

Title

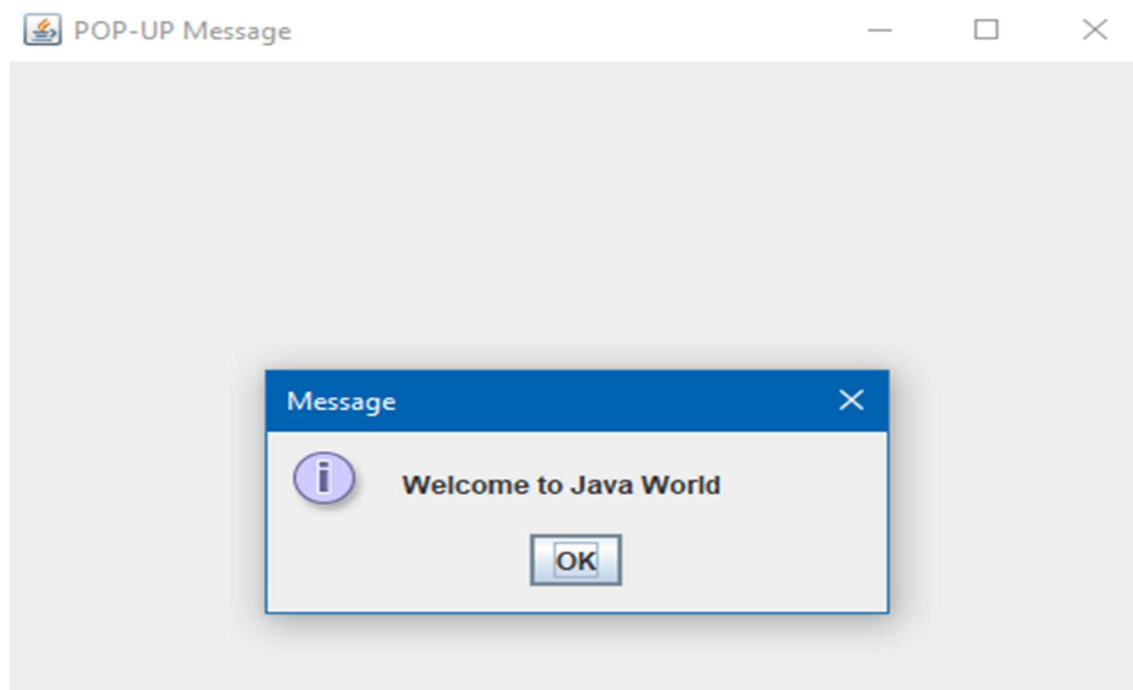
Create a GUI to show “Welcome to Java World” pop up message when GUI is launched.

Objective

To learn how to create a GUI that displays Welcome to Java World.

Program code:

```
import javax.swing.*;
public class Main {
    public static void main(String[] args) {
        JFrame frame = new JFrame("POP-UP Message");
        frame.setSize(500,500);
        frame.setVisible(true);
        frame.setLayout(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JOptionPane.showMessageDialog(frame,"Welcome to Java
World");
    }
}
```

OUTPUT:

Discussion

The practice for making a GUI that displays the message as passed in the program was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a GUI and display a pop up message.

Title

Create a GUI having two textbox that adds both values when button is clicked.

Objective

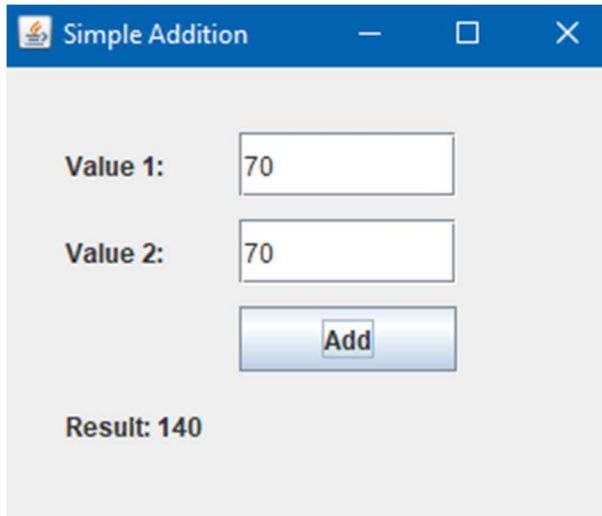
To learn how to create a GUI with two textbox and a button which performs addition when button is clicked.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class Main {
    public static void main(String[] args) {
        JFrame f1 = new JFrame("Simple Addition");
        f1.setSize(300, 250);
        f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f1.setLayout(null);
        f1.setVisible(true);
        JLabel labelnum1 = new JLabel("Value 1:");
        labelnum1.setBounds(30, 30, 80, 30);
        f1.add(labelnum1);
        JTextField num1 = new JTextField();
        num1.setBounds(110, 30, 100, 30);
        f1.add(num1);
        JLabel labelnum2 = new JLabel("Value 2:");
        labelnum2.setBounds(30, 70, 80, 30);
        f1.add(labelnum2);
        JTextField num2 = new JTextField();
        num2.setBounds(110, 70, 100, 30);
        f1.add(num2);
        JButton add = new JButton("Add");
        add.setBounds(110, 110, 100, 30);
        f1.add(add);
        JLabel result = new JLabel("Result:");
        result.setBounds(30, 150, 200, 30);
        f1.add(result);
        add.addActionListener(new ActionListener() {
            @Override
```

```
public void actionPerformed(ActionEvent e) {  
    try {  
        int val1 = Integer.parseInt(num1.getText());  
        int val2 = Integer.parseInt(num1.getText());  
        int sum = val1 + val2;  
        result.setText("Result: " + sum);  
    } catch (NumberFormatException ex) {  
        result.setText("Please enter valid integers.");  
    }  
}  
});  
}
```

OUTPUT:



Discussion

The practice for making a GUI consisting two textbox which took inputs & performed addition on button click in the program was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a GUI with textbox and events and performing desired actions on click.

Title

Create a GUI having two buttons and swap the text when button is clicked.

Objective

To learn how to create a GUI with two buttons and button click swaps the value among the buttons on click.

Program Code:

```
import javax.swing.*;
import java.awt.event.*;

public class Main {
    public static void main(String[] args) {
        JFrame f1=new JFrame("Swap");
        f1.setLayout(null);
        f1.setSize(500,500);
        f1.setVisible(true);
        f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JButton b1=new JButton("Ayush");
        b1.setBounds(100,150,100,25);

