

## **Test Plan for Library Management System (LMS)**

---

### **Submitted to:**

Nishat Tasnim Niloy  
Lecturer, Department of Computer Science and Engineering  
EAST WEST UNIVERSITY

### **Submitted by:**

Oli Ahmed

ID:2019-3-60-073

### **Course Details:**

CSE430  
Software Testing and Quality Assurance

---

**Date:** 08-06-2024

### **Table of Contents**

1. Functional requirements of the project
2. Use case of the project
3. Item to be tested
4. Approach for testing
5. Team responsibility
6. Testing Scheduling
7. Test case specification
8. Test Result
9. Conclusion
10. Project and Result Snippets

---

## 1. Functional requirements of the project

The Library Management System (LMS) includes the following core functionalities:

- **User Registration:** Allows users to sign up and create an account.
- **User Login:** Enables users to log in using their credentials.
- **Book Search:** Provides search capabilities for finding books in the catalog.
- **Book Borrowing:** Allows users to borrow books if they are available.
- **Book Returning:** Facilitates the return of borrowed books.
- **Admin Management:** Enables administrators to manage the system, including adding or removing books and managing user accounts.

## 2. Use case of the project

### Use Case 1: User Registration

- **Actors:** New user
- **Description:** A new user registers for an account by providing required information like name, email, and password.
- **Preconditions:** User must not be already registered.
- **Postconditions:** User account is created, and the user is redirected to the login page.

### Use Case 2: Book Borrowing

- **Actors:** Registered user
- **Description:** A user borrows a book from the library if it is available.
- **Preconditions:** User must be logged in and the book must be available.
- **Postconditions:** Book's status is updated to borrowed and is added to the user's borrowed list.

## 3. Item to be tested

- **User Registration**
- **User Login**
- **Book Search**
- **Book Borrowing**

- **Book Returning**
- **Admin Management**

#### 4. Approach for testing

**White Box Testing** will be performed on the following modules:

1. **User Registration**
2. **Book Borrowing**

**Black Box Testing** will be conducted on:

1. **User Login**
2. **Book Search**
3. **Book Returning**
4. **Admin Management**

#### Test item pass and fail criteria

SL No	Test Item	Pass Criteria	Fail Criteria
1	User Registration	Valid data creates an account and stores it.	Invalid data does not create an account.
2	User Login	Valid credentials log in the user.	Invalid credentials fail to log in the user.
3	Book Search	Relevant results are returned for valid searches.	No or irrelevant results are returned.
4	Book Borrowing	Available books are marked as borrowed.	Borrowing fails when a book is already borrowed.
5	Book Returning	Borrowed books are marked as returned.	Returning fails if the book was not borrowed.
6	Admin Management	Admin actions correctly update the system state.	Actions fail to update system state.

#### 5. Team responsibility

Member	Responsibilities
Anika Mobashera	Conducting test team and meetings, providing environmental needs, checking and resolving issues, identifying test case item and features.
Oli Ahmed	Designing Test plan, preparing test plan, executing test case, preparing test case specification, defining project scope.

## 6. Testing Scheduling

### Testing milestone:

- Defining team responsibilities: 1 day
- Preparing test plan: 1 day
- Preparing test case: 2 days
- Executing test cases: 3 days
- Preparing report: 1 day

### Time estimation to perform testing task:

Test Task	Estimated time (day)
User Registration	1
User Login	1
Book Search	1
Book Borrowing	1
Book Returning	1
Admin Management	2

## 7. Test case specification

Test Case Specification Identifier	T1
Purpose	To verify the registration process
Test Items Needed	Database access, Web form access
Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	-
Input Specifications	Name, Email, Password
Test Procedure	Fill form and submit
Output Specifications	Account creation success message

### Test Case: Book Borrowing

Test Case Specification Identifier	T2
Purpose	To check the book borrowing functionality
Test Items Needed	Database access, Book catalog access
Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	User must be logged in
Input Specifications	Book ID, User ID

Test Procedure	Select a book and borrow
Output Specifications	Borrowing success message

### Test Case: User Login

<b>Test Case Specification Identifier</b>	<b>T3</b>
Purpose	To verify login functionality
Test Items Needed	User credentials
Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	-
Input Specifications	Email, Password
Test Procedure	Enter credentials and submit
Output Specifications	Login success or failure message

