Criterion A: Planning

Scenario

In today's digital world, the security of our online accounts has become increasingly crucial. Sadly, my mother discovered this the hard way when her email password was compromised, resulting in a chain reaction of other stolen accounts. We both realized that using the same weak password for numerous accounts had been a security nightmare waiting to happen.

When we evaluated the damage and discussed possible solutions, we concluded that her conventional approach of manually setting and memorizing unique and strong passwords for each account was impractical. This method would either result in weak passwords or information overload.

At that point, I proposed utilizing a password manager. However, my past research revealed that most existing password managers lacked offline functionality and simplicity for a non-tech-savvy person. Moreover, the recent hacking of LastPass, a popular password manager, made me even warier about entrusting her sensitive information to third-party software, especially web-based applications.

My mother and I agreed on the primary goal of alleviating the stress of password management and preventing any future hacks. For this reason, I took matters into my own hands and suggested creating a custom password manager to be stored locally on a USB drive. It would generate unique and strong passwords for each account, store them in an encrypted database, and allow easy retrieval and input. Following some additional discussion, as evidenced in Appendix I, my mother took me up on my offer and emphasized the urgency of having a solution. I later consulted my IB Computer Science teacher, who thought it was a good idea and agreed to be my advisor.

Rationale

The inspiration and urgency of this project arose from a security incident that my mother experienced when her weak and reused password was compromised. We recognized the importance of finding a solution that could prevent similar attacks and alleviate the stress of password management.

After evaluating various programming languages, I decided to use Java:

1. I had prior experience with Java from my IB Computer Science class, which would make development easier, quicker, and more efficient.
2. Java's extensive documentation and support from a large user base, including encryption libraries like AES, made it an excellent choice for ensuring the password manager's security.

3. It could use the NetBeans IDE to build my GUI.
4. Java's platform independence allowed the password manager to run on different devices without compatibility issues.
5. We specifically wanted a password manager that fit on a small USB drive, and Java allowed me to create a runnable application that fulfilled this requirement.

My research on existing password managers revealed that they were either too expensive, too complicated, or untrustworthy, given recent data breaches. Therefore, I set out to design an intuitive yet secure password manager, providing my mother with a practical and effective tool for managing her online accounts. Developing a custom password manager allowed me to tailor the solution to my mother's specific needs, such as simplicity, easy-to-use GUI, and secure storage.

Success Criteria

Agreed to by the client in Appendix I.

I. A master password is required to access the main menu; the user can create one if one does not exist.
II. There are limited login attempts with an incorrect master password.
III. An account list, retrieved from an XML file and displayed in the main menu, is sorted in descending order, with the most-used accounts at the top.
IV. Users can add, remove, and filter/search items from the account list.
V. When adding an account, users can generate strong, randomized passwords that meet standard password complexity requirements.
VI. Users can select a renewal period for a password when adding an account. The program will display a message when a password is about to expire.
VII. The data is secure and encrypted with AES-256 using a password-based encryption key.
VIII. The client, my mother, considers the program easy to use.

Word Count: 396