GESTIONNAIRE DE MOTS DE PASSE : SECUREPASS MANAGER

REALISE PAR : NGO MBEDEG LE-NYE ESPERANCE AUDREY

Cursus : Master 2 CYBER

Table des matières

I. Etat de l'art et objectif du projet _____ *1*

1. Problématique et Objectif du Projet _____ 1
2. Etat de l'art _____ 1

II. Planning et Organisation du Projet _____ *2*

1. Planning du projet _____ 2
2. Organisation du Projet _____ 3

III. Fonctionnalités et Modèle Conceptuel _____ *3*

1. Fonctionnalités de Base _____ 3
2. Fonctions de Sécurité et Gestion des erreurs _____ 4
3. Fonctionnalités d'Interaction Utilisateur _____ 5

IV. Tutoriel d'exécution et d'utilisation de SecurePass Manager _____ *7*

V. Limites et axes d'amélioration du projet _____ *10*

Annexes _____ *11*

Webographie _____ *11*

I. Etat de l'art et objectif du projet

1. *Problématique et Objectif du Projet*

Dans l'ère numérique actuelle, la gestion sécurisée des mots de passe est essentielle face à la multiplication des services en ligne demandant des authentifications fréquentes. Les utilisateurs doivent souvent utiliser un grand nombre de mots de passe selon le cas, ce qui rend les gestionnaires de mots de passe sécurisés indispensables. Ces outils permettent de stocker et gérer de manière sûre les informations d'authentification, réduisant la nécessité de mémoriser plusieurs mots de passe.

Le projet SecurePass Manager consiste en la conception et le développement d'un gestionnaire de mots de passe sécurisé, destiné à offrir une solution pratique et fiable pour gérer les informations de connexion de manière sécurisée. Le gestionnaire de mots de passe sera conçu pour être autonome, il fonctionnera localement sur l'appareil de l'utilisateur, sans nécessiter de connexion Internet afin de garantir la confidentialité des données, offrant ainsi une solution pratique et sécurisée pour stocker et accéder aux informations de connexion.

Les informations seront stockées sur une base de données locale, l'objectif étant de créer une interface utilisateur conviviale permettant aux utilisateurs de stocker, d'organiser et d'accéder facilement à leurs identifiants et mots de passe, tout en assurant la sécurité des informations sensibles.

2. *Etat de l'art*

La sécurité de ces gestionnaires est primordiale car les violations de données peuvent entraîner des conséquences financières graves et nuire à la réputation. Il est donc crucial que ces systèmes résistent efficacement aux attaques par force brute, au phishing et à d'autres menaces, grâce à un cryptage solide et des protocoles d'authentification sécurisés.

Des solutions existantes et dont l'efficacité a été reconnue, nous pouvons relever :

- **KeePass** (1): est un gestionnaire open-source qui se distingue par sa flexibilité et son

extensibilité. KeePass stocke les mots de passe dans une base de données chiffrée localement, que l'utilisateur peut synchroniser manuellement à travers différents dispositifs via des services de stockage en nuage de son choix

- **1Password** (2): est un gestionnaire de mots de passe renommé conçu pour sécuriser, stocker et gérer les identifiants numériques et autres informations sensibles de manière centralisée. Les utilisateurs accèdent à leurs données à travers un mot de passe principal unique, ce qui simplifie la gestion des multiples identifiants tout en renforçant la sécurité.

II. Planning et Organisation du Projet

1. Planning du projet

Le développement du projet sera planifié sur une période de 45 heures de cours dont 20h supplémentaires où nous travaillerons à la finalisation du rapport final. Le planning sera organisé comme suit :

Semaine 1-2 : Analyse des besoins et des exigences fonctionnelles du gestionnaire de mots de passe.

Semaine 3-4 : Apprentissage du langage python pour le développement du gestionnaire de mots de passe, en tenant compte de la logique applicative et une bibliothèque GUI pour l'interface utilisateur.

Semaine 5-6 : Conception détaillée de l'architecture logicielle, y compris la modélisation des données, des fonctionnalités, des interactions utilisateur et la rédaction du cahier des charges complet.

Semaine 7-8 : Développement des fonctionnalités essentielles du gestionnaire de mots de passe, en commençant par l'implémentation de la gestion des comptes utilisateurs et du stockage sécurisé des données.