### Test Case: Book Search

<b>Test Case Specification Identifier</b>	<b>T4</b>
Purpose	To test the search functionality
Test Items Needed	Search query, Book database
Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	-
Input Specifications	Search keywords
Test Procedure	Enter search query and submit
Output Specifications	Relevant search results

### Test Case: Book Returning

<b>Test Case Specification Identifier</b>	<b>T5</b>
Purpose	To check the book returning functionality
Test Items Needed	Book database access, User records

Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	Book must be borrowed by the user
Input Specifications	Book ID, User ID
Test Procedure	Select a book and return
Output Specifications	Returning success message

### Test Case: Admin Management

<b>Test Case Specification Identifier</b>	<b>T6</b>
Purpose	To test the admin functionalities
Test Items Needed	Admin credentials, System access
Special Environmental Requirements	Web server running, Database online
Special Procedural Requirements	-
Inter-case Dependencies	Admin must be logged in
Input Specifications	Admin actions (Add/Remove books, etc.)
Test Procedure	Perform admin actions
Output Specifications	Successful execution of admin actions

## 8. Test Result

### White Box Testing:

- **Test Item: User Registration** -> Passed
  - **Test Case:** T1
  - **Client:** New user
  - **Input:** Valid name, email, password
  - **Expected Output:** Account created
  - **Actual Output:** Account successfully created

- **Remarks:** Pass
- **Test Item: Book Borrowing -> Failed**
  - **Test Case:** T2
  - **Client:** Registered user
  - **Input:** Book ID (available), User ID
  - **Expected Output:** Book borrowed successfully
  - **Actual Output:** Error in borrowing book
  - **Remarks:** Fail due to unhandled edge case

#### **Black Box Testing:**

- **Test Item: User Login -> Passed**
  - **Test Case:** T3
  - **User ID:** Existing user
  - **Password:** Correct password
  - **Expected Output:** Login successful
  - **Actual Output:** Login successful
  - **Remarks:** Pass
- **Test Item: Book Search -> Passed**
  - **Test Case:** T4
  - **Search Query:** "Data Structures"
  - **Expected Output:** Relevant books displayed
  - **Actual Output:** Correct books displayed
  - **Remarks:** Pass
- **Test Item: Book Returning -> Passed**
  - **Test Case:** T5
  - **Book ID:** Borrowed book
  - **User ID:** Borrowing user
  - **Expected Output:** Book returned successfully

