Roll no: 312056 **Seat no: 348**

Java Mini Project

"Password Protected Notepad"

Source code:-

// AddNewUser.java

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

public class AddNewUser implements ActionListener {

    JFrame frame = new JFrame();
    JButton saveButton = new JButton("Save");
    JButton goToLoginButton = new JButton("Go to Login");
    JTextField userIDField = new JTextField();
    JPasswordField userPasswordField = new JPasswordField();
    JLabel userIDLabel = new JLabel("Craete UserID:");
    JLabel userPasswordLabel = new JLabel("Set Password:");
    JLabel messgaLabel = new JLabel();

    AddNewUser() {

        // Creating the frames for userid, pass and message
        userIDLabel.setBounds(50, 100, 75, 25);
        userPasswordLabel.setBounds(50, 150, 75, 25);
        userIDField.setBounds(125, 100, 200, 25);
        userPasswordField.setBounds(125, 150, 200, 25);

        // Creating save and go to Login button and adding action listner to the
        saveButton.setBounds(125, 200, 100, 25);
        saveButton.addActionListener(this);
        saveButton.setFocusable(false);
        goToLoginButton.setBounds(225, 200, 100, 25);
        goToLoginButton.addActionListener(this);
        goToLoginButton.setFocusable(false);

        // Adding the elements to the frame
        frame.add(saveButton);
        frame.add(goToLoginButton);
        frame.add(userIDLabel);
        frame.add(userPasswordLabel);
        frame.add(userIDField);
        frame.add(userPasswordField);

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(420, 420);
        frame.setLocationRelativeTo(null);
        frame.setLayout(null);
        frame.setVisible(true);
    }
```

```
    // Addding functionality to the buttons
    @Override
    public void actionPerformed(ActionEvent e) {

        if (e.getSource() == saveButton) {
            frame.dispose();
            IDandPasswords IdPass = new IDandPasswords();
            IdPass.loginInfo.put(userIDField.getText(),
String.valueOf(userPasswordField.getPassword()));
            LoginPage loginpage = new LoginPage(IdPass.getLoginInfo());
        }
        if (e.getSource() == goToLoginButton) {
            frame.dispose();
            IDandPasswords IdPass = new IDandPasswords();
            LoginPage loginpage = new LoginPage(IdPass.getLoginInfo());
        }
    }
}


// IDandPasswords.java

import java.util.*;

public class IDandPasswords {
    HashMap<String, String> loginInfo = new HashMap<String, String>();

    IDandPasswords() {
        loginInfo.put("Admin", "Admin@123");
    }

    public void AddNewUser(){
        Scanner sc = new Scanner(System.in);
        IDandPasswords newUserDetails = new IDandPasswords();
        newUserDetails.loginInfo.put(sc.nextLine(), sc.nextLine());
    }

    protected HashMap getLoginInfo() {
        return loginInfo;
    }
}

// LoginPage.java
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class LoginPage implements ActionListener {

    JFrame frame = new JFrame();
    JButton loginButton = new JButton("Login");
```

```
    JButton addUserButton = new JButton("New User");
    JButton resetButton = new JButton("Reset");
    JTextField userIDField = new JTextField();
    JPasswordField userPasswordField = new JPasswordField();
    JLabel userIDLabel = new JLabel("User ID :");
    JLabel userPasswordLabel = new JLabel("Password :");
    JLabel messgaLabel = new JLabel();

    HashMap<String, String> loginInfo = new HashMap<String, String>();

    LoginPage(HashMap<String, String> loginInfoOriginal) {
        loginInfo = loginInfoOriginal;

        // Creating the frames for userid, pass and message
        userIDLabel.setBounds(50, 100, 75, 25);
        userPasswordLabel.setBounds(50, 150, 75, 25);
        messgaLabel.setBounds(125, 250, 250, 35);
        messgaLabel.setFont(new Font(null, Font.ITALIC, 25));
        userIDField.setBounds(125, 100, 200, 25);
        userPasswordField.setBounds(125, 150, 200, 25);

        // Creating login, reset and new user button and adding action listner to the
        // buttons
        loginButton.setBounds(55, 200, 100, 25);
        loginButton.addActionListener(this);
        loginButton.setFocusable(false);
        resetButton.setBounds(155, 200, 100, 25);
        resetButton.addActionListener(this);
        resetButton.setFocusable(false);
        addUserButton.setBounds(255, 200, 100, 25);
        addUserButton.addActionListener(this);
        addUserButton.setFocusable(false);

        // Adding the elements to the frame
        frame.add(userIDLabel);
        frame.add(userPasswordLabel);
        frame.add(messgaLabel);
        frame.add(userIDField);
        frame.add(userPasswordField);
        frame.add(loginButton);
        frame.add(resetButton);
        frame.add(addUserButton);

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(420, 420);
        frame.setLocationRelativeTo(null);
        frame.setLayout(null);
        frame.setVisible(true);
    }

    // Addding functionality to the buttons
    @Override
    public void actionPerformed(ActionEvent e) {
```

```
        // Reset button
        if (e.getSource() == resetButton) {
            userIDField.setText("");
            userPasswordField.setText("");
        }

        // Login button
        if (e.getSource() == loginButton) {
            String userID = userIDField.getText();
            String password = String.valueOf(userPasswordField.getPassword());

            if (loginInfo.containsKey(userID)) {
                if (loginInfo.get(userID).equals(password)) {
                    messgaLabel.setForeground(Color.green);
                    messgaLabel.setText("Login successful !!!");
                    frame.dispose();
                    new TextEditor();

