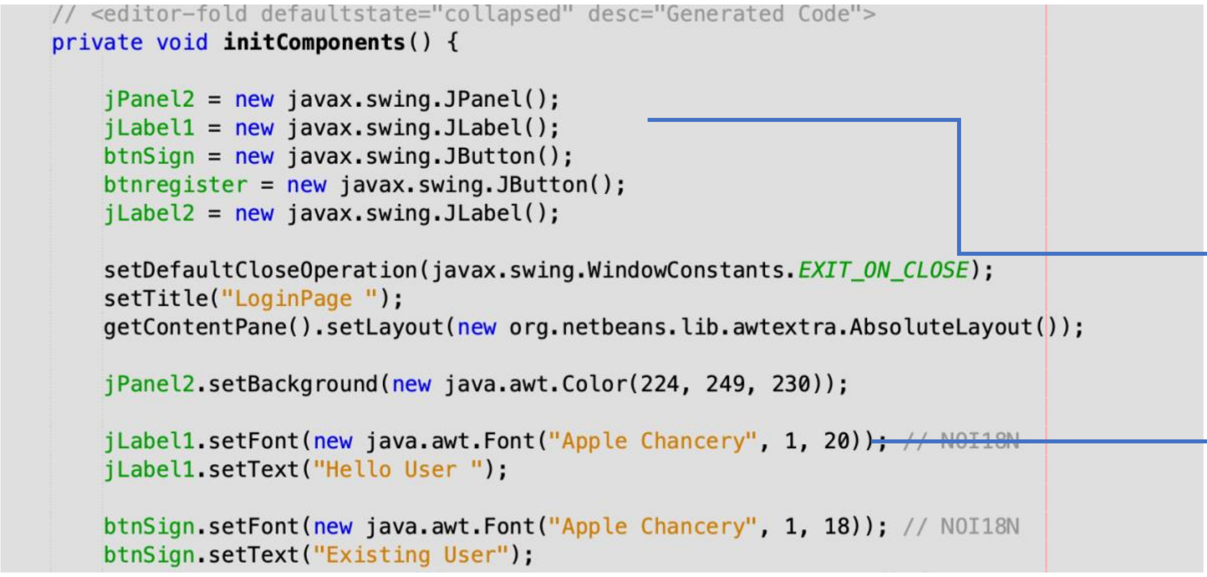
<u>Criteria C</u>

Table of Contents

To create this product, I used NetBeans, Java with Maven

GUI and interface design

Make my day app caters to students and their planning therefore the interface plays an important role to attract the student and motivate them to use the app every day, the screenshot of the code below has the code for the GUI elements added to the interface and their font style, size and background color.(1BestCsharp blog)

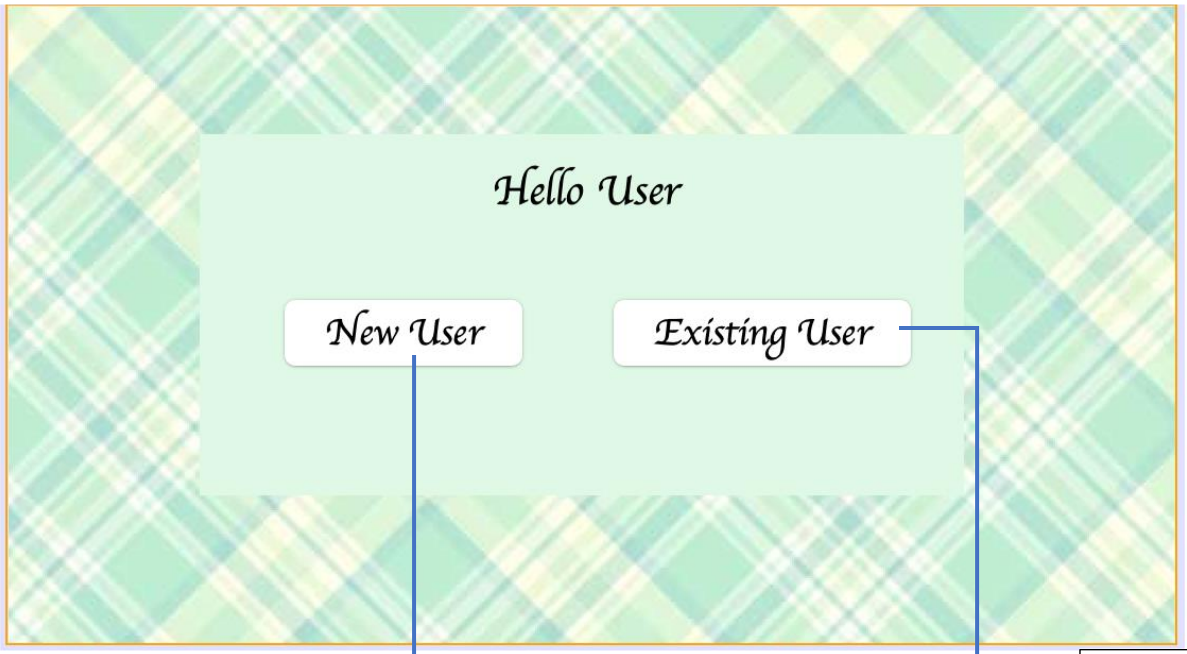


```
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jPanel2 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    btnSign = new javax.swing.JButton();
    btnregister = new javax.swing.JButton();
    jLabel2 = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    setTitle("LoginPage ");
    getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

    jPanel2.setBackground(new java.awt.Color(224, 249, 230));

    jLabel1.setFont(new java.awt.Font("Apple Chancery", 1, 20)); // NOI18N
    jLabel1.setText("Hello User ");

    btnSign.setFont(new java.awt.Font("Apple Chancery", 1, 18)); // NOI18N
    btnSign.setText("Existing User");
```

