

**JAVA PROGRAMMING PROJECT**

**Title: Library Management System in Java**

Group by

R Deekshitha (18030141CSE027)

Neha Pallavi M (18030141CSE007)

Chandana S (18030141CSE066)

INTRODUCTION

Library Management System is one of the most popular projects which is created using Java. So, in this article.

The project titled Library Management System is Library management software for monitoring and controlling the transactions in a library .The project “Library Management System” is developed in java, which mainly focuses on basic operations in a library like adding new member, new books, and updating new information, searching books and members and facility to borrow and return books.

**Library Management System Project**

For your better understanding, I have divided the code into the following functions and I will be explaining you function-wise:

1. [**Login**](https://www.edureka.co/blog/library-management-system-project-in-java#Login)
2. [**Connect**](https://www.edureka.co/blog/library-management-system-project-in-java#Connect)
3. [**Create/ Reset**](https://www.edureka.co/blog/library-management-system-project-in-java#CreateorReset)
4. [**User Menu**](https://www.edureka.co/blog/library-management-system-project-in-java#UserMenu)
5. [**Admin Menu**](https://www.edureka.co/blog/library-management-system-project-in-java#AdminMenu)

Also, to create a GUI, I will be using Swing. Swing is a library or a set of program components used to create graphical user interface components such as scroll bars, buttons, dialog boxes, etc.

Before I discuss the code of functions with you, let me show you the code for the main class and the libraries to be imported:

**PROGRAM**

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql. \*;

import java.text. Date Format;

import java.text. Parse Exception;

import java.text. SimpleDateFormat;

import java. util. Array List;

import java. util. Date;

import java. util. Locale;

import java. util. concurrent. Time Unit;

import javax. swing. \*;

import net.proteanit.sql. DbUtils;

public class main {

public static class ex{

public static int days=0;

}

public static void main (String [] args) {

login ();

//create ();

}

Now in this article on Library Management System in Java, let us understand the code of all the above functions.

#### **Login**

I have created this function to enable the user and the admin login. So, initially when a user logs in for the first time, that user will be an admin by default, and the username and password will be {admin, admin}. Refer below.

For this schema, I have considered only one admin. So, once a user logs in as an admin, he or she will be redirected to the admin menu as below. I will discuss the functions of the admin in the [admin menu](https://www.edureka.co/blog/library-management-system-project-in-java#AdminMenu) section.

**PROGRAM**

public static void login () {

JFrame f=new JFrame("Login”) ;//creating instance of JFrame

JLabel l1, l2;

l1=new JLabel("Username"); //Create label Username

l1. set Bounds (30,15, 100,30); //x axis, y axis, width, height

l2=new JLabel("Password"); //Create label Password

l2. set Bounds (30,50, 100,30);

JTextField F\_user = new JTextField (); //Create text field for username

F\_user. set Bounds (110, 15, 200, 30);

JPasswordField F\_pass=new JPasswordField (); //Create text field for password

F\_pass. set Bounds (110, 50, 200, 30);

Button login but=new JButton("Login”) ;//creating instance of JButton for Login Button

login but. set Bounds (130,90,80,25) ;//Dimensions for button

login\_but. add ActionListener (new ActionListener () { //Perform action

public void actionPerformed(ActionEvent e){

String username = F\_user. get Text (); //Store username entered by the user in the variable "username"

String password = F\_pass. get Text (); //Store password entered by the user in the variable "password"

if(username.equals("")) //If username is null

{

JOptionPane.showMessageDialog(null, “Please enter username"); //Display dialog box with the message

}

else if(password. equals ("")) //If password is null

{

JOptionPane.showMessageDialog(null, “Please enter password"); //Display dialog box with the message

}

else { //If both the fields are present then to login the user, check whether the user exists already

//System.out.println("Login connect");

Connection connection=connect (); //Connect to the database

try

{

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY"); //Use the database with the name "Library"

String st = ("SELECT \* FROM USERS WHERE USERNAME='"+username+"' AND PASSWORD='"+password+"'"); //Retrieve username and passwords from users

Result Set rs = stmt. execute Query(st); //Execute query

if (rs. next () ==false) { //Move pointer below

System.out.print("No user");

