

NOTES & PASSWORD MANAGER



A Minor Project Report

in partial fulfillment of the degree

Bachelor of Technology in **Computer Science & Artificial Intelligence**

By

2103A51304

B.SAHITH

2103A51091

J.TEJA

2103A51328

M.DEEKSHITH

2103A51494

S.TEJASWI

2103A51408

K.SAMPRADHA

Under the Guidance of

B.Narasimha

Submitted to



SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE
SR UNIVERSITY, ANANTHASAGAR, WARANGAL

April, 2024.



SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

CERTIFICATE

This is to certify that this project entitled “**NOTES&PASSWORD MANAGER** ” is the bonafied work carried out by **B.SAHITH,J.TEJA,M.DEEKSHITH,S.TEJASWI,K.SAMPRADHA** as a Minor Project for the partial fulfillment to award the degree **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE** during the academic year 2023-2024 under our guidance and Supervision.

Dr. B.Narasimha

Assistant Professor,

SR University,

Ananthasagar, Warangal.

Dr. M.Sheshikala

Assoc. Prof. & HOD(CSE),

SR University,

Ananthasagar, Warangal.

External Examiner

ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our project guide **Dr.B.Narasimha, Assistant Professor** as well as Head of the CSE Department **Dr. M.Sheshikala, Associate Professor** for guiding us from the beginning through the end of the Minor Project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We express our thanks to the project co-ordinators **Dr. P Praveen, Assoc. Prof** for their encouragement and support.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Dean, **Dr.Indrajeet Gupta**, for his continuous support and guidance to complete this project in the institute.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

B.SAHITH

J.TEJA

M.DEEKSHITH

S.TEJASWI

K.SAMPADHA

ABSTRACT

The difficulties of organizing chaotic notes and complicated passwords are addressed by this java program. Along with features for making, modifying, and organizing notes, it provides a safe platform to store passwords encrypted with industry-standard techniques. Unauthorized access is restricted by user authentication.

Password generation and note classification for better organization are optional features. Data portability and backup are made possible by exporting and importing. Jca, or java cryptography architecture, is a possible encryption library. A graphical user interface can be created with java swing or javafx.

The program helps users by increasing note organization, strengthening password security, and providing a practical all-in-one solution. Future development will include investigating cloud storage integration for safe backups and multi-device access, as well as adding features that users have requested. This program is intended for anyone looking for a safe and efficient method to keep track of passwords and notes.

Table of Contents

1.INTRODUCTION

1.1. EXISTING SYSTEM

1.2. PROPOSED SYSTEM

2. LITERATURE SURVEY

2.1. RELATED WORK

2.2. SYSTEM STUDY

3. DESIGN

3.1. REQUIREMENT SPECIFICATION (S/W & H/W)

4. IMPLEMENTATION

4.1. MODULES

4.2. OVERVIEW TECHNOLOGY

5. TESTING

5.1. TEST CASES

5.2 RESULTS

6. CONCLUSION

7. FUTURE SCOPE

BIBLIOGRAPHY

1. INTRODUCTION

1.1 EXISTING SYSTEM

Different tools and techniques are widely used in today's encryption and password management environment. Without integration and security, most users use notebooks, writing apps, and actual books to write. Password management also comes with the risk of security breaches and is not easy as it relies on memory, typing passwords or using simple browser functions. This inconsistency leads to security breaches, potential for data loss, and poor performance. There is an urgent need for an integrated system that combines password management and writing functionality. Such a platform will provide fast access, strong security features (such as encryption and password generation), and seamless organization. By integrating these important functions into a single system, users can increase efficiency, reduce security threats and ensure the protection of confidential information.

1.2 PROPOSED SYSTEM

Creating a notes and password manager system using Java involves several key steps. First, you need to design a user interface (UI) using Java Swing or JavaFX, providing separate interfaces for notes and passwords management. Next, develop data models for notes and passwords, incorporating attributes like title, content, and creation date. Implementing storage mechanisms, whether file-based or database-driven, comes next, ensuring sensitive data like passwords is securely encrypted.

Functionality encompasses adding, viewing, editing, and deleting notes and passwords, alongside search features and password generation capabilities. Security is paramount, requiring authentication measures and secure coding practices to prevent vulnerabilities. For password storage, employ robust encryption, hashing, and salting techniques to safeguard against breaches.

Thorough testing is crucial, including unit tests for individual components, integration testing for system-wide functionality, and user testing to refine usability. Documentation should accompany the codebase, facilitating understanding and contribution. Finally, deployment involves packaging the application into an executable JAR file for distribution and potentially deploying it on cloud platforms or creating installers for various operating systems.

By following these steps, you can create a reliable and secure notes and password manager system using Java. This project not only hones your Java skills but also familiarizes you with essential software development practices, making it a valuable learning experience.

