
# Pimpri Chinchwad Education Trust's

# Pimpri Chinchwad College of Engineering

# Department of Computer Engineering

## Mini Project 1 Report

**On**

Library Management System

**Group Members**


| | |
|---|---|
| 1.Adwait Deshmukh | BECOA124 |
| 2.Sangram Deshmukh | BECOA123 |
| 3.Sujeet Jawale | BECOA138 |


**Guide**

Prof. Manjiri Ranjanikar

# ABSTRACT

library Review system is a project which aims in developing a computerized system to maintain all the daily work of a library like collecting reviews, managing rating of dishes etc. This project has many features which are generally not available in normal library management systems like facility of user login. This login feature for the user is provided in order to provide the security to the application. It also has a facility of viewing all the available dishes available in librarys. Through the login, the admin can monitor the whole system by the proficient use of the login through which he can monitor the whole system. It also has the facility of an online view- board where librarian can put up information about the books those are currently available in the library. It has also a facility where a record is maintained of the books those have been issued by the students. When the student issues the book, the book name gets added in the list of the issued books and whenever the student s returns the book, then the books gets removed from the list of issued books. The librarian after logging into his account i.e. admin account can perform various functions such as adding books, removing books and issuing of books.

Overall this project of ours is being developed staff of library to maintain the library in the best way possible and also reduce the human efforts.

# Index

# INTRODUCTION

This chapter gives an overview about the aim, objectives, background and operation environment of the system.

## PROJECT AIMS AND OBJECTIVES:

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

- Online book issue
- Request column for librarian for providing new books
- A separate column for digital library
- Student login page where students can find books issued by him/her and date of return.
- Admin should be able to add librarian and remove librarian. Admin have all the privileges.
- Librarians can add the books, can issue books to students and also can enter returning details at the time of returning into it.

## BACKGROUND OF THE PROJECT:

library Management System is an application which refers to library systems which are generally small or medium in size. It is used by librarian to manage the library using a computerized system where he/she can record various transactions like issue of books, return of books, addition of new books, addition of new students etc.

Books and student maintenance modules are also included in this system which would keep track of the students using the library and also a detailed description about the books a library contains. With this computerized system there will be no loss of book record or member record which generally happens when a non-computerized system is used.

In addition, report module is also included in library Management System. If user's position is admin, the user is able to generate different kinds of reports like lists of students registered, list of books, issue and return reports.

All these modules are able to help librarian to manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

# FUNCTIONAL REQUIREMENTS

## Librarian Login:

**Description:**

Provides login to the librarian for the purpose of the security.

**Functional Requirement:**

This feature used by the user to login into system. They are required to enter user id and password before they are allowed to enter the system. The user id and password will be verified and if invalid id is there user is allowed to not enter the system.

## REGISTER NEW BOOK:

**Description:**

This feature allows to add new book in the library.

**Functional requirements:**

System must be able to verify information.

System must be able to not allow two books having same book.

## VIEW BOOKS:

**Description:**

This feature is found in book maintenance part. we can search book based on book id, book name, publication or by author name.

**Functional requirements**

System must be able to search the database based on select search type

- System must be able to show the book in table view

# NON-FUNCTIONAL REQUIREMENTS

## EFFICIENCY REQUIREMENT

When a library management system will be implemented librarian can easily access library as searching and book transaction will be very faster.

