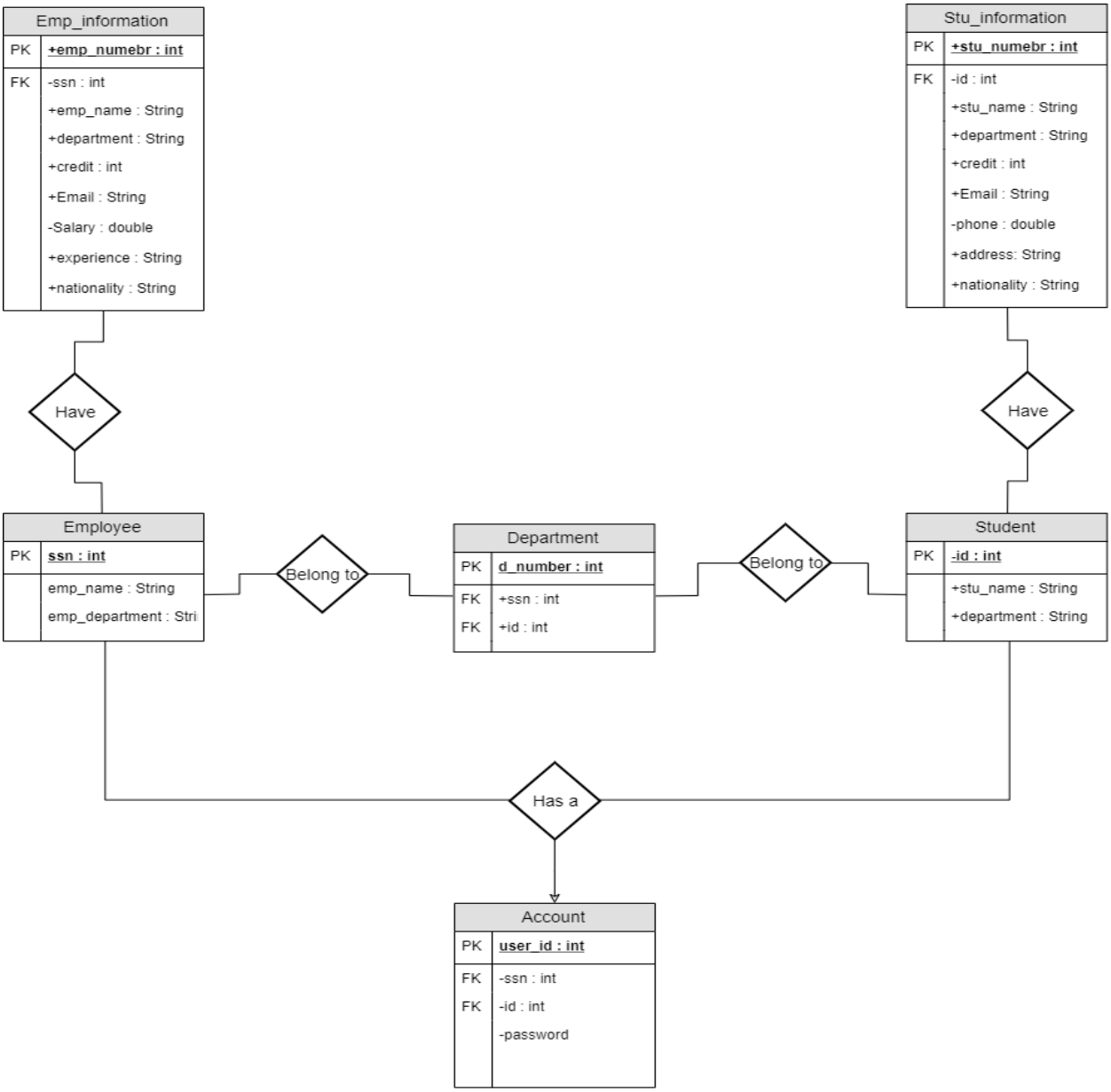
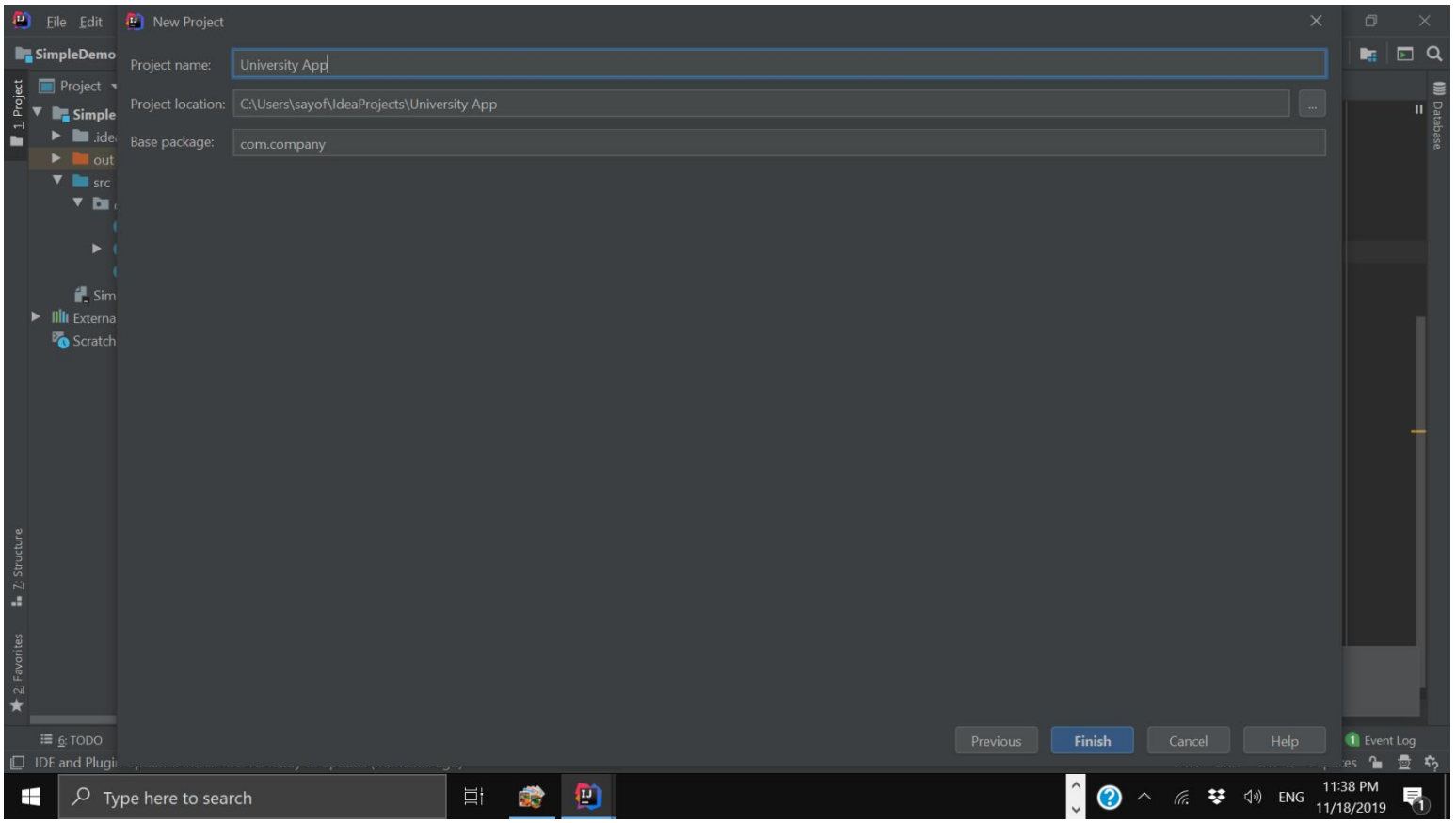
Welcome to my project that called " **University App "** that enable many features for the faculty's members and students which will be very helped for them.

