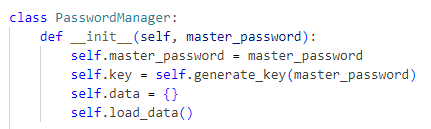
# Dokumentation Password-Manager Gruppenarbeit Python

Das Programm Password Manager dient zur sicheren Verwaltung und Speicherung von Passwörtern für verschiedene Websites. Es enthält Funktionen zum Hinzufügen, Anzeigen, Bearbeiten und Löschen von Passworteinträgen.

Mitglieder: Tobais Stoppelkamp, Christopher Gorshert, David Prinz, Lion Wicki

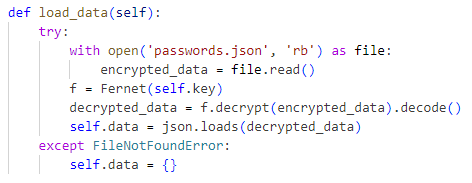
# Utility-Functions in main.py:



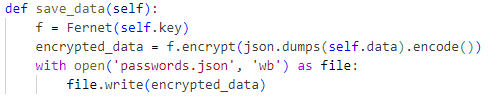
* Initialisiert den Password Manager mit einem Master-Passwort. Das Master-Passwort wird verwendet, um Daten aus der verschlüsselten JSON-Datei zu laden.



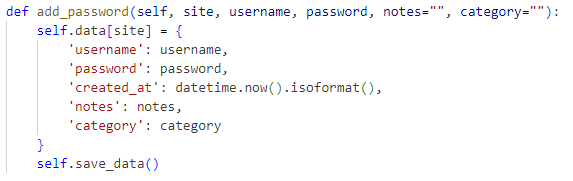
* Generiert einen Key, der aus dem SHA- 256 Hash des angegebenen Passwortes abgeleitet wird



* Lädt die gespeicherten Passwortdaten, entschlüsselt mit dem Master-Passwort, aus der passwords.json



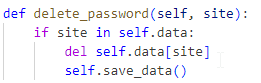
* Verschlüsselt die aktuellen Passwortdaten und speichert sie in passwords.json



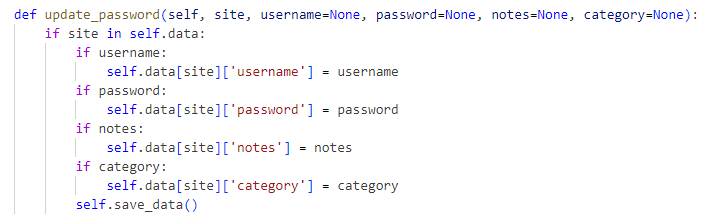
* Fügt neue Passworteinträge für bestimmte Websites hinzu, inklusive der gegebenen Parameter



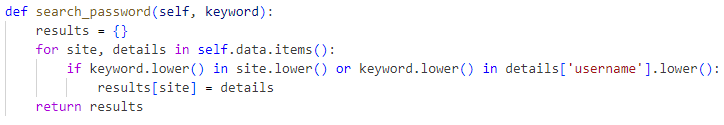
* Gibt die Passwortdaten einer bestimmten Website zurück



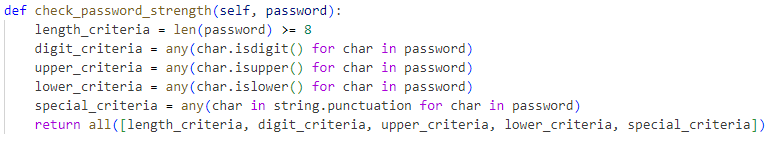
* Löscht die Daten einer bestimmten Website



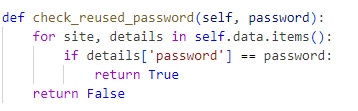
* Aktualisiert einen bestimmten Eintrag, jeder Parameter, der auf None gesetzt wird, bleibt unverändert.



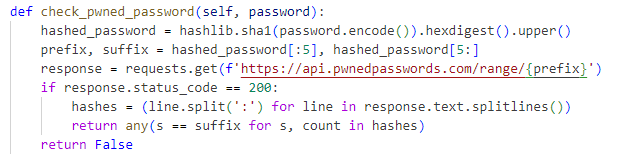
* Sucht nach bestimmten Passworteinträgen anhand des Benutzernamens oder der Website und gibt sie aus