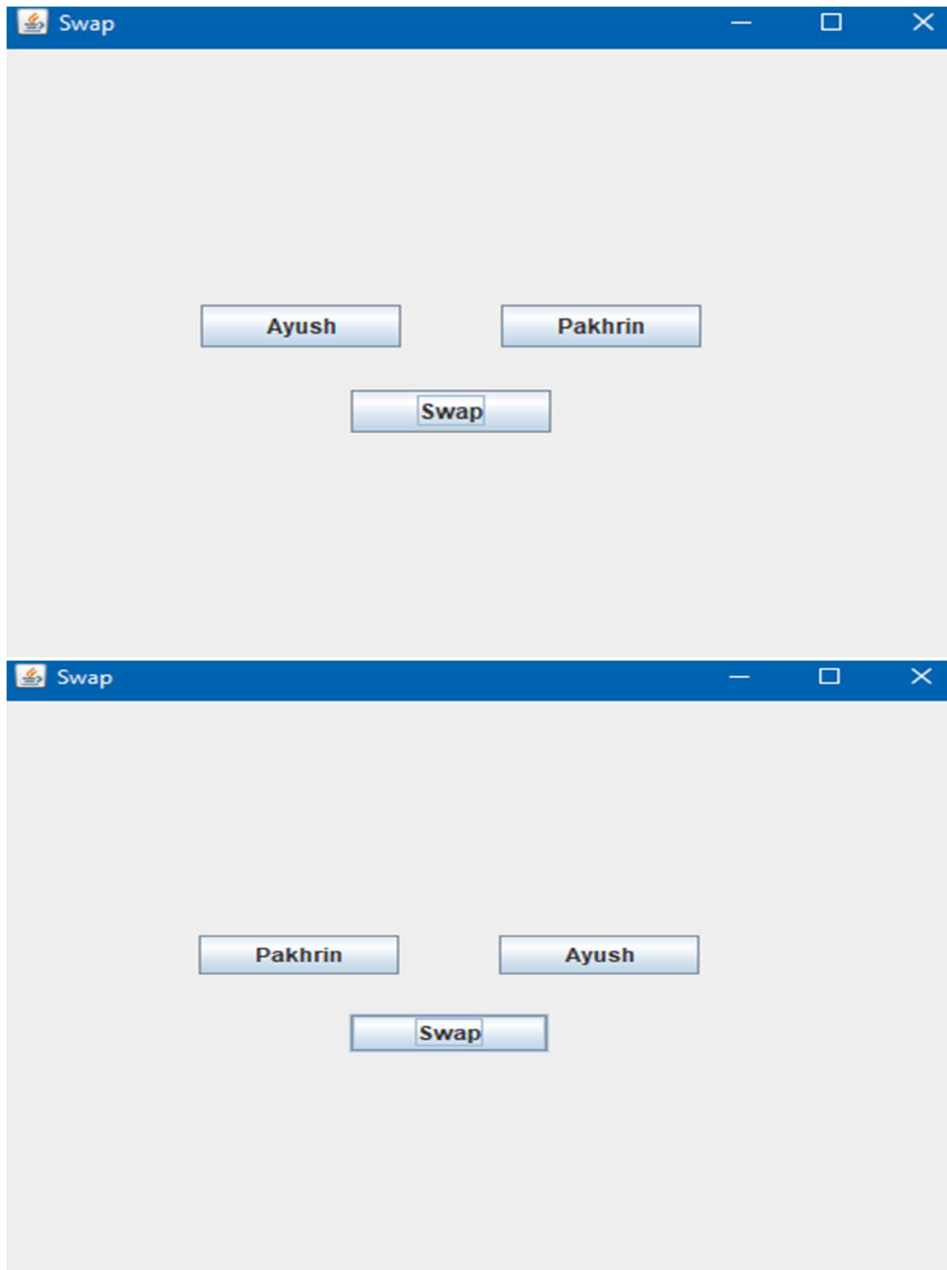
        JButton b2=new JButton("Pakhrin");
        b2.setBounds(250,150,100,25);

        JButton swap=new JButton("Swap");
        swap.setBounds(175,200,100,25);
```

```
f1.add(b1);  
f1.add(b2);  
f1.add(swap);
```

```
swap.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        String data1= b1.getText();  
        String data2= b2.getText();  
        String temp="";  
  
        temp=data1;  
        data1=data2;  
        data2=temp;  
  
        b1.setText(data1);  
        b2.setText(data2);  
  
    }  
});  
}  
}
```

OUTPUT:



Discussion

The practice for making a GUI having two buttons which swapped each other values onclick was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a GUI consisting buttons which swapped values onclick.

Title

Create a GUI of a calculator having four button(+,-,*,/) and perform related arithmetic operation on values when button is clicked.

Objective

To learn how to create a GUI of a calculator having all arithmetic operations performed on values when button is clicked.

Program Code:**Main.java**

```
public class Main {  
    public static void main(String[] args) {  
        Calculator calc=new Calculator();  
    }  
}
```

Calculator.java

```
import javax.swing.*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
class Calculator implements ActionListener {  
    public JFrame frame;  
    public JLabel l1,l2,result;  
    public JTextField val1,val2;  
    public JButton add,sub,mul,div;  
  
    public Calculator(){  
        frame=new JFrame("Calculator");  
        frame.setSize(500,500);  
        frame.setVisible(true);  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.setLayout(null);  
  
        l1=new JLabel("Value 1:");  
        l1.setBounds(100,100,75,25);
```

```
val1=new JTextField();  
val1.setBounds(160,100,100,25);
```

```
l2=new JLabel("Value 2:");  
l2.setBounds(100,150,75,25);
```

```
val2=new JTextField();  
val2.setBounds(160,150,100,25);
```

```
result=new JLabel("Result:");  
result.setBounds(100,200,100,25);
```

```
add=new JButton("+");  
add.setBounds(100,225,50,25);
```

```
sub=new JButton("-");  
sub.setBounds(160,225,50,25);
```

```
mul=new JButton("*");  
mul.setBounds(220,225,50,25);
```

```
div=new JButton("/");  
div.setBounds(280,225,50,25);
```

```
frame.add(l1);  
frame.add(l2);  
frame.add(val1);  
frame.add(val2);  
frame.add(result);  
frame.add(add);  
frame.add(sub);  
frame.add(mul);  
frame.add(div);
```

```
add.addActionListener(this);  
sub.addActionListener(this);  
mul.addActionListener(this);
```

```

        div.addActionListener(this);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            Double firstVal = Double.parseDouble(val1.getText());
            Double secondVal = Double.parseDouble(val2.getText());
            Double res=0.0;
            switch (e.getActionCommand()){
                case "+":
                    res = firstVal + secondVal;
                    break;
                case "-":
                    res = firstVal - secondVal;
                    break;
                case "*":
                    res = firstVal * secondVal;
                    break;
                case "/":
                    if (secondVal != 0) {
                        res = firstVal / secondVal;
                    } else {
                        JOptionPane.showMessageDialog(null, "Cannot divide by
zero");
                    }
                    return;
                }
                break;
            }
            result.setText("Result: " + res);
        }
        catch (Exception ex){
            System.out.println("Enter Valid Number!");
        }
    }
}

```

OUTPUT:

Calculator

Value 1:

Value 2:

Result:

Calculator

Value 1:

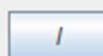
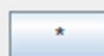
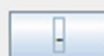
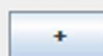
Value 2:

Result: 65.0

Value 1:

Value 2:

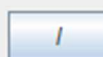
Result: 35.0

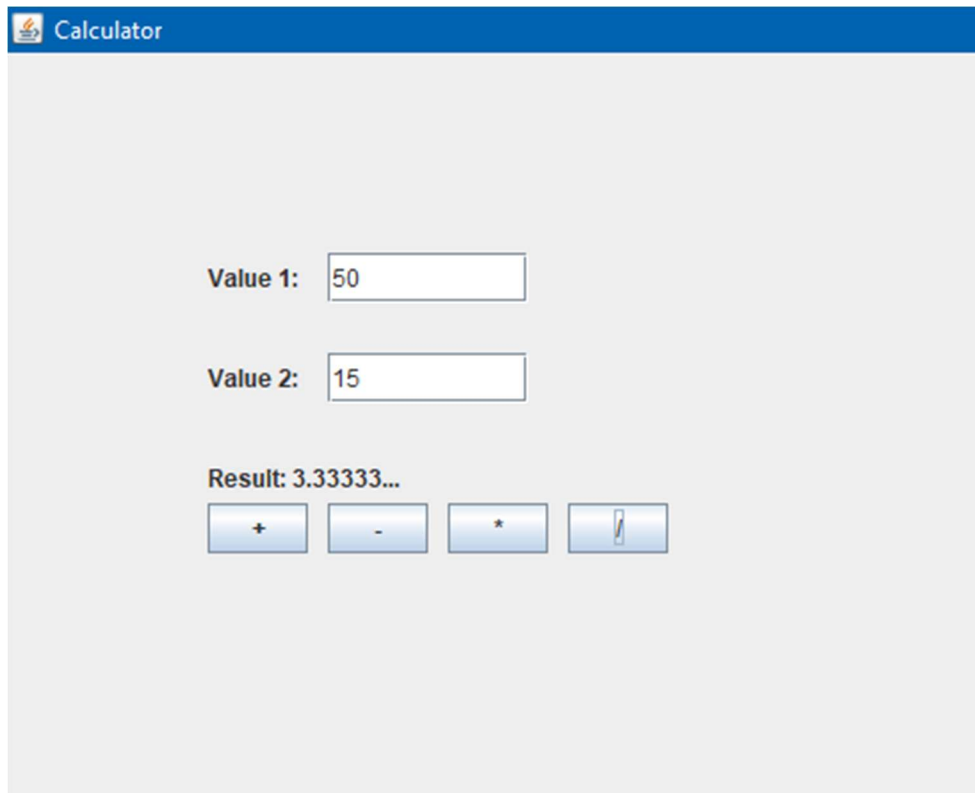


Value 1:

Value 2:

Result: 750.0





Discussion

The practice for making a GUI of a calculator which performed arithmetic operations on values when button clicked was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a GUI of a calculator and performed desired arithmetic operations.

Title

Create a simple calculation that takes input in a box and display results in same box.

Objective

To learn how to create a GUI of a calculator having all input values in a box which is passed on click of input value buttons and arithmetic operations is performed on the button clicks and result displayed.

Program Code:

Main.java

