Documentation for the Library Management System Project

1. Clean Code Practices

Clean code practices ensure that the code is readable, maintainable, and easy to understand. Below are of how clean code practices are implemented in this project:

- Meaningful Naming: The class and method names are descriptive and self-explanatory. For example, the method ***isAvailable()*** clearly indicates its purpose, and the class name ***Book r***eflects its role in the system.
- Each class has a single responsibility. For example, the ***Library*** class is responsible only for managing books and users:
- The ***Library*** class focuses solely on managing books and users, adhering to the Single Responsibility Principle.

2. Project Explanation

What the Project Does

This project is a Library Management System that allows users to add books to the library, add users to the system, issue books to users (if available and within borrowing limits), and search for books by title.

How It Works

The system consists of three main classes.

- ***Book***: Represents a book with attributes like ***title, author,*** and ***isAvailable.***
- ***User***: Represents a user with attributes like ***name*** and ***borrowedBooks.***
- ***Library***: Manages the collection of books and users, and provides methods for issuing, returning, and searching books.

Test Cases

Three unit tests are written to validate the functionality of the application.

- Test Book Availability: Validates that a newly added book is available for borrowing.
- Test User Borrowing Limit: Ensures a user cannot borrow more than the allowed number of books.
- Test Search Functionality: Validates the search functionality in the library.

3. Dependencies

The project uses the following dependencies.

- JUnit 5: Used for writing and running unit tests and added to the pom.xml file:

- Maven Surefire Plugin: Used to execute tests during the build process and dded to the pom.xml file:

Where Dependencies Are Sourced From

- All dependencies are sourced from the Maven Central Repository.

4. Problem

Constant "*Resource not accessible by integration*" errors on GitHub Actions workflow after commit and push.

The problem was resolved after Searching through Google and ChatGPT by going to GitHub Setting and selecting "*Read and write permissions"* under Workflow permission.