**PasswordVault**

CMSC 413 Intro to Cyber Security

Class Project

Jorge Zaquitzal

Pranaav Rao

Software Design

1.**Overview**:

The PasswordVault organizer is a Java built software with its own graphical user interface (GUI). As you can see the UML diagram is simple, however the power is in the encryption and decryption process. The encryption uses an 8-bit user key that the user creates and must maintain. It provides the simplest of usability with a very straightforward user interface. We factored out the bells and whistles of a generic program because a password organizer should be the simplest of programs to use. It requires an IDE that runs Java on any operating system Windows 8 or higher.

2.**System overview**:

The PasswordVault was designed with simplicity in mind. Using the robust Java encryption libraries, the PasswordVault simply takes in user input to either create, add, view, or delete contents of a password file then encrypts the entire file.

3.**Design Platform**:

This program was built using Eclipse Java IDE that was installed on a Windows 10 machine.

4. **Future Contingencies**:

It is possible the Java Libraries change so must the code when implementing encryption. Also upgrade encryption to AES for more security.

5.**Point of Contact**

School or Institution: Virginia Commonwealth University School of Engineering

Project Advisor: Carol Fung - cfung@vcu.edu

Student: Jorge Zaquitzal - zaquitzalja@vcu.edu

Student: Pranaav Rao - raop@vcu.edu

5.**Project References**

We used a multitude of references to learn the encryption and decryption process

Links:

<https://www.tutorialspoint.com/java_cryptography/java_cryptography_encrypting_data.htm>

<https://docs.oracle.com/javase/7/docs/api/javax/crypto/Cipher.html>

<https://docs.oracle.com/javase/8/docs/api/javax/crypto/spec/DESKeySpec.html>

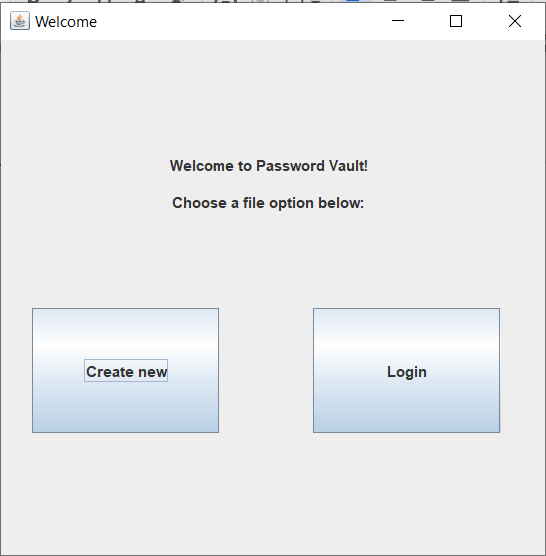
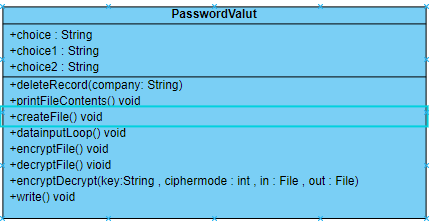
<https://docs.oracle.com/javase/7/docs/api/javax/crypto/SecretKeyFactory.html>

<https://docs.oracle.com/cd/E17802_01/j2se/j2se/1.5.0/jcp/beta1/apidiffs/java/security/SecureRandom.html>

<https://docs.oracle.com/javase/tutorial/uiswing/>

6. **Detailed System Software Architecture**

The system is built using the Java programming language. It consists of a main class that creates the GUI (graphical user interface) using the swing java libraries and the PasswordVault Class that implements the logic under the hood. Users enter options to create or add, view or delete to an existing file of contents which is then added to a unsecure password.txt that is stored in the directory of the application workspace. Once the user is done with the program, the entire file gets encrypted. Password.txt is then created into encrypted\_password.txt that is saved to the same directory as before and password.txt is deleted. This system is designed to run on any OS that can support a Java IDE. Below is the UML diagram breaking down the PasswordVault Class.



*deleteRecords()*

This method takes in a website name i.e. Netflix, and the logic searches the file for this website and recreates the file without this website specifically.

\*Future implementation would be number based positions to avoid deleting websites of the same name\*

*printFileContents()*

This method prints the file character at a time. ‘\* Future implementation would be read in file by line to save memory\*

*createFile()*

This method creates a new encrypted\_password.txt which stores the encrypted data after encryption using java.io.File libraries.

*dataInputLoop()*

This is the main loop that cycles to have the user enter a website, username, and password.

This is done by creating a file that is called password.txt and adding data to that file.

