Dokumentation des Password Managers

Inhaltsverzeichnis

Aufbau und Funktionen

Der Password Manager besitzt zwei Hauptfenster:

1. Loginmenü
2. Hauptmenü

In allen Fenstern kann mit den Pfeiltasten durch die Optionen navigiert werden. Mit Enter wird eine Option bestätigt.

Im Loginmenü kann das Programm beendet, ein neuer Benutzer registriert werden und sich ein bestehender Nutzer einloggen.

Wurde ein Benutzer registriert und konnte nun dieser sich erfolgreich einloggen wird der Nutzer in das Funktionsmenu des Password Manager weitergeleitet. Dort befinden sich 10 Funktionen:

1. Hinzufügen eines Eintrags
2. Generieren eines Passwords
3. Editieren eines Eintrags
4. Löschen eines Eintrags
5. Suchen eines Eintrags
6. Liste aller Einträge und deren Inhalt
7. Laden von einer json Datei mit Einträgen
8. Speichern der Einträge in eine json datei
9. Optionen -> Bisher nur aktivieren der 2-Faktor-Authentifizierung
10. Ausloggen und zurück zum Hauptmenu

Die wichtigen Benutzerinformationen werden in einer json Datei gespeichert, dort ist das Masterpassword als sha256 Hash gespeichert. Im json Format sind die Einträge verschlüsselt mit AES256 in der entsprechenden *entries.enc* Datei gespeichert.

Bibliotheken

requests==2.32.3
cryptography==43.0.0
qrcode==7.4.2
pyotp==2.9.0

Architektur

Vor dem starten:

- Zuerst muss die Umgebungsvariable PYTHONPATH durch die ***setup.sh*** Datei gesetzt werden.

GUI:

- Um das Programm zu starten muss die ***main.py*** Datei ausgeführt werden.
- Für die Terminal GUI wird die menu.py Datei zur Initialisierung und Verwaltung der ***curses*** Bibliothek aufgerufen.

Benutzer Management:

- Das Modul ***userManagement.py*** dient zur Erstellung neuer Benutzer und ebenso zur Validierung von Login Versuchen.

Datei Speicherung und Verschlüsselung:

- Das Modul ***diskManagement.py*** ist für die Speicherung und Ladens der Dateien zuständig.
- Mit dem Modul ***cryptographyManager.py*** werden die Einträge des Password Managers vor der Speicherung mit AES256 sicher verschlüsselt oder vor dem laden wieder entschlüsselt.
- Das Masterpassword wird als sha256 gehashter Wert in der {Benutzername}_user.json, zur Authentifizierung, gespeichert.

Menü Optionen:

- Die Module: ***checkPassword.py, checkPwned.py, secondFactor.py*** und ***findPassword.py*** dienen zur Umsetzung der Menü Funktionen bzw. der Anforderungen des Password Managers.
- Das Modul ***entry.py*** bietet die gleichnamige Klasse die zur vereinfachten Speicherung von Einträgen im Password Manager dient.

Speicherformat

Alle Dateien werden im resources Ordner gespeichert.

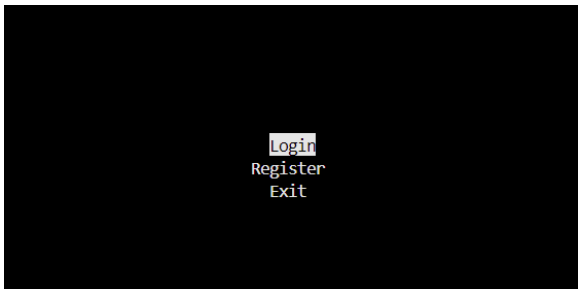
Die Benutzerinformationen wie Benutzername, Masterpassword und Informationen für den Brute-Force-Schutz sowie für die 2FA werden in der *{Benutzername}_user.json* im JSON Format **unverschlüsselt** gespeichert. Das Masterpassword wird als sha256 Hash gespeichert.

Die Einträge des Password Managers werden mit AES256 **verschlüsselt** ebenfalls im JSON Format in der *{Benutzername}_entries.enc* gespeichert.