                } else {
                    messgaLabel.setForeground(Color.red);
                    messgaLabel.setText("Wrong password !!!");
                }
            } else {
                messgaLabel.setForeground(Color.red);
                messgaLabel.setText("Username not found !!!");
            }
        }

        // Add a new user button
        if(e.getSource()==addUserButton){
            frame.dispose();
            new AddNewUser();
        }
    }
}

// TextEditor.java
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.filechooser.*;

public class TextEditor extends JFrame implements ActionListener {
    JTextArea textArea;
    JScrollPane scrollPane;
    JSpinner fontSizeSpinner;
    JLabel fontLabel;
    JButton fontColorButton;
    JComboBox fontBox;
    JMenuBar menuBar;
    JMenu fileMenu;
```

```
    JMenuItem openItem;
    JMenuItem saveItem;
    JMenuItem exitItem;

    TextEditor() {

        // Creating a notepad window
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setTitle("Notrpadf");
        this.setSize(500, 500);
        this.setLayout(new FlowLayout());
        this.setLocationRelativeTo(null);

        // Adding a text area to the notepad window
        textArea = new JTextArea();
        textArea.setLineWrap(true);
        textArea.setWrapStyleWord(true);
        textArea.setFont(new Font("Arial", Font.PLAIN, 20));

        // Adding a scroll bar to the notepad window
        scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(460, 450));
        scrollPane.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAY
S);

        fontSizeSpinner = new JSpinner();
        fontSizeSpinner.setPreferredSize(new Dimension(50, 25));
        fontSizeSpinner.setValue(20);
        fontSizeSpinner.addChangeListener(new ChangeListener() {
            @Override
            public void stateChanged(ChangeEvent e) {
                textArea.setFont(
                        (new Font(textArea.getFont().getFamily(), Font.PLAIN, (int)
fontSizeSpinner.getValue())));
            }
        });

        fontLabel = new JLabel("Font Size");

        fontColorButton = new JButton("Font Color");
        fontColorButton.addActionListener(this);

        String[] fonts =
GraphicsEnvironment.getLocalGraphicsEnvironment().getAvailableFontFamilyNames();
        fontBox = new JComboBox(fonts);
        fontBox.addActionListener(this);
        fontBox.setSelectedItem("Arial");

        // ---: MENU BAR :---
        menuBar = new JMenuBar();
        fileMenu = new JMenu("File");
        openItem = new JMenuItem("Open");
        saveItem = new JMenuItem("Save");
        exitItem = new JMenuItem("Exit");
```

```
        // Adding action listener to the menu buttons
        openItem.addActionListener(this);
        saveItem.addActionListener(this);
        exitItem.addActionListener(this);

        menuBar.add(fileMenu);
        fileMenu.add(openItem);
        fileMenu.add(saveItem);
        fileMenu.add(exitItem);
        // ---: MENU BAR :---

        this.setJMenuBar(menuBar);
        this.add(fontLabel);
        this.add(fontSizeSpinner);
        this.add(fontColorButton);
        this.add(fontBox);
        this.add(scrollPane);
        this.setVisible(true);
    }

    // Adding the actions performed by the each of the provided options
    @Override
    public void actionPerformed(ActionEvent e) {

        // Setting a text color
        if (e.getSource() == fontColorButton) {
            JColorChooser colorChooser = new JColorChooser();
            Color color = colorChooser.showDialog(null, "Chose a color", Color.black);
            textArea.setForeground(color);
        }

        // Setting the font size
        if (e.getSource() == fontBox) {
            textArea.setFont(new Font((String) fontBox.getSelectedItem(), Font.PLAIN,
textArea.getFont().getSize()));
        }

        // Adding functionality to menu buttons
        // Open button
        if (e.getSource() == openItem) {
            JFileChooser fileChooser = new JFileChooser();
            fileChooser.setCurrentDirectory(new File("."));

            int response = fileChooser.showOpenDialog(null);
            if (response == JFileChooser.APPROVE_OPTION) {
                File file = new File(fileChooser.getSelectedFile().getAbsolutePath());
                Scanner fileIn = null;
                try {
                    fileIn = new Scanner(file);
                    if (file.isFile()) {
                        while (fileIn.hasNextLine()) {
                            String line = fileIn.nextLine() + "\n";
                            textArea.append(line);
```

```
                    }
                }
            } catch (FileNotFoundException e1) {
                e1.printStackTrace();
            } finally {
                fileIn.close();
            }

        }
    }

    // Save button
    if (e.getSource() == saveItem) {
        JFileChooser fileChooser = new JFileChooser();
        fileChooser.setCurrentDirectory(new File("."));

        int response = fileChooser.showSaveDialog(null);
        if (response == JFileChooser.APPROVE_OPTION) {
            File file;
            PrintWriter fileOut = null;
            file = new file(fileChooser.getSelectedFile().getAbsolutePath());
            try {
                fileOut = new PrintWriter(file);
                fileOut.println(textArea.getText());
            } catch (FileNotFoundException e1) {
                e1.printStackTrace();
            } finally {
                fileOut.close();
            }
        }
    }
    // Exit button
    if (e.getSource() == exitItem) {
        System.exit(0);
    }
}

}

// Main.java
public class Main {
    public static void main(String[] args) {
        IDandPasswords IdPass = new IDandPasswords();
        LoginPage loginpage = new LoginPage(IdPass.getLoginInfo());
    }
}
```