

Strong Pass: Secure Password Generator & Vault

Students: Ulyana Clarke, Marcell Placencia, Christian Colon, Zeyon Charles
Instructor: Masoud Sadjadi, Florida International University

PROBLEM

- ❖ Most built-in password generators lock users into specific ecosystems
- ❖ Users have little to no control over password composition or character types
- ❖ Weak or reused passwords remain the #1 cause of account compromise
- ❖ No easy way to check if a generated password has been previously breached
- ❖ Storing secure passwords remain a challenge for everyday users

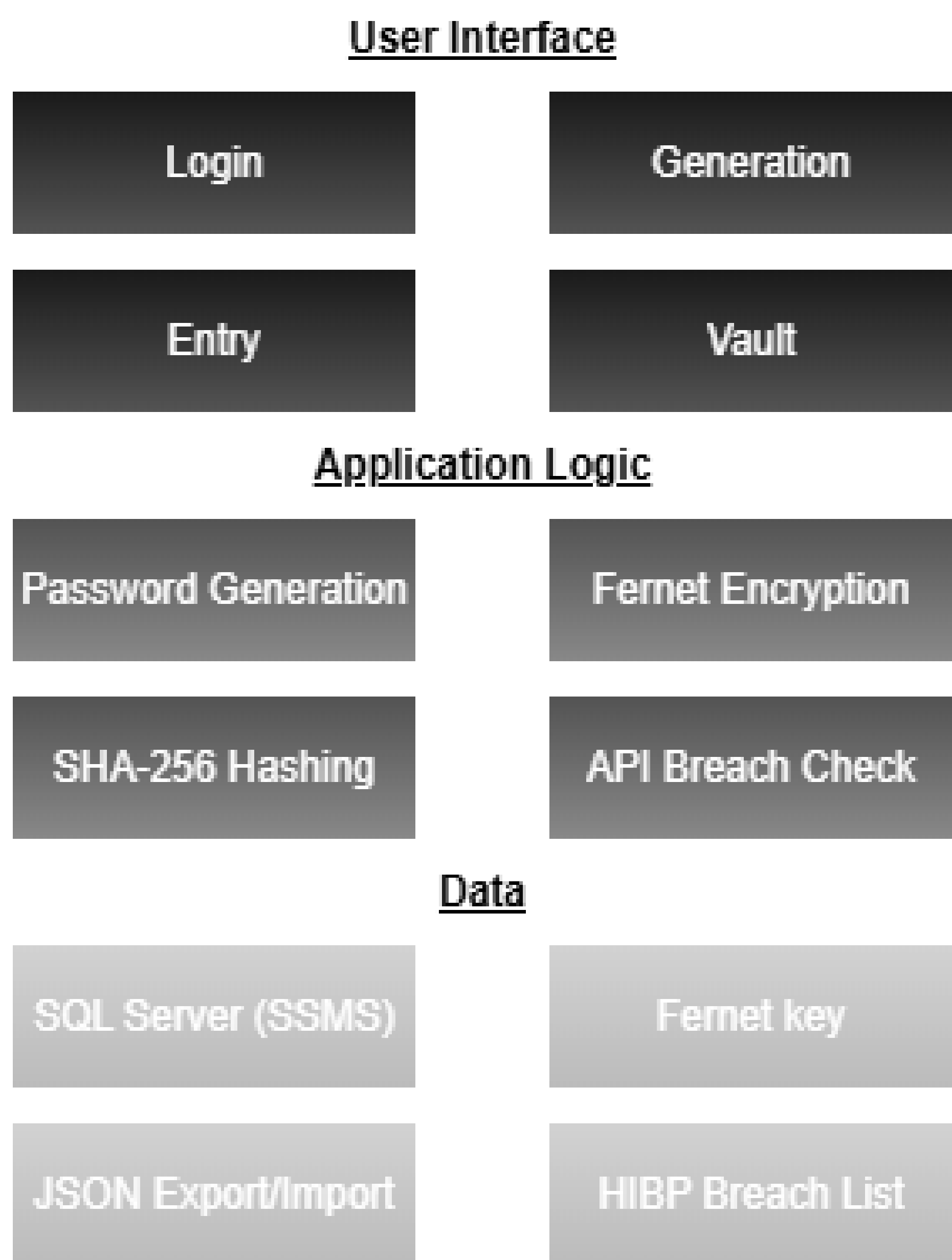
SOLUTION

- ❖ Generates strong, customizable passwords
- ❖ Stores and manages passwords for any application
- ❖ Profile-specific password vaults
- ❖ Breach detection via HavelBeenPwned API
- ❖ Strength meter for every generated password
- ❖ Save credentials to vault with one click
- ❖ Installed locally on your system for easy, secure access