JOptionPane.showMessageDialog(null, “Wrong Username/Password!"); //Display Message

}

else {

f. dispose ();

rs. before First (); //Move the pointer above

while (rs. next ())

{

String admin = rs. get String("ADMIN"); //user is admin

//System.out.println(admin);

String UID = rs. get String("UID"); //Get user ID of the user

if (admin. equals ("1")) {//If Boolean value 1

admin\_menu (); //redirect to admin menu

}

else{

user menu (UID); //redirect to user menu for that user ID

}

}

}

}

catch (Exception ex) {

ex. printStackTrace ();

}

}

}

});

f.add(F\_pass); //add password

f.add(login\_but) ;//adding button in JFrame

f.add(F\_user); //add user

f.add(l1); // add label1 i.e., for username

f.add(l2); // add label2 i.e. for password

f. set Size (400,180) ;//400 width and 500 height

f. set Layout(null);//using no layout managers

f. set Visible(true);//making the frame visible

f. setLocationRelativeTo(null);

}

#### **Connect**

The connect function is used to connect the database to the [GUI](https://www.edureka.co/blog/java-swing/). So, to do that, I have mentioned the below code:

**PROGRAM**

public static Connection connect()

{

try {

Class.forName("com. mysql.cj. jdbc. Driver");

//System.out.println("Loaded driver");

Connection con = DriverManager.getConnection("jdbc: mysql://localhost/mysql?user=root&password=edureka");

//System.out.println("Connected to MySQL");

return con;

}

catch (Exception ex) {

ex. printStackTrace ();

}

return null;

}

In the above function, we are connecting our **MySQL database** with the**username “root”** and **password “edureka”** to our application. Now, once the application is connected to the database, our next step is to create or reset the database. So, next in this article on Library Management System Project in Java, let us discuss the Create function.

#### **Create**

The create function is used to create the database, tables and add data into these tables. So, to do that, [SQL statements](https://www.edureka.co/blog/sql-commands) will be used as below.

**PROGRAM**

public static void create () {

try {

Connection connection=connect ();

Result Set result Set = connection. get Metadata (). getCatalogs ();

//iterate each catalog in the ResultSet

while (resultSet.next()) {

// Get the database name, which is at position 1

String database Name = resultSet.getString(1);

if(databaseName.equals("library")) {

//System.out.print("yes");

Statement stmt = connection. create Statement ();

//Drop database if it pre-exists to reset the complete database

String sql = "DROP DATABASE library";

stmt. execute Update(sql);

}

}

Statement stmt = connection. create Statement ();

String sql = "CREATE DATABASE LIBRARY"; //Create Database

stmt. execute Update(sql);

stmt. execute Update ("USE LIBRARY"); //Use Database

//Create Users Table

String sql1 = "CREATE TABLE USERS (UID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, USERNAME VARCHAR (30), PASSWORD VARCHAR (30), ADMIN BOOLEAN)";

stmt. execute Update(sql1);

//Insert into users table

stmt. execute Update ("INSERT INTO USERS (USERNAME, PASSWORD, ADMIN) VALUES ('admin','admin’, TRUE)");

//Create Books table

stmt. execute Update ("CREATE TABLE BOOKS (BID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, BNAME VARCHAR (50), GENRE VARCHAR (20), PRICE INT)");

//Create Issued Table

stmt. execute Update ("CREATE TABLE ISSUED (IID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, UID INT, BID INT, ISSUED\_DATE VARCHAR (20), RETURN\_DATE VARCHAR (20), PERIOD INT, FINE INT)");

//Insert into books table

stmt. execute Update ("INSERT INTO BOOKS (BNAME, GENRE, PRICE) VALUES ('War and Peace', 'Mystery', 200), ('The Guest Book', 'Fiction', 300), ('The Perfect Murder’, ‘Mystery', 150), ('Accidental Presidents', 'Biography', 250), ('The Wicked King’, ‘Fiction', 350)");

resultSet.close();

}

catch (Exception ex) {

ex. printStackTrace ();

}

}

Now, that we have created the database, connected with GUI and enables the login function, next in this article on Library Management System Project in Java, let us now discuss the functions of the User Menu.

#### **User Menu**

The User Menu is designed to show details of all the books present in the library and the books issued by the user.

