Library Management System – Final Project Report

1. Introduction

The Library Management System (LMS) is a web-based application designed to automate core library operations such as book management, user handling, issuing and returning books, overdue tracking, and intelligent recommendations. The project implements key Software Construction and Design (SCD) concepts including Inheritance, Multithreading, Synchronization, Locks, Generics, Exception Handling, GUI development, Unit Testing, and basic AI features. The backend is built using Java Spring Boot, the frontend uses HTML, CSS, and JavaScript, and the database is PostgreSQL running inside Docker using WSL.

2. Objectives

• Automate core library management tasks.

• Implement role-based authentication (Admin and Student).

• Provide separate dashboards for Admin and Student users.

• Ensure a simple and user-friendly interface.

• Maintain accurate and consistent data records.

• Apply all required SCD concepts.

• Integrate AI features such as recommendations, smart search, overdue prediction, and chatbot.

• Ensure synchronization and data consistency.

• Use Dockerized PostgreSQL as the database backend.

3. Tools & Technologies Used

• Java Spring Boot

• HTML, CSS, JavaScript

• PostgreSQL (Dockerized)

• Docker and WSL

• Spring Data JPA

• JUnit and Mockito

• Maven

• Eclipse / IntelliJ IDEA

4. System Architecture

The LMS follows a 4-layer architecture:
1. Presentation Layer – HTML, CSS, JavaScript
2. Service Layer – Spring Boot
3. Repository Layer – Spring Data JPA
4. Database Layer – PostgreSQL (Docker)

This architecture ensures modularity, scalability, and clean separation of concerns.

5. Core Features Implemented

• Login and Signup system (Admin and Student)

• Role-based dashboards

• Book CRUD operations

• Issue and return management

• Category management

• Search and filter functionality

• Overdue book notifications

6. AI Features Implemented

• Rule-based book recommendation system

• Smart auto-suggest search

• Overdue prediction logic

• Rule-based chatbot for library assistance

7. Implementation of SCD Concepts

• Inheritance using Item as parent class with Book, Newspaper, and Magazine as child classes

• Multithreading with background overdue checker thread

• Thread control using start, sleep, and interrupt

• Synchronization using ReentrantLock during issue and return operations

• Generics using a generic Library<T> class

• Exception handling using custom exceptions and global handler

• GUI development using HTML, CSS, and JavaScript

• Unit testing using JUnit and Mockito

8. Database Design – PostgreSQL

Main tables:
• users (id, name, email, password, role)
• books (id, title, author, category, quantity)
• issued_books (id, user_id, book_id, issue_date, due_date, return_date)
• categories (id, name)

Relationships:
• One-to-Many: user → issued_books
• One-to-Many: book → issued_books

9. AI Module Design

The AI module consists of a RecommendationEngine for book suggestions, a SmartSearchEngine for keyword-based predictions, an OverduePredictor for due-date analysis, and a rule-based LibraryChatbot for user assistance.

10. Conclusion

The Library Management System successfully fulfills academic requirements by implementing essential SCD concepts along with AI-based features. The use of Dockerized PostgreSQL improves portability and reliability, making the system suitable for evaluation and future enhancement.