2. LITERATURE SURVEY

2.1. RELATED WORK

Several existing software solutions provide notes and password management functionalities, offering insights and inspiration for your Java-based project:

- **LastPass:** LastPass is a popular password manager that securely stores passwords and personal information in a vault. It offers features like password generation, autofill, and multi-factor authentication. Analyzing its design and features can provide valuable insights for implementing password management in your Java application.
- **Evernote:** Evernote is a versatile note-taking application that allows users to create, organize, and synchronize notes across devices. It supports various media types, including text, images, and audio recordings. Examining Evernote's user interface and synchronization mechanisms can inspire the development of your Java-based notes manager.
- **1Password:** 1Password is another prominent password manager that stores passwords, credit card information, and other sensitive data in a secure vault. It features strong encryption, password auditing, and sharing capabilities. Studying its security features and user experience can inform the development of your password manager component.
- **Microsoft OneNote:** OneNote is a digital notebook application offered by Microsoft, enabling users to create and organize notes in a free-form manner. It supports multimedia content, handwriting recognition, and collaborative editing. Exploring OneNote's organization features and rich text editing capabilities can influence the design of your Java-based notes manager.
- **KeePass:** KeePass is an open-source password manager that stores passwords in an encrypted database. It offers strong security features, including two-factor authentication and plugin support. Reviewing KeePass's encryption techniques and plugin architecture can provide guidance for implementing security measures in your Java application.

2.2 SYSTEM STUDY

System study for developing a notes and password manager system using Java involves gathering requirements through interviews or surveys, analyzing existing systems, and defining objectives and constraints. It aims to understand user needs, functionality, and feasibility. By assessing technical, economic, and operational aspects, the study determines the viability of the project within specified constraints such as budget and time frame. User preferences and feedback are integral, guiding feature prioritization and interface design. Risks are identified, and mitigation strategies are devised to ensure project success.

This comprehensive analysis lays the groundwork for subsequent phases of development, facilitating effective design, implementation, testing, and deployment. By conducting a thorough system study, developers can ensure that the Java-based notes and password manager system meets user requirements, aligns with organizational goals, and is developed within the specified constraints, ultimately leading to a successful and impactful solution.

3.DESIGN

3.1. REQUIREMENT SPECIFICATION(S/W & H/W)

Requirement Specification for the notes and password manager system using Java and XAMPP (Apache, MySQL, PHP, Perl):

Software Requirements:

- **Java Development Kit (JDK):** Required for developing the Java-based application.
- **Integrated Development Environment (IDE):** Recommended IDEs include Eclipse, IntelliJ IDEA, or NetBeans for Java development.
- **XAMPP:** Provides an Apache web server, MySQL database, and PHP for server-side scripting.
- **MySQL Database:** Used to store user data including notes and passwords.
- **JDBC (Java Database Connectivity):** Java API for connecting Java applications with relational databases like MySQL.
- **JavaFX or Swing:** For developing the graphical user interface (GUI) of the application.

Hardware Requirements:

- **Computer or server capable of running XAMPP:** Minimum requirements depend on the expected usage and scale of the application.
- **Adequate RAM and storage space:** Sufficient memory and storage to run XAMPP and store user data.
- **Network connectivity:** Required for accessing the application if deployed on a server.

Database Connection:

- Establish a connection between the Java application and the MySQL database using JDBC.
- Configure XAMPP to ensure MySQL is running and accessible.
- Use appropriate JDBC drivers to connect to the MySQL database from Java code.
- Implement CRUD (Create, Read, Update, Delete) operations to interact with the database, managing notes and passwords.

TECHNOLOGY DESCRIPTION

Java:

Java is a versatile and widely-used programming language known for its platform independence and robustness. It provides extensive libraries and frameworks for developing various types of applications, including desktop and web-based solutions.

Java Database Connectivity (JDBC):

JDBC is a Java API that enables Java applications to interact with relational databases. It provides a standard interface for connecting to databases, executing SQL queries, and managing database connections, transactions, and result sets.

JavaFX or Swing:

JavaFX and Swing are Java libraries used for developing graphical user interfaces (GUIs). They provide components and tools for creating interactive and visually appealing user interfaces for desktop applications.

XAMPP:

XAMPP is a free and open-source cross-platform web server solution stack that includes Apache, MySQL, PHP, and Perl. It provides an easy-to-install package for setting up a local web server environment for development and testing purposes.

Apache:

Apache is a widely-used open-source web server software that powers a large portion of the internet. In the context of XAMPP, Apache serves as the web server component responsible for handling HTTP requests and serving web pages.

MySQL:

MySQL is an open-source relational database management system (RDBMS) known for its reliability, performance, and scalability. It is used as the database component of the XAMPP stack for storing and managing user data, including notes and passwords.

PHP:

PHP is a server-side scripting language used for developing dynamic web applications and interacting with databases. While not directly used in the Java-based application, PHP may be utilized for server-side scripting in conjunction with XAMPP for certain functionalities like user authentication or data validation.

4. IMPLEMENTATION

4.1MODULES

1. User Interface (UI):

This module focuses on designing and implementing the graphical user interface (GUI) for the application.

Sub-modules may include separate interfaces for notes management and password management, as well as screens for authentication and user settings.

2. Data Access Layer:

Responsible for establishing connections to the MySQL database using JDBC.

Handles CRUD operations (Create, Read, Update, Delete) for managing notes and passwords in the database.

Provides an abstraction layer between the application logic and the database, ensuring separation of concerns.

3. Notes Manager:

Handles functionalities related to managing notes, such as adding, viewing, editing, and deleting notes.

Includes features for organizing notes into categories or folders, searching for specific notes, and tagging notes with keywords.

4. Password Manager:

Manages functionalities related to storing, retrieving, updating, and deleting passwords.

Includes features for generating strong passwords, categorizing passwords by type or usage, and securely storing passwords with encryption.

5. User Authentication and Authorization:

Manages user authentication and authorization processes to ensure secure access to the application.

