Password Manager with Python

Created by: Mario Getaw, Jarrett Wilson, & Joshua Moore

May 8th, 2025

COMP-460

Senior Capstone 2025

Computer Science Department

Dr. Sunday Ngwobia

Abstract

This project presents the design and development of a cross-platform password manager application created using Python. The goal was to implement core security features such as local encryption, user authentication, and secure password storage while maintaining simplicity and accessibility. The application includes a graphical user interface (GUI) and integrates two-factor authentication (2FA) to enhance account security. Notably, instead of relying on a dedicated backend or database, the app utilizes Google Drive for cloud-based password backup and recovery, allowing users to retrieve credentials across devices. By combining cryptographic techniques with a cloud storage API, the system demonstrates how secure password management can be achieved within a lightweight desktop application. This project serves as a proof of concept that explores user-focused security features and cross-device accessibility, while acknowledging the limitations of a prototype environment in contrast to industry-grade password managers. The final product delivers a working model that balances security, usability, and practical design within the scope of an undergraduate capstone.

Introduction

In the digital age, individuals rely on a growing number of online accounts, each requiring secure and unique login credentials. With the rise in data breaches and cyber threats, managing these credentials has become both a necessity and a challenge. Password managers have emerged as a practical solution, offering users a secure and centralized way to store, retrieve, and manage passwords. This capstone project explores the development of a lightweight, standalone password manager built entirely in Python.

The goal of the project was to create an application that supports essential password management functionality—such as secure storage, encryption, user authentication, and cloud-based backup—while remaining accessible to non-technical users. To enhance security and user trust, the application implements local encryption using the Fernet module from the cryptography library and integrates two-factor authentication (2FA) via time-based one-time passwords (TOTP). For remote data access and backup, the system uniquely leverages Google Drive instead of a conventional database, enabling cross-device synchronization without requiring complex backend infrastructure.

By focusing on core security principles and usability, this project offers a practical example of how password management solutions can be built from the ground up. It is intended as a learning tool and prototype, emphasizing the feasibility of developing secure and functional applications with limited resources and within an academic environment.

Literature Review

The need for secure password management has become more critical with the increasing volume of digital identities users must maintain. Over the past decade, a variety of password manager architectures have been developed, analyzed, and critiqued to address these growing challenges. This section examines foundational studies and advancements that have shaped the current understanding of the security of password managers.

Li et al. (2014), in their landmark study “The Emperor’s New Password Manager,” provided a comprehensive security analysis of five web-based password managers and uncovered serious vulnerabilities, including credential leakage and logic flaws stemming from poor implementation of browser extensions and bookmarklets. The study emphasized the need for a defense-in-depth approach and highlighted the risks of centralized cloud-based platforms, which can become single points of failure if not designed securely.

Complementing this, Luevanos et al. (2017) conducted an in-depth analysis of three open-source password managers - Passbolt, Padlock, and Encryptr - focusing on the implementation of cryptographic techniques and user interface vulnerabilities. Their findings underscored the importance of transparent development practices and revealed that even popular tools can suffer from predictable flaws such as weak randomness and susceptibility to clipboard or phishing attacks.

The literature also reflects a growing interest in hybrid and collaborative approaches. A 2023 IEEE paper introduced an “Enhanced Password Manager using Hybrid Approach,” which combined symmetric encryption (AES) with SHA-256 hashing to balance performance and security. This method aligns with widely accepted cryptographic standards and supports lightweight, browser-based implementations. The emphasis on encryption flexibility is particularly relevant for applications intended to operate offline or without backend dependencies.

Further, Amarnadhreddy et al. (2024) explored secure password sharing in collaborative environments, introducing blockchain and role-based access control in their proposed models. Their work identified a critical shortcoming in traditional password managers: the lack of secure, granular permission controls when credentials must be shared among multiple users. Their review also outlined industry trends, such as the use of biometric authentication and the integration of enterprise systems, as directions for future innovation.

Together, these studies provide a robust framework for understanding both the potential and the pitfalls of password manager design. The recurring themes - centralized vs. distributed architecture, encryption and authentication standards, UI vulnerabilities, and collaboration features - guided the design and security decisions of this capstone project. By drawing from both academic critiques and technical implementations, the development of this application aimed to synthesize best practices while avoiding common architectural flaws.

System/Hardware Requirements

The password manager has minimal requirements, as it works on any computer that can install the Python library. To use the Cloud feature, you first need to set up a Google account with Google Drive. When storing a password, it will direct you to Google, where you can select your account or create one if you don't already have one.

