**VIETNAM NATIONAL UNIVERSITY – HOCHIMINH CITY**

**INTERNATIONAL UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

****

**OBJECT-ORIENTED PROGRAMMING**

**PROJECT REPORT**

**LIBRARY MANAGEMENT SYSEM**

**Member’s name:**

Trần Hoàng Thịnh - ITITIU19054

Phạm Nguyễn Ngọc Mai – ITITWE19019

Phan Ánh Ngọc – ITITWE19018

**Lecturers:**

MSc. Lê Thanh Sơn

MSc. Nguyễn Quang Phú

**CONTENTS**

**1. Introduction 4**

**2. Library Management System 4**

**1.1. Login Page 4**

**2.2. Home page 7**

**2.2.1. Adding New Students, New Books 8**

**2.2.2. Issue and Return Book 9**

**3. UML Diagram for Library Management System 11**

**4. Explain code 11**

**1.2. JOptionPane 12**

**1.3. javax.swing package 12**

**1.3.1. JPanel 13**

**1.3.2. JTextField 13**

**1.3.3. JLabel 13**

**1.3.4. JButton 14**

**1.3.5. JPasswordField 15**

**1.3.6. JcomboBox 15**

**1.3.7. JMenuBar and JmenuItem 15**

**1.4. GroupLayout 17**

**1.5. JCalendar 18**

**5. Singleton class 18**

**6. Database code 18**

**7. Some experiences when using Microsoft SQL Server 19**

**8. References 20**

**Table of Figure**

**Figure 1:** Login Page 5

**Figure 2:** Announcement wrong account 5

**Figure 3:** New Account 6

**Figure 4:** Forgot Password 6

**Figure 5:** Loading 7

**Figure 6:** Home page 8

**Figure 7:** Add Book and Add Student 9

**Figure 8:** Issue Book 10

**Figure 9:** Return Book 10

**Figure 10:** The class diagram of Library 11

**Figure 11:** Declaration 12

**Figure 12:** Border Factory 13

**Figure 13:** Insert Image 14

**Figure 14:** JButton and Action Performed 14

**Figure 15:** First Model 15

**Figure 16:** Second Model 15

**Figure 17:** JMenu diagram 16

**Figure 18:** JMenu and Action Performed 16

**Figure 19:** Example of GroupLayout 17

**Figure 20:** Singleton class 17

**Figure 21:** Connect to database server 18

**Figure 22:** Explain code 18

# 

# **1. Introduction**

In this school year, 2020 – 2021, the Object – Oriented Programming course requires students to do project in order to consolidate the object-oriented knowledge. Students are asked to do either game project or application project. So, our group has decided to do Library Management System. There are two reasons why we choose Library Management System. First, Library Management System is needed in the real life. Second, we can prepare knowledge for next semester course, Principle of Database Management.

This application concentrates mostly on User Interface (UI) and database so we need a beautiful and clear graphic then Objected – Oriented method by Java language is the best tool for us to design because Java has provided sufficient, convenient classes and method for designing UI. The IDE we use is NetBeanse IDE because it helps us a lot in drawing and designing the graphic. For database, after choosing and trying, we used Microsoft SQL Server.

The advantages of our application are colorful graphic, easy to implement, can sign-up, login, forgot password, can add new books or new students or can see list of books and students after adding. The disadvantages are we haven’t found out how to delete books and students in the list.

# **2. Library Management System**

# **Login Page**

In the login page, we have login button, sign up button and forgot password button. The login class operate when the account available in the database system. Otherwise, the system will announce and users need to sign-up. Other case is users type wrong password, then the users will change to forgot password page. Users will get back the password by answering the question they chose when they sign – up. This password we save in the database system. Each page users can use Back button to get back login page if not necessary.