```
public class Main {  
    public static void main(String[] args) {  
        CalculatorGUI calc=new CalculatorGUI();  
    }  
}
```

CalculatorGUI.java

```
import javax.swing.*.*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class CalculatorGUI implements ActionListener {  
    JFrame f1;  
    JTextField display;  
    JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,plus,minus,mul,div,zero,clear,res;  
    double num1 = 0, num2 = 0, result = 0;  
    char operator;  
  
    public CalculatorGUI() {  
        f1=new JFrame("Calculator");  
        f1.setVisible(true);  
        f1.setSize(600,800);  
        f1.setLayout(null);  
        f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
        display=new JTextField();
```

```
display.setBounds(50,20,250,50);
```

```
b1=new JButton("1");  
b2=new JButton("2");  
b3=new JButton("3");  
b4=new JButton("4");  
b5=new JButton("5");  
b6=new JButton("6");  
b7=new JButton("7");  
b8=new JButton("8");  
b9=new JButton("9");  
plus=new JButton("+");  
minus=new JButton("-");  
mul=new JButton("*");  
div=new JButton("/");  
clear=new JButton("C");  
zero=new JButton("0");  
res=new JButton("=");
```

```
b1.setBounds(180,175,50,40);  
b2.setBounds(115,175,50,40);  
b3.setBounds(50,175,50,40);  
b4.setBounds(180,130,50,40);  
b5.setBounds(115,130,50,40);  
b6.setBounds(50,130,50,40);  
b7.setBounds(180,85,50,40);  
b8.setBounds(115,85,50,40);  
b9.setBounds(50,85,50,40);  
plus.setBounds(245,85,50,40);  
minus.setBounds(245,130,50,40);  
mul.setBounds(245,175,50,40);  
div.setBounds(245,225,50,40);  
res.setBounds(180,225,50,40);  
zero.setBounds(115,225,50,40);  
clear.setBounds(50,225,50,40);
```

```
b1.addActionListener(this);
```



```
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
b6.addActionListener(this);
b7.addActionListener(this);
b8.addActionListener(this);
b9.addActionListener(this);
plus.addActionListener(this);
minus.addActionListener(this);
mul.addActionListener(this);
div.addActionListener(this);
clear.addActionListener(this);
res.addActionListener(this);
zero.addActionListener(this);
```

```
f1.add(b1);
f1.add(b1);
f1.add(b2);
f1.add(b3);
f1.add(b4);
f1.add(b5);
f1.add(b6);
f1.add(b7);
f1.add(b8);
f1.add(b9);
f1.add(plus);
f1.add(minus);
f1.add(mul);
f1.add(div);
f1.add(clear);
f1.add(res);
f1.add(zero);
```

```
f1.add(display);
```

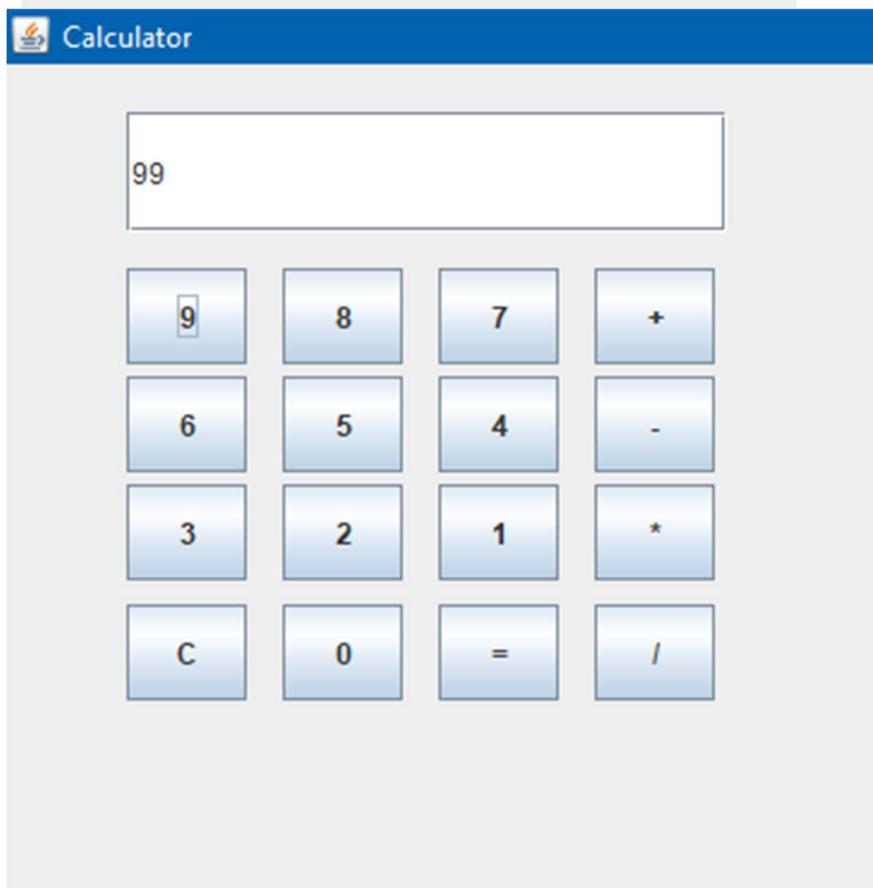
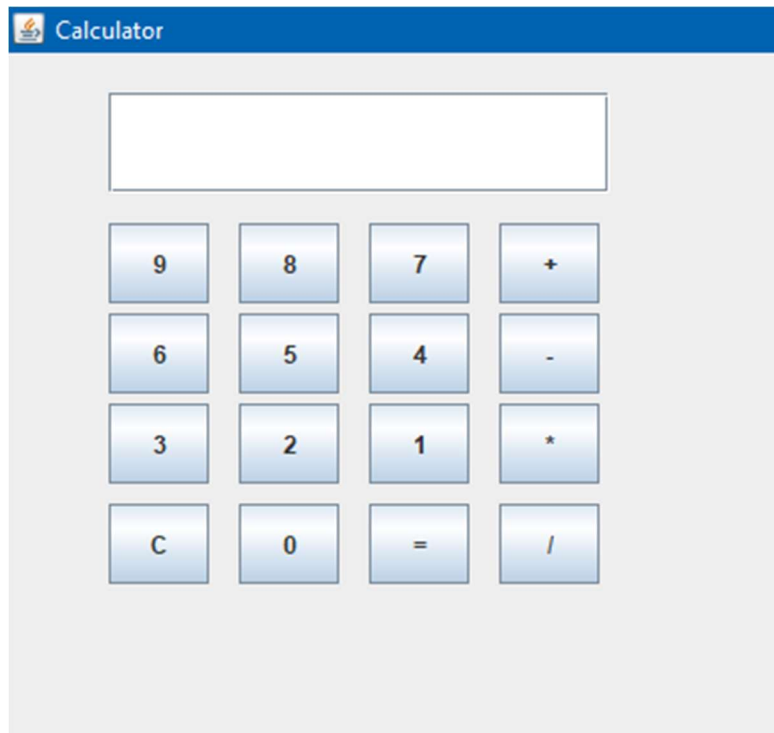
```
}
```

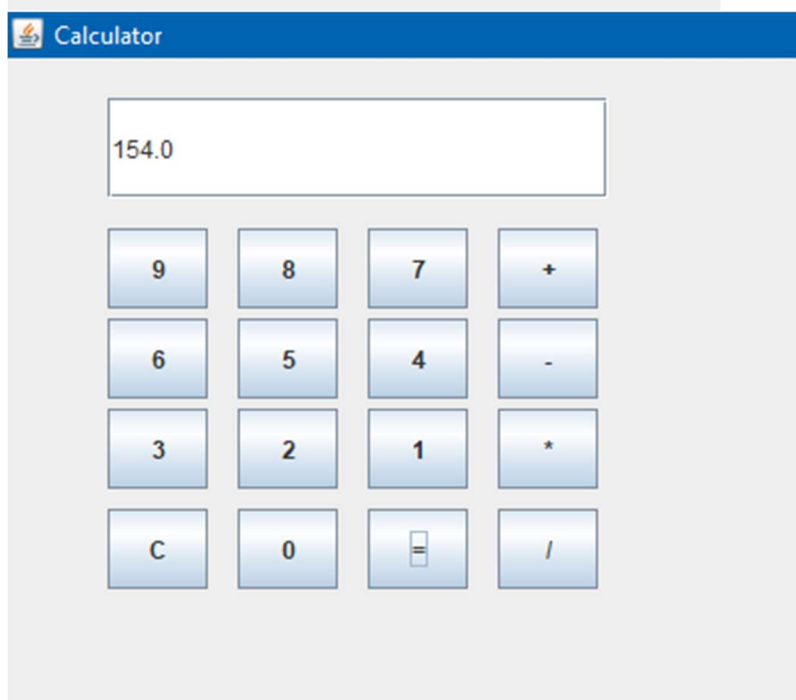
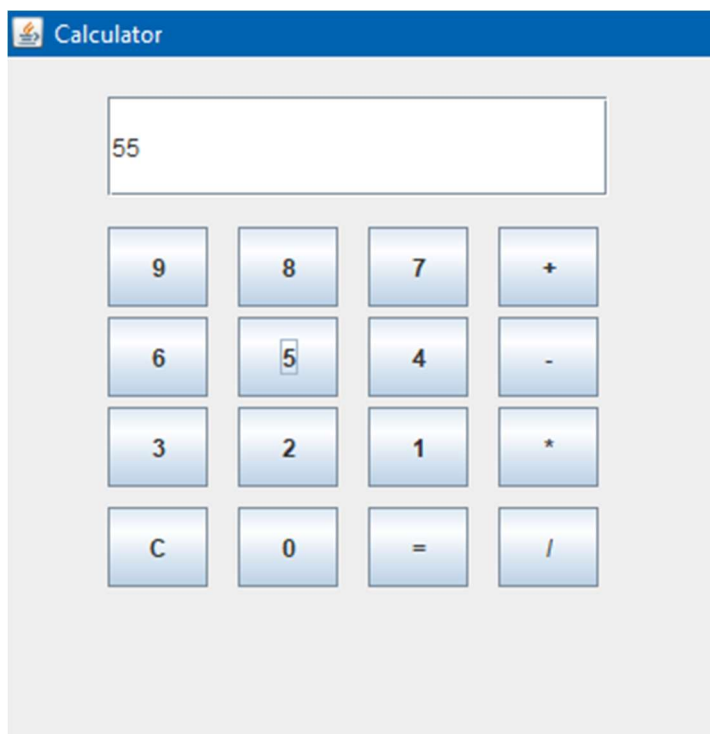
```

