**Senior Project Defense**

**Xecure Password Manager**

Student Name: Carlos R. Ocasio Rodriguez

Degree and Major: Bachelor of Science in Cybersecurity

Project Advisor Name: Prof. Henderson

Date: March 27, 2022

Expected Graduation Date: May 2022

**Problem Statement**

The management of passwords is one of the most important aspects of Cybersecurity and it is imperative that users implement and practice safe password management practices. In order to ensure safe password management practices users must create strong and unique passwords for every one of their accounts. Unfortunately, most users have far too many online accounts to remember a unique password for each account. A survey conducted by Nate Lord (2020) indicates that the average user has over 90 different online accounts. There is a limit to how many passwords a user can remember, and this survey presents a problem that can lead to users reusing the same passwords. A Google Online Security Survey (2019) found that 52 percent of users reuse the same password for multiple accounts. The practice of reusing passwords leaves users vulnerable to intrusion attacks, which can result in data loss, monetary loss and more. A report conducted by IBM (2020) found that the average cost of a data breach is $3.68 million, while a data breach in the healthcare industry is estimated to cost around $7.13 million. Furthermore, IBM found that data breaches in the United States are among the costliest averaging around $8.64 million.

By using a password manager users can easily manage unique and strong passwords for their accounts while eliminating all the difficulties that are associated with it. With a password manager users can store all their passwords in a secured vault that can be accessed with a “master” password. This eliminates the need to remember multiple passwords and greatly reduces the chances of having multiple accounts compromised. The management of passwords is one of the most important and overlooked factors of information security. In order to eliminate the difficulties that come with remembering passwords it is recommended that users utilize a password management application.

**Project Description**

The project is a password manager application for Windows computers used to securely store and manage log in credentials. It uses a “master” password to provide access into the user’s account with all the records and credentials. The application also has connectivity to a MySQL database that is used to securely store the user’s information. In order to protect the information in the database the application uses the well-established AES 256 encryption standard and the SHA 256 hashing algorithm. Additionally, the users have the option to generate unique and complex passwords for every record entered. Users will also have the option of entering their own passwords and the application will guide them every step of the way. The application can also copy a record’s password into the computer’s clipboard. This password will then be wiped from the clipboard after fifteen seconds or if the user closes the application. Lastly, the program has features that allows users to reset their “master” password and email if they forget it or wish to change it.

The application is called Xecure Password Manager and it was inspired by applications like KeePass, and LastPass and it contains all the major features that these application have. All of the previously mentioned features are the standard in the password manager market, and they will be included in the Xecure Password Manager application at no cost to the users. In addition to being a free application, Xecure Password Manager will not contain any ads, there are no password or account limits, and it will not store or use user data for any other purpose. This is what separates Xecure Password Manager from the competition.

**Personal Motivation**

The practice of securing networks, systems, and other digital infrastructures are an essential aspect of information systems and cybersecurity. As a Cybersecurity major, I strive to broaden these previously mentioned technical and functional skills. Through the development of this application, I was hoping to get a better understanding of the various hashing and encryption practices that are common in the field of cybersecurity. Another skill I wanted to learn was database management. I have made databases in the past, but I never got the opportunity to use or manage my own databases. I also wanted to strengthen my problem solving and debugging skills that are so vital as a programmer. Lastly, I wanted to learn how create and prevent SQL injections.

This project required me to learn and understand the various methods that attackers use to exploit passwords and databases. Through this project I was able to expand my computer analysis skills, and it gave me additional resources that can be used to thwart malicious attacks. I believe that after completing this project I have become a better cybersecurity professional.

**Research & Background**

In order to create this application, extensive research was done into the password manager industry. Xecure was inspired by applications like KeePass and LastPass and it is meant to be an alternative to these applications. In order to do this, it must offer the same features as the most popular password manager applications in the industry. After conducting research, the current and more popular password managers in the market were found to be:

* KeePass
* LastPass
* 1Password
* DashLane
* NordPass

All of these applications are very similar and contain the following features that I see as an industry standard:

* Encryption of passwords
* Hashing of credentials
* Save password to clipboard
* Wipe passwords from clipboard
* Generate strong and unique passwords
* Support multiple users
* Store multiple records
* Email authentication

The previously mentioned features will all be part of the Xecure Password Manager application and will be offered to the users at no charge. Additionally, other features will be added to separate the application from the competition.

**Proposed Implementation Languages**

The password manager application was developed on a Windows 10 computer by using the latest versions of Python 3 (version 3.9.7) and MySQL Server (version 8).

**Libraries, packages, development kits**

The password manager application uses the following libraries:

* hashlib – Used for hashing.
* cryptography.fernet – Used to encrypt.
* msvcrt – Used to prevent the command screen from continuing until a key is pressed.
* mysql.connector – Used to connect to the database.
* os – Used to change the title of the command line screen.
* sys – Used alongside getpass to hide passwords in the command line screen.
* getpass – Used to hide password in the command line screen.
* random – Used to generate random passwords and codes.
* smtplib – Used for emailing capabilities.
* socket – Used alongside smtplib to send emails, and email error handling.
* time – Used alongside pyperclip to determine when to wipe clipboard.
* pyperclip – Used to save record’s password into the computer’s clipboard.
* threading – Used to wipe the clipboard while the user continues to use the application.

**Additional Software & Equipment Needed**

A Windows operating system is required, as well as Python (version 3.9.7) and a MySQL (version 8) database. See below for instructions on how to download and setup both Python and MySQL.

**MySQL Installation Guide:**

These steps will guide you through the installation and setup of the MySQL Community Server for the required database of this application.

1. Visit <https://dev.mysql.com/downloads/mysql/>
2. Look for the “MySQL (MSI) Installer for Windows” and click on the “Go to Download Page” button (You can also use this link: <https://dev.mysql.com/downloads/windows/installer/8.0.html>)
3. There will be two Windows MSI Installer options here. Please find the “(mysql-installer-community-8.0.27.0.msi)” option and press the download button.
4. You will be redirected to a screen that asks you to login or sign up, but this is not needed. Scroll down to the bottom of the page and click on the “[No thanks, just start my download.](https://dev.mysql.com/get/Downloads/MySQLInstaller/mysql-installer-community-8.0.27.0.msi)” link.
5. A download prompt will be displayed. Click on the “Save File” button. This will start the download of the file.
6. Wait for the download to finish.
7. Once it has downloaded, look for the “mysql-installer-community-8.0.26.0” (version number might change) file and run it. The file is typically located in your downloads folder.
8. After double clicking the file, it will begin preparing the installation and it will then ask you for permission to install the application. Click the “yes” button.
9. It may ask you “Do you want to allow this app to make changes to your pc?”. Click on the “yes” button. (Skip this step if you did not see this alert)
10. The MySQL installation windows will now be displayed, and it will ask you to read over the License Agreement. After doing so, press the “I accept the license terms” check box and press the “Next” button.
11. On the “Choosing a Setup Type” screen select the “Developer Default” option and press next.
12. A screen may alert you that some product requirements have not been satisfied and it will ask you if you wish to continue. Select yes. (Skip this step if you did not see this alert)
13. On this installation screen you can see all of the products that will be installed. When you are ready select the “Execute” button. This will begin installing all of the products.
14. Wait for the Installation to finish.
15. After the download has finished you will see a green checkmark on all of the products downloaded. If everything downloaded successfully press the “Next” button.
16. On the “Product Configuration” screen you will be able to see the configuration for all of your downloaded products. Once you have reviewed this information press the “Next” button.
17. On the “Type and Networking” screen select the “Standalone MySQL Server/Classic MySQL Replication” option and press the “Next” button.
18. On this screen leave the “Config Type” as “Development Machine”. Make sure that the “TCP/IP” checkbox and “Open Firewall port for network access” checkbox are selected. Lastly, the “Port Number” is 3306. After confirming these options press the “Next” button.
19. On the Accounts and Roles screen enter a strong password on the “MySQL Root Password” fields (**make sure you remember this password**). Then press the “Add User” button to create a user account.
20. A screen will appear and ask you for username information. Enter a username for your account (**make sure you remember this username**). Then enter a strong password for your account (**make sure you remember this password**). Once this is done press the “Ok” button. After creating your user account press on the “Next” button.
21. On the “Windows Service” screen leave all settings as default and press on the “Next” button.
22. On the “Plugins and Extensions” screen leave all settings as default and press on the “Next” button.
23. On the “Apply Configuration” screen you will see an overview of the configurations. Once you are ready press the “Execute” button and the listed changes will be applied. Once this is done press the “Finish” button.
24. On the “Product Configuration” Screen you will see that the MySQL Server is completed. Press the “Next” button to continue.
25. On the “MySQL Router Configuration” screen you will leave all settings as default and press on the “Next” button.
26. On the “Product Configuration” screen you will review the configuration and press on the “Next” button.
27. On the “Connect To Server” screen, select the “MySQL Server 8.0.27.0 (version may be different for you) text. Then enter press the “Check” button to test the connection. Once the connection is established and you see a “Connection Successful” message press the “Next” button.
28. On the apply configurations, press the “Execute” button. Once all the configurations are finished press the “Finish” button.
29. On the “Product Configuration” screen you will review the configuration and press on the “Next” button.
30. You will see the installation complete screen. Press the “Finish” button.
31. You have successfully installed the MySQL database.

**To use with MySQL Workbench:**

1. Go to start and search for “workbench”
2. Open the MySQL Workbench
3. Once the application opens, double click the “Local instance” below the “MySQL Connections”.
4. A screen will appear, and you will be prompted to enter the password for the database. After entering the password, press the “OK” button.
5. This is your database serve.

**Requirements**

Requirement ID Number: 1 Requirement Type: 1, 2 (Functional, Look and Feel)

Description: The application will have a command-line-based user interface.

Rationale: An interface will be needed so that users can interact with the application in an efficient and simple manner. A broader audience will be able to use the application if a command-line-based user interface is implemented.  
Fit Criterion: Users must be able to use all features and accomplish all tasks by using the command-line-based interface.

Priority: Very High  
Dependencies: None

Requirement ID Number: 2 Requirement Type: 1, 6 (Functional, Security)

Description: A login section must be implemented.

Rationale: The login section is needed to authenticate users and grants them access to their data.   
Fit Criterion: When presented with valid credentials the application must authenticate users and grant them access to their data. If presented with invalid credentials, the application will display an error message and will not grant them access to the data. Users will not be able to get past the login screen without valid credentials.

Priority: Very High  
Dependencies: None

Requirement ID Number: 3 Requirement Type: 1, 6 (Functional, Security)

Description: The application must have connectivity to a MySQL database.

Rationale: The database provides storage and security features.

Fit Criterion: The application must be able to save data to the database. The data will be inaccessible if there is no database connectivity. Additionally, it will not allow the user to create a record if the database cannot be reached.

Priority: Very High  
Dependencies: None

Requirement ID Number: 4 Requirement Type: 6 (Security)

Description: The application must store user records in the MySQL database.

Rationale: The database provides storage and security features. It also provides features against data redundancy and reduces inconsistencies. All these features are important for managing log-in credentials.

Fit Criterion: The application must be able to save data to the database. This data must then be accessible to authorized users only. An error message will alert the user if the data was not saved to the database.

Priority: Very High  
Dependencies: The application must have connectivity to the database for this requirement to be fulfilled.

Requirement ID Number: 5 Requirement Type: 1, 6 (Functional, Security)

Description: A “Register” section must be implemented.

Rationale: Users need to be able to create an account to use the application. Therefore, a “Register” section must be implemented.

Fit Criterion: The “Register” section must allow users to create a master username and password. If correctly done the username and password will be stored in the database. These credentials can then be used to log in to the application. Users cannot use the application without creating an account.