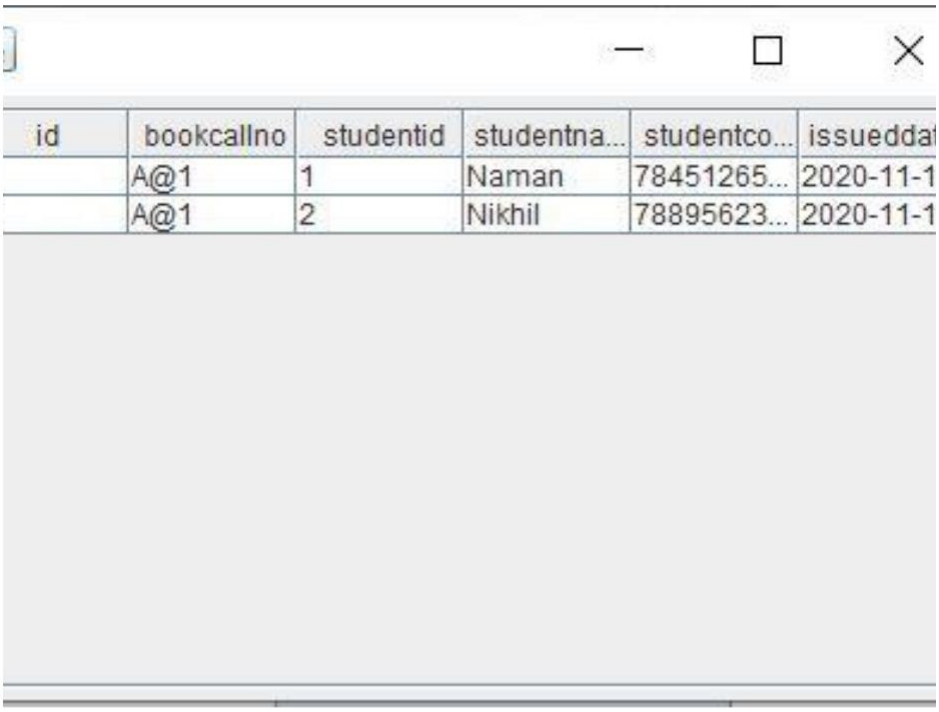
## RELIABILITY REQUIREMENT

The system should accurately perform the list generation, book transaction and database maintenance.

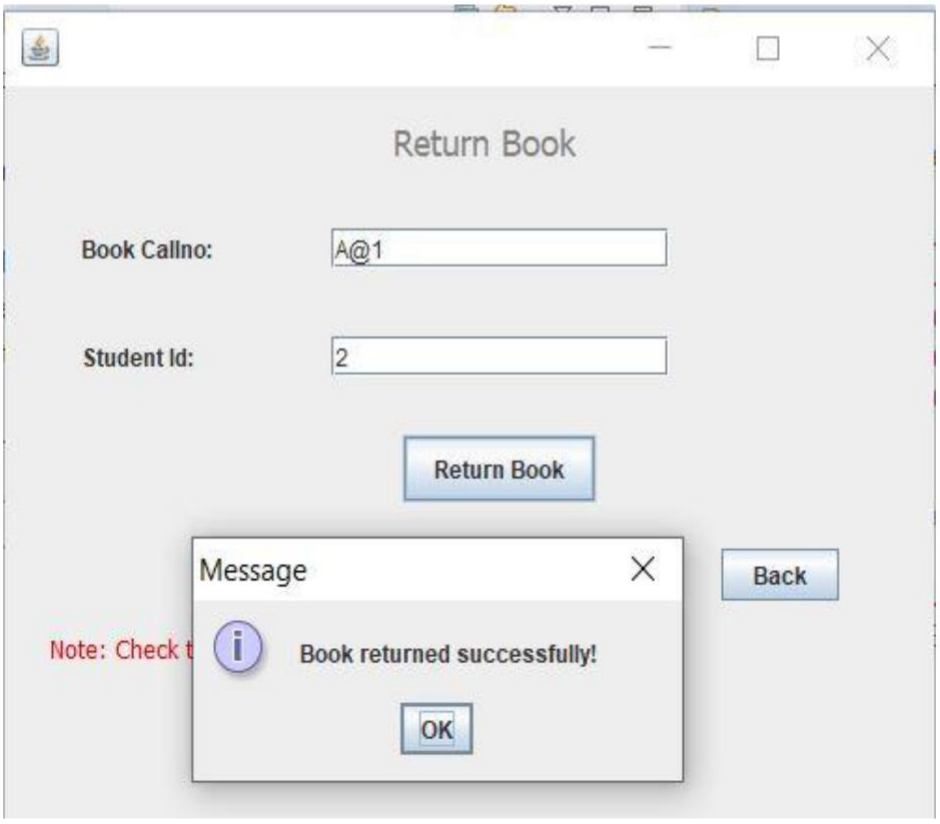
## USABILITY REQUIREMENT

The system is designed for a user-friendly environment so librarian can perform the various tasks easily and in an effective way.

