Python Programming

Assignment 2- Password Manager software

<u>Password Manager software:</u>

```
import random
import string

class PasswordManager:
    passwords = {}

    def generate_password(length=12):
        characters = string.ascii_letters + string.digits + string.punctuation
        generated_password = ''.join(random.choice(characters) for _ in range(length))
        return generated_password

    def encrypt_password(password):
        key = ''.join(random.choice(string.ascii_letters) for _ in range(16))
        encrypted_password = ''.join(chr(ord(char) + 3) for char in password)
        return key, encrypted_password

    def decrypt_password(key, encrypted_password):
        decrypted_password = ''.join(chr(ord(char) - 3) for char in encrypted_password)
        return decrypted_password

    def save_password(website, username, password):
        key, encrypted_password = PasswordManager.encrypt_password(password)
        PasswordManager.passwords[website] = {'username': username, 'key': key,
'encrypted_password': encrypted_password}

    def search_saved_password(website):
        if website in PasswordManager.passwords:
            details = PasswordManager.passwords[website]
            decrypted_password = PasswordManager.decrypt_password(details['key'],
details['encrypted_password'])
            return f"Username: {details['username']}, Password: {decrypted_password}"
        else:
            return "Password not found for the given website."
```

```
generated_password = PasswordManager.generate_password()
print(f"Generated Password: {generated_password}")

PasswordManager.save_password("example.com", "user123",
generated_password)

search_result = PasswordManager.search_saved_password("example.com")
print(search_result)
```

<u>**Password Manager Software Description and Summary:**</u>

The Password Manager software created with the help of Python is a helpful tool for keeping your website passwords safe. It uses basic programming to make it easy to generate, save, and find your passwords securely.

Approach Used:

- I used the ‘random’ module and string to generate strong and random passwords. The length of the password is customizable, providing flexibility to users.

- I employed a basic encryption technique using character shifting. The ‘encrypt_password’ function shifts each character in the password by three positions, ensuring a necessary level of security. The ‘decrypt_password’ function reverses this process.

- I stored passwords and relevant details (username, key, encrypted password) in a class variable dictionary (‘passwords’). The ‘save_password’ method adds a new entry, and the ‘search_saved_password’ method retrieves the stored information.

Future scope:

- Soon, you might need a special key or a fingerprint to make sure only you can access your passwords.

- There's a plan to store your passwords in a safe place that lasts even if you close the tool. This makes it easier to manage your passwords over time.

- The tool might tell you if your password is very strong or if it needs to be a bit stronger. It's like having guidance for making passwords.

- Another plan is to add extra checks to make sure it's you trying to access your passwords. This adds an extra layer of safety.
- In the future, you'll be able to use your passwords on different devices without any trouble. So, you can access your passwords whenever and wherever you need them.

References:

W3Schools