- The application requires to implement:
    1. **IntelliJ IDEA:** It will be used as preferred integrated development environment (IDE) for developing Java programs.
    2. **MySQL:** In order to store our data more efficiency.

1. First of all, to be clear at the begging we would like to introduce what kind of data we will deal with, and how would the date base of " **University App**" will be?
    - Our ER-diagram will be as following:



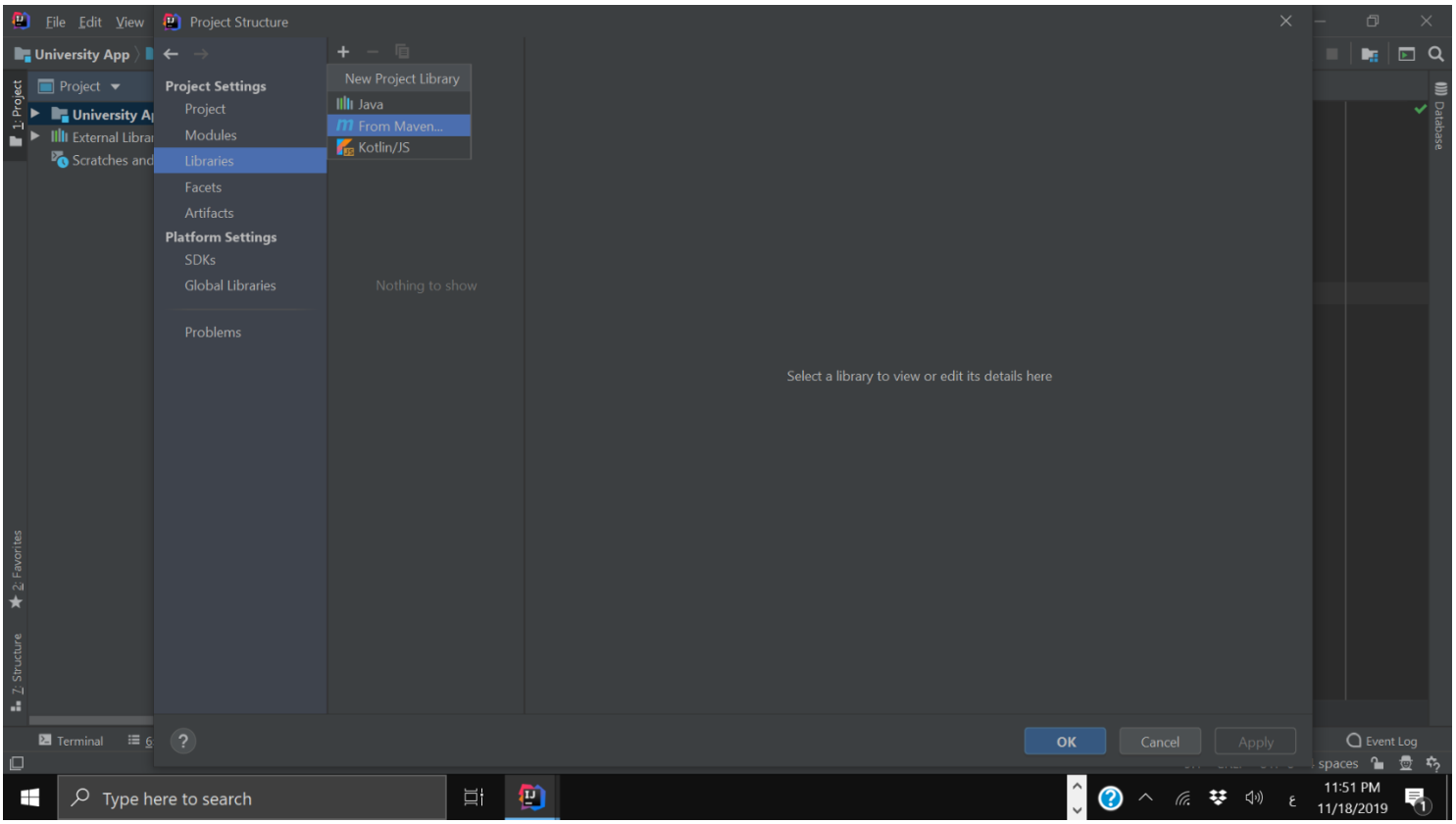
2. Now we will go to IntelliJ and create a new project (file-> new-> project). Our project will be named as " **University App".**