@Override
public void actionPerformed(ActionEvent e) {
    String command = e.getActionCommand();
    if ((command.charAt(0) >= '0' && command.charAt(0) <= '9') ||
command.charAt(0) == '.') {
        display.setText(display.getText() + command);
    } else if (command.charAt(0) == 'C') {
        display.setText("");
        num1 = num2 = result = 0;
        operator = '\0';
    } else if (command.charAt(0) == '=') {
        num2 = Double.parseDouble(display.getText());
        switch (operator) {
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                result = num1 / num2;
                break;
        }
        display.setText(String.valueOf(result));
    } else {
        if (!display.getText().isEmpty()) {
            num1 = Double.parseDouble(display.getText());
            operator = command.charAt(0);
            display.setText("");
        }
    }
}
}

```

OUTPUT:





Discussion

The practice for making a GUI of a calculator which performed arithmetic operations on values which are passed through input value buttons button clicked was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a GUI of a calculator which had all input values as button inputs and performed desired arithmetic operations.

Title

Create a gallery having some thumbnails with action listener and show full screen image when clicked.

Objective

To learn how to create a gallery of desired images thumbnails and on click the desired images to be displayed on full screen.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Main {
    public static void main(String[] args) {

        JFrame f1 = new JFrame("Thumbnails");
        f1.setLayout(null);
        f1.setSize(1000, 1000);
        f1.setVisible(true);
        f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        ImageIcon img1 = new ImageIcon("C:\\Users\\pakhr\\Desktop\\6th
sem\\AdvancedJava\\Lab reports\\lab6\\Images\\img1.jpg");
        ImageIcon img2 = new ImageIcon("C:\\Users\\pakhr\\Desktop\\6th
sem\\AdvancedJava\\Lab reports\\lab6\\Images\\img2.jpg");
        ImageIcon img3 = new ImageIcon("C:\\Users\\pakhr\\Desktop\\6th
sem\\AdvancedJava\\Lab reports\\lab6\\Images\\img3.jpg");
        ImageIcon img4 = new ImageIcon("C:\\Users\\pakhr\\Desktop\\6th
sem\\AdvancedJava\\Lab reports\\lab6\\Images\\img4.jpg");
        JButton b1 = new JButton(img1);
        JButton b2 = new JButton(img2);
        JButton b3 = new JButton(img3);
        JButton b4 = new JButton(img4);

        b1.setBounds(0, 0, 400, 400);
```

```

b2.setBounds(400, 0, 400, 400);
b3.setBounds(0, 400, 400, 400);
b4.setBounds(400, 400, 400, 400);
b1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        displayZoomedImage(img1);
    }
});

b2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        displayZoomedImage(img2);
    }
});

b3.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        displayZoomedImage(img3);
    }
});

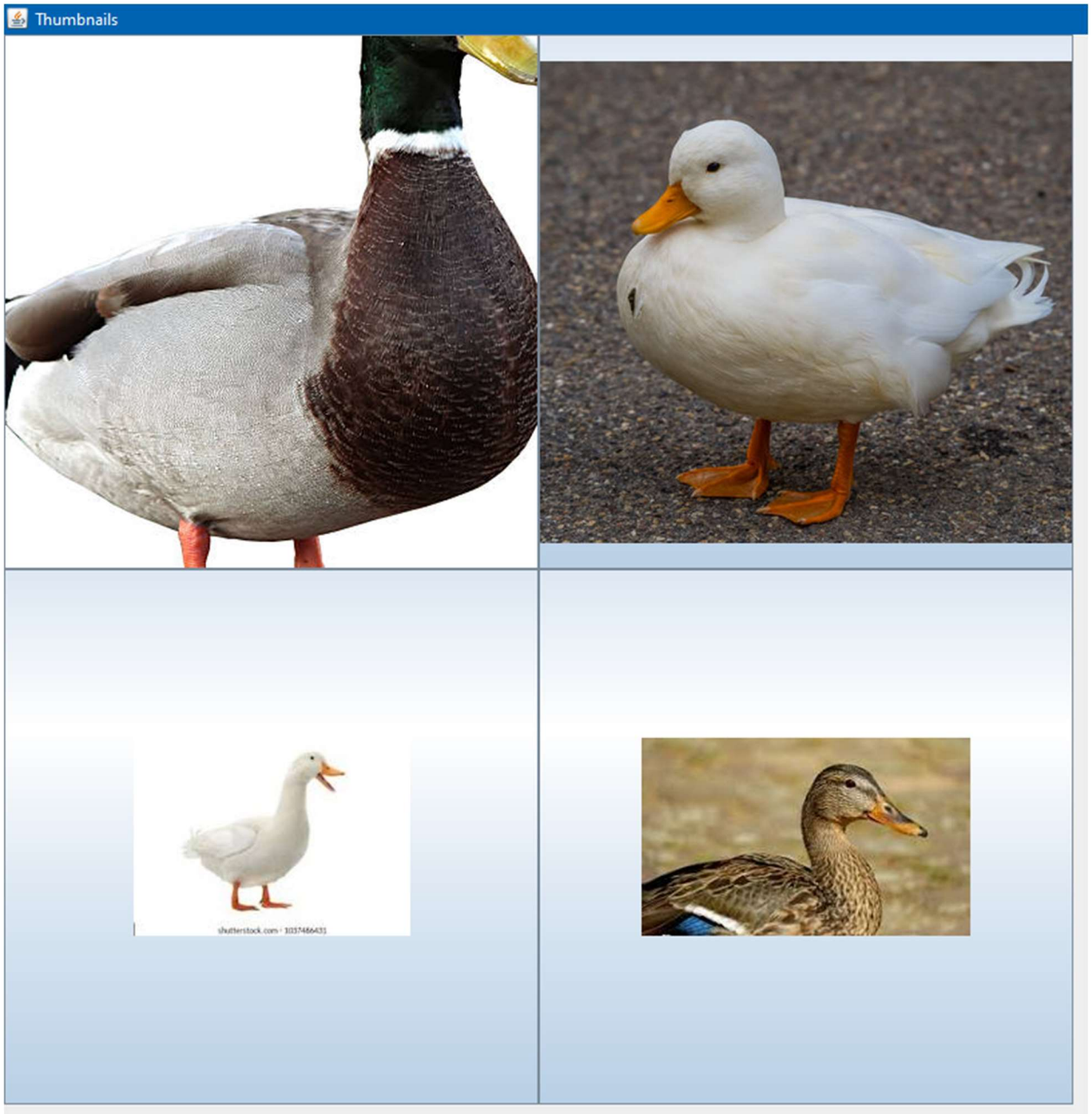
b4.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        displayZoomedImage(img4);
    }
});

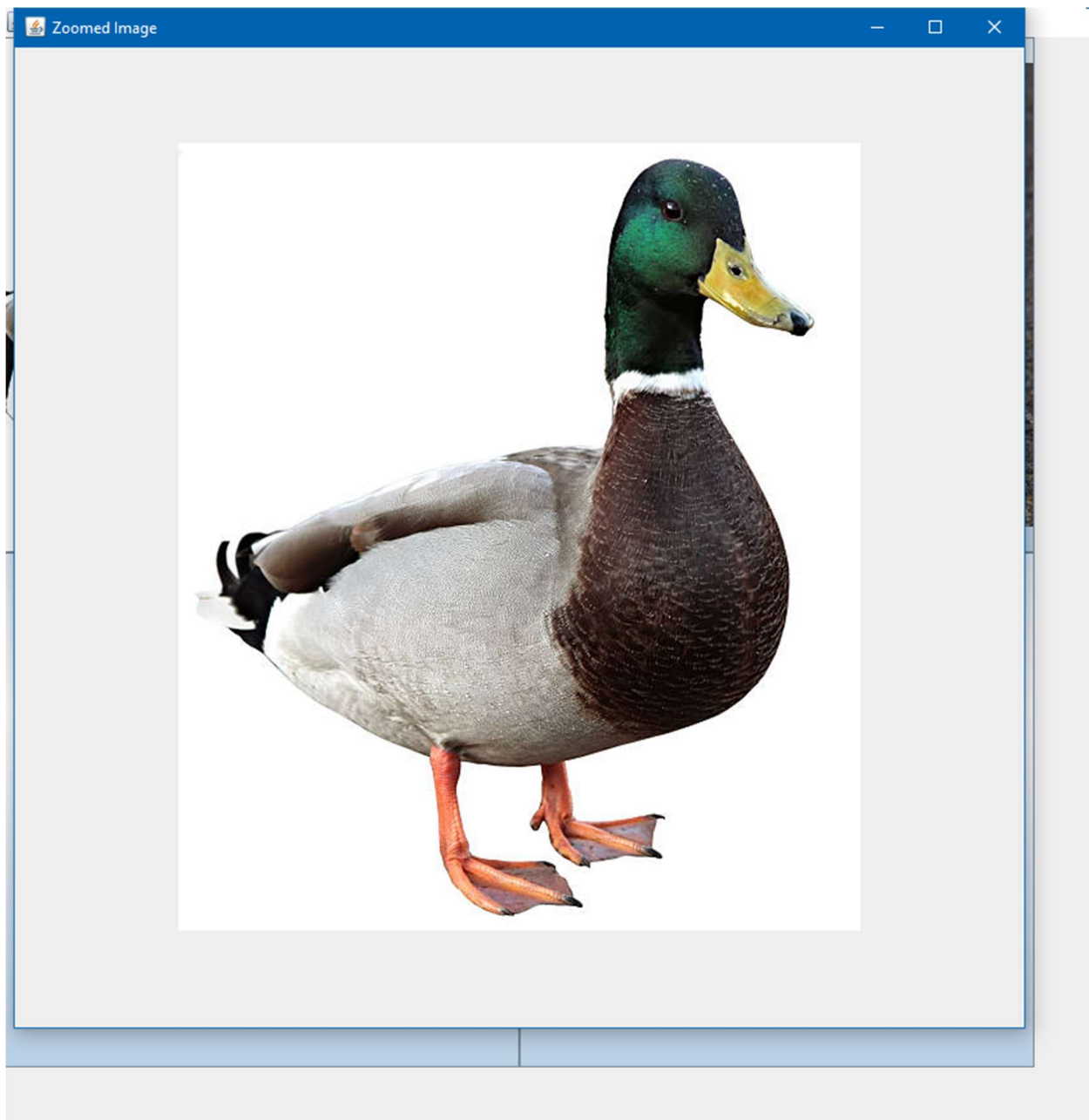
f1.add(b1);
f1.add(b2);
f1.add(b3);
f1.add(b4);
}
public static void displayZoomedImage(ImageIcon icon) {
    JFrame f2 = new JFrame("Zoomed Image");
    f2.setSize(800, 800);
    JLabel l2 = new JLabel(icon);
    f2.add(l2);

