

Individual Assignment

***by
Wahidul Alam Riyad***

***Systems Analysis & Design
School of Diploma
Asia Pacific University of Technology & Innovation***

***18th September 2018
Intake - UCDF1611ICTSE***

TABLE OF CONTENTS

1- INTRODUCTION	4
2- USER INTERFACE	5
2.1- Login.....	5
2.2- Register Users	6
2.3- Loading.....	7
2.4- Home	8
2.5- Add Book.....	9
2.6- Register Students	10
2.7- Issue Book.....	11
2.8- Return Book	12
2.9- Statistics	13
2.10- File	14
2.11- View.....	15
2.12- All Books	16
2.13- Registered Students Lists	17
2.14- About	18
3- CODING	19
3.1- Java Connect	19
3.2- Login.....	20
3.3- Signup	21
3.4- Loading.....	22
3.5- Home	23
3.6- Add Book.....	24
3.7- New Student	25
3.8- Issue Book.....	26
3.9- Return Book	27
3.10- Statistics	28
3.11- Registered Students	29
3.12- All Books	30
3.13- About	31

4- CONCLUSION & ASSUMPTION	32
5- REFERENCES	33
References	33

1- INTRODUCTION

Library Management System is a software used to manages the catalog of a library. This helps to keep the records of whole transactions of the books available in the library (Ampletrails, 2018). *Library Management System (LMS)* is my project for the Java Programming Module. The LMS can *Login, Register Users, Add Books, Register New Students, Issue Book, Return Book*, shows *Statistics of Issued Book and Returned Book*, shows list of *Registered Students and All Books*. It has great UI, such as navigation bar, application theme and many more. It follows every command of the user properly and runs the application as long as it takes.

2- USER INTERFACE

2.1- Login

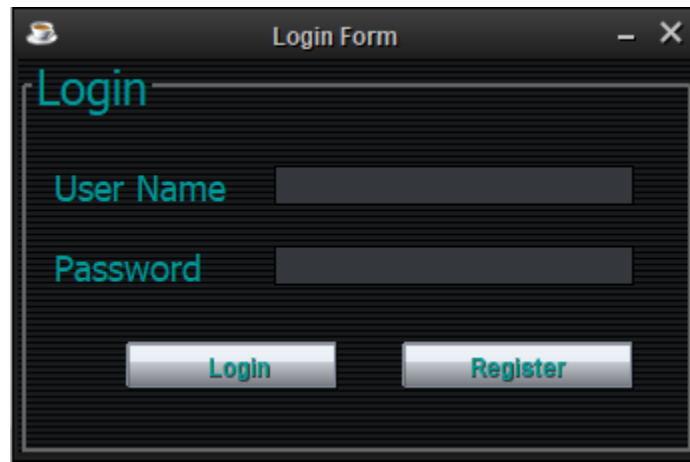
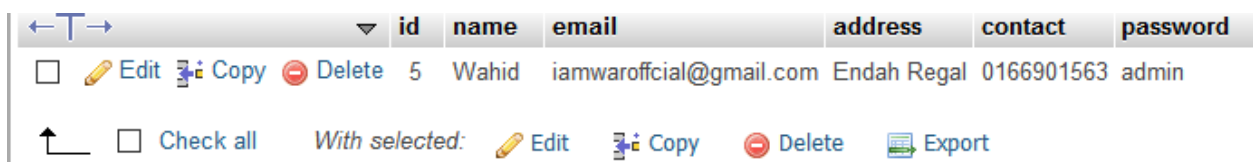


Figure: Login Form

When the application is executed, Login form is opened. Here only the registered users can login. The user details are retrieve from the users table in MySQL database. If the login details are empty or the username is invalid, the user will get an error message. If the user clicks Register Button, he will be taken to Register form.



	id	name	email	address	contact	password
<input type="checkbox"/>	5	Wahid	iamwarofficial@gmail.com	Endah Regal	0166901563	admin