Semaine 9-10 : Intégration des fonctionnalités avancées telles que l'amélioration de l'interface utilisateur

Semaine 11-12 : Tests intensifs pour garantir la fiabilité, la sécurité et les

performances du logiciel. Correction des bugs et optimisation des performances.

Semaine 13 : Finalisation de la conception du projet avec une révision complète de toutes les fonctionnalités et une préparation pour la phase de présentation.

2. *Organisation du Projet*

En tant que développeur travaillant seule sur le projet, une organisation efficace du travail est essentielle pour atteindre les objectifs fixés dans le temps imparti. Pour se faire, les tâches seront réparties de manière équilibrée sur les différentes périodes de travail, en tenant compte des exigences spécifiques de chaque étape du développement. Une planification minutieuse sera effectuée pour maximiser l'utilisation du temps disponible et garantir une progression régulière du projet.

La communication avec les différents intervenants du module de « Projet de Master » sera maintenue régulièrement lors des cours afin de discuter des progrès, résoudre les problèmes potentiels et obtenir des conseils et des retours d'expérience.

En outre, des séances de travail régulières seront planifiées, en alignement avec les plages horaires disponibles. Nous nous sommes fixés au moins une séance de 4h de travail deux fois par semaine pendant les heures de cours et quatre heures le weekend. Cela garantira une utilisation efficace du temps et permettra de maintenir un bon équilibre entre les exigences du projet et les autres engagements académiques.

III. Fonctionnalités et Modèle Conceptuel

1. *Fonctionnalités de Base*

- **Interface Graphique Utilisateur (GUI)**

Notre interface graphique a été faite grâce à l'utilisation de Tkinter permettant aux utilisateurs d'interagir facilement avec l'application grâce à son côté simpliste. Tkinter a ainsi permis d'initialiser tous les screens des fonctionnalités pour la connexion, la création de compte, la gestion des mots de passe, et l'affichage des informations enregistrées dans une zone d'enregistrement.

- **Gestion de la Base de Données**

Nous avons décidé d'utiliser la base de données locale SQLite pour stocker les données de manière persistante, incluant les informations utilisateur et les mots de passe chiffrés.

Nous comptons deux tables principales : « **users** » et « **passwords** »

- « users » stocke les **usernames**, **les mots de passe hachés** et les **sels**
- « passwords » stocke les informations relatives aux différents comptes et mots de passe que les utilisateurs souhaitent sauvegarder.

2. *Fonctions de Sécurité et Gestion des erreurs*

- **Hachage de Mots de Passe**

Afin d'avoir une base de données sécurisée, nous avons utilisé « **bcrypt** » pour hacher les mots de passe avant de les stocker, offrant ainsi une protection contre les attaques par force brute.

- **Chiffrement et Déchiffrement**

Nous avons utilisé « **cryptography.fernet** » pour chiffrer et déchiffrer les données sensibles, assurant que les mots de passe stockés ne peuvent pas être lus sans la clé de chiffrement appropriée.

Afin d'augmenter également la sécurité lors du stockage des mots de passe, nous avons utilisé des fonctions pour générer un sel aléatoire « **generate_salt()** » et pour dériver une clé de chiffrement à partir d'un mot de passe « **derive_key()** »

- **Sécurité et Gestion des Erreurs**

Pour gérer les erreurs, nous avons utilisé des boîtes de dialogues « **messagebox** » pour informer les utilisateurs des erreurs ou des actions réussies, contribuant à une meilleure expérience utilisateur et à une interaction claire.

3. *Fonctionnalités d'Interaction Utilisateur*

- **Connexion/Déconnexion et Gestion des Tentatives**

Nous avons initié un mécanisme de gestion des tentatives de connexion, avec blocage temporaire après trois tentatives ratées pour sécuriser davantage les comptes contre les attaques par force brute.

Une fois l'utilisateur déconnecté, il est redirigé toutes les informations de session sont réinitialisées ramenant l'utilisateur à l'écran de connexion.

- **Création de Compte**

Cette fonction permet à l'application d'être utilisée par plusieurs utilisateurs ayant chacun leur espace d'enregistrement ou à un utilisateur unique d'avoir plusieurs comptes. Des exigences sur la robustesse du mot de passe devant être de 14 caractères minimum (majuscule, minuscule, chiffre, caractère spécial) ont été mises en place lors de la création d'un nouveau compte.