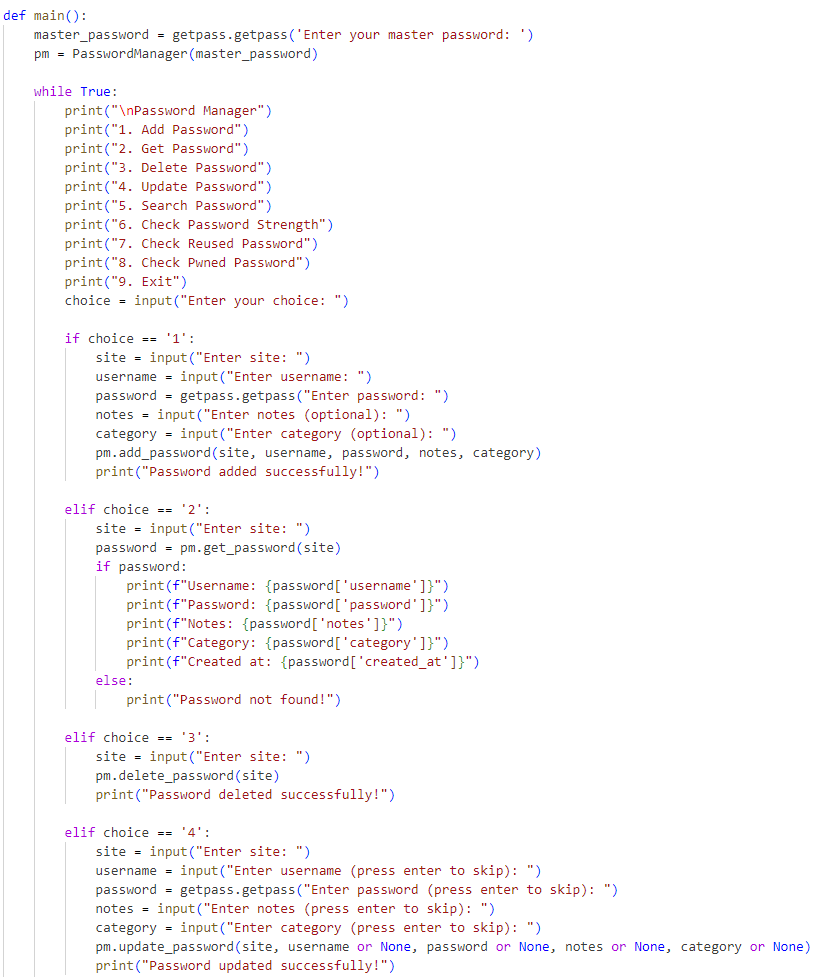
* Überprüft das gegebene Passwort anhand von gängigen Kriterien, wie z.b. Lowercase, Uppercase, Zahlen bzw. Sonderzeichen

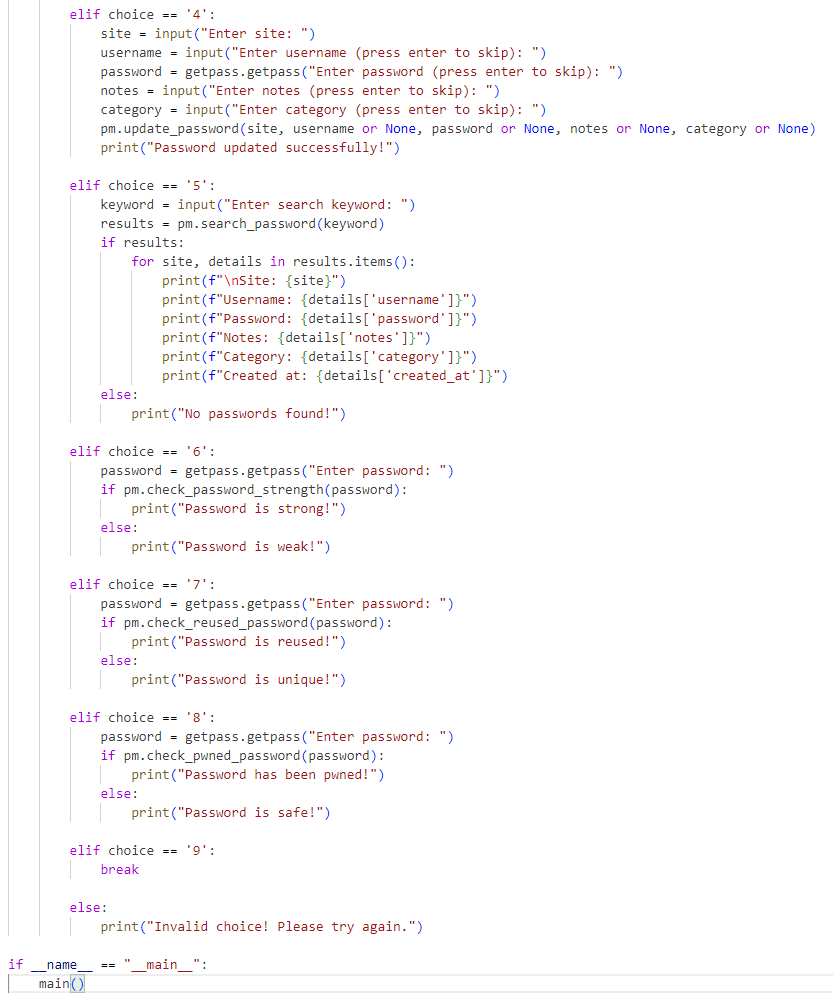


* Überprüft, ob das gegebene Passwort bereits für einen gespeicherten Eintrag verwendet wurde

