A Project Report Submitted to

Department of Computer Science

S.I.E.S College of Arts, Science and Commerce (Autonomous)
Sion (W), Mumbai– 400 022.

"Library Management System"

For the Partial Fulfillment for The Degree Of

Bachelor of Science (Computer Science)

2021-2022

Head of Department	**Project Guide**
Prof. Manoj Singh	Prof. Manoj Singh

Submitted by

Meenakshi Sambari

Aanam Shaikh

CONTENTS

Acknowledgement .....

Introduction .....

Description of the System .....

Objective .....

Stakeholders .....

Modules and Brief Description of every module .....

Limitations of the existing system .....

Proposed system .....

Unique Features .....

Purpose and Scope .....

Requirements Specification .....

GANTT Chart .....

Survey

System Analysis .....

Event Table .....

Function Point Analysis .....

Lines of Code .....

System Design .....

ER Diagram .....

Use Case Diagram .....

Activity Diagram .....

Code .....

Pert &CPM .....

Code and Screen-caps .....

References .....

Acknowledgement

The success of this project would not have been possible without the help and guidance of teachers. However, we would like to extend our sincere thanks to all of them.

We respect and thank our Head of Department **Mr. Manoj Singh** for giving us an opportunity and support to complete this project.

We sincerely thank to all our teachers for their timely and prestigious guidance.

INTRODUCTION

A library management is a project that manages and stores books information electronically according to members needs. The system helps both members and librarian to keep a constant track of all the books available in the library. It allows both the admin and the member to search for the desired book. It becomes necessary to keep a continuous check on the books issued and returned. This task if carried out manually will be tedious and includes chances of mistakes. These errors are avoided by allowing the system to keep track of information such as issue date, last date to return the book and even fine information and thus there is no need to keep manual track of this information which thereby avoids chances of mistakes.
Thus this system reduces manual work to a great extent allows smooth flow of library activities by removing chances of errors in the details.

Description of the System:

- A Library is a collection of different types of books.
- It provides books for large number of readers and to maintain a record is difficult.
- Here comes the need for a software for managing library.
- Thus a Library Management System that provide the systematic handling records of books and users.

OBJECTIVE

- The implementation of Library management System starts with entering and updating master records like book detail and library information.
- Any further transaction like book issue,book return will automatically update the current books.
- It will track how many books are available in the library and books issued to the members.

STAKEHOLDERS

- Admin
- Staff Members REPONSIBLE FOR RECORDING THE BOOKS ISSUED AND
- User Members

MODULES

Modules of Admin

- Admin can add/update/delete books
- Admin can keep the track of members returning the book.
- Admin can view the book details.
- Admin can view member details.
- Admin can add/update/view/edit member details.
- Admin can add/update/view/delete staff details.
- Admin can view staff details.
- Admin can access everything.

Modules of Staff Members

- Staff members can add/update/delete books
- Staff members can keep the track of members returning the book.
- Staff members can view the book details.
- Staff members can view member details.
- Staff members can add/update/view/edit member details.
- Staff members cannot view staff details.
- Staff members cannot access everything those right are only to the admin.

Modules of User Members

- Users can check or view available books of the library
- Users can check or view his/her issued books.
- Users can check or view his/her returned books status.

LIMITATIONS OF LIBRARY MANAGEMENT SYSTEM

- The data stored is vulnerable to cyber attacks .
- To maintain and manage it is both costly, expensive and complex to some extent.
- Management and operating it can be complicated.
- Requires high speed connectivity.
- The previous system lacked the about section. The revised version contains a well versed about section.
- It is a step by step procedure i.e., all steps comes after one another. So, every steps is need to be first completed then only we can move the other steps, we need to wait.
- Some times very high data interaction so server is down.
- Very high security needed to this system.

PROPOSED SYSTEM OF LIBRARY MANAGEMENT SYSTEM

The proposed Online Library Management System project will help the members and librarian to maintain the details of the library. It will assist the librarian before the shortage of books while they can know the details of the number of currently available in the library according to

the author by accessing the system. A member can view the details of the book issued by them. At the time of issue of a book, the member will get assisted by the system about different authors of a similar book so that they can get the best available book from the library.

UNIQUE FEATURES

- It works in all the major operating systems including Windows, Mac, and Linux.
- You can generate and print your own barcodes.
- Digital cataloguing and indexing of books. Manage circulation and acquisitions system for library asset management.
- Manage reading details of students, employees, and keep an online record of college books details.
- A user-friendly search interface based on factors like categories of books, students, etc.
- Maintains a complete information database about the authors, publishers, etc.

PURPOSE AND SCOPE

- The purpose of this case study is provide a friendly environment to maintain the details of books and library members.
- The purpose of this case study is to maintain easy circulations using computers.
- Improved customer service and
- Better access to accurate information.

HARDWARE REQUIREMENTS:

Processor – atleast 2.0 GHZ

RAM – atleast 2GB

SYSTEM REQUIREMENTS:

Operating system – Windows

Language – Java

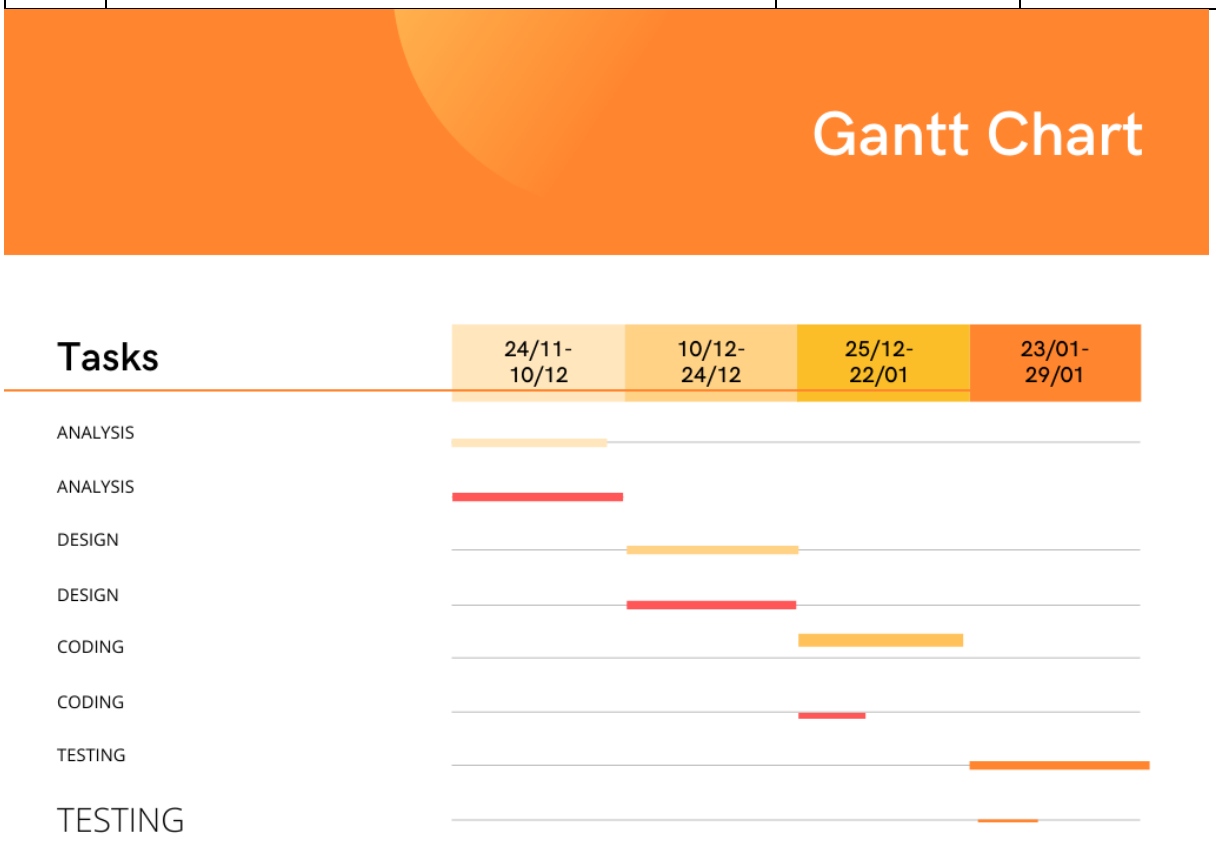
Concept – swing and AWT

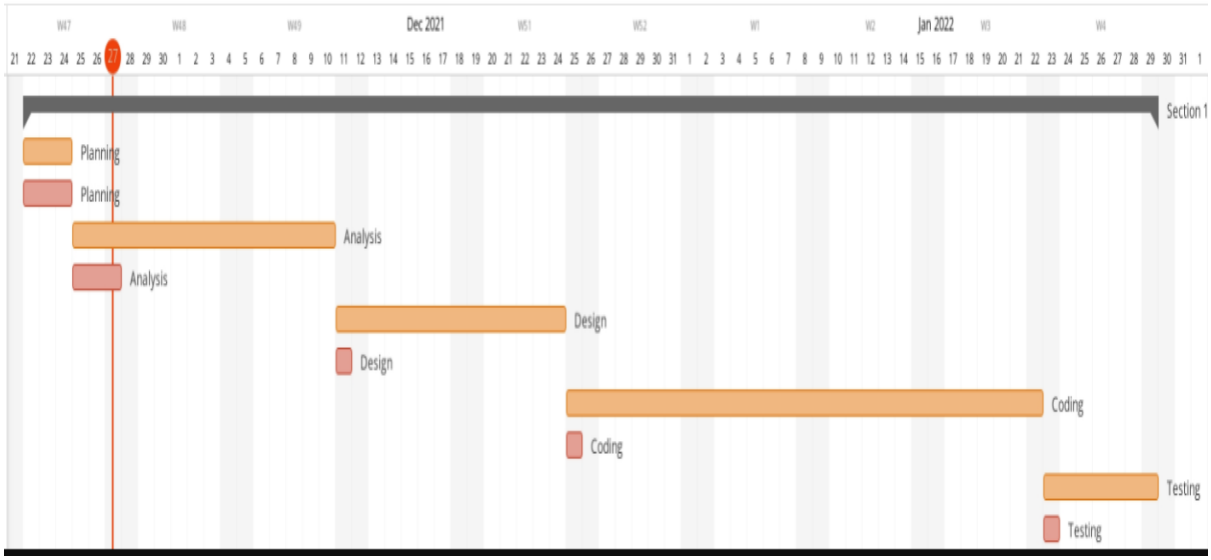
SOFTWARE MODEL

- NetBeans IDE 8.0.1
  - Java SE Development Kit 8u301
- Xampp [MySQL sever]
  - Mysql-connector-java5.1.40

