

**Project Report On**  
**Password Protected Notepad Using Java**

Submitted in Partial Fulfilment of Requirements

Of the Course

**Bachelor of Engineering**

**In**

**Computer Engineering**

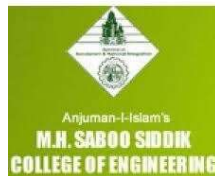
**(Semester III)**

**By**

**Agarwal Gunjan Hemant (312002)**

**Ishan Ahmed (312018)**

**Tarmale Sudarshan Sanjay (312056)**



**Computer Engineering Department**

**M.H. Saboo Siddik College of Engineering**

**2021-22**

## **ACKNOWLEDGEMENT**

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion of my project work. Whatever I have done is only due to such guidance and assistance and I would not forget to thank them.

I respect and thank Principal Dr. Ganesh Kame and Head of the Department (H.O.D) Dr. Zainab Pirani for giving me an opportunity to do the project work in M.H. Saboo Siddik College of Engineering and providing us all support and guidance which made me complete the project on time. I am extremely grateful to him for providing such a nice support and guidance though he had busy schedule managing the company affairs.

I owe my profound gratitude to our project guide **Er. Fatima Anees Ansari**, who took keen interest on our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

I would not forget to remember my group members **Ishan Ahmed and Tarmale Sudarshan Sanjay** for their unlisted encouragement and more over for their timely support and guidance till the completion of our project work.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of Department of computer science which helped us in successfully completing our project work. Also, I would like to extend our sincere regards to all the non-teaching staff of department of computer science for their timely support.

**Agarwal Gunjan Hemant**

## **Project Report Approval for S.E.**

This declaration report entitled **Notepad** by **The Team** is approved for the degree of Third Year Computer Engineering.

### **Examiners**

- 1.
- 2.

### **Supervisors**

- 1.
- 2.

Date: (write date)

Place: Mumbai

## **Abstract**

This application is a password protected notepad created using Java. It is a replication of the text editors and notepads which we use on day-to-day basis and are quite familiar with. The only distinction in this application is that it has been created using for the front-end interface and it is platform independent compile once and run everywhere. Our application contains numerous basic operations like file, edit, save and any new features that one needs can be easily coded making it quite feasible to use. One of the highlights of our system is that it is a password protected application where a user can login using his username and password and his or her account details will be stored and saved for later use. This way a person can access his notes easily without having to worry about any third person having the access to his personal notes. All the features in our system are developed using java swings.

## TABLE OF CONTENTS

Chapter.	Title	Page No.
	<b>Acknowledgement</b>	<b>i</b>
	..... <b>Abstract</b>	<b>4</b>
	.....	
<b>Chapter 01</b>	<b>Introduction .....</b>	<b>7</b>
<b>Chapter 02</b>	<b>Proposed System .....</b>	<b>8</b>
2.1	Problem Definition .....	8
2.2	Aim, Objective & Outcome .....	8
2.3	Overview Of Proposed Approach .....	8
<b>Chapter 03</b>	<b>Implementation .....</b>	<b>9</b>
3.1	System Requirements .....	9
3.2	<i>Software Requirements</i> .....	9
3.3	<i>Hardware Requirements</i> .....	9
3.4	Technologies Used .....	9
3.5	Code .....	10
<b>Chapter 04</b>	<b>Results .....</b>	<b>20</b>
4.1	Output Screens .....	20
<b>Chapter 05</b>	<b>Conclusion .....</b>	<b>23</b>

<b>Chapter 06</b>	<b>Future Scope .....</b>	<b>23</b>
<b>Chapter 07</b>	<b>Limitaions .....</b>	<b>24</b>

	<b>Title</b>	<b>Page No.</b>
	<b>References .....</b>	<b>24</b>

## **Chapter 01: Introduction**

Notepad is a text editor, i.e., an app specialized in editing plain text. It can edit text files (bearing the ".txt" filename extension) and is compatible with formats, such as batch files, INI files, and log files. Notepad is text editor which can perform basic operations like changing the colour, style and size of the font. In this project, we can access the notepad with the help of our username and password. For the purpose of the same we have created a login page. If we have protected our notepad with password there is no chance that a third party person can have access to it.

## **Chapter 02: Proposed System**

**2.1 Problem Definition:** We have created a password protected notepad which can perform basic text editing operations like changing colour, style and size of the font and the main highlight of our application being that it is password protected to avoid piracy.

