Week 2 Report

Report Date: 13-june-2023

Password manager using Python

By Sai Harika. P, Poojitha. V, Sai

1. Introduction:

In this report, I will provide an overview of the progress made during Week 3 of the development of a password manager using Python. The password manager aims to securely store and manage user passwords for various online accounts.

2. Accomplishments:

a. Database Integration: During Week 3, we successfully integrated a database system into the password manager project. This allows us to store and retrieve encrypted user passwords securely. We chose SQLite as the database engine due to its lightweight nature and ease of integration with Python.

b. User Interface Improvements: We made significant improvements to the user interface (UI) of the password manager application. We implemented a more intuitive and user-friendly design using the Tkinter library in Python. The UI now includes features such as login, registration, password entry, and retrieval.

c. Encryption and Decryption: To enhance security, we implemented encryption and decryption mechanisms for storing and retrieving passwords from the database. We used the cryptography library in Python to achieve this. The encryption algorithm ensures that passwords are securely stored, even if the database is compromised.

d. Testing and Bug Fixes: We conducted extensive testing during Week 3 to identify and fix any bugs or vulnerabilities. We performed unit tests on different components of the password manager, including user registration, login, password storage, and retrieval. We also addressed any identified issues promptly.

3. Challenges:

a. Database Schema Design: Designing an efficient and secure database schema was a challenging task. We needed to carefully plan the structure of the tables and relationships to ensure optimal performance while maintaining data integrity and security.

b. User Interface Design: Creating an intuitive and user-friendly interface required thorough consideration of user experience (UX) principles. We conducted several iterations to improve the UI and gather feedback from users to make it as user-friendly as possible.

4. Next Steps:

a. Password Generation: In the upcoming week, we plan to implement a password generation feature within the password manager. This feature will allow users to generate strong and unique passwords for their accounts automatically.

b. Browser Integration: We aim to integrate the password manager with popular web browsers to enable seamless autofill functionality. This will further enhance the user experience and convenience.

c. Multi-Factor Authentication: To strengthen security, we will implement multi-factor authentication (MFA) options in the password manager. Users will have the choice to enable additional authentication methods, such as OTP (one-time password) or biometric authentication.

d. Security Audit: We will conduct a comprehensive security audit of the password manager to identify and address any potential vulnerabilities. This will include code reviews, penetration testing, and threat modeling.

5. Conclusion:

Week 3 was productive in terms of database integration, user interface improvements, encryption, and testing. We are making steady progress towards developing a secure and user-friendly password manager. The upcoming week will focus on implementing password generation, browser integration, multi-factor authentication, and conducting a security audit to ensure the reliability and robustness of the application.. K, Rizwana. D, Revathi.M

**Project Overview:**

The password manager Python project aims to create a secure and user-friendly application for managing passwords. The password manager will allow users to store their passwords securely, generate strong passwords, and retrieve passwords when needed.

**Progress Made:**

During the second week of development, we the team of 5 have a significant progress was made in the following areas:

User Interface Design:

* Incorporated necessary input fields, buttons, and labels for password management functionalities. Developed a basic graphical user interface (GUI) using a Python GUI library such as Tkinter or PyQt.
* Implemented a clean and intuitive design to provide a user-friendly experience.

Database Integration:

* Set up a database management system (DBMS) to store and retrieve password records securely.
* Created the necessary database tables and columns to store user credentials.
* Implemented SQL queries to handle database operations, such as inserting, retrieving, and updating passwords.

Password Encryption and Hashing:

* Implemented a strong encryption mechanism to securely store user passwords in the database.
* Utilized cryptographic libraries, such as bcrypt or hashlib, to hash the passwords.
* Ensured that the passwords are securely stored and cannot be easily decrypted.

Password Generation:

* Developed a password generator function that can generate strong and random passwords.
* Allowed users to specify the length and complexity requirements for the generated passwords.
* Integrated the password generator into the user interface for easy accessibility.

Challenges Faced:

During the second week of development, the following challenges were encountered:

GUI Development Complexity:

* Designing an intuitive and aesthetically pleasing user interface required careful consideration of user experience (UX) principles.
* Handling user input validation and error handling for various fields presented a challenge in maintaining a smooth user flow.

Security Considerations:

* Ensuring the highest level of security for password storage demanded rigorous testing and implementation of encryption and hashing algorithms.
* Striking a balance between usability and security when generating strong passwords required thoughtful decision-making.

Goals for Next Week:

The upcoming week will focus on the following tasks:

Password Retrieval:

* Implement functionality to retrieve stored passwords for a selected account or website.
* Display the retrieved passwords securely, adhering to best security practices.

Account Management:

* Develop features to manage user accounts, such as creating new accounts, modifying existing accounts, and deleting accounts.
* Implement necessary input validation and error handling to ensure data integrity.

Testing and Bug Fixes:

* Conduct comprehensive testing to identify and fix any potential bugs or vulnerabilities.
* Gather feedback from testers to improve the user experience and address any issues.

Documentation:

* Create detailed documentation describing the application's features, installation instructions, and usage guidelines.
* Provide clear instructions on how to run the program, set up the database, and navigate the user interface.

**Conclusion:**

The second week of development for the password manager Python project was productive, with significant progress made in GUI design, database integration, password encryption, and password generation. Challenges were encountered but successfully addressed. Next week's focus will be on implementing password retrieval, account management functionalities, conducting testing, and finalizing the documentation. The project is on track, and efforts will be made to ensure a robust and secure password manager for users.

By Sai Harika.P

Mail : potturusaiharika19@gmail.com