Beim exportieren der Passwörter werden sie in der Datei *{Benutzername}_exports.json* im **Klartext** gespeichert

Nutzerinterface

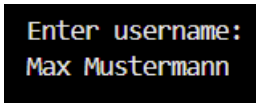
Beim Start des Programmes kommt man in das Loginmenü mit folgenden Funktionen:

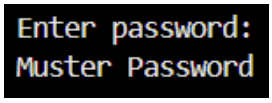


1. Login zu einem Bestehenden Benutzer Konto
2. Registrierung eines neuen Benutzers
3. Verlassen der Applikation

Registrierung:

- Um einen neuen Benutzer zu Registrieren muss ein Benutzername und Masterpassword gewählt werden





Login:

- Für das Einloggen muss Benutzer Name und Password eingegeben werden
- Falls 2FA eingeschalten ebenso ein Code

- Bei wiederholter Eingabe eines falschen Masterpassword wird der Account für 1 Minute gesperrt:

```
Account locked due to multiple failed attempts. Try again later.
```

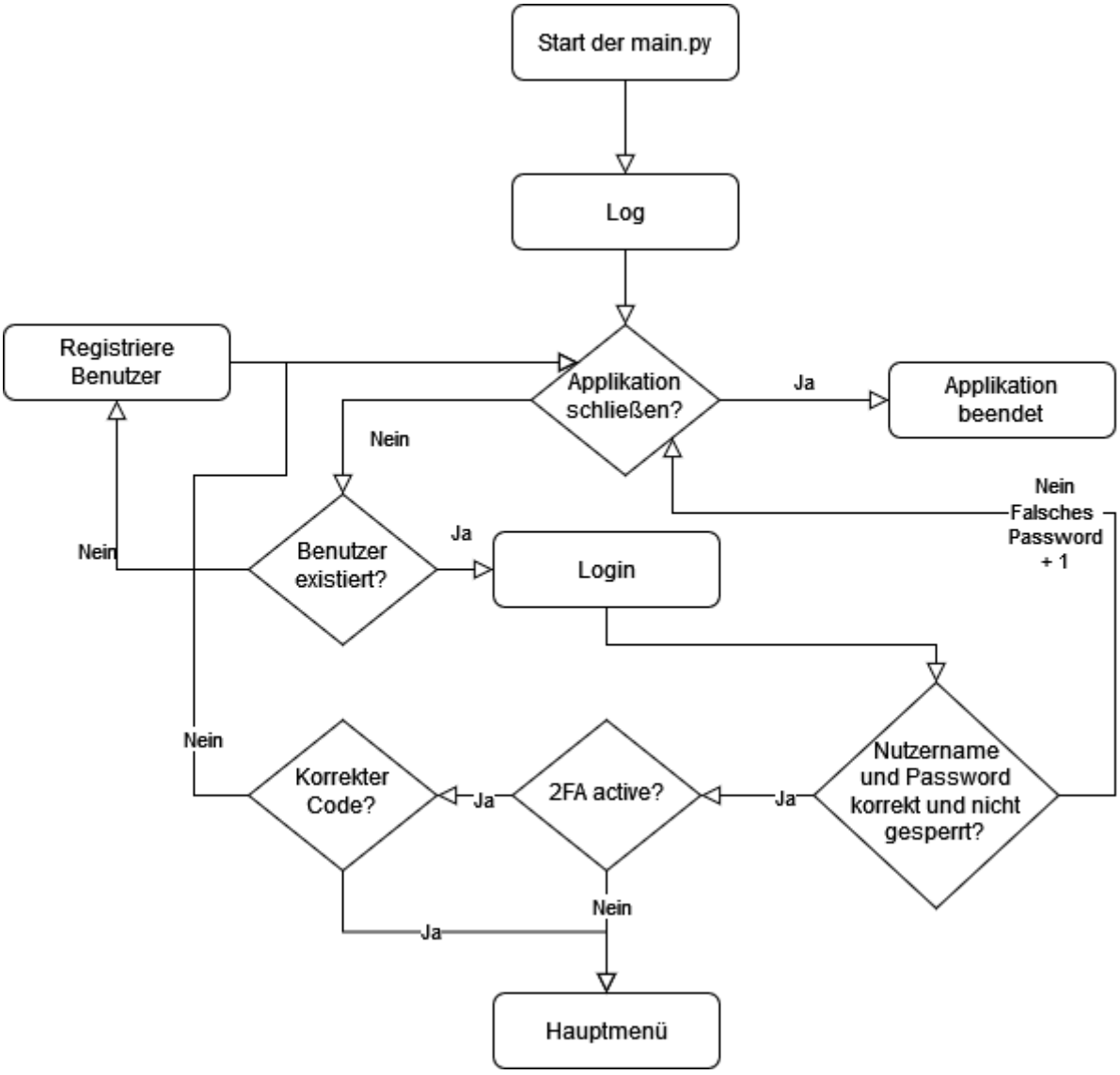
```
Add Password
Generate Password
Edit Password
Delete Password
Find Password
View All Sites
Load from File
Export to File
Options
Logout
```

