**Mini Project Report on**



**Notes and Password Manager**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**  **University Roll No.:**

**ADARSH PATEL** **2019527**

***Under the Mentorship of***

**Ms. Tanusha Mittal**

**Designation**



**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**January 2023**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Notes and Password Manager”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineeringof the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Ms. Tanusha Mittal,** Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

Name **ADARSH PATEL** University Roll no. **2019527 signature: adarshpatel**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction | 2 - 3 |
| Chapter 2 | Literature Survey | 4 |
| Chapter 3 | Methodology | 5 - 6 |
| Chapter 4 | Result and Discussion | 8 |
| Chapter 5 | Conclusion and Future Work | 9 |
|  | References | 10 |

**1**

**Chapter 1**

**Introduction**

In the Following sections, a brief introduction and the problem statement for the work has been included.

The given Problem Statement is to make **Notes and password manager app.**  
It’s a GUI-based project used with the swing library to organize all the elements that work under the **Notes and Password Manager.**

# Cryptographic Use and Management

Cryptography is used to ensure the confidentiality and integrity of server data in memory

and in the repository, as well as all data transmitted between the Waveset

server and Gateways.

Notes and Password Manager is an application that helps users securely store and organize their notes and passwords. It offers features such as **Password Generate, Encryption text, Decrypt text, Store Password, Search Password, Delete Password, Add Note and Display Note.** The app allows users to store all of their confidential information in one place. The app also offers a variety of customization options and the ability to share information with others. In addition, Notes and Password Manager also provides users with an easy-to-use and intuitive interface, making it easy to manage their information.

**Features**

* Generate a random Password of a length >4
* Store and Retrieve passwords
* Delete a password
* Encrypt a plain text with a secret key
* Decrypt the encrypted text with the secret key
* Add a note and view the note added

**2**

The code is written in Java. It includes various imports such as **java.awt, javax.swing, java.awt.event, and java.util**. It also imports various packages related to encryption and decryption such as **java.security, java.io, and javax.crypto.** The code includes classes such as **ActionEvent, ActionListener, and SecureRandom** and interfaces such as **AlgorithmParameterSpec and KeySpec**. The code also includes methods related to encryption and decryption such as **init, doFinal, and generateSecret**. Finally, the code uses **Base64 for encoding and decoding.**

**3**

**Chapter 2**

**Literature Survey**

Java is an object-oriented programming language and is one of the most popular

programming languages in the world. It is mainly used for developing **web**

**applications, mobile applications, and desktop applications**.

**java.awt:** The java.awt package contains classes and interfaces for creating graphical user interfaces (GUIs) and for painting graphics and images. It includes classes for components like buttons, labels, and text fields, as well as layout managers and event handling classes.

**javax. swing:** The javax. swing package contains classes and interfaces for creating lightweight GUI components. It contains classes for components like JFrame, JPanel, JButton, as well as layout managers and event handling classes. It also contains utility classes for providing a look and feel that is consistent across platforms.

**java.awt.event:** The java.awt.event package contains classes and interfaces for handling events generated by AWT components. It contains classes for event listeners, event objects, and event adapters.

**java.util:** The java. util package contains classes for various utility purposes. It contains classes for collections, date/time handling, logging, internationalization, regex, and other miscellaneous utilities.

**Java.security:** The java.security package provides a set of classes and interfaces for the security framework. It contains classes for cryptography, public key management, access control, secure communication, and other security-related functions.

**Java.io:** The java.io package provides APIs for reading and writing data from/to files, sockets, and other input/output sources. It contains classes for dealing with files, streams, serialization, and other IO related tasks.

**Javax.crypto:** The javax.crypto package provides APIs for cryptography and security. It contains classes for symmetric and asymmetric encryption, digital signatures, message authentication codes, and other cryptographic functions. It also contains classes for secure communication, authentication, and access control.