Software Requirements

The software must allow users to create an account to save passwords and to log in later. Along with this functionality, the software must be compatible with a 2 factor authentication app to allow for added protection. The software must be able to encrypt the user's data to allow for a level of protection for the passwords. The system must also have a user-friendly GUI. Furthermore, the application should work quickly and smoothly to allow for a seamless user experience.

System/Hardware Requirements

The password manager requires the system to be able to install Python libraries to run locally on the device. To use the Cloud feature, you first need to set up a Google account with Google Drive. When storing a password, it will direct you to Google, where you can select your account or create one if you don't already have one. Therefore, for cloud services, an internet connection is also required.

User Requirements

The users of the program would require a simple GUI, so that they can feel confident navigating the app. They would also require the ability to edit and retrieve existing passwords, as well as generate new secure passwords on the app. Users would also expect some level of encryption for their stored passwords and login information, so they can feel confident their data will not be breached. Finally, users would expect there to be a cloud backup of their data in case they wanted to log in across devices, or if their local data was corrupted or deleted for whatever reason.

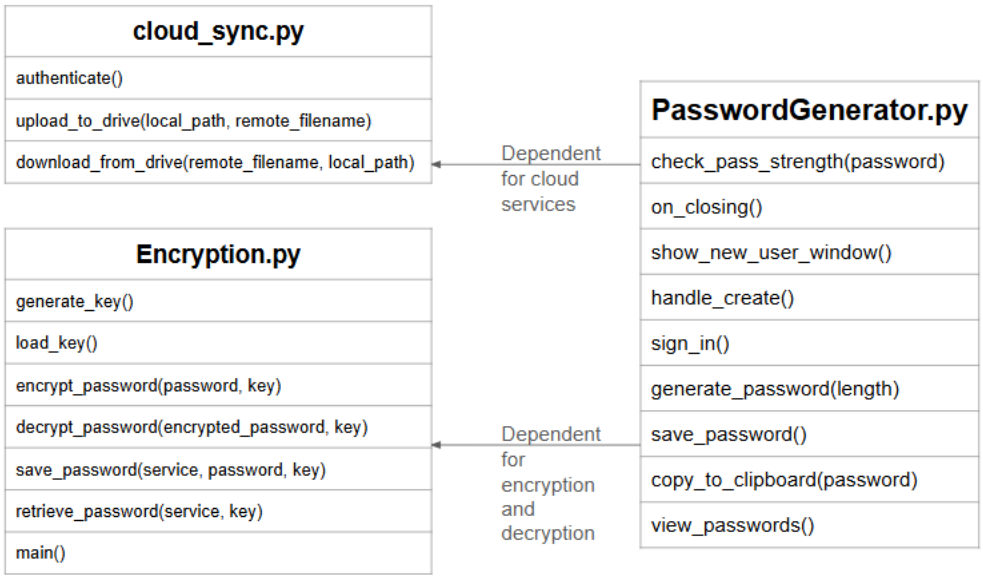
Project Performance, Uniqueness, and Similarity to Others

Our app performs as intended. A simple password manager app was created as a proof of concept to demonstrate the fundamentals of secure credential storage and user authentication. The primary goal was to explore encryption techniques and user interface design in a controlled development environment. It served as a learning tool or prototype rather than a fully deployed product, lacking the rigorous security auditing, compliance certifications, and scalability features necessary for real-world use. Because it was not intended to handle sensitive user data outside