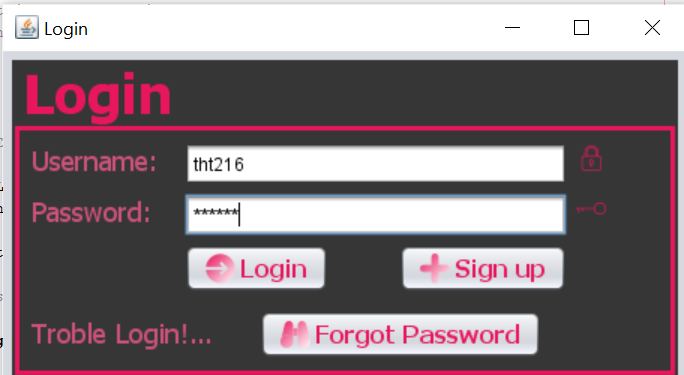


Figure 1: Login Page

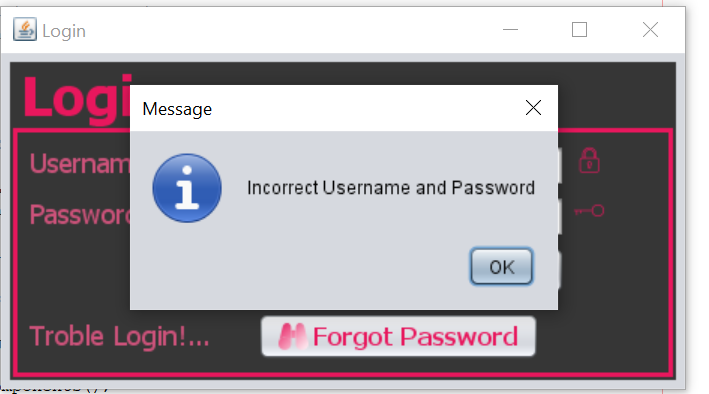


Figure 2: Announcement wrong account

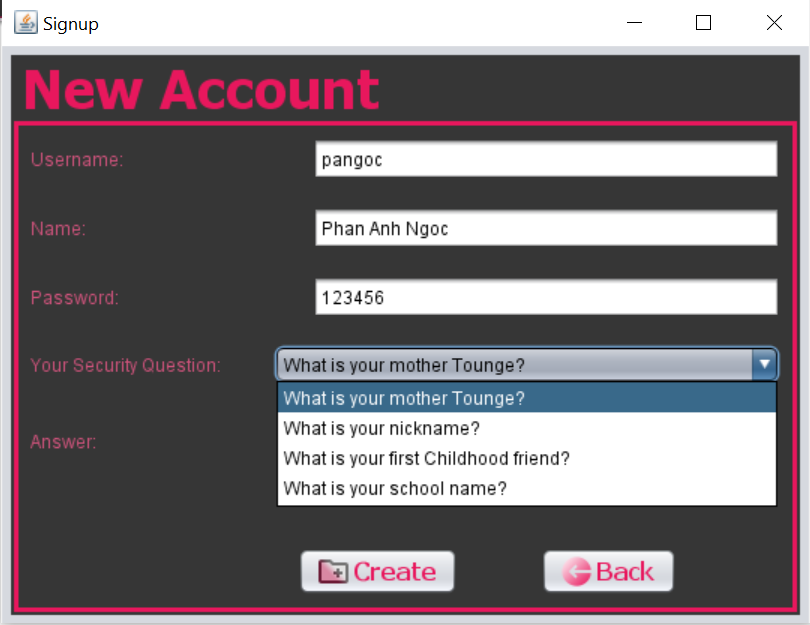


Figure 3: New Account

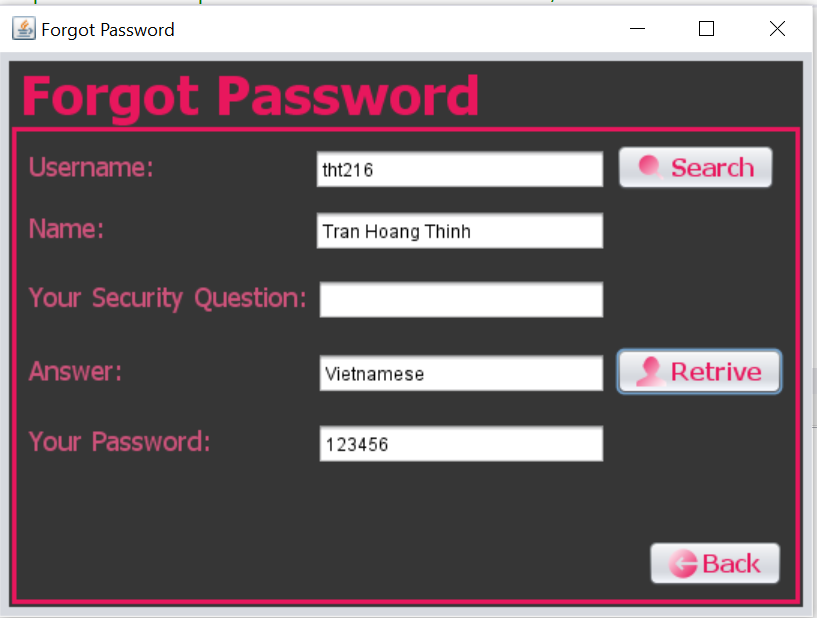


Figure 4: Forgot Password

Also, the special function we have in the Library is Loading page with the gif image to get to the Home page.



Figure 5: Loading

# **2.2. Home page**

The system has 2 main parts: Operation and Action.