Figure: Users Table Database

2.2- Register Users

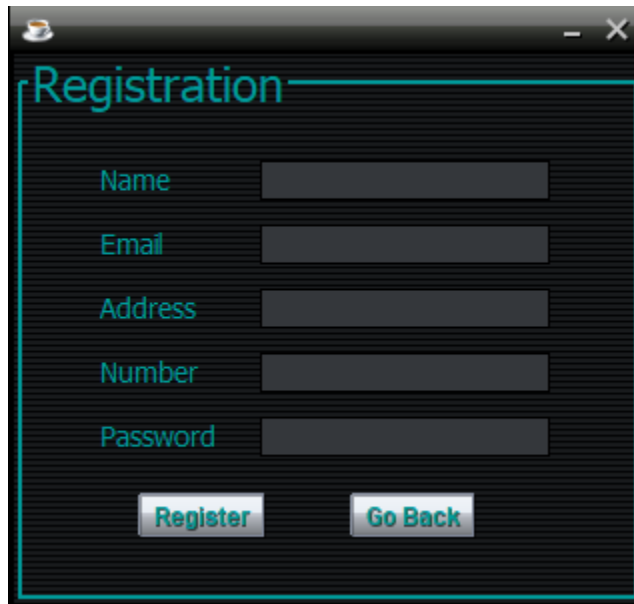


Figure: Register User

When the registration form is open, user will input his/her details which will be stored in the user table in the database. If all the details are filled properly user can register, otherwise the user will get an error message. Users with similar name are also not allowed. After the user is registered, the user can back to the login page by clicking back button.

2.3- Loading

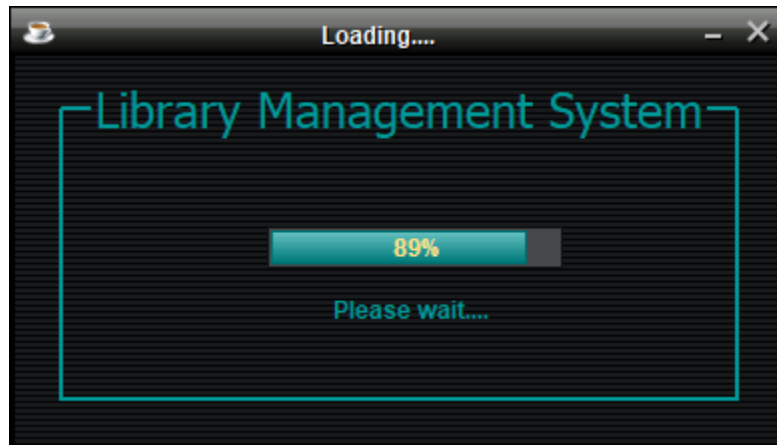
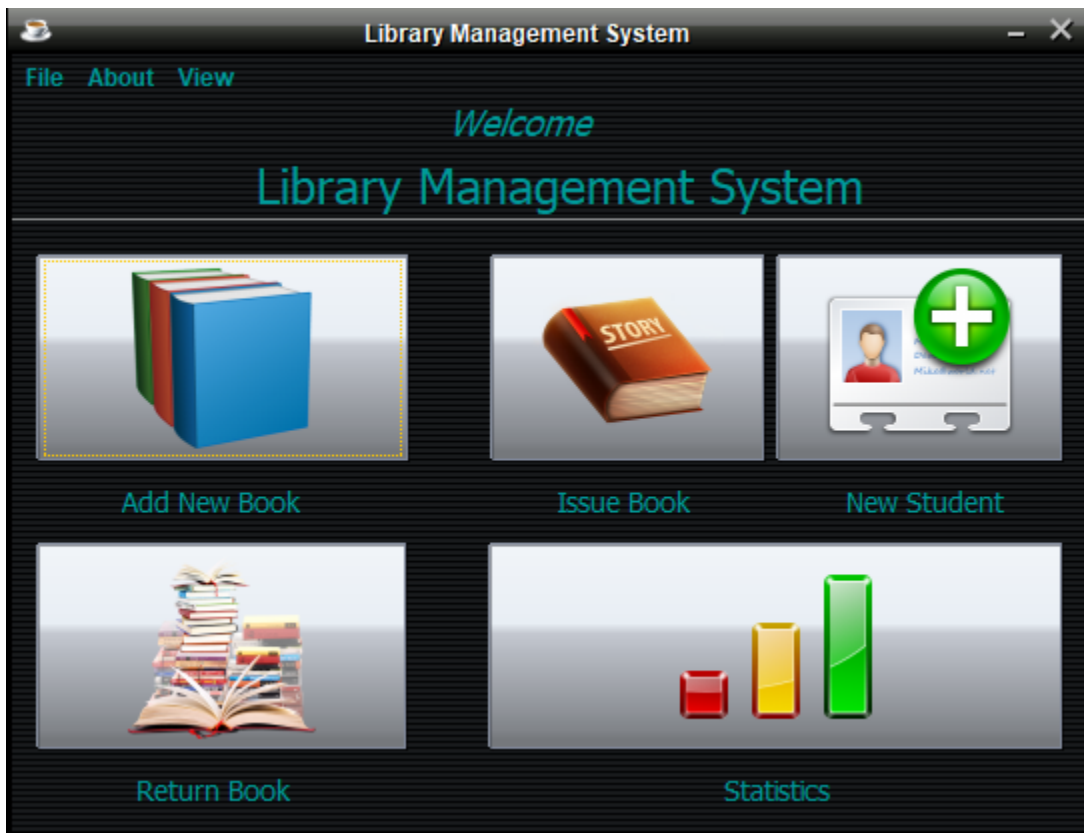


Figure: Loading

After the user details are accepted in login form, the user is taken to the home page through this nice loading UI. As the bar reaches 100 percent, the user is taken to the home page.