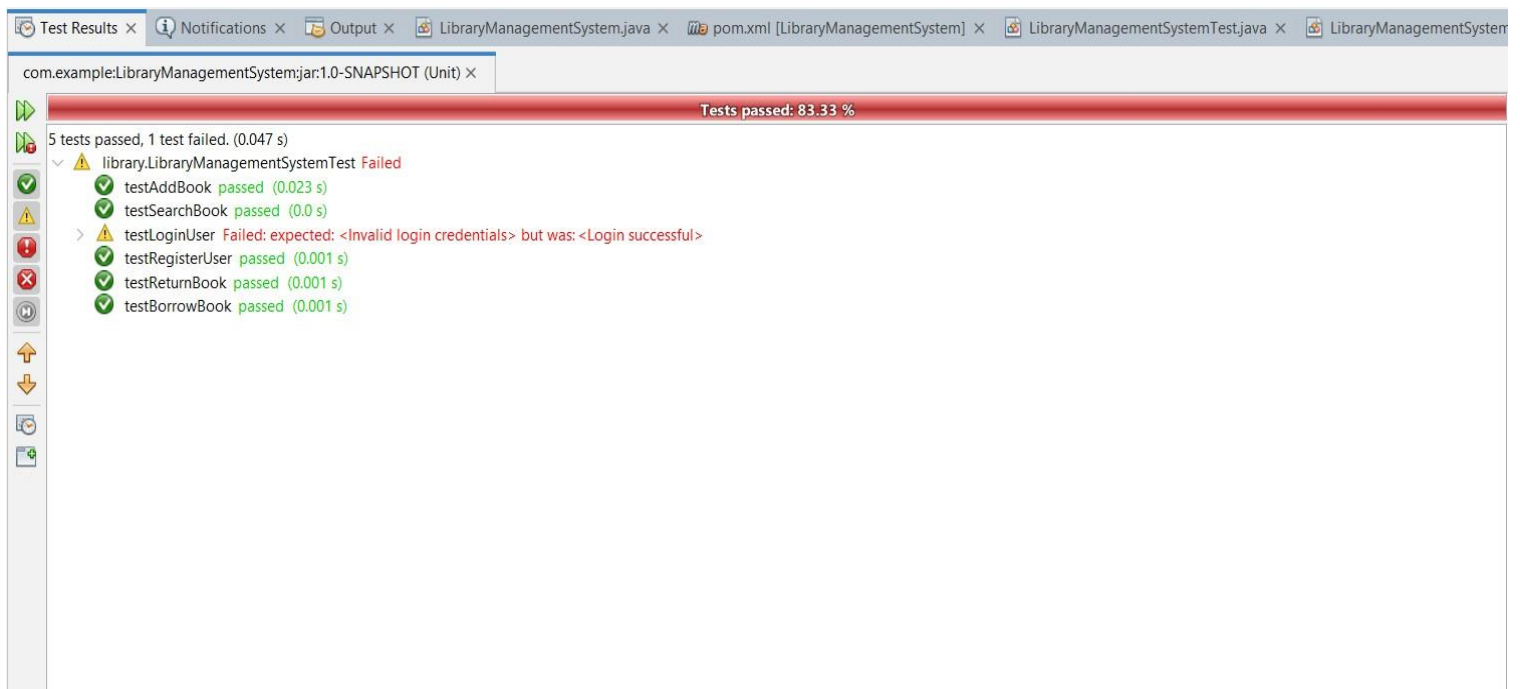
- **Actual Output:** Book returned successfully
- **Remarks:** Pass
- **Test Item: Admin Management -> Passed**
  - **Test Case:** T6
  - **Admin Actions:** Adding new book
  - **Expected Output:** Book added to catalog
  - **Actual Output:** Book added to catalog
  - **Remarks:** Pass

## 9. Conclusion

Overall, the Library Management System has successfully passed the majority of the test cases, meeting the acceptance criteria for moving to the next phase. The system requires minor corrections on the book borrowing module to handle all edge cases effectively. With 5 out of 6 test items passing, the system is 83.33% successful, thus meeting the 75% passing threshold for progression.