**PROGRAM**

public static void user menu (String UID) {

JFrame f=new JFrame ("User Functions"); //Give dialog box name as User functions

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE); //Exit user menu on closing the dialog box

JButton view but=new JButton ("View Books”) ;//creating instance of JButton

view\_but. set Bounds (20,20,120,25) ;//x axis, y axis, width, height

view\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

JFrame f = new JFrame ("Books Available"); //View books stored in database

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

Connection connection = connect ();

String sql="select \* from BOOKS"; //Retrieve data from database

try {

Statement stmt = connection. create Statement (); //connect to database

stmt. execute Update ("USE LIBRARY"); // use library

stmt=connection. Create Statement ();

ResultSet rs=stmt. execute Query (sql);

JTable book\_list= new JTable (); //show data in table format

book\_list. set Model (DbUtils.resultSetToTableModel(rs));

JScrollPane scroll Pane = new JScrollPane(book\_list); //enable scroll bar

f.add(scroll Pane); //add scroll bar

f. set Size (800, 400); //set dimensions of view books frame

f. set Visible(true);

f. setLocationRelativeTo(null);

} catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

}

);

JButton my\_book=new JButton ("My Books”) ;//creating instance of JButton

my\_book. set Bounds (150,20,120,25) ;//x axis, y axis, width, height

my\_book. add ActionListener (new ActionListener () {//Perform action

public void actionPerformed (ActionEvent e) {

JFrame f = new JFrame ("My Books"); //View books issued by user

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

int UID\_int = Integer.parseInt(UID); //Pass user ID

//. iid, issued.uid, issued.bid, issued.issued\_date,issued.return\_date,issued,

Connection connection = connect (); //connect to database

//retrieve data

String sql="select distinct issued. \*, books. bname, books. genre, books. Price from issued, books " + "where ((issued.uid=" + UID\_int + ") and (books.bid in (select bid from issued where issued.uid="+UID\_int+"))) group by iid";

String sql1 = "select bid from issued where uid="+UID\_int;

try {

Statement stmt = connection. create Statement ();

//use database

stmt. execute Update ("USE LIBRARY");

stmt=connection. create Statement ();

//store in array

Array List books list = new Array List();

ResultSet rs=stmt. execute Query(sql);

JTable book\_list= new JTable (); //store data in table format

book\_list. set Model (DbUtils.resultSetToTableModel(rs));

//enable scroll bar

JScrollPane scroll Pane = new JScrollPane(book\_list);

f.add (scroll Pane); //add scroll bar

f. set Size (800, 400); //set dimensions of my books frame

f. set Visible(true);

f. setLocationRelativeTo(null);

} catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

}

);

f.add(my\_book); //add my books

f.add(view\_but); // add view books

f. set Size (300,100) ;//400 width and 500 height

f. set Layout(null);//using no layout managers

f. set Visible(true);//making the frame visible

f. setLocationRelativeTo(null);

}

Next, in this article on Library Management System Project in Java, let us discuss the code for Admin Menu function.

#### **Admin Menu**

The Admin Menu is designed to show details of users, books, issued books, add books, return books, add user, and create or reset the database.

**PROGRAM**

public static void admin\_menu () {

JFrame f=new JFrame ("Admin Functions"); //Give dialog box name as admin functions

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE); //

JButton create but=new JButton("Create/Reset");//creating instance of JButton to create or reset database

create but. set Bounds (450,60,120,25) ;//x axis, y axis, width, height

create but. add ActionListener (new ActionListener () {//Perform action

public void actionPerformed (ActionEvent e){

create (); //Call create function

JOptionPane.showMessageDialog(null, “Database Created/Reset!"); //Open a dialog box and display the message

}

});

JButton view\_but=new JButton ("View Books”) ;//creating instance of JButton to view books

view\_but. set Bounds (20,20,120,25) ;//x axis, y axis, width, height

view\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

JFrame f = new JFrame ("Books Available");

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

Connection connection = connect (); //connect to database

String sql="select \* from BOOKS"; //select all books

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY"); //use database

stmt=connection. create Statement ();

ResultSet rs=stmt. execute Query (sql);

JTable book\_list= new JTable (); //view data in table format

book\_list. setModel(DbUtils.resultSetToTableModel(rs));