2.4- Home

**Figure: Home**

In home page, the user gets many options to navigate around. The user can add new book, issue book, add new student, return book and see the statistics. At the top of the LMS, there is a navigation bar which can be accessed at the top. The Home page has buttons and images for ease of access.

2.5- Add Book

Figure: Add New Book

The user can access the add book through home page. After the user is taken to new book page, the system generates random ID. If the user wants to change the Book Id, the user can change. After all the information is filled up, the user can add new book to the library which saves all the information in the books table of the database.

				book_id	name	publisher	edition	author
<input type="checkbox"/>	Edit	Copy	Delete	138	Computer Science	Pearson	5th	George
<input type="checkbox"/>	Edit	Copy	Delete	357	Business Statistics	Oxford	2nd	Alice
<input type="checkbox"/>	Edit	Copy	Delete	495	Business Management	Oxford	4th	Bruce
<input type="checkbox"/>	Edit	Copy	Delete	588	Java Programming	Pearson	3rd	Alfred
<input type="checkbox"/>	Edit	Copy	Delete	589	Artificial Intelligence	Pearson	4th	Michael
<input type="checkbox"/>	Edit	Copy	Delete	674	VB Net	Edlearn	3rd	Lebnan
<input type="checkbox"/>	Edit	Copy	Delete	727	HTML 5	Galvin	2nd	Griffin
	<input type="checkbox"/> Check all	With selected:		Edit	Copy	Delete	Export	

Figure: Book Table

2.6- Register Students

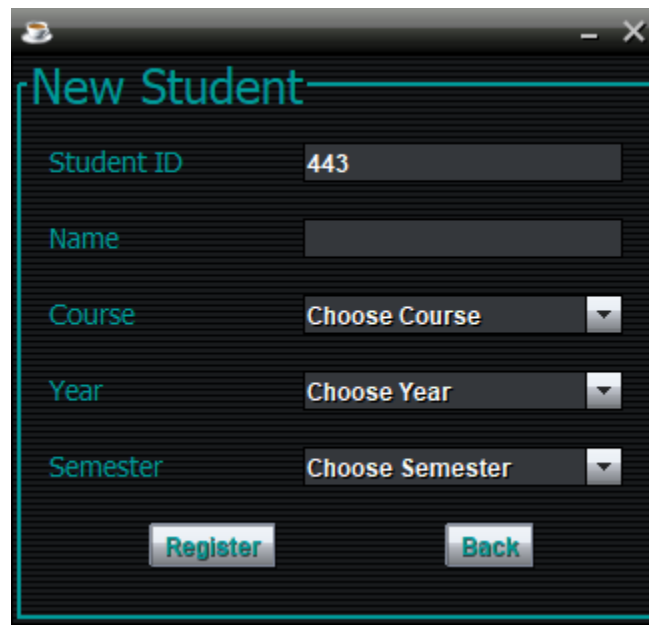


Figure: New Student

When the user access new student option, the user can register for new students. Here also it generates random Student Id, but the user also can give different ID for the student. Here the user can choose course, year and semester details from the combo button. All these details will be stored in student table of the database.

		student ID	name	course	year	semester
<input type="checkbox"/>	Edit Copy Delete				2nd	4th
<input type="checkbox"/>	Edit Copy Delete	642	Wahid	ICTSE	2nd	5th
<input type="checkbox"/>	Edit Copy Delete	891	Kenzo	ICTSE	2nd	5th
<input type="checkbox"/>	Edit Copy Delete	895	Omar	ICT	1st	2nd
<input type="checkbox"/>	Edit Copy Delete	980	Chloe	ICT	2nd	5th
<input type="checkbox"/> Check all With selected: Edit Copy Delete Export						

Figure: Student Table

2.7- Issue Book

Issue Book

Library Management System

Book Details

Book ID

Book Name

Publisher

Edition

Author

Student Details

Student ID

Name

Course

Year

Semester

Date of Issue

Figure: Issue Book

The user can also access the Issue page from the Home menu. The user can type the required book id and click search. After clicking search, the book details will automatically have retrieved from the book table. Then the user will key in the student id and click the search button. Afterwards the student details are retrieved automatically. Finally, the user will choose the date from the date of issue, and the book will be issued which will be stored in the issue table.

