

Project Report of
University Library Management System
Enterprise Application Development – 1

By

Fernando L C D

IAODSE231F-001

14.02.2025

Table of Contents

Motivation	2
Objectives	3
Development Tools	4
Graphical User Interface (GUI).....	5
Database Design	16
Table Definitions.....	16
Connection Diagram	17
User Input Validation.....	18
Challenges	20
Future Directions	21
Appendix	22
login.java	22
home.java	23
newstudent.java	25
studentdetails.java.....	29
newbook.java	31
bookdetails.java.....	35
issubook.java	37
returnbook.java.....	39
statistics.java.....	41

Motivation

The primary motivation for developing the Library Management System is to improve the efficiency and organization of library operations. Library serve as essential hubs for knowledge and information, and it is crucial to have a system that ensures smooth book transactions, accurate record-keeping, and easy access to library resources.

This will enable librarians to efficiently manage book inventory, track issued and returned books, and maintain up-to-date student records. By automating these processes, the system minimizes human errors, reduces paperwork, and enhances the overall user experience for both librarians and students. Additionally, the system provides a centralized database, ensuring that book availability, due dates, and borrower details are easily accessible.

With the implementation of this system, the goal is to create a seamless work-flow that allows librarians to focus on improving library services while ensuring that book management remains organized and efficient.

Objectives

- **User friendly interface**
 - To design an intuitive and accessible interface that allows employees to navigate the system easily, ensuring minimal training time.
- **CRUD operations**
 - To enable the librarian to perform Create, Read, Update, and Delete operations on book entries allowing them to have track of book details dynamically.
- **Book detail accessibility**
 - To provide detailed book description and images, aiding librarians in assisting students more efficiently.
- **Report capabilities**
 - To facilitate the generation of reports about issuing and burrowing of books.
- **Secure login**
 - To keep data safe, only authorized users will be give the username and the password to login to the system. That will ensure that the details of book transactions stay protected.

Development Tools

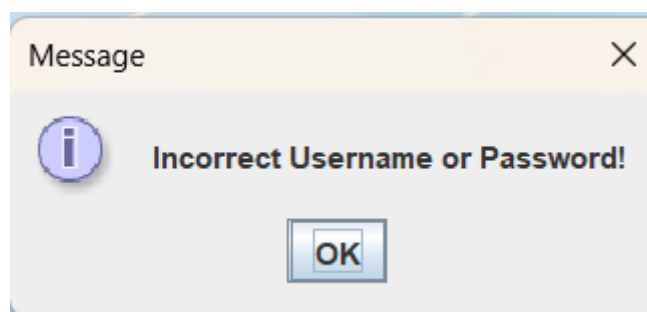
- **Programming Language: JAVA**
 - **Version:** JDK 22
 - **Artifact ID:** jdk
 - **Group ID:**
- **Integrated Development Environment (IDE): NetBeans**
 - **Version:** IDE 23
 - **Artifact ID:** netbeans
 - **Group ID:**
- **Database Management System (DBMS): MySQL(xampp)**
 - **PHP**
 - **Version:** XAMPP v3.3.0
 - **Artifact ID:** php
 - **Group ID:**
 - **MySQL**
 - **Version:** MySQL 9.1.0
 - **Artifact ID:** mysql-connector-java-9.1.0
 - **Group ID:**
- **User Interface Framework: Swing**
 - **Version:** JDK 22
 - **Artifact ID:**
 - **Group ID:** javax.swing
- **Reporting Tool: Jaspersoft Studio**
 - **Version:** Jaspersoft Studio 7.0.0
 - **Artifact ID:** jasperreports-7.0.1
 - **Group ID:** net.sf.jasperreports

Graphical User Interface (GUI)

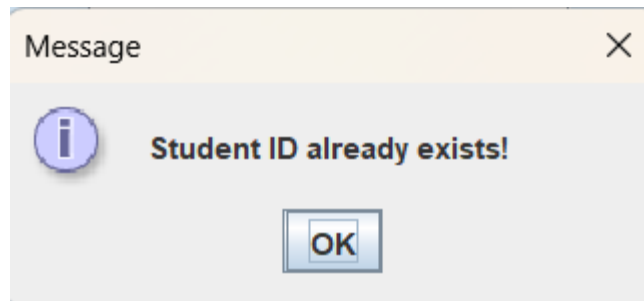
- **Login Form**



- **Exception**
 - **Incorrect Username or Password**



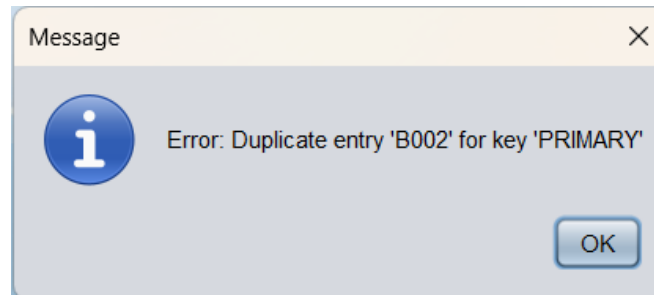
- Student ID already existing



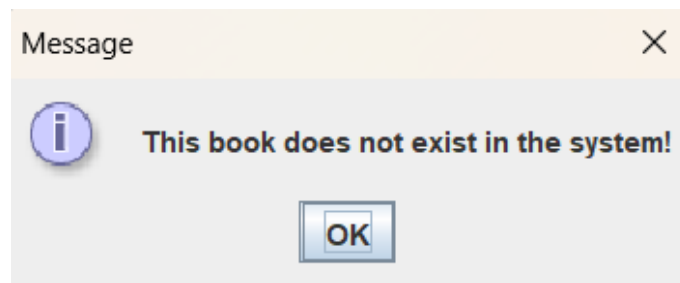
- Student ID not existing (searching the student)

