# Lab 11: Graphical User Interface (GUI) in Java

# Lab Engineer: Ms. Bareera Anam

# Class: BSDS 2

# Subject: OOP

**Class: BSDS -02A**

**AILYA ZAINAB**

**( 5 2 3 5 0 6 )**

### **Lab Task 1: Calculator GUI**

**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class SwingGUI extends JFrame implements ActionListener {

private JTextField display;

private String currentInput = "";

private double firstNumber = 0;

private String operation = "";

private boolean newInput = true;

public SwingGUI() {

setTitle("Colorful Calculator");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(350, 500);

setLocationRelativeTo(null);

setResizable(false);

// Main panel with gradient background

JPanel mainPanel = new JPanel() {

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

Graphics2D g2d = (Graphics2D) g;

Color color1 = new Color(159, 18, 18); // Cornflower blue

Color color2 = new Color(147, 112, 219); // Medium purple

GradientPaint gp = new GradientPaint(0, 0, color1, getWidth(), getHeight(), color2);

g2d.setPaint(gp);

g2d.fillRect(0, 0, getWidth(), getHeight());

}

};

mainPanel.setLayout(new BorderLayout(10, 10));

mainPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

// Display

display = new JTextField();

display.setEditable(false);

display.setHorizontalAlignment(JTextField.RIGHT);

display.setFont(new Font("Arial", Font.BOLD, 36));

display.setBackground(new Color(240, 248, 255)); // Alice blue

display.setBorder(BorderFactory.createLineBorder(new Color(70, 130, 180), 3));

mainPanel.add(display, BorderLayout.NORTH);

// Button panel

JPanel buttonPanel = new JPanel(new GridLayout(5, 4, 10, 10));

buttonPanel.setOpaque(false);

// Button colors

Color numberButtonColor = new Color(255, 255, 255, 200); // Semi-transparent white

Color operationButtonColor = new Color(0, 255, 255, 200); // Semi-transparent gold

Color clearButtonColor = new Color(220, 20, 60, 200); // Semi-transparent crimson

// Button texts

String[] buttonLabels = {

"7", "8", "9", "/",

"4", "5", "6", "\*",

"1", "2", "3", "-",

"0", "C", "=", "+"

};

// Create buttons

for (String label : buttonLabels) {

JButton button = new JButton(label);

button.setFont(new Font("Arial", Font.BOLD, 24));

button.setFocusPainted(false);

button.addActionListener(this);

// Set different colors for different button types

if (label.matches("[0-9]")) {

button.setBackground(numberButtonColor);

} else if (label.matches("[+\\-\*/]")) {

button.setBackground(operationButtonColor);

} else if (label.equals("C")) {

button.setBackground(clearButtonColor);

} else { // Equals button

button.setBackground(new Color(50, 205, 50, 200)); // Semi-transparent lime green

}

button.setBorder(BorderFactory.createRaisedBevelBorder());

buttonPanel.add(button);

}

mainPanel.add(buttonPanel, BorderLayout.CENTER);

add(mainPanel);

}

@Override

public void actionPerformed(ActionEvent e) {

String command = e.getActionCommand();

if (command.matches("[0-9]")) {

if (newInput) {

currentInput = command;

newInput = false;

} else if (currentInput.length() < 4) { // Still limit to 4 digits

currentInput += command;

}

display.setText(currentInput);

} else if (command.matches("[+\\-\*/]")) {

if (!currentInput.isEmpty()) {

firstNumber = Double.parseDouble(currentInput);

operation = command;

newInput = true;

// Don't clear the display yet

}

} else if (command.equals("=")) {

if (!operation.isEmpty() && !newInput) {

double secondNumber = Double.parseDouble(currentInput);

double result = 0;

switch (operation) {

case "+":

result = firstNumber + secondNumber;

break;

case "-":

result = firstNumber - secondNumber;

break;

case "\*":

result = firstNumber \* secondNumber;

break;

case "/":

if (secondNumber != 0) {

result = firstNumber / secondNumber;

} else {

display.setText("Error");

resetCalculator();

return;

}

break;

}

// Format result nicely (remove decimal if integer)

if (result == (long) result) {

display.setText(String.format("%d", (long) result));

} else {

display.setText(String.format("%s", result));

}

currentInput = display.getText();

newInput = true;

operation = "";

}

} else if (command.equals("C")) {

resetCalculator();

}

}

private void resetCalculator() {

currentInput = "";

firstNumber = 0;

operation = "";

newInput = true;

display.setText("");

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

try {

// Set look and feel for modern appearance

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

} catch (Exception e) {

e.printStackTrace();

}

SwingGUI calculator = new SwingGUI();