**4**

**Chapter 3**

**Methodology**

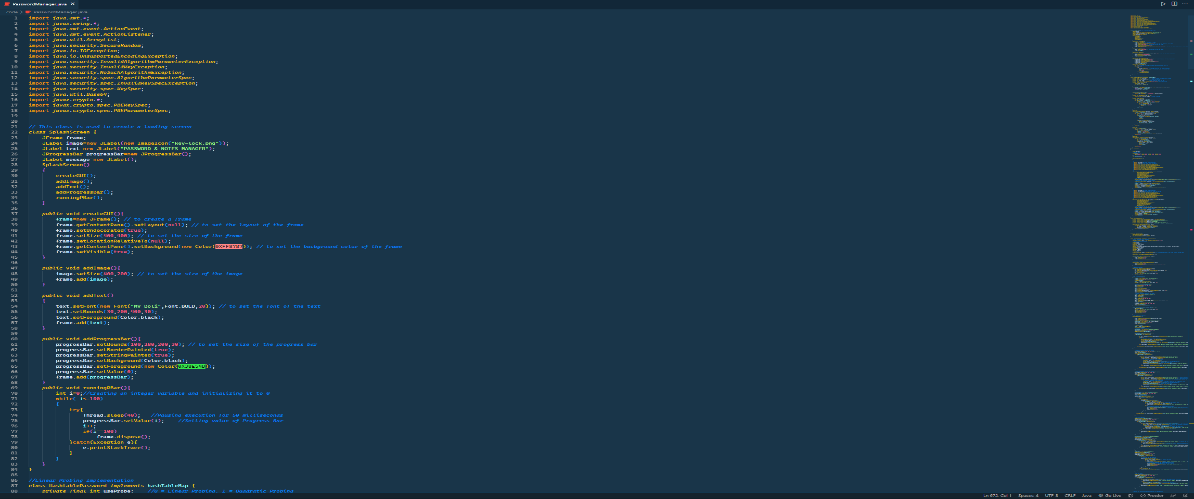
1. **Design:** This is the initial step of the development process. It involves

understanding the requirements of the project and designing the architecture of the application. This includes deciding on the technologies to be used, the programming language to be used, the database structure, and the overall design of the application.

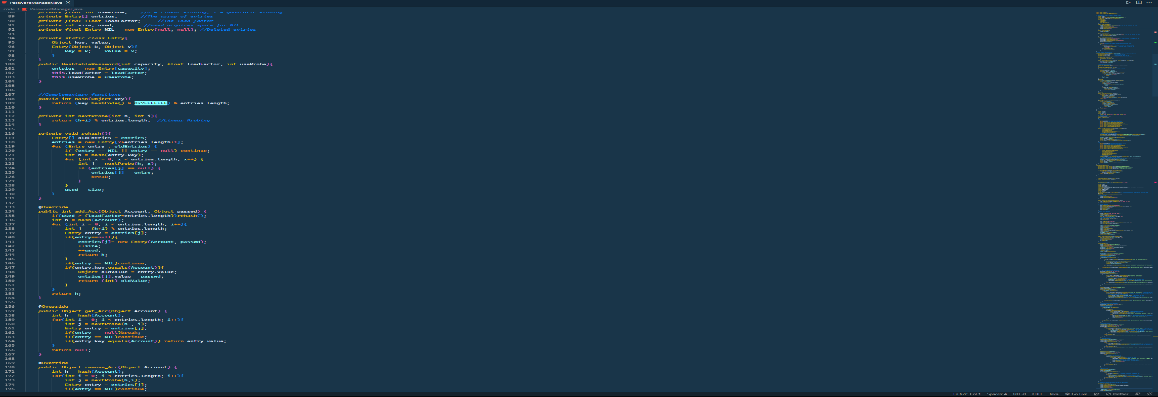


**Fig 3.1**

1. **Coding:** After the design phase is complete, the coding begins. This involves writing the code for the application in the chosen programming language. This includes writing the code for the user interface, the database structure, the business logic, and integrating the application with any external services.