A screenshot of a web application interface for adding a new student. The background is light blue with white wavy lines. At the top, the text 'Add New Student' is displayed in large, bold, dark blue font. Below this, there are four labels on the left: 'Student ID', 'Name', 'Contact Number', and 'Course Name'. To the right of 'Student ID' is a text input field containing '1010'. To the right of 'Name' is an empty text input field. To the right of 'Contact Number' is an empty text input field. To the right of 'Course Name' is an empty text input field. Below these fields is a label 'Branch Name' followed by a dropdown menu showing 'CO'. To the right of the 'Student ID' field is a blue button labeled 'Search'. At the bottom, there are four blue buttons: 'Save', 'Update', 'Delete', and 'Close'. A message dialog box is overlaid on the form. The dialog box has a yellow title bar with 'Message' and a close button (X). The main area is light gray and contains a purple circular icon with a white 'i' on the left. To the right of the icon, the text 'This student does not exist in the system!' is displayed in bold. At the bottom center, there is a blue button with the text 'OK'.

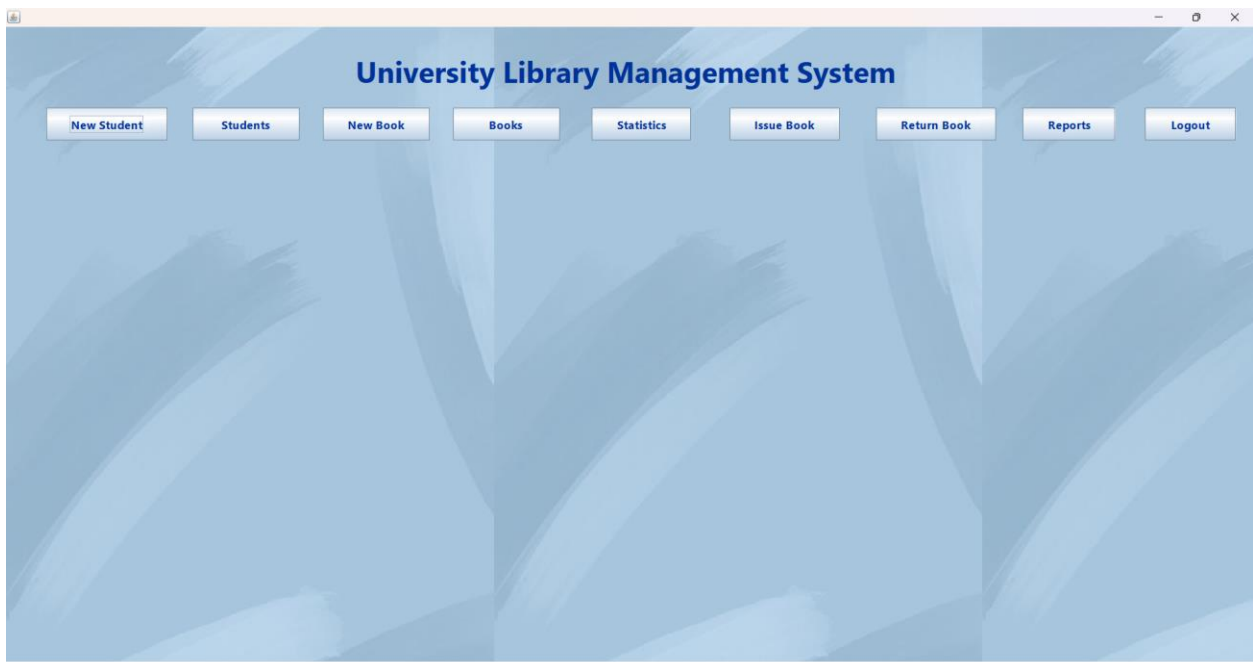
- **Book ID already existing**



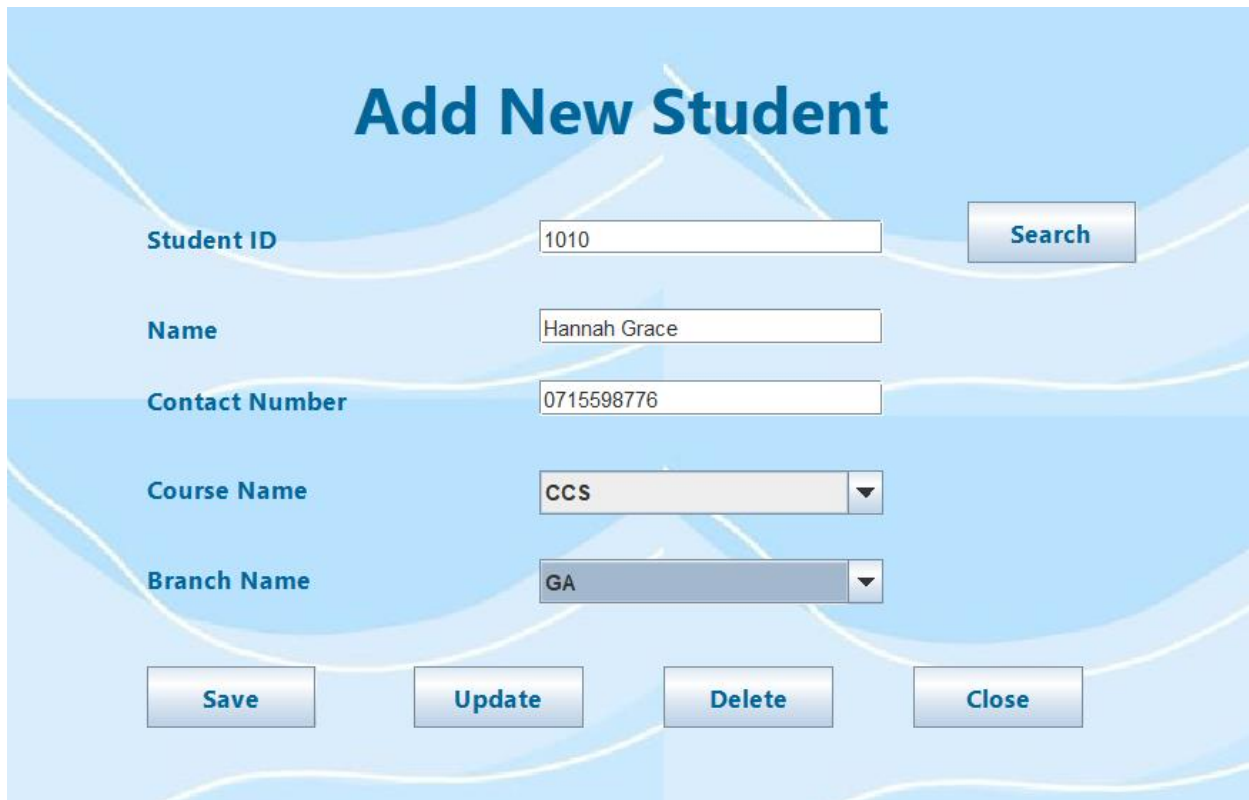
- **Book ID not existing (searching the book)**