GANTT CHART

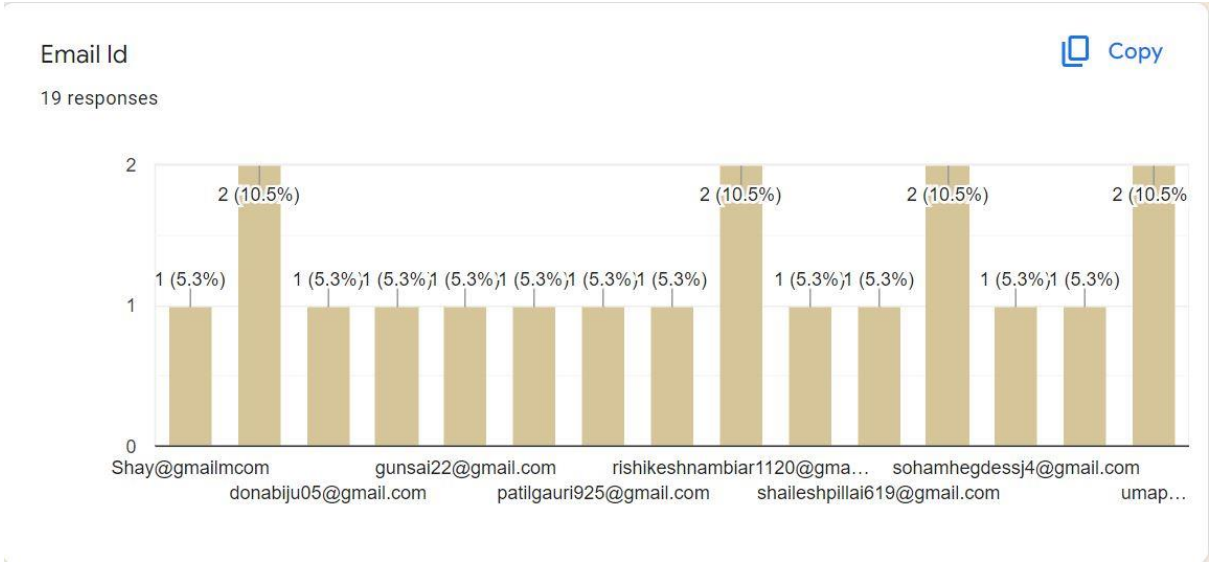
No.	Phase	Start Date	End Date
1.	Planning	22 / Nov	24 / Nov
2.	Analysis	25 / Nov	10 / Dec
3.	Design	11 / Dec	25 / Dec
4.	Coding	25 / Dec	22/ Jan
5.	Testing/Documentation	23 / Jan	29 / Jan





SURVEY QUESTIONS

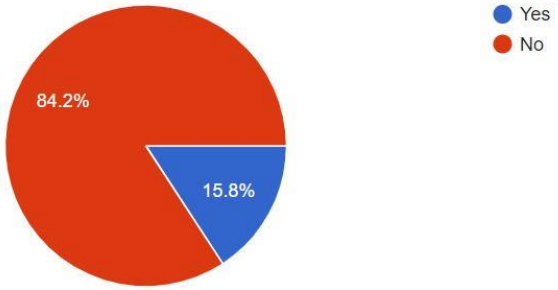
RESPONSES:



Are you facing problem in any section of our software

Copy

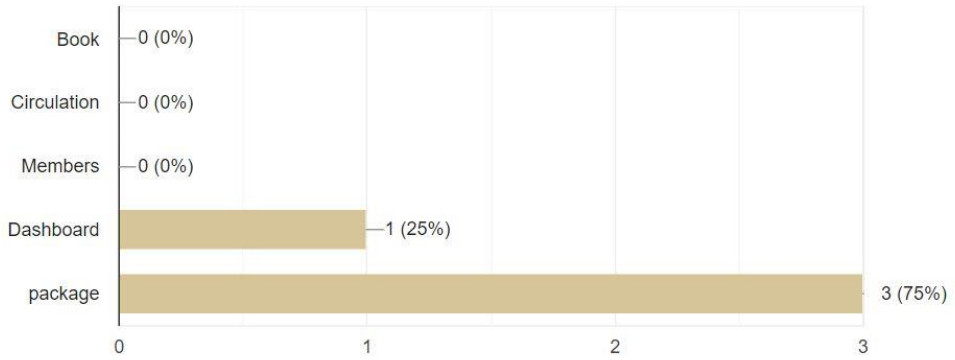
19 responses



if yes , please specify the sections

Copy

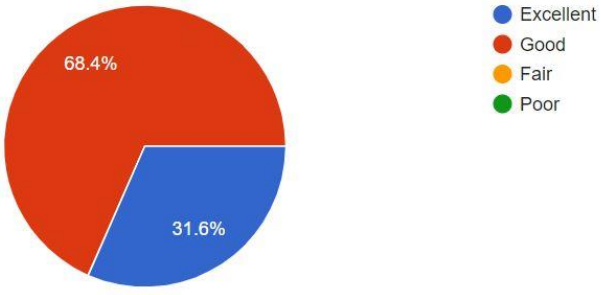
4 responses



How was your experience while Adding a Book or Member from this software?

Copy

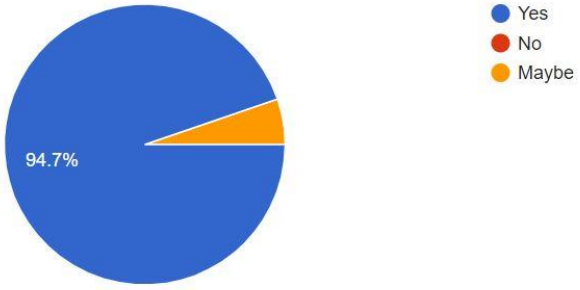
19 responses



Did you like the interface of this software?

Copy

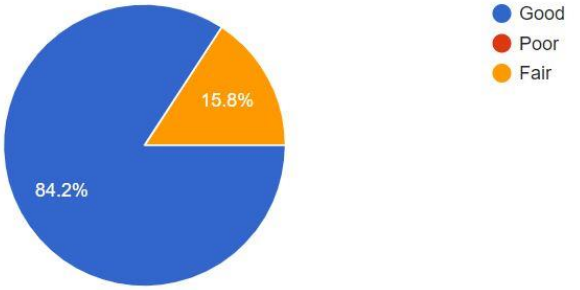
19 responses



How was your Experience with the Circulation section?

Copy

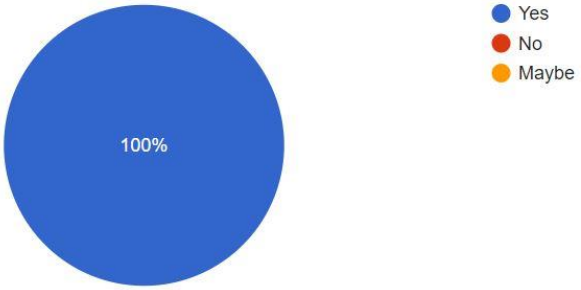
19 responses



Did you like the Latest Book Section on Dashboard

Copy

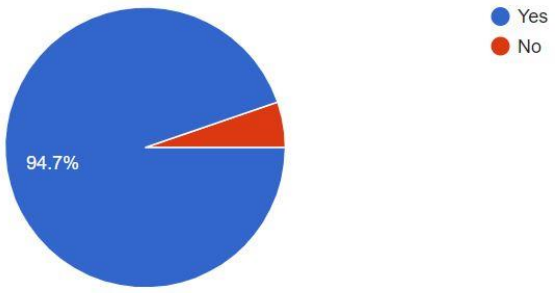
7 responses



Are you satisfied with overall performance of the software?

Copy

19 responses



If not , please recommend us the changes

3 responses

Nil

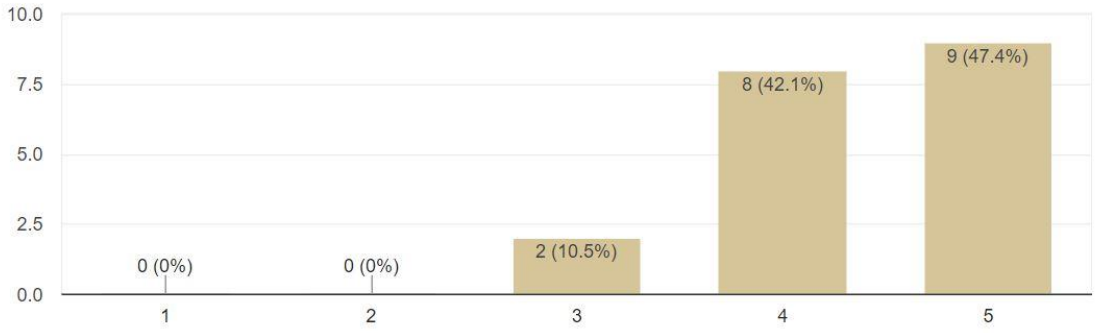
You can use firbase

Try to make it available online

Please rate our software on the scale of 1 to 5

Copy

19 responses



If you would like to leave some suggestions/compliments/issues/reviews, please do so!!

