



SZABIST

Object Oriented Programming

PROJECT REPORT

Project = Library Management System

GROUP MEMBERS

Ilyaan Umatia - (2212288)

Salman Hamzo - (2212303)

Ovais Salam - (2212297)

ACKNOWLEDGMENT

We are really grateful because we managed to complete our project within the time given by our teachers [Syed Muhammad Hassan & Bakhtawar Abbasi]. This project cannot be completed without the effort and cooperation of our group members, Group members [Illyaan Umatia, Salman Hamzo & Ovais Salam].

We also sincerely thank our teachers [Syed Muhammad Hassan & Bakhtawar Abbasi] for their guidance and encouragement in finishing this project and also for teaching us in this course.

Object Oriented Programming

Library Management System

3	Project Description Functionalities & Features
4-23	Forms and Source Code
4-6	Login Form
7-8	Home Form
9-10	Forgot Form
11-12	NewBook Form
12-13	Signup Form
14	Loading Form
15-16	Return Book Form
16-17	Statistics(Records) Form
18-19	Add Student Form
19-20	Issue Book Form
21-23	Print Card Form
24	Conclusion



PROJECT DESCRIPTION

Our Library Management System is an automated software solution designed to efficiently manage library operations, resources, and services. This project aims to develop a user-friendly and scalable application that caters specifically to the needs of a library.

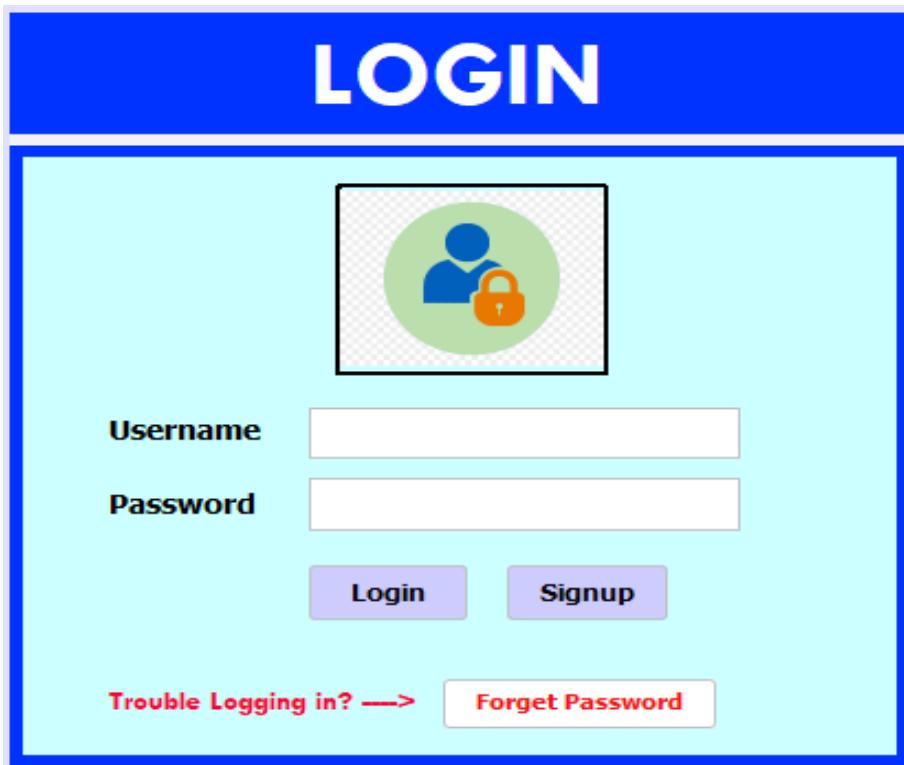
It aims to improve library operations, access to resources, and enhance user satisfaction. By automating various processes, it will save time, reduce manual effort, and provide a good experience for both librarians and students.

Functionalities & Features:

- **Login System:** Admin account for the librarian to access the records and operations.
- **Add Book:** Easily add new books to the library's database.
- **Records:** Maintain comprehensive records of books, students, and transactions.
- **Add Student:** Effortlessly add new students to the library system.
- **Issue Book:** Facilitate the process of borrowing books from the library.
- **Return Book:** Organized return process for borrowed books.
- **Print Library Card:** Generate library cards for students to access library resources.

SOURCE CODE AND FORMS

Login Form:



```
package Library_Management;

import DatabaseConnector.DatabaseHelper;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;

public class Login extends javax.swing.JFrame {
    Connection con = null;
    ResultSet rs = null; // Declaration and initialization of variables for database operations.
    PreparedStatement ps = null;
    /**
     * Creates new form Login
     */
    public Login() {
        initComponents();
        con = DatabaseHelper.getConnection();
    }

    /**
     * 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.
     */
}
```

The screenshot shows the Apache NetBeans IDE 15 interface. The main window displays the `Login.java` file under the `Source` tab. The code implements event handlers for three buttons: `btnsignupActionPerformed`, `btnforgotActionPerformed`, and `btnloginActionPerformed`. The `btnloginActionPerformed` method retrieves user input from text fields, prepares a database query, and creates a `Loading` form to handle file uploads.

```
private void btnsignupActionPerformed(java.awt.event.ActionEvent evt) {
    Signup si = new Signup(); // Creates a new instance of the "Signup" class, representing the signup form.
    si.setVisible(true);
    this.dispose();
}

private void btnforgotActionPerformed(java.awt.event.ActionEvent evt) {
    Forgot fo = new Forgot(); // Creates a new instance of the "Forgot" class, representing the forgot form.
    fo.setVisible(true);
    this.dispose();
}

private void btnloginActionPerformed(java.awt.event.ActionEvent evt) {
    // Retrieves the username and password entered by the user from the txtusername and txtpassword text fields.
    String sql = "select * from account where username=? and password=?";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, x: txtusername.getText());
        ps.setString(parameterIndex: 2, x: txtpassword.getText());
        rs = ps.executeQuery();
        if (rs.next()) {
            rs.close();
            ps.close();
            Loading lod = new Loading(); // Creates an instance of the "Loading" class, representing a loading form.
            lod.setUploading(); // Calls the setUploading() method of the loading form.
            lod.setVisible(true); // Shows the loading form.
        } else{
    
```

The screenshot shows the Apache NetBeans IDE 15 interface with the `Login.java` file open. The code has been modified to include a `ps.close()` call after the `if (rs.next())` block. A `else` block is added to show an error dialog. The `main` method is also partially visible at the bottom.