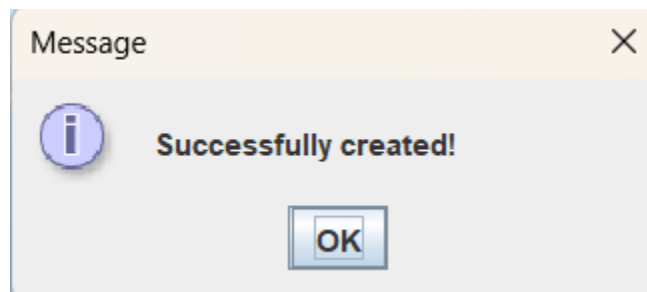
- **Dashboard**



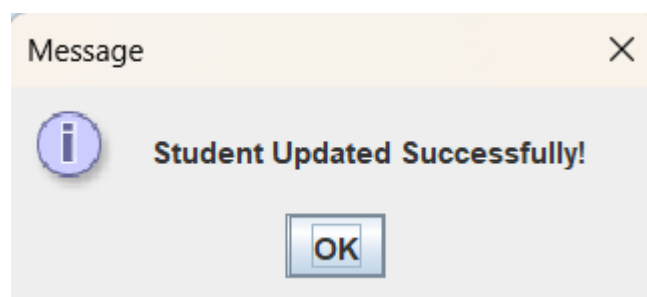
- **Add New Student**



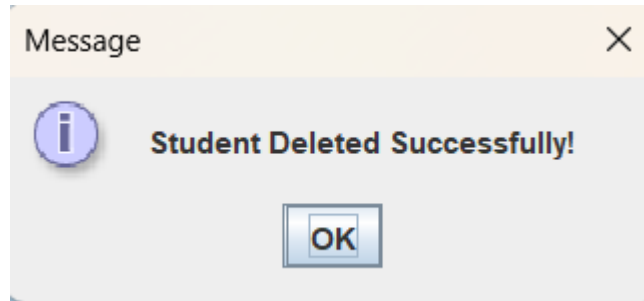
The 'Add New Student' form features a light blue background with white wavy lines. At the top center, the title 'Add New Student' is displayed in a large, bold, dark blue font. Below the title, there are five input fields arranged vertically on the left, each with a corresponding label in bold blue text: 'Student ID' (with the value '1010'), 'Name' (with the value 'Hannah Grace'), 'Contact Number' (with the value '0715598776'), 'Course Name' (a dropdown menu showing 'CCS'), and 'Branch Name' (a dropdown menu showing 'GA'). To the right of the 'Student ID' field is a 'Search' button. At the bottom of the form, there are four buttons: 'Save', 'Update', 'Delete', and 'Close', all in a light blue box with dark blue text.



- **Update**



- Delete



- View Students

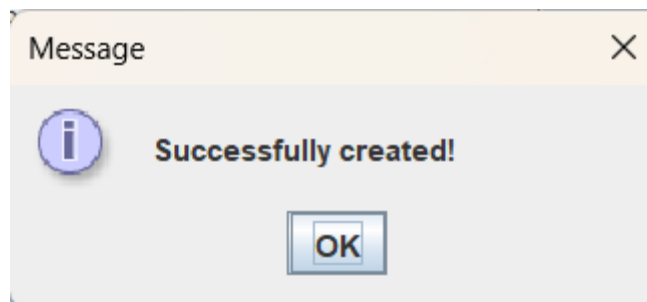
Student Details

Student ID	Student Name	Student Contact	Student Course	Student Branch
1001	Chanulya Fernando	775676552	DSE	CO
1002	June Obrid	779638495	CSE	PE
1003	Alice Johnson	779684359	CCS	PE
1004	Bob Smith	719685229	HNDIS	KU
1005	Charlie Brown	719866478	DSE	PE
1006	David Wilson	778964487	BSE	GA
1007	Emma Davis	726588934	BSE	GA
1008	Frank Martin	779521669	DCSD	CO
1009	Grace Lee	779684435	BIS	MT