5 responses

Best software in the world

You should add an recommendation system

Nil

Customer module can be improved

Nicely done!

SYSTEM ANALYSIS

EVENT TABLE

ADMIN EVENT TABLE

Sr no.	Event	Trigger	Source	Destination	Activity
1	Online registration	Admin registering their credentials	admin	admin's database	A admin can register in the software for further using the software
2	login	Admin can login with their id	admin	Admin's database	Admin can login in software using their id
3	Add book	Admin can add the book in softwae	admin	Admin's databse	Admin can add the book in the software with the details
4	Update books	Admin can update books	Admin	Admin's database	Admin can update the books in the software
5	Delete books	Admin can delete books	admin	Admin's database	Admin can delete the books from the software

6	View books	Admin can view books	admin	Admin's database	Admin can view book details from the software
7	Add staff member	Admin can add staff member	admin	Admin's database	Admin can add staff member in the software
8	Update staff member	Admin can update staff member	admin	Admin's database	Admin can update staff member in the software
9	Delete staff member	Admin can delete staff member	admin	Admin's database	Admin can delete staff member in the software
10	View staff member	Admin can view staff member	admin	Admin's database	Admin can view staff member in the software
11	Add user member	Admin can add user member	admin	Admin's database	Admin can add user member in the software
12	update user member	Admin can update user member	admin	Admin's database	Admin can update user member in the software

13	delete user member	Admin can delete user member	admin	Admin's database	Admin can delete user member in the software
14	View user member	Admin can view user member	admin	Admin's database	Admin can view user member in the software
15	View member card	Admin can view member card	admin	Admin's database	Admin can member card in the software
16	View defaulter list	Admin can view issue date	admin	Admin's database	Admin can view defaulter list in the software
17	View issue date	Admin can view issue date	admin	Admin's database	Admin can view issue date of books in the software
18	View due date	Admin can view due date	admin	Admin's database	Admin can view due date of booksin the software
19	View live count of no of books	admin can view live count of no of books	admin	admins database	admin can view live count of no of books in

					the software
20	View live count of no of issued boo ks	admin can view live count of no of issued books	admi n	admins database	admin can view live count of no of issued books in the software
21	View live count of defaulter list	admin can view live count of defaulter list	admi n	admins database	admin can view live count of defaulter list in the software
22	View live statistics of books	Admin can view live statistics of books	admi n	Admin's database	Admin can view live statistics of books in dashboard from the software
23	check returned books status	Admin ca n check returned books status	admi n	admin's database	Admin can check returned books statusin the software

24	View returned books status	admin can view returned books status	admi n	admins database	admin can view returned books statusin the software
25	View live count of no students	admin can view live count of no students	admi n	admins database	admin can view live count of no students in the software
26	View latest books	Admin can view latest books	admi n	Admin's database	Admin can view latest book in dashboard from the software

Staff Members

27	Online registration	Staff registering their credentials	staff	staff's database	A staff can register in the software for further using the software
28	login	staff can login with their id	staff	staff's database	staff can login in software

					using their id
29	Add book	staff can add the book in software	staff	staff's database	Staff can add the book in the software with the details
30	update book	staff can update the book in software	staff	staff's database	Staff can update the book in the software with the details
31	delete book	staff can delete the book in software	staff	staff's database	Staff can delete the book in the software with the details
32	View book	staff can view the book in software	staff	staff's database	Staff can view the book in the software with the details
33	View defaulter list	staff can view issue date	staff	staff's database	staff can view defaulter list in the software

34	View issue date	staff can view issue date	staff	staff's database	staff can view issue date in the software
35	View due date	staff can due issue date	staff	staff's database	staff can view due date in the software
36	Add user member	staff can add user member	staff	staff's database	staff can add user member in the software
37	update user member	staff can update user member	staff	staff's database	staff can update user member in the software
38	delete user member	staff can delete user member	staff	staff's database	staff can delete user member in the software
39	view user member	staff can view user member	staff	staff's database	staff can view user member in the software
40	View member card	staff can View member card	staff	staff's database	staff can View member card

					in the software
41	View latest books	staff can view latest books	staff	staff's database	Staff can view latest book in dashboard from the software
42	View live statistics of books	staff can view live statistics of books	staff	staff's database	Staff can view live statistics of books in dashboard from the software
43	check returned books status	staff can check returned books status	staff	staff's database	Staff can check returned books status in the software
44	View returned books status	Staff can view returned books status	user	staff's database	Staff can view returned books status in the software
45	View live count of no of books	Staff can view live count of no of books	staff	admins database	Staff can view live count of no of books in the software

46	View live count of no of issued books	Staff can view live count of no of issued books	staff	Staff database	Staff can view live count of no of issued books in the software
47	View live count of defaulter list	Staff can view live count of defaulter list	staff	Staff database	Staff can view live count of defaulter list in the software
48	View live count of no students	Staff can view live count of no students	staff	Staff database	Staff can view live count of no students in the software
49	View latest books	Staff can view latest books	staff	staff's database	Staff can view latest book in dashboard from the software

User Member

50	View books	User can view books	user	User's database	Users can view books in the software

51	check books	check can view books	user	User's database	Users can check books in the software
52	check issued books	check can check issued books	user	User's database	Users can check issued books in the software
53	View issued books	User can view issued books	user	User's database	Users can view issued books in the software
54	check returned books status	user can check returned books status	user	User's database	Users can check returned books statusin the software
55	View returned books status	user can view returned books status	user	User's database	Users can view returned books statusin the software
56	View live count of no students	User can view live count of no students	user	User database	User can view live count of no students in the software

57	View latest books	User can view latest books	user	user's database	usercan view latest book in dashboard from the software
58	View live count of no of books	User can view live count of no of books	user	User database	User can view live count of no of books in the software
59	View live count of no of issued books	User can view live count of no of issued books	user	User database	User can view live count of no of issued books in the software
60	View live count of defaulter list	User can view live count of defaulter list	user	User database	User can view live count of defaulter list in the software
61	View live statistics of books	User can view live statistics of books	user	User database	User can view live statistics of books in dashboard from the software

FUNCTION POINT ANALYSIS

FPA is a technique used to measure software requirements based on the different functions that the requirement can be split into. Each function is assigned with some points based on the FPA rules and then these points are summarized using the FPA formula.

FUNCTION TYPE	SIMPLE	COMPLEX	AVERAGE
Inputs	**2**	**4**	**6**
Output	**3**	**5**	**7**
Inquiries	**3**	**4**	**6**
Files	**7**	**10**	**15**
Interface	**5**	**7**	**10**

Inputs:

6 simple x 2 = 12

4 average x 4 = 16

3 complex x 6= 18

Inputs = 46

Outputs:

6 simple x 3 = 18

6 average x 5 =30

2 complex x 7=14

Outputs=62

Files:

3 average x 10= 30

Inquires:

2 average x 4 = 8

Interface:

2 average x7 = 14

Unadjusted functional points = 160

Sr no.	Questions	Degree of influence (Fi)
1.	Does the system require reliable backup and recovery?	3
2.	Are data communications required?	1
3.	Are there distributed processing functions?	0
4.	Is performance critical?	2
5.	Will the system run in an existing, heavily utilized operational environment?	3
6.	Does the system require on-line data entry?	5
7.	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	2
8.	Are the master files updated on-line?	2
9.	Are the inputs, outputs, files, or inquiries complex?	2
10.	Is the internal processing complex?	1
11.	Is the code designed to be reusable	4
12.	Are conversion and installation included in the design?	3
13.	Is the system designed for multiple installations in different organizations?	4
14.	Is the application designed to facilitate change and ease of use by the user?	4
	TOTAL:	36

Unadjusted FP = 160
Adjustment Factor = 36

- Adjustment Calculation:
  Adjusted FP :- Unadjusted FP × [0.65 + (adjusted factor × 0.01)]
  =160 × [0.65 + (36 × 0.01)]
  = 160 × [0.65 + 0.36]
  = 161.6
- Adjusted Function Points Is 161.6

Time taken

There are 2 programmers thus we can do 80.5 fp per person.

Per Month (Per Programmer)= 40.25 fp approx

80.50 divided by 40.25 = 2months

Project will take 2 months.