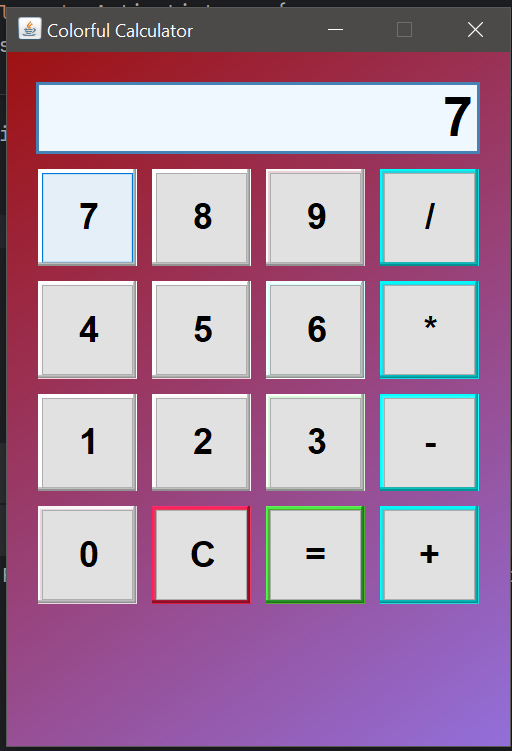
calculator.setVisible(true);

});

}

}

**OUTPUT:**



### **Lab Task 2: Login Form GUI**

**CODE:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.geom.RoundRectangle2D;

class Login extends JFrame {

private JTextField usernameField;

private JPasswordField passwordField;

public Login() {

setTitle("SYSTEM ACCESS");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(500, 400);

setUndecorated(true);

setShape(new RoundRectangle2D.Double(0, 0, 500, 400, 30, 30));

setLocationRelativeTo(null);

// Main panel with cyberpunk styling

JPanel mainPanel = new JPanel() {

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

Graphics2D g2d = (Graphics2D) g;

// Dark background with grid lines

g2d.setColor(new Color(5, 5, 15));

g2d.fillRect(0, 0, getWidth(), getHeight());

// Grid lines

g2d.setColor(new Color(0, 255, 100, 30));

for (int i = 0; i < getWidth(); i += 20) {

g2d.drawLine(i, 0, i, getHeight());

}

for (int i = 0; i < getHeight(); i += 20) {

g2d.drawLine(0, i, getWidth(), i);

}

// Neon border

g2d.setStroke(new BasicStroke(3));

g2d.setColor(new Color(0, 255, 100));

g2d.drawRoundRect(5, 5, getWidth()-10, getHeight()-10, 25, 25);

}

};

mainPanel.setLayout(new GridBagLayout());

mainPanel.setBorder(BorderFactory.createEmptyBorder(40, 40, 40, 40));

GridBagConstraints gbc = new GridBagConstraints();

gbc.gridwidth = GridBagConstraints.REMAINDER;

gbc.fill = GridBagConstraints.HORIZONTAL;

gbc.insets = new Insets(10, 0, 10, 0);

// Title label

JLabel titleLabel = new JLabel("SECURE ACCESS TERMINAL");

titleLabel.setForeground(new Color(0, 255, 100));

titleLabel.setFont(new Font("Consolas", Font.BOLD, 24));

titleLabel.setHorizontalAlignment(SwingConstants.CENTER);

mainPanel.add(titleLabel, gbc);

// Username field

JLabel userLabel = new JLabel("USER ID:");

userLabel.setForeground(new Color(0, 255, 100));

userLabel.setFont(new Font("Consolas", Font.BOLD, 14));

mainPanel.add(userLabel, gbc);

usernameField = createCyberpunkTextField();

mainPanel.add(usernameField, gbc);

// Password field

JLabel passLabel = new JLabel("AUTH CODE:");

passLabel.setForeground(new Color(0, 255, 100));

passLabel.setFont(new Font("Consolas", Font.BOLD, 14));

mainPanel.add(passLabel, gbc);

passwordField = createCyberpunkPasswordField();

mainPanel.add(passwordField, gbc);

// Login button

JButton loginButton = new JButton("AUTHENTICATE");

styleCyberpunkButton(loginButton);

loginButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

if (username.equals("admin") && password.equals("1234")) {

showMessage("ACCESS GRANTED\nWELCOME TO THE SYSTEM", new Color(0, 255, 100));

} else {

showMessage("INVALID CREDENTIALS\nACCESS DENIED", new Color(255, 50, 50));

}

}

});

mainPanel.add(loginButton, gbc);

// Exit button

JButton exitButton = new JButton("TERMINATE CONNECTION");

styleCyberpunkButton(exitButton);

exitButton.addActionListener(e -> System.exit(0));

mainPanel.add(exitButton, gbc);

add(mainPanel);

}

private JTextField createCyberpunkTextField() {

JTextField field = new JTextField(20);

field.setForeground(new Color(0, 255, 100));

field.setBackground(new Color(20, 20, 30));

field.setCaretColor(new Color(0, 255, 100));

field.setFont(new Font("Consolas", Font.PLAIN, 16));

field.setBorder(BorderFactory.createCompoundBorder(

BorderFactory.createLineBorder(new Color(0, 255, 100, 100)),

BorderFactory.createEmptyBorder(5, 5, 5, 5) // padding inside the border

));

return field;