Logged in:

- Wenn man nun Zugriff auf das Hauptmenü hat, werden alle Funktionen des Password Managers in diesem angezeigt (s. Abb. links)
- Die Auswahl der Optionen führen zu den jeweiligen Rutinen der Funktionen.

Programmablauf

Einloggen:



Hauptmenü Optionen:

1. *Add Password* Einträge:

    - Name der Webseite / URL o.ä.
    - Benutzername. (auf Webseite)
    - Password. (für Webseite)
    - Password bestätigen.
        - Überprüfung ob auf HaveIBeenPwned Einträge existieren.
        - Ob das Password bereits verwendet wurden. (als altes oder in einem andere Eintrag)
        - Sicherheitsüberprüfung. (Länge, Groß-Klein, Zahlen, spezielle Buchstaben)
    - Notizen hinzufügen.
2. *Generate Password* Optionen:
    - Größe des Passwortes (12 - 128) Symbole.
    - Auswahl der Bedingungen für das Password. (Großbuchstaben, Kleinbuchstaben, spezielle Buchstaben, Zahlen)
3. *Edit Password* Optionen:
    - Angabe des Webseiten Namen des zu bearbeitenden Eintrags.
    - Wird ein passender Eintrag gefunden wird gefragt ob man diesen nun bearbeiten will.
        - Des weiteren wird das Datum der letzten Bearbeitung des Eintrags angezeigt.
4. *Delete Password* Optionen:
    - Eingabe des Webseiten Namens. (Eintrag wird **ohne** wiederholtem Fragen gelöscht)
5. *Find Password* Optionen:
    - Auswahl der Suchmethode.
        - Mit URL suchen.
        - Mit Pattern suchen.
    - Rückgabe aller passenden Einträge.
6. *View All Sites* Optionen:
    - Rückgabe aller gespeicherten Einträge.
7. *Load from File* Optionen:
    - Dateipfad zur JSON-Datei eingeben
    - Einträge werden für diesen Benutzer hinzugefügt
8. *Export to File* Optionen:
    - Alle Einträge werden in einer JSON-Datei *{Benutzername}_exports.json* im *resources* Ordner gespeichert.
9. *Options*:
    - Aktivierung und Deaktivierung der 2-Faktor-Authentifizierung

Test Ergebnisse

Unittests

```
kali@rob:~/PythonPasswordManager/PPP-PM$ /bin/python3.11 /home/kali/PythonPasswordManager/PPP-PM/tests/testMain.py
..................................
----------------------------------------------------------------------
Ran 34 tests in 0.675s

OK
```

Coverage

Link zur index.html Erstellt durch coverage: hier

Pylint

```
kali@rob:~/PythonPasswordManager/PPP-PM$ pylint source

-------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 9.99/10, +0.01)

kali@rob:~/PythonPasswordManager/PPP-PM$ pylint tests

-------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)
```

Mypy

```
kali@rob:~/PythonPasswordManager/PPP-PM$ mypy tests
Success: no issues found in 9 source files
kali@rob:~/PythonPasswordManager/PPP-PM$ mypy source
Success: no issues found in 10 source files
```