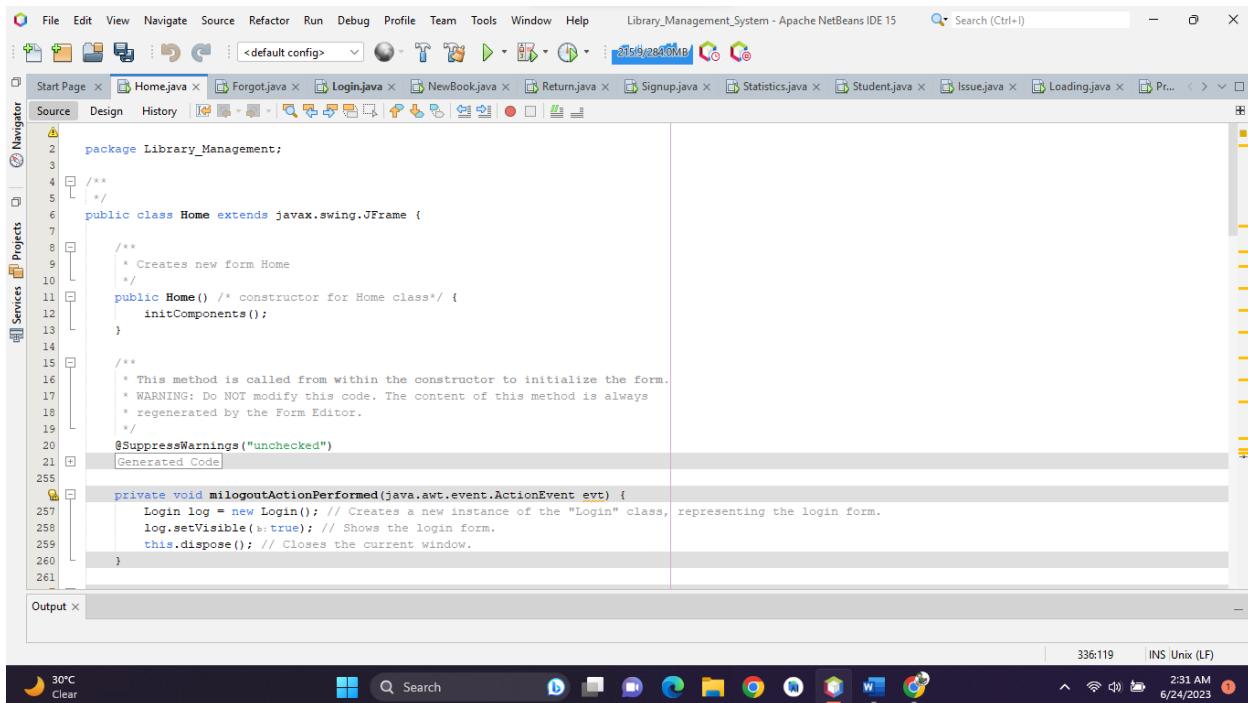
```
ps.close();
        Loading lod = new Loading(); // Creates an instance of the "Loading" class, representing a loading form.
        lod.setUploading(); // Calls the setUploading() method of the loading form.
        lod.setVisible(true); // Shows the loading form.
    } else{
        JOptionPane.showMessageDialog(parentComponent: null, message: "Incorrect Username and Password");
    }
    this.dispose();
} catch (Exception e) {
    JOptionPane.showMessageDialog(parentComponent: null, message: e); // Handles any exception.
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    // Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Login().setVisible(true); // Creates a new instance of the "Login" class that shows the login form.
        }
    });
}
// Variables declaration - do not modify

```

Home Form:



The screenshot shows the Apache NetBeans IDE 15 interface. The main window displays the source code for the `Home.java` file. The code handles various button actions:

```
private void milogoutActionPerformed(java.awt.event.ActionEvent evt) {
    Login log = new Login(); // Creates a new instance of the "Login" class,
    log.setVisible(true); // Shows the login form.
    this.dispose(); // Closes the current window.
}

private void miexitActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(status:0); // Exits the application by calling exit method.
}

private void btnnewbookActionPerformed(java.awt.event.ActionEvent evt) {
    NewBook ob = new NewBook(); // Creates a new instance of the "NewBook" class, representing the form for adding a new book.
    ob.setVisible(true);
    this.dispose();
}

private void btnnewstudentActionPerformed(java.awt.event.ActionEvent evt) {
    Student ob = new Student(); // Creates a new instance of the "Student" class, representing the form for adding a new student.
    ob.setVisible(true);
    this.dispose();
}

private void btnstatisticsActionPerformed(java.awt.event.ActionEvent evt) {
    Statistics ob = new Statistics(); // Creates a new instance of the "Statistics" class, representing the form for displaying records.
    ob.setVisible(true);
    this.dispose();
}
```

The status bar at the bottom shows the date and time as 6/24/2023, 2:32 AM.

The screenshot shows the Apache NetBeans IDE 15 interface with the same `Home.java` file open. The code now includes several more methods for handling different actions:

```
private void btnstatisticsActionPerformed(java.awt.event.ActionEvent evt) {
    Statistics ob = new Statistics(); // Creates a new instance of the "Statistics" class, representing the form for displaying records.
    ob.setVisible(true);
    this.dispose();
}

private void btnbookActionPerformed(java.awt.event.ActionEvent evt) {
    Issue ob = new Issue(); // Creates a new instance of the "Issue" class, representing the form for issuing a book.
    ob.setVisible(true);
    this.dispose();
}

private void btnrbookActionPerformed(java.awt.event.ActionEvent evt) {
    Return ob = new Return(); // Creates a new instance of the "Return" class, representing the form for returning a book.
    ob.setVisible(true);
    this.dispose();
}

private void btnprintActionPerformed(java.awt.event.ActionEvent evt) {
    Print ob = new Print(); // Creates a new instance of the "Print" class, representing the form for printing the library .
    ob.setVisible(true);
    this.dispose();
}

private void jMenu1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

The status bar at the bottom shows the date and time as 6/24/2023, 2:32 AM.

Forgot Form:



The screenshot shows the Apache NetBeans IDE interface with the "Forgot.java" file open in the Source editor. The code is as follows:

```
10  public class Forgot extends javax.swing.JFrame {
11      Connection con = null;
12      ResultSet rs = null; // Declaration and initialization of variables for database operations.
13      PreparedStatement ps = null;
14      /**
15       * Creates new form Forgot
16       */
17      public Forgot() // Constructor for the "Forgot" class.
18      {
19          initComponents();
20          con = DatabaseHelper.getConnection();
21      }
22
23      /**
24       * This method is called from within the constructor to initialize the form.
25       * WARNING: Do NOT modify this code. The content of this method is always
26       * regenerated by the Form Editor.
27       */
28      @SuppressWarnings("unchecked")
29      // Generated Code
30
31      private void btnbackActionPerformed(java.awt.event.ActionEvent evt) {
32          Login lo = new Login();
33          lo.setVisible(true);
34          this.dispose();
35      }
36
37  }
```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Library_Management_System - Apache NetBeans IDE 15 Search (Ctrl+F) 233/42840MB

Start Page | Forgot.java | NewBook.java | Return.java | Signup.java | Statistics.java | Student.java | Issue.java | Loading.java | Print.java

Source Design History