COST:

Total Members-2

Time period- 2MONTHS

2 employees salary- 55,000 EACH

55,000 x 2 Months x 2 Employee = Rs. 2,20,000

Buffer(maintenance(40,000),Software testing(1 lakh),Data storage(1lakh))- 2,40,000

TOTAL-4,60,000 Lakhs

LINES OF CODE

Expected Lines of Code = 5000

Actual Lines of Code = 6961

- Lines of code= Expected Lines of Code / No. Of days
  = 5000/ 60
  = 83 Lines of code per day

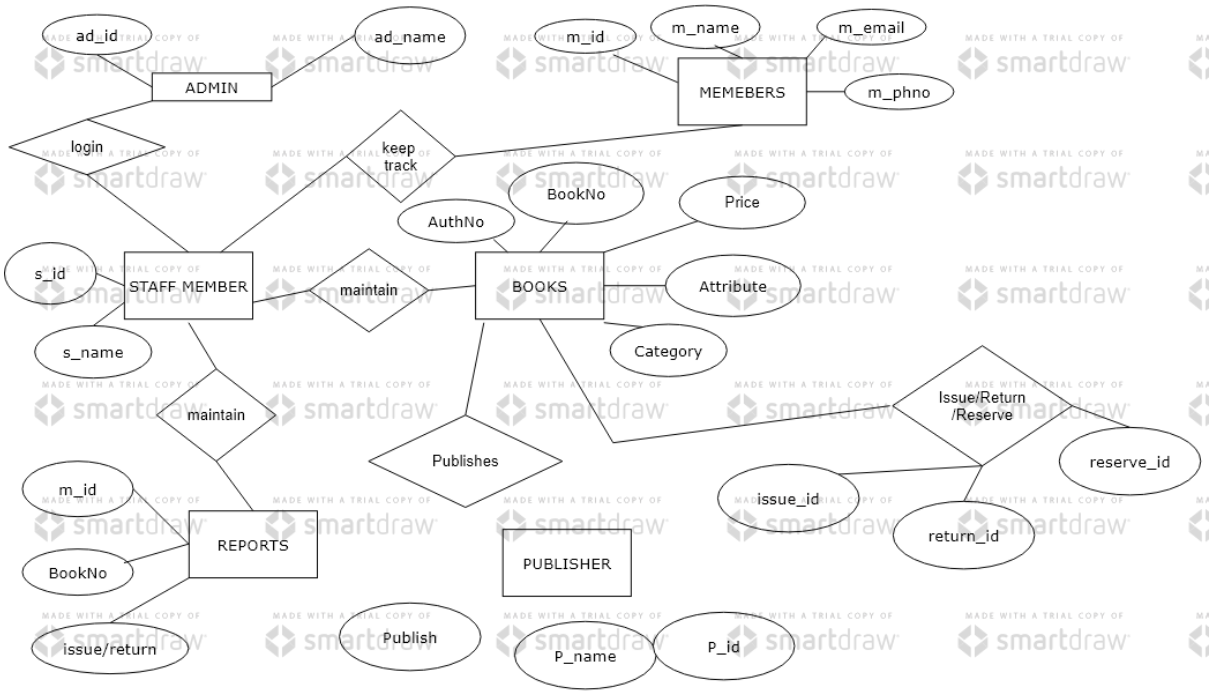
- Lines of code= Actual Lines of Code / No. Of days
  = 6961 / 60
  = 116 Lines of code per day

SYSTEM DIAGRAM

ENTITY RELATIONSHIP DIAGRAM

An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.



USE CASE DIAGRAM

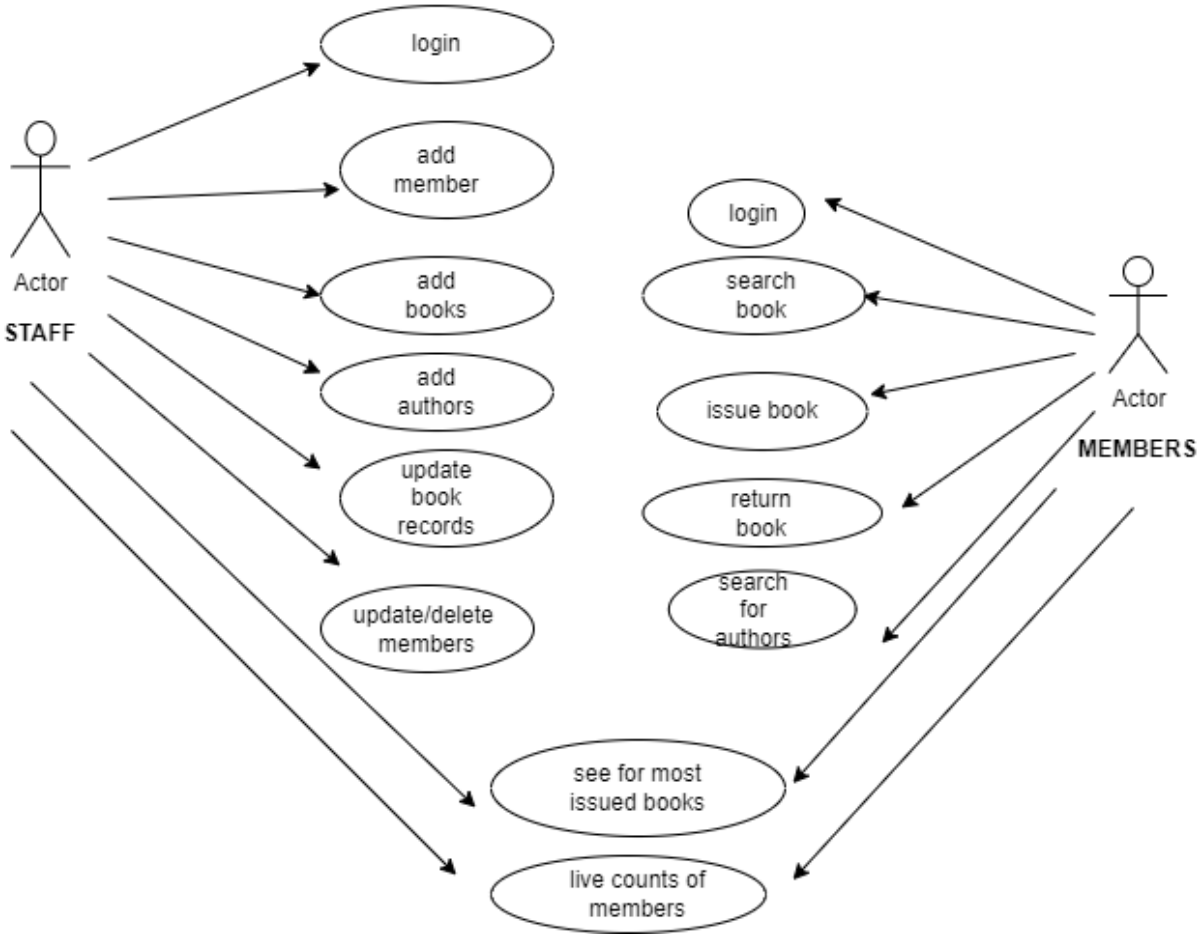
A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.

They also help identify any internal or external factors that may influence the system and should be taken into consideration.

They provide a good high-level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

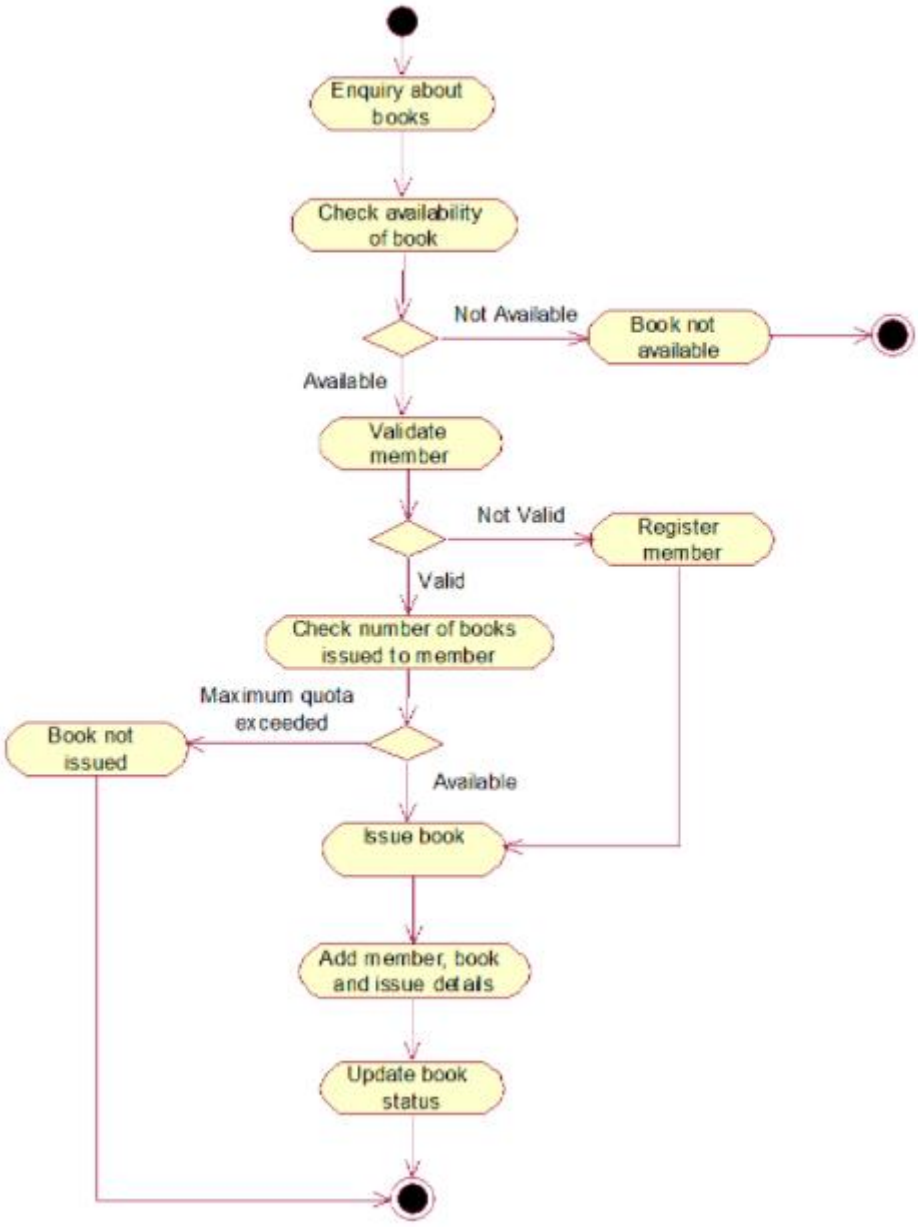
USE CASE DIAGRAM FOR LIBRARY MANAGEMENT SYSTEM