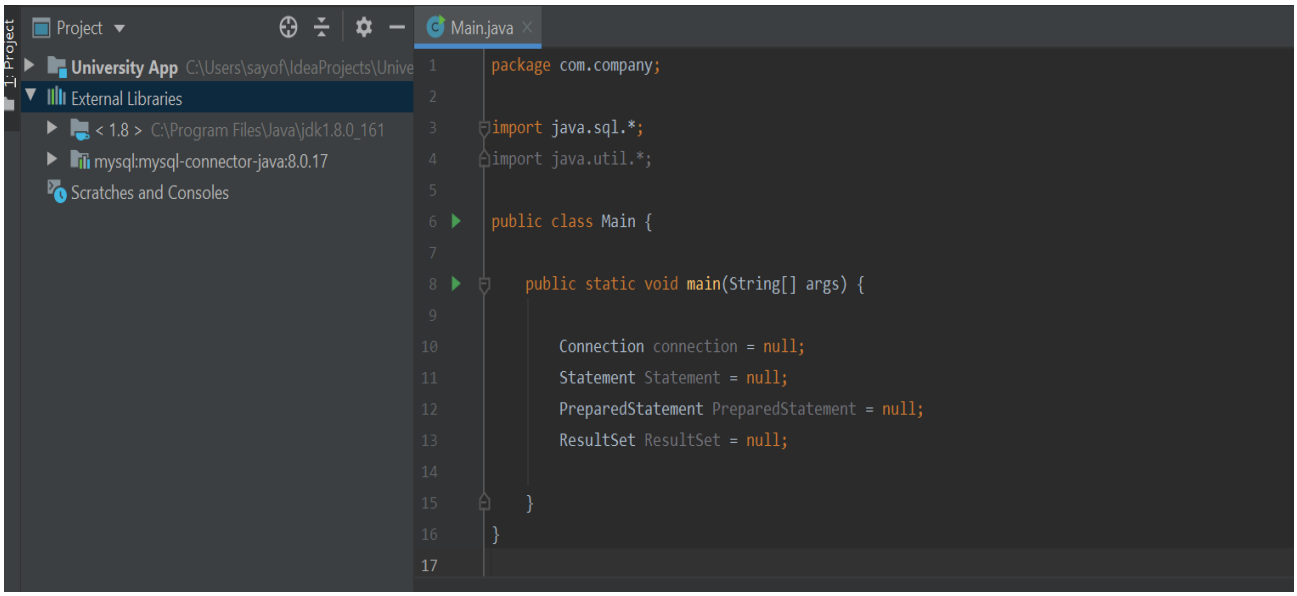
3. Next step is to add the Connector/J for connecting to MySQL database. This will be done inside IntelliJ project.

4. Let's create a new Command Line App project in IntelliJ through the wizard.

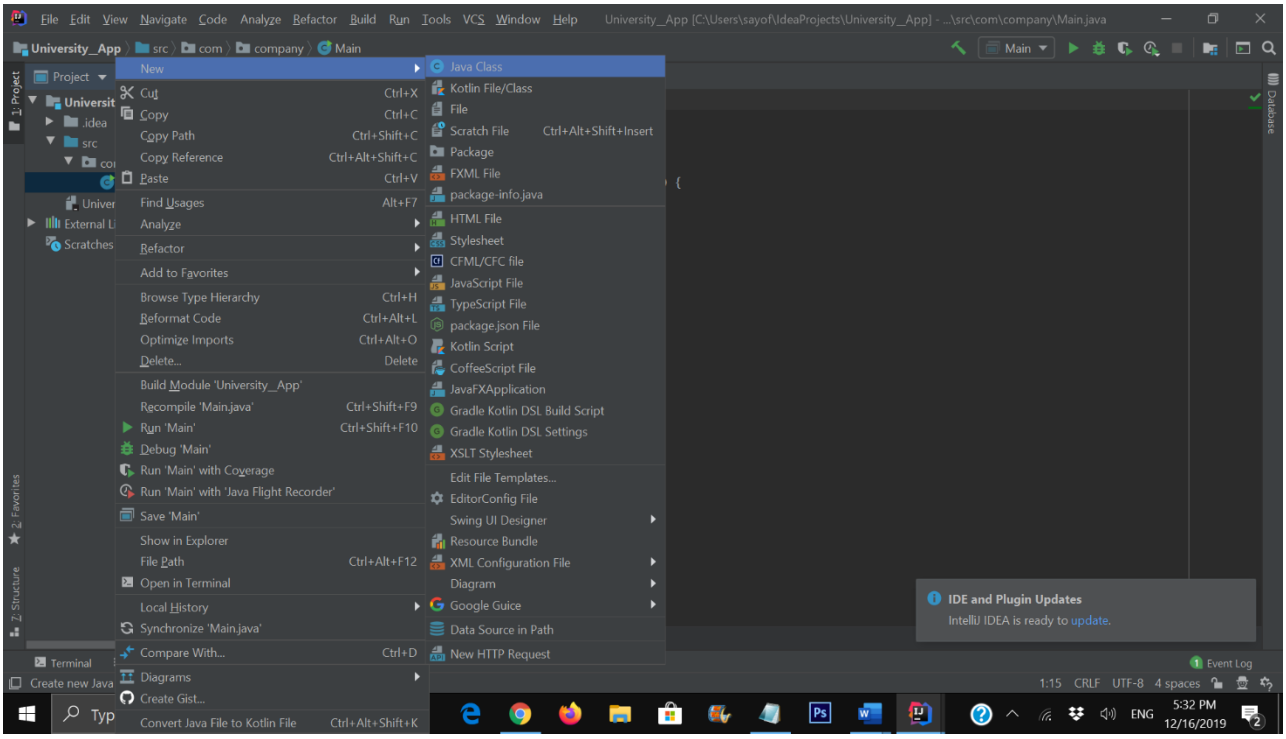
5. Go to (File-> Project Structure -> Libraries -> click + -> From maven). As showing below:



6. Type **mysql:mysql-connector-java:8.0.17** in the search box for using 8.0.17 version of the connector. Change the version accordingly to match the installation of MySQL.
7. Now we have to import some Libraries from SQL. It's such easier if we import all things that inside of SQL, so we'll write " import java.sql.*; ".
8. let's move on to connecting to the Database. We need to create some objects first.
    - **connection object:** to connected with the date base.
    - **Statement object:** to execute queries in date base.
    - **PreparedStatement object:** to execute the same or similar SQL statements repeatedly with high efficiency.
    - **ResultSet object:** to contains the results of executing an SQL query.



9. Now after we added the SQL library and took idea about the connection about sql server let's move to real programming.
10. We need to create class for the home page that will be an interface for the users. Pic below shows you how to create a new class .



11. Name it as HomePage.

```
12. extends JFrame implements ActionListener
```

13. After we create, we have to extend JFrame and implements ActionListener.

14. In fact, we don't have to implement ActionListener to our class but it's a good way organize your class and it will look better.

15. In case we implement ActionListener we have to override actionPerformed method.

16. Usually we will need this in every class in our project.

17. In this class we will design the first page which it will show to every who runs the application.

18. This page needs label to write the welcome sentence and two bottoms (one for Login and the other for exit).

19. In order to defined it we have to write:

```
 JButton bLogin, bExit;
JLabel l1;
```

20. And don't forget to override the actionPerformed method.

```
public class HomePage extends JFrame implements ActionListener {

    JButton bLogin, bExit;
    JLabel l1;



    //=========================(actionPerformed)======================
    @Override
    public void actionPerformed(ActionEvent a) {


        }
```

21. Now we will create the constrictor for this class

```
public class HomePage extends JFrame implements ActionListener {

    JButton bLogin, bExit;
    JLabel l1;



    //=========================(actionPerformed)=======================
    @Override
    public void actionPerformed(ActionEvent a) {


        }



    //=================================================================
    //======================(Constructor)==============================

    public HomePage() {


    }
```

22. Now we will write inside the constructor our code that we want to display it in this page.

```
//=============================================================
//====================(Constructor)============================

public HomePage() {

    setTitle("University application");
    setSize( i: 600,  i1: 600);
    setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    setLayout(null); // Absolute Layout
    setBackground(Color.BLACK);

    l1 = new JLabel( s: "Welcome To University Application");
    l1.setFont(new Font( s: "Tahoma", Font.PLAIN,  i1: 25));
    l1.setBounds( i: 100,  i1: 100,  i2: 400,  i3: 200);
    add(l1);
```

