1. **Git**

I have used git to upload all the relevant files of the project. First I uploaded a readme which includes the details of the project then uploaded the java files the src files nd the test files on git.

2. **UML**

UML is a standard language for better understanding the project visually. I have developed three diagrams of the project which includes the project use case diagram, activity diagram, and class diagram.

The link to my uml diagrams is: https://github.com/MuhammadAli-library/LibraryManagamentSystem/tree/main/images/UML

3. **DDD**

It helped me in developing an approach which lead to abstraction of the project. It helped in understanding the business logic and bridge the gap between reality and code. It is used to ease the development of my project. I have created and added the DDD in my readme file in git. The link to file is: https://github.com/MuhammadAli-library/LibraryManagamentSystem/blob/main/README.md

4. **METRICES**

Metrices are also included in the readme files. The two metrices I have chosen are complexity and security of the system. I have utilized the sonarcube metrices.

5. **CLEAN CODE DEVELOPMENT**

I have utilized the clean code development approach for writing the code. I would like to highlight the points that I have used for clean code development.

1. Build requires only one step. With just one click the project is build successfully.
2. Tests are also executed with just one step.
3. Naming conventions for methods, classes and variables are followed.
4. Good commenting can be found in my code.
5. Each Method does only one thing in my code.
6. My code is consistent.
7. Names are descriptive in my code.
8. My code is simple (KISS) followed

Link to my CCD cheat sheat:

https://github.com/MuhammadAli-library/LibraryManagamentSystem/tree/main/CCD

6. BUILD

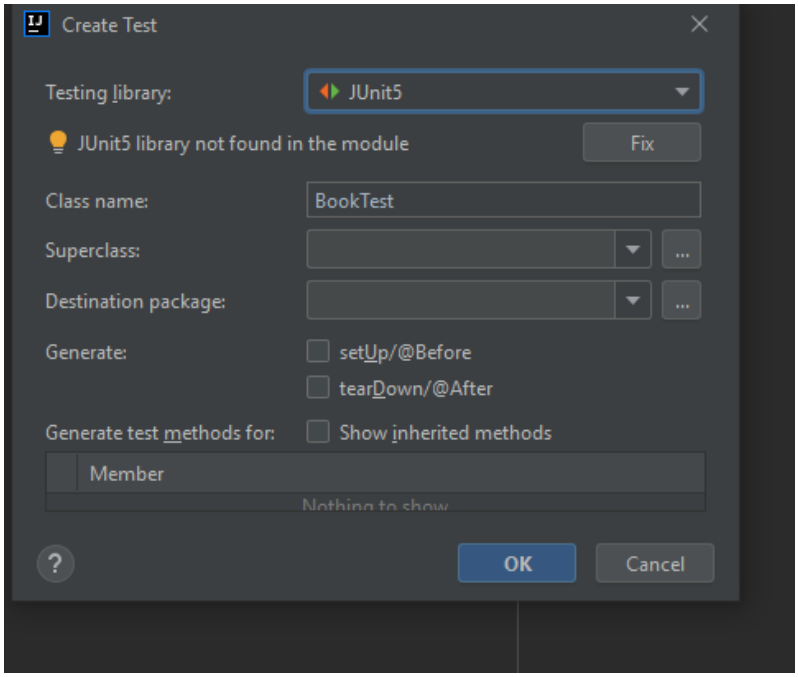
I have used maven for my project build. It is an open source automation tool that can be used for complete build lifecycle.

pom.xml (LiberaryManagementSystem) | Book.java | books.java | booksTest.java | Student.java | students.java | studentsTest.java | Library.java

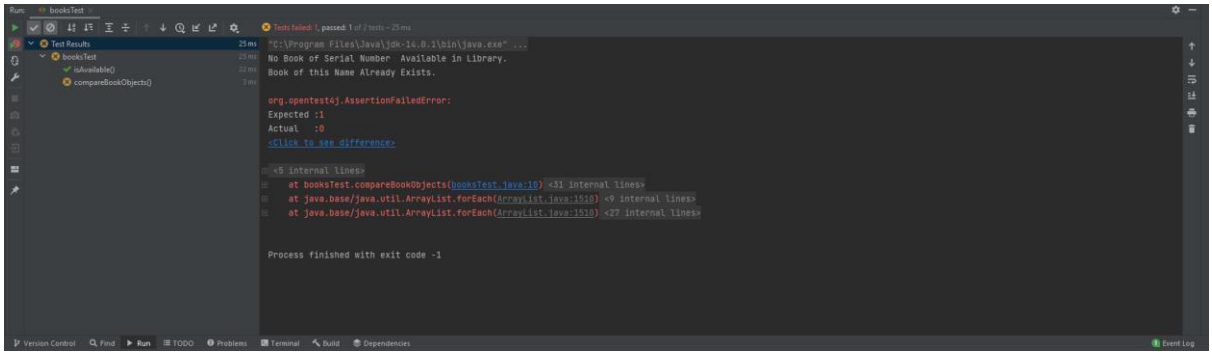
```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>LiberaryManagementSystem</artifactId>
    <version>1.0-SNAPSHOT</version>
<dependencies>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <version>5.8.1</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>RELEASE</version>
        <scope>test</scope>
    </dependency>

</dependencies>
    <properties>
        <maven.compiler.source>14</maven.compiler.source>
        <maven.compiler.target>14</maven.compiler.target>
    </properties>

</project>
```

7. UNIT TESTS

I have created and upload test files on git. I have used Junit5 library in IntelliJ to create test cases for different methods in different classes of my code.