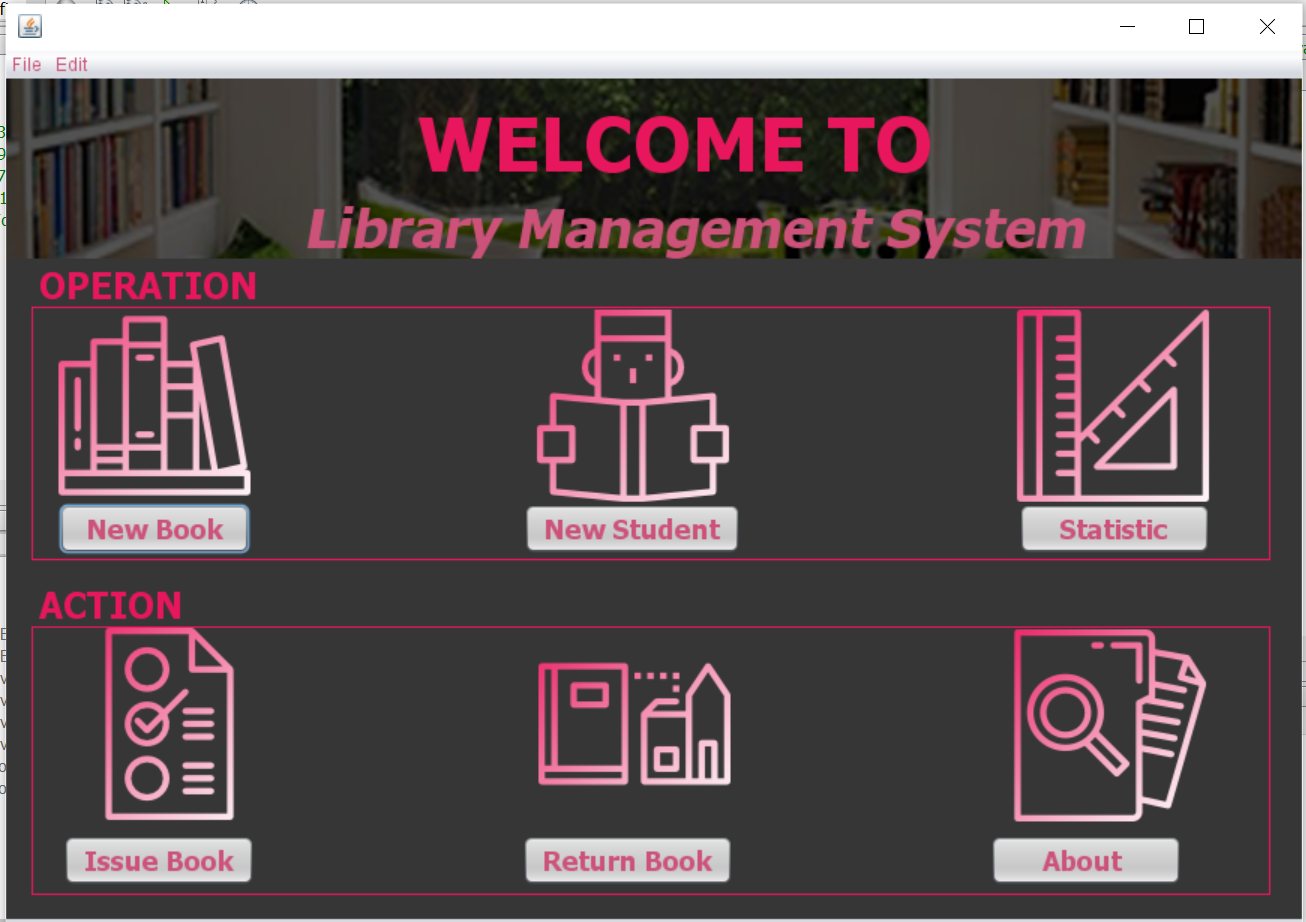
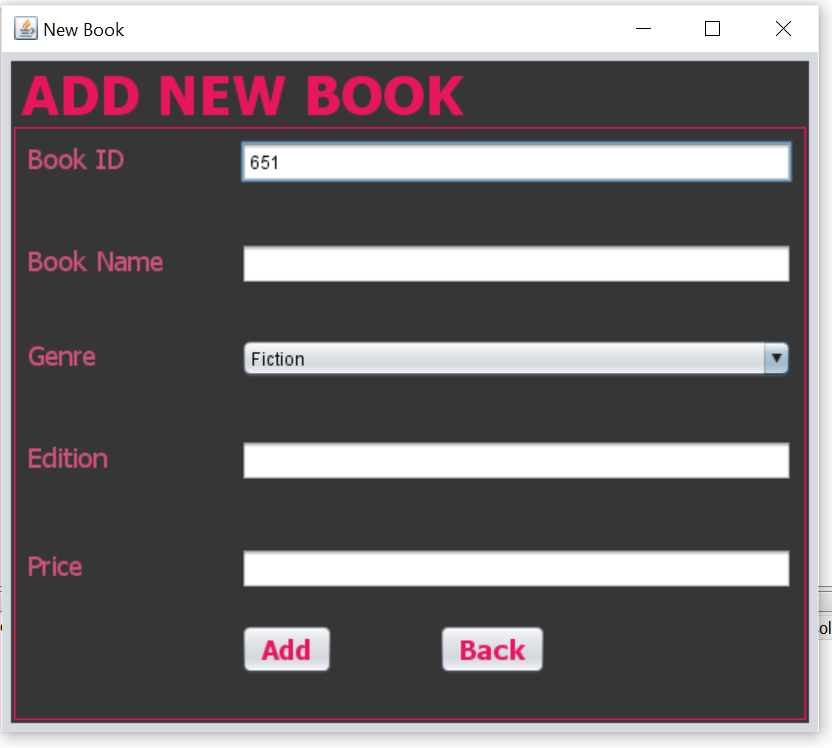
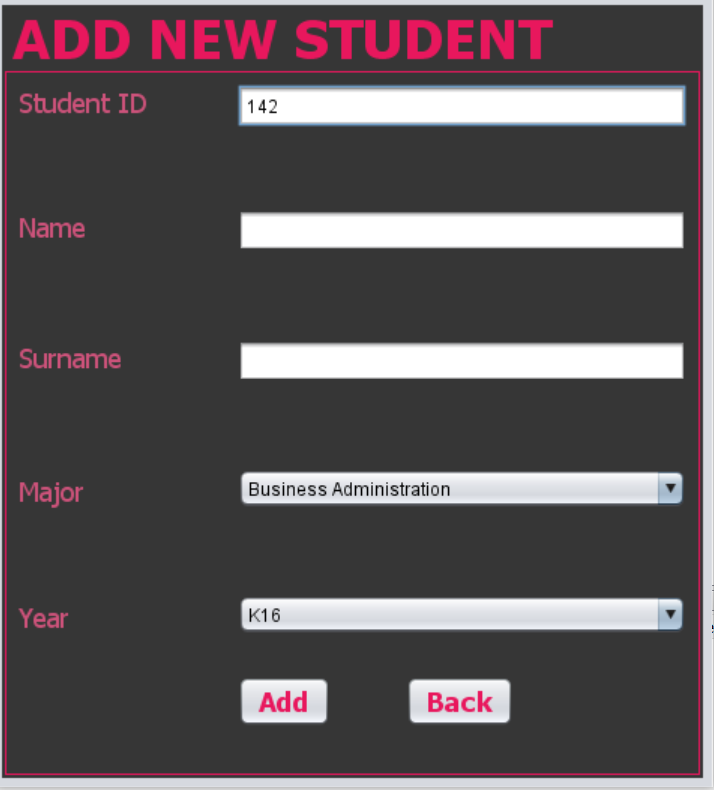