23. First thing we would have to do is to put a title for this frame by using setTitle() method.
24. Then we have to determine the size of this page by using setSize() method.
25. Then we will use setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE) method that is used to close your JFrame as well as Java process. Test it with a JFrame sample program without the setDefaultCloseOperaton. Close your Jframe by clicking the windows close. Check your task manager/top command, you will find the java process will be still running.
26. Now we have to specified the layout. The swing supports many layouts but we will write null it means absolute layout it is a simple x,y oriented layout. During layout, the live component is shown moving with a tooltip showing its current location or size.
27. After we defined the frame, we will move to what we want to write in our label.

```
l1 = new JLabel("Welcome To University
Application");
l1.setFont(new Font("Tahoma", Font.PLAIN, 25));
l1.setBounds(100, 100, 400, 200);
add(l1);
```

28. The setFont() method it gives us the freedom to choose the type and size of the font.
29. The setBounds(x, y, width, height) to specify the position and size of a GUI component.
30. Then we add our label to the frame.
31. Let's move to define a Jbottom.

```
bLogin = new JButton("Login");
bLogin.setBounds(320, 400, 150, 50);
bLogin.setFont(new Font("Tahoma", Font.PLAIN,
20));
add(bLogin);
```

32. After we set the bounds and specify the font we have to add ActionListener in order to put action for this bottom. The bottom is nothing but ActionListener.

```
bLogin.addActionListener(this);
```

33. The back bottom will be the same.

```
bExit = new JButton("Exit");
bExit.setBounds(100, 400, 150, 50);
bExit.setFont(new Font("Tahoma", Font.PLAIN,
20));
add(bExit);
bExit.addActionListener(this);
```

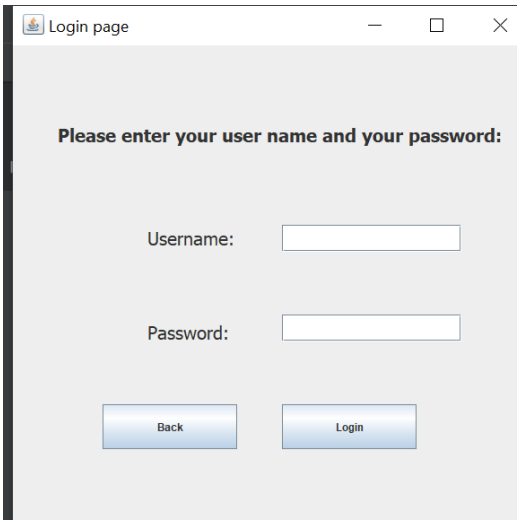
34. After we define every things we HAVE TO set these stuff visible by using setVisible() method.

```
setVisible(true);
```

35. Now let's move to actionPerformed() method where will put action for our bottom.

```java
@Override
public void actionPerformed(ActionEvent a) {


    if (a.getSource() == bLogin) {

        dispose();
        LoginPage login = new LoginPage();
        login.setVisible(true);



        } else if (a.getSource() == bExit) {
            System.exit(0);
        }
    }
```

36. As shown we will make if statement to determined which bottom is clicked.
37. The dispose() method will close the current window.
38. If the user clicked login it will create object from LoginPage and move there.
39. So, let's go to create a new class and name it as LoginPage.



40. this is a login page so we have to defined 3 label, 2 bottoms, a text filed and password filed.

```
JButton bLogin, bBcak;
JLabel l5, l6, l7;
JTextField textField;
private JPasswordField passwordField;
```

41. Inside the Constructor we will write:

```
this.setTitle("Login Page");
this.setSize(600, 600);
this.setDefaultCloseOperation(WindowConstants.EXI
T_ON_CLOSE);
this.setLayout(null); // Absolute Layout
this.setBackground(Color.BLACK);


setTitle("Login page");
setSize(600, 600);
setDefaultCloseOperation(WindowConstants.EXIT_ON_
CLOSE);
setLayout(null); // Absolute Layout


l5 = new JLabel("Please enter your user name and
your password: ");
l5.setFont(new Font("Tahoma", Font.BOLD, 20));
l5.setBounds(50, 50, 600, 100);
add(l5);


//=============================================
================
//======================(UserName)=============
================

l6 = new JLabel("Username: ");
l6.setBounds(150, 200, 200, 30);
l6.setFont(new Font("Tahoma", Font.PLAIN, 20));
add(l6);

textField = new JTextField();
textField.setBounds(300, 200, 200, 30);
```

```
add(textField);


//=================================================
==================
//=======================(Password)===============
===============

l7 = new JLabel("Password: ");
l7.setBounds(150, 300, 300, 40);
l7.setFont(new Font("Tahoma", Font.PLAIN, 20));
add(l7);

passwordField = new JPasswordField();
passwordField.setBounds(300, 300, 200, 30);
add(passwordField);


//=================================================
==================
//=========================(Login)===============
==================

bLogin = new JButton("Login");
bLogin.setBounds(300, 400, 150, 50);
add(bLogin);
bLogin.addActionListener(this);


//=================================================
==================
//=========================(Back)================
==================

bBcak = new JButton("Back");
bBcak.setBounds(100, 400, 150, 50);
add(bBcak);
bBcak.addActionListener(this);


setVisible(true);
```