# GUI of the Application Software

| id | bookcallno | studentid | studentna... | studentco... | issueddat |
|---|---|---|---|---|---|
| | A@1 | 1 | Naman | 78451265... | 2020-11-1 |
| | A@1 | 2 | Nikhil | 78895623... | 2020-11-1 |



# Source Code

```
package library;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

public class LibrarianDao {



        public static int save(String name,String password,String email,String address,String city,String

                contact){ int status=0;

                try{

                        Connection con=DB.getConnection();

                        PreparedStatement ps=con.prepareStatement("insert into
librarian(name,password,email,address,city,contact) values(?,?,?,?,?,?)");

                        ps.setString(1,name);

                        ps.setString(2,password);

                        ps.setString(3,email);

                        ps.setString(4,address);

                        ps.setString(5,city);

                        ps.setString(6,contact);

                        status=ps.executeUpdate();

                        con.close();

                }catch(Exception e){System.out.println(e);}

                return status;

        }

        public static int delete(int id){

                int status=0;

                try{

                        Connection con=DB.getConnection();

                        PreparedStatement ps=con.prepareStatement("delete from librarian where id=?");
```

```
                ps.setInt(1,id);
                status=ps.executeUpdate();
                con.close();
            }catch(Exception e){System.out.println(e);}
            return status;
        }

        public static boolean validate(String name,String
            password){ boolean status=false;
            try{
                Connection con=DB.getConnection();
                PreparedStatement ps=con.prepareStatement("select * from librarian where name=? and
password=?");
                ps.setString(1,name);
                ps.setString(2,password);
                ResultSet rs=ps.executeQuery();
                status=rs.next();
                con.close();
            }catch(Exception e){System.out.println(e);}
            return status;
        }

}
```

# Test Plan Description

- **Introduction:**

  The application is intended for the library of the particular institute or the organization. It manages the different tasks of the librarian such as adding the books, issuing the books, returning the issued books, etc.

.

- **Intended Audience**:

  This test plan is made for system testing of this application. The test plan will be referred by,

  - Development manager and development team
  - Librarian

- **Intended Users** :

  This application will be used by the librarian of the particular library for the efficient and the effective management of the library.

- **Test Scope**:

  System testing shall cover,

  - User interface testing
  - Functionality testing

- **Out of Scope:**

  The following types of testing are out of scope for this application

  - Security testing
  - Performance, volume, and stress testing
  - Usability testing

- **Test Objectives:**

  - Targeted number of defects—50
  - Targeted number of test scenario—7
  - Targeted number of test cases—10
  - Number of iterations required—5
  - Test scenario writing per day—10

- Test case writing per day—50
- Test case execution per day—100
- Coverage of requirements P1 = 100%
  P2 = 50%
  P3 = 10%

- **Assumptions of Testing :**

  - Users understand English and they are capable of using a computer
  - Users understand only integers (they are not expected to use decimal)
  - Application will be working on Windows 2000 professional. Other operating systems, if any, are compatible.
  - Application is tested on Pentium P3, 64MB RAM, 250 MHz processor. Other configurations used, if any, are compatible.
  - Testing considers right-handed user

- **Risk Analysis:**

  - Application will not be able to handle non-integers
  - Compatibility on other operating system and machine configuration can be a problem.
  - Application may not work for left-handed users
  - User may not understand English

- **Workflow**:

The project manager communicates the availability of new build along with the records of reviews and unit testing to test manager. The application is installed on the machine as defined in prerequisites (P3, 64MB RAM, 250 MHz professor). Test manager checks the entry criteria for testing. If the application passes the entry criteria, it is taken for iteration testing.