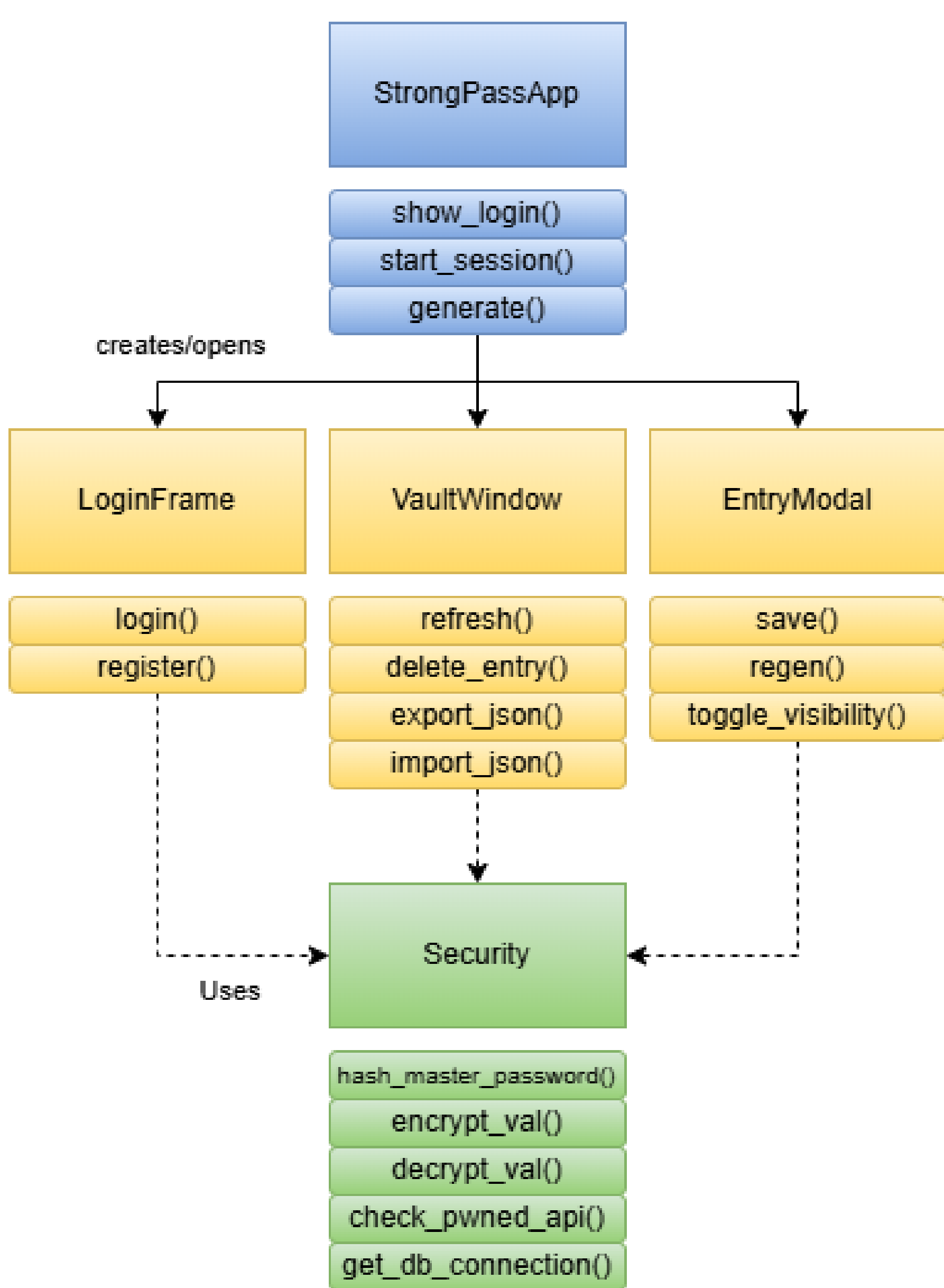
SUMMARY

Strong Pass is a Python-based password manager designed to solve the gaps that other password tools fail to address, which is customization, usability, and security. Beyond generating strong, user-specified passwords, the app features a Fernet-encrypted credential vault, master account authentication, and real-time breach detection via the HavelBeenPwned API. It is simple to understand yet built to NIST security standards. Strong Pass runs on any system with Python installed, with no cloud, no browser, and no vendor lock-in.

SYSTEM DESIGN



OBJECT DESIGN



IMPLEMENTATION

```
def evaluate_strength(self, pwd):  
    """Returns a score from 0.0 to 1.0 based on password complexity."""  
    # Simple algorithm to give a rough "score" of how good the password is  
    score = 0.0  
    if len(pwd) >= 8: score += 0.2  
    if len(pwd) >= 12: score += 0.3  
    if any(c.islower() for c in pwd): score += 0.1  
    if any(c.isupper() for c in pwd): score += 0.1  
    if any(c.isdigit() for c in pwd): score += 0.15  
    if any(c in "!@#$%^&*()-_+=[]{}|;:,<.>?" for c in pwd): score += 0.15  
    return min(score, 1.0)  
  
def logic_gen(self, l, n, s, word):  
    # Build a big list of random characters based on what the user asked for  
    pool = random.choices(string.ascii_letters, k=l) + \  
          random.choices(string.digits, k=n) + \  
          random.choices("!@#$%^&*()-_+=[]{}|;:,<.>?", k=s)  
    # Throw the custom keyword in there if they typed one  
    if word: pool.append(word.upper() if random.random() > 0.5 else word.lower())  