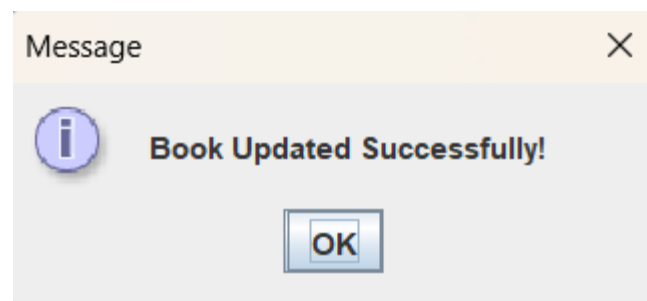
- **Add New Book**



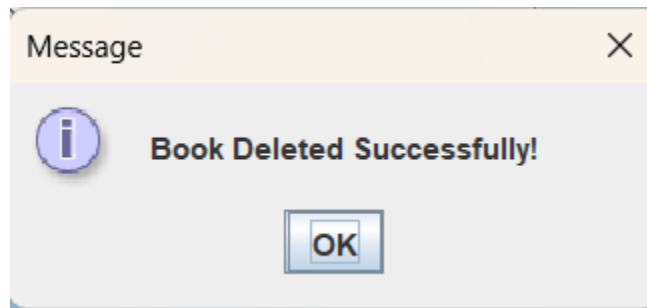
The 'Add New Book' form is displayed on a light blue background with white wavy lines. It features a title 'Add New Book' at the top center. Below the title, there are five input fields with labels to their left: 'Book ID' (containing 'B010'), 'Book Name' (containing 'Power and Prediction'), 'Author' (containing '/ Agrawal, Joshua Gans, and Avi Goldfarb'), 'Publisher' (containing 'Harvard Business Review Press'), and 'Published Year' (containing '2022'). To the right of the 'Book ID' field is a 'Search' button. At the bottom of the form, there are four buttons: 'Save', 'Update', 'Delete', and 'Close'.



- **Update**



- **Delete**



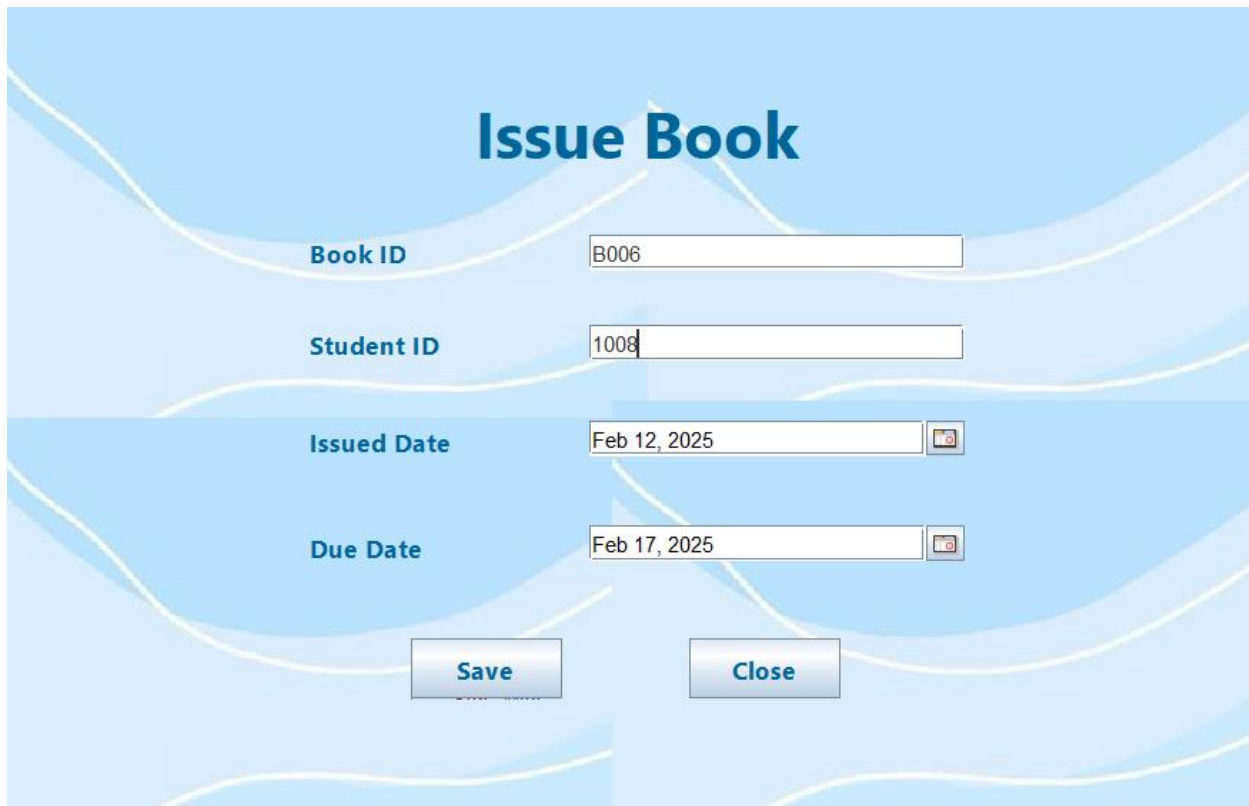
- **View Book Details**

Book Details

Book ID	Book Name	Author	Publisher	Published Year
B001	Computing Technology a	Emanuelle Burton, Judy ..	The MIT Press	2023
B002	More Than a Glitch	Meredith Broussard	The MIT Press	2023
B003	Working with AI	Thomas H. Davenport a...	The MIT Press	2022
B004	Practicing Trustworthy ...	Yada Pruksachatkun, M...	O Reilly Media	2023
B005	AI at the Edge	Daniel Situnayake and J...	O Reilly Media	2023
B006	Digitalization of Financial	Jamil Mina, Armin Ward...	O Reilly Media	2023
B007	I, Human	Tomas Chamorro-Prem...	Harvard Business Revie...	2023
B008	The Black Technical Obj...	Ramon Amaro	Sternberg Press	2023

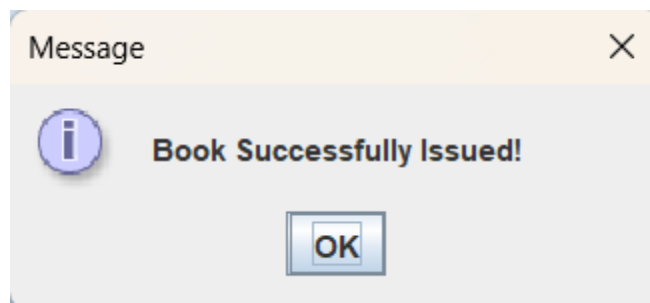
Close

- **Issue Book**




The 'Issue Book' form is displayed on a light blue background with white wavy lines. It contains four input fields with labels to their left: 'Book ID' with the value 'B006', 'Student ID' with the value '1008', 'Issued Date' with the value 'Feb 12, 2025', and 'Due Date' with the value 'Feb 17, 2025'. Each date field has a small calendar icon to its right. At the bottom of the form are two buttons: 'Save' and 'Close'.