Test cases are stored in configuration library. Test cases are executed as per the sequence mentioned and results are added in the test case as actual results. Difference between expected result and actual result is taken for evaluation. All differences between expected results and actual results are reproduced. Video clips are created for the defects. Defects are added in defect management tool

.

- **Test Design**:

The application will be tested as per steps. Verification activities will be covered in quality plan. This is a system test plan and other levels of testing are covered by separate plans at each level.

- **Roles and Responsibilities**:

  Roles and responsibilities of stakeholders are as defined in Table below

Roles and responsibilities

| Responsibilities/Role | Test manager | Senior tester | Tester | Customer |
|---|---|---|---|---|
| test planning | X | | | |
| test scenario writing | | X | | |
| test cases writing | | X | X | |
| Test data definition | | | X | |
| test execution | | | X | |
| Query resolution | | | | X |

- **Test Data Management**
  - Test plan, test scenario and test cases will be under configuration management control and kept on centralized server
  - Test reports will be stored on the same server
  - Backup will be taken on tape, once per week

- **Test Environment** :

  Test environment will be made of individual machines.
  - Windows 2000 professional
  - 64MB RAM
  - 256 MHz processor, Pentium P3

- **Test Entry Criteria:**
  - All review comments are closed
  - All unit testing defects are closed
  - Application can be installed and launched

- **Test Exit Criteria**:

  Test cases are completed and all defects found in testing are closed, retested and found to be closed.

- **Test Suspension Criteria** :

Tests will be suspended in the following circumstances,

  - Test cases written do not find sufficient number of defects
  - Test case writing and test data definition are not completed
  - Application cannot be installed
  - Defects found in $1^{st}/2^{nd}/3^{rd}$ iteration are more than 500

- **Test Resumption Criteria:**
  - All defects are reviewed and corrective/preventive actions are taken
  - All review comments are closed
  - All unit testing defects are fixed, retested and found to be closed
  - Application can be installed

- **Tester's Training**:

  The following training is planned for testers.

  - Testers will be trained on basics of triangle geometry
  - Tester's training for writing test plan, test scenarios, test cases, execution of test cases and defect logging is already done.

# Test scenario

✔ Common for All Test Scenarios (TS00—Priority P1)

An application can be launched by clicking an icon on the desktop.

✔ Scenario 1 (TS01)

To check Login Credentials are accepted successfully or not.

✔ Scenario 2 (TS02)

To check whether components are displayed and are enabled.

✔ Scenario 3 (TS03)

To check whether components are functional.

## Test Case 1

- Test Precondition: *Application must be installed Computer must be working.*
- Test Sequence: Not applicable
- Test Scenario Traceability: TS00
- Test Case Name and Number: SMK/MOD1/0001
- Type of Testing : Smoke testing
- Objectives: To check whether application can be launched or not
- Valid/invalid Conditions: Valid condition
- Expected Results: Application is launched. (Screen is seen with cursor in text box next to label ‘a’)

## Test Case 2

- Test Precondition: *Application is launched*
- Test Sequence: SMK/MOD1/000
- Test Scenario Traceability : TS00
- Test Case Name and Number: UI/MOD1/0002
- Type of Testing : User Interface testing
- Objectives: To check all components are displayed or not
- Valid/Invalid Conditions: Valid condition
- Test Data: Not applicable
- Expected Results: Application opens with two buttons

## Test Case 3

- Test Precondition: *Application is launched*
- Test Sequence: SMK/MOD1/000
- Test Scenario Traceability: TS01
- Test Case Name and Number: UI/MOD1/0003
- Type of Testing: User Interface testing
- Objectives: To check whether Login Button is Enabled

- Expected result: Software should display Login Button and should be clickable
- Valid/invalid Conditions: Valid condition

## Test Case 4

- Test Precondition: *Application is launched*
- Test Scenario Traceability: TS01
- Test Case Name and Number: FUN/MOD1/0001
- Type of Testing: Functionality testing
- Objectives: To check whether correct login credentials are entered or not.