```
private void btnsearchActionPerformed(java.awt.event.ActionEvent evt) {
    search(); // Calls the search method to perform the search operation.
}

private void btnretrieveActionPerformed(java.awt.event.ActionEvent evt) {
    retrieve(); // Calls the retrieve method to perform the search operation.
}

public void search() /* This method searches the database for a user account based on the entered username */ {
    String al= txtuname.getText(); // Takes the username from text user name field and stores it in String al.
    String sql = "select * from account where username='"+al+"'"; // SQL query to select from the account table where the username matches the input.
    try {
        ps = con.prepareStatement(sql);
        rs = ps.executeQuery();
        if (rs.next()) {
            txtname.setText(rs.getString(columnIndex:2));
            txtsques.setText(rs.getString(columnIndex:4));
            rs.close();
            ps.close();
        } else {
            JOptionPane.showMessageDialog(parentComponent: null, message: "Incorrect Username"); // // If no matching record is found, it displays a "Incorrect
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: e); // Handles any exceptions that occurs during the operation.
    }
}
```

Output X

30°C Mostly clear 317:109 INS Unix (LF) 2:38 AM 6/24/2023

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Library_Management_System - Apache NetBeans IDE 15 Search (Ctrl+F) 230/2840MB

Start Page | Forgot.java | NewBook.java | Return.java | Signup.java | Statistics.java | Student.java | Issue.java | Loading.java | Print.java

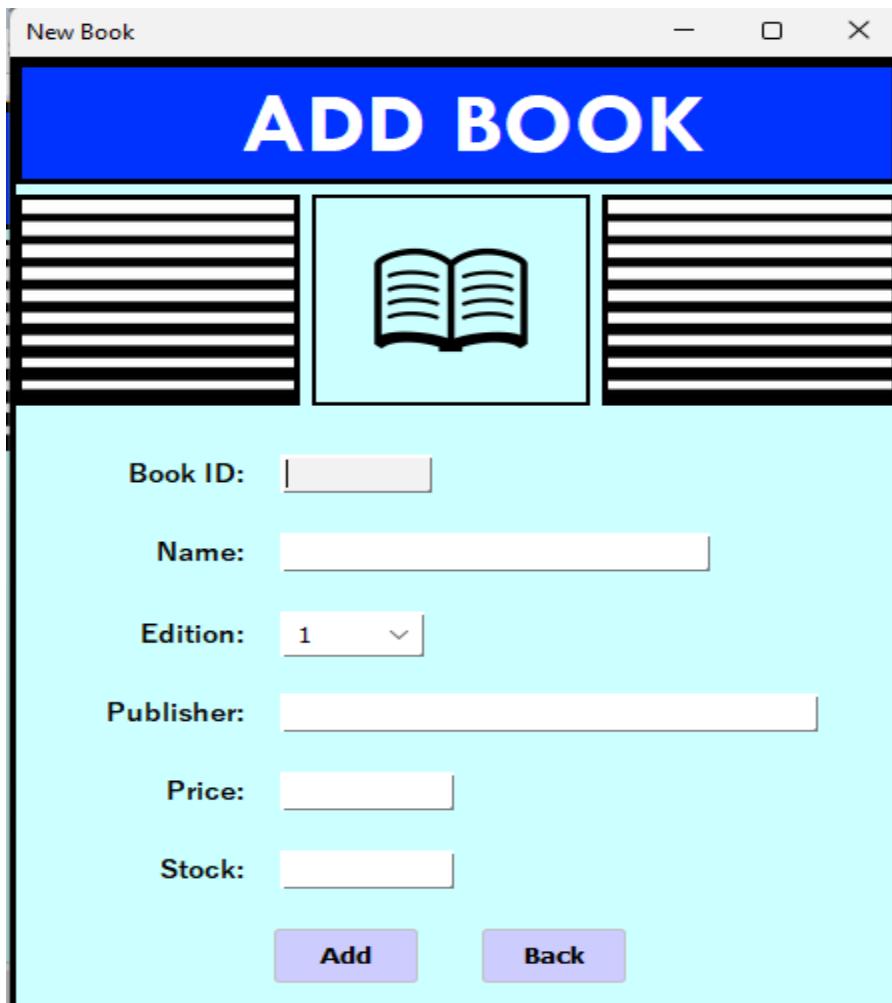
Source Design History

```
public void retrieve() // This method retrieves the password from the database based on the provided security question answer.
{
    String al = txtuname.getText();
    String a2 = txtanswer.getText();
    String sql = "select * from account where answer='"+a2+"'";
    try {
        ps = con.prepareStatement(sql);
        rs = ps.executeQuery();
        if (rs.next()) {
            txtpassword.setText(rs.getString(columnIndex:3));
            rs.close();
            ps.close();
        } else {
            JOptionPane.showMessageDialog(parentComponent: null, message: "Incorrect Answer to the security question!"); // If no matching record is found, it
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: e); // Handles any exceptions and displays an error message.
    }
}
/** 
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    LookAndFeel setting code (optional)
}
```

Output X

30°C Mostly clear 317:109 INS Unix (LF) 2:39 AM 6/24/2023

New Book Form:



The screenshot shows the Apache NetBeans IDE 15 interface. The title bar reads "Library_Management_System - Apache NetBeans IDE 15". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for file operations like Open, Save, Print, and a search bar. The Projects and Services pantries are visible on the left. The Navigator pane shows the current file is "NewBook.java". The code editor displays the following Java code:

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Library_Management_System - Apache NetBeans IDE 15 Search (Ctrl+F) 160/0/3840MB 30°C Humid 2:40 AM 6/24/2023
<default config>
Start Page x NewBook.java x Return.java x Signup.java x Statistics.java x Student.java x Issue.java x Loading.java x Print.java x
Source Design History
13 */
14 public class NewBook extends javax.swing.JFrame {
15
16     Connection con = null;
17     ResultSet rs = null;
18     PreparedStatement ps = null;
19
20     /**
21      * Creates new form NewBook
22      */
23     public NewBook() // Constructor for the NewBook class.
24     {
25         initComponents();
26         con = DatabaseHelper.getConnection();
27         Random(); // Calls the Random() method to generate a random book ID.
28     }
29
30     public void Random()
31     {
32         Random rd= new Random();
33         txtbid.setText(""+rd.nextInt(1000+1)); // Generates a random number between 0 and 1000 and sets it in txtbid field.
34     }
35
36     /**
37      * This method is called from within the constructor to initialize the form.
38      * WARNING: Do NOT modify this code. The content of this method is always
39      * regenerated by the Form Editor.
40     */
41     @SuppressWarnings("unchecked")
42     // Generated Code
43 }
```

Screenshot of Apache NetBeans IDE 15 showing the code editor for NewBook.java. The code implements action listeners for two buttons: btnback and btnadd. The btnback listener creates a new instance of the Home class and sets it visible. The btnadd listener inserts a new book record into the 'book' table using a prepared statement. The code includes imports for java.awt.event.ActionEvent, java.sql.PreparedStatement, and javax.swing.JOptionPane.

```

private void btnbackActionPerformed(java.awt.event.ActionEvent evt) {
    Home ob = new Home(); // Creates a new instance of the "Home" class, representing the home form.
    ob.setVisible(true);
    this.dispose();
}

private void btnaddActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String sql = "insert into book" // SQL query to insert a new book record into the "book" table with the provided values.
                    +"(book_id, name, edition, publisher, price, stock)"
                    +"values (?, ?, ?, ?, ?, ?)";
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, txtbid.getText());
        ps.setString(parameterIndex: 2, txtname.getText());
        ps.setString(parameterIndex: 3, (String)cmbedition.getSelectedItem());
        ps.setString(parameterIndex: 4, txtpub.getText());
        ps.setString(parameterIndex: 5, txtprice.getText());
        ps.setString(parameterIndex: 6, txtstock.getText());
        ps.execute();
        JOptionPane.showMessageDialog(null, "New book added");
        rs.close();
        ps.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
}

```

Signup Form:



The screenshot shows the Apache NetBeans IDE 15 interface. The title bar reads "Library_Management_System - Apache NetBeans IDE 15". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for file operations like Open, Save, and Build. The status bar at the bottom shows memory usage (2517/2840MB), search field, and system information (26:38, INS Unix (LF), 6/24/2023).

