**Password Manager with Encryption**

**Name:** Vinusha Garlapati

1. **Introduction of Password Manager:**

In the current landscape, safeguarding passwords stands as a paramount concern. To address this, we're crafting a Password Manager that utilizes a master key for login validation and employs AES encryption to secure passwords, storing them within an SQLite database. This initiative prioritizes a user-friendly interface, facilitating easy interaction and the addition of details to the Database.

1. **Architecture of Password Manager:**

Below is the workflow or architecture of how our Password Manager works.

A diagram of a diagram

Description automatically generated

The workflow of our password is as shown above. Let’s look at them in details:

1. First a window pops up asking to create master password as shown below and we need to create the password and confirm the password again.

A screenshot of a computer

Description automatically generated

1. If both the passwords match, then we will be returned to the UI.
2. In the third step a new window appears, and we need to login in with the Master Password.

A screenshot of a computer

Description automatically generated

1. If the Password Matches with Master Password the Password Manager will let you interact with it. Now a new welcome window pops up and provides you option to add your passwords to database or to view the passwords.

A screenshot of a computer

Description automatically generated

1. First let us add some passwords by clicking on Add Passwords option. A new window appears asking you to enter the website name, username, password. There is an option to generate strong and random password by clicking on generate password option. You can also see the password by clicking on the show password button.

A screenshot of a computer screen

Description automatically generated

1. There is an option to generate strong and random password by clicking on generate password option. You can also see the password by clicking on the show password button.
2. After pressing the submit button the password will be stored in the SQLite database named GarlapatiDB. The data will be stored in the table named Passwords under respective columns.
3. There is an option to view the database, by clicking on View Password.
4. Now we need to retrieve the Passwords table from database by using the SQL query “SELECT \* FROM Passwords;”.
5. A new window pops up and shows us the Passwords table with encrypted passwords stored in it.

In this way our password manager works and secures the data of the users.

1. **Technologies Used:**

In this project we made use of different tools and frameworks. Let’s discuss them in depth.

1. **Python:** Python stands out as a flexible, user-friendly programming language, prized for its clear, straightforward code. Its wealth of libraries supports a range of uses, spanning from web development and data analysis to AI and automation. This accessibility and adaptability appeal to both newcomers and seasoned developers.
2. **SQL:** SQL (Structured Query Language) is a specialized language used for managing and manipulating databases. It allows users to retrieve, store, and modify data within relational database management systems (RDBMS). With its powerful querying capabilities, SQL forms the backbone for interacting with and organizing structured data in various applications and industries.
3. **Libraries Used:**

The code starts by importing necessary Python libraries such as:

1. **Tkinter:** Tkinter library facilitates building the graphical user interface (GUI) for the application.
2. **ttk from Tkinter:** It offers themed widgets to enhance the visual aesthetics and functionality of the GUI elements.
3. **Sqlite3:** The Sqlite3 library allows the interaction with SQLite databases, enabling storage and retrieval of data.
4. **Cryptography:** This library provides the functions for secure password hashing, key derivation, and encryption operations.
5. **Base64:** It facilitates encoding and decoding data in base64 format, used in encoding encrypted passwords.
6. **OS:** OS library provides functionalities for interacting with the operating system, used for various system-related tasks.
7. **String:** Offers utilities for string operations, used here for generating a pool of characters for password generation.
8. **Secrets:** This library helps in generating cryptographically secure random numbers, utilized in creating strong and random passwords.
9. **Implementation:**
10. **Database Connection:** In this project we areusing SQlite Database. We need to install and initialize the database (In our case it is GarlapatiDB). Then we need to copy the path of the database and give it in the backend to connect with database, so that we can send and retrieve data from it. To check whether the data is being sent to database or not we can use sql query in command prompt.

The backend code to connect and send data to database:

A black screen with text on it

Description automatically generated

Here is the sql query used in cmd prompt to view and make changes to database:

A screenshot of a computer

Description automatically generated

1. **Backend (Python code):** Here we are making use of python code to develop and run Password Manager. And we are using VScode to run the code. Let’s discuss functions defined in the code.
2. Login function by entering the Master Password.

A screen shot of a computer code

Description automatically generated

1. Deriving 256 bit key from Master Password using PBKFD2 algorithm.

A computer screen shot of a program code

Description automatically generated

1. Encrypting the Passwords using the above key by making use of AES algorithm.

A screen shot of a computer code

Description automatically generated

1. Decrypting the encrypted Passwords using the 32-byte key by making use of AES algorithm.

A screen shot of a computer code

Description automatically generated

1. Generating strong and random passwords.

A black background with colorful text

Description automatically generated

1. Function to get the details from user using the add passwords function and sending them to database.

A screen shot of a computer program

Description automatically generated

1. Retrieving data from database when view password button is pressed.

A computer screen shot of text

Description automatically generated

1. Function to show and hide passwords.

A screen shot of a computer code

Description automatically generated

1. **Frontend:** There is a library named Tkinter in python which helps us to develop a GUI. We are importing this library as tk and we need to download ttk from it to develop the widgets in UI. Here is the code how we defined the main window. In the same way order windows are also developed.

A screen shot of a computer program

Description automatically generated

1. **Future Scope:** Since the necessity for strong and secure passwords is growing in importance in the current digital age, the future potential for a password management tool project is highly promising. Here are some potential directions for this project's future growth and development:
2. **Integration with many devices:** Given how many devices people use daily, a password management application must be usable on a variety of platforms, including desktop, mobile, and tablet.
3. **Biometric authentication:** To provide an additional layer of protection to password management software, biometric authentication methods such as voice recognition, fingerprint scanning, and face recognition may be included.