Figure 6: Home page

# **Adding New Students, New Books**

In both adding new students and new books pages, when the users press Add button, the data will write on the database system. The limitation of this version is the system cannot distinguish which account is student, which account is administration.

Figure 7: Add Book and Add Student



# **2.2.2. Issue and Return Book**

In Action, Issue Book and Return Book are the most important of the Library Management System. User can search book by input the ID of that book, so they can have other information about it. The same with students who want to borrow books, just search their student ID. After that, this system also help administrator manage the date that books were issued and returned.

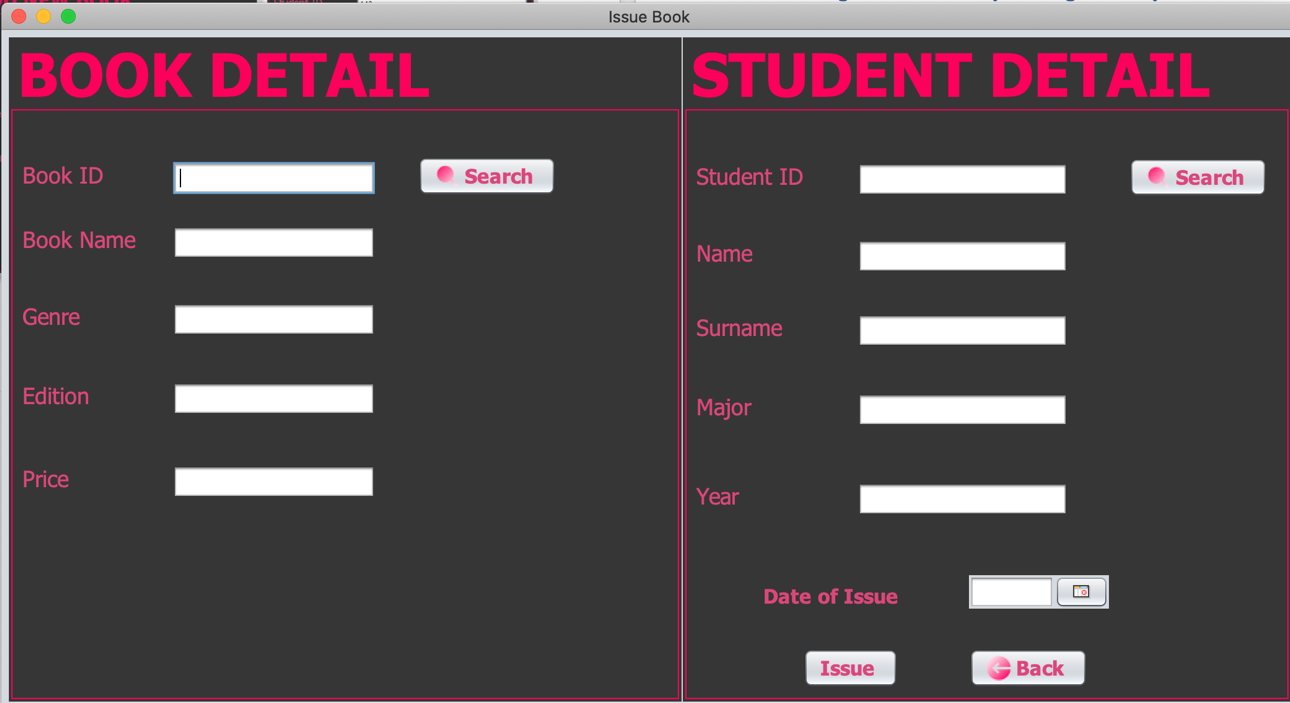


Figure 8: Issue Book

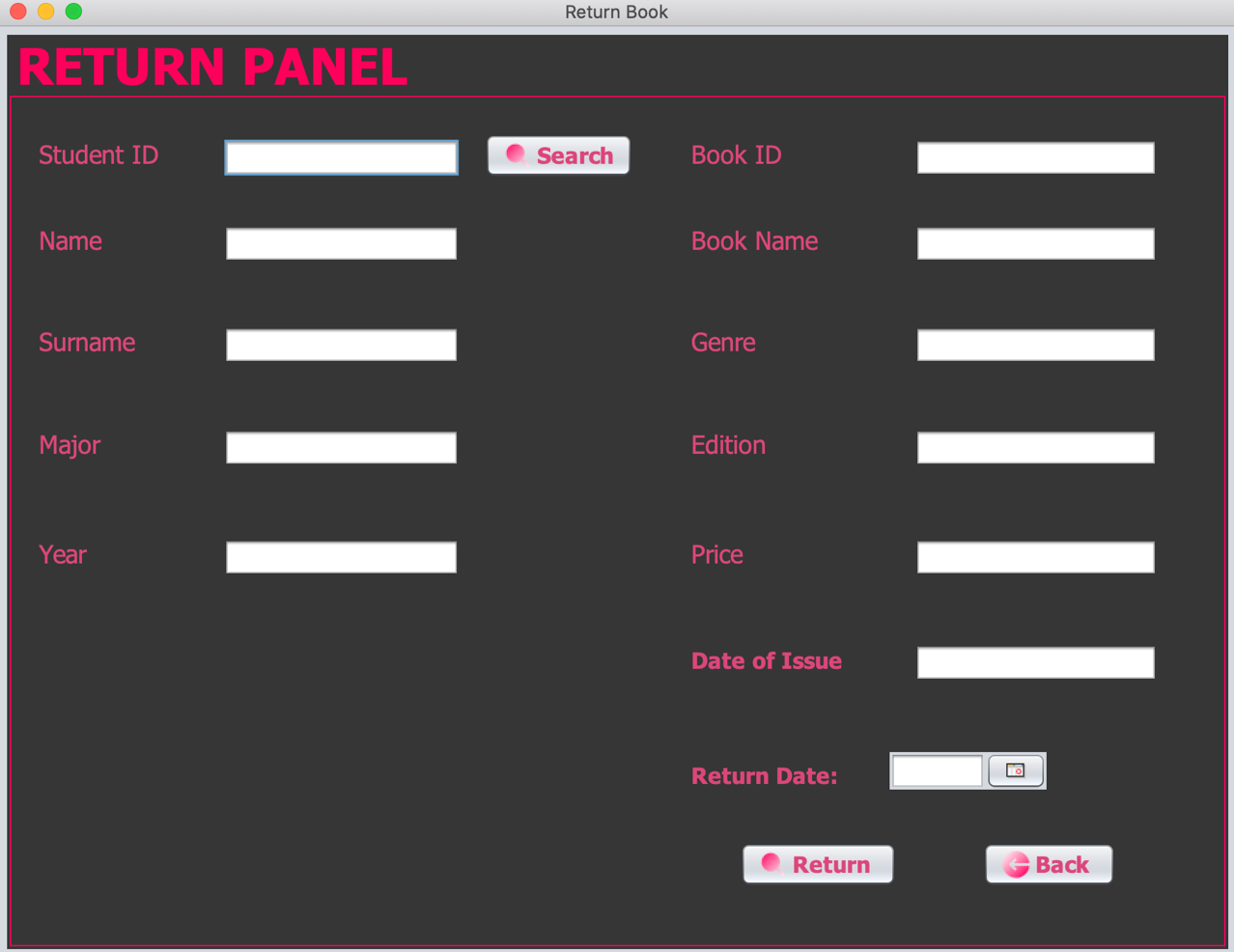


Figure 9: Return Book

# 

# **3. UML Diagram for Library Management System**



Figure 10: The class diagram of Library

# **4. Explain code**

In this project, we have 16 classes in total. They are: Login, Sign-up, Forgot Password, Loading, Home, NewStudent, NewBook, Issue, Statistic, Return, About, IForgot, ILogin, PrintInfo, jMonthChooser11 and javaconnect.

# 

# **JOptionPane**

JoptionPane class is used to provide standard dialog boxes such as message dialog box, confirm dialog box and input dialog box. These dialog boxes are used to display information or get input from the user. It inherits form JComponent class

# **javax.swing package**

This Library Management System we have used mostly is **javax.swing.JFrame** from javax.swing package to use components like **JPanel, JTextField, JLabel, JPasswordField** and **JButton** to create UI. Also, Jframe has the option to hide or close the window by using **setDefaultCloseOperation** method with **EXIT\_ON\_CLOSE** field and **setVisible**.



Figure 11: Declaration

# 

# **JPanel**

**JPanel** inherits from the **JComponent** class. It is known as a simplest container that can store other components and organize them with methode setBackground and setBorder. Various layouts can be set in **JPanel**, however it does not have a title bar. Instead, we use **BoderFactory**.