```

```
f2.setVisible(true);  
}  
}
```

OUTPUT:





Discussion

The practice for making a GUI of a gallery of desired images thumbnails and on click the desired images to be displayed on full screen with the help of action listener was understood and performed successfully.

Conclusion:

By completing this program, I learnt to create a gallery of images and display desire image on full screen via the help of action listener.

Title

Create a login form with database verification.

Objective

To learn to verify a user login from the existing records in the database.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class DBLogin extends JFrame {

    private JTextField usernameField;
    private JPasswordField passwordField;
    private JButton loginButton;
    private JPanel mainPanel;

    public DBLogin() {
        setTitle("Login Form");
        setSize(350, 250);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        mainPanel = new JPanel();
        mainPanel.setLayout(null);

        JLabel userLabel = new JLabel("Username:");
        userLabel.setBounds(30, 30, 80, 25);
        mainPanel.add(userLabel);

        usernameField = new JTextField(20);
        usernameField.setBounds(120, 30, 185, 25);
        mainPanel.add(usernameField);

        JLabel passwordLabel = new JLabel("Password:");
```

```

passwordLabel.setBounds(30, 70, 80, 25);
mainPanel.add(passwordLabel);

passwordField = new JPasswordField(20);
passwordField.setBounds(120, 70, 185, 25);
mainPanel.add(passwordField);

loginButton = new JButton("Login");
loginButton.setBounds(30, 110, 275, 30);
mainPanel.add(loginButton);

add(mainPanel);

loginButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/userlogin",
"root", "");
            String query = "SELECT * FROM userlist WHERE
username=? AND password=?";
            PreparedStatement preparedStatement =
connection.prepareStatement(query);
            preparedStatement.setString(1, usernameField.getText());
            preparedStatement.setString(2, new
String(passwordField.getPassword()));
            ResultSet resultSet = preparedStatement.executeQuery();

            if (resultSet.next()) {
                JOptionPane.showMessageDialog(null, "Login successful!");
            } else {
                JOptionPane.showMessageDialog(null, "Invalid username or
password.");
            }

            connection.close();
        } catch (Exception ex) {
            System.out.println("Error");
        }
    }
}

```

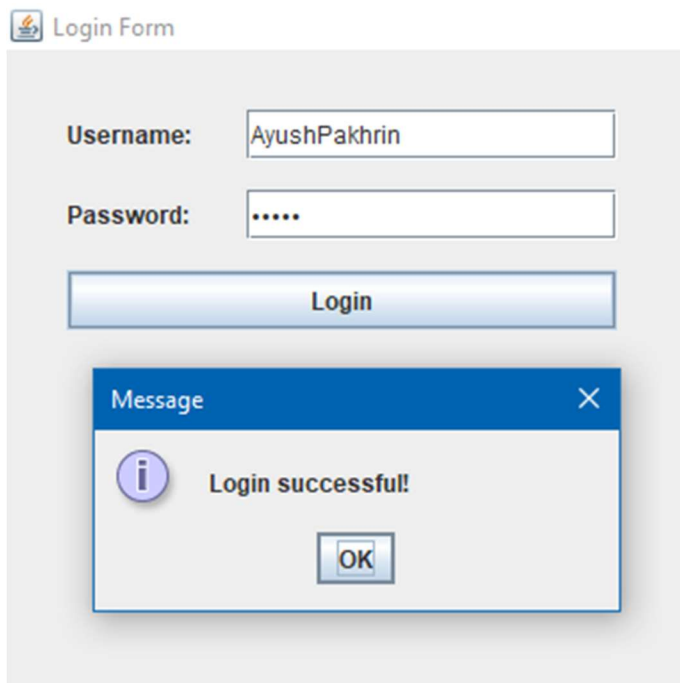
```