Includes features for user registration, login, password reset, and session management.

Enforces access control policies to restrict unauthorized access to sensitive functionalities or data.

6. Settings and Preferences:

Allows users to customize application settings and preferences according to their preferences.

Includes options for configuring display settings, notification preferences, and security settings.

7. Integration Testing:

Conducts integration testing to ensure that individual modules interact correctly with each other.

Verifies that data flows smoothly between modules and that functionalities work as expected in a unified system.

4.2 OVERVIEW TECHNOLOGY

1.Java:

Java is the primary programming language used for developing the application logic, including user interface, business logic, and database connectivity.

Its platform independence and extensive libraries make it suitable for creating cross-platform desktop applications.

2.JavaFX or Swing:

JavaFX or Swing is utilized for building the graphical user interface (GUI) of the application.

These libraries provide components and tools for creating interactive and visually appealing interfaces for users to interact with.

3.MySQL Database:

MySQL is employed as the relational database management system (RDBMS) for storing and managing user data, including notes and passwords.

Its reliability, performance, and scalability make it suitable for handling data storage requirements of the application.

4.JDBC (Java Database Connectivity):

JDBC is used for establishing connections to the MySQL database from the Java application.

It provides a standard API for executing SQL queries, managing database connections, and processing result sets, enabling seamless interaction between the application and the database.

5.XAMPP:

XAMPP provides a local development environment consisting of Apache, MySQL, PHP, and Perl. It simplifies the setup of a web server environment for testing and development purposes, enabling developers to work with the Apache web server and MySQL database locally.

5.TESTING

5.1 Implementation of code in python:

LoginPage.java :

```
import java.awt.EventQueue;
import javax.swing.JFrame; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JTextField; import
javax.swing.JPasswordField; import
javax.swing.JButton; import
javax.swing.JOptionPane;
public class LoginPage {
private JFrame frame;
private JTextField textField;
private JPasswordField passwordField;
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
LoginPage window = new LoginPage();
window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}
5
public LoginPage() {
initialize();
}
private void initialize() {
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
JLabel lblNewLabel = new JLabel("USERNAME :");
lblNewLabel.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
lblNewLabel.setBounds(127, 30, 133, 23);
frame.getContentPane().add(lblNewLabel);
textField = new JTextField();
textField.setBounds(127, 70, 133, 30);
```

```
frame.getContentPane().add(textField)
```

```
; textField.setColumns(10);  
JLabel lblNewLabel_1 = new JLabel("PASSWORD :");  
lblNewLabel_1.setFont(new Font("Times New  
Roman", Font.BOLD | Font.ITALIC, 14));  
lblNewLabel_1.setBounds(127, 111, 97, 23);  
frame.getContentPane().add(lblNewLabel_1);  
JButton btnNewButton = new JButton("Login");  
btnNewButton.setFont(new Font("Times New Roman",  
Font.BOLD | Font.ITALIC, 14));  
btnNewButton.setBounds(145, 202, 97, 23);  
frame.getContentPane().add(btnNewButton);
```

```
6
```

```
passwordField = new JPasswordField();  
passwordField.setBounds(127, 145, 133, 30);  
frame.getContentPane().add(passwordField);  
btnNewButton.addActionListener(e -> { String username  
= textField.getText(); char[] password =  
passwordField.getPassword();  
if (isValidCredentials(username, password)) {  
frame.dispose();  
openPasswordManager();  
} else {  
JOptionPane.showMessageDialog(frame, "Login  
failed. Please check your credentials.", "Error",  
JOptionPane.ERROR_MESSAGE);  
}  
}); } private boolean isValidCredentials(String  
username,  
char[] password) {  
String validUsername = "JAVAPRO"; String  
validPassword = "Password"; return  
username.equals(validUsername) && new  
String(password).equals(validPassword);  
}  
// Open the PasswordManager application private  
void openPasswordManager() {  
PasswordManager.main(new String[]{});  
}  
}  
PasswordManager.java : import  
java.awt.EventQueue; import  
javax.swing.JFrame; import  
javax.swing.JButton; import
```

```
7
```

```
java.awt.Font; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener; import
javax.swing.JLabel; import
```

```
javax.swing.SwingConstants;
public class PasswordManager {
private JFrame frame; private PasswordLength
passwordLengthWindow; private EncryptText
encryptTextWindow; private StoreYourPassword
storeYourPasswordWindow; private EnterAccount
enterAccountWindow; private AddNote
addNoteWindow;
private GetYourNotes getYourNotesWindow;
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
PasswordManager window = new
PasswordManager(); window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}
public PasswordManager() {
initialize();
}
private void initialize() {
8
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
JButton btnGeneratePassword = new
JButton("GENERATE PASSWORD");
btnGeneratePassword.setFont(new Font("Arial",
Font.BOLD | Font.ITALIC, 11));
btnGeneratePassword.setBounds(137, 44, 173, 23);
frame.getContentPane().add(btnGeneratePassword);
JButton btnEncryptText = new JButton("ENCRYPT
Text"); btnEncryptText.setFont(new Font("Arial",
Font.BOLD |
Font.ITALIC, 11)); btnEncryptText.setBounds(137,
```



```

76, 173, 23);
frame.getContentPane().add(btnEncryptText);
JButton btnStorePassword = new JButton("STORE
PASSWORD");
btnStorePassword.setFont(new Font("Arial",
Font.BOLD | Font.ITALIC, 11));
btnStorePassword.setBounds(137, 110, 173, 23);