**BoderFactory** is a Factory class which provides different types of borders in Java. In this case, we create a line border with titled border. The problem we have found out is we cannot set color or font properly in swing so we have to use java.awt. So throughout the project, we will use java.awt with color palettes code and font name when needed

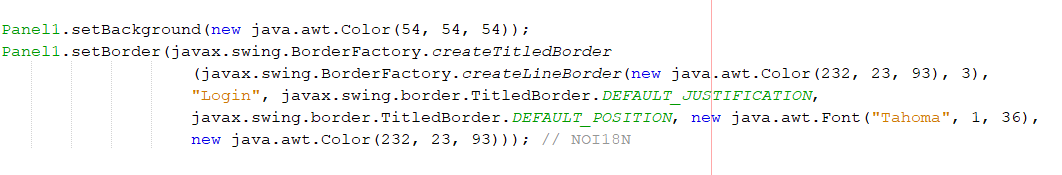


Figure 12: Border Factory

# **JTextField**

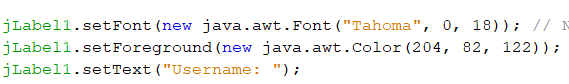
**JTextField** class inherits from **JTextComponent** class. It allows the users to input a

single line text and database server will get that text if necessary

# **JLabel**

**JLabel** is a component for placing text in a container. It is used to display a read – only text which mean the text can be changed only by the programmer but not the users. The methods we are using are:

* setFont with font name and font size
* setForeground to set the text color
* setText to enter the text



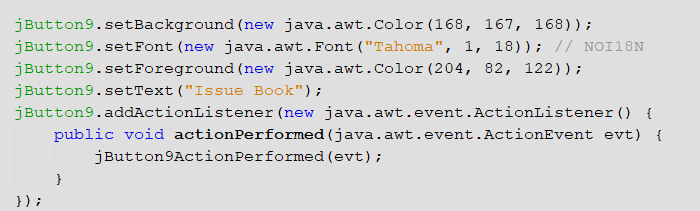
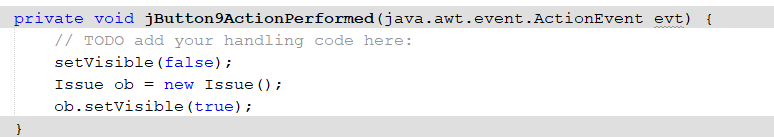
Also, we are using method setIcon from **javax.swing.ImageIcon** to insert image



Figure 13: Insert Image

# **JButton**

**JButton** in Java Swing inherits from **AbstractButton** class and it is used to create button that user can click on. This component has a label and create an event when it is clicked.



That’s why we use **addActionListener** form **java.awt.event package**. Like JLabel, we can also use **setBackground** (set Background colour), **setFont** (set Font name and Font size), **setForeground** (set Text color), **setText** (insert lable) methods.

Figure 14: JButton and Action Performed

In this code, we see that method **jButton9ActionPerformed** method are called after **addActionListener**, which mean after clicking the button, the system will lead the users to the page they want by **setVisible** method.

*Example:* In Home page we have “Issue Book” button so it will change to Issue Book Page when the button is clicked.

# **JPasswordField**

**JpasswordField** is a lightweight component that allows the users input a text component specialized for password entry, does not shw the original characters. It inherits JTextField class. The password of the users will be written on the database server.

# **JcomboBox**

JComboBox lets the user choose one of several choices. This class has two models. The first one is the default model, which is the unedited combo box. It has a button and a drop-down list of items. The second one is called the edited combo box which users can either type in the text field or click the button to display a drop-down item.

For example:

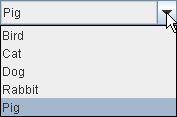
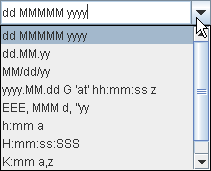


Figure 15: First Model

Figure 16: Second Model

In this project, we use the first model.

# **JMenuBar and JmenuItem**

The **JMenuBar** class is used to display menu bar on the frame.

The **JMenu** class is a pull-down menu component which is displayed from the menu bar. It inherits the **JMenuItem** class.

The **JMenuItem** class helps to add menu item for JMenu.

Again, we can use methods like: **setBackground**, **setForeground** and **setText** methods. For the item Exit and Logout, we have to use **addActionListener** to add event for them.



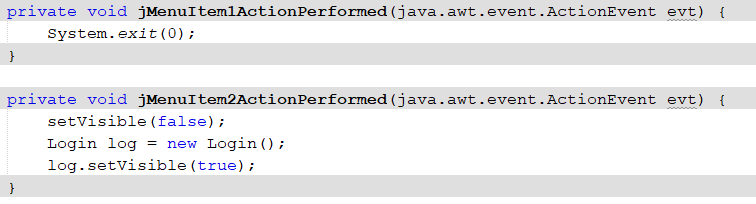
JMenuItem

JMenuBar

JMenu

Figure 17: JMenu diagram

Figure 18: JMenu and Action Performed



# **GroupLayout**

**GroupLayout** help developer to group the components and arranges them in a Container.

**GroupLayout** treats each axis independently. There is a group representing the horizontal axis and a group representing the vertical axis. Each component must exist in both a horizontal and vertical group. Because we have used **JPanel** to create a group in the project so instead of using GroupLayout, we can also use **JPanelLayout.setHorizontalGroup** and JPanelLayout.setVerticalGroup.