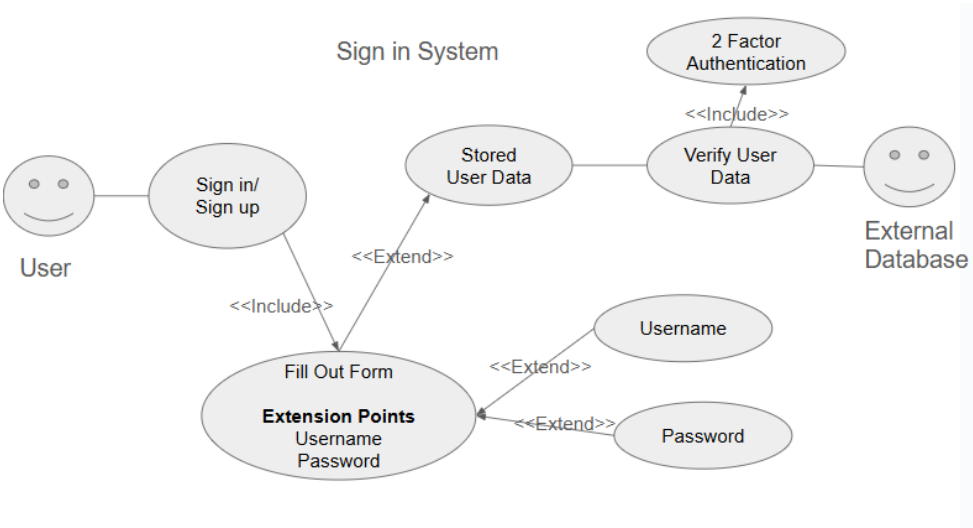
of testing scenarios, it remained a conceptual project without exposure to live users or production environments. Our application takes a unique approach by utilizing Google Drive for password storage instead of a traditional database. This decision was driven by the goal of simplifying implementation while ensuring cross-device accessibility. By leveraging Google Drive's cloud infrastructure, we enabled users to securely access their stored passwords from any device with an internet connection, streamlining the user experience without the complexity of managing a custom backend database. This method also allowed for easier integration within the scope of our project, aligning with our focus on accessibility and ease of use.

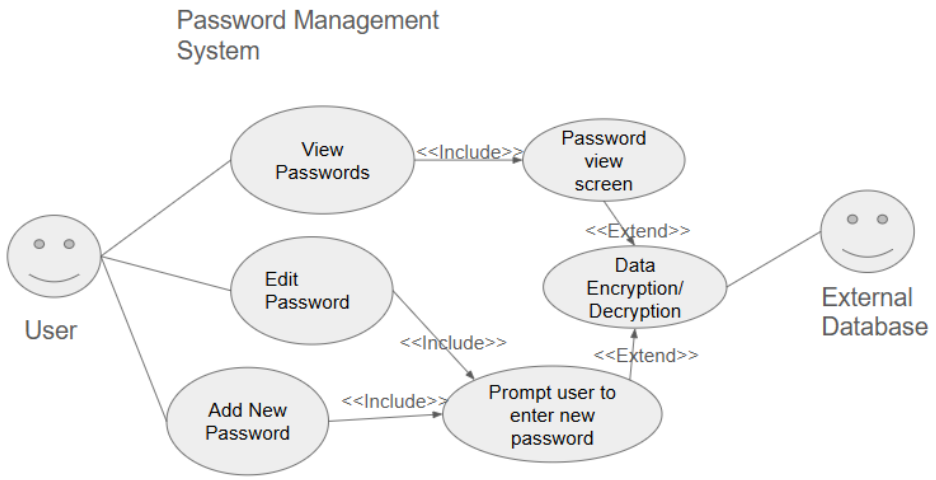
Before beginning development, we conducted research on leading password managers such as NordPass and 1Password to gain insight into industry standards and functionality. Through this comparison, we identified key features that make these platforms robust and user-friendly, including cloud-based database systems and seamless browser integration. However, given the scope of our project and the limited time and resources available, implementing such advanced capabilities was not feasible. Instead, we focused on achievable goals, such as designing a clean, intuitive user interface and enabling cross-device accessibility. While our app does not yet include the full range of enterprise-level security features, it reflects a thoughtful balance between functionality and practicality within the constraints of our development environment.