    # Mix it all up so it's random  
    random.shuffle(pool)  
    return "".join(pool)
```

CURRENT SYSTEM

Strong Pass is a fully functional desktop password manager currently running on Python 3.13 with a CustomTkinter dark-mode interface. The following features are live and operational:

- Master account login with salted SHA-256 authentication
- Customizable password generation (letters, numbers, symbols, keyword)
- 7-tier password strength scoring with color-coded feedback
- Real-time breach detection via HavelBeenPwned API (k-Anonymity)
- Fernet-encrypted credential vault backed by SQL Server Express
- Copy-to-clipboard, undo last password, regenerate
- Import and export vault as encrypted JSON

The application is installed locally, requires no internet connection for core generation, and stores no credentials in plaintext at any layer.

REQUIREMENTS

Python 3.13
TKinter & CustomTKinter
SQL: Server
Cryptography.fernet
HavelBeenPwned API
Hashlib (password hashing)
Pyodbc (database connection)
GitHub

REFERENCES

[1] NIST. 2017. *Digital Identity Guidelines: Authentication and Lifecycle Management*. SP 800-63B. csrc.nist.gov

[2] OWASP Foundation. 2024. *Password Storage Cheat Sheet*. cheatsheetseries.owasp.org

[4] Verizon. 2025. *Data Breach Investigations Report*. verizon.com/dbir

[8] Troy Hunt. 2023. *Have I Been Pwned API v3 — k-Anonymity Model*. haveibeenpwned.com

[11] Python Cryptography Authority. 2024. *Fernet Symmetric Encryption*. cryptography.io

[12] Microsoft. 2024. *SQL Server Express Documentation*. microsoft.com/sql-server