id	book_id	user_id	issue_date
Query results returned			

Figure: Issue Table

2.8- Return Book

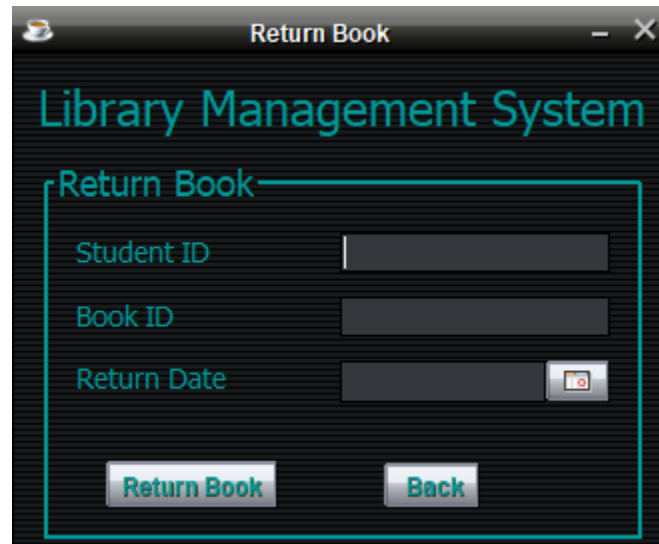
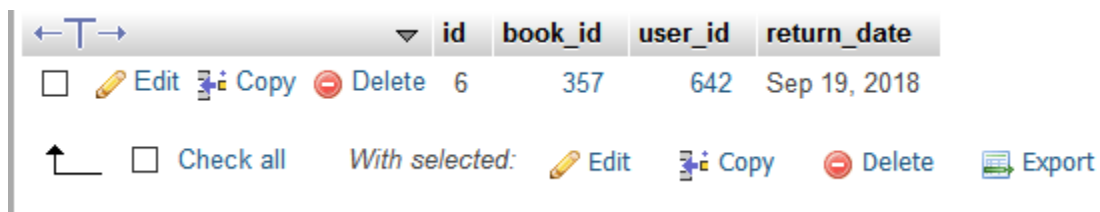


Figure: Return Book

The user can access the Return Book page from the home menu. Here the user will give the valid student id and book id in the text field. Finally, the user will choose the return date and click return book. If all the details are valid, the book will be moved from issued to return table. Hence, the book will be available again for issuing.



	id	book_id	user_id	return_date
<input type="checkbox"/> Edit Copy Delete	6	357	642	Sep 19, 2018

↑ ☐ Check all With selected: Edit Copy Delete Export

Figure: Return Table

2.9- Statistics

Statistics

Library Management System

Issued Books

Student_Name	Course	Student_ID	Book_ID	Issue_Date
--------------	--------	------------	---------	------------

Return Books

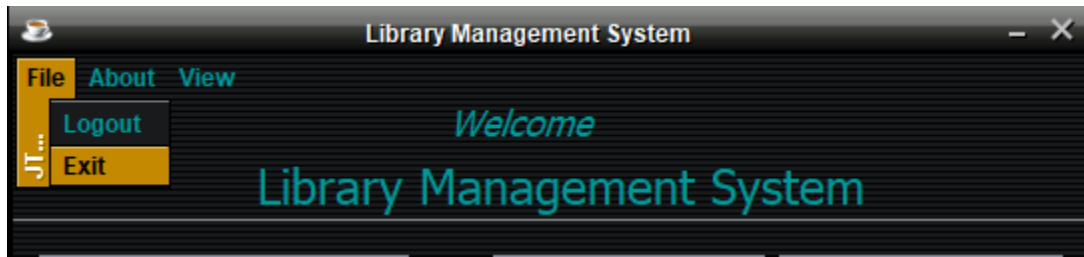
Book_Name	Edition	Book_ID	Student_ID	Return_Date
Business Statisti...	2nd	357	642	Sep 19, 2018

Close

Figure: Statistics

The user can view the statistics of both issued book and returned book from this form. Both the tables are retrieved from the database. It is also updated instantaneously and the user can view the current information.

2.10- File

**Figure: File**

Here the user can navigate the navigation menu. Under file the user can logout or close the application instantaneously.

