Library Management System - Architecture Documentation

March 19, 2025

1 Overview

This document outlines the architecture and implementation details of the Library Management System, focusing on Object-Oriented Programming (OOP) principles and design patterns.

2 Architecture Diagram

```
+------------------+      +------------------+      +------------------+
|      Model       |      |     Service      |      |      Test        |
+------------------+      +------------------+      +------------------+
| - Book          |      | - BookService    |      | - BookServiceTest|
+------------------+      +------------------+      +------------------+
```

3 OOP Principles Implementation

3.1 Encapsulation

Encapsulation is demonstrated in the `Book` class through:

- Private fields (isbn, title, author, publicationYear, isAvailable)
- Public getters and setters
- Constructor with validation

3.2 Abstraction

Implemented in the `BookService` class through:

- High-level interface for book operations
- Hidden implementation details
- Clear method signatures

3.3 Inheritance

The system is designed to be extensible through inheritance. Future classes can extend `Book` for specialized types (e.g., `EBook`, `AudioBook`).

3.4 Polymorphism

Polymorphism is demonstrated through:

- Method overloading in service classes
- Generic collections handling different book types
- Optional return types

4 Class Definitions

4.1 Book Class

Purpose: Represents a book entity in the system.
Responsibilities:

- Store book information
- Provide access to book properties
- Maintain book state (availability)

4.2 BookService Class

Purpose: Manages book-related operations.
Responsibilities:

- Add/remove books

- Find books by ISBN
- Update book availability
- Maintain book collection

5 Testing Strategy

5.1 Unit Testing

- Individual class testing
- Method-level testing
- Edge case coverage

5.2 Test Coverage

- Positive test cases
- Negative test cases
- Boundary conditions

6 Implementation Details

6.1 Data Structures

- `ArrayList` for book storage
- `Optional` for null-safe operations
- Stream API for functional operations

6.2 Error Handling

- Null checks
- Input validation
- Defensive programming

6.3 Code Organization

- Package-based structure
- Clear separation of concerns
- Consistent naming conventions

7 Future Enhancements

1. Database Integration
2. User Interface
3. Additional Book Types
4. Advanced Search Features
5. Borrowing System

8 Dependencies

- JUnit 5 for testing
- Mockito for mocking
- Java 17 features