Code to add GUI elements

Code to manipulate text (font style, etc)

The outcome:

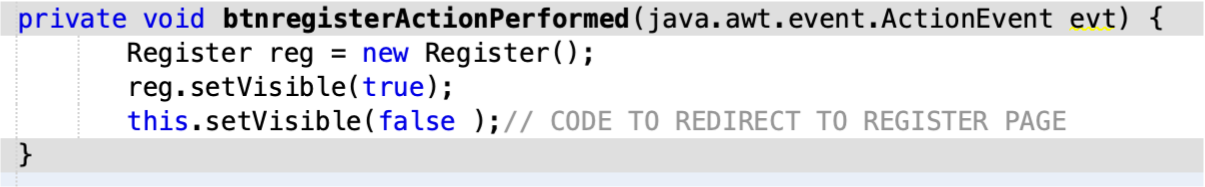


Icon to redirect existing user to the sign in page

Icon to redirect a new user to register page

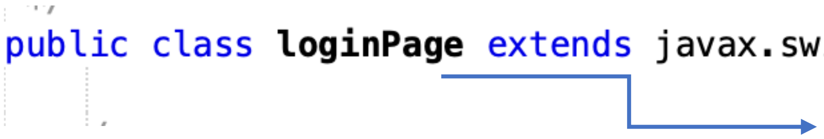
Object Instantiation

To perform the actions specified above ("new user") the action is performed by instantiating the object and then calling that class.



```
private void btnregisterActionPerformed(java.awt.event.ActionEvent evt) {
       Register reg = new Register();
       reg.setVisible(true);
       this.setVisible(false );// CODE TO REDIRECT TO REGISTER PAGE
}
```

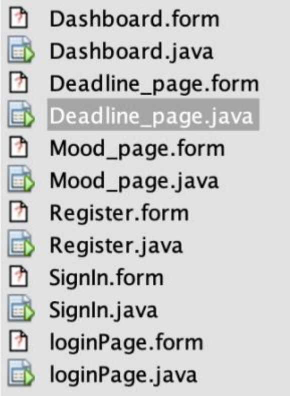
In the code above the action performed by "new user" button instantiates the register page class, this creates the object in memory and returns a reference to the newly created object. The method set visible has been inherited by the default java frame class provided by NetBeans:



```
public class loginPage extends javax.swing.JFrame {
```

Inheritance to the Javax.swing.JFrame class allows the login_page class to inherit the setvisible method