//mention scroll bar

JScrollPane scroll Pane = new JScrollPane(book\_list);

f.add (scroll Pane); //add scroll pane

f. set Size (800, 400); //set size for frame

f. set Visible(true);

f. setLocationRelativeTo(null);

} catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

}

);

JButton users but=new JButton ("View Users”) ;//creating instance of JButton to view users

users but. set Bounds (150,20,120,25) ;//x axis, y axis, width, height

users\_but. add ActionListener (new ActionListener () {//Perform action on click button

public void actionPerformed (ActionEvent e) {

JFrame f = new JFrame ("Users List");

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

Connection connection = connect ();

String sql="select \* from users"; //retrieve all users

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY"); //use database

stmt=connection. create Statement ();

ResultSet rs=stmt. execute Query(sql);

JTable book\_list= new JTable ();

book\_list. setModel (DbUtils.resultSetToTableModel(rs));

//mention scroll bar

JScrollPane scroll Pane = new JScrollPane(book\_list);

f.add (scroll Pane); //add scroll pane

f. set Size (800, 400); //set size for frame

f. set Visible(true);

f. setLocationRelativeTo(null);

} catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

}

);

JButton issued but=new JButton ("View Issued Books");//creating instance of JButton to view the issued books

issued\_but. set Bounds (280,20,160,25) ;//x axis, y axis, width, height

issued\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e){

JFrame f = new JFrame ("Users List");

//f. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

Connection connection = connect ();

String sql="select \* from issued";

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY");

stmt=connection. create Statement ();

ResultSet rs=stmt. execute Query (sql);

JTable book\_list= new JTable ();

book\_list. setModel (DbUtils.resultSetToTableModel(rs));

JScrollPane scroll Pane = new JScrollPane(book\_list);

f.add (scroll Pane);

f. set Size (800, 400);

f. set Visible (true);

f. setLocationRelativeTo(null);

} catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

}

);

JButton add\_user=new JButton ("Add User"); //creating instance of JButton to add users

add\_user. set Bounds (20,60,120,25); //set dimensions for button

add\_user. add ActionListener (new ActionListener() {

public void actionPerformed (ActionEvent e){

JFrame g = new JFrame ("Enter User Details"); //Frame to enter user details

//g. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

//Create label

JLabel l1,l2;

l1=new JLabel("Username"); //label 1 for username

l1. set Bounds (30,15, 100,30);

l2=new JLabel("Password"); //label 2 for password

l2. set Bounds (30,50, 100,30);

//set text field for username

JTextField F\_user = new JTextField ();

F\_user. set Bounds (110, 15, 200, 30);

//set text field for password

JPasswordField F\_pass=new JPasswordField ();

F\_pass. set Bounds (110, 50, 200, 30);

//set radio button for admin

JRadioButton a1 = new JRadioButton("Admin");

a1. set Bounds (55, 80, 200,30);

//set radio button for user

JRadioButton a2 = new JRadioButton("User");

a2. set Bounds (130, 80, 200,30);

//add radio buttons

Button Group bg=new ButtonGroup ();

bg.add(a1); bg.add(a2);

JButton create\_but=new JButton("Create”) ;//creating instance of JButton for Create

create\_but. set Bounds (130,130,80,25) ;//x axis, y axis, width, height

create\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

String username = F\_user. get Text ();

String password = F\_pass. get Text ();

Boolean admin = false;

if(a1. is Selected ()) {

admin=true;

}

Connection connection = connect ();

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY");

stmt. execute Update ("INSERT INTO USERS (USERNAME, PASSWORD, ADMIN) VALUES ('"+username+"','"+password+"',"+admin+")");

JOptionPane.showMessageDialog(null,” User added!");

g. dispose ();

}

catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

});

g.add(create\_but);

g.add(a2);

g.add(a1);

g.add(l1);

g.add(l2);

g.add(F\_user);

g.add(F\_pass);

g. set Size (350,200) ;//400 width and 500 height

g. set Layout(null);//using no layout managers

g. set Visible(true);//making the frame visible

g. setLocationRelativeTo(null);

}

});

