Documentation technique : GDM-001

Pour les programmeurs / programmeuses.

Indication : Seul les fonctions / objets qu'on a jugé important et difficile de compréhension sont référencés ici. Les autres fonctions ainsi que celle référencés ici sont tout de même documenté dans le code avec des docstring. Merci de votre compréhension.

Le fichier à exécuter est le fichier menu.py qui se trouve dans le répertoire sources (gestionnaire_de_mot_de_passe\sources\menu.py). Pour exécuter le code il faut obligatoirement se placer dans le répertoire sources.

Module : codage_mawa

- ### Class CodageMawa :

CodageMawa est un système de chiffrement symétrique (qui permet de chiffrer et de déchiffré du texte avec la même clé).

Comme tous les systèmes de chiffrement moderne la sécurité du CadageMawa repose sur le principe de Kerckhoffs qui dit que "la sécurité d'un algorithme de chiffrement ne doit pas être basée sur la non-connaissance par l'attaquant de l'algorithme de chiffrement ou du système utilisé. Elle doit uniquement être basée sur le fait que l'attaquant ne connaît pas la clé.". De ce fait la clé de chiffrement dois être le plus sécuriser possible l'attaquant ne dois pas pouvoir le deviner ou le prédire. Pour se faire, la clé de chiffrement est générée aléatoirement par un GNA (générateur de nombre aléatoire) non déterministe ou cryptographiquement sécurisé qui utilise l'entropie externe (mouvement de la sourie, température du processeur). Dans le cas du CodageMawa nous avons choisi d'utilisé un GNPA (générateur de nombre pseudo aléatoire)

non déterministe car le GNPA utilise moins de ressource que le GNA. Pour se faire nous avons utilisé l'objet SystemRandom() du module random.

- La méthode master_key_generator :

CodageMawa.master_key_generator permet de générer une master key.

Fonctionnement de CodageMawa.master_key_generator:

Ne prend pas d'argument.

Etape 1 : On génère des nombres pseudo aléatoires (mode longueur du tableau de caractère -1 (pour éviter d'avoir le signe 'Σ' dans la clé et avoir des erreurs lors de son déchiffrement par le master password)) cryptographiquement sécurisés avec random.SystemRandom().randint(). On enregistre les nombres dans un tableau.

Etape 2 : On remplace chaque nombre par la lettre correspondant dans le tableau.

Etape 3 : On transforme le tableau de lettre en chaine de caractères.

- La méthode encode :

CodageMawa.encode permet de chiffrer du texte.

Fonctionnement de CodageMawa.encode:

Arguments : le mot de passe a chiffrer, la clé de chiffrement.

Pour faire le chiffrement il faut que la longueur du password soit égale à la longueur de la master key.

Etape 1 : Si la longueur du password est inférieure à la longueur de la master key on incrémente le password avec des lettres pseudo aléatoires préfixé de 'Σ' pour pouvoir les repérer lors du déchiffrement. 'Σ' marque la fin du mot de passe et le début de la chaine aléatoire. Si au contraire c'est la master key qui a une longueur inférieure à la longueur du password alors on incrémente la master key avec ses (longueur password – longueur master key) premières valeurs, (longueur password – longueur master key) fois pour avoir longueur password = longueur master key.

Etape 2 : On change les caractères de la master key et du password par leurs index dans le tableau de caractère (self.elements).

```
Etape 3 : On change chaque valeur du password par son image par
__algorithme (mode longueur du tableau de caractère).

Etape 4 : On additionne le résultat de l'étape 3 par le master key
(mode longueur du tableau de caractère).

Etape 5 : On change chaque valeur par le caractère correspondant
dans le tableau (self.elements)
```

- La méthode decode

CodageMawa.decode permet de déchiffrer du texte.

Fonctionnement CodageMawa.decode:

Arguments : le chiffré, la clé de chiffrement.

```
Etape 1 : Si la master key a une longueur inférieure à la longueur
du password alors on incrémente la master key avec ses (longueur
password – longueur master key) premières valeurs, (longueur
password – longueur master key) fois pour avoir longueur password =
longueur master key

Etape 2 : On change les caractères de la master key et du chiffré
par leurs index dans le tableau de caractère (self.elements).

Etape 3 : On change chaque valeur du chiffré par son image par
__inv_algorithme (mode longueur du tableau de caractère).

Etape 4 : On additionne le résultat de l'étape 2 par le master key
(mode longueur du tableau de caractère).

Etape 5 : On change chaque valeur par le caractère correspondant
dans le tableau (self.elements)

Etape 6 : On ne garde que les éléments situer avant le 'Σ'.
```

Module : gestion_data

```
Gestion des données de la table gestionnaire.
```

- La fonction save_password

Permet de sauvegarder un compte dans la base de données.