Field	Value
Book ID	B006
Student ID	1008
Issued Date	Feb 12, 2025
Due Date	Feb 17, 2025

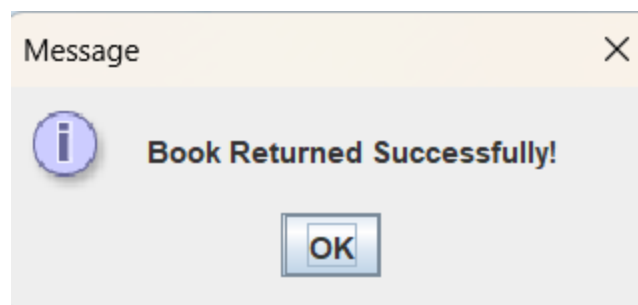


- **Return Book**



The 'Return Book' form is displayed on a light blue background with white wavy lines. It contains four input fields for data entry: 'Book ID' with the value 'B003', 'Student ID' with the value '1004', 'Issue Date' with the value '12-02-2025', and 'Due Date' with the value '13-02-2025'. A 'Search' button is located to the right of the 'Student ID' field. At the bottom of the form are two buttons: 'Return' and 'Close'.

Field	Value
Book ID	B003
Student ID	1004
Issue Date	12-02-2025
Due Date	13-02-2025



- View Statistics

Issued Book Details

Student ID	Student Name	Book ID	Book Name	Issue Date	Due Date
1005	Charlie Brown	B008	The Black Tec...	12-02-2025	14-02-2025
1008	Frank Martin	B006	Digitalization ...	12-02-2025	17-02-2025

Returned Book Details

Student ID	Student Name	Book ID	Book Name	Issue Date	Due Date
1001	Chanulya Fern...	B001	Computing Te...	09-02-2025	16-02-2025
1004	Bob Smith	B003	Working with ...	12-02-2025	13-02-2025

Close

○ **Create Report**

University Library

Wednesday 12 February

Student ID	Issued Date	Due Date	Return Status
B008			
1005	12-02-2025	14-02-2025	NO
B006			
1008	12-02-2025	17-02-2025	NO
B003			
1004	12-02-2025	13-02-2025	YES
B001			
1001	09-02-2025	16-02-2025	YES
1009	18-02-2025	20-02-2025	NO

Wednesday 12 February 2025

Page 1 of 1

Database Design

Table Definitions

- **Table student**

```
CREATE TABLE `student` (  
  
    `stid` varchar(10) PRIMARY KEY NOT NULL,  
  
    `stname` varchar(255) NOT NULL,  
  
    `stcontact` int(10) NOT NULL,  
  
    `stcourse` varchar(255) NOT NULL,  
  
    `stbranch` varchar(255) NOT NULL  
  
)
```

- stid (PRIMARY KEY) – to uniquely identify the students

- **Table book**

```
CREATE TABLE `book` (  
  
    `bid` varchar(10) PRIMARY KEY NOT NULL,  
  
    `bname` varchar(255) NOT NULL,  
  
    `author` varchar(255) NOT NULL,  
  
    `publisher` varchar(255) NOT NULL,  
  
    `publishedyear` varchar(5) NOT NULL  
  
)
```

- bid (PRIMARY KEY) – to uniquely identify the books in the library

- **Table issue**

```
CREATE TABLE `issue` (

  `bid` varchar(10) NOT NULL,

  `stid` varchar(10) NOT NULL,

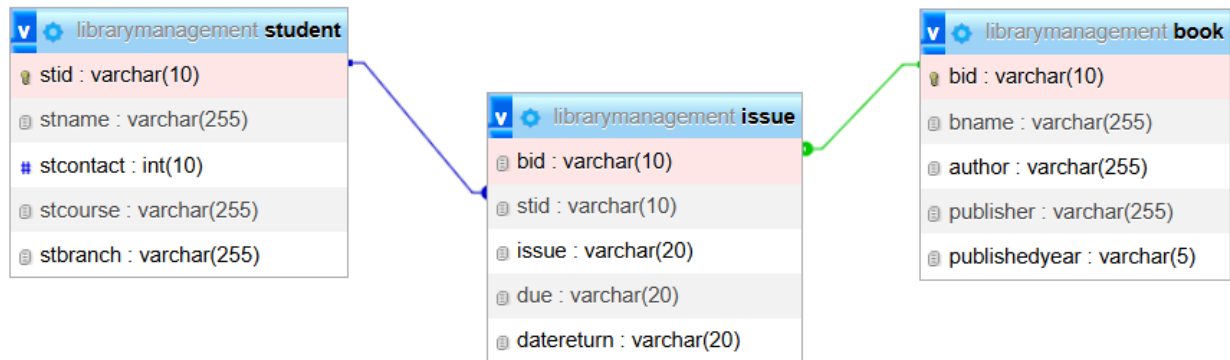
  `issue` varchar(20) NOT NULL,

  `due` varchar(20) NOT NULL,

  `datereturn` varchar(20) DEFAULT NULL

)
```

