--SQLite.sql

CREATE TABLE IF NOT EXISTS user (

    user\_id INTEGER PRIMARY KEY,

    username VARCHAR(100) UNIQUE,

    first\_name VARCHAR(100),

    last\_name VARCHAR(100),

    password VARCHAR(50),

    status VARCHAR(50),

    date VARCHAR(50),

    medical\_files VARCHAR(250),

    updated\_passwd\_date DATE,

    isPasswdNeedToChange BIT

isPasswdNeedToChange BOOL

);

* Il n’y a pas de type Boolean pour les serveurs SQL mais on peut utiliser l’option BIT pour stocker des valeurs binaires.

En Lançant le main.py :

File "c:\Users\cleme\Documents\Sco\Python\Mini projet\Python-Ajouts\_Mordjane\main.py", line 19, in <module>

isConnected, status,updated\_passwd\_date,isPasswdNeedToChange,username\_co=auth.auth(conn)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

ValueError: not enough values to unpack (expected 5, got 3)

Main.py

* isConnected, status, updated\_passwd\_date, isPasswdNeedToChange, username\_co
* = auth.auth(conn)
* L’ordre d’appel des variables dans le main est important, si tu mets updated\_passwd\_date à la place de username\_co, tout est faux. Ordre correct :
* isConnected, status, username\_co, updated\_passwd\_date, isPasswdNeedToChange
* = auth.auth(conn)
* Tu appelles « updated\_passwd\_date, isPasswdNeedToChange » depuis auth mais tu ne les as pas return dans le fichier auth.py

return False, None, None

return True, status, username\_co

return False, None, None, None, None

return True, status, username\_co, updated\_passwd\_date, isPasswdNeedToChange

* Pour pouvoir return les valeurs que tu veux vers le main.py il faut d’avoir que tu récupères ces valeurs dans la db.

 cursor = conn.execute("SELECT status, username, updated\_passwd\_date, isPasswdNeedToChange FROM user WHERE username = ? AND password = ?", (username\_co, pwd))

status, username\_co, updated\_passwd\_date, isPasswdNeedToChange = cursor.fetchone()

* Il faut récupérer les infos du user dans la db et ensuite utiliser la méthode fetchone pour récupérer la ligne résultant de la requête et renvoyer une séquence unique d’élément réutilisable pour la suite du programme ou None si aucune ligne n’est disponible.
* Attention au « » dans la db, parce que quand on print updated\_passwd\_date :

print(updated\_passwd\_date)

On obtient par exemple : 1997

Parce que le format DATE nous fait une soustraction puisqu’il manque les ‘’

Formate date : YYYY-MM-DD = 2023-01-25 = 1997

('MARRAR','Mordjane', 'ARRAR','9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08', 'patient','2022-01-25', 'medical file',2022-01-25,1),

('MARRAR','Mordjane', 'ARRAR','9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08', 'patient','2022-01-25', 'medical file','2022-01-25',1),

Après authentification :

File "c:\Users\cleme\Documents\Sco\Python\Mini projet\Python-Ajouts\_Mordjane\main.py", line 56, in <module>

expiration\_passwd\_check = current\_date - updated\_passwd\_date

~~~~~~~~~~~~~^~~~~~~~~~~~~~~~~~~~~

TypeError: unsupported operand type(s) for -: 'datetime.date' and 'str'

* Ici le problème est que tu demandes à comparer une date de type « date » et une date de type « string »

Pour pouvoir soustraire une date à une autre il faut que les deux soient de type « date ». Donc quand on récupère la valeur de updated\_passwd\_date de type « string » on va la reformater pour avoir un type « date ».

current\_date = datetime.datetime.today()

current\_date=datetime.datetime.today()

updated\_passwd\_date=datetime.datetime.strptime(updated\_passwd\_date,'%Y-%m-%d %H:%M:%S.%f')

print(type(updated\_passwd\_date))

print(type(current\_date)) //pour vérifier le type de la variable

<class 'datetime.datetime'> 🡪 Les deux sont bien de même type.

<class 'datetime.datetime'>

* Attention datetime.datetime.today() datetime.date.today() ne renvoient pas la même chose.
* J’ai trouvé la methode .strptime pour convertir un str en date mais ca ne marche pas avec datetime.date.

Comme on doit comparer ensuite avec current\_date on est obligé d’avoir datetime.datetime. Du coup on aura les heures/minutes/secondes. Je regarderai si je ne trouve pas autre chose mais bon en attendant ca fait le taff

* Après avoir demandé la modification du password, il ne faut pas proposer d’autres changements puisqu’on est pas dans edit\_user.py.

test=input("Other changes ? : Y/N  ")

if test.lower()=='y':

   continue

else:

   element='6'

current\_date = datetime.datetime.today()# pk t'as fait ca?

logging.info('[ %s ], %s, User %s generated', username\_co, current\_date, username)

* Pourquoi j’ai fais ca ? : Pour avoir la date actuelle pour les logs. Comme ca on a la date et l’heure ou l’action a été effectuée.

if (expiration\_passwd\_check.days >= 90) AND (isPasswdNeedToChange == 0) :

* Attention cette condition ne fonctionne plus si l’utilisateur entre dans la boucle mais qu’il fait que des inccorrect password.
* Modification de la condition :

if (expiration\_passwd\_check.days >= 90) or (isPasswdNeedToChange == 1) :

* On prend en compte que l’utilisateur n’est pas satisfait de son mdp et rep non pour l’ajouter à la db. Dans ce cas là le user ne doit pas se retrouver dans le menu du main.py mais doit saisir un autre pwd.

resp=input("Add this password in the db ? Y/N : ")

if resp.lower() == 'y':

//ajout de la modification dans la db

else:

   pwd\_check=False

   print("You have to change your password")

            while menu!='5':

                print()

                print("Actions :")

                print ("1 - Create user")

                print ("2 - Supp user")

                print ("3 - Edit user")

                print ("4 - Show user table")

                print ("5 - change user passwd")

                print ("6 - Logout")

* Quand tu ajoutes qqc dans le menu, penses a modififer la valeur de fin du while. Parce du coup ton « 6 – Logout » ne fonctionne pas.

case '5' :

   req = cur.execute("SELECT username,status FROM user WHERE isPasswdNeedTochange = 1 ")

* Attention quand tu fais une requête sql avec ton cursor vérifie que ton cur est bien défini.

cur = conn.cursor()

req = cur.execute("jhkaeh")

* Je ne suis pas sûre mais je crois que ta condition :

if user[‘status’]== « doctor » ne fonctionne pas.

for user in users:

  print("changement de mot de passe de l'utilisateur: "+user['username'] )

  if(user['status'] == "doctor"):

     i=0

* Comme tu es dans un tableau il faut préciser l’index dans lequel se trouve le champ que tu veux utiliser donc 0 pour le username et 1 pour le status.

print("changement de mot de passe de l'utilisateur: " + user[0] )

if(user[1] == "doctor"):

* Pour la condition juste au dessus, je pense que ca serait plutôt

== « patient » puisque le doctor peut se connecter. L’update de leur mdp se fera du coup au début du main.py

On obtient du coup :

cur = conn.cursor()

req = cur.execute("SELECT username, status FROM user WHERE isPasswdNeedTochange = 1 ")

users = req.fetchall()

for user in users:

   print("Changement de mot de passe de l'utilisateur: " + user[0])

   if(user[1] == "patient"):