Experiment No.5

Date:

Aim: Design real world problem system based on AWT components.

CO Mapping – CO 1

Objective:

1. To define the problem that will be solved using an AWT (Abstract Window Toolkit) GUI application.
2. To understand and utilize AWT components such as Frame, Panel, Label, TextField, Button, Checkbox, Choice, List, Canvas, and Menu.
3. To use appropriate layout managers (FlowLayout, BorderL
4. ayout, GridLayout, or GridBagLayout) to organize components systematically.
5. To implement event handling using the ActionListener, MouseListener, KeyListener, or other event interfaces.
6. To design a user-friendly and visually appealing interface and use appropriate labels, tooltips, and alignment to guide users effectively.

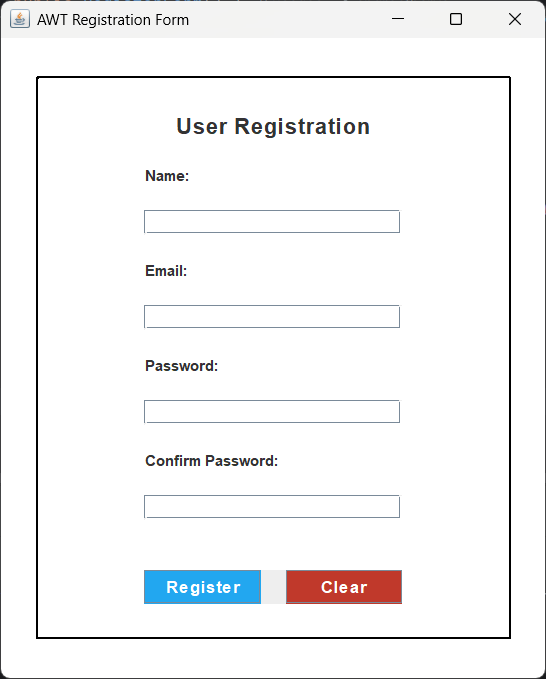
Lab Exercise:

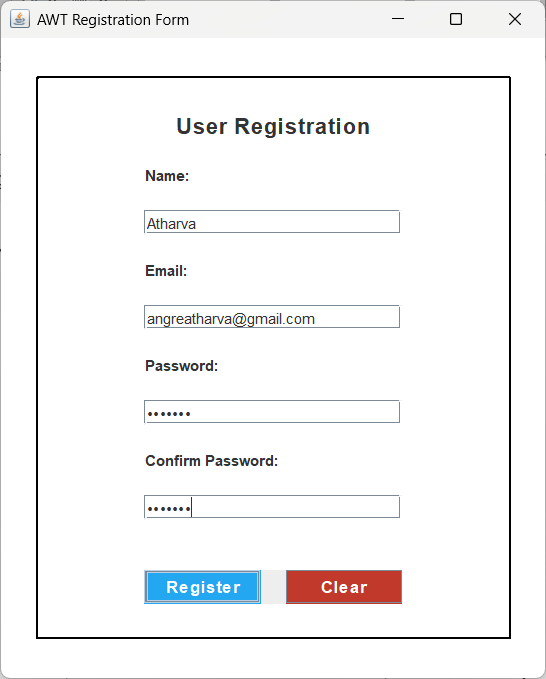
* Implement a user registration form using AWT and Swing components.
* Use JFrame, JLabel, JTextField, JPasswordField, JButton, and layout managers.
* Validate user input (email format, password match, required fields).
* Handle events using ActionListener for button clicks.

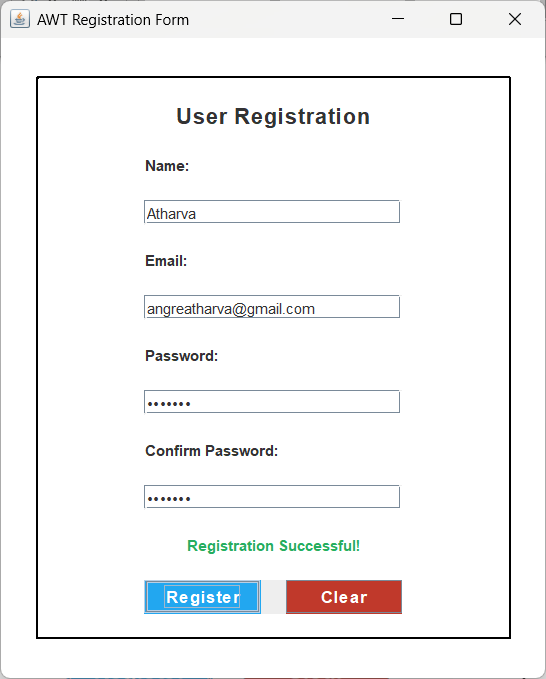
Code:

Main.java  
public class Main {  
 public static void main(String[] args) {  
 new RegisterForm();  
 }  
}

RegisterForm.java  
import java.awt.\*;  
import java.awt.event.\*;  
import java.util.regex.Pattern;  
import javax.swing.\*;  
import javax.swing.border.\*;  
  
public class RegisterForm extends JFrame implements ActionListener {  
 // UI Components  
 JLabel labelTitle, labelName, labelEmail, labelPassword, labelConfirmPassword, labelMessage;  
 JTextField textName, textEmail;  
 JPasswordField textPassword, textConfirmPassword;  
 JButton btnRegister, btnClear;  
  