The code editor displays the `Signup.java` file:19 public class Signup extends javax.swing.JFrame {
20 Connection con = null;
21 ResultSet rs = null;
22 PreparedStatement ps = null;
23 /**
24 * Creates new form Signup
25 */
26 public Signup() // Calls the signup constructor
27 {
28 initComponents();
29 con = DatabaseHelper.getConnection();
30 }
31 /**
32 * This method is called from within the constructor to initialize the form.
33 * WARNING: Do NOT modify this code. The content of this method is always
34 * regenerated by the Form Editor.
35 */
36 @SuppressWarnings("unchecked")
37 Generated Code
38 private void btnCreateActionPerformed(java.awt.event.ActionEvent evt) {
39 try {
40 // SQL query to insert values into the 'account' table
41 String sql = "insert into account"
42 +"(username, name, password, security_ques, answer)"
43 +"values (?, ?, ?, ?, ?);";
44 ps = con.prepareStatement(sql);
45 ps.setString(parameterIndex, txtuname.getText());
46 ps.setString(parameterIndex, txtname.getText());
47 ps.setString(parameterIndex, txtpassword.getText());
48 ps.setString(parameterIndex, cmbques.getSelectedItem());
49 ps.setString(parameterIndex, txtans.getText());
50 ps.execute();
51 // Display a dialog box to notify that a new account has been created
52 JOptionPane.showMessageDialog(null, "New account created");
53 rs.close();
54 ps.close();
55 } catch (Exception e) {
56 JOptionPane.showMessageDialog(null, e);
57 }
58 }
59 private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
60 Login lo = new Login(); // Create a new instance of the Login class and display it.
61 lo.setVisible(true);
62 this.dispose();
63 }
64 }

This screenshot is identical to the one above, showing the `Signup.java` code in the NetBeans IDE. The code is identical, and the interface is the same, including the toolbar, status bar, and file tabs.

Loading Form:

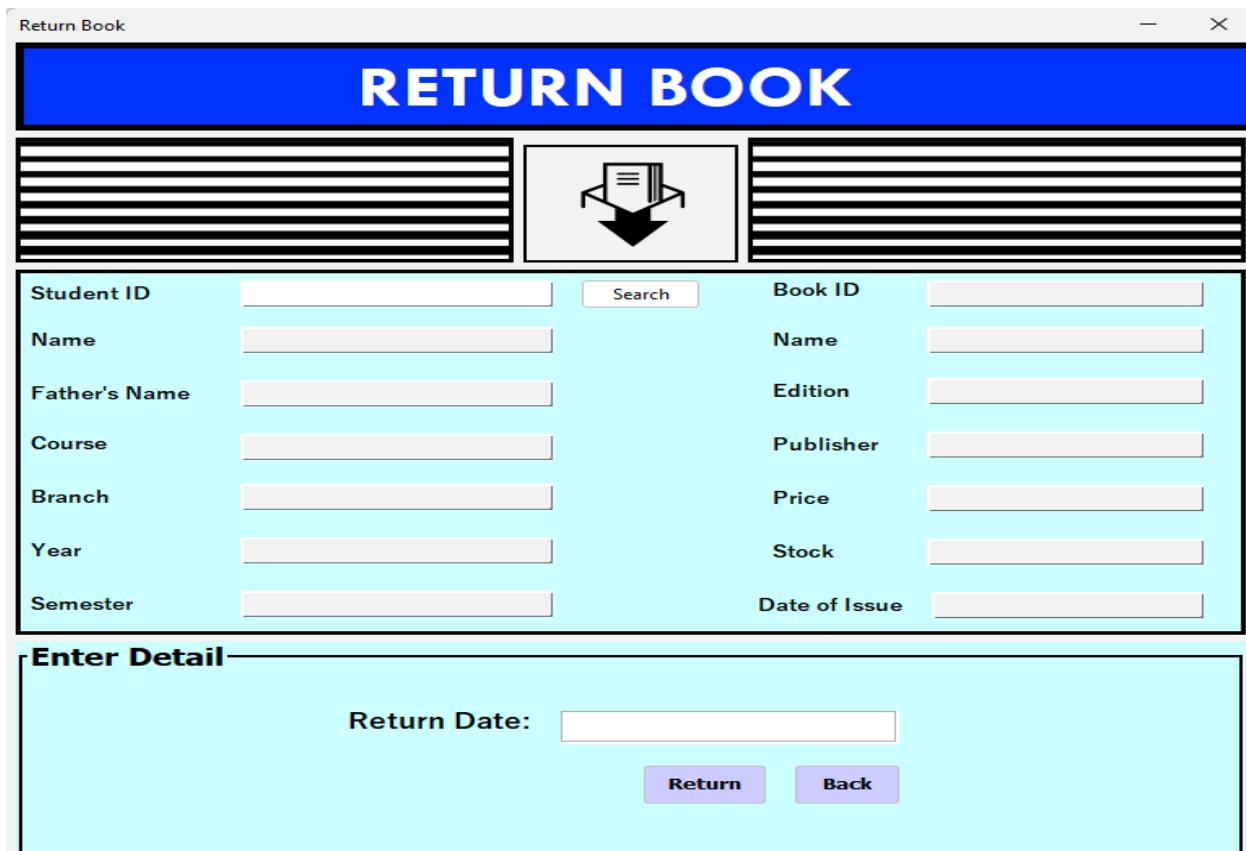


The screenshot shows the Apache NetBeans IDE 15 interface with the "Loading.java" file open in the Source editor. The code implements a loading process using a thread and a progress bar:

```
23 public Loading() {
24     initComponents();
25     con = DatabaseHelper.getConnection();
26     th = new Thread((Runnable)this); // Create a new thread for the loading process
27 }
28
29 public void setUploading(){
30     th.start();
31 }
32
33 @Override
34 public void run(){
35     try {
36         // Loop to update the progress bar
37         for (int i = 0; i <= 100; i++) {
38             int m = jProgressBar1.getMaximum();
39             int v = jProgressBar1.getValue();
40             if (v < m) {
41                 jProgressBar1.setValue(jProgressBar1.getValue() + 3);
42             }else{
43                 i=101;
44                 setVisible(false);
45                 Home ob = new Home();
46                 ob.setVisible(true);
47                 Thread.sleep(millis: 50); // Delay for smooth progress bar update
48             }
49         } catch (Exception e) {
50             JOptionPane.showMessageDialog(parentComponent: null, message: e);
51         }
52     }
53 }
```

The code includes imports for `java.awt`, `java.sql`, `java.util`, and `javax.swing` packages. The `setVisible` method is used to hide the current window and show a new `Home` window. The `Thread.sleep` method is used to delay the update of the progress bar.