```

```

frame.getContentPane().add(btnStorePassword);
JButton btnSearchPassword = new JButton("SEARCH
PASSWORD");
btnSearchPassword.setFont(new Font("Arial",
9
Font.BOLD | Font.ITALIC, 11));
btnSearchPassword.setBounds(137, 147, 173, 23);
frame.getContentPane().add(btnSearchPassword);
JButton btnAddNote = new JButton("ADD NOTE");
btnAddNote.setFont(new Font("Arial", Font.BOLD |
Font.ITALIC, 11)); btnAddNote.setBounds(137,
181, 173, 23);
frame.getContentPane().add(btnAddNote);
JButton btnGetNote = new JButton("GET NOTES"); //
Changed button text btnGetNote.setFont(new
Font("Arial", Font.BOLD |
Font.ITALIC, 11)); btnGetNote.setBounds(137,
215, 173, 23);
frame.getContentPane().add(btnGetNote);
JLabel lblNewLabel = new JLabel("HOME");
lblNewLabel.setHorizontalAlignment(SwingConstants.CEN
TER); lblNewLabel.setFont(new Font("Times New
Roman",
Font.BOLD, 14)); lblNewLabel.setBounds(0,
11, 434, 22);
frame.getContentPane().add(lblNewLabel)
;
addNoteWindow = new AddNote();
getYourNotesWindow = new GetYourNotes();
btnGeneratePassword.addActionListener(new
ActionListener() {
public void actionPerformed(ActionEvent e) {
10
if (passwordLengthWindow == null) {
passwordLengthWindow = new
PasswordLength();
}
passwordLengthWindow.getFrame().setVisible(true);

```

```

}
});
btnEncryptText.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) { if
(encryptTextWindow == null) {
encryptTextWindow = new EncryptText();
}
encryptTextWindow.getFrame().setVisible(true);
}

```

```

});
btnStorePassword.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) {
if (storeYourPasswordWindow == null) {
storeYourPasswordWindow = new
StoreYourPassword();
}
storeYourPasswordWindow.getFrame().setVisible(true);
}
});

```

```

btnSearchPassword.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) {
if (enterAccountWindow == null) {
11
enterAccountWindow = new EnterAccount();
}
enterAccountWindow.setStoreYourPassword(storeYourPass
wordWindow);
enterAccountWindow.getFrame().setVisible(true); }
});
btnAddNote.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
addNoteWindow.getFrame().setVisible(true);
}
});
btnGetNote.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
getYourNotesWindow.getFrame().setVisible(true);
}
});
}
}

```

PasswordLength.java : import

```

java.awt.EventQueue; import
javax.swing.JFrame; import
javax.swing.JTextField; import
java.awt.Font; import
javax.swing.JLabel; import
javax.swing.JButton; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener;
public class PasswordLength {
private JFrame frame;
12
private JTextField textField; private
CopyYourPassword passwordLengthWindow; private

```

```

String generatedPassword;
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
PasswordLength window = new
PasswordLength(); window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}
public PasswordLength() {
initialize();
}
public JFrame getFrame() {
return frame;
}
private void initialize() {
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CL
OSE); frame.getContentPane().setLayout(null);
textField = new JTextField();
textField.setBounds(110, 107, 163, 20);
13
frame.getContentPane().add(textField);
textField.setColumns(10);
JLabel lblNewLabel = new JLabel("Enter the password
length :");
lblNewLabel.setFont(new Font("Times New Roman",

```

```

Font.PLAIN, 13)); lblNewLabel.setBounds(110,
57, 196, 26);
frame.getContentPane().add(lblNewLabel);
JButton btnNewButton = new JButton("OK");
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 13));
btnNewButton.setBounds(120, 149, 52, 23);
frame.getContentPane().add(btnNewButton);
JButton btnNewButton_1 = new JButton("Cancel");
btnNewButton_1.setFont(new Font("Times New
Roman", Font.BOLD, 13));
btnNewButton_1.setBounds(182, 149, 78, 23);
frame.getContentPane().add(btnNewButton_1);
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) {
int passwordLength =

```

```

Integer.parseInt(textField.getText());
generatedPassword =
generatePassword(passwordLength);
frame.dispose(); if
(passwordLengthWindow == null) {
passwordLengthWindow = new
CopyYourPassword(generatedPasswo
rd);
14
}
passwordLengthWindow.getFrame().setVisible(true);
}
});
btnNewButton_1.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) {
frame.dispose();
}
});
}
private String generatePassword(int length) {
String charset =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*";
StringBuilder password = new StringBuilder();
for (int i = 0; i < length; i++) {
int randomIndex = (int) (Math.random() *
charset.length());
password.append(charset.charAt(randomIndex));

```

```

    }
    return password.toString();
}
}
CopyYourPassword.java : import
java.awt.EventQueue; import
javax.swing.JFrame; import
15
javax.swing.JTextField; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JButton; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener;
public class CopyYourPassword {
    private JFrame frame;
    private JTextField textField;
    public static void main(String[] args) {

```

```

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    CopyYourPassword window = new
                    CopyYourPassword(""); window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
    public CopyYourPassword(String generatedPassword) {
        initialize(generatedPassword);
    }
    public JFrame getFrame() {
        return frame;
    }
16
    private void initialize(String generatedPassword) {
        frame = new JFrame();
        frame.setBounds(100, 100, 450, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
    );
        frame.getContentPane().setLayout(null);
        textField = new JTextField();
        textField.setBounds(62, 50, 276, 154);