Class Diagram –

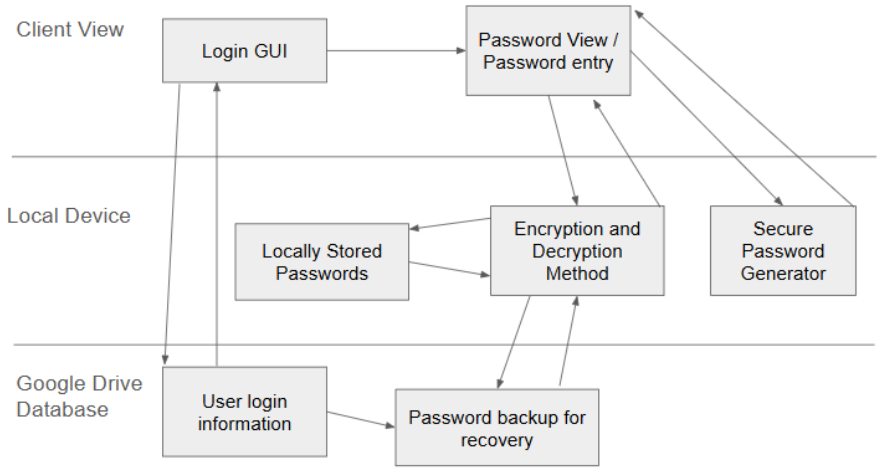


Use Case Diagrams –





Architecture –



Ethical/ Legal and Security Concerns

While our app adheres to ethical and legal standards, it does present notable security limitations. It was not developed to provide enterprise-level or high-security protection. Although passwords are stored using basic encryption methods, the implementation lacks the robustness needed to prevent decryption by individuals with advanced knowledge of password-cracking techniques. As such, we do not recommend relying on this application for safeguarding sensitive credentials. Instead, it is intended as a simplified demonstration of how a password manager functions, serving educational or conceptual purposes rather than secure, real-world usage.

Project Performance, Uniqueness, and Similarity to Others

Our app performs as intended. A simple password manager app was created as a proof of concept to demonstrate the fundamentals of secure credential storage and user authentication. The primary goal was to explore encryption techniques and user interface design in a controlled development environment. It served as a learning tool or prototype rather than a fully deployed product, lacking the rigorous security auditing, compliance certifications, and scalability features necessary for real-world use. Because it was not intended to handle sensitive user data outside of testing scenarios, it remained a conceptual project without exposure to live users or production environments. Our application takes a unique approach by utilizing Google Drive for password storage instead of a traditional database. This decision was driven by the goal of simplifying implementation while ensuring cross-device accessibility. By leveraging Google Drive's cloud infrastructure, we enabled users to securely access their stored passwords from any device with an internet connection, streamlining the user experience without the complexity of managing a custom backend database. This method also allowed for easier integration within the scope of our project, aligning with our focus on accessibility and ease of use.

Before beginning development, we conducted research on leading password managers such as NordPass and 1Password to gain insight into industry standards and functionality. Through this comparison, we identified key features that make these platforms robust and user-friendly, including cloud-based database systems and seamless browser integration. However, given the scope of our project and the limited time and resources available, implementing such advanced capabilities was not feasible. Instead, we focused on achievable goals, such as designing a clean, intuitive user interface and enabling cross-device accessibility. While our app does not yet include the full range of enterprise-level security features, it reflects a thoughtful balance between functionality and practicality within the constraints of our development environment.

Conclusion

his project successfully demonstrates the core principles behind secure password management through the development of a lightweight, Python-based desktop application. By combining symmetric encryption, two-factor authentication, and cloud-based file synchronization via

Google Drive, the system offers a streamlined user experience without relying on a dedicated backend or third-party database. While the application is not intended for enterprise or production use, it functions effectively as a proof of concept and educational tool.
The development process highlighted critical considerations in user security, interface design, and cryptographic implementation. It also emphasized the importance of balancing functionality and usability, especially when resources are limited. By incorporating elements inspired by commercial password managers and research from academic literature, the final product reflects a thoughtful approach to secure credential storage. Future iterations could benefit from enhanced encryption protocols, full security auditing, and broader platform compatibility. Nonetheless, this project affirms that fundamental security features can be implemented with clarity and purpose within a limited scope.

Individual Contributions

- Mario Getaw: Source Control, Proof of Concept, Video Demo/Deliverables
- Jarrett Wilson: Architecture, Literature Review, Requirements
- Joshua Moore: Ethical & Security Concerns, Requirements, User Stories
- Everyone: Design

References

1. Amarnadhreddy, P. S., UmaShankar, A., & Kumar, A. V. (2024). A Review on Secure Password Manager. International Journal of Innovative Research in Science, Engineering and Technology, 13(5), 6734–6736.

2. Luevanos, C., Elizarraras, J., Hirschi, K., & Yeh, J. (2017). Analysis on the Security and Use of Password Managers. In Proceedings of the 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), IEEE.

3. Pandare, P., Mali, S., Uniyal, S., Rumao, P., & Vani, R. (2023). Enhanced Password Manager using Hybrid Approach. Proceedings of the International Conference on Inventive Computation Technologies (ICICT), IEEE.

4. Li, Z., He, W., Akhawe, D., & Song, D. (2014). The Emperor’s New Password Manager: Security Analysis of Web-based Password Managers. In USENIX Security Symposium.

5. Florêncio, D., & Herley, C. (2007). A Large-Scale Study of Web Password Habits. In Proceedings of the 16th International Conference on World Wide Web (WWW).

6. Whitten, A., & Tygar, J. D. (1999). Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In USENIX Security Symposium.