

Mise en place de la solution d'authentification :

J'ai créé trois fichiers CreationCompte.py, ConnexionAppli.py et Accueil.py.

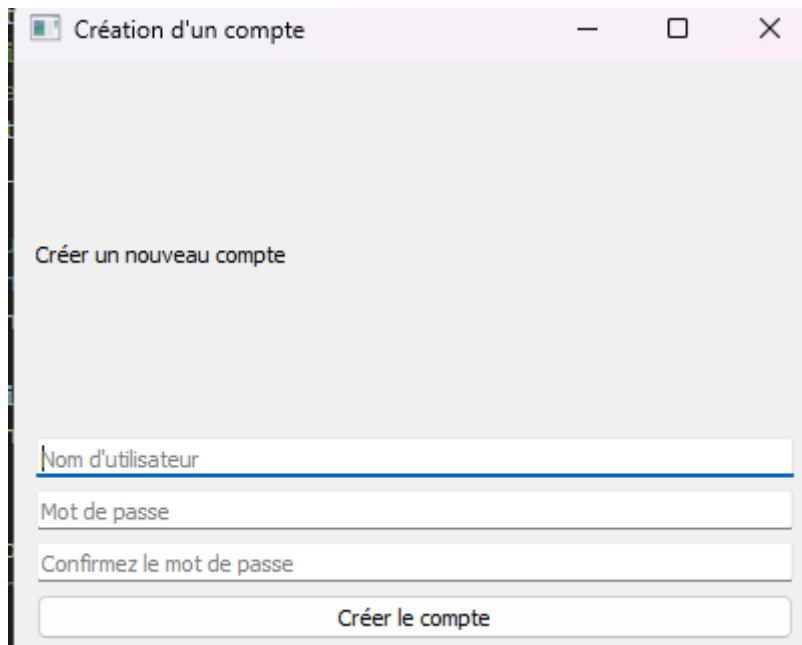
La solution d'authentification pour l'obligation d'avoir un mot de passe robuste se trouve dans le fichier CreationCompte.py (chaque création de compte se fera par ce fichier). La connexion permet de vérifier si l'utilisateur entre des logins corrects, s'ils sont corrects, il est renvoyé vers la page d'accueil contenue dans (Accueil.py).

Jeu de test :

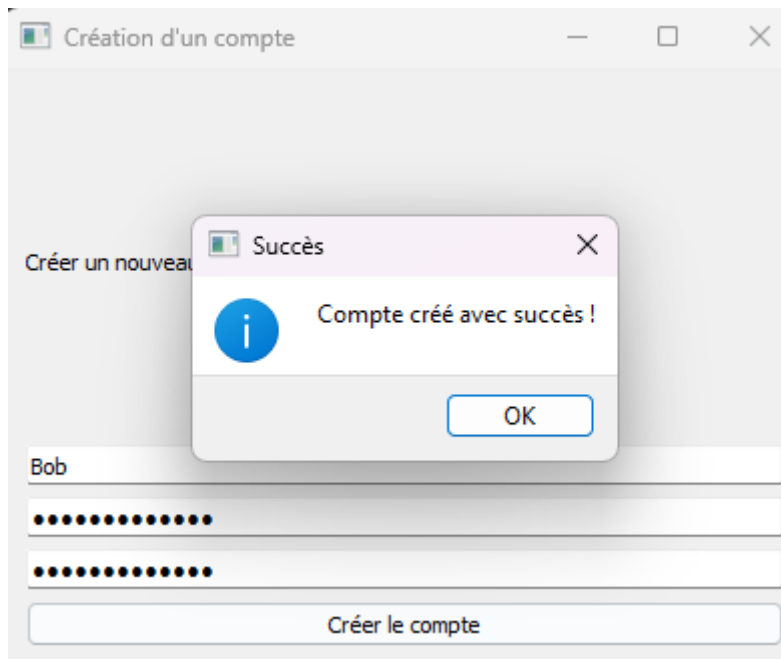
J'ai d'abord créé une table Utilisateurs, créé un utilisateur, créé un système de création de compte, puis j'ai créé le système de connexion.