ACTIVITY DIAGRAM

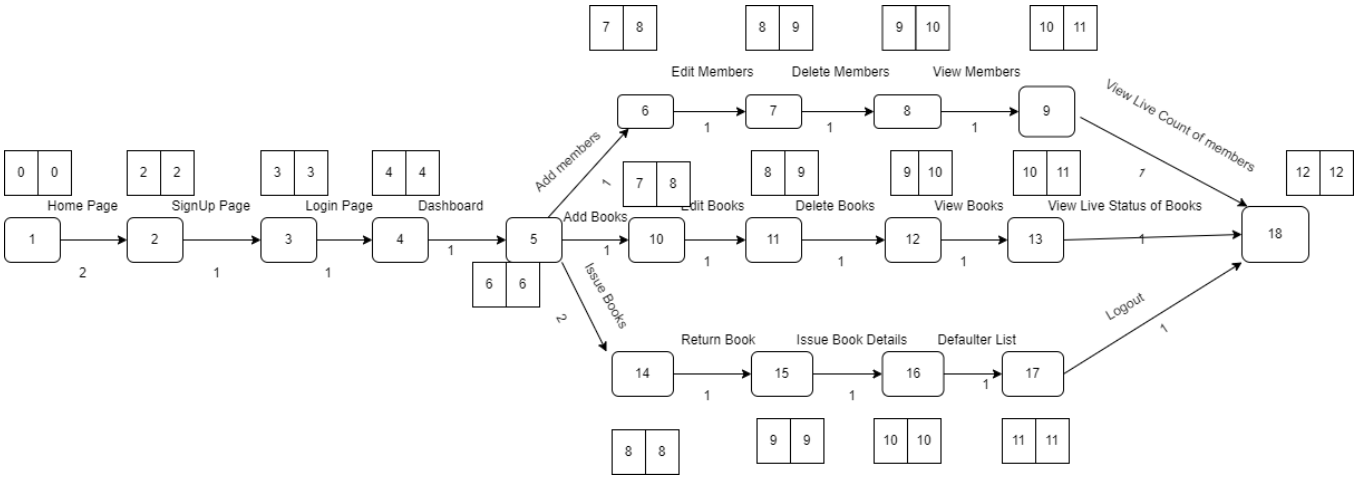
Activity diagram is another important behavioural diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modelling the flow from one activity to another activity.

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modelling. They can also describe the steps in a use case diagram. Activities modelled can be sequential and concurrent. In both cases an activity diagram will have a beginning (an initial state) and an end (a final state).



PERT AND CPM

CPM



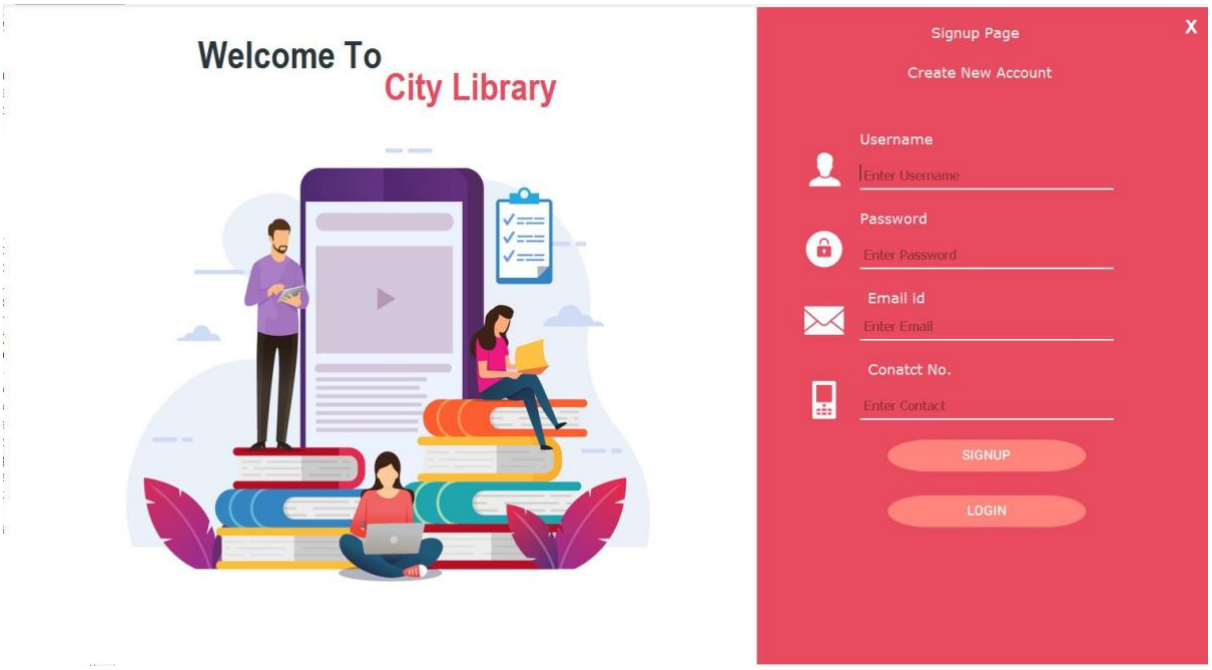
PERT

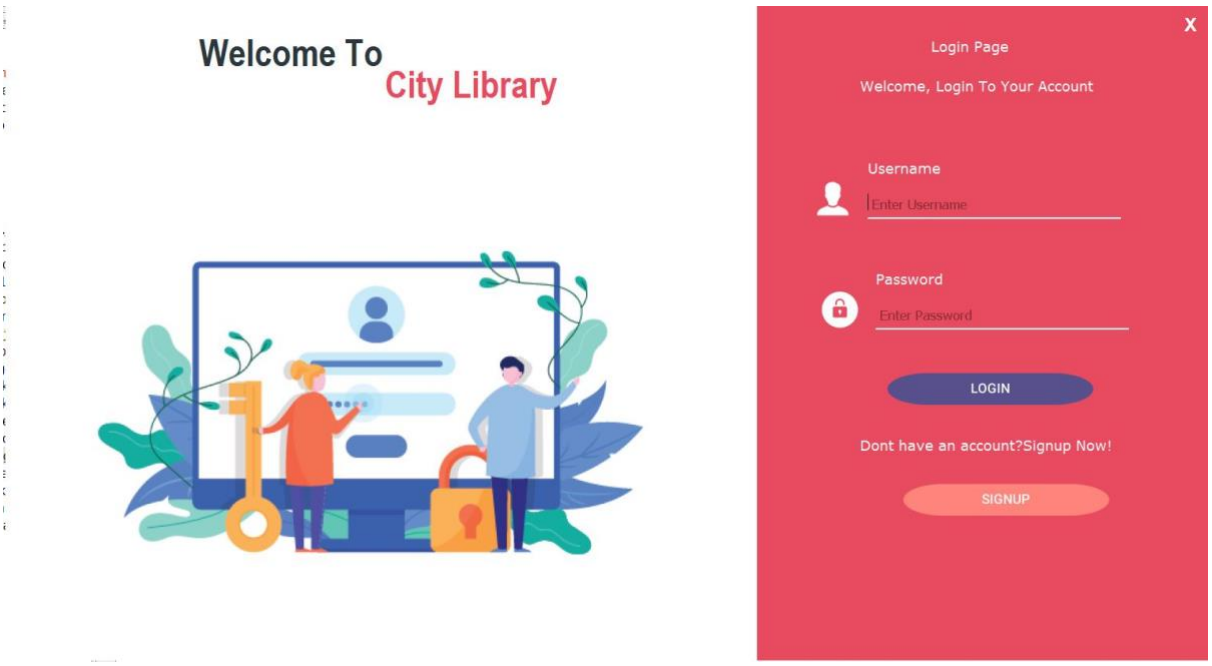
ACTIVITY	DURATION	EST	EFT	LST	LFT
1-2	2	0	2	0	2
2-3	1	2	3	2	3
3-4	1	3	4	3	4
4-5	1	4	5	5	6
5-6	1	6	7	7	8
6-7	1	7	8	8	9
7-8	1	8	9	8	9
8-9	1	9	10	9	10
9-18	1	10	11	11	12
5-10	1	6	7	7	8
10-11	1	7	8	8	9
11-12	1	8	9	9	10
12-13	1	9	10	10	11
13-18	1	10	11	11	12
5-14	2	6	8	6	8
14-15	1	8	9	8	9
15-16	1	9	10	9	10
16-17	1	10	11	10	11
17-18	1	11	12	11	12