- Expected result: Software should display main window if credentials are right otherwise display error message box.
- Valid/invalid Conditions: Valid condition

## Test Case 5

- Test Precondition: *Application is launched*
- Test Sequence SMK/MOD1/000
- Test Scenario Traceability TS02
- Test Case Name and Number UI/MOD1/0004
- Type of Testing : UI testing
- Objectives: To check all components are displayed on add book
- Valid/Invalid Conditions Valid condition
- Expected Results: Success message on successful addition of books.

## Test Case 6

- Test Precondition: *Application is launched*
- Test Scenario Traceability TS03
- Test Case Name and Number FUN/MOD1/0003
- Type of Testing : Functionality testing
- Objectives: To submit the issue request on clicking ‘Add Book’ button
- Valid/Invalid Conditions Valid condition
- Expected Results: Success message on successful Addition of books.

## Test Case 7

- Test Precondition: *Application is launched*
- Test Sequence SMK/MOD1/000
- Test Scenario Traceability TS02
- Test Case Name and Number UI/MOD1/0005
- Type of Testing : UI testing
- Objectives: To check all components are displayed on issue book screen.
- Valid/Invalid Conditions Valid condition
- Expected Results: All components are displayed or not.

## Test Case 8

- Test Precondition: *Application is launched*
- Test Scenario Traceability TS03
- Test Case Name and Number FUN/MOD1/0004
- Type of Testing : Functionality testing
- Objectives: To check functionality of issue book button.
- Valid/Invalid Conditions Valid condition
- Expected Results: success message after issued books is pressed.

## Test Case 9

- Test Precondition: *Application is launched*
- Test Sequence SMK/MOD1/000
- Test Scenario Traceability TS02
- Test Case Name and Number UI/MOD1/0006
- Type of Testing : UI testing
- Objectives: To check all components are displayed on Return book screen.
- Valid/Invalid Conditions Valid condition
- Expected Results: All components are displayed or not.

## Test Case 10

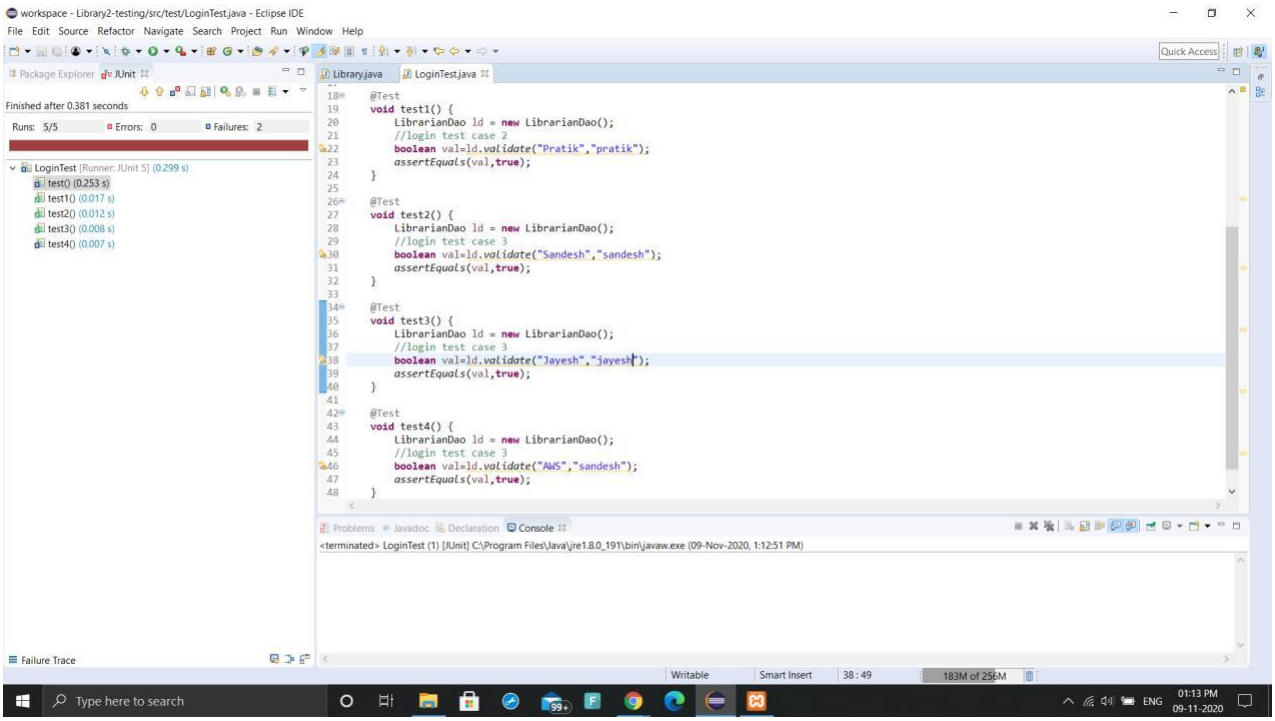
- Test Precondition: *Application is launched*
- Test Scenario Traceability TS03
- Test Case Name and Number FUN/MOD1/0005
- Type of Testing : Functionality testing
- Objectives: To check functionality of return book button.
- Valid/Invalid Conditions Valid condition
- Expected Results: success message after return books is pressed.