J'ai donc créé CreationCompte.py puis j'ai créé un utilisateur pour essayer :

CreationCompte.py :



Création d'un utilisateur avec un mot de passe conforme :



Base de données avant la création de l'utilisateur :

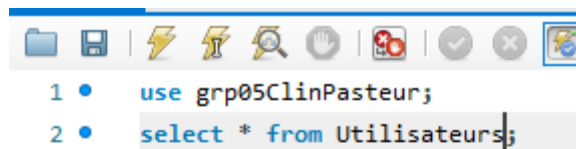
```

1 • use grp05ClinPasteur;
2 • select * from Utilisateurs;

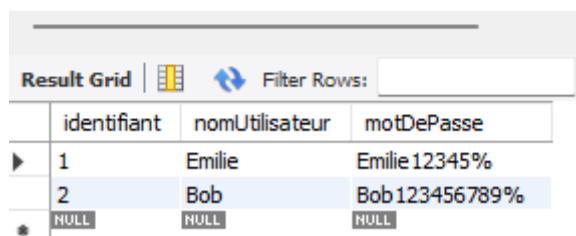
```

	identifiant	nomUtilisateur	motDePasse
▶	1	Emilie	Emilie12345%
*	NULL	NULL	NULL

Base de données après la création de l'utilisateur :

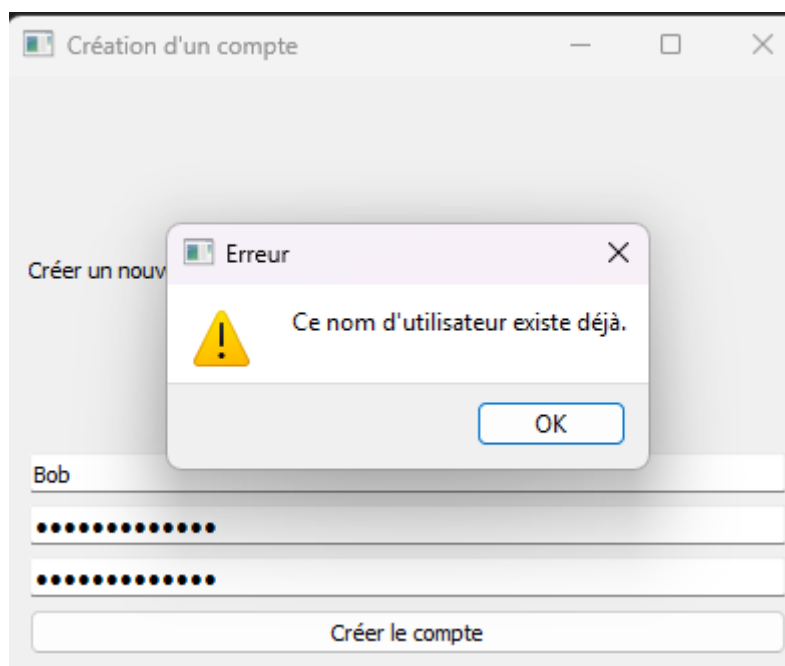


```
1 • use grp05ClinPasteur;  
2 • select * from Utilisateurs;
```