Fonctionnement de la fonction save_password:

Arguments : l'id de l'utilisateur, la plateforme pour laquelle l'utilisateur souhaite enregistrer un mot de passe, son pseudo sur cette plateforme, le mot de passe, la clé de chiffrement, son mail (paramètre optionnelle), la base de données (paramètre optionnelle).

Etape 1 : On vérifier si le mot de passe ne contient pas de caractère inconnue (voir liste des caractères autoriser ci-dessous)

Si Etape vérifier est ok :

```
Etape 2 : On chiffre le mot de passe avec le master key

Etape 3 : On écrit l'id du nouveau mot de passe qui correspond à
l'id de l'utilisateur dans la table user, on écrit la plateforme, le
nom, le mot de passe chiffré et le mail.

Etape 4 : On exécute la requête.

Etape 5 : On commit pour sauvegarder.
```

- La fonction lecture

La fonction lecture permet de lire dans la base de données gestionnaire pour afficher toutes les données ou faire de la recherche

Fonctionnement de la fonction lecture :

Arguments : return_all (indique si l'utilisateur souhaite afficher tout ou rechercher), la clé de chiffrement, l'id de l'utilisateur, la base de données (paramètre optionnelle), nom (paramètre optionnelle, à renseigner pour la recherche), platforme ((paramètre optionnelle, à renseigner pour la recherche).

```
Mode 1 : afficher tout (return_all == True)

Etape 1 : On sélectionne tous les mots de passe ayant pour id l'id de
l'utilisateur passé en paramètre

Etape 2 : On trie la liste de tuple retourné avec la 3éme valeur (nom)

Etape 3 : On transforme la liste de tuples retourner en une liste de
dictionnaires avec la clé ['platforme', 'nom', 'password', 'mail'] chaque
dictionnaire de la liste correspond à un compte avec la plateforme le nom, le
mot de passe déchiffré et le mail.

Mode 2 : rechercher (return_all == False)

Etape 1 : On sélectionne tous les mots de passe ayant pou id l'id de
l'utilisateur passé en paramètre

Etape 2 : On recherche dans cette liste le compte qui a pour nom le nom passé
en en paramètre et qui a pour plateforme la plateforme passée en paramètre.
```

Documentation technique:
GDM-001

- La fonction importation

La fonction importation permet d'importer des données à partir d'un fichier csv.

Fonctionnement de la fonction importation :

Arguments : fichier de l'utilisateur (csv), la clé de chiffrement, l'id de l'utilisateur, la base de données (paramètre optionnelle).

Etape 1 : On vérifie si fichier_user existe.

Si Etape 1 vérifier est ok :

Etape 2 : On vérifie si le fichier contient les entêtes ['platforme', 'nom', 'password', 'mail']

Si Etape 2 vérifier est ok :

Etape 3 : On vérifie que les mots de passe ne contiennent pas de caractères inconnus.

Si l'étape 3 vérifier est ok :

Etape 4 : On écrit l'id du nouveau mot de passe qui correspond à l'id de l'utilisateur dans la table user, on écrit la plateforme, le nom, le mot de passe chiffré, et le mail.

- La fonction supprimer

La fonction supprimer permet de supprimer des données dans la table gestionnaire.

Fonctionnement de la fonction supprimer :

Arguments : Delt_all(permet de savoir s'il souhaite tout supprimer ou supprimer une seule donnée), l'id de l'utilisateur, la base de données (paramètre optionnelle), nom (paramètre optionnelle), platforme (paramètre optionnelle).

Mode 1 : supprimer tout (delt_all == True)

Etape 1 : On supprime tous les données ayant pour id l'id de l'utilisateur.

Mode 2 : supprimer un compte (delt_all == False)

Etape 1 : On supprime le compte ayant pour id l'id en paramètre, le nom en paramètre, la platforme en paramètre.

- La fonction modifier

La fonction modifier permet de modifier des données de la table gestionnaire.

Fonctionnement de la fonction modifier :

Arguments : platforme, nom, l'id de l'utilisateur, la clé de chiffrements, new_nom:str, new_password:str, le base de données (paramètre optionnelle)

Etape 1 : On chiffre le nouveau mot de passe avec le master key

Etape 2 : On modifie avec UPDATE si l'id ,le nom et la platforme correspondent.