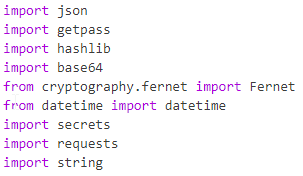


* Überprüft anhand der Pwned-API, ob das gegebene Passwort kompromittiert wurde

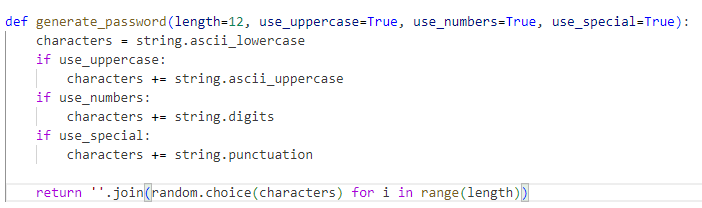




* Eigentliche Main.py, die auf die davor erklärten Functions verweist bzw. diese ausführt und den User durch das Programm führt



# Utility-Functions in example.py:



* Erzeugt ein zufälliges Beispielpasswort

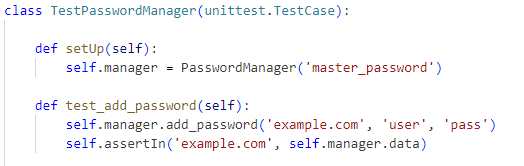


# Utility-Functions in test\_example.py:

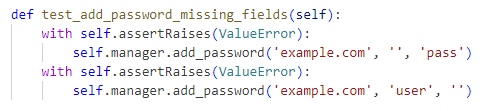
* Stellt sicher, dass die Funktion generate\_password richtig funktioniert – Überprüft ob die Anforderungen an Länge, Großbuchstaben, Ziffern und Sonderzeichen erfüllt



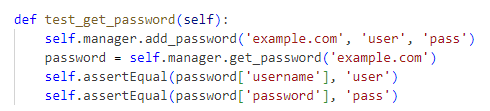
# Utility-Functions in test\_main.py:



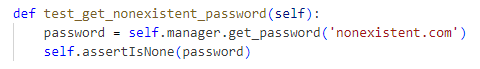
* Initialisiert eine Instanz mit dem Master-password vor jedem Test
* Verifiziert, ob ein Passwort dem Manager hinzugefügt werden kann



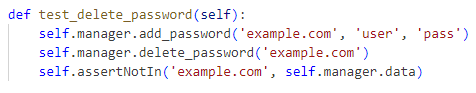
* Stellt sicher, dass eine Fehlermeldung gezeigt wird, wenn die Felder „username“ oder „password“ leer sind



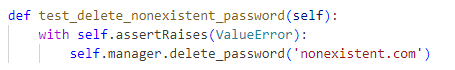
* Testet, ob ein existierendes Passwort aus dem Manager geholt werden kann



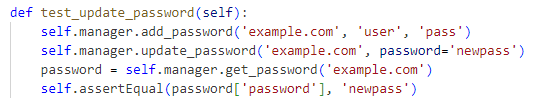
* Testet das Verhalten, wenn ein Passwort angefordert wird, das nicht existiert



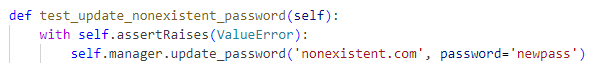
* Stellt sicher, dass ein Password gelöscht werden kann



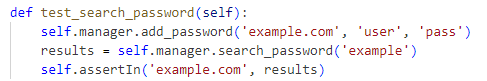
* Testet, ob es eine Fehlermeldung gibt, wenn versucht wird ein nicht existierendes Passwort zu löschen



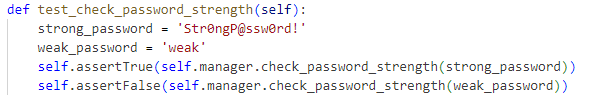
* Stellt sicher, dass ein Passwort verändert werden kann



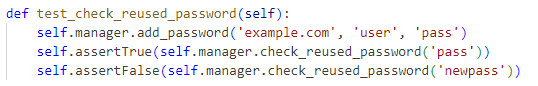
* Prüft, ob es eine Fehlermeldung gibt, wenn versucht wird ein nicht existierendes Passwort zu verändern