Return Book Form:



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Title Bar:** Library_Management_System - Apache NetBeans IDE 15
- Toolbar:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Status Bar:** 2284/36410MB, Search (Ctrl+F), 1:1, INS Unix (LF), 2:48 AM, 6/24/2023, 30°C Mostly clear.
- Project Explorer (Services tab):** Shows a project structure with files like Return.java, Statistics.java, Student.java, Issue.java, and Print.java.
- Navigator:** Shows the current file is Return.java.
- Source Editor:** Displays the Java code for the Return class. The code includes constructor logic, form initialization code generated by the Form Editor, and an actionPerformed event handler for a button.

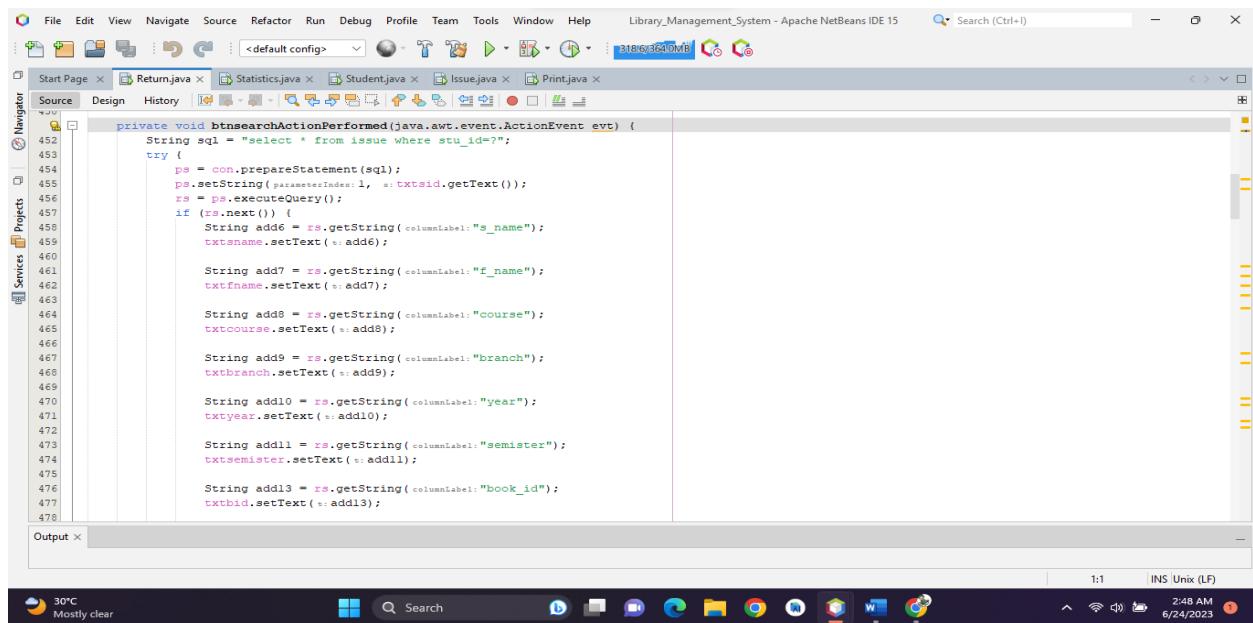
```
/*
 * 
 */
public class Return extends javax.swing.JFrame {

    Connection con = null;
    ResultSet rs = null;
    PreparedStatement ps = null;
    /**
     * Creates new form Return
     */
    public Return() { // Constructor for the Return class.
        initComponents();
        con = DatabaseHelper.getConnection();
    }

    /**
     * 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.
     */
    @SuppressWarnings("unchecked")
    // Generated Code

    private void btnbackActionPerformed(java.awt.event.ActionEvent evt) {
        Home ob = new Home(); // Creates a new instance of the "Home" class, representing the home form.
        ob.setVisible(b.setVisible);
        this.dispose();
    }
}
```

- Output:** A panel at the bottom left showing build logs or other output information.



The screenshot shows the Apache NetBeans IDE 15 interface. The main window displays a Java code editor for a file named `Statistics.java`. The code is a part of a class and contains a method `btnsearchActionPerformed` which performs a database query to retrieve student information based on a provided ID. The code uses JDBC to prepare a statement, set parameters, and execute it, then loops through the results to update text fields with student details like name, course, branch, year, semester, and book ID.

```

private void btnsearchActionPerformed(java.awt.event.ActionEvent evt) {
    String sql = "select * from issue where stu_id=?";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, :txtsid.getText());
        rs = ps.executeQuery();
        if (rs.next()) {
            String add6 = rs.getString(columnLabel: "s_name");
            txtname.setText(:add6);

            String add7 = rs.getString(columnLabel: "f_name");
            txtfname.setText(:add7);

            String add8 = rs.getString(columnLabel: "course");
            txtcourse.setText(:add8);

            String add9 = rs.getString(columnLabel: "branch");
            txtbranch.setText(:add9);

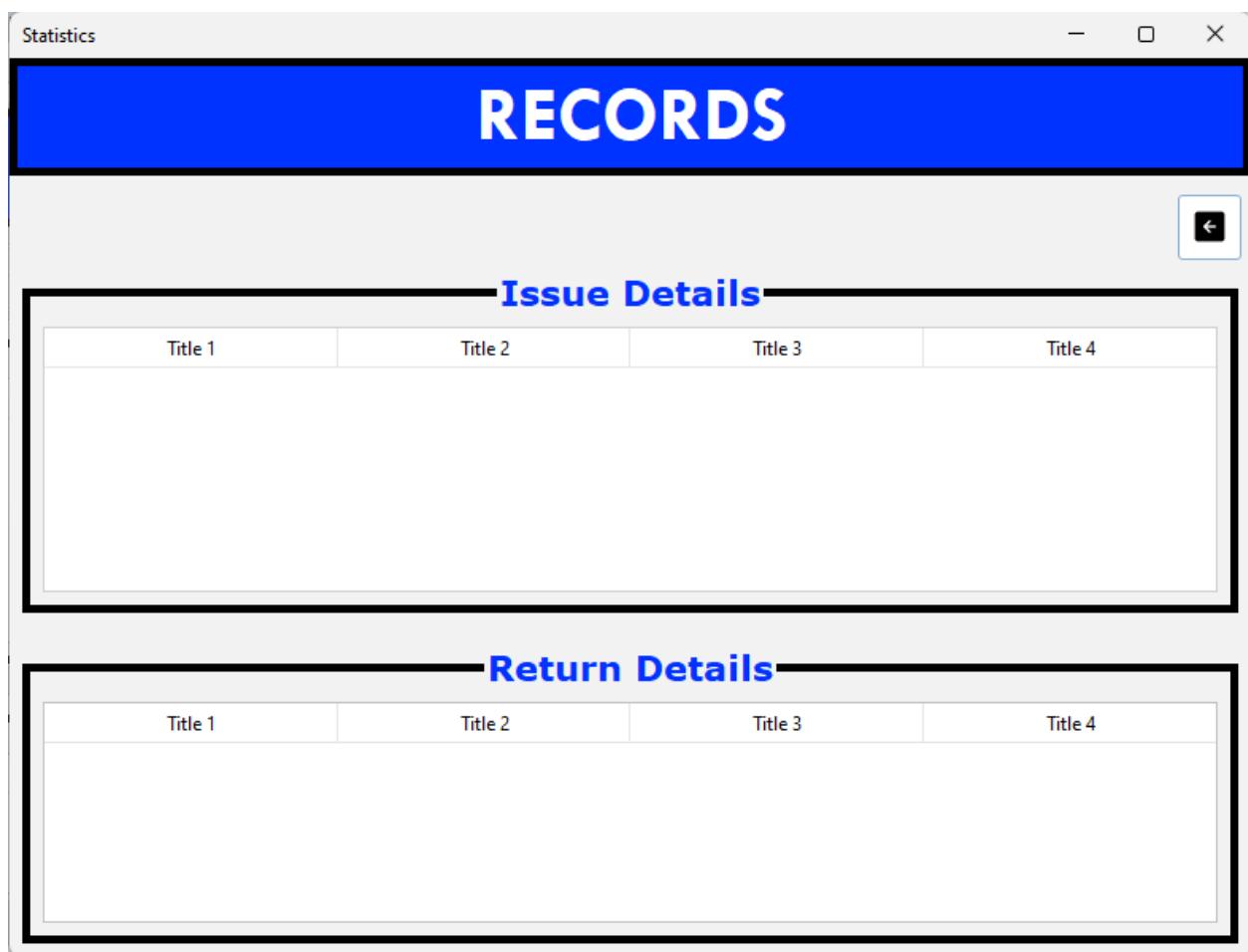
            String add10 = rs.getString(columnLabel: "year");
            txtyear.setText(:add10);

            String add11 = rs.getString(columnLabel: "semister");
            txtsemister.setText(:add11);

            String add13 = rs.getString(columnLabel: "book_id");
            txtbid.setText(:add13);
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
}