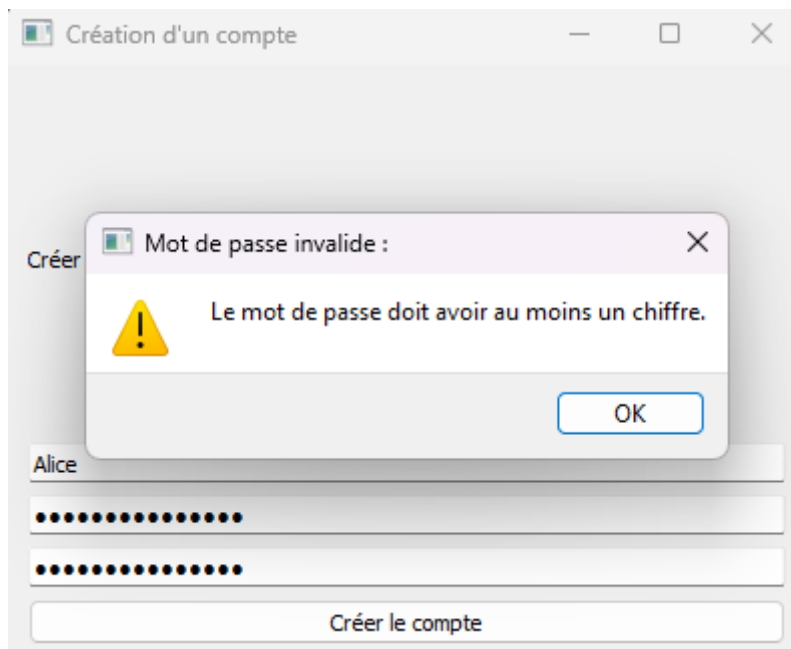
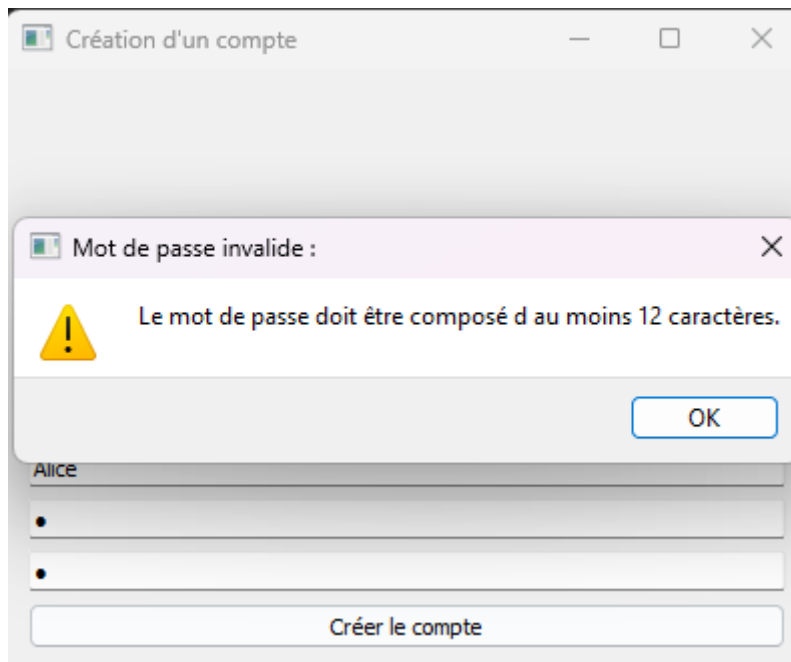


	identifiant	nomUtilisateur	motDePasse
▶ 1		Emilie	Emilie12345%
2		Bob	Bob123456789%
*	NULL	NULL	NULL