}

private JPasswordField createCyberpunkPasswordField() {

JPasswordField field = new JPasswordField(20);

field.setForeground(new Color(0, 255, 100));

field.setBackground(new Color(20, 20, 30));

field.setCaretColor(new Color(0, 255, 100));

field.setFont(new Font("Consolas", Font.PLAIN, 16));

field.setBorder(BorderFactory.createCompoundBorder(

BorderFactory.createLineBorder(new Color(0, 255, 100, 100)),

BorderFactory.createEmptyBorder(8, 10, 8, 10)

));

return field;

}

private void styleCyberpunkButton(JButton button) {

button.setForeground(Color.BLACK);

button.setBackground(new Color(0, 255, 100));

button.setFont(new Font("Consolas", Font.BOLD, 14));

button.setFocusPainted(false);

button.setBorder(BorderFactory.createEmptyBorder(10, 25, 10, 25));

button.setCursor(new Cursor(Cursor.HAND\_CURSOR));

button.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseEntered(java.awt.event.MouseEvent evt) {

button.setBackground(new Color(50, 255, 150));

}

public void mouseExited(java.awt.event.MouseEvent evt) {

button.setBackground(new Color(0, 255, 100));

}

});

}

private void showMessage(String message, Color color) {

JLabel messageLabel = new JLabel("<html><center>" + message.replace("\n", "<br>") + "</center></html>");

messageLabel.setForeground(color);

messageLabel.setFont(new Font("Consolas", Font.BOLD, 18));

messageLabel.setHorizontalAlignment(SwingConstants.CENTER);

JOptionPane.showMessageDialog(this, messageLabel, "SYSTEM MESSAGE",

JOptionPane.PLAIN\_MESSAGE, new ImageIcon());

}

public static void main(String[] args) {

try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

} catch (Exception e) {

e.printStackTrace();

}

SwingUtilities.invokeLater(() -> {

Login login = new Login();

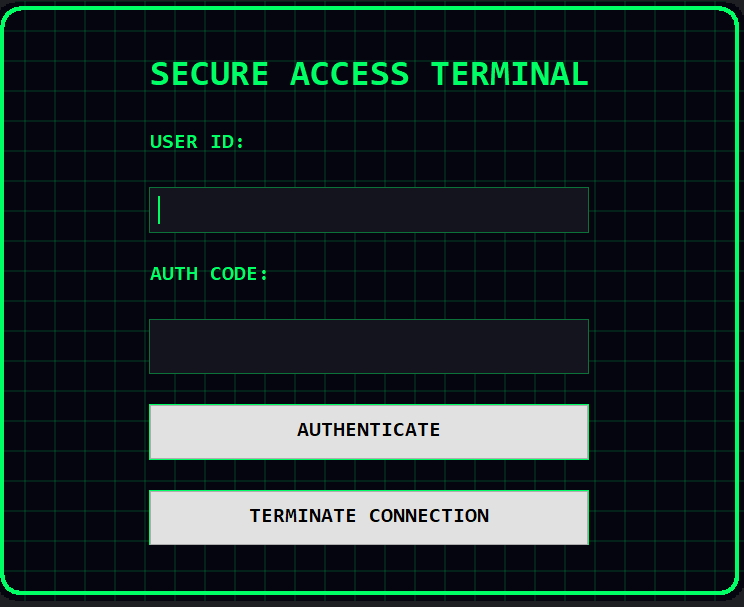
login.setVisible(true);

});

}

}

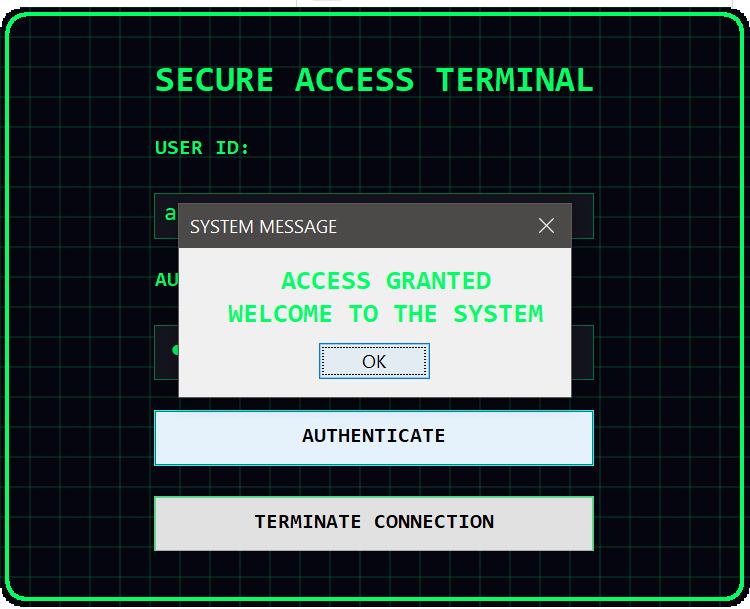
**OUTPUT:**



**AFTER ENTERING CREDENTIALS**

**admin**

**123**



**WRONG CREDENTIALS**