Create Test
Testing library:
JUnit5
JUnit5 library not found in the module
Fix
Class name:
BookTest
Superclass:
Destination package:
Generate:
setUp/@Before
tearDown/@After
Generate test methods for:
Show inherited methods
Member
Nothing to show
OK
Cancel


pom.xml (LiberaryManagementSystem)
Book.java
books.java
booksTest.java
Student.java
students.java
import static org.junit.jupiter.api.Assertions.*;
class booksTest {
books b=new books();
@org.junit.jupiter.api.Test
void compareBookObjects() {
Book b1=new Book(s: 1, n: "Book 1", a: "Author 1", q: 4);
Book b2=new Book(s: 1, n: "Book 1", a: "Author 1", q: 4);
int compare=1;
assertEquals(compare,b.compareBookObjects(b1,b2));
}
@org.junit.jupiter.api.Test
void isAvailable() {
Book b1=new Book(s: 1, n: "Book 1", a: "Author 1", q: 4);
Book b2=new Book(s: 1, n: "Book 1", a: "Author 1", q: 4);
assertEquals(-1,5);
}
}


Run: booksTest
Test Results
booksTest
isAvailable()
compareBookObjects()
No Book of Serial Number Available in Library.
Book of this Name Already Exists.
org.opentest4j.AssertionFailedError:
Expected :1
Actual :0
Click to see difference
at booksTest.compareBookObjects(booksTest.java:10)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1510)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1510)
Process finished with exit code -1

8. CONTINOUS DELIVERY

Git action is used for continuous delivery of the project. It is the automation to speed up the release of new code. It pushes the changes made into code to repository or container registry. The link to git actions for the project is: https://github.com/MuhammadAli-library/LibraryManagamentSystem/blob/main/.github/workflows/maven.yml

9. IDE

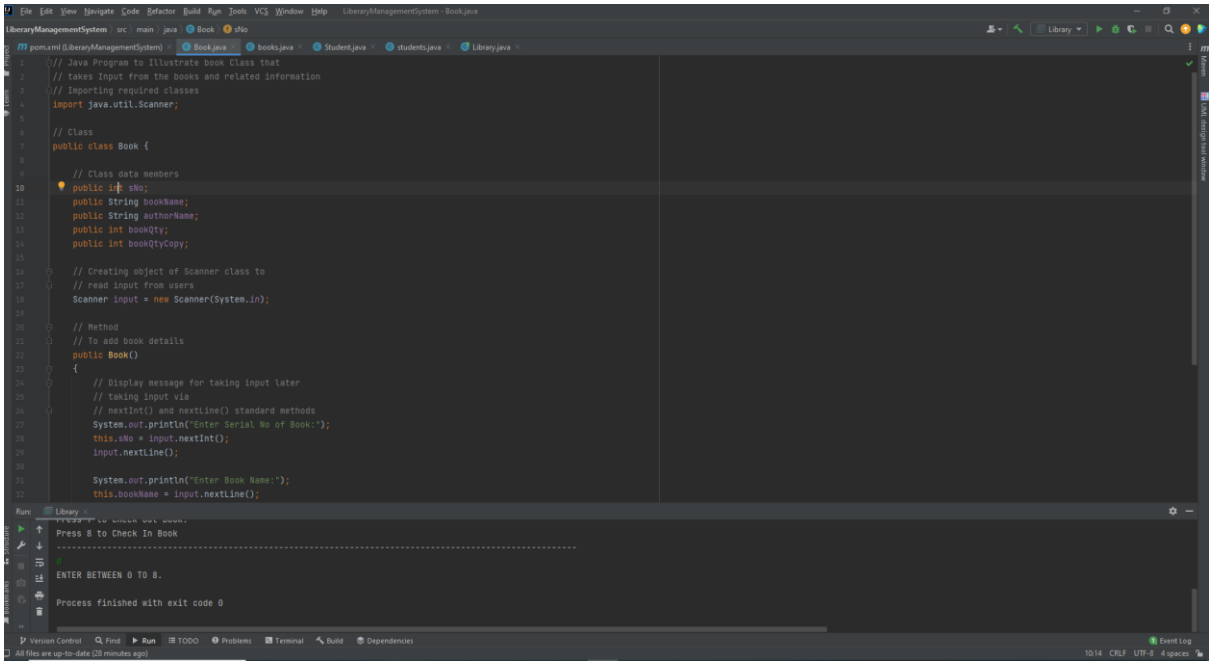
I have used IntelliJ for my project development. It is a very simple and easy IDE to use. I like working in it. I have created my project using maven on IntelliJ. The shortcuts I like in IntelliJ are following:

Shift+F2 navigate between different issues in code

Ctrl+Alt+L Reformat code

Ctrl+/ add a block comment

Alt+1 focus on tools of project



URL for test files: https://github.com/MuhammadAli-library/LibraryManagamentSystem/tree/main/liberaryManagment/src/test/java

10. DSL

The demo DSL code is not related to my project but I have added the demo at the following links: https://github.com/MuhammadAli-library/LibraryManagamentSystem/tree/main/DSL

11. FUNCITONAL PROGRAMMING

I have utilized functional programming in my code which can be seen in different methods, and different points in my code which is uploaded on git.

The link to source code: https://github.com/MuhammadAli-library/LibraryManagamentSystem/tree/main/liberaryManagment/src/main/java