**2.2 Aim:** To develop a Notepad in Java that is platform independent that can be easily used by all the users and if there are any changes required then a programmer can easily code and see the changes and it is most adaptable.

### **2.2 Objectives:**

The objectives of our project are :

1. To create a login page.
2. To create a notepad.
3. Implementing the created login page to open the notepad.

### **2.3 Over-view Of Proposed System:**

This project "Notepad" is software which can edit plain text. It is made using Java Swings and AWT. In this project all the frames are designed in Swing. Today most programmers use Swing. Swing is a set of classes that provides more powerful and flexible GUI components than does the AWT. Swing provides the look and feel of the modern Java GUI.

Swing did not exist in the early days of Java. Rather, it was a response to deficiencies present in Java's original GUI subsystem: The Abstract Window Toolkit. The AWT defines a basic set of controls, windows, and dialog boxes that support a usable, but limited graphical interface.



## Chapter 3: Implementation

### ➤ *3.1 System Requirement:*

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended.

### ➤ *3.2 Software Requirements:*

- Operating System : Windows 7 or above

### ➤ *3.3 Hardware Requirements:*

- Pentium 4 processor or higher
- 512 MB RAM. (1 GB RAM Recommended)
- 3 MB Hard Disk Space (100 to 200 MB Recommended)

### ➤ *3.4 Technologies Used:*

- JDK 16
- Java. (JSwing)
- Visual Studio Code

### 3.5 Code:

```
//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("Notepad");
        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_A
LWAYS);

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);
}
}
}

```

//IDandPassword.java

```
import java.util.*;
```

```

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

    IDandPasswords() {
        loginInfo.put("Admin", "Admin@123");
        loginInfo.put("Gunjan", "312002");
        loginInfo.put("Ishan", "312018");
        loginInfo.put("Sudarshan", "312056");
    }

    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);
}

// Adding 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();
        }
    }
}

```

//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("Create 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 listener 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);
}

```

```

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

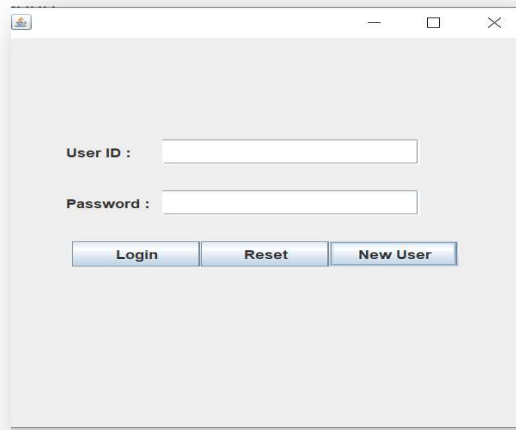
    if (e.getSource() == saveButton) {
        frame.dispose();
        IDandPasswords IdPass = new IDandPasswords();
        IdPass.loginInfo.put(userIDField.getText(),
String.valueOf(userPasswordField.getPassword()));
        LoginPage loginpage = new LoginPage(IdPass.getLoginInfo());
    }
    if (e.getSource()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