Connection Diagram



User Input Validation

- **Username and password validation**
 - **Non-empty check**
 - Ensure both username and password fields are not left empty on the login screen.
 - **Similarity check**
 - Check if the username and the password are similar to the username and the password saved in the system.
- **Student detail validation**
 - **Non-empty check**
 - Ensure all the fields are filled and none are left blank.
 - **Uniqueness check**
 - Check if the student ID is unique.
 - **Format check**
 - Check if the data entered are in the format mentioned in the database.
- **Book detail validation**
 - **Non-empty check**
 - Ensure all the fields are filled and none are left blank.
 - **Uniqueness check**
 - Check if the Book ID is unique.
 - **Format check**
 - Check if the data entered are in the format mentioned in the database.
- **Book issue validation**
 - **Data existence check**
 - Check if the data entered (student ID and book ID) already exist in the system.
 - **Non-empty check**
 - Make sure none of the fields are left empty.

- **Book return validation**
 - When a book is returned the return status of the book will be changed from 'NO' to 'YES' in the database.
- **Error messaging**
 - Provide real-time feedback for invalid inputs.
 - Display specific error messages in JOptionPane dialogs.

Challenges

- **Database Connection**
 - Getting JAVA and MySQL to connect correctly.
- **User Input Checks**
 - Validating updates.
- **User-friendly Interface**
 - Making simple, easy-to-use interfaces in JAVA Swing required careful design.
- **Generating Reports**
 - Setting up and connecting reports using Jaspersoft Studio took time.
- **Data Updates**
 - Ensuring accurate data updates.
- **Debugging**
 - Fixing issues with database and report connection and input errors required a lot of testing.
- **Learning New Tools**
 - Getting familiar with tools like Jaspersoft Studio and working NetBeans connections took extra effort.

Future Directions

- **Cloud Storage**
 - Shift to cloud storage for secure, remote access to library data.
- **User access levels**
 - Define different user roles for added security, limiting access based on role.

Appendix

login.java

```
import librarymanagementsystem.home;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
private void btnloginActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(txtuser.getText().equals("admin") && txtpswd.getText().equals("admin123"))
    {
        setVisible(false);
        new home().setVisible(true);
    }
    else
    {
        JOptionPane.showMessageDialog(null,"Incorrect Username or Password!");
    }
}

private void btncloseActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    System.exit(0);
}
```

home.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperReport;
import net.sf.jasperreports.engine.util.JRLoader;
import net.sf.jasperreports.view.JasperViewer;
private void btnnewbookActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new newbook().setVisible(true);
}

private void btnlogoutActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    setVisible(false);
    new login().setVisible(true);
}

private void btnnewstudentActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new newstudent().setVisible(true);
}

private void btnissuebookActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new issuebook().setVisible(true);
}

private void btnreturnbookActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new returnbook().setVisible(true);
}

private void lblstudentsActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new studentdetails().setVisible(true);
}
```



```

private void lblbooksActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new bookdetails().setVisible(true);
}

private void btnstatisticsActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new statistics().setVisible(true);
}

private void btnreportsActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try
    {
        String reportfile = "C:\\Users\\Chanulya Fernando\\Desktop\\SE\\EAD1\\CW
Practice\\LibraryManagementSystem\\src\\librarymanagementsystem\\rptnew.jasper";

        JasperReport jasperReport = (JasperReport) JRLoader.loadObjectFromFile(reportfile);

        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";

        Connection con = DriverManager.getConnection(url, user, pswd);

        JasperPrint jasperPrint = JasperFillManager.fillReport(jasperReport, null, con);

        JasperViewer.viewReport(jasperPrint, false);
    }
    catch (SQLException ex)
    {
        JOptionPane.showMessageDialog(this, "Database connection error: " +
ex.getMessage());
        ex.printStackTrace();
    }
    catch (JRException ex)
    {
        JOptionPane.showMessageDialog(this, "Error generating report: " + ex.getMessage());
        ex.printStackTrace();
    }
}

```

newstudent.java

```
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    String stid = txtid.getText();
    String stname = txtname.getText();
    String stcontact = txtcontact.getText();
    String stcourse = (String)cmbcourse.getSelectedItem();
    String stbranch = (String)cmbbranch.getSelectedItem();

    try {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException ex) {
            JOptionPane.showMessageDialog(null,"Class not found!");
            setVisible(false);
            new newstudent().setVisible(true);
        }
        // TODO add your handling code here:
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";

        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();

        stmt.execute("INSERT INTO student
VALUES('"+stid+"','"+stname+"','"+stcontact+"','"+stcourse+"','"+stbranch+"')");
        JOptionPane.showMessageDialog(null,"Successfully created!");
        setVisible(false);
        new newstudent().setVisible(true);

    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Student ID already exists!");
        setVisible(false);
        new newstudent().setVisible(true);
    }
}
```

```

private void btnCloseActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.dispose();
}

private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String stid = txtid.getText();
    String stname = txtname.getText();
    String stcontact = txtcontact.getText();
    String stcourse = (String)cmbcourse.getSelectedItem();
    String stbranch = (String)cmbbranch.getSelectedItem();

    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();
        stmt.executeUpdate("UPDATE student SET stname = '"+stname+"',stcontact = 
''+stcontact+'',stcourse = '"+stcourse+"',stbranch = '"+stbranch+"' WHERE stid = '"+stid+"'");
        JOptionPane.showMessageDialog(null,"Student Updated Successfully!");
        setVisible(false);
        new newstudent().setVisible(true);
    }
    catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
        setVisible(false);
        new newstudent().setVisible(true);
    }
}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String stid = txtid.getText();