Sub-Programming

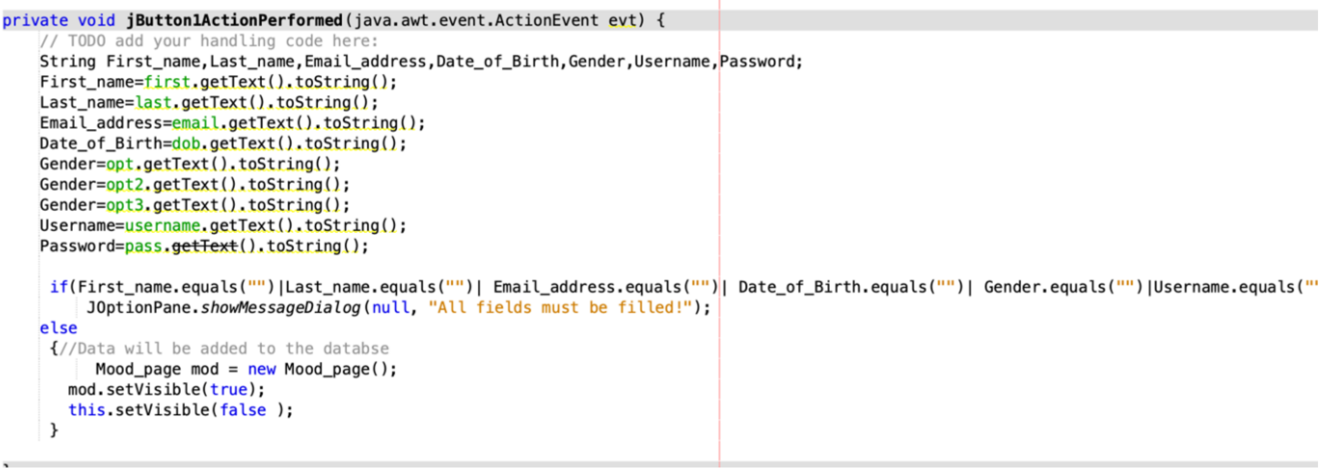


```
Dashboard.form
Dashboard.java
Deadline_page.form
Deadline_page.java
Mood_page.form
Mood_page.java
Register.form
Register.java
SignIn.form
SignIn.java
loginPage.form
loginPage.java
```

To make the overall coding for the app easier and decrease the unnecessary complexity for myself, I decided to divide the entire app into sub classes, every different interface had a different class this was incredibly beneficial, by this modularity confusion was avoided and I could make a clear link between each class. It also helped me reuse code and just change the variable name and if I made any error, my entire process was not affected but just that module thus debugging was a lot more easier.

IF- Else statements

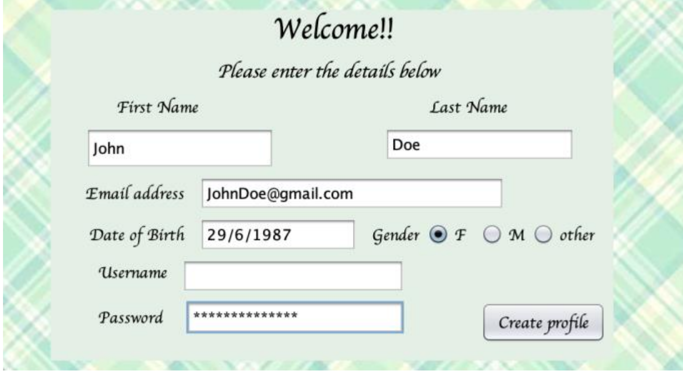
Throughout this application If statements are incredibly important, as the output is based on the user mood, thus for specific moods there are specific activity suggestions.



```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String First_name,Last_name,Email_address,Date_of_Birth,Gender,Username,Password;
    First_name=first.getText().toString();
    Last_name=last.getText().toString();
    Email_address=email.getText().toString();
    Date_of_Birth=dob.getText().toString();
    Gender=opt.getText().toString();
    Gender=opt2.getText().toString();
    Gender=opt3.getText().toString();
    Username=username.getText().toString();
    Password=pass.getText().toString();

     if(First_name.equals("")|Last_name.equals("")| Email_address.equals("")| Date_of_Birth.equals("")| Gender.equals("")|Username.equals("'
         JOptionPane.showMessageDialog(null, "All fields must be filled!");
    else
     {//Data will be added to the databse
          Mood_page mod = new Mood_page();
       mod.setVisible(true);
       this.setVisible(false );
     }
```

In the following code the IF-else statement has been used to ensure that no fields are left in the registration form, as the app is about the users personal life, the details asked for are crucial therefore to ensure that we have collected all the information this if else combination is added. (DIMITRIOU)

The process is:



Welcome!!