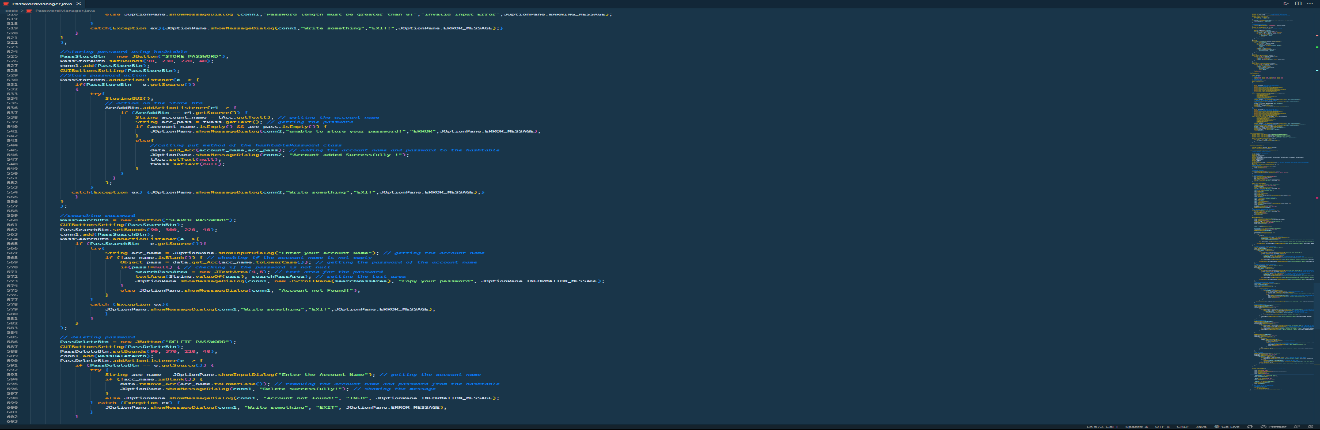
****

**Fig. 3.2**

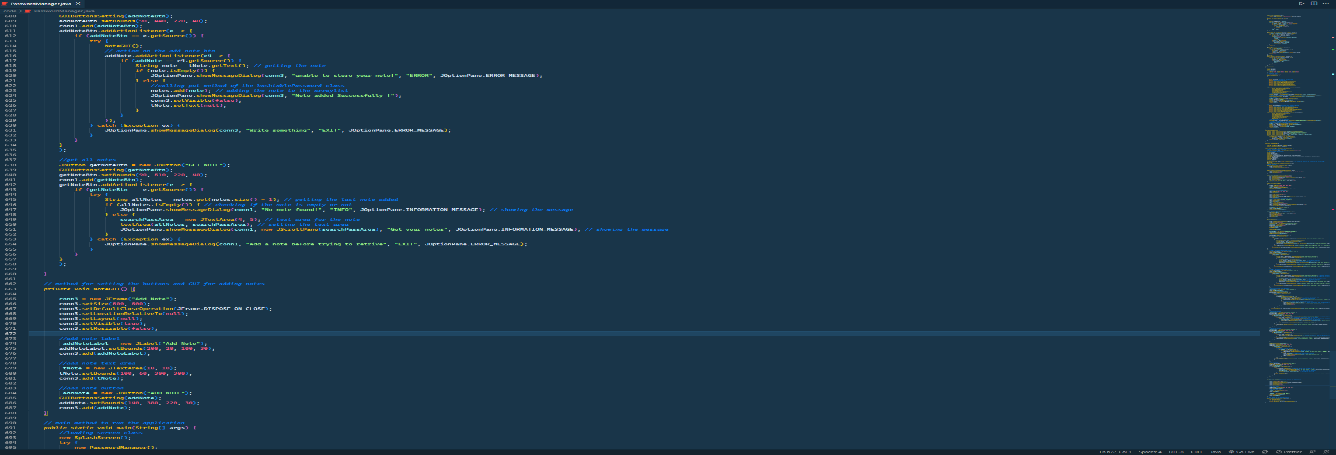


**Fig 3.2.1**

**5**



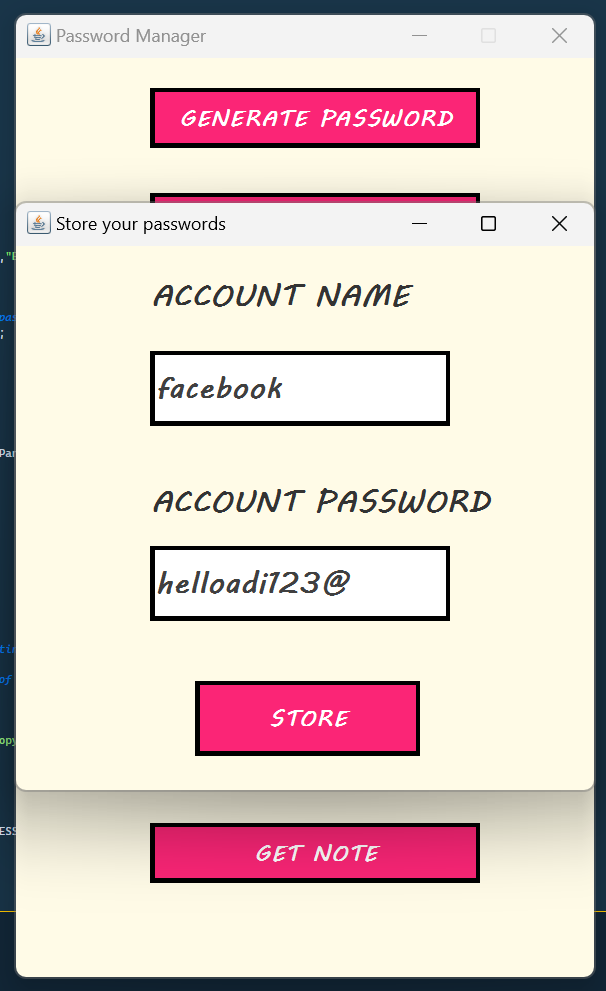
**Fig 3.2.2**



**Fig 3.2.3**

1. **Testing:** Once the coding phase is complete, the application needs to be tested to ensure that it works as expected. This involves testing the application for bugs, performance, and usability.