- **Gestion des Enregistrements de Mots de Passe**

Après connexion, une fenêtre apparait et permet de sauvegarder, mettre à jour et supprimer des informations de connexion pour divers sites ou services, ainsi qu'une fonction de recherche dynamique pour retrouver facilement les informations enregistrées.

La Figure 1 illustre le modèle conceptuel de notre projet :

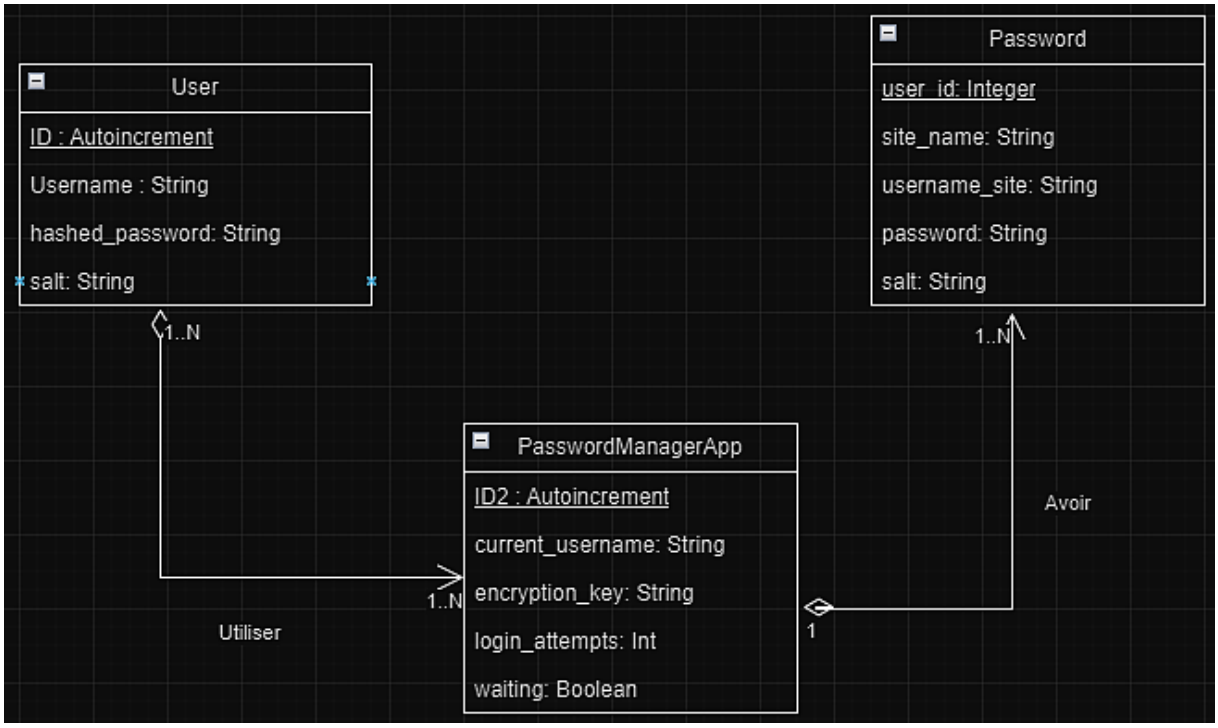

Figure 1: Modèle Conceptuel de SecurePass

Les interactions liées à ce modèle conceptuel sont définies comme suit :

Création de compte : L'utilisateur remplit les champs nécessaires et crée un nouveau compte. La validation des entrées et le hachage sécurisé du mot de passe sont effectués.

Connexion : L'utilisateur saisit ses identifiants pour se connecter. Après trois tentatives échouées, une question secrète est posée.

Gestion des mots de passe : Une fois connecté, l'utilisateur peut ajouter, visualiser et supprimer des mots de passe stockés pour différents sites.

Déconnexion : L'utilisateur peut se déconnecter de l'application, ce qui efface les données temporaires et retourne à l'écran de connexion.