Priority: Very High  
Dependencies: A “Login” section and the database must be implemented before this requirement can be fulfilled.

Requirement ID Number: 6 Requirement Type: 6 (Security)

Description: Application will lock-out an account after six failed log-in attempts and the password will be reset.

Rationale: Unauthorized users may attempt to guess the login credentials of a user. To prevent these brute force attacks a lock-out feature will be implemented.

Fit Criterion: After six failed log-in attempts the application will place a lock on the account and the user will not be able to log in. The lock-out feature will cause the application to force close.

Priority: Very High  
Dependencies: The “Login” section and the “Register” section must be implemented before this requirement can be fulfilled.

Requirement ID Number: 7 Requirement Type: 6 (Security)

Description: All data stored and used by the application must only be accessible to authorized users.

Rationale: The data stored and used by the application must only be visible to authorized users. To prevent a data breach, registered users will only have access to their data. Users that are not registered will not be able to use the application and will not have access to any data.

Fit Criterion: Registered users must only be able to see their data, while unregistered users will not be able to use the application.

Priority: Very High  
Dependencies: The “Login”, “Register”, and database sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 8 Requirement Type: 6 (Security)

Description: A SHA256 hash will be used as a log-in authentication method.

Rationale: In order to improve security, the username and password along with a hashing algorithm will be used to authenticate the user.

Fit Criterion: The SHA256 will be used to hash the username and password provided. If the username and password provided match the login credential stored in the database, the user will be authenticated, and access will be granted. If the hash does not match the login credential stored in the database access to the application will be denied.

Priority: Very High  
Dependencies: The “Login”, “Register”, and database sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 9 Requirement Type: 3, 6 (Usability, Security)

Description: The application must be able to generate random passwords.

Rationale: Users may need help creating secure passwords. The application will provide them with the option of generating a strong random password for their desired account.

Fit Criterion: A random password must be generated and displayed to the user after opting for this feature. An error will alert the user if the random password was not generated.

Priority: Medium  
Dependencies: None

Requirement ID Number: 10 Requirement Type: 3, 6 (Usability, Security)

Description: The application will allow users to create their own password.

Rationale: Users may not like having randomly generated passwords. They may want to use their own passwords.

Fit Criterion: Users must be able to enter their own passwords. A confirmation message will alert the user that their password was accepted and saved.

Priority: High  
Dependencies: The “Login”, “Register”, and database sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 11 Requirement Type: 3, 6 (Usability, Security)

Description: The rules for the record passwords cannot be strict

Rationale: The application cannot dictate what users have chosen in the past for their passwords. It must accept all kinds of passwords no matter how weak or strong.

Fit Criterion: Any password less than thirty-five characters long will be accepted as a record password, but all other password combinations are valid. The user will be notified if the password was saved successfully.

Priority: High  
Dependencies: The “Login”, and “Register” sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 12 Requirement Type: 3 (Usability)

Description: Simple controls must be implemented on the front end to send SQL queries to the database.

Rationale: Users do not want to create SQL queries for every task while using the application.

Fit Criterion: Queries will be created in the back end and the users must be able to utilize them on the front end. Users must be able to view, add, edit, and delete records on the database from the front end. A confirmation message will be displayed after a successful query.

Priority: High  
Dependencies: The “Login”, and “Register” sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 13 Requirement Type: 3 (Usability)

Description: Data stored in the database will be encrypted by using AES 256 encryption.

Rationale: Encrypting the data prevents unauthorized users from using it. Encryption increases the security of the application and protects the data.

Fit Criterion: No plaintext must be visible in the database. If plaint text is found in the database this requirement has not been fulfilled. The information in the database must be decrypted before it can be used. If an encryption attempt fails an error message will be displayed to the screen.

Priority: Very High  
Dependencies: The database must be implemented before this requirement can be fulfilled.

Requirement ID Number: 14 Requirement Type: 3, 6 (Usability, Security)

Description: Registered users must be able to retrieve their unencrypted data from the database.

Rationale: Users need to be able to see their saved unencrypted login credentials.

Fit Criterion: Users must be able to select a specific account to retrieve their unencrypted password. If the data retrieved is encrypted this requirement has not been fulfilled. If the data cannot be retrieved or if it cannot be unencrypted an error message will be displayed to the user.

Priority: High  
Dependencies: The database and the encryption algorithm must be implemented before this requirement can be fulfilled.

Requirement ID Number: 15 Requirement Type: 3 (Usability)

Description: Passwords retrieved from the database must be copied to the user’s clipboard.

Rationale: This is done so that the user can immediately use the password. After retrieving the password users can paste it into their desired field.

Fit Criterion: After retrieving the data the user must be able to use the “paste” command and the password will be “pasted” into their desired field. An error message will be displayed if the password was not placed in the clipboard. On the other hand, a confirmation message will be displayed if the password was placed into the user’s clipboard.

Priority: Low  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 16 Requirement Type: 3 (Usability)

Description: The application must guide and direct users on how to accomplish all tasks.

Rationale: Users may not understand how to use the application. Clear and concise instructions will help the user navigate the application.

Fit Criterion: Prompts and alerts will guide the user on how to accomplish their task. The success of the directions will be based on task completion time. The average task completion time should be under five minutes.

Priority: Medium  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 17 Requirement Type: 3 (Usability)

Description: The users must be able to go back to the previous menu at any point of the process.

Rational: The users may decide they wants to cancel their current action.

Fit Criterion: A back option must be present in every part of the application. Once the option has been selected the user will be taken back to the previous screen.

Priority: Medium  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 18 Requirement Type: 6 (Security)

Description: The users must be able to log out of the application.

Rational: The users need to be able to log out of the application in a secure manner.

Fit Criterion: Once the option has been selected an alert message will prompt the user to confirm the action. The user will be logged out of the application if they confirm the action.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 19 Requirement Type: 3 (Usability)

Description: The users must be able to create, view, edit, and delete records from their account.

Rational: Users need to have full control over their data. They may want to create new records or delete old ones.

Fit Criterion: A confirmation message will be displayed if the action selected was completed successfully. On the other hand, if one of these actions cannot be completed an error message will be displayed to the user.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 20 Requirement Type: 6 (Security)

Description: Passwords that are considered too weak will not be accepted as the master password.

Rational: Short passwords or passwords that are considered too weak will not be accepted as the master password as this is a security risk.

Fit Criterion: A message will alert the user if their password is too weak. They will also be unable to save their password and will be asked to come up with a new one. Once a strong enough password has been entered a confirmation message will be displayed to the screen.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 21 Requirement Type: 6 (Security)

Description: Input validation must be performed throughout every section of the application.

Rational: Users may enter invalid data accidentally or purposefully and thus every input must be properly validated and sanitized to prevent a security risk. All prompts will be validated and will include exceptions to handle all scenarios.

Fit Criterion: All invalid inputs will produce an error message that will be displayed to the screen.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 22 Requirement Type: 3 (Usability)

Description: Error messages must provide guidance on how to complete the task.

Rational: Users may not understand why they can’t proceed and may need help fixing their mistakes.

Fit Criterion: The error messages must provide clear and concise instruction on how to complete the current task. The success of the directions will be based on task completion time. The average task completion time should be under five minutes.

Priority: Low  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 23 Requirement Type: 6 (Security)

Description: The application must have measures in place to prevent SQL injection attacks.

Rational: Unauthorized users may attempt to use SQL injections to retrieve information from the database.

Fit Criterion: Inserting SQL injections will result in an invalid input error.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 24 Requirement Type: 6 (Security)

Description: The average user should be able to retrieve a password in under three minutes.

Rational: Users typically won’t use an application that takes too long to complete a task.

Fit Criterion: Users must be able to retrieve a password in under three minutes.

Priority: Medium  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 25 Requirement Type: 6 (Security)

Description: The application must have emailing capabilities in order to reset passwords.

Rational: The application may need to send emails to the users. To reset passwords, the application will require emailing capabilities.

Fit Criterion: Application must be able to send emails to the users. Once the email has been sent a confirmation message will be displayed to the screen.

Priority: Medium  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 26 Requirement Type: 6 (Security)

Description: A verification code must be sent to the users email in order to verify their account.

Rational: Users may forget their password or may need to change their email. Before doing so, users must verify their account by entering the code sent to their email.

Fit Criterion: Users must enter the correct code before they can change their password or email. If the entered code does not match the user will be given an error message. If the code matches, they will be able to change their password or email.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, emailing and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 27 Requirement Type: 6 (Security)

Description: Errors in the middle of data entry will display an error and no data will be saved to the database.

Rational: This requirement will prevent the application from saving invalid or incomplete data into the database. The error will inform the user what went wrong and how to fix the problem.

Fit Criterion: An error message will be displayed if something goes wrong in the middle of data input. The user will have to acknowledge the error before moving on.

Priority: High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 28 Requirement Type: 6 (Security)

Description: Clipboard will be cleared after fifteen seconds.

Rational: The clipboard can be exploited to retrieve sensitive data. Thus, the clipboard must be wiped after a certain period of time.

Fit Criterion: After fifteen seconds the user will not be able to paste their password. Trying to paste a password after 15 seconds will result in an empty character.

Priority: Very High  
Dependencies: The “Login”, “Register”, clipboard, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 29 Requirement Type: 6 (Security, Usability)

Description: Users must be able to retrieve their username

Rational: Users may forget their username.

Fit Criterion: Users should be able to retrieve their username by verifying their email account. After this verification an email will be sent to the user with their registered username.

Priority: Very High  
Dependencies: The “Login”, “Register”, clipboard, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 30 Requirement Type: 6 (Usability)

Description: Users must be able to display only the name of their records.

Rational: Users may not want to display all the information of their records, and thus they should be able to retrieve only the record names. This name can later be used to retrieve the specific record’s information.

Fit Criterion: Users should be able to see all their record names through the “Retrieve Record Names Only” screen. No record names will be seen if the database is empty.

Priority: Very High  
Dependencies: The “Login”, “Register”, clipboard, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 31 Requirement Type: 6 (Usability)

Description: The application cannot be accessed if there are no registered users.

Rational: The application should not be accessible if no user has registered.

Fit Criterion: A “Database is empty! Please register to use the application.” message will be displayed if no user has registered.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 32 Requirement Type: 6 (Usability)

Description: Users should be able to delete all their records.

Rational: Users should have an option to delete all their records stored in the database. They should not have to do it one at a time.

Fit Criterion: All records must be deleted from the database and application if the user selects and confirms the “Delete all records” option.

Priority: Very High  
Dependencies: The “Login”, “Register”, delete all, database, and encryption sections must be implemented before this requirement can be fulfilled.

Requirement ID Number: 33 Requirement Type: 6 (Security)

Description: All SQL statements will be composed of prepared statements

Rational: Using prepared statement protects the application from SQL injections.

Fit Criterion: Users should not get any information when attempting to perform SQL injections.

Priority: Very High  
Dependencies: The “Login”, “Register”, database, and encryption sections must be implemented before this requirement can be fulfilled.

**Additional Requirements:**

**Securing Database (Security)**

In order to secure the database and the information within it, access into the database will be limited and will require a password to authenticate. Ideally only the root user will have control and access to the database. Other accounts will need well defined rules for database access, views access, and data storing procedures based on needs and the role of the individuals. To protect the information within the database, all the data will be encrypted before being stored in the database. In order to prevent unauthorized connections into the database, access will be limited to local host. Additionally, in order to prevent SQL injections, all queries will be pre-defined within the application, and then the parameters will be passed on to the query after being sanitized. This will validate the user input and prevent users from creating their own queries.