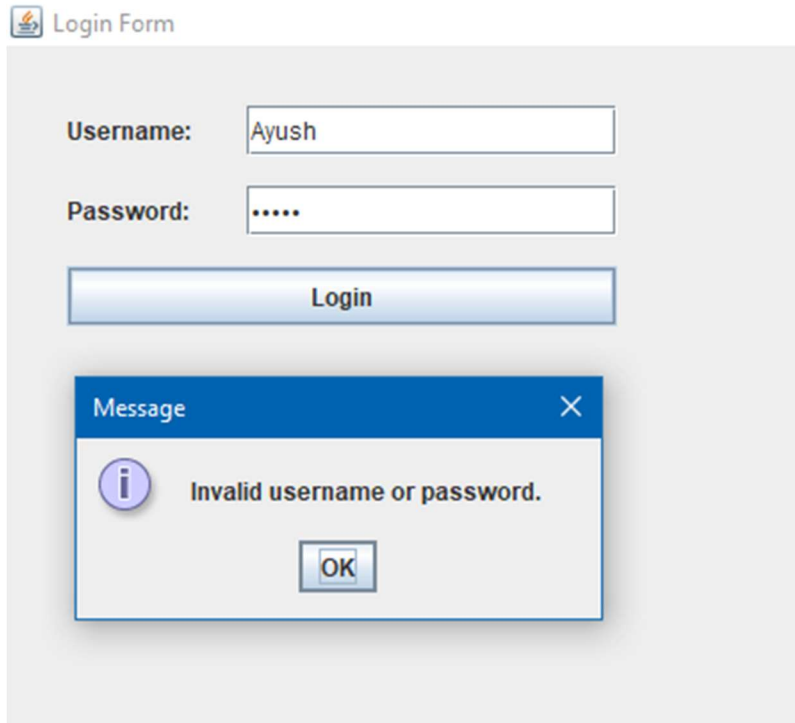
    }
    });
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new DBLogin().setVisible(true);
        }
    });
}
}

```

OUTPUT:





Discussion

The practice for validating an user login from the existing records in the database which was accessed through the sqlconnector library imported in the program was understood and performed successfully.

Conclusion:

By completing this program, I learnt to validate an user login with the existing records in the database.

Title

Create a registration form containing text fields, passwords, radio button, checkbox, label, button with confirm password field which should be validated.

Objective

To learn to confirm the two user input passwords and register the user only if the two passwords match each other in the form along with other details.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class RegistrationForm extends JFrame {

    private JTextField usernameField;
    private JPasswordField passwordField;
    private JPasswordField confirmPasswordField;
    private JRadioButton maleRadioButton;
    private JRadioButton femaleRadioButton;
    private JCheckBox agreeCheckBox;
    private JButton registerButton;
    private JPanel mainPanel;

    public RegistrationForm() {
        setTitle("RegistrationForm");
        setSize(400, 350);
```

```
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
setLocationRelativeTo(null);
```

```
mainPanel = new JPanel();  
mainPanel.setLayout(null);
```

```
JLabel userLabel = new JLabel("Username:");  
userLabel.setBounds(30, 30, 80, 25);  
mainPanel.add(userLabel);
```

```
usernameField = new JTextField(20);  
usernameField.setBounds(150, 30, 200, 25);  
mainPanel.add(usernameField);
```

```
JLabel passwordLabel = new JLabel("Password:");  
passwordLabel.setBounds(30, 70, 80, 25);  
mainPanel.add(passwordLabel);
```

```
passwordField = new JPasswordField(20);  
passwordField.setBounds(150, 70, 200, 25);  
mainPanel.add(passwordField);
```

```
JLabel confirmPasswordLabel = new JLabel("Confirm Password:");  
confirmPasswordLabel.setBounds(30, 110, 120, 25);  
mainPanel.add(confirmPasswordLabel);
```

```
confirmPasswordField = new JPasswordField(20);  
confirmPasswordField.setBounds(150, 110, 200, 25);  
mainPanel.add(confirmPasswordField);
```

```
JLabel genderLabel = new JLabel("Gender:");  
genderLabel.setBounds(30, 150, 80, 25);  
mainPanel.add(genderLabel);
```

```
maleRadioButton = new JRadioButton("Male");  
maleRadioButton.setBounds(150, 150, 80, 25);  
mainPanel.add(maleRadioButton);
```

```
femaleRadioButton = new JRadioButton("Female");  
femaleRadioButton.setBounds(230, 150, 80, 25);  
mainPanel.add(femaleRadioButton);
```

```
ButtonGroup genderGroup = new ButtonGroup();  
genderGroup.add(maleRadioButton);  
genderGroup.add(femaleRadioButton);
```

```
agreeCheckBox = new JCheckBox("I agree to the terms and  
conditions.");  
agreeCheckBox.setBounds(30, 190, 320, 25);  
mainPanel.add(agreeCheckBox);
```



```

registerButton = new JButton("Register");
registerButton.setBounds(150, 230, 100, 30);
mainPanel.add(registerButton);

add(mainPanel);

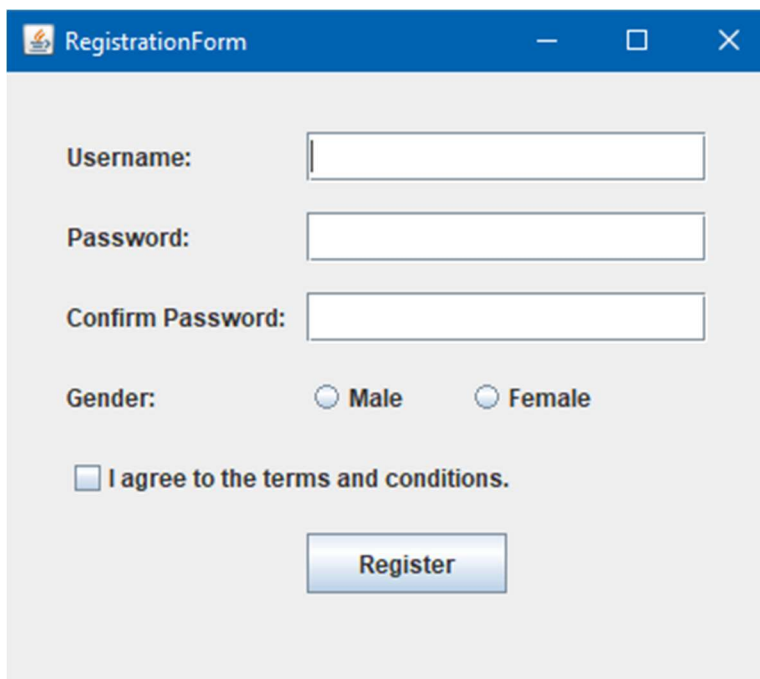
registerButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String password = new String(passwordField.getPassword());
        String confirmPassword = new
String(confirmPasswordField.getPassword());

        if (!password.equals(confirmPassword)) {
            JOptionPane.showMessageDialog(null, "Passwords do not
match!");
        } else if (!agreeCheckBox.isSelected()) {
            JOptionPane.showMessageDialog(null, "You must agree to the
terms and conditions.");
        } else {
            JOptionPane.showMessageDialog(null, "Registration
successful!");
        }
    }
});
}

```

```
public static void main(String[] args) {  
    SwingUtilities.invokeLater(new Runnable() {  
        @Override  
        public void run() {  
            new RegistrationForm().setVisible(true);  
        }  
    });  
}
```

OUTPUT:



The screenshot shows a Java Swing window titled "RegistrationForm" with a standard Windows-style title bar (blue background, white text, and minimize, maximize, and close buttons). The window contains a registration form with the following elements:

- Username:** A text input field.
- Password:** A text input field.
- Confirm Password:** A text input field.
- Gender:** Two radio buttons labeled "Male" and "Female".
- Terms and Conditions:** A checkbox followed by the text "I agree to the terms and conditions."
- Register:** A button with a blue gradient and a shadow effect.