IV. Tutoriel d'exécution et d'utilisation de SecurePass Manager

- ✓ L'application est sous forme **.exe** dans le dossier « **SecurePass Manager** » où toutes les dépendances comme sur la Figure 2 doivent être dans le même dossier que l'exécutable pour que l'application démarre. Un double-clic sur le « SecurePassBy.exe » lance l'application. Le dossier d'exécution se trouve dans le projet GitHub sous forme d'archive et nommé « **SecurePass Manager.rar** ».
  Il est possible qu'il y ait des délais dans l'exécution si tel est le cas double cliquez à nouveau sur le '.exe'.

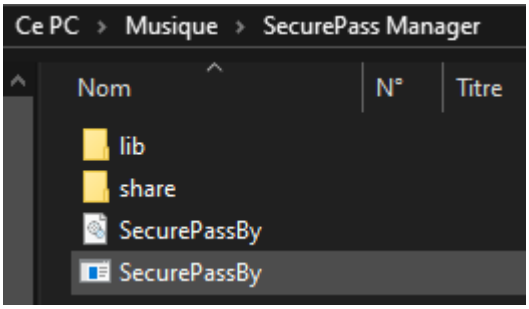

Figure 2: Dossier d'exécution

- ✓ Une fois l'application lancée, la page d'accueil de la Figure 3 se présente à l'utilisateur où il peut rentrer ses identifiants pour se connecter au SecurePass Manager

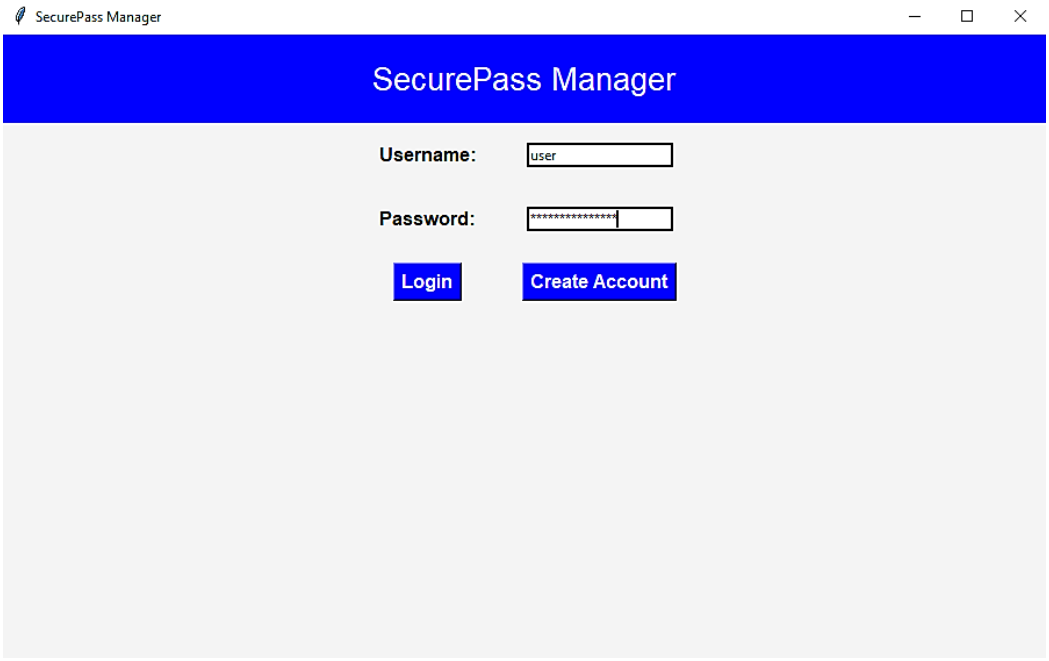

Figure 3: Page d'accueil du SecurePass Manager

✓ Si l'utilisateur n'a pas encore de compte, il peut cliquer sur le bouton « **Create Account** » ce qui lui permettra de créer son compte comme sur la Figure 4 . Une boite de notification lui sera affichée pour confirmer la création de son compte.