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

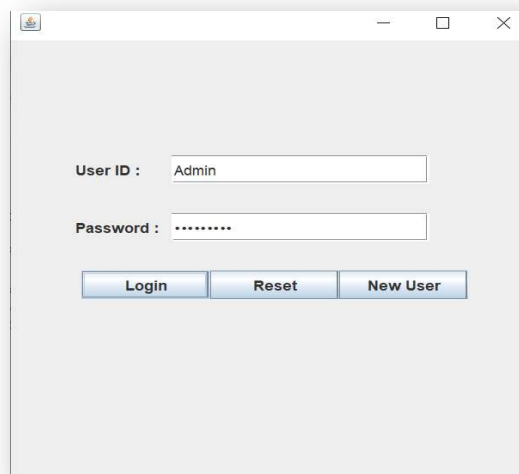
```

## Chapter 4: Results

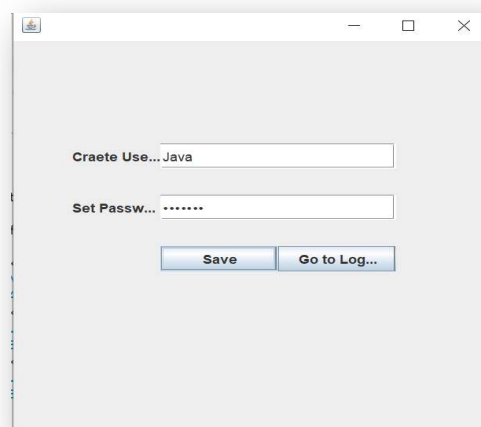
### 4.1 Output:



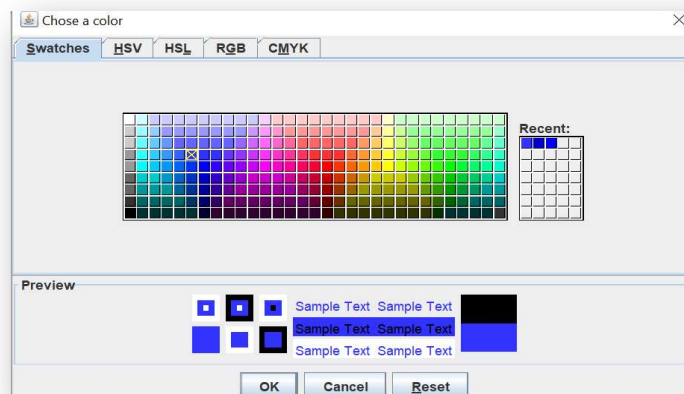
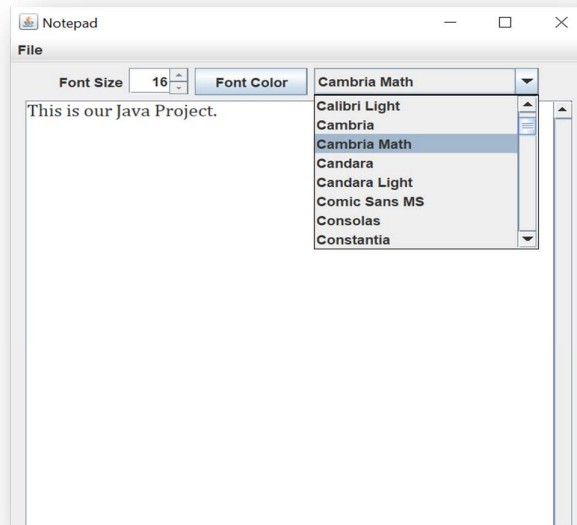
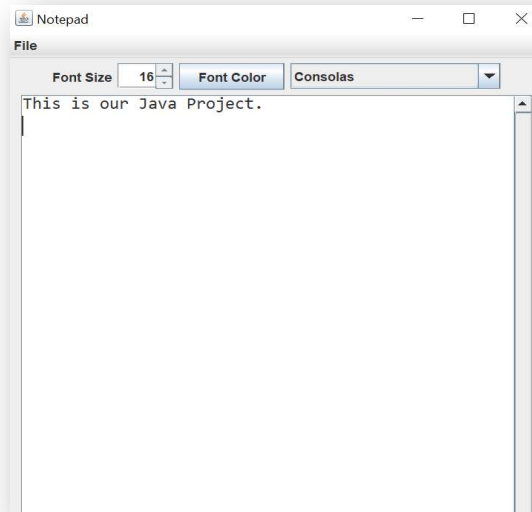
A screenshot of a login window. It features a title bar with a small icon and standard window controls. The main area has a light gray background. There are two input fields: "User ID :" and "Password :". Below these fields are three buttons: "Login", "Reset", and "New User".

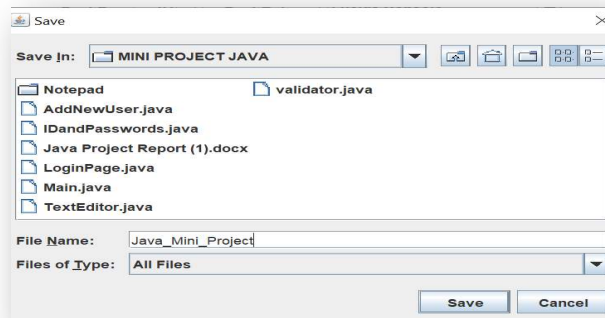
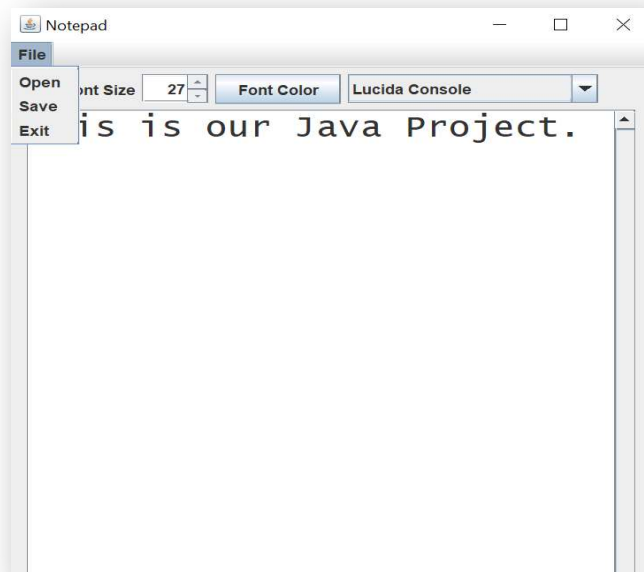
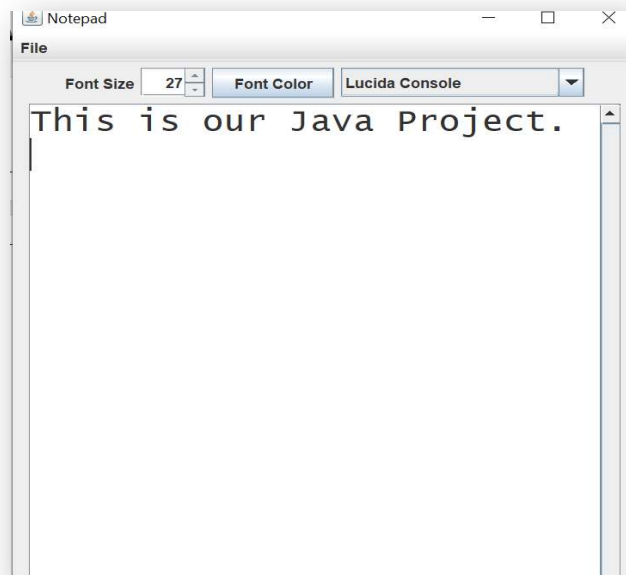


A screenshot of the same login window. The "User ID :" field now contains the text "Admin". The "Password :" field is masked with seven dots ".....". The "Login", "Reset", and "New User" buttons remain at the bottom.



A screenshot of a "Create User" window. It has a title bar with a small icon and standard window controls. The main area has a light gray background. There are two input fields: "Craete Use...Java" and "Set Passw... ..". Below these fields are two buttons: "Save" and "Go to Log...".





## **Chapter 05: Conclusion**

This project that we undertook was truly a very rewarding experience for us in more than one way. It has given a big thrust to our technical knowledge as prospective Software professional. It has also helped us enhance our skills on the personal front. We feel extremely satisfied by the fact that we have managed to develop the project of course with equal contribution and team efforts. We think we have exploited the opportunity that came our way to the fullest extent by increasing our technical know-how and also gaining the valuable work experience.