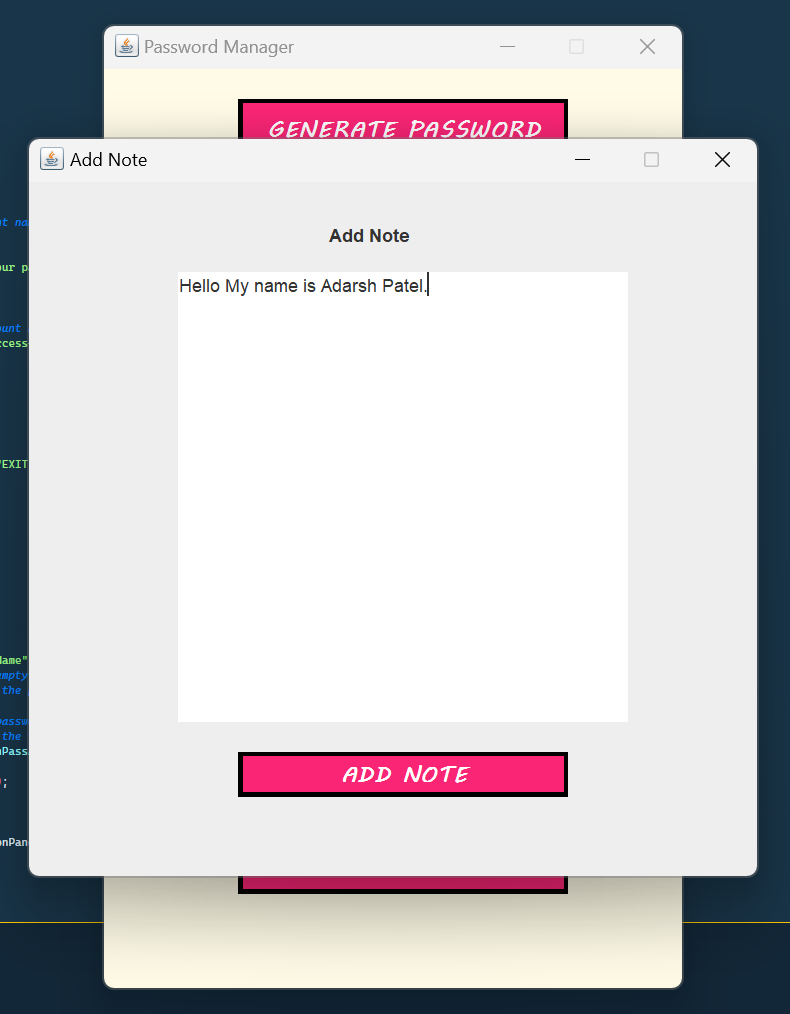
**Password Testing-:**



**Fig 3.1**

**6**

**Notes Testing-:**

****

**Fig 3.2**

1. **Maintenance:** After the application is deployed, it needs to be maintained. This involves fixing any bugs that are discovered, adding new features, and updating the application as required.