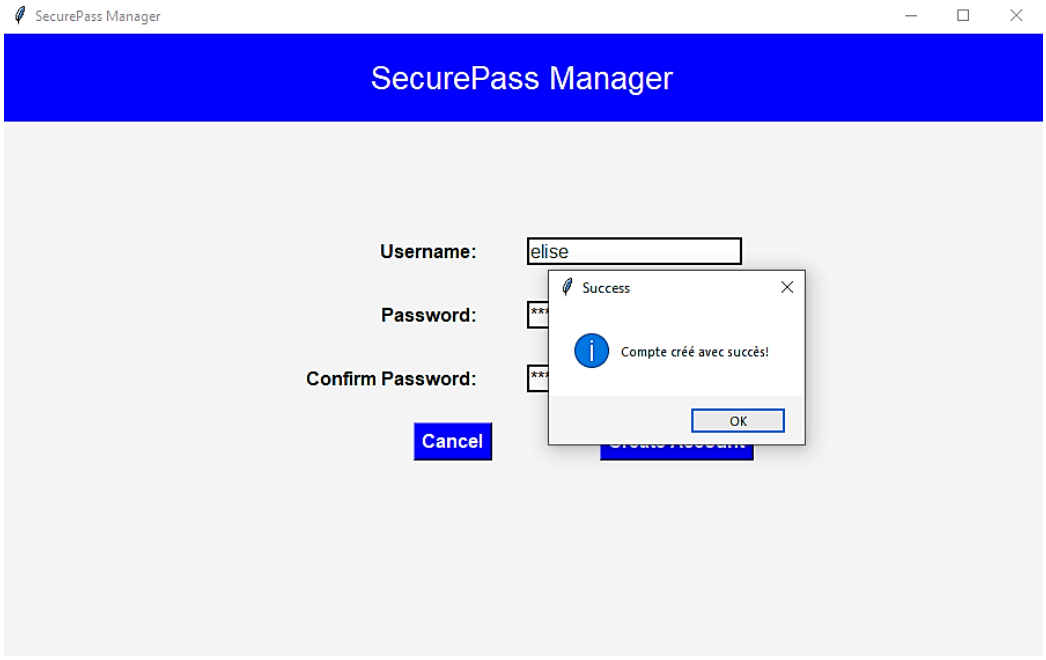

Figure 4 : Page de Connexion

✓ Le compte créé, l'utilisateur pourra se connecter avec ses identifiants comme

expliqué plus haut et ainsi lui sera affiché la page d'enregistrement où il pourra comme montré sur la Figure 5 enregistrer ses mots de passe, les supprimer ou les mettre à jour.

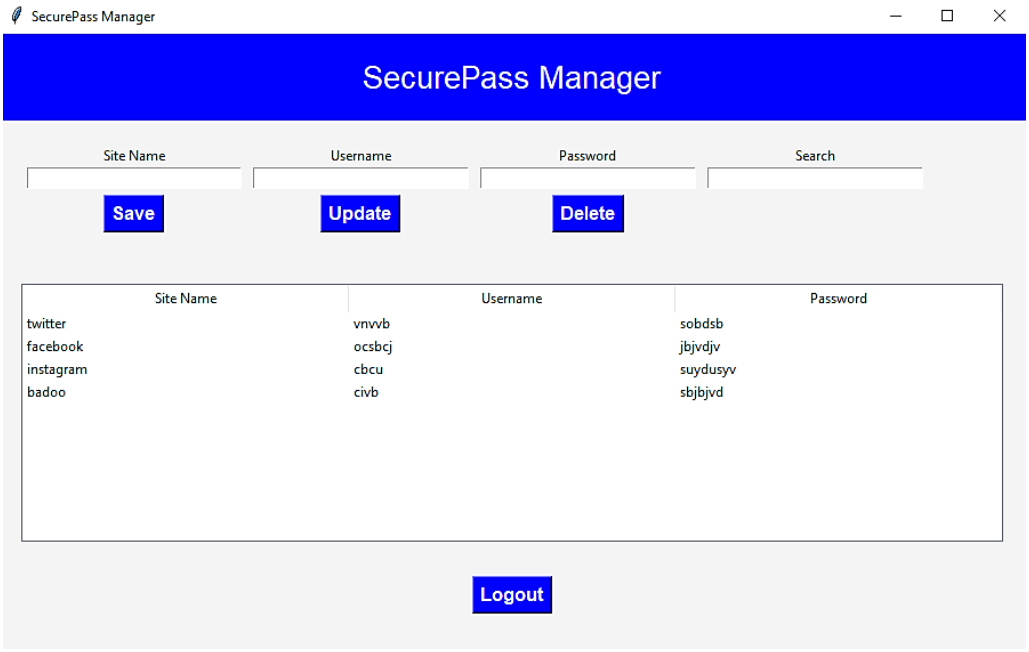

Figure 5 : Page d'enregistrement des mots de passe

✓ Une fois que l'utilisateur a fini ses opérations il peut se déconnecter simplement en appuyant sur le bouton « **Logout** » et il est redirigé automatiquement vers la page d'accueil et une boite de notification disant qu'il a été déconnecté (Figure 6).