Please enter the details below

First Name: John

Last Name: Doe

Email address: JohnDoe@gmail.com

Date of Birth: 29/6/1987

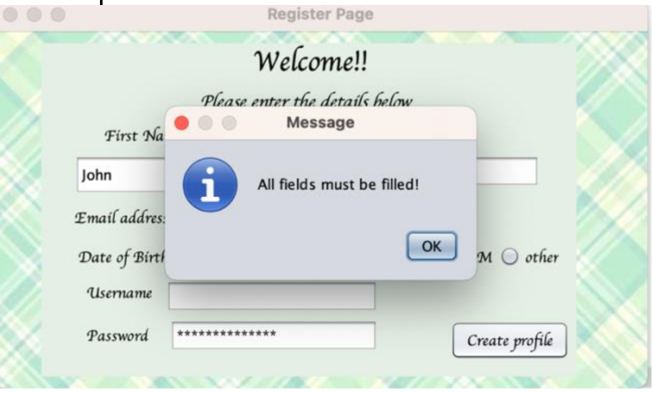
Gender ● F ○ M ○ other

Username:

Password: **************

In the following picture all the details are added except of the username

The output will be:



Welcome!!

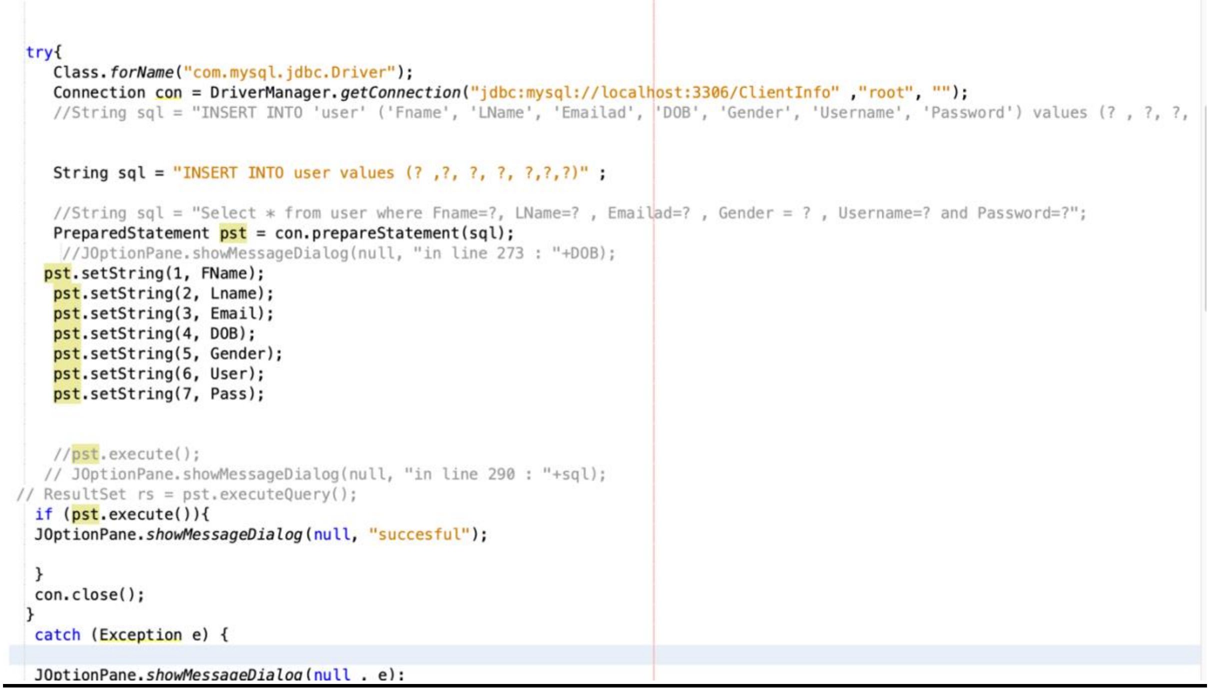
Please enter the details below

Message

All fields must be filled!

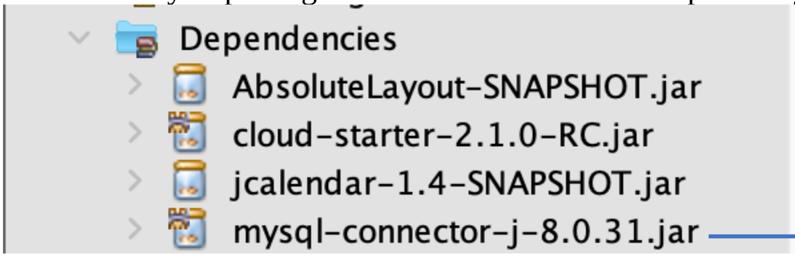
As one of the fields is empty a message will pop up to alert the user to fill all the details required

