transaction.java → Records borrow/return transactions.
5. BookDAO.java → Handles book storage and search operations.
6. UserDAO.java → Handles user storage and retrieval.
7. LibraryService.java → Implements core library operations.
8. LibraryMenu.java → Provides the console-based user interface.

7. Sample Execution Flow

Example run of the program:

1. Add Book → Added: ID:1 | Java Programming by James Gosling | ISBN:12345 | Copies:3
2. List Books → ID:1 | Java Programming by James Gosling | ISBN:12345 | Copies:3
3. Search Books → Enter keyword 'Java' → Found Java Programming
4. Borrow Book → Book borrowed successfully.
5. Return Book → Book returned successfully.
6. Add User → Added: ID:1 | Alice (USER)
7. List Users → ID:1 | Alice (USER)
0. Exit → Goodbye!

8. Advantages

• Simple and easy to use.
• Demonstrates clear separation of concerns.
• Scalable – more features can be added (e.g., fines, book reservations).

9. Limitations

• Data is stored only in memory (not persistent).
• No authentication system.
• Limited to console-based interface.

10. Future Enhancements

• Add database integration (MySQL/SQLite).
• Introduce GUI (Swing/JavaFX) or Web UI.
• Implement user login and authentication.
• Track overdue books and fines.