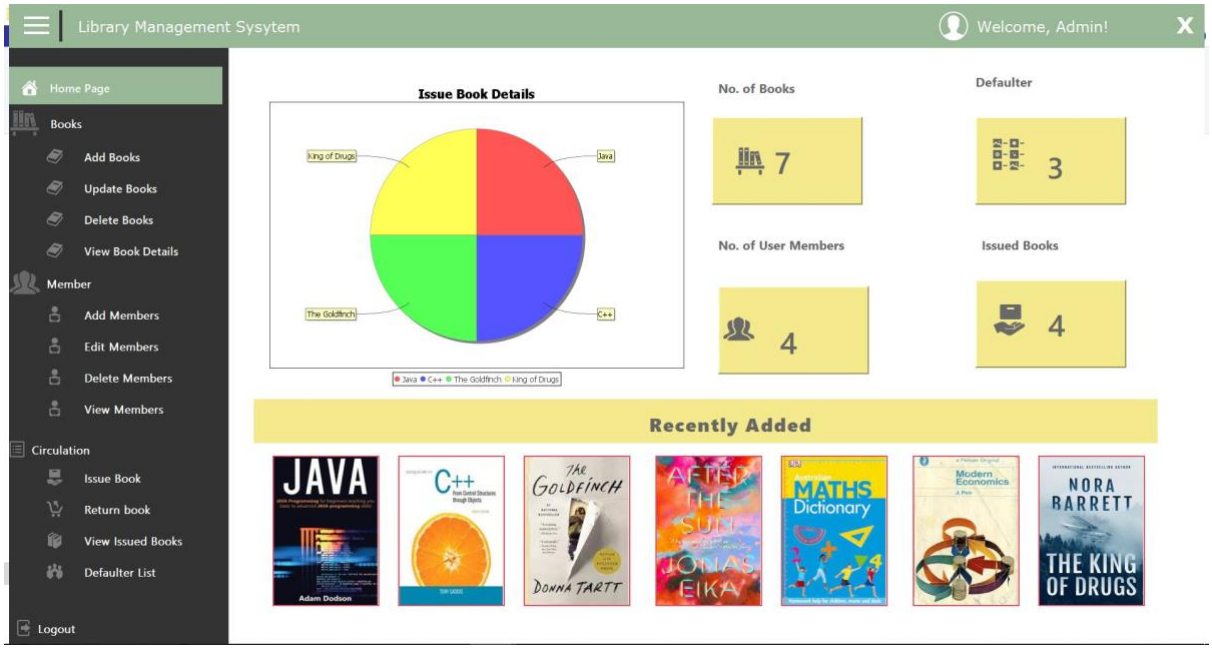
1-2-3-4-5-8-9-10-11-18

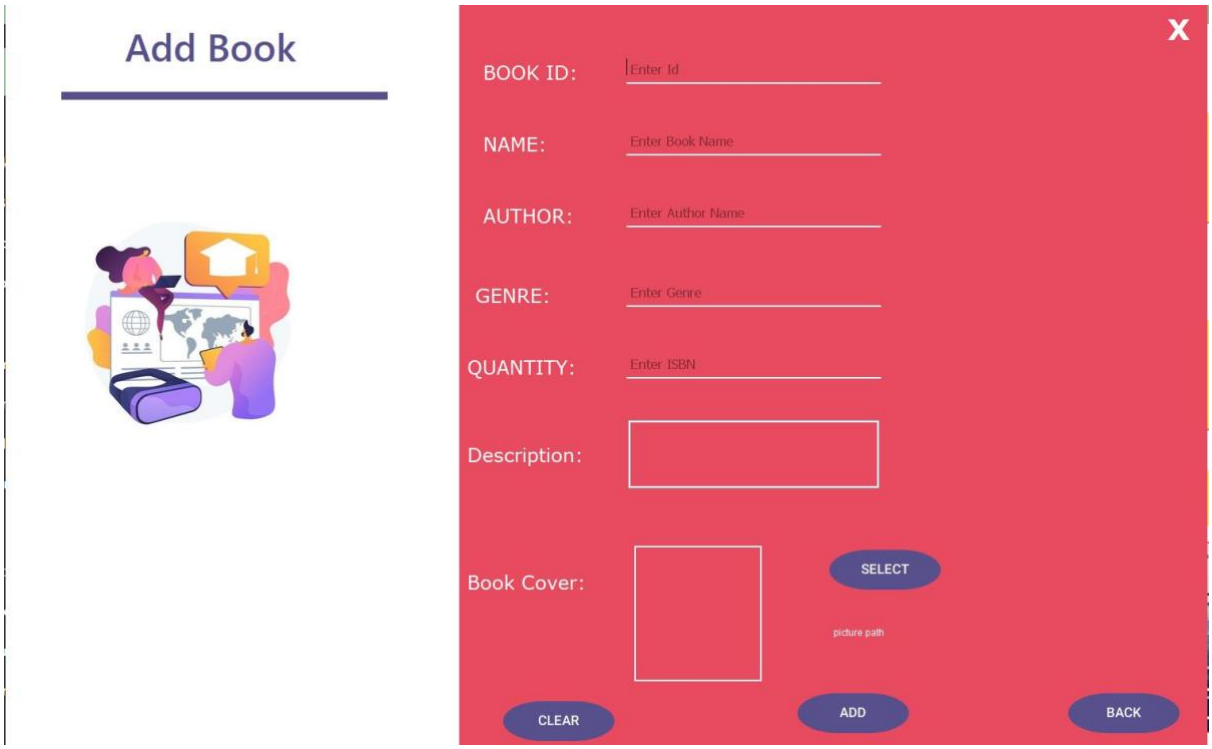
=65 Days

LayOut


Welcome To
City Library
Signup Page
Create New Account
X
Username
Enter Username
Password
Enter Password
Email id
Enter Email
Conatct No.
Enter Contact
SIGNUP
LOGIN


Welcome To
City Library
Login Page
Welcome, Login To Your Account
X
Username
Enter Username
Password
Enter Password
LOGIN
Dont have an account?Signup Now!
SIGNUP


Library Management Sysytem
Welcome, Admin!
Home Page
Books
Add Books
Update Books
Delete Books
View Book Details
Member
Add Members
Edit Members
Delete Members
View Members
Circulation
Issue Book
Return book
View Issued Books
Defaulter List
Logout
Issue Book Details
king of Drugs
Java
The Goldfinch
C++
Java C++ The Goldfinch king of Drugs
No. of Books
7
Defaulter
3
No. of User Members
4
Issued Books
4
Recently Added


Add Book
BOOK ID:
Enter Id
NAME:
Enter Book Name
AUTHOR:
Enter Author Name
GENRE:
Enter Genre
QUANTITY:
Enter ISBN
Description:
Book Cover:
SELECT
picture path
CLEAR
ADD
BACK

Edit Book

BOOK ID: Enter Id

NAME Enter Book Name

AUTHOR: Enter Author Name

GENRE: Enter Genre

QUANTITY: Enter ISBN

Description:

CLEAR EDIT BOOK BACK

X

BOOK ID:

Enter Id

DELETE BACK

X

Book Id	Name	Author	Genre	Quantity
1	Java	Adam Dodson	3	
2	C++	Tony Gaddis	19	
3	The Goldfinch	Donna tart	9	
4	After the sun	Jonas Ekay	8	
5	Math Dictionary	john smith	15	
6	Mordern Economics	J pen	27	
7	King of Drugs	Nora Barette	4	