```

Statistics(Records) Form:



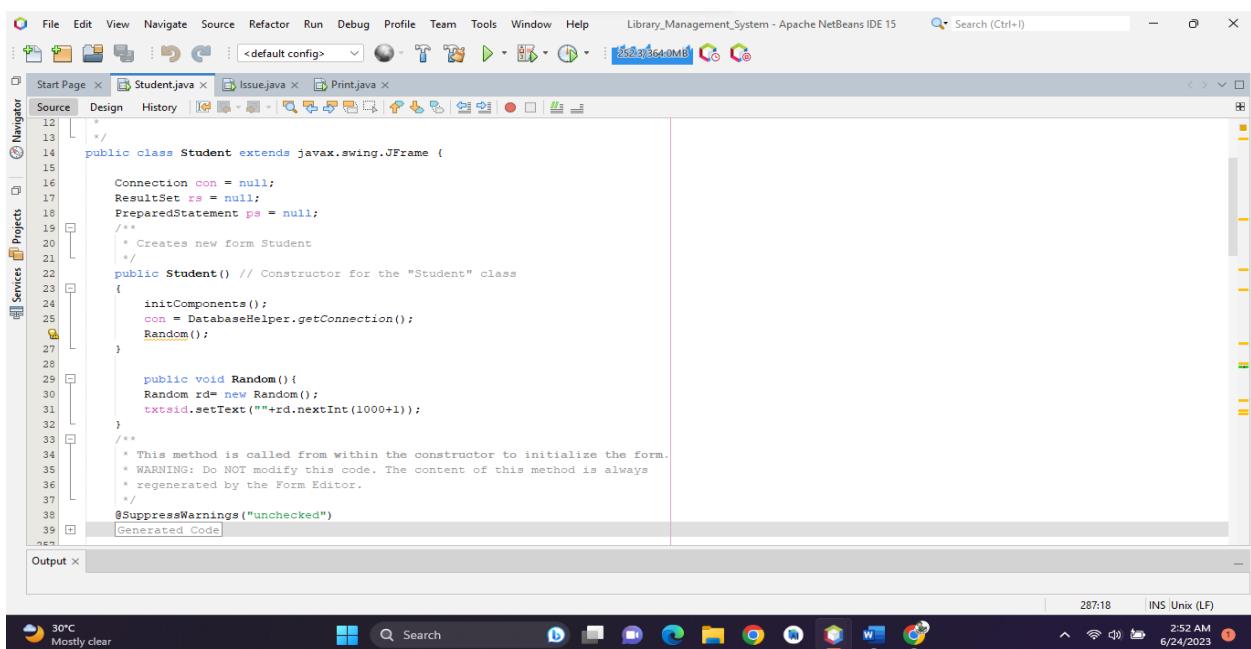
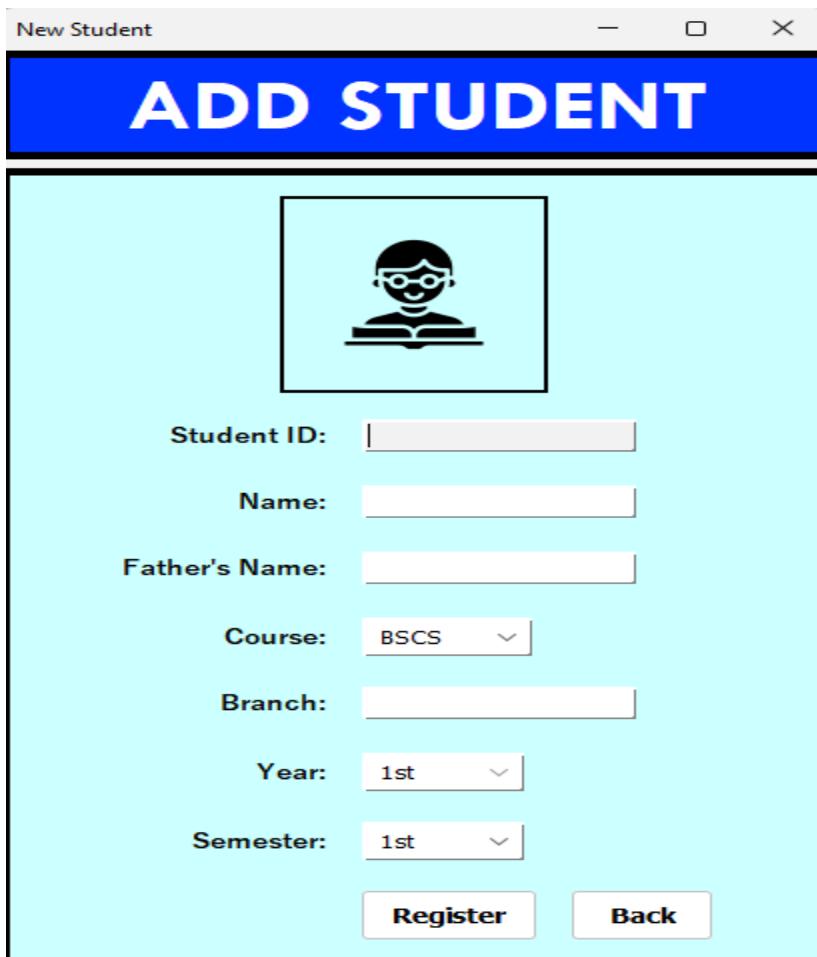
The screenshot shows the Apache NetBeans IDE 15 interface. The main window displays the Java file `Statistics.java` under the `Source` tab. The code implements a `JFrame` for displaying statistics, specifically student and book information. It uses JDBC to query a database and populate a `JTable`. The `tblissue()` method is shown in detail, demonstrating how it prepares a SQL statement, executes it, and sets the result set to a table model. The `tblreturn()` method is also partially visible. The IDE's toolbar, Navigator, Projects, and Services panes are visible on the left, and an Output pane is at the bottom.

```
16 public class Statistics extends javax.swing.JFrame {  
17     Connection con = null;  
18     ResultSet rs = null;  
19     PreparedStatement ps = null;  
20     /**  
21      * Creates new form Statistics  
22      */  
23     public Statistics() {  
24         initComponents();  
25         con = DatabaseHelper.getConnection();  
26         tblissue(); // Calls the method tblissue() in constructor to fill all the fields/table as soon as the constructor is called.  
27         tbldreturn(); // Calls the method tbldreturn() in constructor to fill all the fields/table as soon as the constructor is called.  
28     }  
29     public void tblissue(){  
30         try {  
31             String sql = "select stu_id AS 'Student ID', s_name AS 'Student Name', book_id AS 'Book ID', b_name AS 'Book Name', edition AS 'Edition', publis...  
32             ps = con.prepareStatement(sql);  
33             rs = ps.executeQuery();  
34             tblissue.setModel(DbUtils.resultSetToTableModel(rs)); // Set the result set as the model for the tblissue table  
35         } catch (Exception e) {  
36             JOptionPane.showMessageDialog(parentComponent: null, message: e);  
37         }  
38     }  
39     public void tbldreturn(){  
40         try {  
41             String sql = "select stu_id AS 'Student ID', s_name AS 'Student Name', book_id AS 'Book ID', b_name AS 'Book Name', course AS 'Course', branch AS 'Branch' from student inner join book on student.book_id = book.book_id where student.stu_id = ?";  
42             ps = con.prepareStatement(sql);  
43             ps.setInt(1, stu_id);  
44             rs = ps.executeQuery();  
45             tbldreturn.setModel(DbUtils.resultSetToTableModel(rs)); // Set the result set as the model for the tbldreturn table  
46         } catch (Exception e) {  
47             JOptionPane.showMessageDialog(parentComponent: null, message: e);  
48         }  
49     }  
50     /**  
51      * This method is called from within the constructor to initialize the form.  
52      * WARNING: Do NOT modify this code. The content of this method is always  
53      * regenerated by the Form Editor.  
54      */  
55     @SuppressWarnings("unchecked")  
56     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
57         Home ob = new Home();  
58         ob.setVisible(b: true);  
59         this.dispose();  
60     }  
61     /**  
62      * Sparam args the command line arguments  
63      */  
64     public static void main(String args[]) {  
65         /* Create and display the form */  
66         java.awt.EventQueue.invokeLater(new Runnable() {  
67             public void run() {  
68                 new Statistics().setVisible(true);  
69             }  
70         });  
71     }  
72 }
```