*encryptFile()*

This method is specifically called with parameter “key” to encrypt the file that the username and passwords will be saved to. This calls the encryptDecrypt method that takes in 3 parameters which are the key the user creates, the encrypt mode, the plaintext file, and the name of the encrypted file that data will be written to.

*decryptFile()*

This method is specifically called with parameter “key” to decrypt the file that the username and passwords will be saved to. This calls the encryptDecrypt method that takes in 3 parameters which are the key the user creates, the decrypt mode, the encrypted file, and the name of the plaintext file that will be decrypted.

*encrypted file()*

This is the meat and bones of encryption and decryption. This method takes in the key the user creates, whichever cipher mode chosen (encrypt or decrypt), the file going in and the file it will be written to.

This creates in instance of a cipher that uses “DES/ECB/PKCS5Padding”’

Detailed in MS Visual Studio “In order to create a Cipher object, the application calls the Cipher's getInstance method, and passes the name of the requested transformation to it. Optionally, the name of a provider may be specified.

A transformation is a string that describes the operation (or set of operations) to be performed on the given input, to produce some output. A transformation always includes the name of a cryptographic algorithm (e.g., AES), and may be followed by a feedback mode and padding scheme.

A transformation is of the form:

"algorithm/mode/padding" or

"algorithm"

(in the latter case, provider-specific default values for the mode and padding scheme are used). For example, the following is a valid transformation:

ex:

Cipher c = Cipher.getInstance("AES/CBC/PKCS5Padding"); ”

Then the cipher is initialized with parameters that take in, the mode (encrypt or decrypt), the secret key (which was converted into an array of bytes which then was converted into deskKeySpec which is then encrypted using the DES algorithm, stored into a SecretKey variable, then a source of randomness using SHA1PRNG with SecureRandom.

A CipherStream is then created that takes in the file its writing to and the cipher itself.

Then written to a file.

*Write()*

Writes to file.

7.**Detailed components of encryption**

In this program we use the “DES “to encrypt and decrypt the password.txt file to then make it a secure\_password.txt file. The DES algorithm is a symmetric key algorithm that is used to encrypt and decrypt electronic data. It uses the same key for both processes. This is known as the block cipher. DES uses a block size of 64 bits and uses a key length of 56 bits. Link Java oracle site in references for additional information on methods within source code.

If a user creates a file then a new master key passcode is made. If the user is returning to modify the file, then all the user needs is to re-enter the master passcode. “

User Manual

Installation:

Download any IDE (integrated development environment) that supports Java onto a Windows machine or Apple computer.

For this project we used Eclipse and MS Visual Studio for Java

Download source code from:

<https://github.com/pranaavr/PasswordVault.git>

Download all the .Java files which are:

*Gui.java*

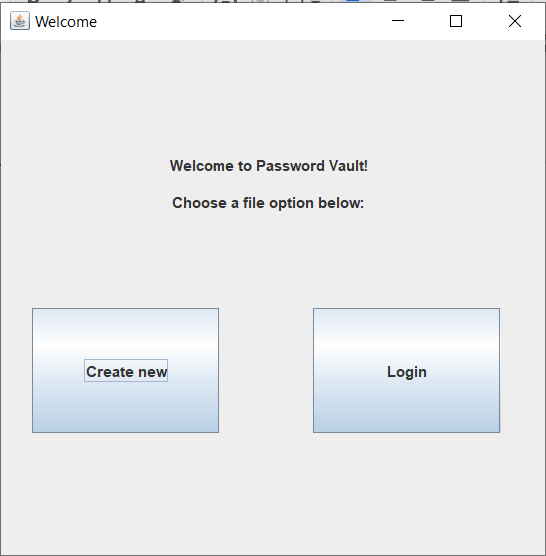
*Main.java*

*PasswordVault.java*

Then all that is left is to run the Main class and you’re all set.

When the program is run, the GUI will be immediately visible displaying the Welcome page for this program.

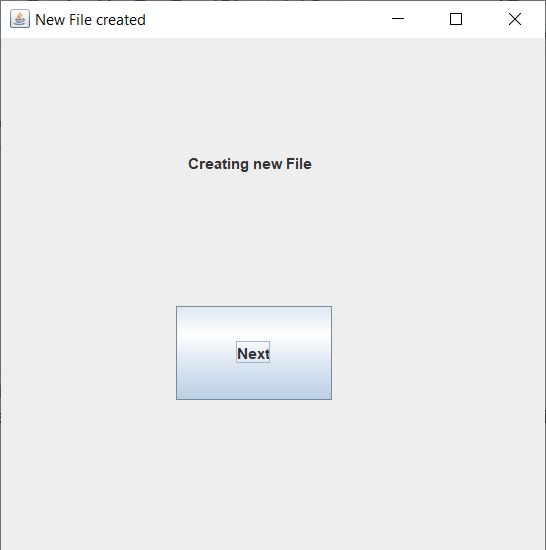
Welcome



Click “Create new” if this is the first time using the program. Click “Login” if otherwise.