DELETE

BACK

X

ADD MEMBERS

Member ID: Enter Member ID

Member Name: Enter Member Name

Member Phone: Enter Member Phone No

Member Email ID: Enter Member Email ID

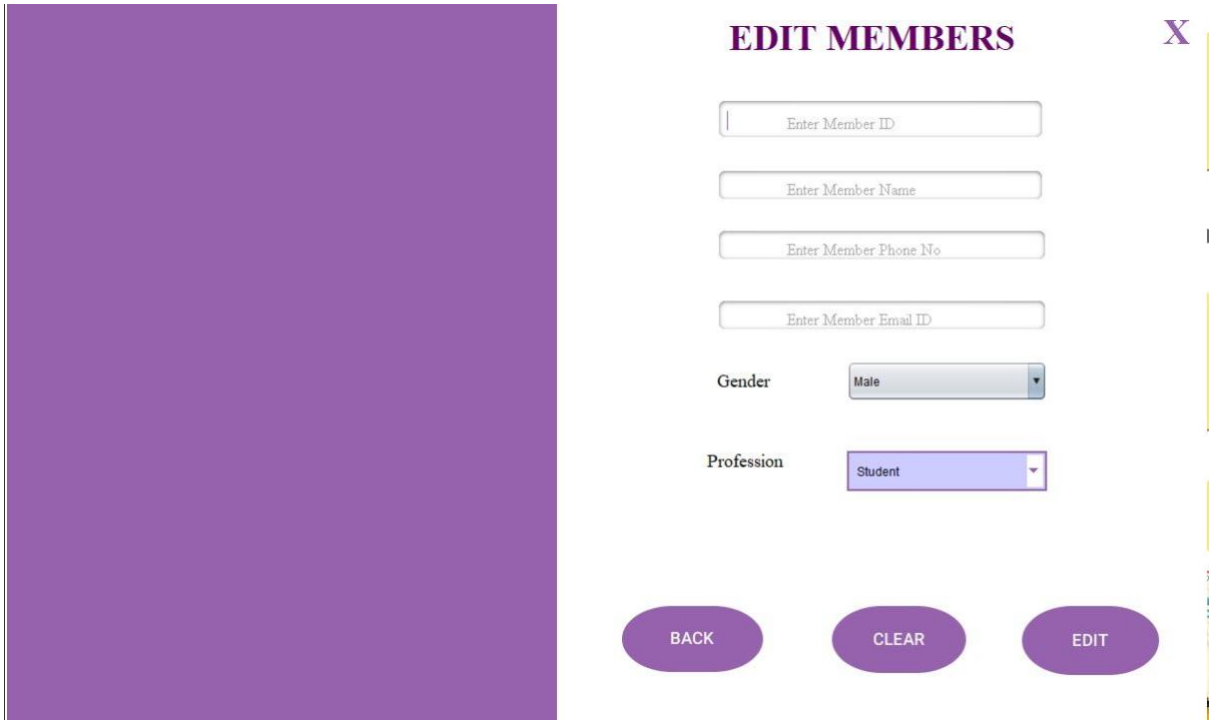
Gender: Male

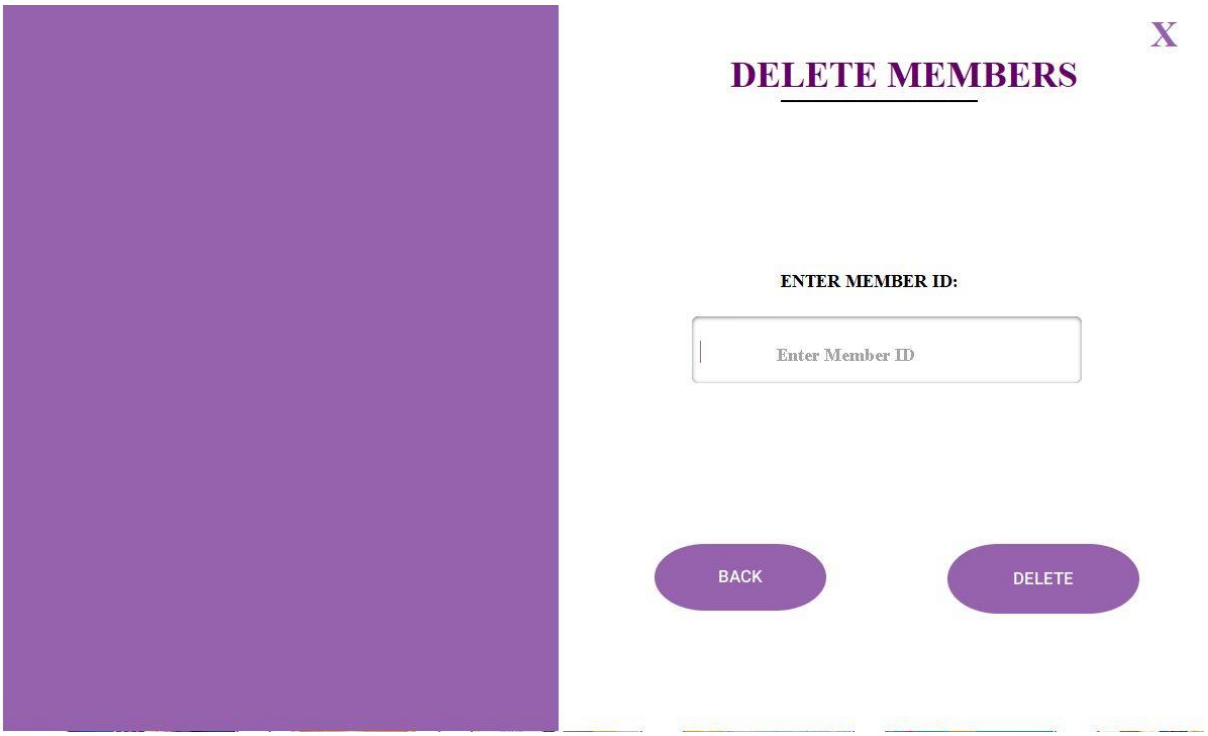
Profession: Student

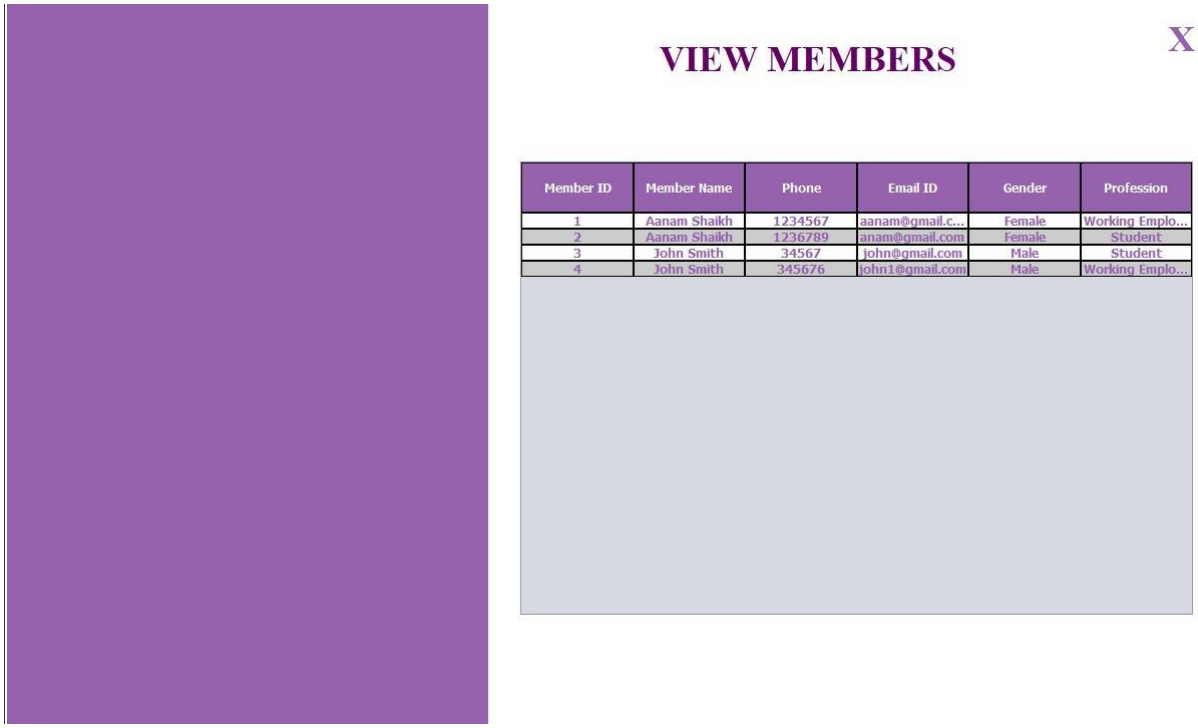
BACK

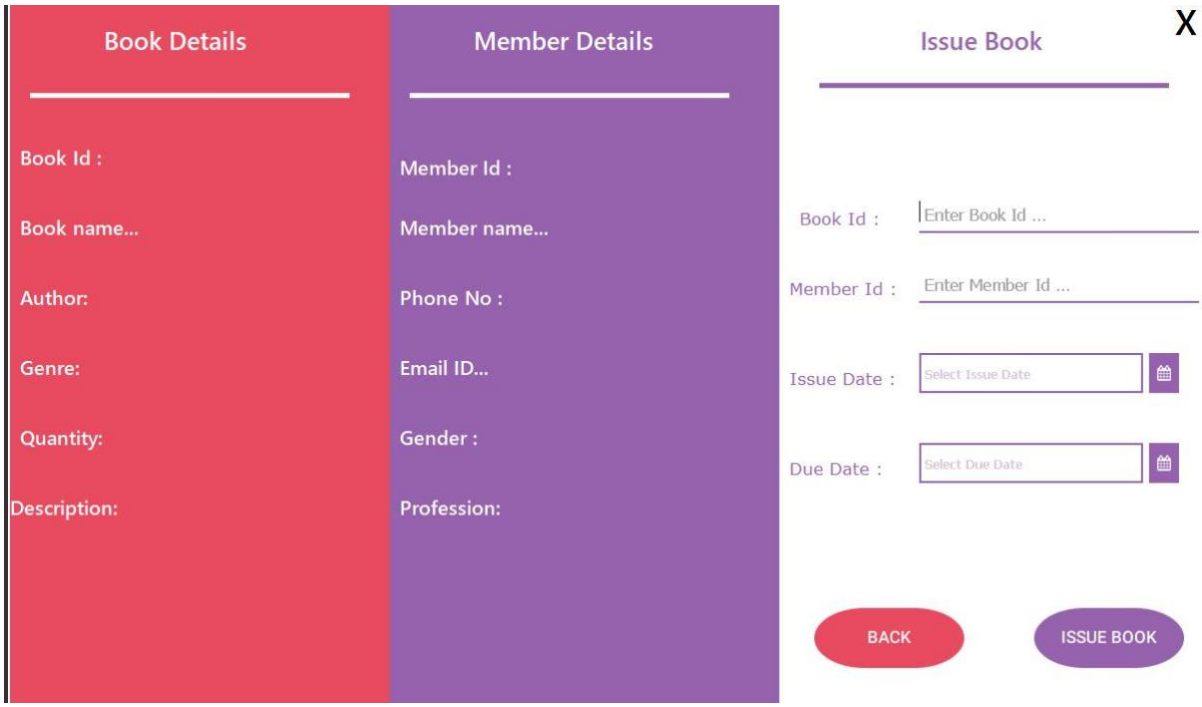
CLEAR

ADD


EDIT MEMBERS
X
Enter Member ID
Enter Member Name
Enter Member Phone No
Enter Member Email ID
Gender
Male
Profession
Student
BACK
CLEAR
EDIT


X
DELETE MEMBERS
ENTER MEMBER ID:
Enter Member ID
BACK
DELETE


VIEW MEMBERS
X
Member ID
Member Name
Phone
Email ID
Gender
Profession
1
Aanam Shaikh
1234567
aanam@gmail.c...
Female
Working Emplo...
2
Aanam Shaikh
1236789
anam@gmail.com
Female
Student
3
John Smith
34567
john@gmail.com
Male
Student
4
John Smith
345676
john1@gmail.com
Male
Working Emplo...