This screenshot shows the same Apache NetBeans IDE 15 interface as the first one, but with a different section of the `Statistics.java` code highlighted. Lines 41 through 56 are selected, showing the implementation of the `tbldreturn()` method. This method performs a similar JDBC operation to the `tblissue()` method, but for a different table and with a different primary key. The code includes a warning about not modifying the generated code. The rest of the code, including the constructor and the `jButton1ActionPerformed` event handler, is also visible.

```
41 }  
42 public void tbldreturn(){  
43     try {  
44         String sql = "select stu_id AS 'Student ID', s_name AS 'Student Name', book_id AS 'Book ID', b_name AS 'Book Name', course AS 'Course', branch AS 'Branch' from student inner join book on student.book_id = book.book_id where student.stu_id = ?";  
45         ps = con.prepareStatement(sql);  
46         ps.setInt(1, stu_id);  
47         rs = ps.executeQuery();  
48         tbldreturn.setModel(DbUtils.resultSetToTableModel(rs)); // Set the result set as the model for the tbldreturn table  
49     } catch (Exception e) {  
50         JOptionPane.showMessageDialog(parentComponent: null, message: e);  
51     }  
52 }  
53 /**  
54  * This method is called from within the constructor to initialize the form.  
55  * WARNING: Do NOT modify this code. The content of this method is always  
56  * regenerated by the Form Editor.  
57  */  
58 @SuppressWarnings("unchecked")  
59 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
60     Home ob = new Home();  
61     ob.setVisible(b: true);  
62     this.dispose();  
63 }  
64 /**  
65  * Sparam args the command line arguments  
66  */  
67 public static void main(String args[]) {  
68     /* Create and display the form */  
69     java.awt.EventQueue.invokeLater(new Runnable() {  
70         public void run() {  
71             new Statistics().setVisible(true);  
72         }  
73     });  
74 }
```

Add Student Form:



File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Library_Management_System - Apache NetBeans IDE 15 Search (Ctrl+F) 1983/3640MB

Start Page Student.java Issue.java Print.java

Source Design History

```

private void btnbackActionPerformed(java.awt.event.ActionEvent evt) {
    Home ob = new Home(); // Creates an instance of Home class and displays it.
    ob.setVisible(true);
    this.dispose();
}

private void btnregisterActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        // SQL statement for inserting data into the 'student' table
        String sql = "insert into student"
                    +"(student_id, name, fathers_name, course, branch, year,"
                    +" semester) values (?, ?, ?, ?, ?, ?)";
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, txtsid.getText());
        ps.setString(parameterIndex: 2, txtname.getText());
        ps.setString(parameterIndex: 3, txtfname.getText());
        ps.setString(parameterIndex: 4, (String) cmbcourse.getSelectedItem());
        ps.setString(parameterIndex: 5, txtbranch.getText());
        ps.setString(parameterIndex: 6, (String) cmbyear.getSelectedItem());
        ps.setString(parameterIndex: 7, (String) cmbsemister.getSelectedItem());
        ps.execute();
        JOptionPane.showMessageDialog(null, "New student registered successfully.");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    } finally {
        try {
            rs.close();
            ps.close();
        }
    }
}

```

Output 287:18 INS Unix (LF)

30°C Mostly clear 2:53 AM 6/24/2023

Issue Form:

Issue Book

ISSUE BOOK



Book Details

Book ID	<input type="text"/>	Search
Name	<input type="text"/>	
Edition	<input type="text"/>	
Publisher	<input type="text"/>	
Price	<input type="text"/>	
Stock	<input type="text"/>	

Student Details

Student ID	<input type="text"/>	Search
Name	<input type="text"/>	
Father's Name	<input type="text"/>	
Course	<input type="text"/>	
Branch	<input type="text"/>	
Year	<input type="text"/>	
Semester	<input type="text"/>	