2.11- View

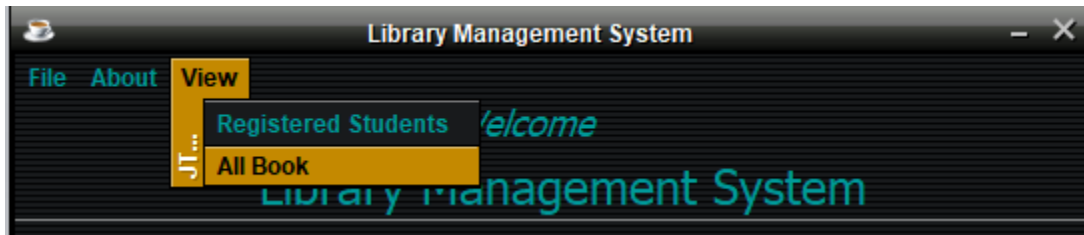
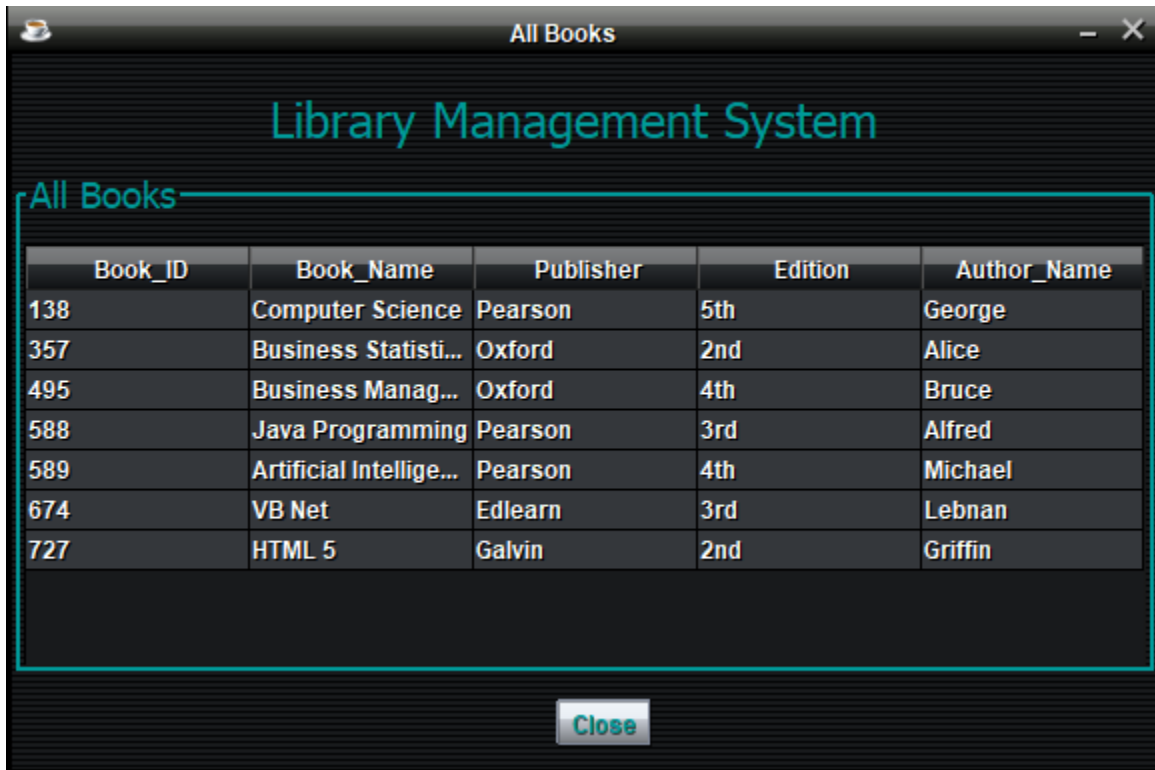


Figure: View

Here the user can view the list of registered students and list of all books.