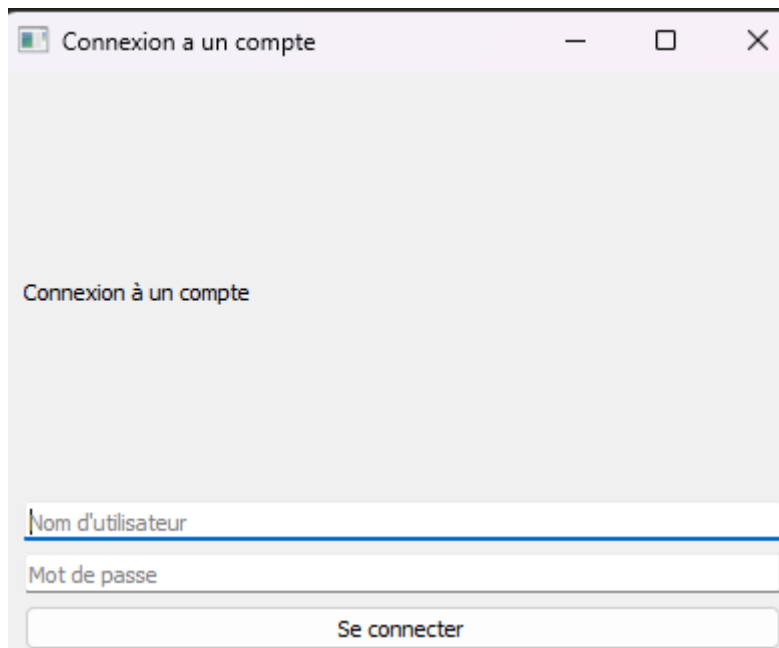
Tentative de création avec un utilisateur déjà existant :



Tentatives de création d'un utilisateur avec un mot de passe pas conforme :



ConnexionAppli.py :



Connexion à un compte

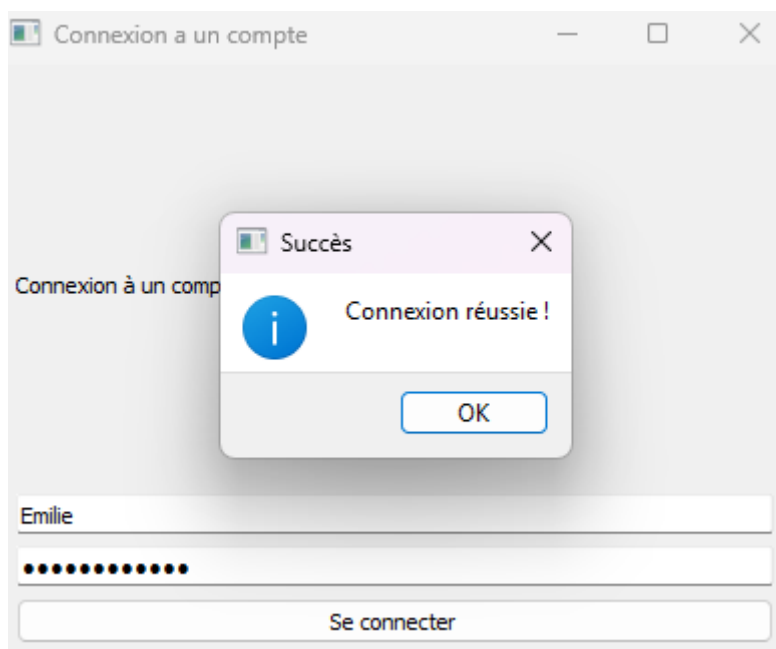
Nom d'utilisateur

Mot de passe

Se connecter

This is a screenshot of a web application's login window. The window has a title bar with the text 'Connexion à un compte' and standard minimize, maximize, and close buttons. The main content area is light gray and contains the text 'Connexion à un compte' at the top. Below this, there are two input fields: the first is labeled 'Nom d'utilisateur' and the second is labeled 'Mot de passe'. At the bottom of the form is a button labeled 'Se connecter'.

Tentative de connexion avec un compte existant :



Connexion à un compte

Succès

Connexion réussie !