Book Details
Member Details
Issue Book
X
Book Id :
Book name...
Author:
Genre:
Quantity:
Description:
Member Id :
Member name...
Phone No :
Email ID...
Gender :
Profession:
Book Id :
Enter Book Id ...
Member Id :
Enter Member Id ...
Issue Date :
Select Issue Date
Due Date :
Select Due Date
BACK
ISSUE BOOK

X

Issued Book Details

Id	Book Name	Member Name	Issue Date	Due Date	Status
1	Java	Aanam Shaikh	2022-03-10	2022-03-11	pending
2	C++	Aanam Shaikh	2022-03-10	2022-03-11	pending
3	The Goldfinch	John Smith	2022-03-10	2022-03-11	pending
7	King of Drugs	John Smith	2022-03-15	2022-03-16	pending

Book Details

Issue Id :

Book name :

Member Name :

Issue Date :

Due Date :

Return Book

X

Book Id : Enter Book Id ...

Member Id : Enter Mmber Id ...

BACK FIND RETURN BOOK

X

Defaulter List

Id	Book Name	Member Name	Issue Date	Due Date	Status

View All Record

X

Issue Date : Select Issue Date

Due Date : Select due Date

SEARCH

ALL

Id	Book Name	Member Name	Issue Date	Due Date	Status

Code:

```
package jframe;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;


public class ViewMembers extends javax.swing.JFrame {
    String membername,memberphoneno,memberemailid,gender,profession;
    int memberid;

    public int getMemberid() {
        return memberid;
    }

    public String getMembername() {
        return membername;
    }

    public String getMemberphoneno() {
        return memberphoneno;
    }

    public String getMemberemailid() {
        return memberemailid;
    }

    public String getGender() {
        return gender;
    }

    public String getProfession() {
        return profession;
    }

    public void setMemberid(int memberid) {
        this.memberid = memberid;
    }

    public void setMembername(String membername) {
        this.membername = membername;
    }

    public void setMemberphoneno(String memberphoneno) {
```

```java
package jframe;


import java.sql.Connection;
import java.sql.DriverManager;

import java.sql.ResultSet;
import java.sql.Statement;

import javax.swing.table.DefaultTableModel;



public class ViewBooks extends javax.swing.JFrame {

    String s;
    //String bookName,author,genre,description;
    //int bookId,quantity;
    DefaultTableModel model;

    AddBook add = new AddBook();
    int Bookid = add.getBookId();
    String Bookname = add.getBookName();
    String author = add.getAuthor();
    String genre= add.getGenre();
    String description= add.getDescription();
    int quantity = add.getQuantity();

     public ViewBooks() {
        initComponents();
         setBookDetailsToTable();
    }

  pressWarnings("unchecked")
     Generated Code

    private void btn_DeleteBookActionPerformed(java.awt.event.ActionEvent evt) {

    }


    public void clearTable(){
        DefaultTableModel model = (DefaultTableModel) tb_Book_Details.getModel();
        model.setRowCount(0);
    }

    public void setBookDetailsToTable(){
```

```java
package jframe;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Date;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;


public class ViewAllRecord extends javax.swing.JFrame {


    DefaultTableModel model;

    public ViewAllRecord() {
        initComponents();
        setIssueBookDetailsToTable();
    }

    //to set the book details into the table
    public void setIssueBookDetailsToTable() {

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/library_management_sys", "root", "");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from issue_book_details");

            while (rs.next()) {
                String bookId = rs.getString("bookId");
                String bookName = rs.getString("bookName");
                String StudentName = rs.getString("membername");
                String issueDate = rs.getString("issue_date");
                String dueDate = rs.getString("due_date");
                String status = rs.getString("status");

                Object[] obj = {bookId, bookName, StudentName, issueDate, dueDate, status};
                model = (DefaultTableModel) tbl_issueBookDetails.getModel();
                model.addRow(obj);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```java
package jframe;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Date;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;


public class ViewAllRecord extends javax.swing.JFrame {


    DefaultTableModel model;

    public ViewAllRecord() {
        initComponents();
        setIssueBookDetailsToTable();
    }

    //to set the book details into the table
    public void setIssueBookDetailsToTable() {

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/library_management_sys", "root", "");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from issue_book_details");

            while (rs.next()) {
                String bookId = rs.getString("bookId");
                String bookName = rs.getString("bookName");
                String StudentName = rs.getString("membername");
                String issueDate = rs.getString("issue_date");
                String dueDate = rs.getString("due_date");
                String status = rs.getString("status");

                Object[] obj = {bookId, bookName, StudentName, issueDate, dueDate, status};
                model = (DefaultTableModel) tbl_issueBookDetails.getModel();
                model.addRow(obj);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```java
package jframe;

import My_Forms.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;


public class LoginPage extends javax.swing.JFrame {


    public LoginPage() {
        initComponents();
    }
    //method to insert details into userss table
        public boolean validateLogin() {
        String name = txt_username.getText();
        String pwd = txt_password.getText();

        if (name.equals("")) {
            JOptionPane.showMessageDialog(this, "please enter username");
            return false;
        }
        if (pwd.equals("")) {
            JOptionPane.showMessageDialog(this, "please enter password");
            return false;
        }

        return true;
    }

    //verify creds
    public void login() {
        String name = txt_username.getText();
        String pwd = txt_password.getText();

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/library_management_sys", "root", "");
            PreparedStatement pst = con.prepareStatement("select * from users where name = ? and pwd = ?");

            pst.setString(1, name);
            pst.setString(2, pwd);

            ResultSet rs = pst.executeQuery();
            if (rs.next()) {
```

```java
package jframe;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.table.DefaultTableModel;

public class IssuebookDetails extends javax.swing.JFrame {

    DefaultTableModel model;
    public IssuebookDetails() {
        initComponents();
        setIssueBookDetailsToTable();
    }

    //to set the book details into the table
    public void setIssueBookDetailsToTable() {

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/library_management_sys","root","");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from issue_book_details where status = '"+"pending"+"'");

            while (rs.next()) {
                String bookId = rs.getString("bookId");
                String bookName = rs.getString("bookName");
                String membername = rs.getString("membername");
                String issueDate = rs.getString("issue_date");
                String dueDate = rs.getString("due_date");
                String status = rs.getString("status");

                Object[] obj = {bookId, bookName, membername, issueDate, dueDate, status};
                model = (DefaultTableModel) tbl_issueBookDetails.getModel();
                model.addRow(obj);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
```

```java
package jframe;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Date;
import java.util.Locale;
import javax.swing.JOptionPane;


public class IssueBook extends javax.swing.JFrame {


    Locale l = null;
    public IssueBook() {
        initComponents();
        l = new Locale("en", "US");
        date_issueDate.setLocale(l);
    }

    //to fetch the book details from the database and display it to book details panel
    public void getBookDetails() {
        int bookId = Integer.parseInt(txt_bookId.getText());

        try {
            Connection con = DBConnection.getConnection();
            PreparedStatement pst = con.prepareStatement("select * from book_details where bookId = ?");
            pst.setInt(1, bookId);
            ResultSet rs = pst.executeQuery();

            if (rs.next()) {
                lbl_bookid.setText(rs.getString("bookId"));
                lbl_bookName.setText(rs.getString("bookName"));
                lbl_author.setText(rs.getString("author"));
                lbl_genre.setText(rs.getString("genre"));
                lbl_quantity.setText(rs.getString("quantity"));
                lbl_description.setText(rs.getString("description"));
            } else {
                lbl_bookError.setText("invalid book id");
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    //to fetch the member details from the database and display it to  details panel
    public void getMemberDetails() {
```

```java
package jframe;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;


public class SignupPage extends javax.swing.JFrame {


    public SignupPage() {
        initComponents();
    }

    //method to insert details into userss table
    public void insertSignupDetails(){
        String name= txt_username.getText();
        String pwd= txt_password.getText();
        String email= txt_email.getText();
        String contact= txt_contact.getText();
        //(name,password,email,contact)


        try{
            Connection con= DBConnection.getConnection();
            String sql= "insert into users (name,pwd,email,contact) values(?,?,?,?)";
            PreparedStatement pst= con.prepareStatement(sql);
            pst.setString(1, name);
            pst.setString(2, pwd );
            pst.setString(3, email);
            pst.setString(4, contact );

            int updatedRowCount= pst.executeUpdate();


            if(updatedRowCount>0){
                JOptionPane.showMessageDialog(this,"Record Inserted Successfully ");
            }else{
                JOptionPane.showMessageDialog(this,"Record Insertion Failure ");

            }

        }catch(Exception e){
            e.printStackTrace();
        }
```

```java
package jframe;
import java.sql.Connection;

import java.sql.PreparedStatement;

import javax.swing.JOptionPane;


public class EditMembers extends javax.swing.JFrame {
    String membername,memberphoneno,memberemailid,gender,profession;
    int memberid;

    public String getMembername() {
        return membername;
    }

    public void setMembername(String membername) {
        this.membername = membername;
    }

    public String getMemberphoneno() {
        return memberphoneno;
    }

    public void setMemberphoneno(String memberphoneno) {
        this.memberphoneno = memberphoneno;
    }

    public String getMemberemailid() {
        return memberemailid;
    }

    public void setMemberemailid(String memberemailid) {
        this.memberemailid = memberemailid;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public String getProfession() {
        return profession;
    }
```