**Clipboard Vulnerability (Security)**

Most Windows password manager applications utilize the clipboard to allow users to copy and paste a specified account password. This is a convenient feature as it prevents users from having to type long and complicated passwords, but it can also be exploited if the computer was infected with malware. When the password is copied into the clipboard this data is temporarily stored in the memory of the computer. The longer the password is kept in the clipboard the longer it will remain in memory. This creates a window of opportunity for an attacker to access the system and retrieve the password from the clipboard or the memory of the computer. To prevent this from happening password managers flush the clipboard after a certain amount of time has passed. A feature like this will be created and implemented in this password manager. The application will alert the user that the password will be wiped from the clipboard after fifteen seconds of retrieving the password. The user may retrieve the password again if they need it after the first fifteen seconds. In addition to clearing the clipboard after fifteen seconds, it will also be cleared when the user logs out, and lastly when the user exits the application. This feature will not completely prevent this vulnerability, but it will reduce the probability of it happening and it is the industry standard.

**Project Implementation Description**

The initial setup of the application consists of modifying some values within the code and creating an email for the application to use. First, user’s need to edit the “database.py” file and add their database username on line four, and then the database password on line five. For more details, please see the example below:

Line 4: 'user': 'root',

Line 5: 'passwd': '12345',

***Note: Replace “root” with your username and “12345” with your password*.**

The next step is to add an email for the database to use. It’s recommended that a new email is created solely for the purpose of running this application. Once the email has been created, users need to modify line twenty-nine of the “passwordRcovery.py” file and add the email address and password of the new email that the application will use. For more details, please see the example below:

Line 29: s.login("xecure@gmail.com", "123")

***Note: Replace “xecure@gmail” with your email and “123” with your password*.**

After making these changes the application will be ready for use. To run the application users, must run the “main.py” file. This will create the MYSQL database along with all the files necessary to start and run the Xecure Password Manager Application.

Once the application is started users will be greeted with the “Main Menu”. From this menu users can log into the application by pressing “1”, create an account by pressing “2”, get help with their login information by pressing “3”, or close the application by pressing “4” (see figure 1 for reference). If the user has not registered, they will not be able to access the “Login” or “Forgot Login” sections. In this case a message will be displayed letting the user know that the database is empty and that they need to register to use the application. If the user already has an account, the “Login” screen will prompt the user to enter their username and their password (see figure 2 for reference). However, if the user wishes to go back to the previous screen, they can enter the “0” command, and it will take them back to the previous screen. This command can be used all throughout the application. In order to authenticate users a SHA256 hash is created from their username and password. If this hash matches what is on the database, they are successfully logged in to the application. If the user fails to provide the proper username and password after six login attempts, the application closes itself. A warning is given to the user after five failed login attempts.

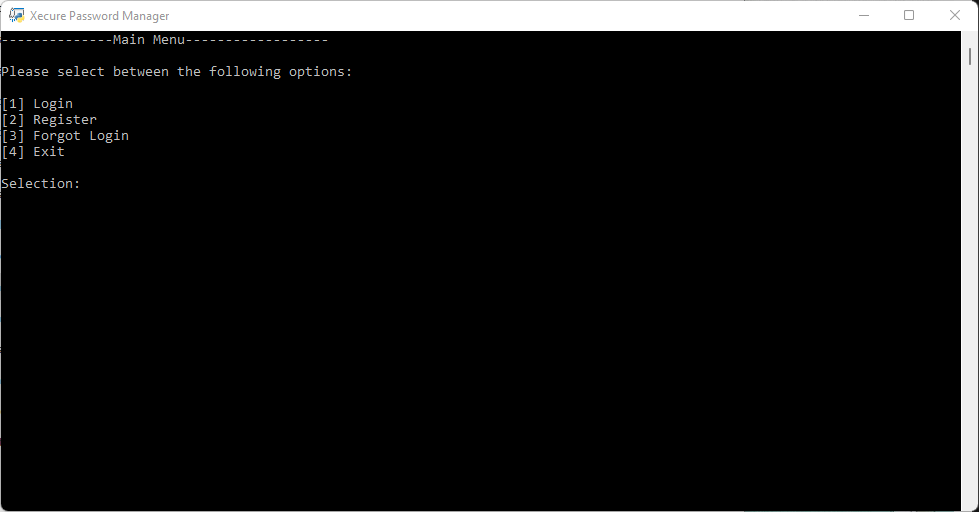
****

Figure 1: Main Menu Screen

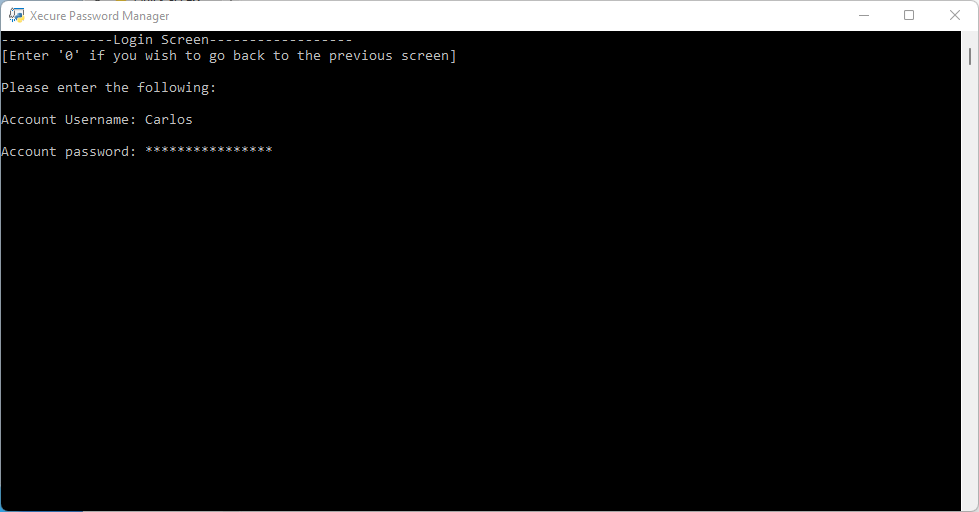


Figure 2: Login Screen

In order to use the application, users need to create an account. This can be done through the “Register” menu. This section will ask users to enter an email address, a username, and a password. The email address and the username must be unique, and the password must be longer than eight characters, it must contain both upper and lowercase letter, and it must contain numbers and symbols. Once a valid email, username, and password are entered the user will be greeted with a “Thank you for registering message” (see figure 3 and 4 for reference). They can then use this information to log into the application.

If a user is having difficulties login into the application, they can access the “Login Help” menu where they will be given the option to retrieve their username or reset their password (see figure 5 for reference).

Text

Description automatically generated

Figure 3: Register Screen

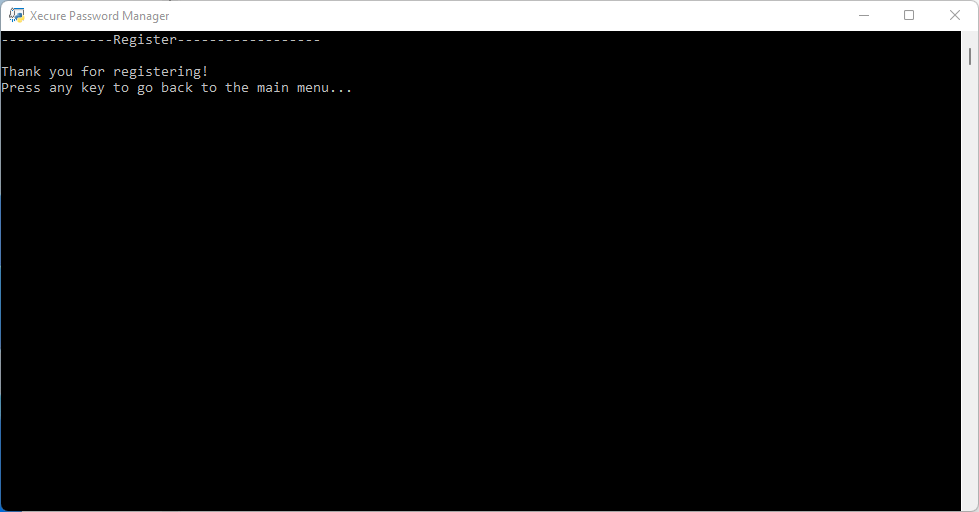


Figure 4: Successful Registration Screen

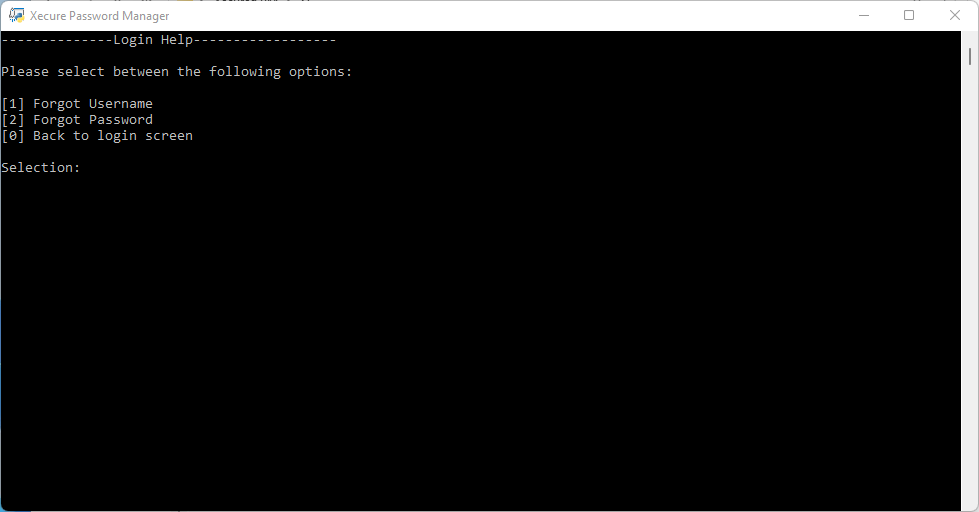


Figure 5: Login Help Screen

If the user selects the “Forgot Username” selection they will be asked to enter their email address, and if the email address has been registered in the database, the username will be sent to the entered email address (see figure 6 for reference). The user can then use this information to log into the application. On the other hand, if the user forgets their password, they can select the “Forgot Password” selection, and this will allow them to reset their password after verifying their email. In the “Reset Password” screen the user will be asked to enter their email, and if the entered email matches an email that has been registered in the database, a verification code will be sent to that same email address. The user will then have to retrieve that code from their email and enter it into the Xecure Password Manager application. If the entered code matches the code sent to the email, the user will be asked to enter a new password. If the code entered does not match the code sent to the email, the user will be sent back to the main menu, and they will not be able to change their password (see figure 7 for reference).

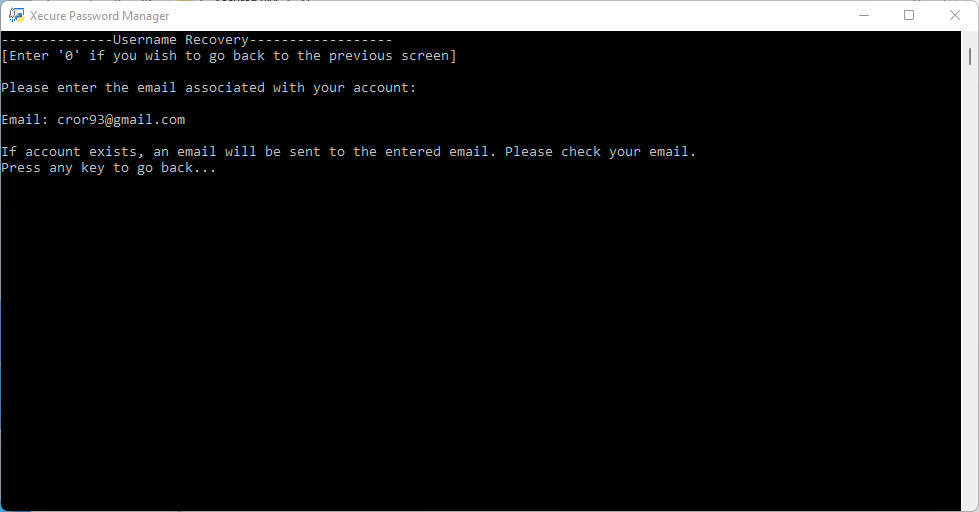


Figure 6: Username Recovery Screen

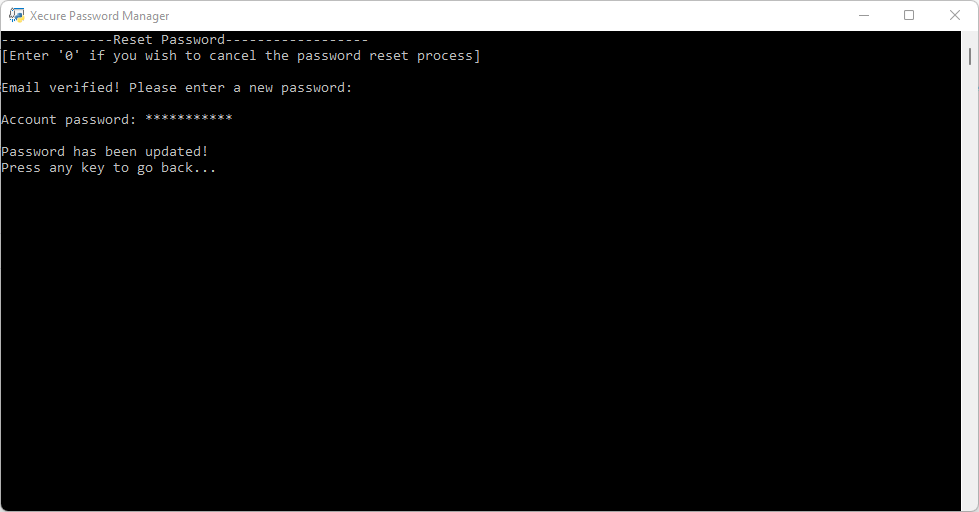


Figure 7: Reset Password Screen

Once a user logs into the application, they will be greeted with the “Dashboard” menu. From this menu users can select between the options of retrieving a record, adding a new record, updating a record, viewing all records, viewing the record names of all their records, settings, logging out, and exiting the application (see figure 8 for reference). However, if the user has not added any records, they will not be able to use the options of retrieving a record, updating a record, viewing all records, and viewing all record names only. They will instead, see a message letting them know that there are no records in the database, and they will be asked to add a new record in order to use those features.

The “Retrieve a Record” option allows users to retrieve the information of a record they have previously entered. Once this option is selected the user will be asked to enter a record name, and if this record is found, all its information will be displayed to the screen (see figure 9 for reference). Additionally, the password of the record will be copied to the clipboard for fifteen seconds. After fifteen seconds the password will be wiped from the clipboard.

Text

Description automatically generated

Figure 8: Dashboard Screen

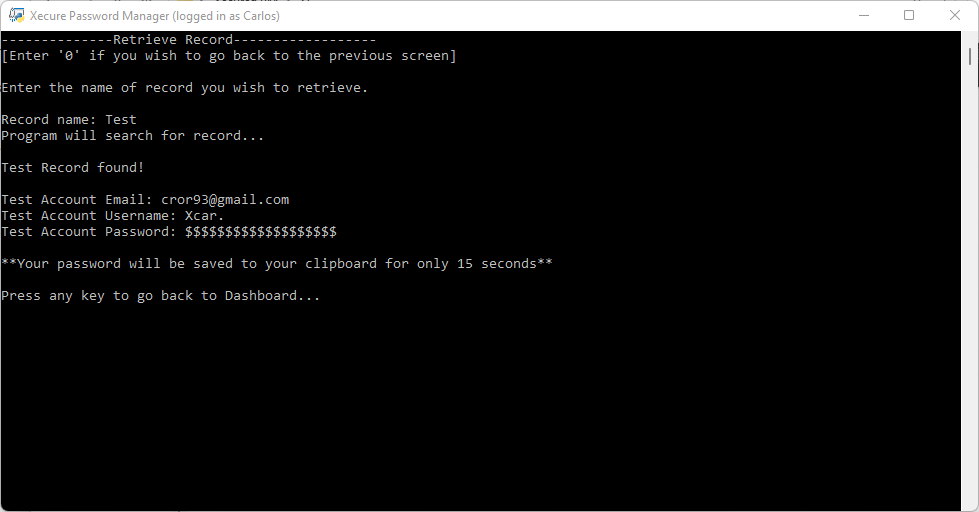


Figure 9: Retrieve Record Screen

To use all the features of the application, users have to add at least one record to their account. This can be done through the “Add a Record” option from the dashboard menu. The “Add Record” menu will prompt the user to enter a record name (see figure 10 for reference). This record name must be unique and cannot be composed of only symbols or numbers. The record name will later be used to retrieve the record. The user will then be prompted to enter a record email and username for the record, but both of this options are optional. Lastly, the user will be asked to enter a password, but the user has the option of generating a random password or entering their own password (see figure 11 for reference). If the user selects the option to generate a new password, a randomly generated password will be displayed to the screen and the user will be asked if he wants to use that password (see figure 12 for reference). If the user is not happy with this password, they can try another randomly generated password or they can add their own password (see figure 13 for reference).

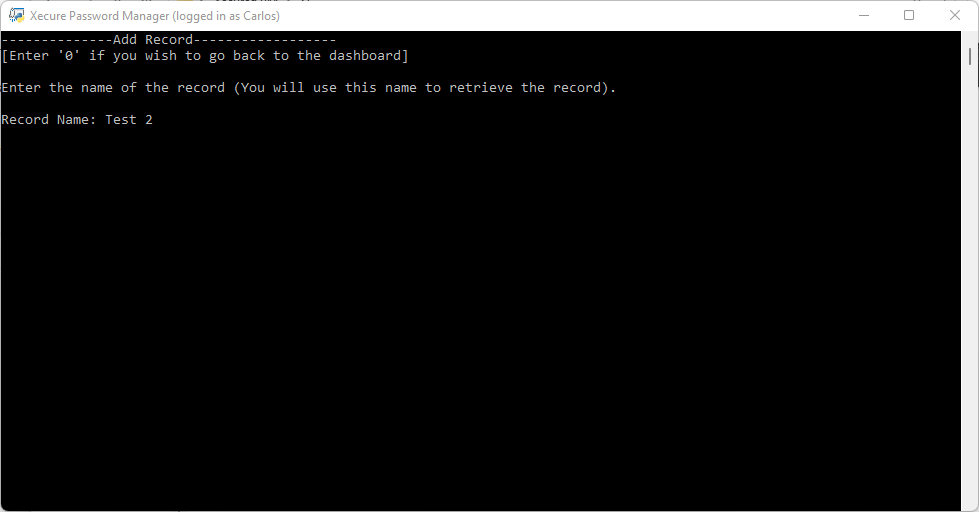


Figure 10: Add Record Screen

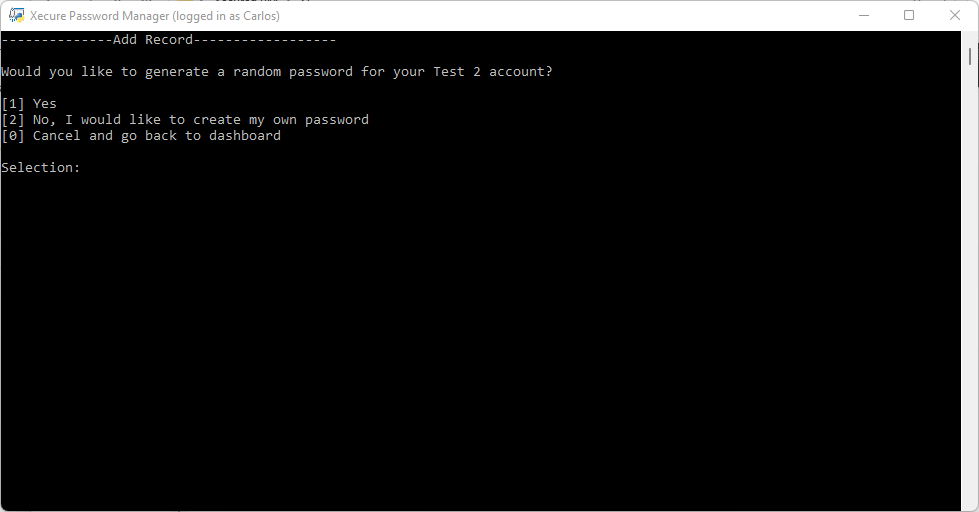


Figure 11: Add Record – Generate Random Password Screen

Text

Description automatically generated

Figure 12: Generate Random Password Screen

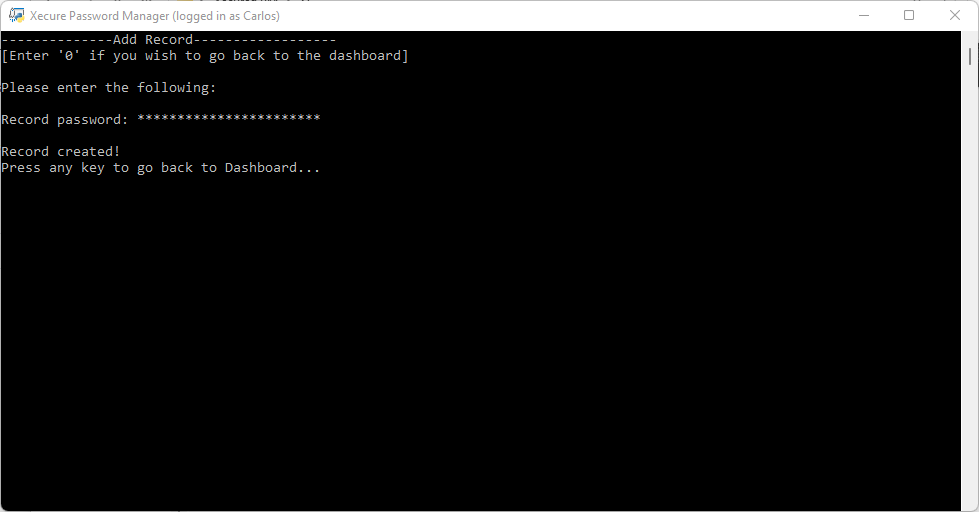


Figure 13: Enter Your Password Screen

Users can edit their records by using the “Update Record” option from the dashboard. Once the user selects the “Update Record” option they will then be prompted to enter the record name of the record they wish to update (see figure 14 for reference). If the record name entered is found within the user’s data, they will be prompted to select what information they want to change about the record (see figure 15 for reference). The user can choose between editing the record’s name, email address, username, or password. Once the user selects an option, they will be asked to enter a new value to update the record to. If the entered input is valid, the record will be updated and the next time the user retrieves that record it will contain the newly updated information (see figure 16 for reference).

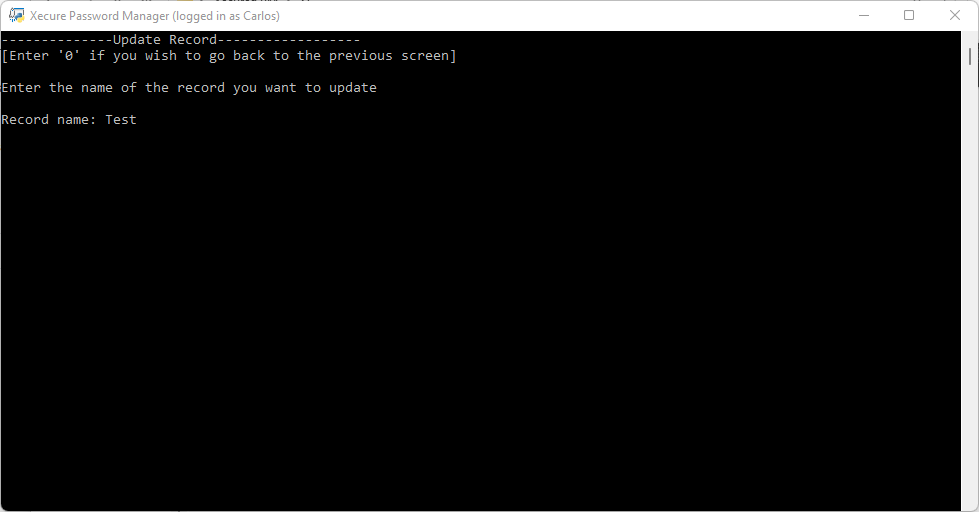


Figure 14: Update Record Screen

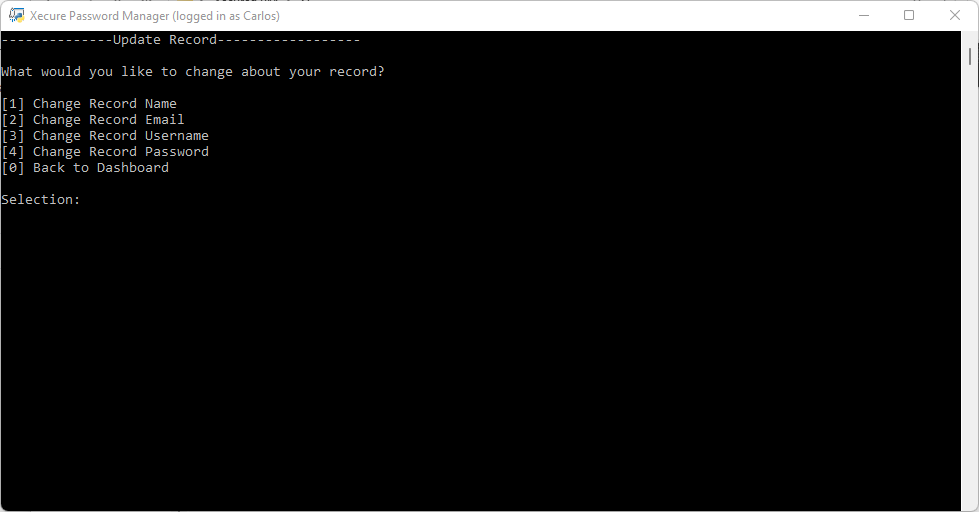


Figure 15: Update Record Screen

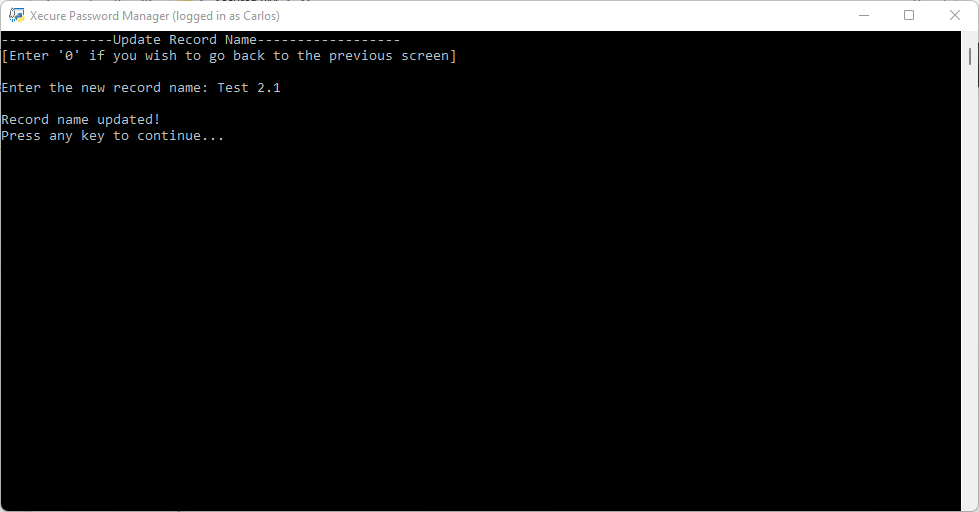


Figure 16: Updating Record Name Screen

Users have the option of displaying all their records by using the option of “View All Record” from the dashboard menu. This will display all the records from the currently logged in user, along with all the information stored within them (see figure 17 for reference).

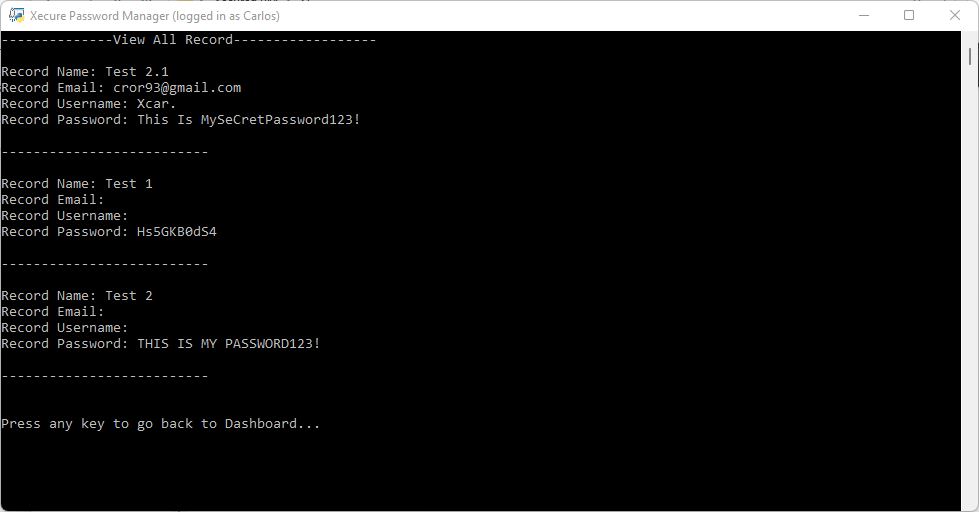


Figure 17: View All Records Screen

Users also have the option of retrieving just the names of their registered records (see figure 18 for reference). This can be used to find a specific record that can then be retrieved individually by the “Retrieve a Record” option from the dashboard. This is a safer option as it will not display the information contained within the record unlike the “View All Records” option.

Text

Description automatically generated

Figure 18: View All Record Names Screen

The application also contains a settings menu that gives users more control over their data within the Xecure Password Manager application. This menu can be accessed by selecting the “Settings & Delete Records” from the dashboard. From the settings menu, users can access the commands to delete a record, delete all records, change master email, or change master password (see figure 19 for reference).

Text

Description automatically generated

Figure 19: Settings Screen

Users can delete records by selecting between the “Delete a Record”, and “Delete All Records” options from the settings menu. If the user wants to delete a single record, they can user the “Delete a Record” option. This menu will prompt the user to enter the name of the record they wish to delete (see figure 20 for reference). If the record name entered is found within the user’s data, the application will prompt the user to confirm the deletion or cancel this action. If the user confirms the deletion of the record, it will be removed from the database and this data cannot be recovered (see figure 21 and 22 for reference). On the other hand, if the user wishes to delete all their records, they can select the “Dele All Records” from the settings menu. This option deletes all the records contained within the user’s account. Once this option is selected the user will be prompted to confirm the action before deleting the records. If the user confirms the deletion, all the records on the user’s account will be deleted (see figure 23 and 24 for reference).

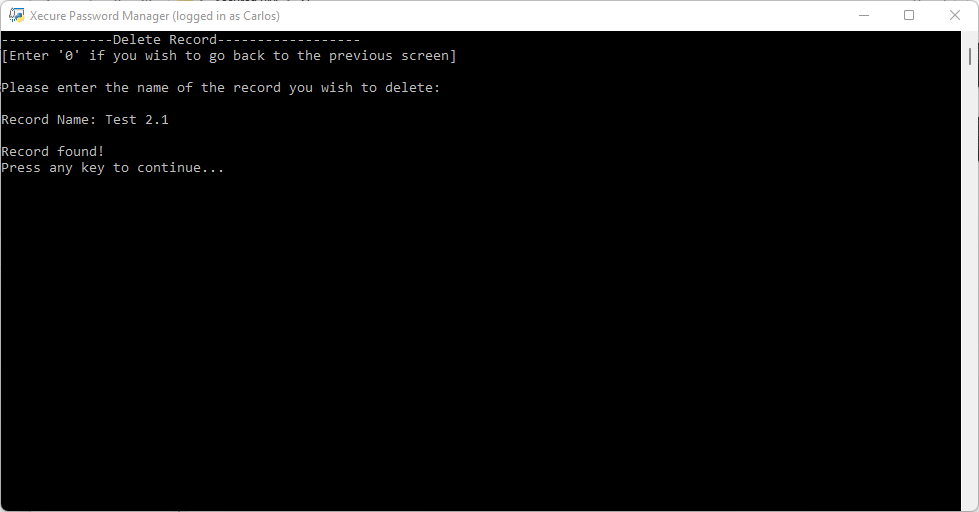


Figure 20: Delete Record Screen

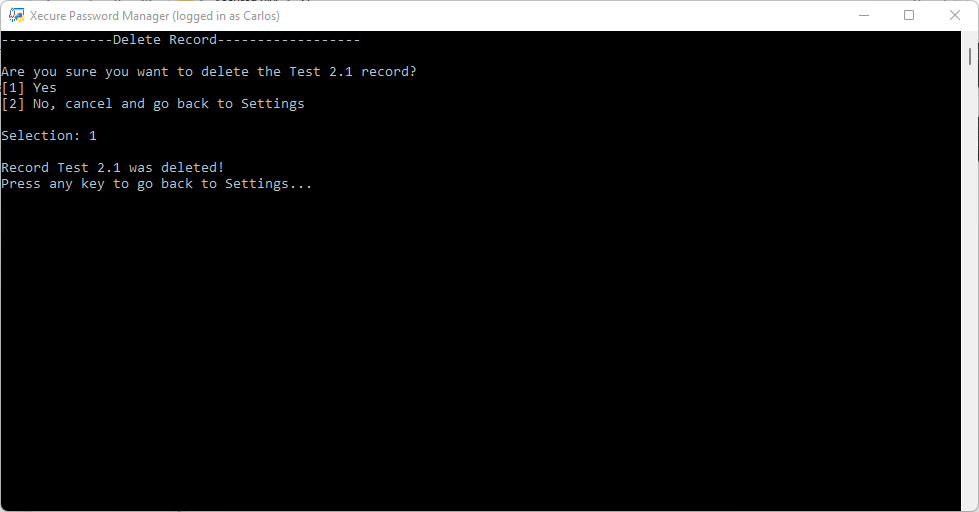


Figure 21: Deleting A Record

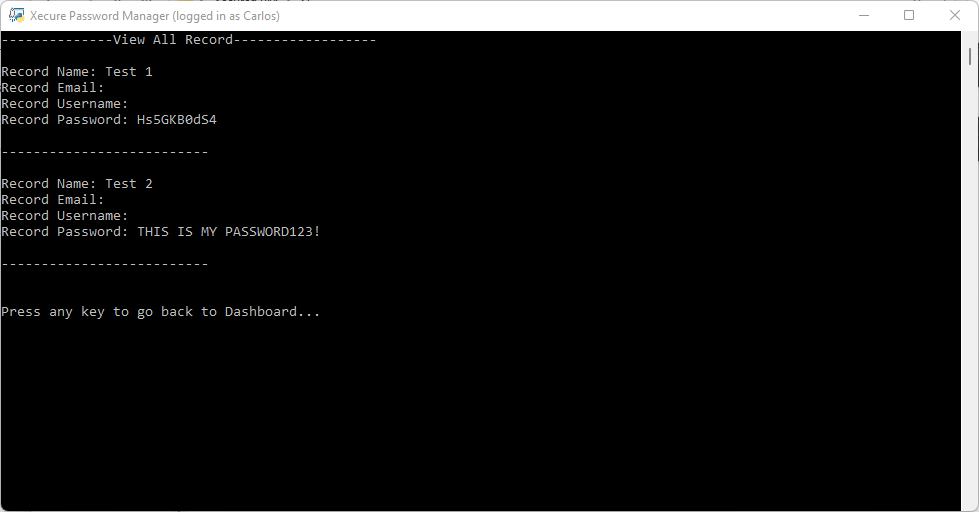


Figure 21: View All Screen After Deleting Record

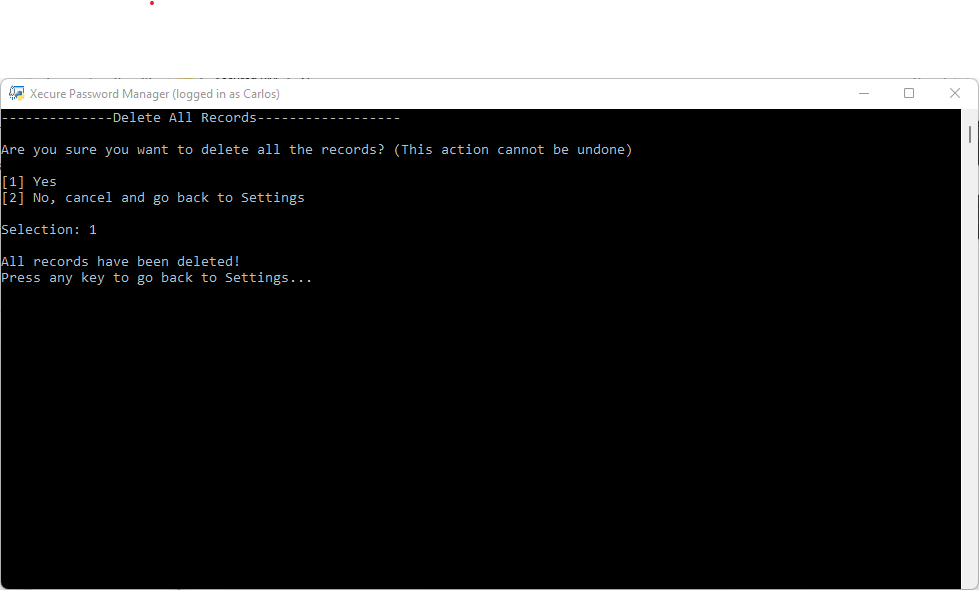


Figure 23: Delete All Records Screen

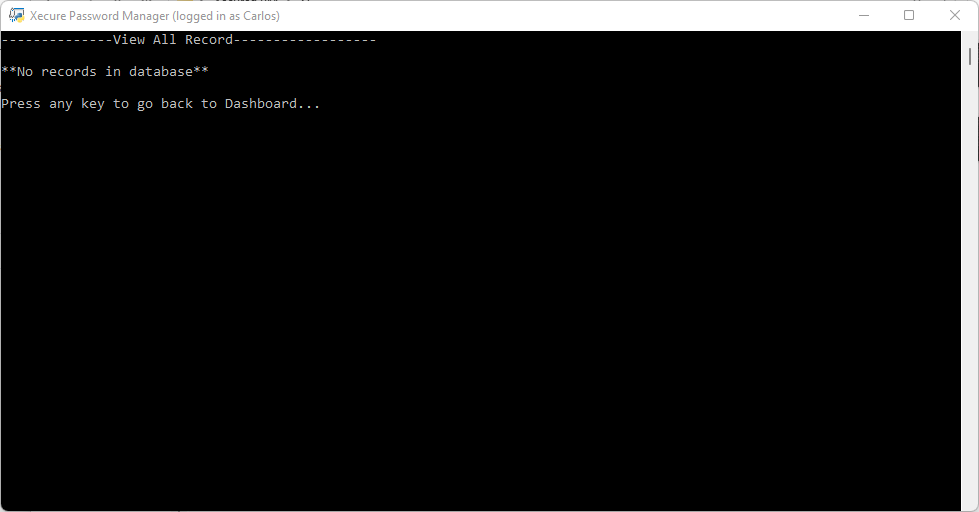


Figure 24: View All Records After Deleting All Records

User can change the master account email address by selecting the “Change Account Email” option from the settings menu. Once selected, the user will be prompted to enter the email associated with their account (see figure 25 for reference). If the email entered matches the account’s email address, the user will be prompted to check their email and enter a code (see figure 26 for reference). After verifying the entered email, the application sends an email to the user with a verification code that needs to be entered on the Xecure Password Manager Application. If the code entered matched the code that was sent to the user, the user will be prompted to enter a new email address. If the entered input is valid, the user’s master email will be updated, and all future emails will be sent to this new email address (see figure 27 for reference).

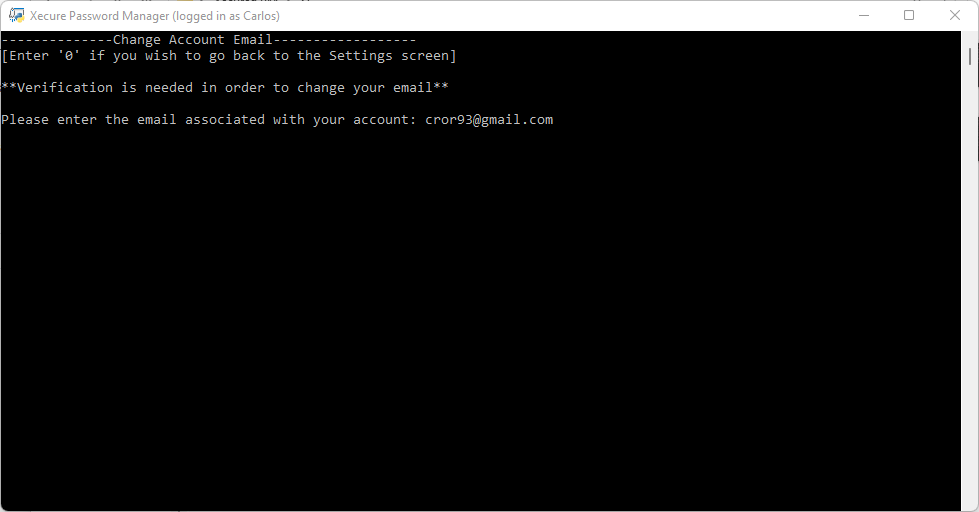


Figure 25: Change Account Email Screen

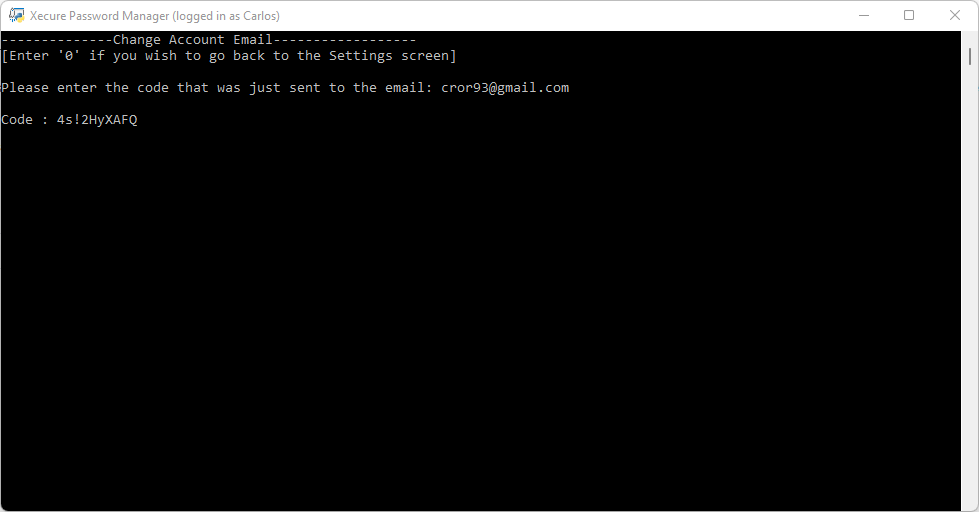


Figure 26: Change Account Email Screen – Verification Code

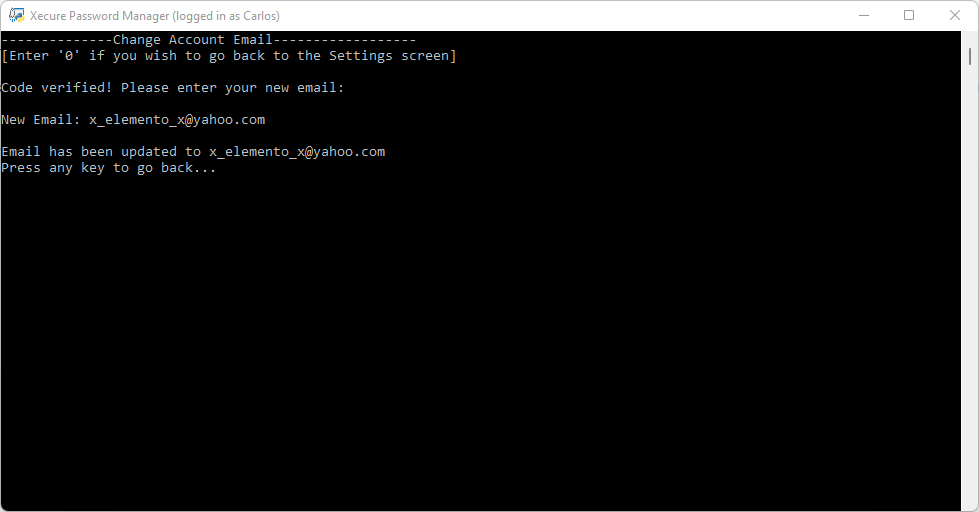


Figure 27: Updating Account Email

User can also change the “master” password used to login by selecting the “Change Account Password” option from the settings menu. Once selected, the user will be prompted to enter the email associated with their account (see figure 28 for reference). If the email entered matches the account’s email address, the user will be prompted to check their email and enter a code (see figure 29 for reference). After verifying the entered email, the application sends an email to the user with a verification code that needs to be entered on the Xecure Password Manager Application. If the code entered matched the code that was sent to the user, the user will be prompted to enter a new password for their account. If the password entered is valid, the user’s master password will be updated, and they must use this new password to login (see figure 30 for reference).

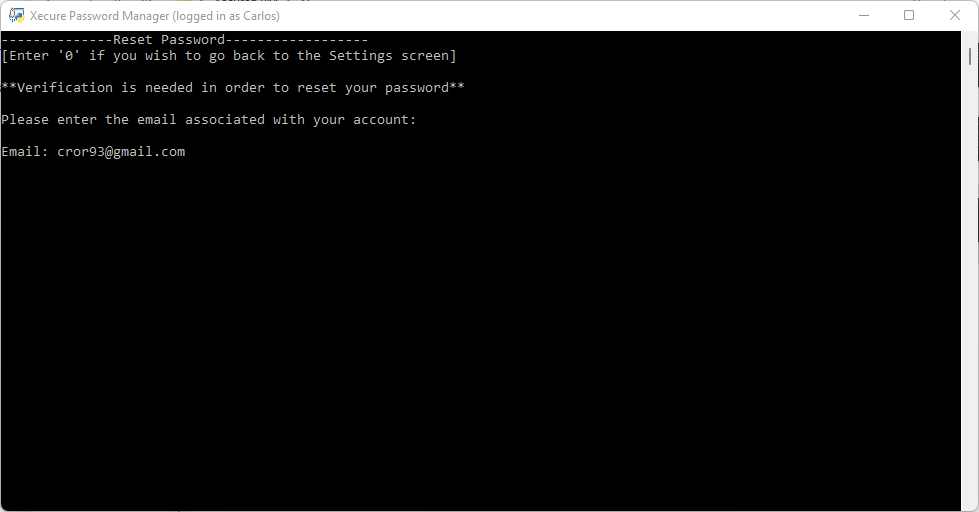


Figure 28: Reset Password Screen

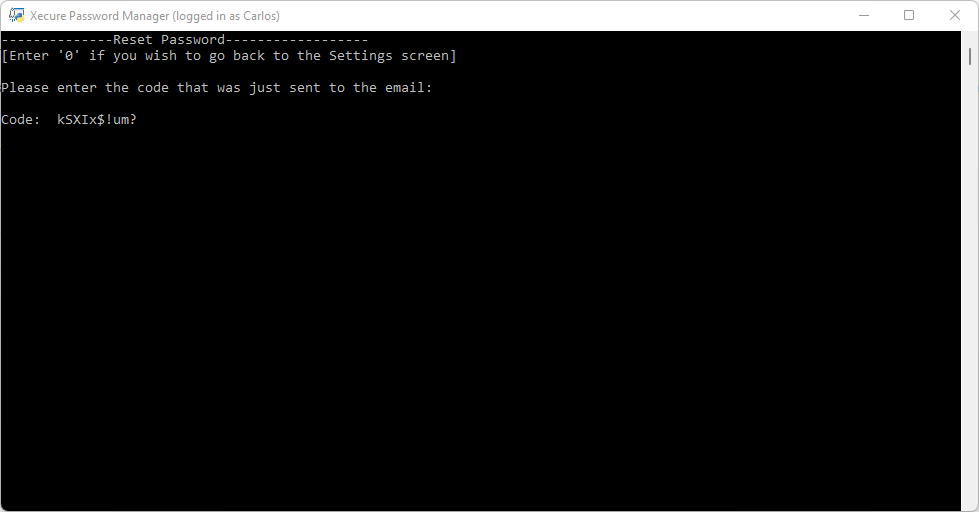


Figure 29: Reset Password Screen – Verification Code

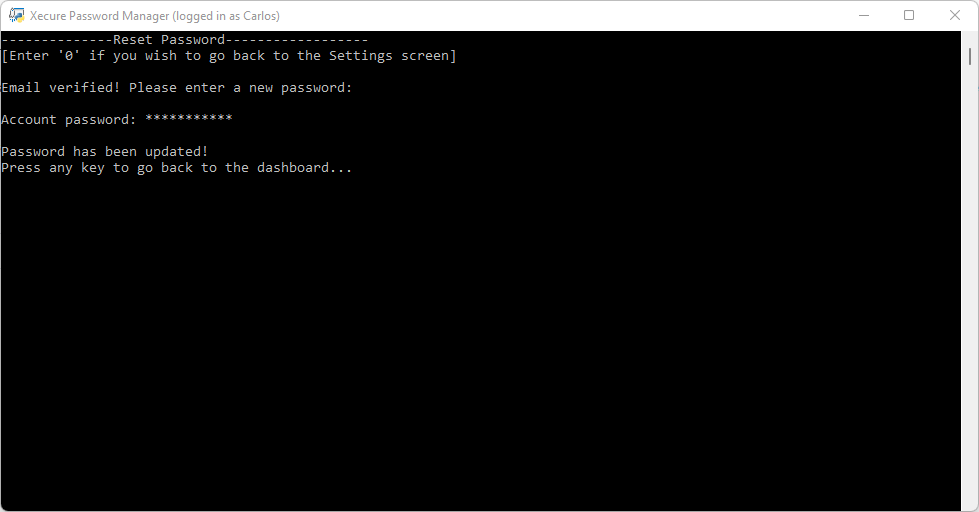


Figure 30: Resetting Account Password Screen

The logout menu can be accessed from the “Logout” option on the dashboard, and it can be used to logout the user that is currently logged in without closing the application. Once this option is selected the user will be asked to confirm their selection (see figure 31 for reference). If the user confirms the selection, they will be logged out of their account and they will be taken back to the main menu. Alternatively, if the user wishes to logout and close the application, they can choose the “Exit” option from the dashboard. Once selected the user will be asked to confirm their selection (see figure 32 for reference). If the user confirms the selection, the application will close, and the user will be logged out. Both of these options will wipe the clipboard to ensure that any passwords that were previously saved will not remain after closing the application.

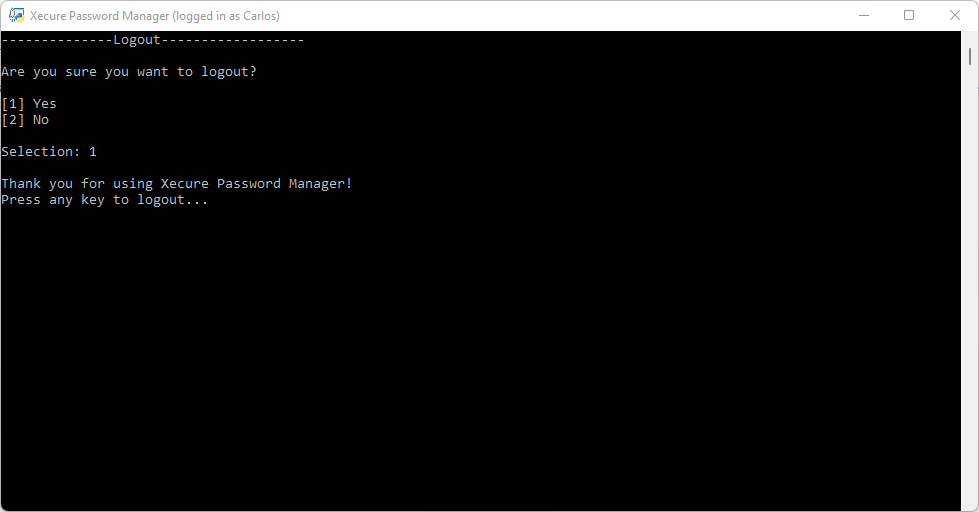


Figure 31: Logout Screen

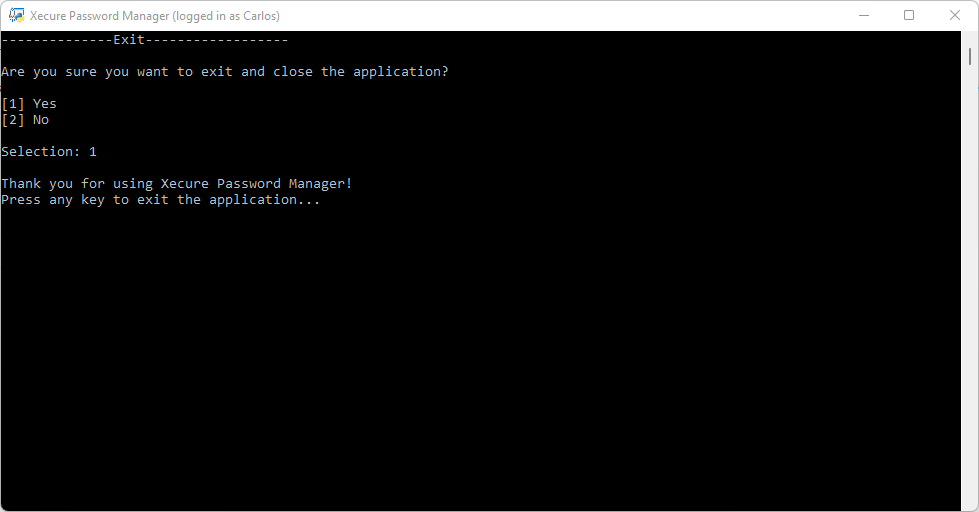


Figure 32: Exit Screen

Various input validation, and error handling methods have been implemented. For example, the menu screen will only accept input that corresponds to the options displayed on the screen. If an invalid input is entered, an error message will be displayed, letting the user know what went wrong (see figure 33 for reference). When attempting to login, carefully constructed error messages are displayed. This is to prevent attackers from getting information about registered accounts (see figure 34 for reference). Inputs that must be unique like usernames, and emails display an error message letting the user know that the entered value is invalid. The application will then prompt the user to enter another value that is unique (see figure 35 for reference). Lastly, creating weak master passwords will result in an error letting the user know the requirements for that input (see figure 36 for reference).

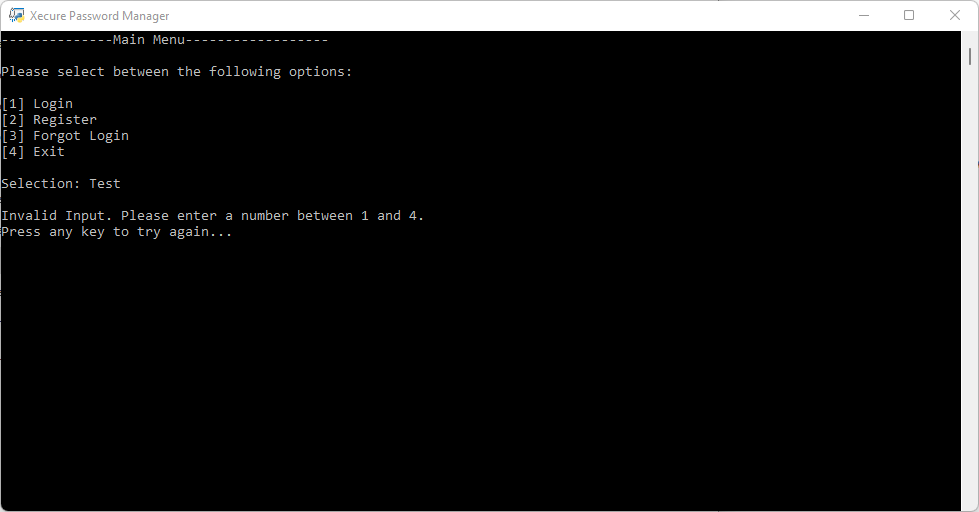


Figure 33: Main Menu Invalid Input

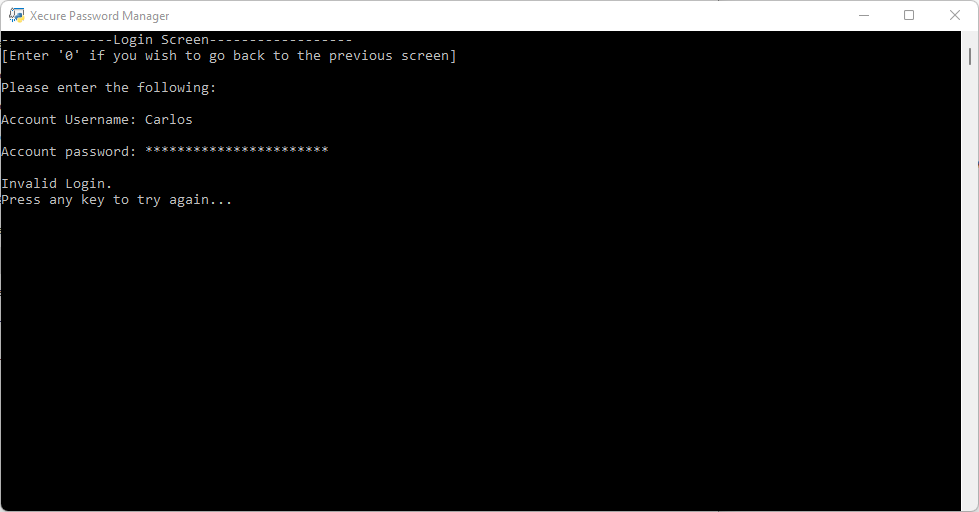


Figure 34: Invalid Login Attempt

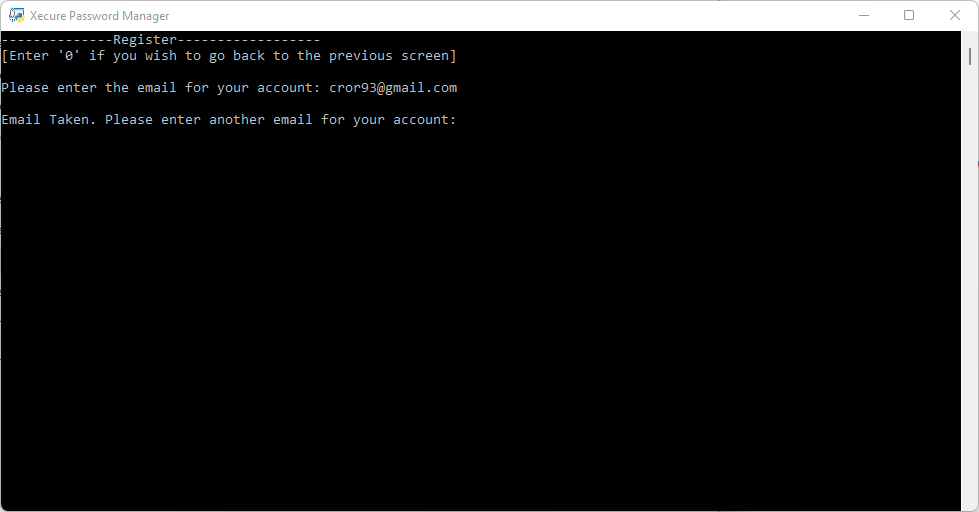


Figure 35: Register Screen Email Taken Error

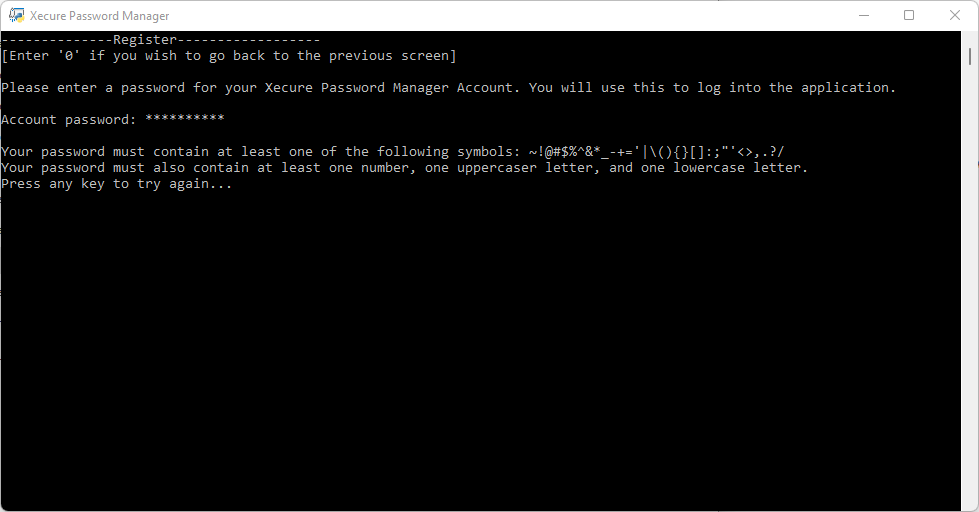


Figure 36: Register Weak Password Error

A link to the GitHub repository can be found here: https://github.com/Xcar17/Xecure-Password-Manager

**Test Plan**

The first step of the testing process involved creating test cases for the application. For a detailed look of the test cases please refer to the “Xecure Password Manager Test Cases” document. This document describes in detail the behavior of the application given an input. The tests cases were created through an iterative process of creating test cases to test the application, and then gathering feedback from volunteers. This process was repeated three times and the result was the current version of the “Xecure Password Manager Test Cases” documents found in the GitHub repository.

The user feedback process involved finding volunteers to test the application for usability and errors. For these tests three users were selected and each one of them represented a specific demographic (inexperienced, experience, and very experienced). All three volunteers were asked to perform the following steps:

Tasks:

1 – Use the Xecure Password Manager application to register and make an account.

2 – Log into the Xecure Password Manager application.

3 – Add two new records to your account. Use a randomly generated password for your first record. For your second record, enter your own password.

4 – Retrieve and display the information for one of your records.

5 – Retrieve and display the information for all of your records.

6 – Update the password of one of your records, then retrieve and display it to the screen.

7 – Delete one of your records and then display all your records.

8 – Change your Master password, then log out and try to login with your new password.

9 – Delete all your records, and then try to display all your records to the screen.

10 – Exit the application.

**Test Results**

Major Findings for the Usability Tests:

1. Application needed a consistent way to back out of menus.
2. Remove User Id information.
3. Better description on verification emails needed.
4. The Delete Record options difficult to find.
5. Email addresses being case sensitive.
6. Verification codes too complicated.

Usability Test Fixes:

1. A “back” option was added to every screen, and input prompt.
2. Information about user ID, was being displayed but this was corrected.
3. Emails now contain a better description of what to do and why the email was sent.
4. The “Settings” option is now “Settings & Delete Records”. This will help users find the delete record options.
5. Application was updated to remove case sensitive emails.
6. Verification codes now limit the number of special characters it uses. Resulting in codes that are easier to read and type.

Major Findings for the Test Cases:

1. Application does not sanitize clipboard if user force closes the application.
2. No email validation before registration.
3. Menu selections accept inputs with leading zeros.
4. Cannot properly exit application until the fifteen seconds of the clipboard are done.
5. Cannot properly logout of the application until the fifteen seconds of the clipboard are done.

Test Case Fixes:

1. Thread handling clipboard functions can now be stopped when exiting application.
2. Thread handling clipboard functions can now be stopped when logging out.

The following findings were not corrected in the current version of the application:

* Menu selections accept inputs with leading zeros.
* No email validation before registration.
* Application does not sanitize clipboard if user force closes the application

The previously mentioned findings were not able to be corrected in the current version of the program due to various constraints. These findings were found late in the testing phase and a fix would have required a lot of the testing to be re-done. The “leading zeros” finding does not affect the usability or security of the application and therefore was identified as a low priority. The next update of the application will validate the input to make sure that selections with leading zeros are not accepted. The emails not being validated before registering is a medium severity finding, as it could lead to users signing up with an incorrect email address due to typos. The next update will send a code to the user’s email to verify the email before registration. Lastly, the current structure of the application cannot prevent the user from force closing the application and this can lead to the program closing without sanitizing the clipboard. This bug is tied to the Windows command line, and a solution has not been proposed as of this moment. To minimize this risk, it is advised that users properly exit the application thorugh the dashboard commands.

**Challenges Overcome**

One of the biggest challenges was the time constraint. Due to financial reasons, I was forced to take the project design and project implementation during the same semester. This was very challenging as I had very little time to research and perform initial tests before I needed to decide on a project and begin its implementation. This also meant that I had I to stick with most of my decisions as I had no time to look for other alternatives. Another big challenge was using the Python language as I had no experience with this programming language. I chose this as my project’s programming language because I had always wanted to learn python and using it to create my graduation project would give me the hands-on experience that I desired. However, the biggest challenge came from the encryption and decryption process. I never worked with encryption or decryption algorithms and understanding and applying some of the concepts proved to be very challenging. I had many issues trying to encrypt and decrypt data from the database and this was by far the most time-consuming part of the project. After succeeding in encrypting and decrypting data from the database I had difficulties linking the data to users (since the data was encrypted). The solution to this problem took several weeks, but I was able to address it by giving every user a “user ID” that would be the link between their account and their encrypted data. This user ID is created from a hash that is created from the user’s account name and their password. After these challenges were solved, it was only a matter of fixing bugs and testing the application.

**Future Enhancements**

Due to the nature of Password Manager applications, it is important to maintain a plan to support and update the application. New security vulnerabilities are discovered every day and with the advancements of computing, cracking passwords becomes a much easier task every year. The features below are considered necessary for the success of the application and should be implemented to improve the usability and security of the Xecure application.

Future Updates:

* Email Validation – The current version of the application cannot validate that the entered email is a valid and working email. To improve the security of the application users should have to verify their email address before they can register or changer their email address.
* Weak Password Identification – A feature that identifies weak passwords for user records will be implemented in the future. This feature will display a warning message letting the user know that their password is considered weak.
* Better Input Validation – Better input validation will improve the security of the application, but this is a time-consuming task, and it requires continuous monitoring and updating.
* Better Master Password Policies – The current version of the application follows the basic guidelines of the NIST 800-63B. However, a stricter password policy should be created for the “master” password. This new policy should prevent users from using well known passwords, and previously used passwords.
* Confirm Password/Email – The current version of the application only prompts the user to enter their email/password once. In a future update the user will be asked to re-enter their password/email in order to confirm the input. This will avoid users getting locked out due to typos.
* Additional Authentication Methods – Adding other authentication methods will improve the security of the application. Additional authentication methods can include one-time codes sent to the user’s phone, user security questions, and or others.
* Better Lockout – The current version of the application only closes the application after six failed login attempts. A feature that locks the account based on the username will be implemented. This will ensure that the locked user cannot login without verifying their account through a code sent to their email address.
* Reject Leading Zeros – The current version of the application accepts menu selections that have leading zeros (as long as the input is composed of only leading zeros and a valid integer at the end). A fix will be implemented to validate input that contains leading zeros and reject the input.
* GUI Version – A version that utilizes a graphical interface would make it easier to use and should be possible now that the foundation has been implemented.

**References**

Google. (2019, February). *Online Security Survey*. Google. Retrieved from https://services.google.com/fh/files/blogs/google\_security\_infographic.pdf.

Greig, J. (2020, July 29). *IBM finds cyberattacks costing companies nearly $4 million per breach*. TechRepublic. Retrieved from https://www.techrepublic.com/article/ibm-finds-cyberattacks-costing-companies-nearly-4-million-per-breach/.

Lord, N. (2020, September 29). *Uncovering password habits: Are users' password security habits improving? (infographic)*. Digital Guardian. Retrieved from https://digitalguardian.com/blog/uncovering-password-habits-are-users-password-security-habits-improving-infographic.

Mavituna, F. (2022, March 14). *SQL Injection Cheat Sheet*. Netsparker. Retrieved from https://www.invicti.com/blog/web-security/sql-injection-cheat-sheet/

Test Cases Preview

https://github.com/Xcar17/Xecure-Password-Manager

(Please Visit GitHub Repository for a detailed look of the test cases)