JButton add book=new JButton("Add Book"); //creating instance of JButton for adding books

add book. set Bounds (150,60,120,25);

add\_book. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e){

//set frame wot enter book details

JFrame g = new JFrame ("Enter Book Details");

//g. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

// set labels

JLabel l1, l2, l3;

l1=new JLabel ("Book Name"); //lebel 1 for book name

l1. set Bounds (30,15, 100,30);

l2=new JLabel("Genre"); //label 2 for genre

l2. set Bounds (30,53, 100,30);

l3=new JLabel("Price"); //label 2 for price

l3. set Bounds (30,90, 100,30);

//set text field for book name

JTextField F\_bname = new JTextField ();

F\_bname. set Bounds (110, 15, 200, 30);

//set text field for genre

JTextField F\_genre=new JTextField ();

F\_genre. set Bounds (110, 53, 200, 30);

//set text field for price

JTextField F\_price=new JTextField ();

F\_price. setBounds (110, 90, 200, 30);

JButton create\_but=new JButton("Submit”) ;//creating instance of JButton to submit details

create\_but. setBounds (130,130,80,25) ;//x axis, y axis, width, height

create\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

// assign the book name, genre, price

String bname = F\_bname. get Text ();

String genre = F\_genre. get Text ();

String price = F\_price. get Text ();

//convert price of integer to int

int price\_int = Integer.parseInt(price);

Connection connection = connect ();

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY");

stmt. execute Update ("INSERT INTO BOOKS (BNAME, GENRE, PRICE) VALUES ('"+bname+"','"+genre+"',"+price\_int+")");

JOptionPane.showMessageDialog(null, “Book added!");

g. dispose ();

}

catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

});

g.add(l3);

g.add(create\_but);

g.add(l1);

g.add(l2);

g.add(F\_bname);

g.add(F\_genre);

g.add(F\_price);

g. set Size (350,200) ;//400 width and 500 height

g. set Layout(null);//using no layout managers

g. set Visible(true);//making the frame visible

g. setLocationRelativeTo(null);

}

});

JButton issue book=new JButton ("Issue Book"); //creating instance of JButton to issue books

issue book. setBounds (450,20,120,25);

issue book. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e){

//enter details

JFrame g = new JFrame ("Enter Details");

//g. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

//create labels

JLabel l1, l2, l3, l4;

l1=new JLabel ("Book ID(BID)"); // Label 1 for Book ID

l1. setBounds (30,15, 100,30);

l2=new JLabel ("User ID(UID)"); //Label 2 for user ID

l2. setBounds (30,53, 100,30);

l3=new JLabel("Period(days)"); //Label 3 for period

l3. setBounds (30,90, 100,30);

l4=new JLabel ("Issued Date (DD-MM-YYYY)"); //Label 4 for issue date

l4. setBounds (30,127, 150,30);

JTextField F\_bid = new JTextField ();

F\_bid. SetBounds (110, 15, 200, 30);

JTextField F\_uid=new JTextField ();

F\_uid. SetBounds (110, 53, 200, 30);

JTextField Period=new JTextField ();

F\_period. setBounds (110, 90, 200, 30);

JTextField F\_issue=new JTextField ();

F\_issue. SetBounds (180, 130, 130, 30);

JButton create\_but=new JButton("Submit”) ;//creating instance of JButton

create\_but. setBounds (130,170,80,25) ;//x axis, y axis, width, height

create\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

String uid = F\_uid. get Text ();

String bid = F\_bid. get Text ();

String period = F\_period. get Text ();

String issued date = F\_issue. get Text ();

int period\_int = Integer.parseInt(period);

Connection connection = connect ();

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY");

stmt. execute Update ("INSERT INTO ISSUED (UID, BID,ISSUED\_DATE,PERIOD) VALUES ('"+uid+"','"+bid+"','"+issued date+"',"+period\_int+")");

JOptionPane.showMessageDialog(null, “Book Issued!");

g. dispose ();

}

catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

});

g.add(l3);

g.add(l4);

g.add(create\_but);

g.add(l1);

g.add(l2);

g.add(F\_uid);

g.add(F\_bid);

g.add(F\_period);

g.add(F\_issue);

g. set Size (350,250) ;//400 width and 500 height

g. set Layout(null);//using no layout managers

g. set Visible(true);//making the frame visible

g. setLocationRelativeTo(null);