    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);
    }
}

```

```

        Statement stmt = con.createStatement();
        stmt.executeUpdate("DELETE FROM student WHERE stid = '"+stid+"'");
        JOptionPane.showMessageDialog(null,"Student Deleted Successfully!");
        setVisible(false);
        new newstudent().setVisible(true);
    }
    catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
        setVisible(false);
        new newstudent().setVisible(true);
    }
}

private void btnsearchActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String stid = txtid.getText();

    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM student WHERE stid = '"+stid+"'");

        if(rs.next())
        {
            txtname.setText(rs.getString(2));
            txtcontact.setText(rs.getString(3));
            cmbcourse.setSelectedItem(rs.getString(4));
            cmbbranch.setSelectedItem(rs.getString(5));
            //txtid.setEditable(false);
        }
        else
        {
            JOptionPane.showMessageDialog(null,"This student does not exist in the system!");
            setVisible(false);
            new newstudent().setVisible(true);
        }
    } catch (SQLException ex) {

```

```
JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());  
setVisible(false);  
new newstudent().setVisible(true);  
}  
}
```

studentdetails.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
private void formComponentShown(java.awt.event.ComponentEvent evt) {
    // TODO add your handling code here:
    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();

        ResultSet rs1 = stmt.executeQuery("SELECT * FROM student");

        DefaultTableModel model1 = new DefaultTableModel();
        model1.setColumnIdentifiers(new String[]{"Student ID", "Student Name", "Student
Contact", "Student Course", "Student Branch"});

        while (rs1.next()) {
            model1.addRow(new Object[]{
                rs1.getString("stid"),
                rs1.getString("stname"),
                rs1.getString("stcontact"),
                rs1.getString("stcourse"),
                rs1.getString("stbranch")
            });
        }
        tblstudent.setModel(model1);
    }
    catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
        setVisible(false);
        new returnbook().setVisible(true);
    }
}

private void btncloseActionPerformed(java.awt.event.ActionEvent evt) {
```

```
// TODO add your handling code here:  
this.dispose();  
}
```

newbook.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String bid = txtbid.getText();
    String bname = txtbname.getText();
    String author = txtbauthor.getText();
    String publisher = txtbpublisher.getText();
    String publishedyear = txtbpubyear.getText();

    try {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException ex) {
            JOptionPane.showMessageDialog(null, "Class not found!");
            setVisible(false);
            new newbook().setVisible(true);
        }
        // TODO add your handling code here:
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";

        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();

        stmt.execute("INSERT INTO book
VALUES('"+bid+"','"+bname+"','"+author+"','"+publisher+"','"+publishedyear+"')");
        JOptionPane.showMessageDialog(null, "Successfully created!");
        setVisible(false);
        new newbook().setVisible(true);

    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Error: "+ ex.getMessage());
        setVisible(false);
        new newbook().setVisible(true);
    }
}
```



```

    }

    private void btncloseActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.dispose();
    }

    private void btnsearchActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String bid = txtbid.getText();

        try {
            String url = "jdbc:mysql://localhost:3306/librarymanagement";
            String user = "root";
            String pswd = "";
            Connection con = DriverManager.getConnection(url, user, pswd);

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM book WHERE bid = '"+bid+"'");

            if(rs.next())
            {
                txtbname.setText(rs.getString(2));
                txtbauthor.setText(rs.getString(3));
                txtbpublisher.setText(rs.getString(4));
                txtbpubyear.setText(rs.getString(5));
                //txtbid.setEditable(false);
            }
            else
            {
                JOptionPane.showMessageDialog(null,"This book does not exist in the system!");
                setVisible(false);
                new newbook().setVisible(true);
            }
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
            setVisible(false);
            new newbook().setVisible(true);
        }
    }

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

```

```

String bid = txtbid.getText();
String bname = txtbname.getText();
String author = txtbauthor.getText();
String publisher = txtbpublisher.getText();
String publishedyear = txtbpubyear.getText();

try {
    String url = "jdbc:mysql://localhost:3306/librarymanagement";
    String user = "root";
    String pswd = "";
    Connection con = DriverManager.getConnection(url, user, pswd);

    Statement stmt = con.createStatement();
    stmt.executeUpdate("UPDATE book SET bname = '"+bname+"',author =
 '"+author+"',publisher = '"+publisher+"',publishedyear = '"+publishedyear+"' WHERE bid =
 '"+bid+"'");
    JOptionPane.showMessageDialog(null,"Book Updated Successfully!");
    setVisible(false);
    new newbook().setVisible(true);
}
catch (SQLException ex) {
    JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
    setVisible(false);
    new newbook().setVisible(true);
}
}

private void btndeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String bid = txtbid.getText();

    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();
        stmt.executeUpdate("DELETE FROM book WHERE bid = '"+bid+"'");
        JOptionPane.showMessageDialog(null,"Book Deleted Successfully!");
        setVisible(false);
        new newbook().setVisible(true);
    }
}

```

```
catch (SQLException ex) {  
    JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());  
    setVisible(false);  
    new newbook().setVisible(true);  
}  
}
```

bookdetails.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
private void formComponentShown(java.awt.event.ComponentEvent evt) {
    // TODO add your handling code here:
    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();

        ResultSet rs1 = stmt.executeQuery("SELECT * FROM book");

        DefaultTableModel model1 = new DefaultTableModel();
        model1.setColumnIdentifiers(new String[]{"Book ID", "Book Name", "Author",
"Publisher", "Published Year"});

        while (rs1.next()) {
            model1.addRow(new Object[]{
                rs1.getString("bid"),
                rs1.getString("bname"),
                rs1.getString("author"),
                rs1.getString("publisher"),
                rs1.getString("publishedyear")
            });
        }
        tblbook.setModel(model1);
    }
    catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
        setVisible(false);
        new returnbook().setVisible(true);
    }
}

private void btncloseActionPerformed(java.awt.event.ActionEvent evt) {
```

```
// TODO add your handling code here:  
this.dispose();  
}
```

issubook.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;
private void btnsaveActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    SimpleDateFormat df = new SimpleDateFormat("dd-MM-yyyy");
    String bid = txtbid.getText();
    String stid = txtstid.getText();
    String issue = df.format(dateissue.getDate());
    String due = df.format(datedue.getDate());
    String datereturn = "NO";

    try {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException ex) {
            JOptionPane.showMessageDialog(null, "Class not found!");
            setVisible(false);
            new issuebook().setVisible(true);
        }
        // TODO add your handling code here:
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";