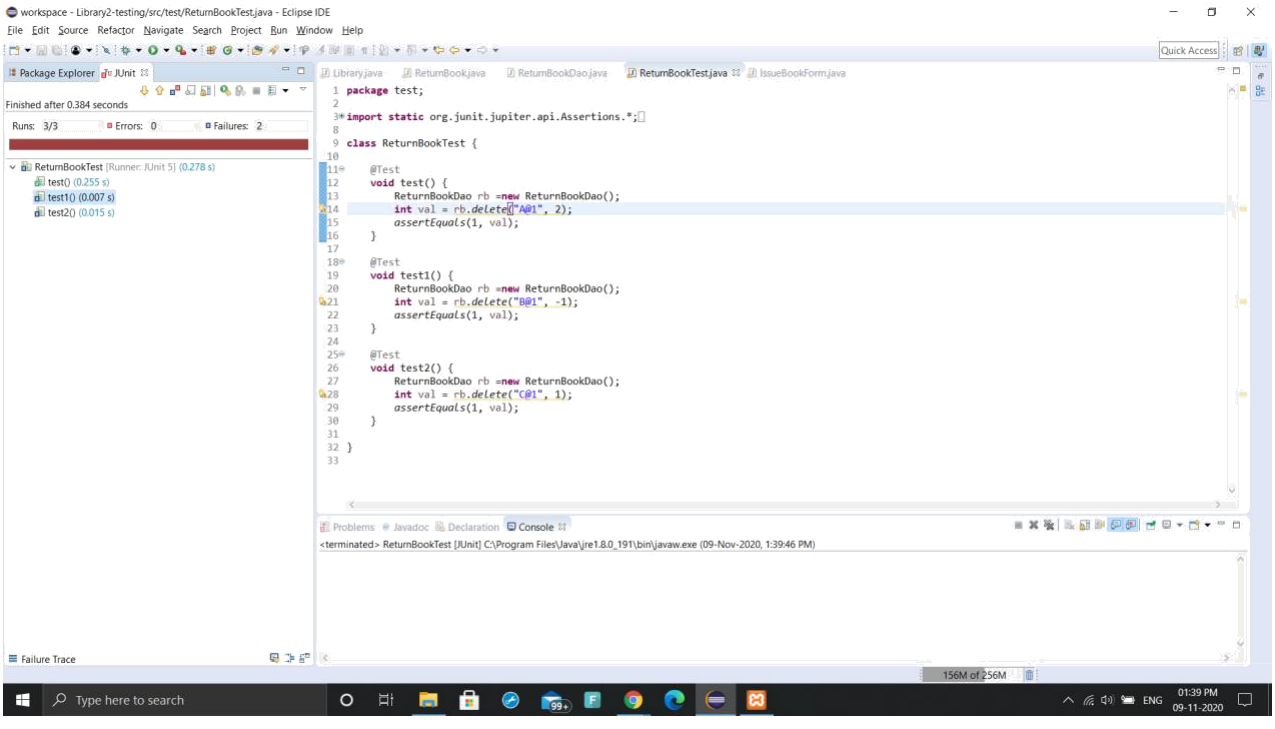
# JUNIT TESTCASE

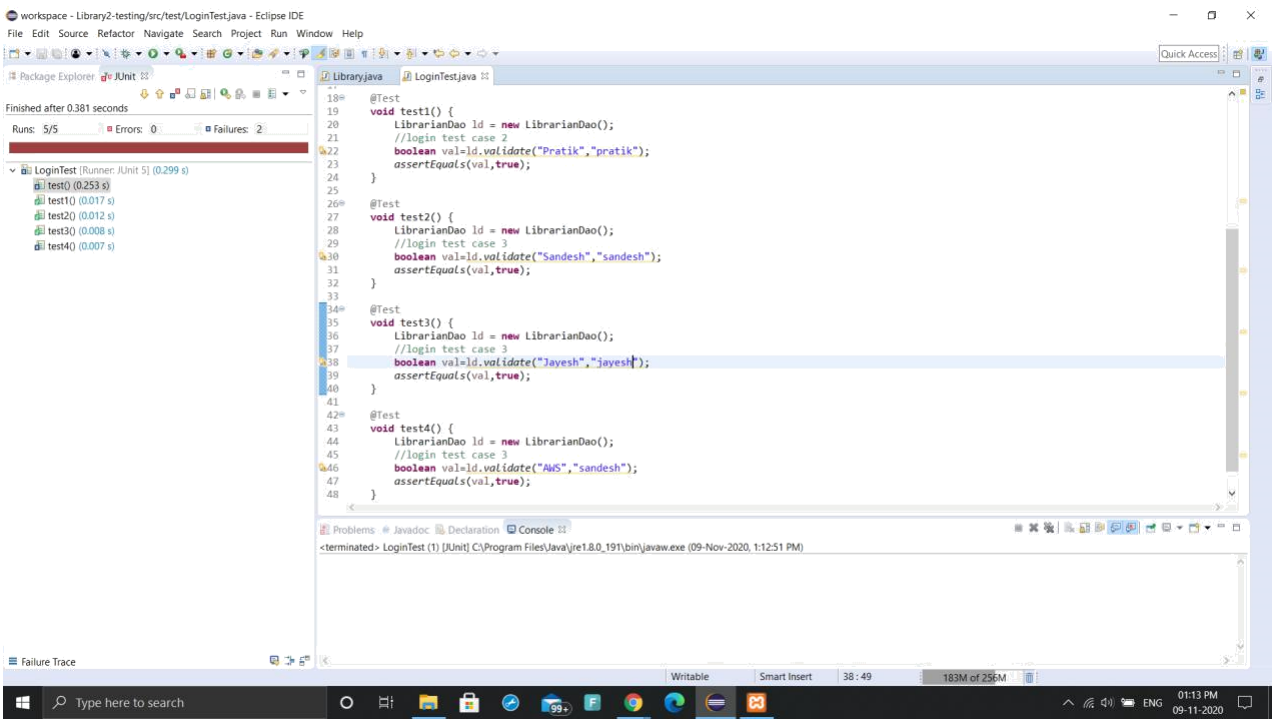
```
package test;
import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

import library.LibrarianDao;

class LoginTest {

	@Test
	void test() {
		LibrarianDao ld = new LibrarianDao();
		//login test case 1
		boolean val=ld.validate("Aayush","123");
		assertEquals(true,val);
	}

	@Test
	void test1() {
		LibrarianDao ld = new LibrarianDao();
		//login test case 2
		boolean val=ld.validate("Pratik","pratik");
		assertEquals(true,val);
	}

	@Test
	void test2() {
		LibrarianDao ld = new LibrarianDao();
```

```
		//login test case 3
		boolean val=ld.validate("Sandesh","sandesh");
		assertEquals(true,val);
	}

	@Test
	void test3() {
		LibrarianDao ld = new LibrarianDao();
		//login test case 3
		boolean val=ld.validate("Jayesh","jayesh");
		assertEquals(true,val);
	}

	@Test
	void test4() {
		LibrarianDao ld = new LibrarianDao();
		//login test case 3
		boolean val=ld.validate("AWS","sandesh");
		assertEquals(true,val);
	}
}
```