NetBeans Connectivity with MySQL



```
try{
   Class.forName("com.mysql.jdbc.Driver");
   Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ClientInfo" ,"root", "");
   //String sql = "INSERT INTO 'user' ('Fname', 'LName', 'Emailad', 'DOB', 'Gender', 'Username', 'Password') values (? , ?, ?,


   String sql = "INSERT INTO user values (? ,?, ?, ?, ?,?,?)" ;

   //String sql = "Select * from user where Fname=?, LName=? , Emailad=? , Gender = ? , Username=? and Password=?";
   PreparedStatement pst = con.prepareStatement(sql);
    //JOptionPane.showMessageDialog(null, "in line 273 : "+DOB);
  pst.setString(1, FName);
   pst.setString(2, Lname);
   pst.setString(3, Email);
   pst.setString(4, DOB);
   pst.setString(5, Gender);
   pst.setString(6, User);
   pst.setString(7, Pass);


   //pst.execute();
  // JOptionPane.showMessageDialog(null, "in line 290 : "+sql);
// ResultSet rs = pst.executeQuery();
 if (pst.execute()){
 JOptionPane.showMessageDialog(null, "succesful");

 }
 con.close();
}
 catch (Exception e) {

 JOptionPane.showMessageDialog(null , e);
```

The code above is a separate class created to establish connection of java and MySQL, this has been possible with the connector JAR file that was externally downloaded from MYSQLs website, as I am working with Java with Maven, instead of libraries I established this connection by importing this JAR file as a form of dependency:

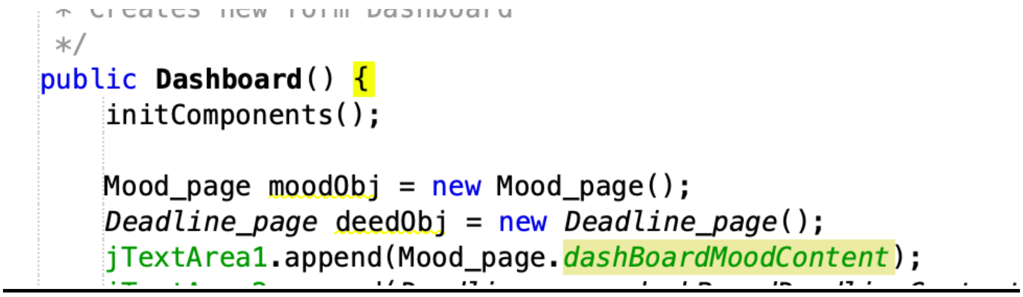


Dependencies
AbsoluteLayout-SNAPSHOT.jar
cloud-starter-2.1.0-RC.jar
jcalendar-1.4-SNAPSHOT.jar
mysql-connector-j-8.0.31.jar

The connector JAR file

In the code I created a method in the same class called “con” which is made using Java’s inbuilt method of getting a connection. The URL “jdbc: mysql://localhost:3306/ClientInfo" ,"root", ""” is the URL used to connect to the database where Clientinfo is the name of the database. The method “pst.setString” is again an inbuilt method used to input the data entered by the client into the database.(Chaudhary)

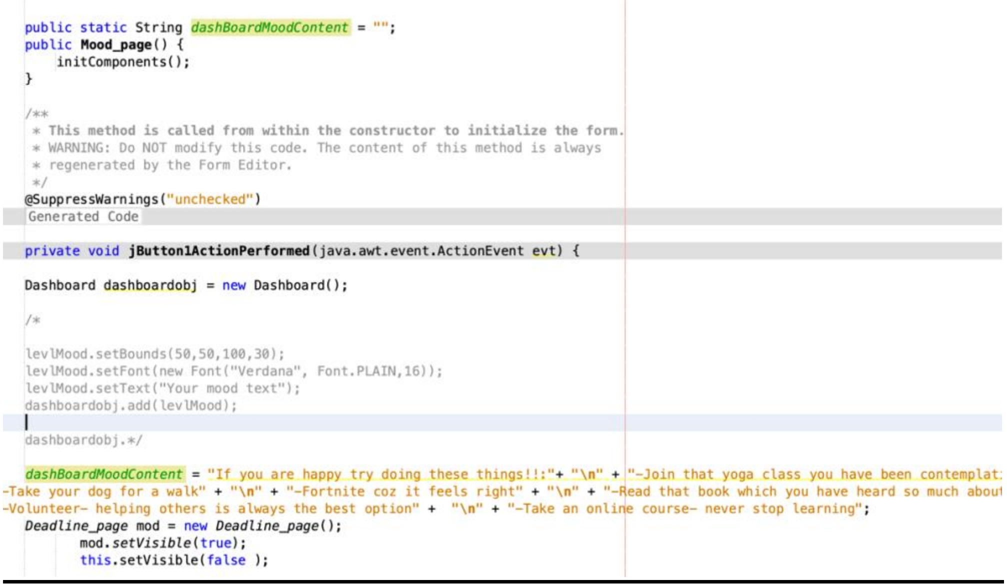
Encapsulation



```
* Creates new form Dashboard
*/
public Dashboard() {
    initComponents();

    Mood_page moodObj = new Mood_page();
    Deadline_page deedObj = new Deadline_page();
    jTextArea1.append(Mood_page.dashBoardMoodContent);
```