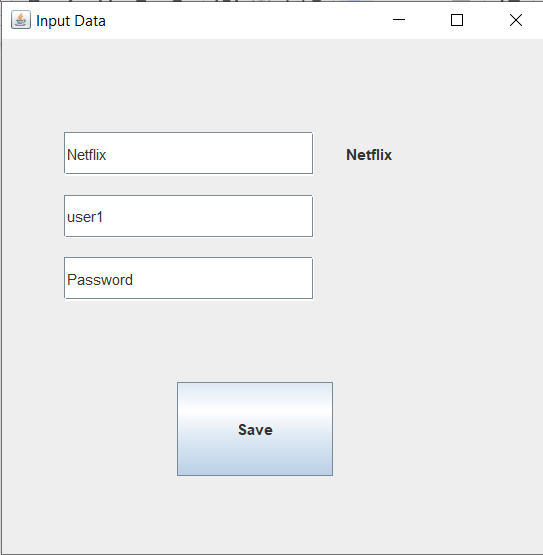
\*\*NOTE: Clicking “Create new” if an existing password file exists will result in that file being overwritten and all previous information being lost :(

BRANCH: “Create new”



Congratulations! A new file has been created. This text file will store all the desired credentials for the PasswordVault program. Click “Next”

Add

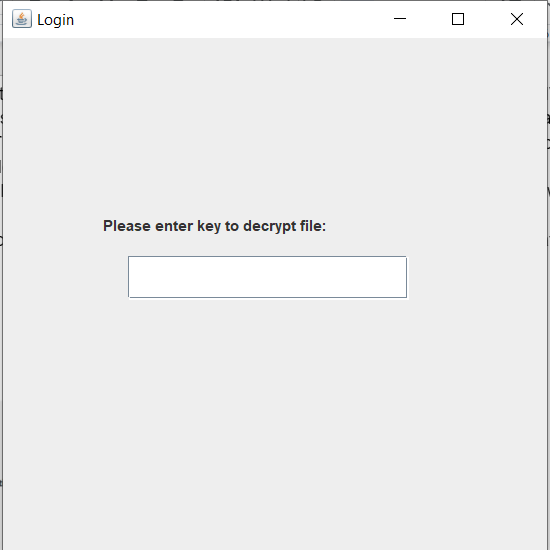


This is where the first record will be created to be stored securely in the PasswordVault! The first field is titled “Website” and acts as a label for which account login information is to be saved. The second field is the associated username and the third is the associated password. Delete existing field text and type in desired text.

IMPORTANT: Press ‘enter’ on keyboard to save input. If this is not done, no input will be saved

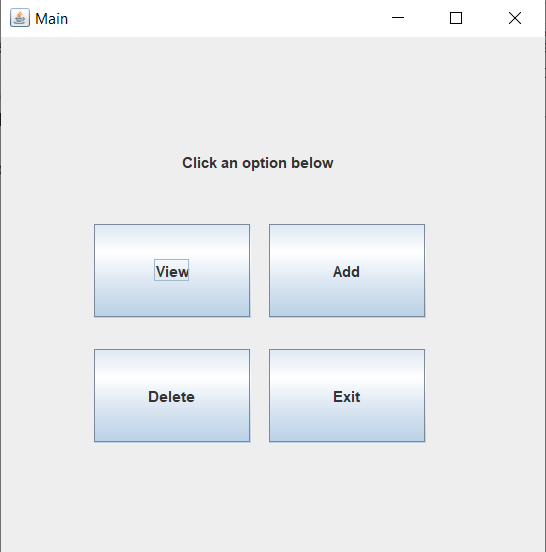
Click “Save” to continue to the Main page. Inputs will automatically be saved to data file

BRANCH: “Login”



Enter previously created key exactly to re-enter the PasswordVault. Press ‘enter’ on keyboard to be taken to the Main page