        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt1 = con.createStatement();
        Statement stmt2 = con.createStatement();
        Statement stmt3 = con.createStatement();

        ResultSet rsb = stmt1.executeQuery("SELECT * FROM book WHERE bid = '"+bid+"'");
        if(rsb.next())
        {
            ResultSet rss = stmt2.executeQuery("SELECT * FROM student WHERE stid = '"+stid+"'");
            if(rss.next())
            {
```

```

        stmt3.executeUpdate("INSERT INTO issue
VALUES('"+bid+"','"+stid+"','"+issue+"','"+due+"','"+datereturn+"')");
        JOptionPane.showMessageDialog(null,"Book Successfully Issued!");
        setVisible(false);
        new issuebook().setVisible(true);
    }
    else
    {
        JOptionPane.showMessageDialog(null, "Invalid Student ID!");
    }
}
else
{
    JOptionPane.showMessageDialog(null, "Invalid Book ID!");
}

rsb.close();
stmt1.close();
stmt2.close();
stmt3.close();
con.close();

} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
    setVisible(false);
    new issuebook().setVisible(true);
}
}

private void btncloseActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.dispose();
}

```

returnbook.java

```
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
private void btnsearchActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String bid = txtbid.getText();
    String stid = txtstid.getText();

    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM issue WHERE bid = '"+bid+"'
AND stid = '"+stid+"'");

        if(rs.next())
        {
            txtissue.setText(rs.getString(3));
            txtdue.setText(rs.getString(4));
            txtbid.setEditable(false);
            txtstid.setEditable(false);
        }
        else
        {
            JOptionPane.showMessageDialog(null,"Book is not issued to this student!");
            setVisible(false);
            new returnbook().setVisible(true);
        }
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
        setVisible(false);
        new returnbook().setVisible(true);
    }
}

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



```

        this.dispose();
    }

    private void btnreturnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String bid = txtbid.getText();
        String stid = txtstid.getText();

        try {
            String url = "jdbc:mysql://localhost:3306/librarymanagement";
            String user = "root";
            String pswd = "";
            Connection con = DriverManager.getConnection(url, user, pswd);

            Statement stmt = con.createStatement();
            stmt.executeUpdate("UPDATE issue SET datereturn = 'YES' WHERE bid = '"+bid+"'
AND stid = '"+stid+"'");
            JOptionPane.showMessageDialog(null,"Book Returned Successfully!");
            setVisible(false);
            new returnbook().setVisible(true);
        }
        catch (SQLException ex) {
            JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
            setVisible(false);
            new returnbook().setVisible(true);
        }
    }
}

```

statistics.java

```
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import net.proteanit.sql.DbUtils;

private void btnCloseActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.dispose();
}

private void tblIssueComponentShown(java.awt.event.ComponentEvent evt) {
    // TODO add your handling code here:
}

private void formComponentShown(java.awt.event.ComponentEvent evt) {
    // TODO add your handling code here:
    try {
        String url = "jdbc:mysql://localhost:3306/librarymanagement";
        String user = "root";
        String pswd = "";
        Connection con = DriverManager.getConnection(url, user, pswd);

        Statement stmt = con.createStatement();

        ResultSet rs1 = stmt.executeQuery("SELECT
issue.stid,student.stname,issue.bid,book.bname,issue.issue,issue.due "
        + "FROM student INNER JOIN book INNER JOIN issue WHERE book.bid =
issue.bid AND student.stid = issue.stid "
        + "AND issue.datereturn = 'NO'");
        //tblIssue.setModel(DbUtils.resultSetToTableModel(rs1));

        DefaultTableModel model1 = new DefaultTableModel();
        model1.setColumnIdentifiers(new String[]{"Student ID", "Student Name", "Book ID",
"Book Name", "Issue Date", "Due Date"});

        while (rs1.next()) {
            model1.addRow(new Object[]{
                rs1.getString("stid"),
                rs1.getString("stname"),
                rs1.getString("bid"),
                rs1.getString("bname"),
                rs1.getString("issue"),
                rs1.getString("due")
            });
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```

    });
}
tblissue.setModel(model1);

ResultSet rs2 = stmt.executeQuery("SELECT
issue.stid,student.stname,issue.bid,book.bname,issue.issue,issue.due "
+ "FROM student INNER JOIN book INNER JOIN issue WHERE book.bid =
issue.bid AND student.stid = issue.stid "
+ "AND issue.datereturn = 'YES'");
//tblreturn.setModel(DbUtils.resultSetToTableModel(rs2));

DefaultTableModel model2 = new DefaultTableModel();
model2.setColumnIdentifiers(new String[] {"Student ID", "Student Name", "Book ID",
"Book Name", "Issue Date", "Due Date"});

while (rs2.next()) {
    model2.addRow(new Object[] {
        rs2.getString("stid"),
        rs2.getString("stname"),
        rs2.getString("bid"),
        rs2.getString("bname"),
        rs2.getString("issue"),
        rs2.getString("due")
    });
}
tblreturn.setModel(model2);
}
catch (SQLException ex) {
    JOptionPane.showMessageDialog(null,"Error: "+ ex.getMessage());
    setVisible(false);
    new returnbook().setVisible(true);
}
}

```