Module : gestion_user

Gestion de l'utilisateurs (enregistrement, authentification, modification du compte, suppression du compte)

- La fonction sing_in

Création du compte de l'utilisateur nom, mot de passe.

Fonctionnement de la fonction gestion user:

Arguments : nom, mot de passe, la base de données (paramètre optionnelle)

Etape 1 : On vérifie si le mot de passe ne contient pas de caractères inconnus

Etape 2 : On vérifie s'il n'y a pas déjà dans la base de données user un utilisateur qui a le nom saisi par l'utilisateur.

Etape 3 : Si Etape 1 == non : on crée une master clé pour l'utilisateur.

Etape 4 : On récupère le hash du mot de passe de l'utilisateur dans une variable.

Etape 5 : On vérifie que le mot de passe à la même taille que la master key.

Etape 6 : Si Etape 4 == non : on incrémente le mot de passe avec les (longueur master key - longueur mot de passe)

Première lettre de mot de passe (longueur master key - longueur mot de passe) fois pour avoir

Longueur master key - longueur mot de passe = 0

Soit :

Longueur master key = longueur mot de passe

Etape 6 : On enregistre le nom, le mot de passe hashé et la master key chiffré avec le mot de passe dans la base de données user.

Etape 7 : On récupère les informations de l'utilisateur dans la base de données. On déchiffre la master key avec le mot de passe

Etape 8 : On retourne un dictionnaire avec l'id le nom et la master key de l'utilisateur.

- La fonction login

Connexion de l'utilisateur nom, mot de passe.

Fonctionnement de la fonction login :

Arguments : nom, mot de passe, la base de données (paramètre optionnelle)

Etape 1 : On vérifie si le mot de passe de l'utilisateur et son nom sont correctes.

Etape 2 : Si Etape 1 == oui : On vérifie que le mot de passe à la même taille que la master key chiffré.

Etape 3 : Si Etape 2 == non : on incrémente le mot de passe avec les (longueur master key chiffré - longueur mot de passe)

Première lettre de mot de passe (longueur master key chiffré - longueur mot de passe) fois pour avoir

Longueur master key chiffré - longueur mot de passe = 0

Soit :

Longueur master key chiffré = longueur mot de passe

Etape 4 : On déchiffre la master key chiffré

Etape 5 : On retourne un dictionnaire avec l'id le nom et la master key de l'utilisateur.

- La fonction modifier

Modification du compte de l'utilisateur.

Documentation technique:
GDM-001

Fonctionnement de la fonction modifier :

Arguments : nom master_password new_nom, new_password, la base de données (paramètre optionnelle)

Etape 1 : On vérifie si le nouveau mot de passe ne contient pas de caractère inconnus

Etape 2 : On vérifie si le mot de passe de l'utilisateur est correct.

Etape 3 : On vérifie que le nouveau nom n’est pas déjà utilisé par un autre utilisateur.

Etape 4 : On vérifie que le mot de passe à la même taille que la master key chiffré

Etape 5 : Si Etape 2 == non : on incrémente le mot de passe avec les (longueur master key chiffré - longueur mot de passe)

Première lettre de mot de passe (longueur master key chiffré - longueur mot de passe) fois pour avoir

Longueur master key chiffré - longueur mot de passe = 0

Soit :

Longueur master key chiffré = longueur mot de passe

Etape 6 : On déchiffre la master key chiffré.

Etape 7 : On vérifie que le nouveau mot de passe à la même taille que la master key.

Etape 8 : Si Etape 4 == non : on incrémente le nouveau mot de passe avec les (longueur master key - longueur nouveau mot de passe)

Première lettre de mot de passe (longueur master key - longueur nouveau mot de passe) fois pour avoir

Longueur master key - longueur nouveau mot de passe = 0

Soit :

Longueur master key = longueur nouveau mot de passe

Etape 9 : On chiffre le master key avec le nouveau mot de passe.

Etape 10 : On modifie le nom par new_nom et master key par la nouvelle valeur chiffrée avec new_password

et master_password par le hasher de new_password.

Caractère autoriser :

```
'a', 'b', 'c', 'd', 'e', 'f', 'g', 'o', 'p', 'q', 'r', 's', 't',
'u', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'v', 'w', 'x', 'y', 'z',
'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z',
'1', '2', '3', '4', '5', '6', '7', '8', '9', '0', ' ', '!', '@',
'#', '$', '%', '^', '&', '*', '(', ')', '/', '-', '_', '\', '{',
'}', '|', '`', '°','[', ']', '+', '.', '?', ';', ':', '!', '§', '¤',
'€', '£', '¨', '<',  '>','~', ''', '\"', ',', '=',
```