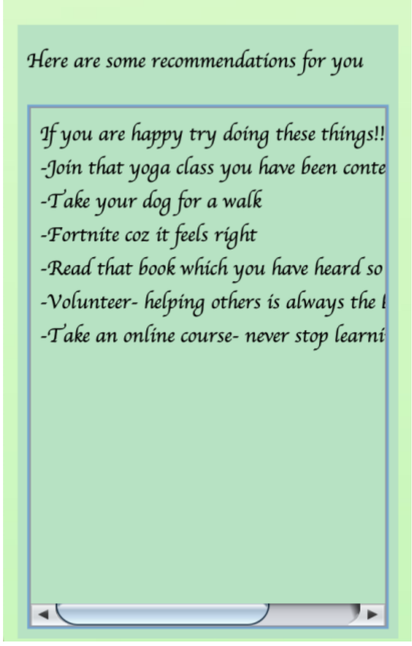
To ensure that the client gets recommendations based on the mood they have selected I have connected the deadline and mood page classes, this is done through encapsulations where I called the object of mood class and then use the append function by java to output the predetermined activity suggestions.(w3schools)

Calling method for mood suggestions



```
public static String dashBoardMoodContent = "";
public Mood_page() {
    initComponents();
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
 Generated Code

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

Dashboard dashboardobj = new Dashboard();

/*

levlMood.setBounds(50,50,100,30);
levlMood.setFont(new Font("Verdana", Font.PLAIN,16));
levlMood.setText("Your mood text");
dashboardobj.add(levlMood);

dashboardobj.*/

dashBoardMoodContent = "If you are happy try doing these things!!:"+ "\n" + "-Join that yoga class you have been contemplat
-Take your dog for a walk" + "\n" + "-Fortnite coz it feels right" + "\n" + "-Read that book which you have heard so much abou
-Volunteer- helping others is always the best option" +  "\n" + "-Take an online course- never stop learning";
Deadline_page mod = new Deadline_page();
        mod.setVisible(true);
        this.setVisible(false );
```

To display the recommended activities DashBoardMoodconetnt which was called from the dashboard page is initialized in the mood page as an empty string first, as the client selects each option, the string is customized to have that mood suggestions, for example here, the code has the working for action performed when the client chooses the mood as happy, here the string DashBoardMoodconetnt which was empty is now filled with values fetched from the data dictionary made in criteria B.