## **Chapter 06: Future Scope**

In today's world, the protection of sensitive data is one of the most critical concerns for organizations and their customers. This, coupled with growing regulatory pressures, is forcing businesses to protect the integrity, privacy, and security of critical information. As a result, cryptography is emerging as the foundation for enterprise data security and compliance and quickly becoming the foundation of security best practices. Cryptography, once seen as a specialized, esoteric discipline of information security, is finally coming of age.

No one would argue that cryptography and encryption are new technologies. It was true decades ago and it is still true today – encryption is the most reliable way to secure data. National security agencies and major financial institutions have long protected their sensitive data using cryptography and encryption. Today the use of encryption is growing rapidly, being deployed in a much wider set of industry sectors and across an increasing range of applications and platforms. Put simply, cryptography and encryption have become one of the hottest technologies in the IT security industry – the challenge now is to ensure

that IT organizations are equipped to handle this shift and are laying the groundwork today to satisfy their future needs.

The scope of this project is to provide the user with an easy option for encryption and decryption of text. It saves time and maintains the privacy of the message. In the future, the system will be more advanced so that it can be accessed from anywhere by a person having an internet connection at any time.

## **Chapter 07: Limitations**

This project can have further implementations like:

- To save the files in encrypted format
- To have templates
- Automatic dictionary
- Help page that connects to the internet

## **References**

- [https://en.wikipedia.org/wiki/Java\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/Java_(programming_language))
- <https://docs.oracle.com/javase/tutorial/>
- <https://www.developer.com/java/>
- <https://stackoverflow.com/questions/721604/programming-with-java-for-beginners>