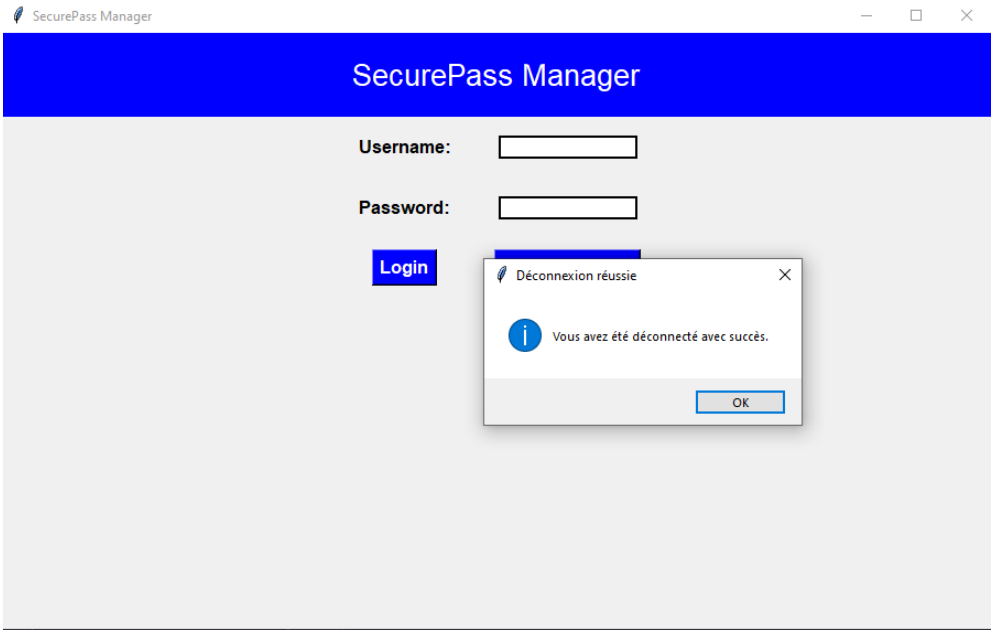

Figure 6 : Notification de déconnexion

V. Limites et axes d'amélioration du projet

Notre projet de gestionnaire de mots de passe présente une bonne base fonctionnelle, mais il existe plusieurs domaines clés où des améliorations sont nécessaires. En matière de sécurité, bien que les mots de passe des sites soient chiffrés avant d'être stockés, la gestion des clés de chiffrement pourrait être renforcée. Utiliser des gestionnaires de clés sécurisés ou des environnements de gestion des secrets améliorerait la sécurité.

Également, l'interface utilisateur, construite avec **tkinter**, pourrait bénéficier d'une mise à jour avec des bibliothèques plus modernes comme **PyQt** ou **Kivy** pour une apparence plus intuitive et moderne. L'ajout de fonctionnalités d'accessibilité, telles que des options de personnalisation de l'affichage et une navigation améliorée, garantirait une meilleure ergonomie pour tous les utilisateurs. En adressant ces points, nous pouvons créer un gestionnaire de mots de passe plus complet, sécurisé et convivial.

Enfin, dû à l'environnement de développement utilisé, il y a des temps de latence à noter tant dans le lancement de l'application que certaines fonctionnalités. Nous pourrons ainsi à l'avenir trouver comment diminuer ces temps de latence.

Annexes

- ✓ Chemin github : https://github.com/Otaku07/SecurePass

Webographie

1. REICHL, Dominik. KeePass Password Safe. https://keepass.info/KeePass

2. Gestionnaire de mots de passe pour les familles et les entreprises | 1Password, https://1password.com/fr