2.12- All Books



Library Management System

All Books

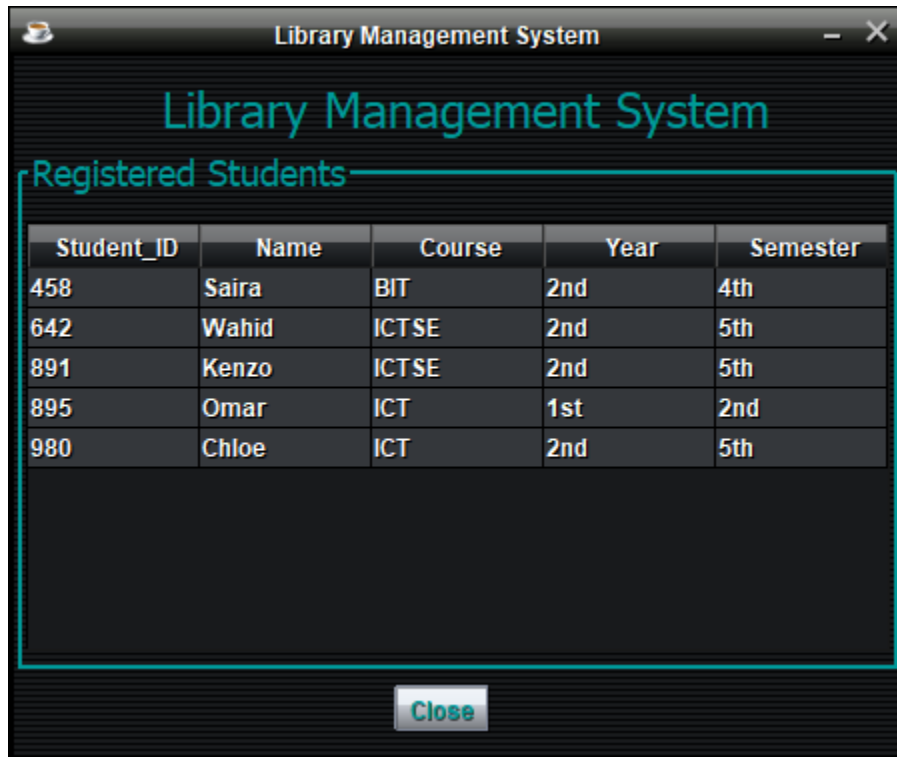
Book_ID	Book_Name	Publisher	Edition	Author_Name
138	Computer Science	Pearson	5th	George
357	Business Statisti...	Oxford	2nd	Alice
495	Business Manag...	Oxford	4th	Bruce
588	Java Programming	Pearson	3rd	Alfred
589	Artificial Intellige...	Pearson	4th	Michael
674	VB Net	Edlearn	3rd	Lebnan
727	HTML 5	Galvin	2nd	Griffin

Close

Figure: All Books

After the user click all book from the navigation menu, the user can get the list of all the book from the book table. It shows all the details of all the books.

2.13- Registered Students Lists



The screenshot shows a window titled "Library Management System" with a dark background. The title bar includes a standard icon and window controls. The main content area displays the text "Library Management System" in a large, light blue font, followed by "Registered Students" in a smaller, light blue font. Below this, a table with five columns is shown. The columns are labeled "Student_ID", "Name", "Course", "Year", and "Semester". The table contains five rows of student data. A "Close" button is located at the bottom center of the window.

Student_ID	Name	Course	Year	Semester
458	Saira	BIT	2nd	4th
642	Wahid	ICTSE	2nd	5th
891	Kenzo	ICTSE	2nd	5th
895	Omar	ICT	1st	2nd
980	Chloe	ICT	2nd	5th

Figure: Registered Students

The user can also view the list of all the registered students which are stored in the registered table in the database.

2.14- About

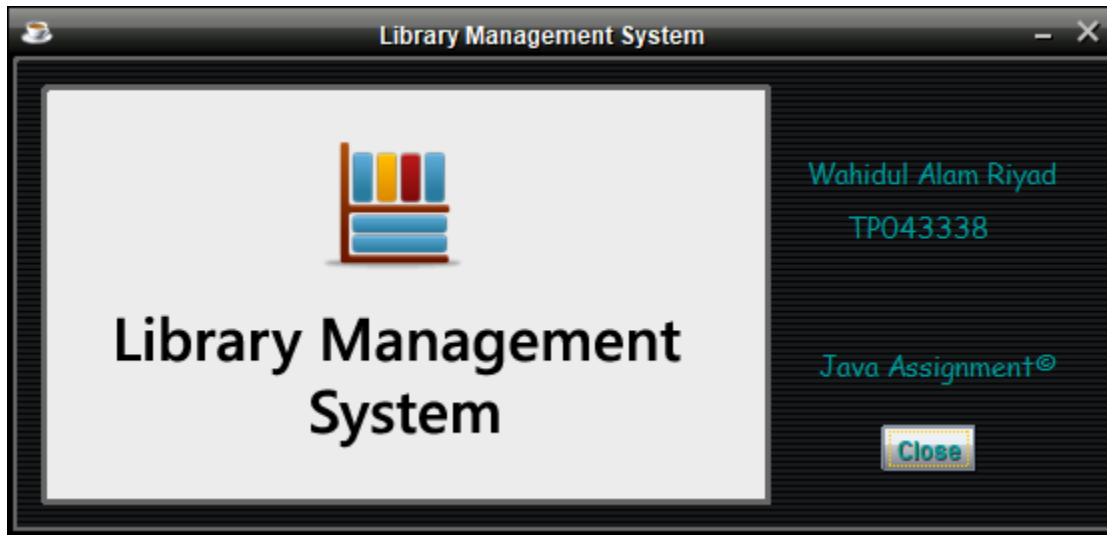


Figure: About

This is the about page which can be accessed from the navigation menu. It shows my credentials.