```

```

textField.setText(generatedPassword); // Set the
generated password
frame.getContentPane().add(textField);
textField.setColumns(10);
JLabel lblNewLabel = new JLabel("Copy your
password :"); lblNewLabel.setFont(new Font("Times New
Roman",
Font.BOLD | Font.ITALIC, 14));
lblNewLabel.setBounds(62, 22, 182, 17);
frame.getContentPane().add(lblNewLabel);
JButton btnNewButton = new JButton("OK");
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(160, 215, 71, 23);
frame.getContentPane().add(btnNewButton);
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) {
frame.dispose();
}
});
}
17
}

```

EncryptText.java :

```

import java.awt.EventQueue; import
javax.swing.JFrame; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JTextField; import
javax.swing.JButton; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener;
public class EncryptText {
private JFrame frame;
private JTextField textField;
private SecretKey secretKeyWindow;
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
EncryptText window = new EncryptText();
window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
}
}

```

```

}
});
}
public EncryptText() {
initialize();
}
18
private void initialize() {
frame = new JFrame();
frame.getContentPane().setFont(new Font("Times New
Roman", Font.PLAIN, 13)); frame.setBounds(100,
100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
JLabel lblNewLabel = new JLabel("Enter the text to
encrypt :"); lblNewLabel.setFont(new Font("Times New
Roman",
Font.PLAIN, 15)); lblNewLabel.setBounds(92,
68, 183, 14);
frame.getContentPane().add(lblNewLabel)
;
textField = new JTextField();
textField.setBounds(92, 105, 168, 20);
frame.getContentPane().add(textField)
; textField.setColumns(10);

```

```

JButton btnNewButton = new JButton("OK");
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(102, 143, 53, 23);
frame.getContentPane().add(btnNewButton);
JButton btnNewButton_1 = new JButton("Cancel"); btnNewButton_1.setFont(new
Font("Times New Roman", Font.BOLD | Font.ITALIC, 14));
btnNewButton_1.setBounds(160, 144, 89, 23);
frame.getContentPane().add(btnNewButton_1);
19
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) { if
(secretKeyWindow == null) {
secretKeyWindow = new SecretKey();
} secretKeyWindow.getFrame().setVisible(true);
}
});
btnNewButton_1.addActionListener(new
ActionListener() { public void

```

```

    actionPerformed(ActionEvent e) {
        frame.dispose(); // Close the EncryptText window
    }
});
}

public JFrame getFrame() {
    return frame;
}
}

SecretKey.java :
import java.awt.EventQueue;
import javax.swing.JFrame; import
javax.swing.JTextField; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JButton; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener;
import java.util.Random;
public class SecretKey {
    20
    private JFrame frame;
    private JTextField textField;
    private CopyYourPassword copyYourPasswordWindow;
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    SecretKey window = new SecretKey();

                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    public SecretKey() {
        initialize();
    }

    public JFrame getFrame() {
        return frame;
    }

    private void initialize() {
        frame = new JFrame(); frame.setBounds(100,
        100, 450, 300);

```



```

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
);
frame.getContentPane().setLayout(null);
21
JLabel lblNewLabel = new JLabel("Enter the secret key : ");
lblNewLabel.setFont(new Font("Times New Roman",
Font.PLAIN, 14)); lblNewLabel.setBounds(77,
62, 202, 23);
frame.getContentPane().add(lblNewLabel)
;
textField = new JTextField();
textField.setBounds(77, 103, 193, 20);
frame.getContentPane().add(textField)
; textField.setColumns(10);
JButton btnNewButton = new JButton("OK");
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(87, 134, 69, 23);
frame.getContentPane().add(btnNewButton);
JButton btnNewButton_1 = new JButton("Cancel");
btnNewButton_1.setFont(new Font("Times New
Roman", Font.BOLD | Font.ITALIC, 14));
btnNewButton_1.setBounds(166, 135, 89, 23);
frame.getContentPane().add(btnNewButton_1);
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) {
if (copyYourPasswordWindow == null) {
String generatedPassword =
generateRandomPassword(); copyYourPasswordWindow =
new CopyYourPassword(generatedPassword);

}
22
copyYourPasswordWindow.getFrame().setVisible(true);
}
});
btnNewButton_1.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) {
frame.dispose(); // Close the SecretKey window
}
});
}
private String generateRandomPassword() {

```

```

String characters =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()-_+=[]{}|;:,<>?"; int
length = 12;
Random random = new Random();
StringBuilder password = new StringBuilder();
for (int i = 0; i < length; i++) {
int index = random.nextInt(characters.length());
password.append(characters.charAt(index));
}
return password.toString();
}
}

```

```

StoreYourPassword.java : import
java.awt.EventQueue; import
javax.swing.JFrame; import
javax.swing.JTextField; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JButton; import
23
java.awt.event.ActionEvent; import
java.awt.event.ActionListener; import
java.util.HashMap; import
java.util.Map; import
javax.swing.JOptionPane;
public class StoreYourPassword {
private JFrame frame; private JTextField
textFieldName; private JTextField textFieldPassword;
private Map<String, String> accountInfoMap = new
HashMap<>();
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
StoreYourPassword window = new