# Output Screenshot




workspace - Library2-testing/src/test/ReturnBookTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
JUnit
Finished after 0.384 seconds
Runs: 3/3
Errors: 0
Failures: 2
ReturnBookTest [Runner: JUnit 5] (0.278 s)
test() (0.255 s)
test1() (0.007 s)
test2() (0.015 s)
Failure Trace
Library.java
ReturnBook.java
ReturnBookDao.java
ReturnBookTest.java
IssueBookForm.java
1 package test;
2
3 import static org.junit.jupiter.api.Assertions.*;
8
9 class ReturnBookTest {
10
11 @Test
12 void test() {
13 ReturnBookDao rb =new ReturnBookDao();
14 int val = rb.delete("A@1", 2);
15 assertEquals(1, val);
16 }
17
18 @Test
19 void test1() {
20 ReturnBookDao rb =new ReturnBookDao();
21 int val = rb.delete("B@1", -1);
22 assertEquals(1, val);
23 }
24
25 @Test
26 void test2() {
27 ReturnBookDao rb =new ReturnBookDao();
28 int val = rb.delete("C@1", 1);
29 assertEquals(1, val);
30 }
31
32 }
33
Problems
Javadoc
Declaration
Console
<terminated> ReturnBookTest [JUnit] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (09-Nov-2020, 1:39:46 PM)
156M of 256M
Type here to search
ENG
01:39 PM
09-11-2020


workspace - Library2-testing/src/test/LoginTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer JUnit
Finished after 0.381 seconds
Runs: 5/5
Errors: 0
Failures: 2
LoginTest [Runner: JUnit 5] (0.299 s)
test() (0.253 s)
test1() (0.017 s)
test2() (0.012 s)
test3() (0.008 s)
test4() (0.007 s)
Failure Trace
Library.java
LoginTest.java
18 @Test
19 void test1() {
20 LibrarianDao ld = new LibrarianDao();
21 //login test case 2
22 boolean val=ld.validate("Pratik","pratik");
23 assertEquals(val,true);
24 }
25
26 @Test
27 void test2() {
28 LibrarianDao ld = new LibrarianDao();
29 //login test case 3
30 boolean val=ld.validate("Sandesh","sandesh");
31 assertEquals(val,true);
32 }
33
34 @Test
35 void test3() {
36 LibrarianDao ld = new LibrarianDao();
37 //login test case 3
38 boolean val=ld.validate("Javesh","jayesh");
39 assertEquals(val,true);
40 }
41
42 @Test
43 void test4() {
44 LibrarianDao ld = new LibrarianDao();
45 //login test case 3
46 boolean val=ld.validate("AWS","sandesh");
47 assertEquals(val,true);
48 }
Problems Javadoc Declaration Console
<terminated> LoginTest (1) [JUnit] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (09-Nov-2020, 1:12:51 PM)
Writable
Smart Insert
38 : 49
183M of 256M
Type here to search
ENG
01:13 PM
09-11-2020

# Conclusion and Future Scope

This application provides a computerized version of library management system which will benefit the students as well as the staff of the library.

It makes entire process online where student can search books, staff can generate reports and do book transactions. It also has a facility for student login where student can login and can see status of books issued as well request for book or give some suggestions. It has a facility of teacher's login where teachers can add lectures notes and also give necessary suggestion to library and also add info about workshops or events happening in our college or nearby college in the online notice board.

There is a future scope of this facility that many more features such as online lectures video tutorials can be added by teachers as well as online assignments submission facility, a feature of group chat where students can discuss various issues of engineering can be added to this project thus making it more interactive more user friendly and project which fulfills each user's need in the best way possible.