Problem Statement:

Internet sites often preach about the safest online password practices. Don't use any personal information in your passwords. Don't use a word from the dictionary as a full password. Use alphabetic and numeric characters. Use different passwords for every account you have. Change your passwords regularly and frequently. All of these imperatives provide great advice. However, this advice neglects how difficult it can be memorizing every password for every account a user owns. Wouldn't it be easier just to remember one, extremely long, secure password?

Objective of the System:

Nyan Security's Password Manager aims to accomplish that goal. Nyan Security's Password Manager will store all your account usernames and passwords in a local database on your most frequently used machine. The database is encrypted, accessible only by one master password, which can be any length up to 64 characters. NSPM will also provide functionality to generate a [pseudo]random password of a user-specified number of characters. This function can be used when registering an account on a site.

Nyan Security's Password Manager will also provide a graphical user interface framework through which users may access their database. It will also support multiple profiles, each with an unlimited number of account descriptions and usernames/passwords. The framework will render a search function, through which users may find the exact account username/password for which they are looking. NSPM also supports add and remove features, as well as copy and paste functionality.

Other similar applications:

Five popular open source password managers include Keypass, Clipperz, Yadabyte, Password Gorilla, and Universal Password Manager. These programs present a variety of features, but the all of them include the basics: encryption of passwords, access through a master password and portability (possible to export and import files). Additional features include Clipperz's web-based design that automatically stores the account and password combination within its database once the user logs into an account. The other four programs are run locally and work on a variety of operating systems, all presented through a GUI. Both Keypass and Clipperz offer password generation and Keypass even shares with you how many bits of encryption your passwords have. Password Gorilla and Keypass provide the ability to organize passwords through grouping. Other useful features include copying a password to clipboard through the use of a hotkey as found in Password Gorilla and Universal Password Manager. Keypass and Universal Password Manager both offer mobile apps. Our project's goal is not so much to provide an innovative solution to password management, but to create a product that incorporates many of the great features found in existing products such as password generation, effective encryption, portability, grouping, password rating, and copy/paste functions.

Technologies and CS Concepts we must learn to complete the project:

In C++, there are two easy ways to implement encryption and both use the logical XOR (exclusive OR). All logical operators do bitwise operations on the individual bits in an integer data type. For example:

This example encrypts ‘s’ using ‘x’ as the encryption key.

Char ‘s’ (int value 115)	01110011
Char ‘x’ (int value 120)	01111000 XOR
	00001011 = 11

These values are stored in an array of chars that represent the encrypted string. To decrypt the byte, you use the same encryption key. It does not take much to make this algorithm more complex: simple mathematical operations can serve to obfuscate the original identity of the to-be-encrypted number even more.
If the user passwords are stored in a database, most database managers have built-in encryption and decryption functions. These functions are inherently more complex and secure than the basic encryption algorithms discussed above, and would not be implemented on our side.

A possible encryption algorithm that is implementable is as follows:
1. Upon creation of a new user account, the program prompts the user to input a series of random letters for use as his/her encryption and decryption key. We can hash this key or encrypt it using a different method.
2. Next, whatever passwords the user enters will be encrypted using the aforementioned key and stored in a database. When the user seeks to retrieve a password, the key will be decrypted and then used in the decryption process for the password.

Graphical User Interface Development

In order to implement the graphical user interface portion of the project we must become familiar with some additional libraries external to C++. Unlike some other languages C++ does not have built-in support for creating graphical front ends for programs. There are several options to choose from when picking external libraries for GUI development with C++. Of these libraries, Qt seems to be the best option for multiple reasons: First, Qt is cross-platform compatible; that is, the only requirement is that the source be compiled on the desired OS before it can be used. Qt is also open source so it benefits from a very large community of developers. It also has very robust documentation that covers a very large number of examples as well as many tutorials available online. Qt also has its own integrated development environment (IDE) which caters to the construction of graphical interfaces. This IDE includes tools that allow the user to create interfaces by using click-and-drag type tools.

Other Possible Applications of the System:

Given enough time, the password manager program can have several extra applications that users may utilize. If possible, we could use the NPAPI system to create a browser plugin for our program, allowing users to access our work from internet applications (e.g. Chrome). Many problems with passwords nowadays are that hackers are attempting to break them and steal other people’s information and data. There will be a great use for data wiping in this program, which would discard the original user’s data should an unauthorized user attempt to break the master password for the original’s accounts. A mobile application for smartphones should also be feasible, allowing for the user to travel without being required to bring the program itself. Another way to help protect the user is to give the option for a multitude of differing encryption algorithms. The user will have the option to choose one among those algorithms in order to protect their password (whichever one they feel is the safest). Last, but not least, the program can be given the ability to mass export or import data, expanding the area of work in which the user can perform personal tasks.