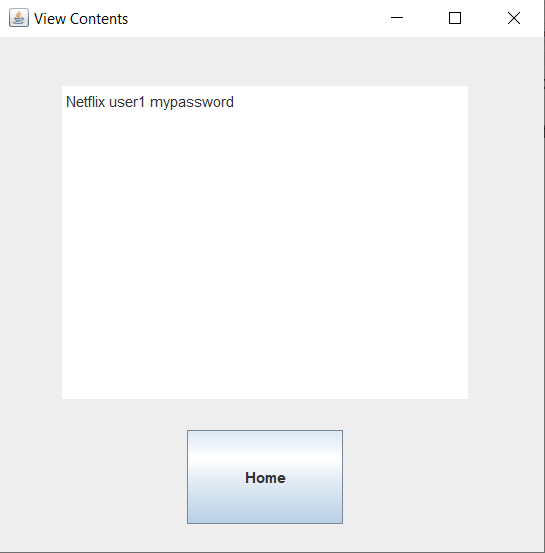
Main



Here are all the options provided to manipulate the data stored in your PasswordVault. Click “View” to view listed credentials, “Add” to add more credentials, or “Delete” to delete an existing credential from the PasswordVault. Clicking “Exit” will open the “Create Key” window.

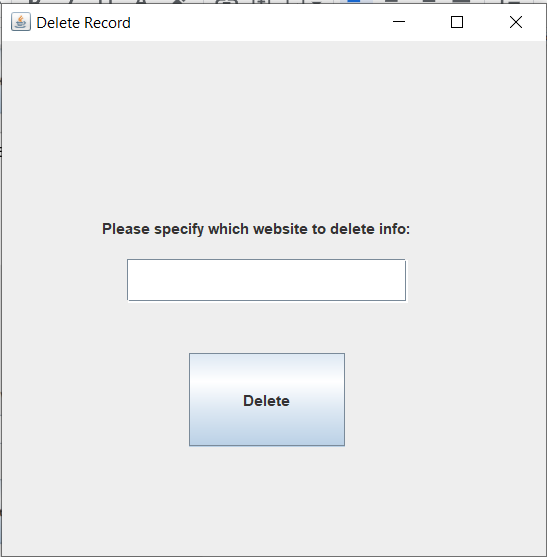
\*\*NOTE: Scroll up to (BRANCH: “Create new”) to learn more about “Add”. More info on other options listed below

View



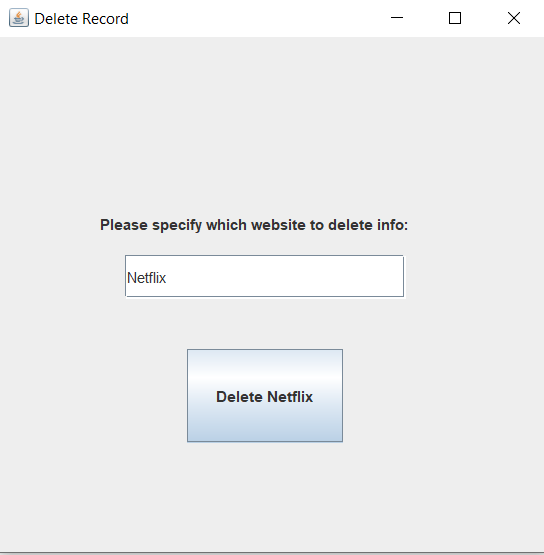
Here you can view the existing credentials in your PasswordVault. Click “Home” to return to the Options page.

Delete

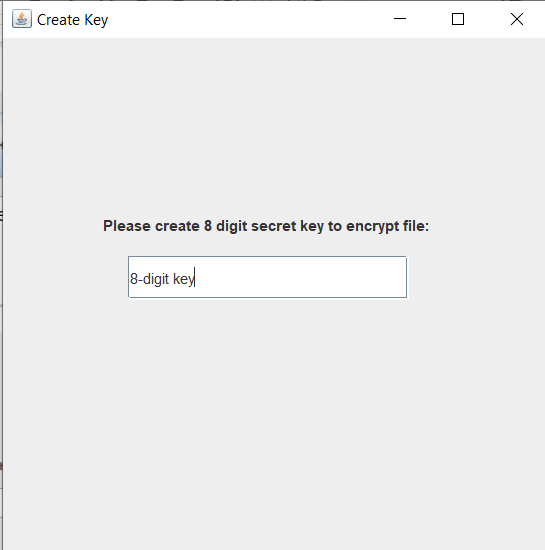


This page will give you the option to delete an existing record. It is based on the website or label of the record so exact input is required. Click “Delete \_\_\_” to delete the record which will then return to the Home page.

IMPORTANT: press ‘enter’ on keyboard after typing input into the field. This will result in the input printing on the button as shown below. This is how you know the data will be deleted.



Create Key



Before exiting the program, a key must be entered to validate access into the encrypted text file. An 8-digit key is recommended for security purposes. After choosing and typing the key, press ‘enter’ on the keyboard to exit and encrypt the text file.