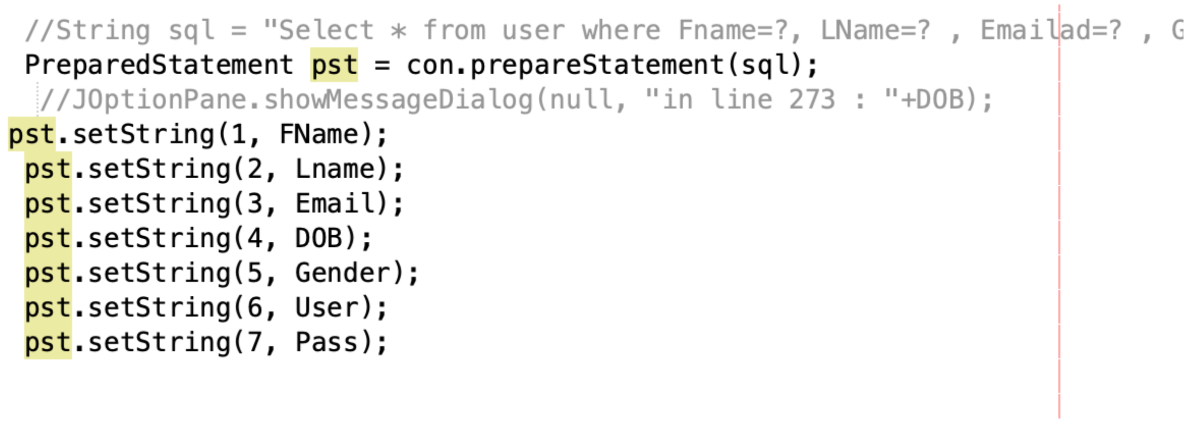


Here are some recommendations for you

If you are happy try doing these things!!
-Join that yoga class you have been conte
-Take your dog for a walk
-Fortnite coz it feels right
-Read that book which you have heard so
-Volunteer- helping others is always the l
-Take an online course- never stop learni

The picture here is the snippet of the code run with the client choosing the mood as happy. The suggestions are shown likewise.

Storing values in the database

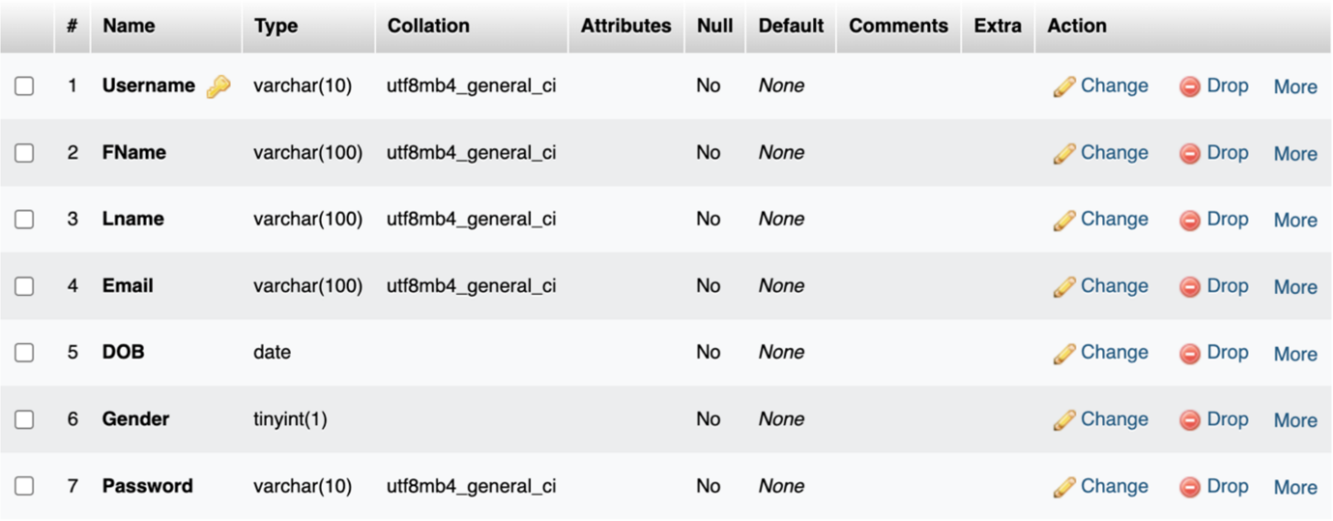


```
 //String sql = "Select * from user where Fname=?, LName=? , Emailad=? , G
 PreparedStatement pst = con.prepareStatement(sql);
  //JOptionPane.showMessageDialog(null, "in line 273 : "+DOB);
pst.setString(1, FName);
 pst.setString(2, Lname);
 pst.setString(3, Email);
 pst.setString(4, DOB);
 pst.setString(5, Gender);
 pst.setString(6, User);
 pst.setString(7, Pass);
```

Following code was used to ensure that the data inputted by user is stored in the database table. The function *getText* is an inbuilt function that NetBeans offers which is used to retrieve the information the client gives. Moreover If-else statement has been used to show a message to the user, if all the fields are full of appropriate data an account created message pops up or error will be shown (similar to the previous demonstration).