```

```

StoreYourPassword(); window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}
public StoreYourPassword() { initialize();
}
}

```

```

public JFrame getFrame() {
return frame;
}
public Map<String, String> getAccountInfoMap() {
24
return accountInfoMap;
}
private void initialize() {
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
JLabel lblNewLabel = new JLabel("ACCOUNT NAME
:");
lblNewLabel.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
lblNewLabel.setBounds(125, 22, 134, 32);
frame.getContentPane().add(lblNewLabel);
textFieldName = new JTextField();
textFieldName.setBounds(125, 66, 139, 32);
frame.getContentPane().add(textFieldName);
textFieldName.setColumns(10);
JLabel lblNewLabel_1 = new JLabel("ACCOUNT
PASSWORD :"); lblNewLabel_1.setFont(new
Font("Times New
Roman", Font.BOLD | Font.ITALIC, 14));
lblNewLabel_1.setBounds(125, 122, 197, 17);
frame.getContentPane().add(lblNewLabel_1);
textFieldPassword = new JTextField();
textFieldPassword.setBounds(125, 164, 139, 32);
25
frame.getContentPane().add(textFieldPassword);
textFieldPassword.setColumns(10);
JButton btnNewButton = new JButton("STORE");
btnNewButton.setFont(new Font("Times New Roman",

```

```

Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(101, 207, 89, 23);
frame.getContentPane().add(btnNewButton);

```

```

JButton btnNewButton_1 = new JButton("Cancel");
btnNewButton_1.setFont(new Font("Times New
Roman", Font.BOLD | Font.ITALIC, 14));

```

```

btnNewButton_1.setBounds(198, 207, 89, 23);
frame.getContentPane().add(btnNewButton_1);
btnNewButton_1.addActionListener(new
    ActionListener() { public void
        actionPerformed(ActionEvent e) {
            frame.dispose();
        }
    });
btnNewButton.addActionListener(new ActionListener()
    { public void actionPerformed(ActionEvent e) {
        String accountName = textFieldName.getText();
        String accountPassword =
            textFieldPassword.getText();
        if (accountName.isEmpty() ||
            accountPassword.isEmpty()) { showError("Fill in
            both account name and
            password properly.");
        } else {
            accountInfoMap.put(accountName,
26
            accountPassword); showMessage("Account
            information saved
            successfully."); textFieldName.setText("");
            textFieldPassword.setText("");
        }
    }
});
}

private void showError(String message) {
    JOptionPane.showMessageDialog(frame, message,
        "Error", JOptionPane.ERROR_MESSAGE);
}

private void showMessage(String message) {
    JOptionPane.showMessageDialog(frame, message,
        "Success", JOptionPane.INFORMATION_MESSAGE);
}
}

EnterAccount.java : import
java.awt.EventQueue; import
javax.swing.JFrame; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JTextField; import
javax.swing.JButton; import

```

```

javax.swing.JOptionPane;
public class EnterAccount {
private JFrame frame;
private JTextField textField;
private String accountName;
private StoreYourPassword storeYourPassword;
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
EnterAccount window = new EnterAccount();
window.frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}
public EnterAccount() {
initialize();
}
public JFrame getFrame() {
return frame;
}
public String getAccountName() {
return accountName;
}
public void setStoreYourPassword(StoreYourPassword
storeYourPassword) { this.storeYourPassword =
storeYourPassword;
}
28
private void initialize() {
frame = new JFrame();
frame.getContentPane().setFont(new Font("Times New
Roman", Font.PLAIN, 14)); frame.setBounds(100,
100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
JLabel lblNewLabel = new JLabel("Enter your
account name :"); lblNewLabel.setFont(new Font("Times
New Roman",
Font.PLAIN, 14)); lblNewLabel.setBounds(78,
64, 166, 17);

```

```

frame.getContentPane().add(lblNewLabel)
;
textField = new JTextField();
textField.setBounds(78, 104, 175, 27);
frame.getContentPane().add(textField)
; textField.setColumns(10);
JButton btnNewButton = new JButton("OK");
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(84, 154, 61, 27);
frame.getContentPane().add(btnNewButton); JButton
btnNewButton_1 = new JButton("Cancel");
btnNewButton_1.setFont(new Font("Times New
Roman", Font.BOLD | Font.ITALIC, 14));
btnNewButton_1.setBounds(155, 154, 89, 27);
frame.getContentPane().add(btnNewButton_1);
btnNewButton.addActionListener(e -> {
29
accountName = textField.getText(); if
(storeYourPassword != null &&
storeYourPassword.getAccountInfoMap().containsKey(accountName)) {
String accountPassword =
storeYourPassword.getAccountInfoMap().get(accountName)
;
CopyYourPassword copyYourPassword = new
CopyYourPassword(accountPassword);
copyYourPassword.getFrame().setVisible(true);
} else { showError("Account not
found!");
}
frame.dispose();
});
btnNewButton_1.addActionListener(e -> {
accountName = null;
frame.dispose();
});
}
private void showError(String message) {
JOptionPane.showMessageDialog(frame, message,
"Error", JOptionPane.ERROR_MESSAGE);
}
}

```

AddNote.java :

```

import java.awt.EventQueue; import
javax.swing.JFrame; import
javax.swing.JLabel; import