**GroupLayout** support two types of groups:

* A sequential group positions its elements sequentially, one after another
* A parallel group align its items on top of each one

So, the GroupLayout class provides methods such as **createParallelGroup()** and **createSequentialGroup()** to create these two groups (using **addGroup**). After creating groups, we need add components to these groups by using method: **addComponent** or **addComponent (component, int min, int pref, int max)** when adds a component with the specified size

Some other methods we use are:

* addContainerGap(): adds the preferred gap between an edge the container and componenents that touch the border of the container
* addGap (int min, int pref, int max): adds a gap with the specified size.
* addPreferredGap (LayoutStyle.ComponentPlacement type, int pref, int max): adds preffered gap between the nearest components

Note that these methods use only for sequential group.

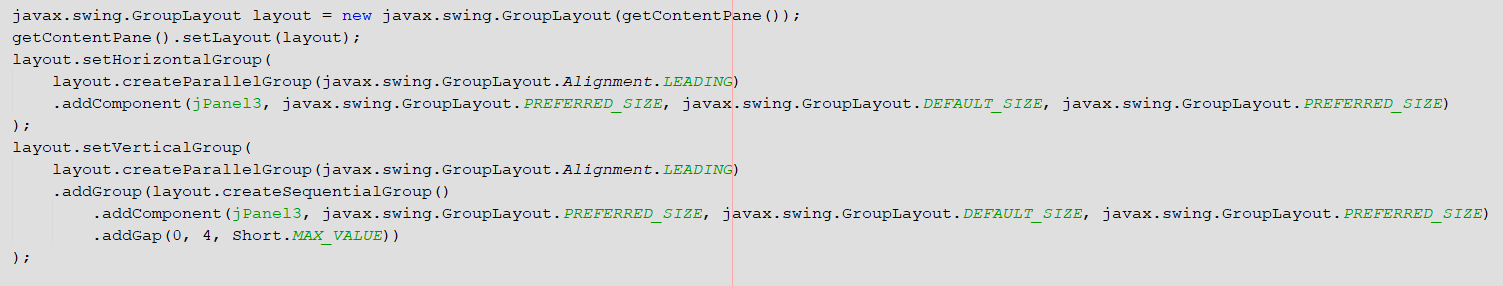


Figure 19: Example of GroupLayout

# 

# **JCalendar**

JCalendar is a java date chooser bean for graphically picking a date. JCalendar is composed of several other java beans like JdayChooser, JmonthChooser and a JYearChooser. In this version of the project, we use the JDayChooser so that we can access the date when the book is issued and when the book is returned.

However, we have to add file jar of it to the Library so that we can get the method **getDayeditor** and **getUiComponents().**

# **5. Singleton class**

This project using Singleton – design pattern to create a message after user login application successfully. Singleton class is made by PrintInfo class which is responsible to create an object while making sure that only single object gets created.

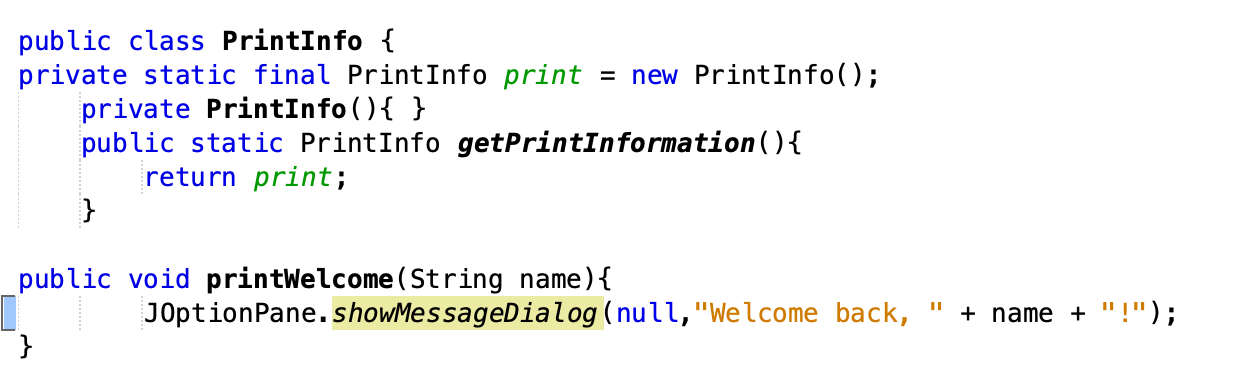


Figure 20: Singleton class

# **6. Database code**

In order to access to the Microsoft SQL Server, we have javaconnect class. In this class, we insert our database server URL, username and password to connect to the database system.