**7**

**Chapter 4**

**Result and Discussion**

We have Successfully created the app and tested our app with some sample inputs

and our outputs are shown below separately for Notes and Password that were

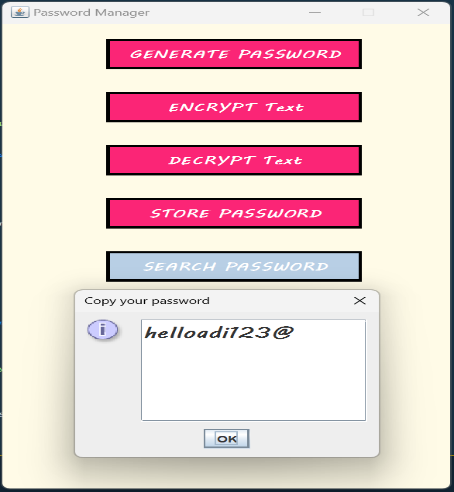
Processed.

**Output of Notes-:**

****

**Fig 4.1**

**Output of Password-:**



**Fig 4.2.1 Fig 4.2.2**

Java is an excellent language for developing applications, including Notes and Password managers. It offers a number of advantages, such as a large library of components and packages, the ability to create custom user interfaces, and excellent security. It should be easy to use, secure, and should be able to store both the passwords and the associated usernames. It should also provide users with the ability to access their passwords from multiple devices. Java provides a high level of security by encrypting data and using digital signatures.

**8**

**Chapter 5**

**Conclusion and Future Work**

At the conclusion of making notes and Password Manager app, the developer should ensure that the app has a good user interface, is secure and efficient and has a backup system. The user should be able to easily store, manage, and track all their passwords, notes, and other sensitive information. The app should also have warning notifications when the user is about to enter a wrong credential. Additionally, the app should provide a secure storage system which is difficult to breach. Furthermore, the app should also have an automatic backup system which saves the data in the cloud. Finally, the developer should also provide a help section and FAQs to help the user with any issues they may have.

1. **Biometrics:** Biometrics is a technology that uses physical characteristics, such as fingerprints and retinal scans, to identify users. This technology is useful for securing notes and passwords as it provides a more secure alternative to traditional passwords.
2. **Artificial Intelligence:** AI-powered notes and password managers can be used to automatically store, encrypt, and secure notes and passwords. These systems can also use predictive analytics to suggest the most secure passwords and detect security threats in real time.
3. **Cloud Storage:** Cloud-based notes and password management systems can provide a secure way to store and access data from any device or location. These systems can also be used to securely share notes and passwords with other users.
4. **Blockchain:** Blockchain-based notes and password managers can be used to store and share data in a secure and immutable way. This technology can also be used to securely store and verify digital signatures.
5. **Natural Language Processing:** Natural language processing can be used to automatically generate secure passwords based on a user’s natural language inputs. This technology can also be used to detect security threats in real time and suggest the most secure passwords.

**9**

**References**

1. JavaFX Tutorial: <https://www.jetbrains.com/help/idea/javafx.html>
2. JavaFX Scene Builder Tutorial: <https://docs.oracle.com/javase/8/scene-builder-2/get-started-tutorial/jfxsb-get_started.htm>
3. JavaFX Properties Tutorial: <https://docs.oracle.com/javase/8/javafx/properties-binding-tutorial/binding.htm>
4. JavaFX Database Connectivity Tutorial: <https://www.tutorialspoint.com/javafx/javafx_database.htm>
5. JavaFX GUI Design Tutorial: <https://docs.oracle.com/javase/8/javafx/get-started-tutorial/form.htm>
6. Core Java Volume I—Fundamentals by Cay S. Horstman
7. JavaFX File System Tutorial: <https://www.tutorialspoint.com/javafx/javafx_file_chooser.htm>
8. JavaFX Animation Tutorial: <https://docs.oracle.com/javase/8/javafx/graphics-tutorial/basics.htm>
9. JavaFX Security Tutorial: <https://docs.oracle.com/javase/8/javafx/security-tutorial/index.html>
10. JavaFX Event Handling Tutorial: <https://docs.oracle.com/javase/8/javafx/events-tutorial/jfxpub-events.htm>

**10**