```

```

java.awt.Font; import
javax.swing.JTextField; import
javax.swing.JButton; import
javax.swing.JOptionPane; import
java.awt.event.ActionListener; import
java.awt.event.ActionEvent;
public class AddNote {
private JFrame frame;
private JTextField textField;
private static String storedNote;
public AddNote() { initialize();
}
public JFrame getFrame() {
return frame;
}
public static String getStoredNote() {
return storedNote;
}
private void initialize() {
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
frame.getContentPane().setLayout(null);
31
JLabel lblNewLabel = new JLabel("Add Note:");
lblNewLabel.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
lblNewLabel.setBounds(45, 30, 174, 22);
frame.getContentPane().add(lblNewLabel);
textField = new JTextField();
textField.setBounds(80, 63, 296, 140);
frame.getContentPane().add(textField)
; textField.setColumns(10);
JButton btnNewButton = new JButton("ADD NOTE");
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) {
String noteText = textField.getText(); if
(noteText.isEmpty()) {
JOptionPane.showMessageDialog(frame,
"Error: The text field is empty.");
} else { storedNote =
noteText;

```

```

JOptionPane.showMessageDialog(frame,
"Success: Note added!");
}
}
});
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(112, 214, 116, 23);
frame.getContentPane().add(btnNewButton); JButton
btnNewButton_1 = new JButton("Cancel");
btnNewButton_1.setFont(new Font("Times New
Roman", Font.BOLD | Font.ITALIC, 14));
btnNewButton_1.setBounds(241, 214, 89, 23);
frame.getContentPane().add(btnNewButton_1);
btnNewButton_1.addActionListener(new

ActionListener() { public void
actionPerformed(ActionEvent e) {
frame.dispose();
}
});
}
}

GetYourNotes.java : import
javax.swing.JFrame; import
javax.swing.JLabel; import
java.awt.Font; import
javax.swing.JTextField; import
javax.swing.JButton; import
java.awt.event.ActionListener; import
java.awt.event.ActionEvent;
public class GetYourNotes {
private JFrame frame;
private JTextField textField;
public GetYourNotes() { initialize();
}
public JFrame getFrame() {
return frame;
}
private void initialize() {
frame = new JFrame();
frame.setBounds(100, 100, 450, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE

);

```



```
frame.getContentPane().setLayout(null);
JLabel lblNewLabel = new JLabel("Get your notes:");
lblNewLabel.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
```

```
lblNewLabel.setBounds(35, 23, 162, 17);
frame.getContentPane().add(lblNewLabel);
textField = new JTextField();
textField.setBounds(55, 51, 297, 140);
```

```
frame.getContentPane().add(textField)
; textField.setColumns(10);
JButton btnNewButton = new JButton("OK");
btnNewButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e) { //
Close the GetYourNotes window
frame.setVisible(false);
}
});
btnNewButton.setFont(new Font("Times New Roman",
Font.BOLD | Font.ITALIC, 14));
btnNewButton.setBounds(237, 202, 57, 23);
frame.getContentPane().add(btnNewButton); JButton
btnNewButton_1 = new JButton("GET
NOTE"); btnNewButton_1.setFont(new Font("Times
New
Roman", Font.BOLD | Font.ITALIC, 14));
btnNewButton_1.setBounds(109, 202, 118, 23);
frame.getContentPane().add(btnNewButton_1);
btnNewButton_1.addActionListener(new
ActionListener() { public void
actionPerformed(ActionEvent e) {
34
textField.setText(AddNote.getStoredNote());
}
});
}
}
```

5.2 Results:

Login Page:



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. The text 'USERNAME :' is displayed above a text input field containing 'JAVAPRO'. Below this, the text 'PASSWORD :' is displayed above a password input field filled with seven dots. At the bottom center is a button labeled 'Login'.

Home Page:



A screenshot of a home window titled 'HOME'. It contains a vertical stack of seven buttons: 'GENERATE PASSWORD', 'ENCRYPT Text', 'STORE PASSWORD', 'SEARCH PASSWORD', 'ADD NOTE', and 'GET NOTES'.

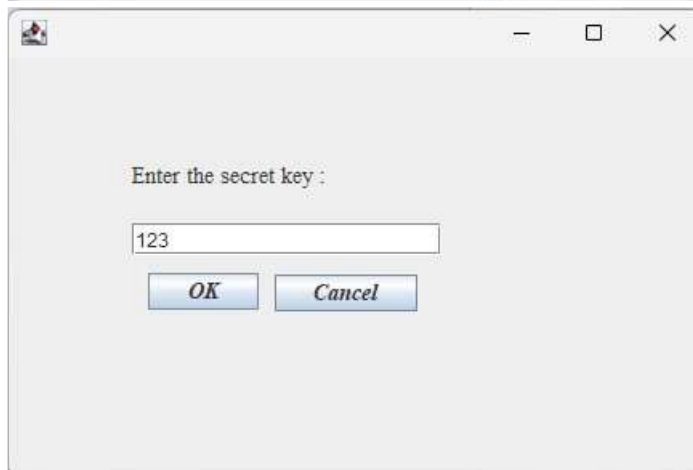
Generate Password:



A screenshot of a 'Generate Password' dialog box. It has a title bar with a small icon and standard window controls. The main area has a light gray background. The text 'Enter the password length :' is displayed above a text input field containing the number '6'. Below the input field are two buttons: 'OK' and 'Cancel'.