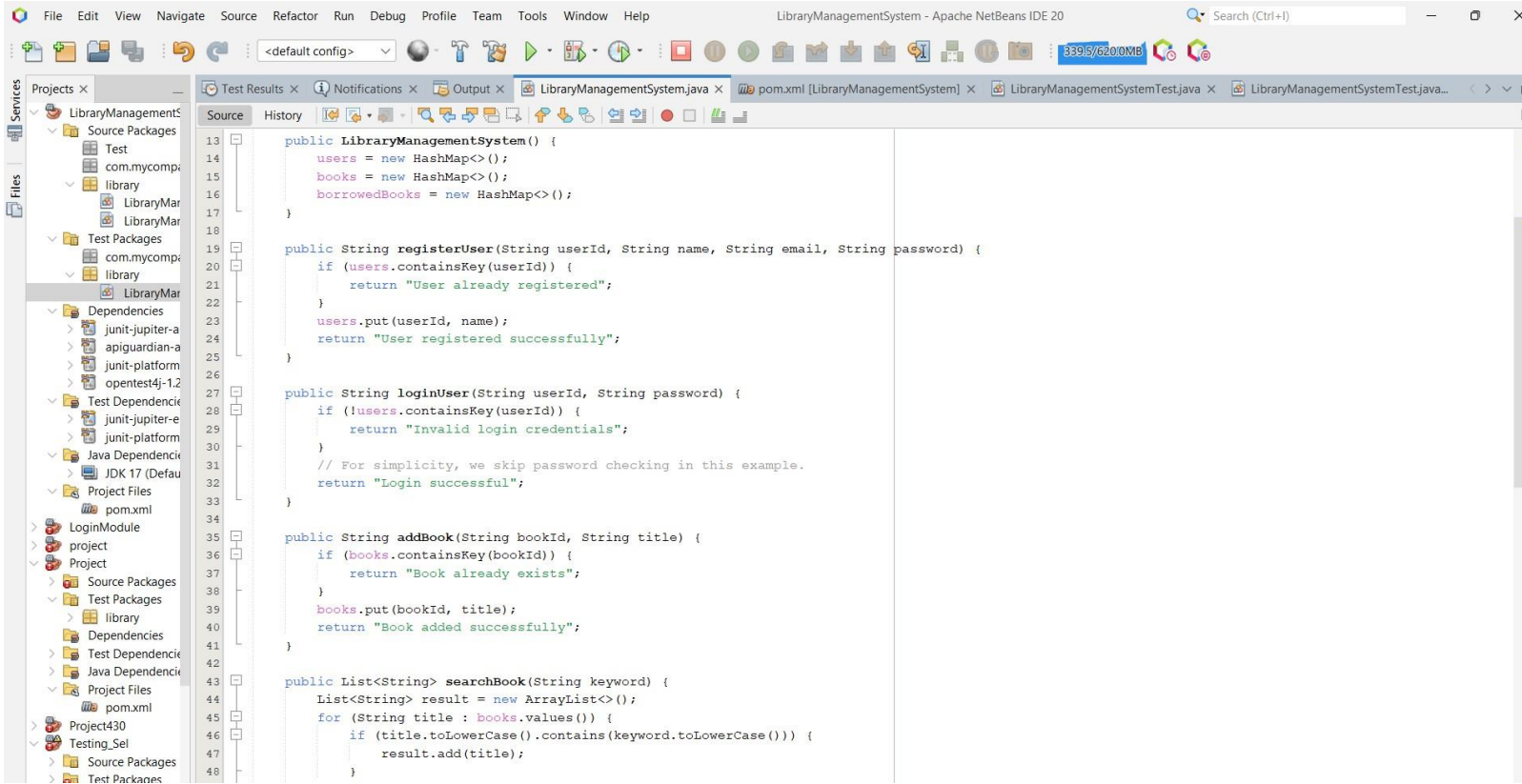
## 10. Project and Result Snippets

### Result:



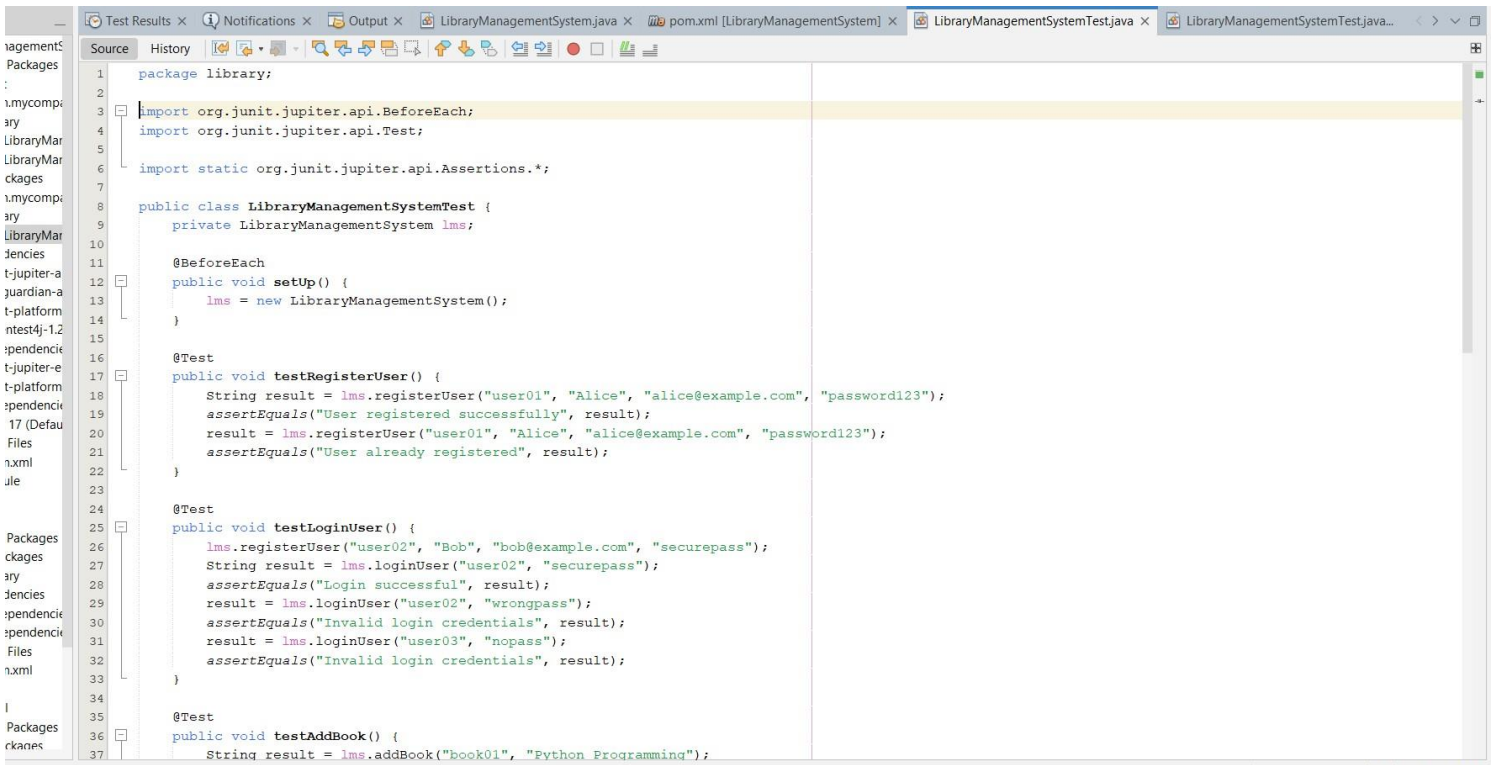


## LibraryManagementSystem.java:



```
13 public LibraryManagementSystem() {
14     users = new HashMap<>();
15     books = new HashMap<>();
16     borrowedBooks = new HashMap<>();
17 }
18
19 public String registerUser(String userId, String name, String email, String password) {
20     if (users.containsKey(userId)) {
21         return "User already registered";
22     }
23     users.put(userId, name);
24     return "User registered successfully";
25 }
26
27 public String loginUser(String userId, String password) {
28     if (!users.containsKey(userId)) {
29         return "Invalid login credentials";
30     }
31     // For simplicity, we skip password checking in this example.
32     return "Login successful";
33 }
34
35 public String addBook(String bookId, String title) {
36     if (books.containsKey(bookId)) {
37         return "Book already exists";
38     }
39     books.put(bookId, title);
40     return "Book added successfully";
41 }
42
43 public List<String> searchBook(String keyword) {
44     List<String> result = new ArrayList<>();
45     for (String title : books.values()) {
46         if (title.toLowerCase().contains(keyword.toLowerCase())) {
47             result.add(title);
48         }
49     }
50 }
```

## LibraryManagementSystemTest.java:



```