RegistrationForm

Username:

Password:

Confirm Password:

Gender: ☒ Male ☐ Female

☒ I agree to the terms and conditions.

Register

Message

Registration successful!

OK

RegistrationForm

Username:

Password:

Confirm Password:

Gender: ☒ Male ☐ Female

☒ I agree to the terms and conditions.

Register

Message

Passwords do not match!

OK

The screenshot shows a Windows application titled "RegistrationForm". The form contains the following fields and controls:

- Username:** A text input field.
- Password:** A text input field.
- Confirm Password:** A text input field.
- Gender:** Two radio buttons labeled "Male" (selected) and "Female".
- Terms and Conditions:** A checkbox labeled "I agree to the terms and conditions." which is currently unchecked.
- Register:** A button to submit the form.

An error message dialog box is displayed in the foreground with the title "Message". It contains an information icon and the text "You must agree to the terms and conditions." with an "OK" button.

Discussion

The practice for registering an user only and only if the two passwords input in the password field and confirm password field match each other and if they don't match return the error message was understood and performed successfully.

Conclusion:

By completing this program, I learnt to validate the user password inputs and confirm passwords input comparing them to each other and only registering in the case of successful password inputs.

Title

Create a GUI that displays record from database.

Objective

To learn to access the database and view the database existing records.

Program Code:

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class DisplayRecordProgram extends JFrame {

    public JTable table;
    public DefaultTableModel tableModel;

    public DisplayRecordProgram() {
        setTitle("Display Records");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        String[] columnNames = {"Username", "Password"};
        tableModel = new DefaultTableModel(columnNames, 0);
        table = new JTable(tableModel);

        JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setBounds(20, 20, 550, 300);

        JPanel mainPanel = new JPanel();
        mainPanel.setLayout(null);
        mainPanel.add(scrollPane);

        add(mainPanel);

        loadDataFromDatabase();
    }
}
```

```

    }

    private void loadDataFromDatabase() {
        String url =
"jdbc:mysql://localhost:3306/userlogin?zeroDateTimeBehavior=convertToN
ull";
        String user = "root";
        String password = "";

        try {
            Connection connection = DriverManager.getConnection(url, user,
password);
            Statement statement = connection.createStatement();
            String query = "SELECT * FROM userlist";
            ResultSet resultSet = statement.executeQuery(query);

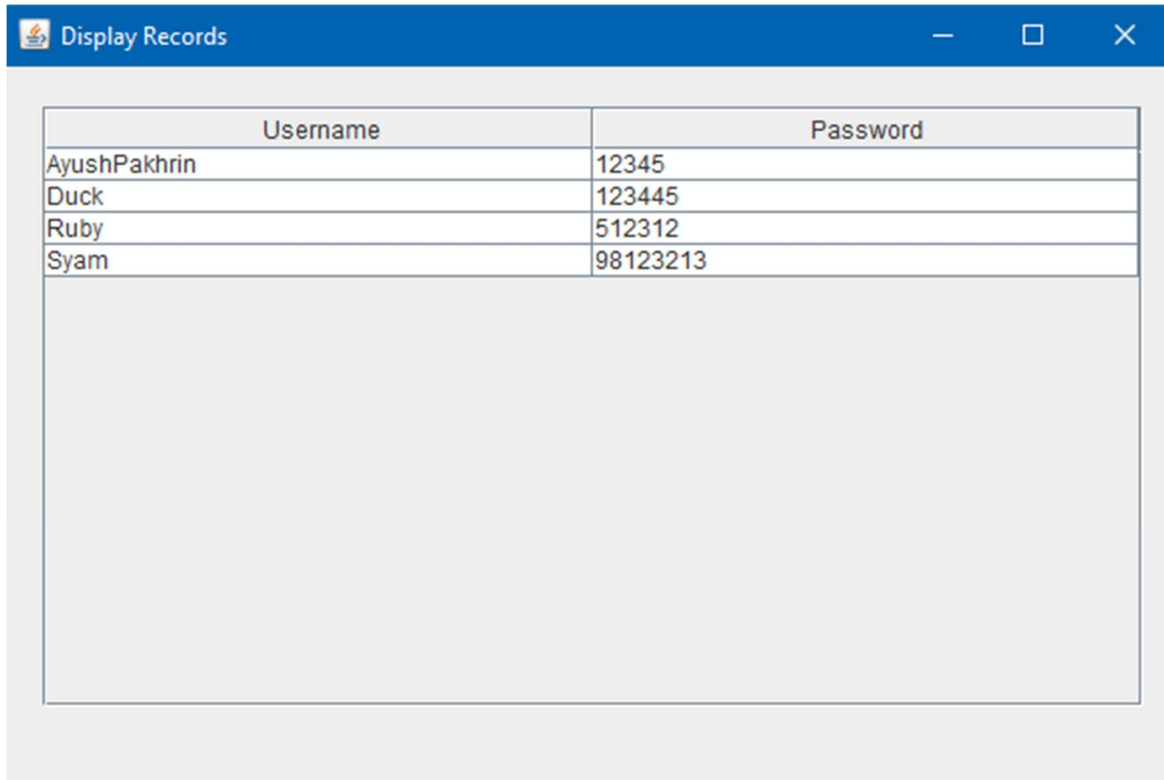
            while (resultSet.next()) {
                String username = resultSet.getString("username");
                String passwords = resultSet.getString("password");
                tableModel.addRow(new Object[]{username, passwords});
            }

            connection.close();
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new DisplayRecordProgram().setVisible(true);
            }
        });
    }
}

```

OUTPUT:



The screenshot shows a Java Swing window titled "Display Records" with a blue title bar. Inside the window, there is a table with two columns: "Username" and "Password". The table contains four rows of data. Below the table, there is a large, empty rectangular area, likely intended for additional information or a message.

Username	Password
AyushPakhrin	12345
Duck	123445
Ruby	512312
Syam	98123213

Discussion

The practice for accessing the database with the help of sqlconnector imported in the library and making the view for the existing database records in a tabular form was understood and performed successfully.

Conclusion:

By completing this program, I learnt to retrieve the data existing in the database with the help of required libraries and present it in a tabular form.

Title

Create a GUI that will receive password of an existing user and change it in database when button is clicked.

Objective

To learn to retrieve the password of an existing user in the database and change the existing password to a new password in the database through the input textfield.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
public class ChangePasswordProgram extends JFrame {

    private JTextField usernameField;
    private JPasswordField newPasswordField;
    private JButton changePasswordButton;
    private JPanel mainPanel;

    public ChangePasswordProgram() {
        setTitle("Reset Password");
        setSize(400, 250);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        mainPanel = new JPanel();
        mainPanel.setLayout(null);

        JLabel usernameLabel = new JLabel("Username:");
        usernameLabel.setBounds(30, 30, 100, 25);
        mainPanel.add(usernameLabel);

        usernameField = new JTextField(20);
        usernameField.setBounds(150, 30, 200, 25);
```



```

mainPanel.add(usernameField);

JLabel newPasswordLabel = new JLabel("New Password:");
newPasswordLabel.setBounds(30, 70, 100, 25);
mainPanel.add(newPasswordLabel);

newPasswordField = new JPasswordField(20);
newPasswordField.setBounds(150, 70, 200, 25);
mainPanel.add(newPasswordField);

changePasswordButton = new JButton("Change Password");
changePasswordButton.setBounds(100, 120, 200, 30);
mainPanel.add(changePasswordButton);

add(mainPanel);

changePasswordButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String username = usernameField.getText();
        String newPassword = new
String(newPasswordField.getPassword());

        String url =
"jdbc:mysql://localhost:3306/userlogin?zeroDateTimeBehavior=convertToN
ull";

        String user = "root";
        String password = "";

        try {
            Connection connection = DriverManager.getConnection(url,
user, password);

            String fetchQuery = "SELECT password FROM userlist
WHERE username=?";
            PreparedStatement fetchStmt =
connection.prepareStatement(fetchQuery);
            fetchStmt.setString(1, username);
            ResultSet resultSet = fetchStmt.executeQuery();