42. And now let's move to actionPerformed() method where will put action for our bottoms.

```
//======================(ActionListener)===========
============

public void actionPerformed(ActionEvent a) {

    if (a.getSource()==bLogin){
        String userName = textField.getText();
        String password = passwordField.getText();
        try {
            Connection connection = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost
/university_db",
                    "root", "root");

            PreparedStatement st =
(PreparedStatement)
connection.prepareStatement("Select user_id,
password from account where user_id=? and
password=?");

            st.setString(1, userName);
            st.setString(2, password);
            ResultSet rs = st.executeQuery();
            if (rs.next()) {
                    dispose();
                    Account account = new
Account(Integer.parseInt(userName));
            } else {

JOptionPane.showMessageDialog(bLogin, "Wrong
Username & Password");
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
    } else if (a.getSource() == bBcak){
        dispose();
        HomePage home = new HomePage();
        home.setVisible(true);
    }
};
```

43. After the user has written his information and press login it will create two string variable one for user name and one for pass word.
44. Now we have to connected with the data base and check if the written information is correct or not.
45. In order to connect we will put the connection between try and catch.
46. Inside the try we will create object from connection class and in order to do that we have to import sql

```
import java.sql.*;
```

47. Inside the getConnetion() method we will write the path of our date base and user name and password.
48. Create object from PreparedStatement class.
49. Inside the Statement we will write sql quire to check if the user information matching or not.
50. If not it will shows a message that tells him "Wrong Username & Password"
51. If it matches it will create object from Account class to go there.
52. The function of account class is to specify the type of user if he/she is a student, doctor or dean.
53. This class dose not to extend or implements from anyone.
54. First of all, we have to save the user id to use it in different class, so we will dealer static variable

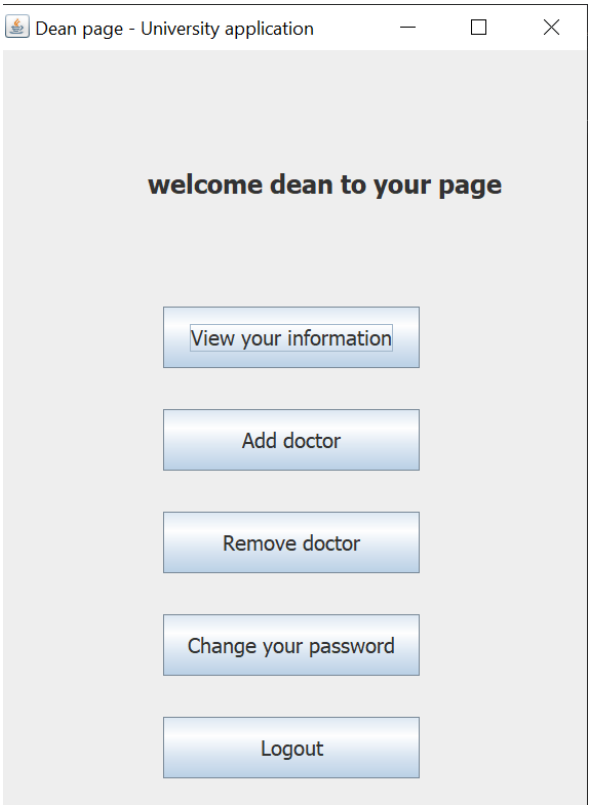
```
public static int saveID;
```

55. Then it goes inside the constructor.

```
//======================(Constructor)==============
================
```

```
public Account(int i){

    int user_id = i;

    if (i>99 && i < 1000) {
         saveID = i;
       new DoctorPage();


    } else if (i == 1000){
         saveID = i;
      new DeanPage();

    } else if(i>1999 && i < 2010) {
         saveID = i;
         new StudentPage();

    }


}
```

56. Assume that the user entered 1000.
57. It enter the if statement the save the id in our static variable then create a random object from DeanPage to go there.
58. Now let's move to DeanPage. The below pic shows the dean jobs.