External Connectivity with MYSQL

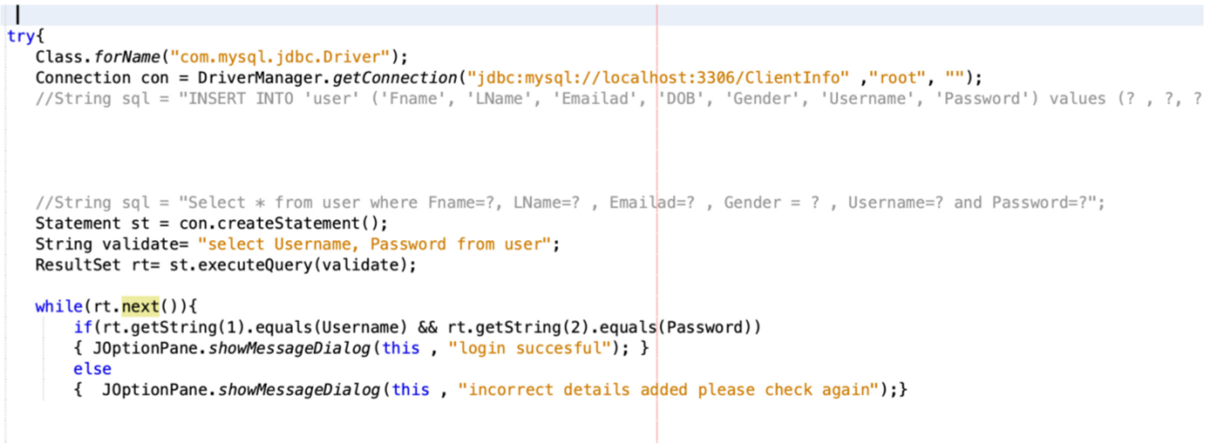
To externally establish connection with MYSQL I used phpMyAdmin, this software service was beneficial to connect to the MYSQL server



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
□	1	**Username** 🔑	varchar(10)	utf8mb4_general_ci		No	*None*			Change	Drop	More
□	2	**FName**	varchar(100)	utf8mb4_general_ci		No	*None*			Change	Drop	More
□	3	**Lname**	varchar(100)	utf8mb4_general_ci		No	*None*			Change	Drop	More
□	4	**Email**	varchar(100)	utf8mb4_general_ci		No	*None*			Change	Drop	More
□	5	**DOB**	date			No	*None*			Change	Drop	More
□	6	**Gender**	tinyint(1)			No	*None*			Change	Drop	More
□	7	**Password**	varchar(10)	utf8mb4_general_ci		No	*None*			Change	Drop	More

As seen in the interface above the columns were made based on the requirements from the register form with every variable having its own data type. We can also see that username has a key sign next to it, this is because the username is a primary key meaning it is unique for every client and helps in identifying that client

Validation



```
try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ClientInfo" ,"root", "");
    //String sql = "INSERT INTO 'user' ('Fname', 'LName', 'Emailad', 'DOB', 'Gender', 'Username', 'Password') values (? , ?, ?



    //String sql = "Select * from user where Fname=?, LName=? , Emailad=? , Gender = ? , Username=? and Password=?";
    Statement st = con.createStatement();
    String validate= "select Username, Password from user";
    ResultSet rt= st.executeQuery(validate);

    while(rt.next()){
        if(rt.getString(1).equals(Username) && rt.getString(2).equals(Password))
        { JOptionPane.showMessageDialog(this , "login succesful"); }
        else
        {  JOptionPane.showMessageDialog(this , "incorrect details added please check again");}
```

For the login of user, I again imported the java functions for connectivity with database tables however here rather than inserting data, data was fetched from the table to check whether the inputted username and password was correct. I ran a while loop to which find and checks for the next data inputted by the scanner, if there is data inputted an if condition is implemented which compares the strings with the. equals function and if they are equal than a login successful message comes up or else the client is asked to check the details again.(DIMITRIOU)

Bibliography

1BestCsharp blog, director. *YouTube*, YouTube, 14 Aug. 2017, www.youtube.com/watch?v=XAowXcmQ-kA.

Chaudhary, Abhishek, director. *YouTube*, YouTube, 14 Jan. 2022, www.youtube.com/watch?v=-nup6K1z4Po. Accessed 24 Feb. 2023.

DIMITRIOU, KOSTAS. *Core Computer Science: For the IB Diploma Program*. EXPRESS PUBLISHING, 2016.

w3schools. "Java Encapsulation." *Java Encapsulation and Getters and Setters*, www.w3schools.com/java/java_encapsulation.asp#:~:text=The%20meaning%20of%20Encapsulation%2C%20is,value%20of%20a%20private%20variable.

Words: 951