```

```

        if (resultSet.next()) {
            String oldPassword = resultSet.getString("password");

            String updateQuery = "UPDATE userlist SET password=?
WHERE username=?";
            PreparedStatement updateStmt =
connection.prepareStatement(updateQuery);
            updateStmt.setString(1, newPassword);
            updateStmt.setString(2, username);
            int rowsUpdated = updateStmt.executeUpdate();

            if (rowsUpdated > 0) {
                JOptionPane.showMessageDialog(null, "Password for
user: " + username + " has been changed from \"" + oldPassword + "\" to \""
+ newPassword + "\".");
            } else {
                JOptionPane.showMessageDialog(null, "Failed to update
password. User not found.");
            }
        } else {
            JOptionPane.showMessageDialog(null, "User not found.");
        }

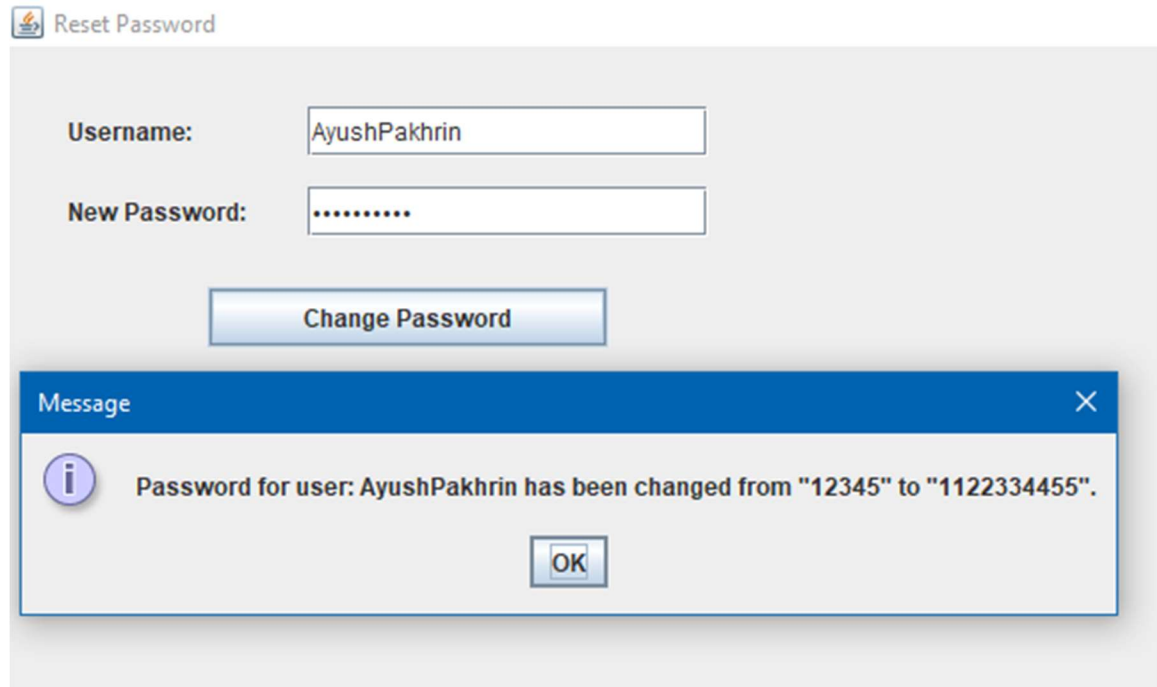
        connection.close();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

});
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new ChangePasswordProgram().setVisible(true);
        }
    });
}
}

```

OUTPUT:



Discussion

The practice for accessing the database with the help of SQLConnector imported in the library retrieving the password of existing user and changing it to a new password was understood and performed successfully.

Conclusion:

By completing this program, I learnt to retrieve the password of an existing user in the database and make desired changes to it.

Title

Create a GUI to ask username and delete that user from the database.

Objective

To delete an existing user in the database with the help of username.

Program Code:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
public class DeleteUser extends JFrame {
    private JTextField usernameField;
    private JButton deleteButton;
    private JPanel mainPanel;
    public DeleteUser() {
        setTitle("Delete a user");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        mainPanel = new JPanel();
        mainPanel.setLayout(null);
        JLabel usernameLabel = new JLabel("username:");
        usernameLabel.setBounds(30, 30, 100, 25);
        mainPanel.add(usernameLabel);
        usernameField = new JTextField(20);
        usernameField.setBounds(150, 30, 200, 25);
        mainPanel.add(usernameField);
```

```

deleteButton = new JButton("Delete User");
deleteButton.setBounds(100, 70, 200, 30);
mainPanel.add(deleteButton);
add(mainPanel);

deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String username = usernameField.getText();

        String url =
"jdbc:mysql://localhost:3306/userlogin?zeroDateTimeBehavior=convertToNull";

        String user = "root";
        String password = "";

        try {
            Connection connection = DriverManager.getConnection(url,
user, password);

            String deleteQuery = "DELETE FROM userlist WHERE
username=?";

            PreparedStatement deleteStmt =
connection.prepareStatement(deleteQuery);

            deleteStmt.setString(1, username);

            int rowsDeleted = deleteStmt.executeUpdate();

            if (rowsDeleted > 0) {

```

```

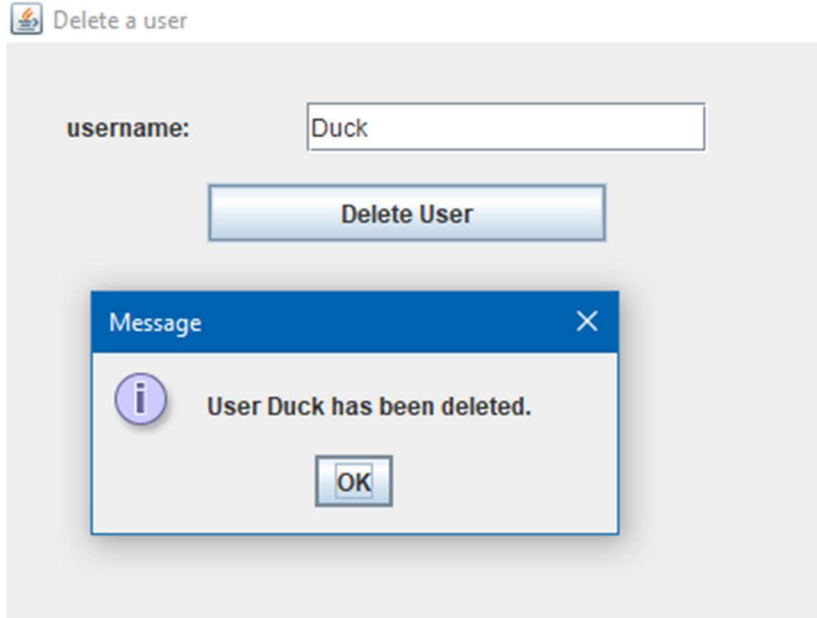
        JOptionPane.showMessageDialog(null, "User " + username +
" has been deleted.");
    } else {
        JOptionPane.showMessageDialog(null, "User not found.");
    }

    connection.close();
} catch (Exception ex) {
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null, "Error occurred while
deleting user.");
}
}
});
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new DeleteUser().setVisible(true);
        }
    });
}
}

```

OUTPUT:



Discussion

The practice for accessing the database with the help of `SQLConnector` imported in the library and delete the existing user from the database with the help of username to that corresponding user was understood and performed successfully.

Conclusion:

By completing this program, I learnt to delete an existing user in the database with the help of username corresponding to that user.