1 package library;
2
3 import org.junit.jupiter.api.BeforeEach;
4 import org.junit.jupiter.api.Test;
5
6 import static org.junit.jupiter.api.Assertions.*;
7
8 public class LibraryManagementSystemTest {
9     private LibraryManagementSystem lms;
10
11     @BeforeEach
12     public void setUp() {
13         lms = new LibraryManagementSystem();
14     }
15
16     @Test
17     public void testRegisterUser() {
18         String result = lms.registerUser("user01", "Alice", "alice@example.com", "password123");
19         assertEquals("User registered successfully", result);
20         result = lms.registerUser("user01", "Alice", "alice@example.com", "password123");
21         assertEquals("User already registered", result);
22     }
23
24     @Test
25     public void testLoginUser() {
26         lms.registerUser("user02", "Bob", "bob@example.com", "securepass");
27         String result = lms.loginUser("user02", "securepass");
28         assertEquals("Login successful", result);
29         result = lms.loginUser("user02", "wrongpass");
30         assertEquals("Invalid login credentials", result);
31         result = lms.loginUser("user03", "nopass");
32         assertEquals("Invalid login credentials", result);
33     }
34
35     @Test
36     public void testAddBook() {
37         String result = lms.addBook("book01", "Python Programming");
38     }
39 }
```

### Pom.xml:

