

Criterion B: Design Overview

My design methodology for creating a password manager involves using object-oriented programming to manage complexity and structure, making it easier to maintain the code over time. I will implement a modular design to accommodate the many different components of my program, such as user interfaces, data storage, and encryption/decryption. Doing so will allow me to develop and test independently, improving code quality and making debugging errors easier. I will also follow the agile development process to ensure all my client's requirements are met.

Input

Input	Data Type	Normal Range	Example
Master Password	String	8-20 characters, including letters, numbers, and special characters	MySecr3tP@ssword
Confirm Master Password	String	8-20 characters, including letters, numbers, and special characters	MySecr3tP@ssword
Account Name	String	Up to 20 characters	Amazon
URL	String	Up to 255 characters	www.amazon.com/login
Email	String	Up to 255 characters	john@doe.com
Username	String	Up to 20 characters	john.doe123
Password	String	8-20 characters, including letters, numbers, and special characters	P@assw0rd321
Confirm Password	String	8-20 characters, including letters, numbers, and special characters	P@assw0rd321
Renewal Period	int	3, 6, 9, or 12	12
Comments/Notes	String	Up to 500 characters	This is an example note for this account.
Current Date	LocalDate	--	2023-01-15
Search	String	Up to 20 characters	amazon

Output

Output	Data Type	Normal Range	Example
Login Attempts remaining	int	2-1	2
Error Message	JOptionPane	--	"Passwords do not match"
Generated Password	String	30+ characters, including letters, numbers, and special characters	x4TOWP\$z,t523C^gDtux0xs*F4Hl#en
Account List	HashMap<int, String>	List of accounts ordered by view count in descending order	{19=Amazon, 15=Bluehost, 13=Google, 9=Facebook}
Account Name	String	Up to 20 characters	Amazon
URL	String	Up to 255 characters	www.amazon.com/login
Email	String	Up to 255 characters	john@doe.com
Username	String	Up to 20 characters	john.doe123
Password	String	8-20 characters, including letters, numbers, and special characters	P@assw0rd321
Comments/Notes	String	Up to 500 characters	This is an example note for this account.
Search Results	HashMap<int, String>	List of accounts ordered by view count in descending order	{13=Google, 9=Facebook}
Renewal Period	int	3, 6, 9, or 12	12
Creation Date	DateTimeFormatter	--	2023-01-15
Renewal Date	DateTimeFormatter	--	2024-01-15
Renewal Message	String	~40 characters	This deadline is more than a week away.
Percent Complete	int	0-100	75
Password Scan Results	String	Many characters	The password for 'Example #5' received a password score of 6/100, a low score. We recommend changing this password. --

Design Prototypes

I created the following design sketches before starting my GUI development in NetBeans. These sketches served as reference material and allowed me to better understand the program's functionality and user interface. Working on the designs beforehand, I could identify potential issues and refine the user experience.

Figure 1 - "Create Master Password" panel

Create Master Password

Create Master Password

Forgetting a master password will result in a loss of account access.
Choose a secure password that you will remember!

New Password

Password strength meter

Confirm New Password

Create Password

Submit Button

Verify match

Figure 2 - "Password Manager Login" panel

Password Manager Login

Enter Master Password:

On Enter, log in

Figure 3 - "Password Manager Menu" panel

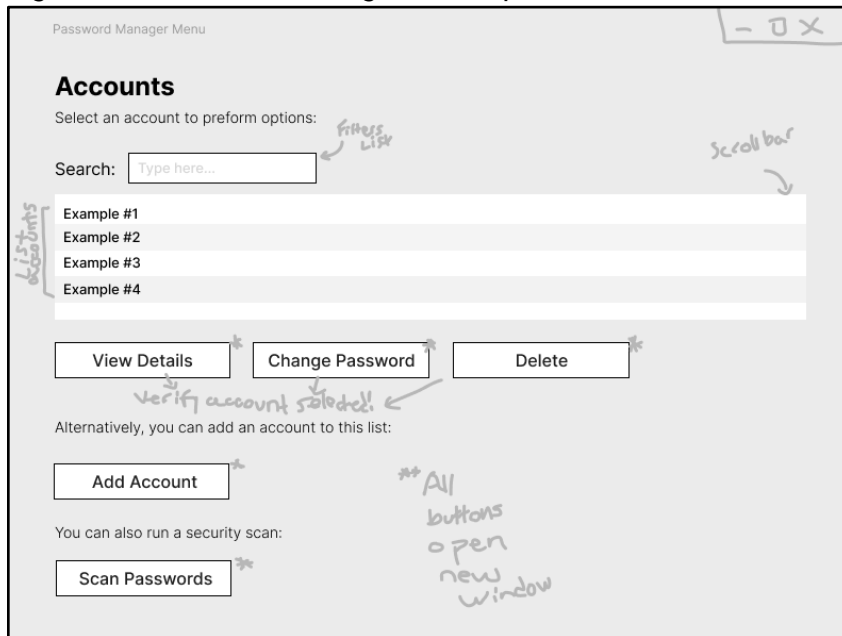


Figure 4 - "Password Scan" panel

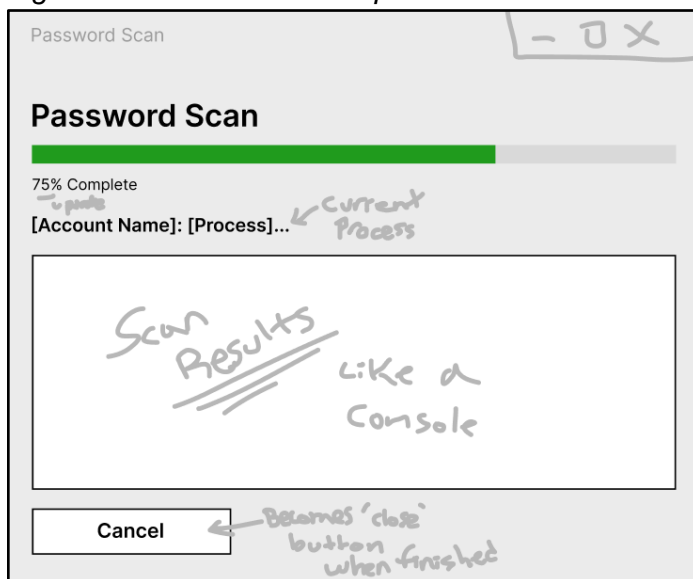


Figure 5 - "Add Account" panel

Add Account

Title

URL optional *Prefix should not be more*

Email optional

Username optional

Password *Auto strong Password both in both password field*

***** *Password Strength Red = Bad Green = good*

Generate

Confirm Password *needs to match*

Renewal Period optional

☐ 3 months

☐ 6 months

☐ 9 months

☐ 12 months

Notes optional

Text area

Add Account *verify required fields*
Submit button

Figure 6 - "Update Password" panel

Update Password

[Account Name]

Your previous password will be over-written.
Only continue if you no longer require your old password.

New Password

Confirm New Password

Handwritten notes:
- Fix fields with stars and pad
- Password strength meter
- Submit Button
- Verify match

Figure 7 - "View Details" panel

View Details

- O X

[Account Name]

URL

[url]

Open

Email

[email]

Copy

Username

[username]

Copy

Password

Show

Copy

Notes

Text area

Copy

Account created on [Creation Date]

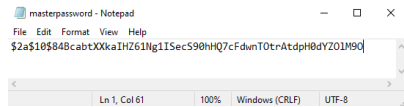
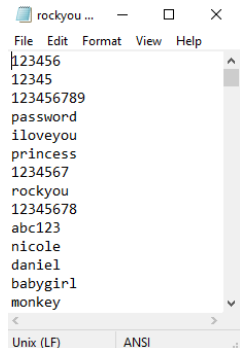
This account is set to renew every [renewal period] months.
The current renewal deadline is [renewal deadline].
This deadline is more than a week away — changes

to text

"hide"

Data Storage

Below is information about how the password manager's data is stored, which I outlined after completing my design prototypes.

File Type	File Name	Contents	Structure
XML	accounts.xml	Encrypted account information: <ul style="list-style-type: none">• Account names• IDs• URLs• Emails• Usernames• Passwords• Notes• Creation dates• Renewal periods• Renewal dates• Use counts	Hierarchical structure defined by XML tags and attributes <i>Figure 8 - accounts.xml structure</i> <pre><?xml version="1.0" encoding="UTF-8"?> <accounts> <account name="Example #1"> <account name="Example #2"> <account name="Example #3"> <account name="Example #4"> <account name="Example #5"> <id id="5628444"/> <url link="XpXKAwfZO/H0Mac5gZVsJg=="/> <email address="y/WL7ctG0a6v4+rMUnYL7Q=="/> <user user="mo/aiiGQOBZYto2GibD2IA=="/> <pass pass="IdKQDTcXQoCdVFAACbmikdw=="/> <renewal period="3"/> <note note="HmQcKZX2bUBFS3NYYJ1uHA=="/> <creation date="2023-03-26"/> <date renewal="2023-06-26"/> <usecount count="0"/> </account> </accounts></pre>
TXT	masterpassword.txt	Single hashed master password for the program to check the entered master password against.	Plain text with no structure <i>Figure 9 - masterpassword.txt structure</i> 
TXT	rockyou.txt	List of commonly used/well-known passwords that can be used for testing password security.	Plain text, with each password listed on a separate line <i>Figure 10 - rockyou.txt structure</i> 

Pseudocode

Due to having trouble getting my program off the ground, I decided to write pseudocode for the main Java file before I began actual development. It allowed me to focus on the logic of the solution rather than the details of the syntax.

Figure 11 - Pseudocode of PasswordManager.java

```
Import the necessary libraries

Define a public class named PasswordManager that extends the JFrame class.

Define the following constants:
- databaseFile as "accounts.xml"
- rockYouFile as "rockyou.txt"
- imageFile as "icon.png"
- masterPasswordFile as "masterpassword.txt"

Define the following variables:
- masterPassword as a static string
- loginAttempts as an integer initialized to 0
- label as a JLabel
- passwordField as a JPasswordField

Define the main method:
- Set the look and feel to Windows theme
- Create a new instance of PasswordManager and set its properties
- Read the master password from the file and store it in the masterPassword variable
- If the master password does not exist, prompt the user to create one using the CreateMasterPassword class

Define the constructor for PasswordManager:
- Create a JLabel and add it to the frame
- Create a JPasswordField and add it to the frame
- Listen for password entry:
  - Retrieve entered password
  - Get the master password from the file
  - If the master password does not exist, show an error message
  - If the entered password is correct, decrypt the database, dispose the window, and open the PasswordManagerMenu
  - If the entered password is incorrect:
    - Increment loginAttempts
    - If loginAttempts is equal to 3, dispose the window and show an error message
    - Otherwise, show a message indicating the number of remaining attempts and clear the password field
```


Data Dictionary

I also wrote a data dictionary for the main Java file before beginning development, as it clarified the requirements, built on my understanding of how to proceed, and prepared me for any errors that I would encounter.

Variable/Function	Data type	Description
databaseFile	String	Name of the file that stores the user's account data
rockYouFile	String	Name of the file that contains the RockYou wordlist
imageFile	String	Name of the file that contains the icon
masterPasswordFile	String	Name of the file that stores the user's master password
masterPassword	String	User's master password
loginAttempts	int	Number of times the user has attempted to log in
label	JLabel	Label that prompts the user to enter their master password
passwordField	JPasswordField	The field where the user enters their master password
main()	void	Main method that initializes the Password Manager GUI and sets up the application
constructor	PasswordManager	Constructor method that creates the Password Manager GUI and sets up the password input field + listeners for password entry

Flowcharts

Creating a flowchart for the main Java file and the entire system would also be helpful.

Figure 12 - Flowchart of PasswordManager.java

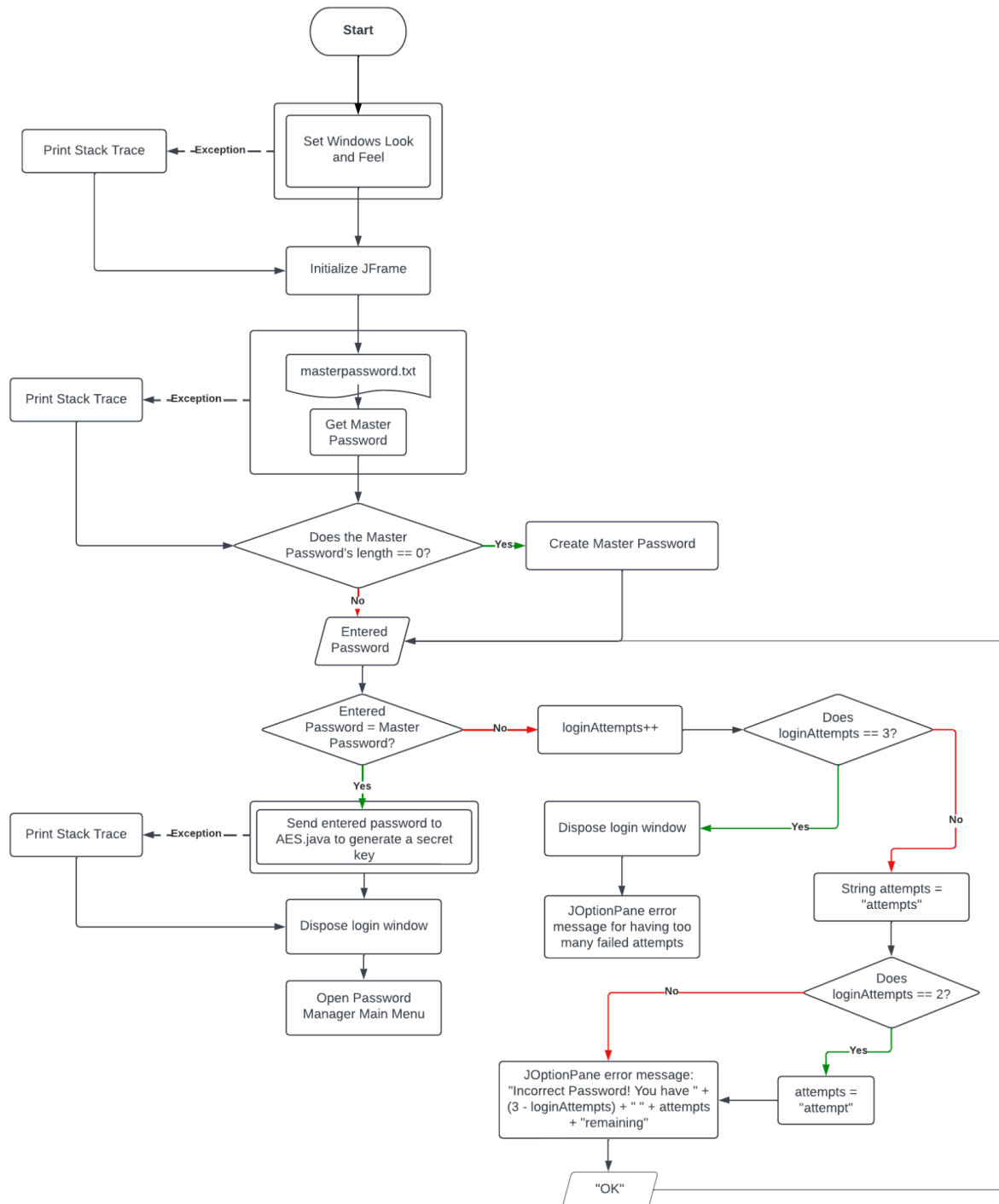
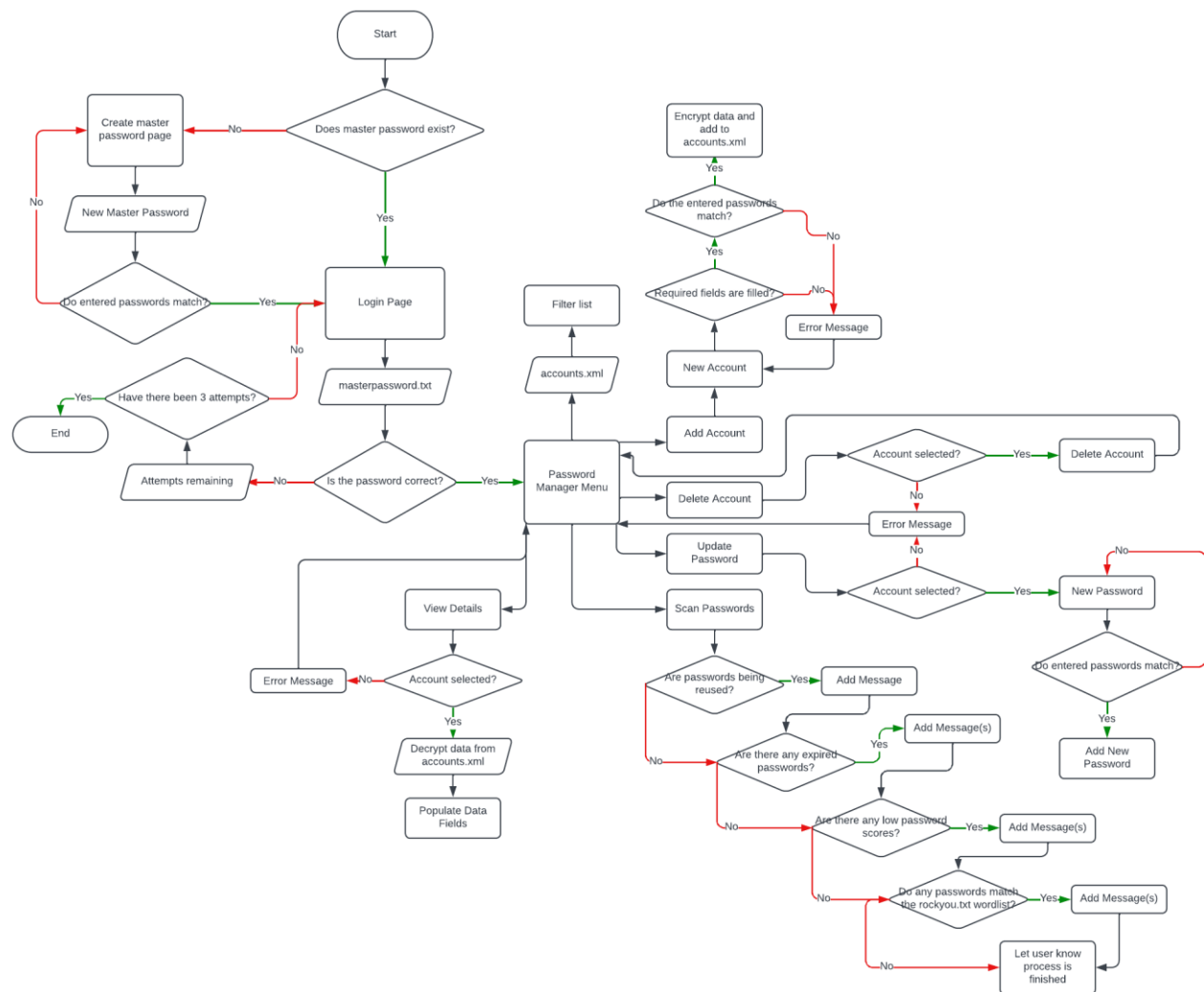


Figure 13 - System Flowchart



Versioning

Version #	Description	Success Criteria Met	Target Completion Date
1	GUI panels have been designed in NetBeans yet no functionality exists	--	12/23/22
2	Functionality for creating a master password and logging in to the main menu has been added, where users have a limited number of attempts	I, II	12/28/22
3	Main menu is fully working, with data being pulled from the XML file, sorted, and filtered. All buttons work.	III	12/29/22
4	Option to add an account fully works, and the list in the main menu is updated correctly. The option to remove an item works as well.	IV, V, VI	01/01/23
5	Information passed to the XML file is encrypted and stored using AES. The view details page correctly decrypts and displays the information.	VII	01/03/23
6	Beta program to give to the client	--	01/04/23
7	Polished program with changes based on client feedback	VIII	01/12/23

Functionality of each class

Class	Functionality
PasswordManager	<ul style="list-style-type: none">• Main class that creates the password manager
AES	<ul style="list-style-type: none">• Utility class for encrypting and decrypting data using AES algorithm
CreateMasterPassword	<ul style="list-style-type: none">• Shows a prompt for the user to create their master password, the first step
PasswordManagerMenu	<ul style="list-style-type: none">• The Password Manager's menu is accessed after the user logs in
AddAccount	<ul style="list-style-type: none">• Class called when the user clicks the 'Add Account' button• Provides a form for the user to add a new account to the XML file
GeneratePW	<ul style="list-style-type: none">• Class called when the user clicks the 'Generate' button next to the password field• Generates a strong + random password with a guaranteed mix of special characters and numbers
BCrypt	<ul style="list-style-type: none">• Third-party class used to hash the master password and check the entered master password against the hash
CheckPasswords	<ul style="list-style-type: none">• Class called when the user clicks the 'Scan Passwords' button• Checks for password reuse, expired passwords, low password scores, and common/well-known passwords.
DeleteAccount	<ul style="list-style-type: none">• Class called when the user clicks the 'Delete' button• Removes the selected account from the XML file
ViewDetails	<ul style="list-style-type: none">• Class called when the user clicks the 'View Details' button• Shows details of the selected account in a window by pulling from the XML file
AddView	<ul style="list-style-type: none">• Class called when the user clicks the 'View Details' button• Increments the use count in the XML file for the selected account
UpdatePassword	<ul style="list-style-type: none">• Class called when the user clicks the 'Update Password' button• Provides a prompt for the user to enter a new password for the account, and updates the XML file once submitted

Test Plan

- Tested in the Criterion D video
- Key: Normal Data, Erroneous Data

Success Criteria #	Test Description	Input	Expected Result
I	"Create Master Password" window shown when a master password is needed	Empty masterpassword.txt file. Master password already set.	"Create Master Password" window appears. "Create Master Password" window doesn't display.
I	Users can create a master password in the "Create Master Password" window	New password and confirm password fields filled with "master". The password fields don't match/are left blank.	BCrypt hashed password for "master" is saved in masterpassword.txt file. Error message displayed for the respective error.
II	Users can only log in with the correct password	Correct master password entered. Incorrect password entered.	Main menu window opens. Message displays showing remaining attempts. After 3 attempts, an error message displays and the program closes.
III	Account list is sorted in descending order with the most-used accounts at the top	User selects an account and clicks "View Details". "View Details" clicked without selecting an account.	View added to the account in the XML file and account ranks higher in the account list. Error message displayed, telling user to select an account.
IV	Users can add an account	User submits the "Add Account" form correctly. Form submitted with errors (non-matching password fields, required fields left blank, account name already exists).	Account added to account list. Error message displayed for the respective error.
IV	Users can remove an account	User selects an account and clicks "Delete". "Delete" clicked without selecting an account.	Confirmation message appears. If confirmed, the account is removed from the account list. Error message displayed, telling user to select an account
IV	Users can filter/search the account list	User types account name in "Filter List" text box.	Only accounts with matching names displayed in the account list.
V	Users can generate strong passwords	"Generate" clicked in the "Add Account" window with the	Password and confirm password fields populated with a very strong

		account name entered. "Generate" clicked in the "Add Account" window without the account name entered.	password (according to passwordmonster.com). Error message displayed, telling user to enter account name first.
VI	Users can select a renewal period for a password	User selects a radio button for 3, 6, 9, or 12 months and adds the account.	Renewal period and next renewal deadline added to XML file.
VI	"Account Details" window displays a message when a password is about to/has expired	"View Details" clicked and the account has a renewal period.	Renewal deadline and message about whether renewal is overdue, due, or in more than one week displayed.
VI	"Password Scan" window displays a message when a password is about to/has expired	"Scan Passwords" clicked and the account has a renewal period.	If a renewal deadline is within the next week, it will list that it is time to update the password. Will also list when a password reset is overdue.
VII	Data is encrypted using password-based encryption.	User logs in with the same master password used to create the accounts. User logs in with a different master password from what was used to create the accounts.	Proper access to encrypted account data from the XML file. "Error While Decrypting" message displayed.
VIII	Client satisfaction	Client uses the program for a week.	Positive reviews and extensibility list added to code and Appendix IV.

Word Count: 247