3- CODING

3.1- Java Connect

```
1
2 import com.mysql.jdbc.Connection;
3 import java.sql.DriverManager;
4 import javax.swing.*;
5
6 /*
7  * To change this license header, choose License Headers in Project Properties.
8  * To change this template file, choose Tools | Templates
9  * and open the template in the editor.
10 */
11
12 /**...4 lines */
13 public class JavaConnect {
14     Connection conn;
15
16     public static Connection ConnectDB(){
17         try{
18             Class.forName("com.mysql.jdbc.Driver");
19             Connection conn = (Connection) DriverManager.getConnection("jdbc:mysql://localhost/lms" , "root", "");
20             return conn;
21         }
22         catch(Exception e)
23         {
24             JOptionPane.showMessageDialog(null, e);
25             return null;
26         }
27     }
28 }
29
30
31
32
33
34
35
```

This is the java connect file. Here I have established connection with MySQL Database. To run this properly “mysql.jdbc.driver” is installed in the library. The database name is “lms” which user name is root and password are blank.

3.2- Login

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    try{  
        String sql = "select * from users where name= ? AND password= ? ";  
        pst= conn.prepareStatement(sql);  
        pst.setString(1, name.getText());  
        pst.setString(2, password.getText());  
        rs = pst.executeQuery();  
        if(rs.next()){  
            setVisible(false);  
            Loading ob = new Loading();  
            ob.setUpLoading();  
            ob.setVisible(true);  
        }  
        else{  
            JOptionPane.showMessageDialog(null,"Incorrect name Or Password ! ", "ERROR", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
    catch(Exception e ){  
        JOptionPane.showMessageDialog(null, e);  
    }  
}
```

When the login button is executed, sql commands retrieve name and password from the users table. If it executes with valid credentials, the user will be taken to loading java file, otherwise it will show an error message.

3.3- Signup

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    try{  
        String sql = "insert into users (name, email, address, contact, password) values (?, ?, ?, ?, ?)";  
        pst=conn.prepareStatement(sql);  
        pst.setString(1, jTextField1.getText());  
        pst.setString(2, jTextField2.getText());  
        pst.setString(3, jTextField3.getText());  
        pst.setString(4, jTextField4.getText());  
        pst.setString(5, jTextField5.getText());  
        pst.execute();  
        JOptionPane.showMessageDialog(null, "Record Inserted");  
        setVisible(false);  
        Login ob = new Login();  
        ob.setVisible(true);  
    }  
  
    catch(Exception e){  
        JOptionPane.showMessageDialog(null, e);  
    }  
}
```

Here the sql commands inserts the details inside the users table. After it is inserted, the user will be taken to the login form.

3.4- Loading

```
public void run(){
    try{
        for(int i =1; i<=200; i++)
        {
            s = s+1;
            int m = jProgressBar1.getMaximum();
            int v = jProgressBar1.getValue();
            if( v < m )
            {
                jProgressBar1.setValue(jProgressBar1.getValue()+1);
            }

            else{
                i =201;
                setVisible(false);
                Home ob = new Home();
                ob.setVisible(true);
            }
            Thread.sleep(50);

        }
    }

    catch(Exception e){
        JOptionPane.showMessageDialog(null, e);
    }
}
```

This is the code for loading progression, if the value reaches 100 percent it will proceed to home java file.

3.5- Home

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {  
    Student ob = new Student();  
    ob.setVisible(true);  
}  
  
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {  
    setVisible(false);  
    Issue ob = new Issue();  
    ob.setVisible(true);  
}  
  
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    Return ob = new Return();  
    ob.setVisible(true);  
}  
  
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    Statistics ob = new Statistics();  
    ob.setVisible(true);  
}  
  
private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt) {  
    AllBooks ob = new AllBooks();  
    ob.setVisible(true);  
}  
  
private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {  
    RegisteredStudents ob = new RegisteredStudents();  
    ob.setVisible(true);  
}
```

These are the codes for hyperlink to different java file. All the button leads to different java file. Hence, all the codes are written under button.

3.6- Add Book

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    String sql = "insert into books (book_id, name, publisher, edition, author) values (?, ?, ?, ?, ?)";  
    try{  
        pst = conn.prepareStatement(sql);  
        pst.setString(1, jTextField1.getText());  
        pst.setString(2, jTextField2.getText());  
        pst.setString(3, jTextField3.getText());  
        pst.setString(4, (String) jComboBox2.getSelectedItem());  
        pst.setString(5, jTextField5.getText());  
        pst.execute();  
        JOptionPane.showMessageDialog(null, "Book Added Successfully !");  
    }  
  
    catch(Exception e){  
        JOptionPane.showMessageDialog(null, e);  
    }  
}
```

Here sql commands inserts the information inside the book table. After the connection is established, the book will be added into the database.