}

});

JButton return book=new JButton ("Return Book"); //creating instance of JButton to return books

return book. setBounds (280,60,160,25);

return book. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e){

JFrame g = new JFrame ("Enter Details");

//g. setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

//set labels

JLabel l1, l2, l3, l4;

l1=new JLabel ("Issue ID(IID)"); //Label 1 for Issue ID

l1. setBounds (30,15, 100,30);

l4=new JLabel ("Return Date (DD-MM-YYYY)");

l4. setBounds (30,50, 150,30);

JTextField F\_iid = new JTextField ();

F\_iid. SetBounds (110, 15, 200, 30);

JTextField F\_return=new JTextField ();

F\_return. SetBounds (180, 50, 130, 30);

JButton create\_but=new JButton("Return");//creating instance of JButton to mention return date and calculcate fine

create\_but. setBounds (130,170,80,25) ;//x axis, y axis, width, height

create\_but. add ActionListener (new ActionListener () {

public void actionPerformed (ActionEvent e) {

String iid = F\_iid. get Text ();

String return date = F\_return. get Text ();

Connection connection = connect ();

try {

Statement stmt = connection. create Statement ();

stmt. execute Update ("USE LIBRARY");

//Initialize date1 with NULL value

String date1=null;

String date2=return date; //Initialize date2 with return date

//select issue date

ResultSet rs = stmt. Execute Query ("SELECT ISSUED\_DATE FROM ISSUED WHERE IID="+iid);

while (rs. Next ()) {

date1 = rs. get String (1);

}

try {

Date date\_1=new SimpleDateFormat("dd-MM-yyyy"). parse(date1);

Date date\_2=new SimpleDateFormat("dd-MM-yyyy"). parse(date2);

//subtract the dates and store in diff

long diff = date\_2. Get Time () - date\_1. Get Time ();

//Convert diff from milliseconds to days

ex. days=(int)(TimeUnit.DAYS.convert(diff, TimeUnit.MILLISECONDS));

} catch (Parse Exception e1) {

// TODO Auto-generated catch block

e1. printStackTrace ();

}

//update return date

stmt. Execute Update ("UPDATE ISSUED SET RETURN\_DATE='"+return date+"' WHERE IID="+iid);

g. dispose ();

Connection connection1 = connect ();

Statement stmt1 = connection1.createStatement();

stmt1.executeUpdate("USE LIBRARY");

ResultSet rs1 = stmt1.executeQuery("SELECT PERIOD FROM ISSUED WHERE IID="+iid); //set period

String diff=null;

while (rs1.next()) {

diff = rs1.getString(1);

}

int diff\_int = Integer.parseInt(diff);

if (ex. days amp; amp;amp;amp;amp;amp;amp;amp;amp;amp;gt;diff\_int) { //If number of days are more than the period then calculcate fine

//System.out.println(ex. days);

int fine = (ex. days-diff\_int) \*10; //fine for every day after the period is Rs 10.

//update fine in the system

stmt1.executeUpdate("UPDATE ISSUED SET FINE="+fine+" WHERE IID="+iid);

String fine\_str = ("Fine: Rs. "+fine);

JOptionPane.showMessageDialog(null, fine\_str);

}

JOptionPane.showMessageDialog(null, “Book Returned!");

}

catch (SQLException e1) {

// TODO Auto-generated catch block

JOptionPane.showMessageDialog(null, e1);

}

}

});

g.add(l4);

g.add(create\_but);

g.add(l1);

g.add(F\_iid);

g.add(F\_return);

g. set Size (350,250) ;//400 width and 500 height

g. set Layout(null);//using no layout managers

g. set Visible(true);//making the frame visible

g.set LocationRelativeTo(null);

}

});

f.add(create\_but);

f.add (return book);

f.add (issue book);

f.add(add\_book);

f.add(issued\_but);

f.add(users\_but);

f.add(view\_but);

f.add(add\_user);

f. set Size (600,200) ;//400 width and 500 height

f. set Layout(null);//using no layout managers

f. set Visible(true);//making the frame visible

f. setLocationRelativeTo(null);

}

}

 }

**THANK YOU…**