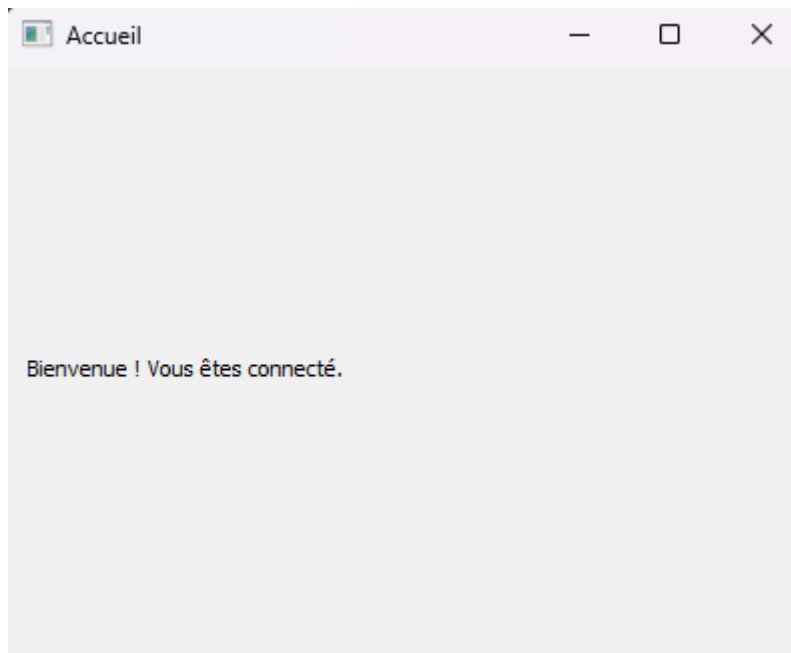
OK

Emilie

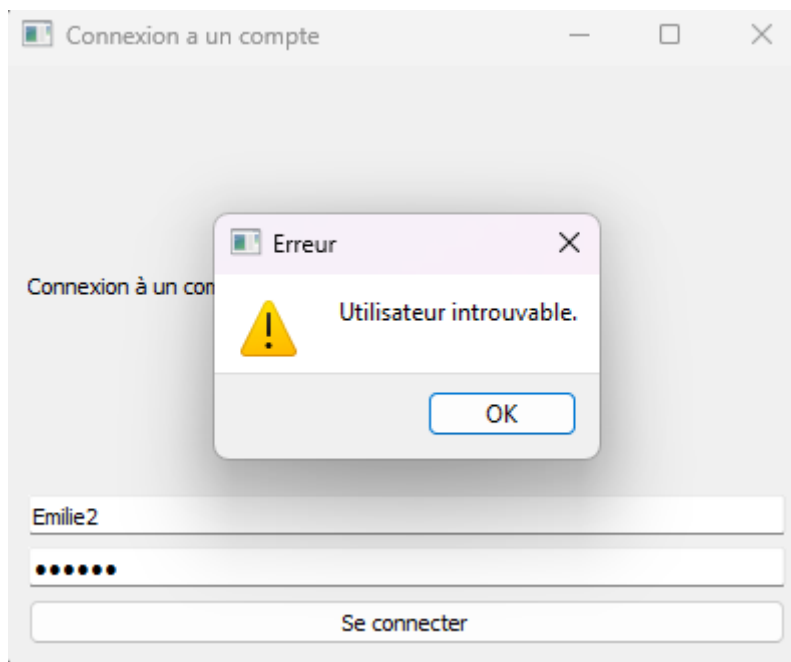
Se connecter

This screenshot shows the same login window as before, but with a modal dialog box overlaid in the center. The dialog box has a title bar that says 'Succès' and a close button. It contains a blue information icon, the text 'Connexion réussie !', and an 'OK' button. In the background, the login form is partially visible, showing the username field filled with 'Emilie' and the password field filled with dots. The 'Se connecter' button is also visible at the bottom.