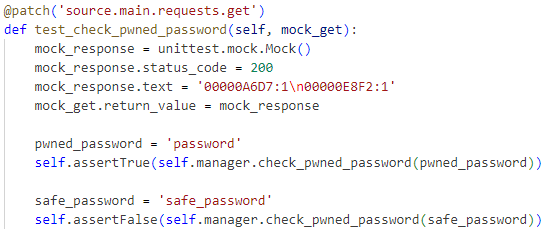
* Testet die Suchfunktion



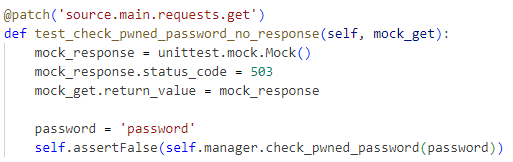
* Prüft die „check-Passwort“ Funktion



* Verifiziert, ob der Manager bereits verwendete Passwörter entdecken kann



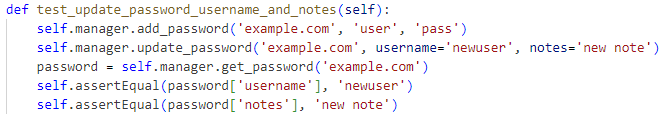
* Testet die „check\_pwned“ Funktion



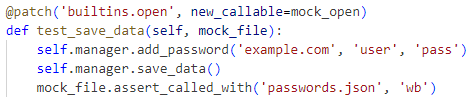
* Prüft das Verhalten, wenn die Pwned-API keine Rückmeldung liefert



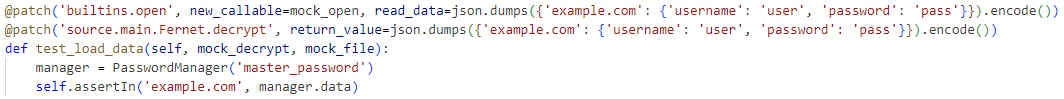
* Testet ein Passwort inklusive Notizen und einer Kategorie hinzuzufügen



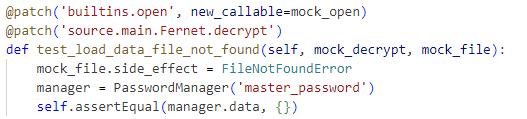
* Stellt sicher, dass ein Passwort mit Notizen und einer Kategorie verändert werden kann



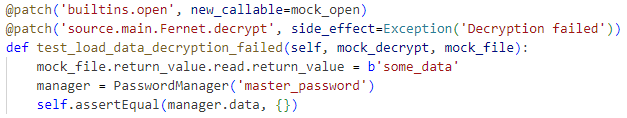
* Stellt sicher, dass die Daten in der Json-Datei gespeichert werden können



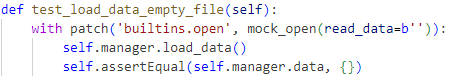
* Prüft, ob Daten aus der Json-Datei geladen und entschlüsselt werden können



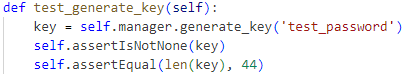
* Prüft das Verhalten bei fehlenden Daten



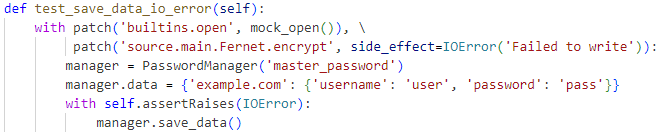
* Prüft das Verhalten bei einer fehlerhaften Entschlüsselung



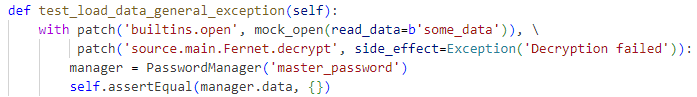
* Prüft das Verhalten bei einer leeren Datei



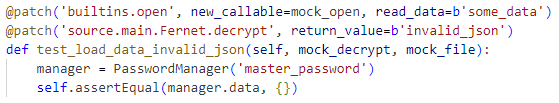
* Testet, ob ein Schlüssel generiert werden kann



* Stellt sicher, dass der Manager mit I/O- Fehler umgehen kann



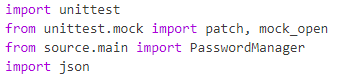
* Verifiziert das Verhalten bei allgemeinen Fehlern



* Stellt sicher, dass der Manager mit invaliden Json-Daten umgehen kann



* Startet alle Tests, wenn das Script ausgeführt wird



# Verwendete Biblotheken:

pylint=3.2.3

coverage=7.5.3

mypy=1.10.0

cryptography=42.0.0