3.7- New Student

```
public void Random() {  
    Random rd = new Random();  
    jTextField1.setText(""+rd.nextInt(1000+1));  
}
```

Here it generates random value in the Student ID. If the user wants to change the default value, the can change back again.

3.8- Issue Book

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    String sql = "insert into issue (book_id, user_id, issue_date) values (?, ?, ?)";  
    try{  
        pst = conn.prepareStatement(sql);  
        pst.setString(1, jTextField1.getText());  
        pst.setString(2, jTextField6.getText());  
        pst.setString(3, ((JTextField)jDateChooser1.getDateEditor().getUiComponent()).getText());  
        pst.execute();  
        JOptionPane.showMessageDialog(null, "Book Issued");  
        setVisible(false);  
        Home ob = new Home();  
        ob.setVisible(true);  
    }  
  
    catch(Exception e){  
        JOptionPane.showMessageDialog(null, e);  
    }  
}
```

Here the sql commands store information including the issue date into the issue table. If all the details are valid the user will be taken back to home menu.

3.9- Return Book

```
public void Delete(){
    String sql = "delete from issue where book_id = ? AND user_id = ? ";
    try{
        pst = conn.prepareStatement(sql);
        pst.setString(1, BID.getText());
        pst.setString(2, SID.getText());
        pst.execute();
    }

    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, e);
    }
}
```

Here it deletes the information from the issue table and store it in return table. It only updates with same Book and User ID.

3.10- Statistics

```
public void jTable1() {
    String sql = "select a.name AS Student_Name, a.course AS Course, b.user_id AS Student_ID, b.book_id AS Book_ID, b.issue_date AS Issue_
try{
    pst= conn.prepareStatement(sql);
    rs= pst.executeQuery();
    jTable1.setModel(DbUtils.resultSetToTableModel(rs));

}

catch(Exception e)
{ JOptionPane.showMessageDialog(null, e); }

}

public void jTable2() {
    String sql = "select a.name AS Book_Name, a.edition AS Edition , b.book_id AS Book_ID, b.user_id AS Student_ID, b.return_date AS Retu
try{
    pst= conn.prepareStatement(sql);
    rs= pst.executeQuery();
    jTable2.setModel(DbUtils.resultSetToTableModel(rs));

}

catch(Exception e)
{ JOptionPane.showMessageDialog(null, e); }

}
```

Here sql commands retrieve information from both the table and it displays in the statistics form.

3.11- Registered Students

```
public RegisteredStudents() {  
    super("Library Management System");  
    initComponents();  
    conn = JavaConnect.ConnectDB();  
    jTable1();  
}  
  
public void jTable1(){  
    String sql = "select student_ID AS Student_ID, name AS Name, course AS Course, year AS Year, semester AS Semester from student";  
    try{  
        pst= conn.prepareStatement(sql);  
        rs= pst.executeQuery();  
        jTable1.setModel(DbUtils.resultSetToTableModel(rs));  
    }  
  
    catch(Exception e)  
    { JOptionPane.showMessageDialog(null, e); }  
}
```

Here after the connection is established, it will retrieve all the student details from the student table. Hence it will show in table mode.

3.12- All Books

```
public void jTable1(){  
    String sql = "select book_id AS Book_ID, name AS Book_Name, publisher AS Publisher, edition AS Edition, author AS Author_Name from books"  
    try{  
        pst= conn.prepareStatement(sql);  
        rs= pst.executeQuery();  
        jTable1.setModel(DbUtils.resultSetToTableModel(rs));  
    }  
  
    catch(Exception e)  
    { JOptionPane.showMessageDialog(null, e); }  
}
```

Here sql string retrieve the information from the book table and it displays as a table in book form.

3.13- About

```
    /**  
    public class About extends javax.swing.JFrame {  
  
        /**  
        * Creates new form About  
        */  
        public About() {  
            super("Library Management System");  
            initComponents();  
        }  
    }
```

This is the codes for the about page. Anything after super shows the title of the java form.

4- CONCLUSION & ASSUMPTION

This project was really helpful to me. As I was researching more and more on database and java, I was gaining great skills to create this LMS which can be helpful in real life purpose. The assumption of this project for me was to implement the fine feature after certain days in the java file. As I was researching on database, it was really difficult for me to implement that feature on my system.

5- REFERENCES

References

Ampletrails, 2018. *Library Management System*. [Online]
Available at: <https://ampletrails.com/library-management-system>
[Accessed 18 September 2018].