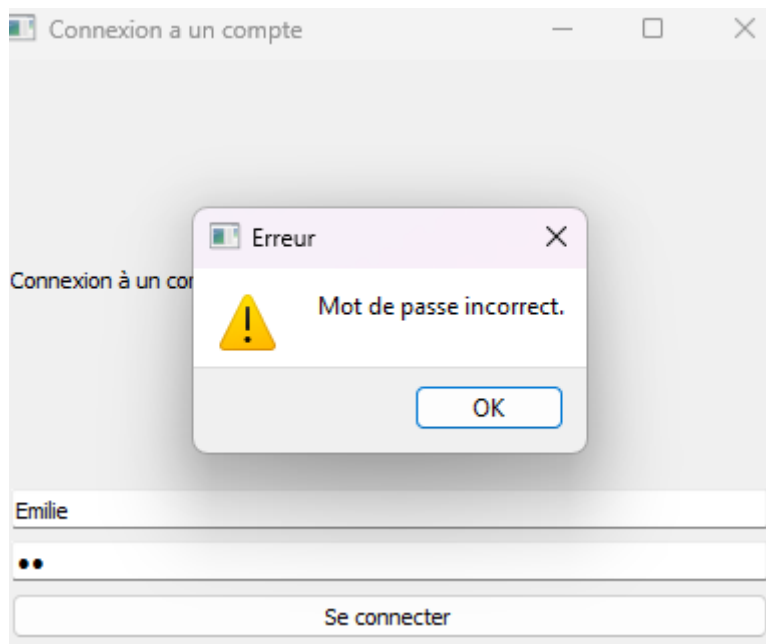
Après avoir cliqué sur "Ok", redirection automatique vers la page d'accueil :



Tentative avec un compte inexistant :



Tentative avec un login incorrecte :



Code pour la mise en place de la solution d'authentification (mot de passe robuste) :

```
def mot_de_passe_valide(password: str):
    if len(password) < 12:
        return False, "Le mot de passe doit être composé d au moins 12 caractères."
    if not re.search("[A-Z]", password):
        return False, "Le mot de passe doit avoir au moins une majuscule."
    if not re.search("[a-z]", password):
        return False, "Le mot de passe doit avoir au moins une minuscule."
    if not re.search("[0-9]", password):
        return False, "Le mot de passe doit avoir au moins un chiffre."
    if not re.search("[<>!$:/;.,?%^$*&]{#~|@()]", password):
        return False, "Le mot de passe doit avoir au moins un caractère spécial."
    return True, "Mot de passe valide."
```

```
#.strip() retire les espaces au début et à la fin du texte
password = self.text_input_saisi_password.text()
confirm = self.text_input_saisi_password_confirme.text()

#Vérification de l'utilisateur et du mot de passe s'ils sont conformes pour être créés
if not username or not password: #si l'username ou le mdp sont vides:
    QMessageBox.warning(self, "Erreur", "Veuillez remplir tous les champs.") #Message d'erreur apparaît
    return
if password != confirm : #mdp différent de la confirmation du mdp
    QMessageBox.warning(self, "Erreur", "Vous avez saisis 2 mots de passe différents.")
    return

valide, message = mot_de_passe_valide(password) #Appelle la fonction avec le mdp entré
#valide va stocké True ou False si le mdp respecte les règles imposées
#message va stocker un message explicatif
if valide == False:
    QMessageBox.warning(self, "Mot de passe invalide : ", message)
    return

#Si le mdp respecte tous ces if :
try:
    #Vérifie si nom d'utilisateur déjà existant
    result = self.db.query("SELECT COUNT(*) AS nb FROM Utilisateurs WHERE nomUtilisateur = %s", (username,))
    if result[0]['nb'] > 0: #S'il y a au moins un resultat, cela signifie qu'il existe déjà un compte avec ce nom d'utilisateur
        QMessageBox.warning(self, "Erreur", "Ce nom d'utilisateur existe déjà.")
        return

    self.db.cursor.execute( #On ajoute le nouvel utilisateur (j'ai créé la table utilisateurs)
        "INSERT INTO Utilisateurs (nomUtilisateur, motDePasse) VALUES (%s, %s)", (username, password)
    )
    self.db.conn.commit() #Enregistre dans la bdd l'insertion
    QMessageBox.information(self, "Succès ", "Compte créé avec succès !")
```