 // Constructor  
 public RegisterForm() {  
 // Set Frame properties  
 setTitle("AWT Registration Form");  
 setSize(450, 550);  
 setLayout(new GridBagLayout()); // Centering the container  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 getContentPane().setBackground(Color.*WHITE*); // Background color  
  
 // Container Panel with Padding & Border  
 JPanel container = new JPanel();  
 container.setPreferredSize(new Dimension(380, 450)); // Fixed container size  
 container.setBackground(Color.*WHITE*);  
 container.setLayout(new GridBagLayout()); // GridBagLayout for spacing  
  
 // Add padding inside the container  
 container.setBorder(BorderFactory.*createCompoundBorder*(  
 new LineBorder(Color.*BLACK*, 2, true  
 ), // Outer curved border  
 BorderFactory.*createEmptyBorder*(20, 20, 20, 20) // Padding inside  
 ));  
  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.insets = new Insets(10, 10, 10, 10); // Space between elements  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
 gbc.gridx = 0;  
 gbc.gridy = 0;  
  
 // Title Label  
 labelTitle = new JLabel("User Registration", SwingConstants.*CENTER*);  
 labelTitle.setFont(new Font("Arial", Font.*BOLD*, 18));  
 container.add(labelTitle, gbc);  
  
 gbc.gridy++;  
 labelName = new JLabel("Name:");  
 textName = new JTextField(15);  
 container.add(labelName, gbc);  
 gbc.gridy++;  
 container.add(textName, gbc);  
  
 gbc.gridy++;  
 labelEmail = new JLabel("Email:");  
 textEmail = new JTextField(15);  
 container.add(labelEmail, gbc);  
 gbc.gridy++;  
 container.add(textEmail, gbc);  
  
 gbc.gridy++;  
 labelPassword = new JLabel("Password:");  
 textPassword = new JPasswordField(15);  
 container.add(labelPassword, gbc);  
 gbc.gridy++;  
 container.add(textPassword, gbc);  
  
 gbc.gridy++;  
 labelConfirmPassword = new JLabel("Confirm Password:");  
 textConfirmPassword = new JPasswordField(15);  
 container.add(labelConfirmPassword, gbc);  
 gbc.gridy++;  
 container.add(textConfirmPassword, gbc);  
  
 gbc.gridy++;  
 labelMessage = new JLabel("", SwingConstants.*CENTER*);  
 labelMessage.setForeground(Color.*RED*);  
 container.add(labelMessage, gbc);  
  
 gbc.gridy++;  
 JPanel buttonPanel = new JPanel(new GridLayout(1, 2, 20, 10));  
  
 btnRegister = new JButton("Register");  
 btnRegister.setBackground(new Color(34, 167, 240));  
 btnRegister.setForeground(Color.*WHITE*);  
 btnRegister.setFont(new Font("Arial", Font.*BOLD*, 14));  
 btnRegister.addActionListener(this);  
 buttonPanel.add(btnRegister);  
  
 btnClear = new JButton("Clear");  
 btnClear.setBackground(new Color(192, 57, 43));  
 btnClear.setForeground(Color.*WHITE*);  
 btnClear.setFont(new Font("Arial", Font.*BOLD*, 14));  
 btnClear.addActionListener(this);  
 buttonPanel.add(btnClear);  
  
 container.add(buttonPanel, gbc);  
  
 add(container); // Add container to frame  
  
 setLocationRelativeTo(null); // Center the frame  
 setVisible(true);  
 }  
  
 // Email validation using regex  
 private boolean isValidEmail(String email) {  
 String emailRegex = "^[A-Za-z0-9+\_.-]+@[A-Za-z0-9.-]+$";  
 return Pattern.*matches*(emailRegex, email);  
 }  
  
 // Action Event Handling  
 public void actionPerformed(ActionEvent e) {  
 if (e.getSource() == btnRegister) {  
 String name = textName.getText().trim();  
 String email = textEmail.getText().trim();  
 String password = new String(textPassword.getPassword());  
 String confirmPassword = new String(textConfirmPassword.getPassword());  
  
 // Validation  
 if (name.isEmpty() || email.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {  
 labelMessage.setText("All fields are required!");  
 return;  
 }  
 if (!isValidEmail(email)) {  
 labelMessage.setText("Invalid email format!");  
 return;  
 }  
 if (!password.equals(confirmPassword)) {  
 labelMessage.setText("Passwords do not match!");  
 return;  
 }  
 if (password.length() < 6) {  
 labelMessage.setText("Password must be at least 6 characters!");  
 return;  
 }  
  
 // Success Message  
 labelMessage.setForeground(new Color(39, 174, 96));  
 labelMessage.setText("Registration Successful!");  
 }  
  
 if (e.getSource() == btnClear) {  
 textName.setText("");  
 textEmail.setText("");  
 textPassword.setText("");  
 textConfirmPassword.setText("");  
 labelMessage.setText("");  
 }  
 }  
}

Output:





Observation:

In this experiment, I designed a user registration form using AWT components. I implemented UI elements and added input validation for email and password. Event handling was used to manage button actions. This helped in understanding form validation, UI design, and event-driven programming. Overall, it provided hands-on experience in building interactive GUI applications using AWT.