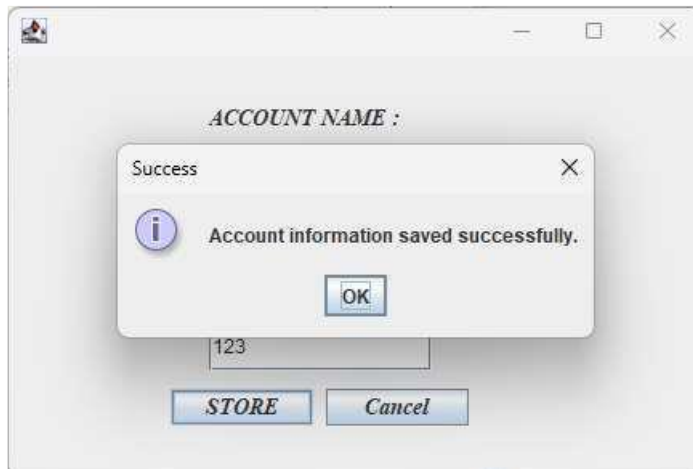
Encrypt Text:



Store Password:

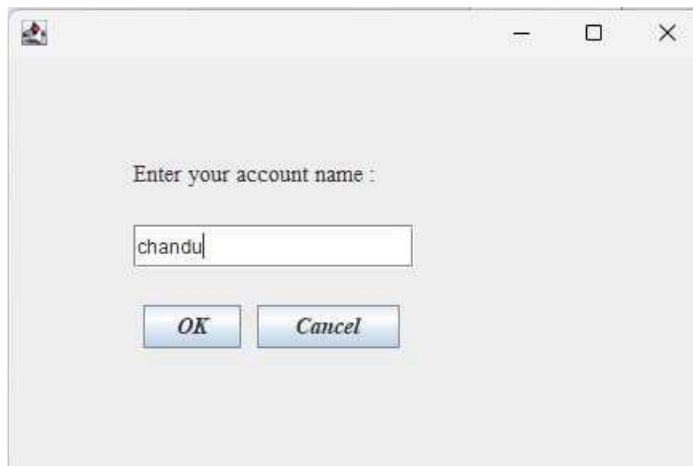


A screenshot of a Windows-style dialog box titled "Store Password". It contains two text input fields. The first field is labeled "ACCOUNT NAME :" and contains the text "chandu". The second field is labeled "ACCOUNT PASSWORD :" and contains the text "123". Below the fields are two buttons: "STORE" and "Cancel".



A screenshot of the same "Store Password" dialog box, but with a smaller "Success" dialog box overlaid on top. The "Success" dialog box has a title bar "Success" and a close button. It contains an information icon and the text "Account information saved successfully." Below this text is an "OK" button. The "STORE" and "Cancel" buttons from the background dialog are still visible.

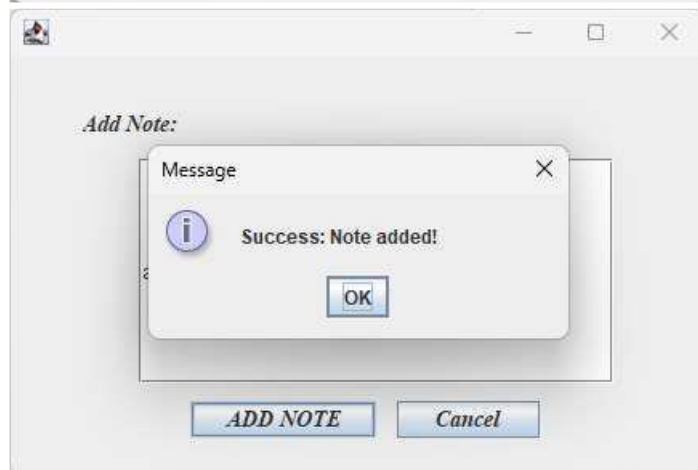
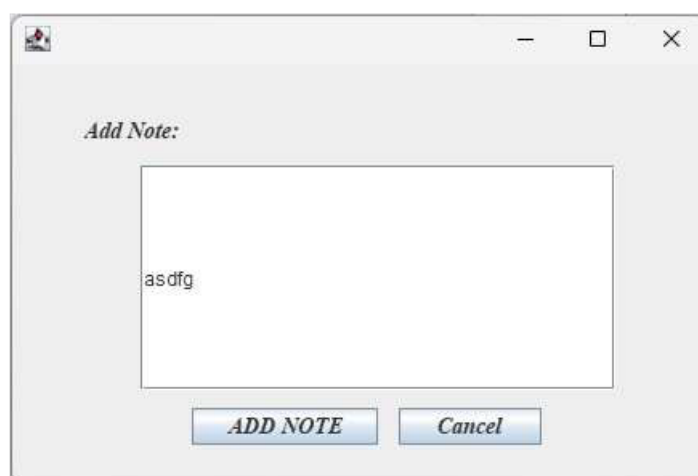
Search Password:




A screenshot of a Windows-style dialog box titled "Search Password". It contains a single text input field labeled "Enter your account name :". The field contains the text "chandu". Below the field are two buttons: "OK" and "Cancel".



Add Note:



Get Note:



-

□

×

Get your notes:

asdfg

GET NOTE

OK

6. CONCLUSION

The Notes and Password Manager project is a versatile and essential software application that addresses the critical need for managing and safeguarding sensitive information in today's digital age. This project's core objective was to create a user-friendly, secure, and feature-rich solution for effectively managing passwords and notes while prioritizing data security and user convenience.