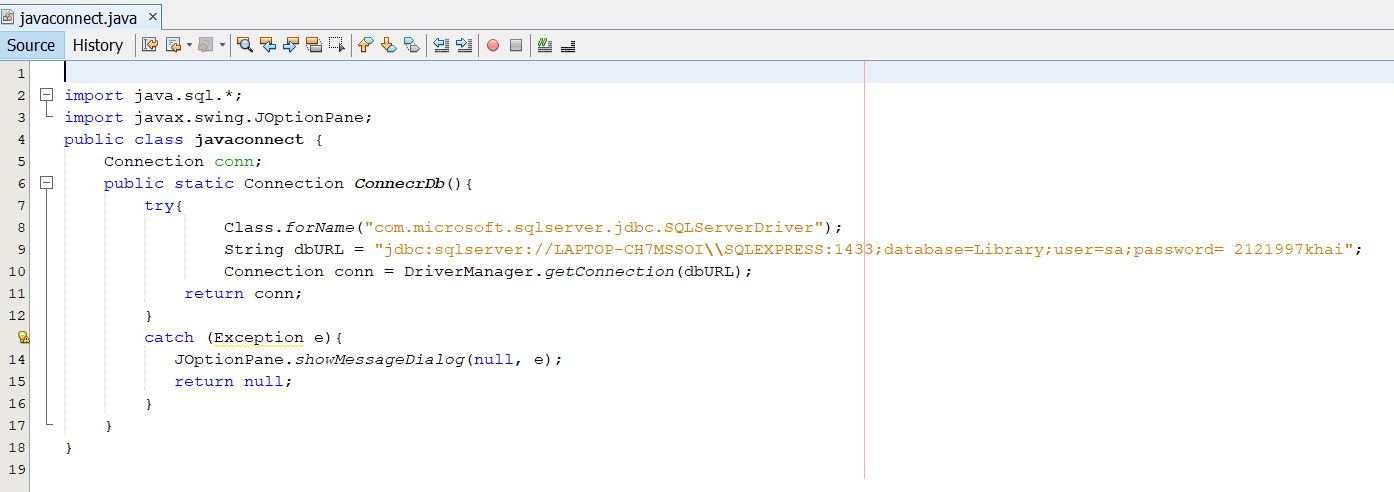


Figure 21: Connect to database server

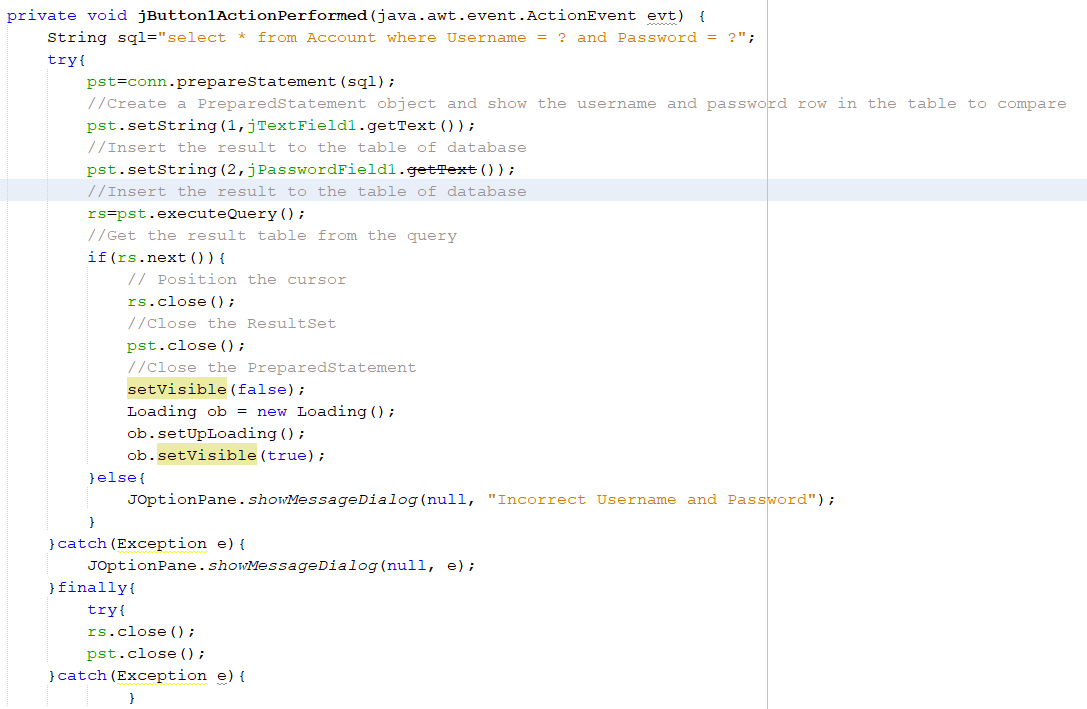


Figure 22: Explain code

# **7. Some experiences when using Microsoft SQL Server**

Before finding out Microsoft SQL Server, we found and tried a number of database tools like: Mongo DB, SQL Management, SQLite, … However, Microsoft SQL Server is easier to use for a simple project. But, there are still many problems in the progress like we

cannot access to the database system until we add file jar of Microsoft JDBC Driver for SQL Server to the Library file of the project folder. Also, we have to set up TCP/IP.

However, the issue we are now facing is we cannot connect server to client server due to the TCP/IP problem. As the result, we decided to create the database table when we had meeting together.

# **8. References**

[www.docs.oracle.com](http://www.docs.oracle.com)

[www.javapoint.com](http://www.javapoint.com)

[www.stackoverflow.com](http://www.stackoverflow.com)

[www.geeksforgeeks.org](http://www.geeksforgeeks.org)

<https://www.ibm.com/support/knowledgecenter/SSEPGG_11.5.0/com.ibm.db2.luw.apdv.java.doc/src/tpc/imjcc_tjvpsxqu.html>

<https://www.tutorialspoint.com/design_pattern/singleton_pattern.htm>