Enter Detail

Date of Issue: Issue Back

Search

```
private void btbnbsearchActionPerformed(java.awt.event.ActionEvent evt) {
    String sql = "select * from book where book_id=?";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, txtbid.getText());
        rs = ps.executeQuery();
        if (rs.next()) {
            String add1 = rs.getString(columnLabel: "name");
            txtbname.setText(add1);

            String add2 = rs.getString(columnLabel: "edition");
            txtedition.setText(add2);

            String add3 = rs.getString(columnLabel: "publisher");
            txtpub.setText(add3);

            String add4 = rs.getString(columnLabel: "price");
            txtprice.setText(add4);

            String add5 = rs.getString(columnLabel: "stock");
            txtstock.setText(add5);
            rs.close();
            ps.close();
        } else {
            JOptionPane.showMessageDialog(parentComponent: null, message: "Book not found");
        }
    } catch (Exception e) {
    }
}
```

```
private void btnissueActionPerformed(java.awt.event.ActionEvent evt) {
    // SQL query to insert an issue record into the database
    String sql = "insert into issue(book_id, b_name, edition, publisher, price, "
    + "stock, stu_id, s_name, f_name, course, branch, year, semister, doi"
    + "values(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(parameterIndex: 1, txtbid.getText());
        ps.setString(parameterIndex: 2, txtbname.getText());
        ps.setString(parameterIndex: 3, txtedition.getText());
        ps.setString(parameterIndex: 4, txtpub.getText());
        ps.setString(parameterIndex: 5, txtprice.getText());
        ps.setString(parameterIndex: 6, txtstock.getText());
        ps.setString(parameterIndex: 7, txtsid.getText());
        ps.setString(parameterIndex: 8, txtsname.getText());
        ps.setString(parameterIndex: 9, txtfname.getText());
        ps.setString(parameterIndex: 10, txtcourse.getText());
        ps.setString(parameterIndex: 11, txtbranch.getText());
        ps.setString(parameterIndex: 12, txtyear.getText());
        ps.setString(parameterIndex: 13, txtsemister.getText());
        ps.setString(parameterIndex: 14, txtdoi.getText());
        ps.execute();
        JOptionPane.showMessageDialog(parentComponent: null, message: "Book issued");
        update(); // Call the update method to adjust the stock value
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: e);
    } finally {
    }
}
```

```
public void update() {
    // Get the current stock value and update it based on the issued quantity.
    int st = Integer.parseInt(txtstock.getText());
    int q = 1;
    int sup = st - q;
    String s = String.valueOf(st-sup);
    txtstock.setText(s);
    try {
        int n = Integer.parseInt(s);
        if (n>0) {
            String val1= txtbid.getText();
            String val2 = txtstock.getText();
            String sql = "update book set book_id='"+val1+"', "
            + "stock='"+val2+"' where book_id='"+val1+"'";
            ps = con.prepareStatement(sql);
            ps.executeUpdate();
            JOptionPane.showMessageDialog(parentComponent: null, message: "Record Updated");
        } else {
            JOptionPane.showMessageDialog(parentComponent: null, message: "Book is not issued");
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: e);
    } finally {
        try {
            rs.close();
            ps.close();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(parentComponent: null, message: e);
        }
    }
}
```

Print Form:

Print Library Card

PRINT LIBRARY CARD

Enter your Student ID:

SZABIST

Student ID:	<input type="text"/>
Name:	<input type="text"/>
Course:	<input type="text"/>
Semester:	<input type="text"/>



Apache NetBeans IDE 15 - Library_Management_System

```
23  /*
24  *  public class Print extends javax.swing.JFrame {
25  *
26  *      Connection con = null;
27  *      ResultSet rs = null;
28  *      PreparedStatement ps = null;
29  *
30  *      /**
31  *       * Creates new form Print
32  */
33  *      public Print() {
34  *          initComponents();
35  *          con = DatabaseHelper.getConnection();
36  *      }
37  *
38  *      /**
39  *       * This method is called from within the constructor to initialize the form.
40  *       * WARNING! Do NOT modify this code. The content of this method is always
41  *       * regenerated by the Form Editor.
42  */
43  *     @SuppressWarnings("unchecked")
44  *     Generated Code
45  */
46  *
47  *     private void jSearchIDActionPerformed(java.awt.event.ActionEvent evt) {
48  *         // TODO add your handling code here:
49  *     }
50  *
51  * }
```

Output x

271:39 | INS Unix (LF)

3:01 AM 6/24/2023

Apache NetBeans IDE 15 - Library_Management_System

```
268 /**
269 * private void jSearchButtonActionPerformed(java.awt.event.ActionEvent evt) {
270 *     // TODO add your handling code here:
271 *     // SQL query to retrieve student information
272 *     String sql = "select * from student where student_id=?";
273 *     try {
274 *         ps = con.prepareStatement(sql);
275 *         ps.setString(parameterIndex: 1, :jSearchID.getText());
276 *         rs = ps.executeQuery();
277 *         if (rs.next()) {
278 *             String add6 = rs.getString(columnLabel: "student_id");
279 *             jTextFieldStdID.setText(:add6);
280 *
281 *             String add5 = rs.getString(columnLabel: "name");
282 *             jTextFieldName.setText(:add5);
283 *
284 *             String add3 = rs.getString(columnLabel: "course");
285 *             jTextFieldCourse.setText(:add3);
286 *
287 *             String add11 = rs.getString(columnLabel: "semister");
288 *             jTextFieldSemester.setText(:add11);
289 *             rs.close();
290 *             ps.close();
291 *         } else {
292 *             JOptionPane.showMessageDialog(parentComponent: null, message: "Student ID not found");
293 *         }
294 *     } catch (Exception e) {
295 *         e.printStackTrace();
296 *     }
297 * }
```

Output x

271:39 | INS Unix (LF)

3:01 AM 6/24/2023

Apache NetBeans IDE 15 - Library_Management_System

```
307 /**
308 * private void jPrintButtonActionPerformed(java.awt.event.ActionEvent evt) {
309 *     // TODO add your handling code here:
310 *     Toolkit tkp = jCardPanel.getToolkit();
311 *     PrintJob pjp = tkp.getPrintJob(frame: this, jobTitle: null, props: null);
312 *     Graphics g = pjp.getGraphics();
313 *     jCardPanel.printAll(g);
314 *     g.dispose();
315 *     pjp.end();
316 * }
317 *
318 * private void jAttachButtonActionPerformed(java.awt.event.ActionEvent evt) {
319 *     JFileChooser chooser = new JFileChooser();
320 *     chooser.showOpenDialog(parent: null);
321 *     File f = chooser.getSelectedFile();
322 *     String path = f.getAbsolutePath();
323 *
324 *     // Load the original image
325 *     ImageIcon originalIcon = new ImageIcon(filename: path);
326 *     Image originalImage = originalIcon.getImage();
327 *
328 *     // Resize the image to the desired dimensions
329 *     Image resizedImage = originalImage.getScaledInstance(width: 130, height: 130, hints: Image.SCALE_SMOOTH);
330 *
331 *     // Create a new ImageIcon from the resized image
332 *     ImageIcon resizedIcon = new ImageIcon(image: resizedImage);
333 *
334 *     jLabel19.setIcon(item: resizedIcon);
335 * }
```

Output x

271:39 | INS Unix (LF)

3:01 AM 6/24/2023

CONCLUSION

The Library Management System project successfully applies object-oriented programming principles to create an efficient and user-friendly system for managing library operations. It incorporates key OOP concepts, offers essential features, prioritizes data security, and provides a valuable learning experience. With potential for future enhancements, the system contributes to improved library management and user experience. The modular design and user-friendly interface of the Library Management System makes it easy for users to perform library related tasks.

THANK YOU