59. This page needs a label and 5 bottoms.

```
JLabel l1;
JButton bAddDoctor, bRemoveDoctor,bViewInfo,
bLogout, bChange;
```

60. Now we have to call saveID from account page, so we will create a variable the contain the calling variable.
    The way of calling is call it by its class name.

```
int i = Account.saveID;
```

61. We will write our coding inside the constructor.

```
62. public DeanPage(){
        setTitle("Dean page - University
    application");
        setSize(600, 800);

    setDefaultCloseOperation(WindowConstants.EXIT_ON_
    CLOSE);
        setLayout(null); // Absolute Layout
```

```
    l1 = new JLabel("welcome dean to your page");
    l1.setBounds(150, 30, 400, 200);
    l1.setFont(new Font("Tahoma", Font.BOLD,
25));
    add(l1);



//================================================
====================

//==========================(Info)===============
====================
    bViewInfo = new JButton("View your
information");
    bViewInfo.setBounds(165, 250, 250, 60);
    bViewInfo.setFont(new Font("Tahoma",
Font.PLAIN, 20));
    add(bViewInfo);
    bViewInfo.addActionListener(this);




//================================================
====================
    // ==========================(Add
doctor)============================

    bAddDoctor = new JButton("Add doctor");
    bAddDoctor.setBounds(165, 350, 250, 60);
    bAddDoctor.setFont(new Font("Tahoma",
Font.PLAIN, 20));
    add(bAddDoctor);
    bAddDoctor.addActionListener(this);



//================================================
====================
```

```
    // ==========================(Remove
doctor)=========================

    bRemoveDoctor = new JButton("Remove doctor");
    bRemoveDoctor.setBounds(165, 450, 250, 60);
    bRemoveDoctor.setFont(new Font("Tahoma",
Font.PLAIN, 20));
    add(bRemoveDoctor);
    bRemoveDoctor.addActionListener(this);



//================================================
==================
    // ==========================(Change
password)======================

    bChange = new JButton("Change your
password");
    bChange.setBounds(165, 550, 250, 60);
    bChange.setFont(new Font("Tahoma",
Font.PLAIN, 20));
    add(bChange);
    bChange.addActionListener(this);



//================================================
==================
    //
==========================(Logout)==============
================

    bLogout = new JButton("Logout");
    bLogout.setBounds(165, 650, 250, 60);
    bLogout.setFont(new Font("Tahoma",
Font.PLAIN, 20));
    add(bLogout);
    bLogout.addActionListener(this);
```

```
        this.setVisible(true);
    }
```

63. In order to put action in our bottom we will go to actionPerformed().

```
@Override
public void actionPerformed(ActionEvent a) {


            if(a.getSource()==bAddDoctor){
                dispose();
                new AddDoctor();


            }else if (a.getSource()== bRemoveDoctor){
                dispose();
               new DeleteDoctor();


            }else if(a.getSource()==bViewInfo){
                dispose();
                new DoctorInfo();


            }else if(a.getSource()==bLogout){
                dispose();
                new LoginPage();

            }else if (a.getSource()==bChange){
                new ChangePassword(i);
            }
}
```

64. Each if statement start with closing the current window and create a random object where it belongs to.
65. Notice that the ChangePassword class took variable i. This i will let me now which member asked for changing password, and in data base will go to account table and check for this id and change his password.

This tutorial shows you the most important things in my application and if you are interesting to see the whole project you can download the project from the link below:
https://up.top4top.net/downloadf-1445o3dy91-zip.html

Finally, I'd like to thanks Dr. Khaled basher who teched me "**Advance computer programming**" for guiding us, inspire us and make us what we are today.