Exercices : interagir avec une base de données via Python

Attention : Afin de garder les choses simples et vous permettre de vous concentrer sur les problématiques liées aux bases de données, les applications que vous produirez dans le cadre de ces exercices seront sans interfaces graphiques et pour un usage théorique sur serveur distant.

Exercice 1: Un forum

Dans cet exercice, vous réaliserez une application simulant le fonctionnement d'un forum. Votre application sera développée en orienté objet. L'application présentera les fonctionnalités suivantes :

- Quand l'utilisateur lance l'application, la page d'accueil lui affiche l'ensemble des messages contenus dans la base de données
- Un message est présenté avec le nom de celui qui l'a écrit, la date de publication, l'heure de publication et son id
- L'utilisateur peut choisir entre voir les messages (action par défaut), écrire un message ou quitter l'application
- Quand l'utilisateur écrit un message, on lui demande successivement le pseudo qu'il souhaite utiliser et le message qu'il souhaite poster

Bien entendu pour stocker les messages écrits par les utilisateurs et pouvoir les retrouver par la suite, vous allez les sauvegarder dans une base de données PostgreSQL.

Cette base de données ne contiendra qu'une seule table : message, dont voici la structure.

id(integer, clé primaire)
content(text, non nul)
publishing_date(timestampz, non nul)
author(varchar maximum 50 caractères, non nul)

Pour vous aider, voici l'ordre des étapes que vous pouvez suivre. Ce n'est pas la seule manière de faire mais cela peut vous aider si vous êtes perdu :

- Créer la table message via la ligne de commande en veillant à respecter la structure demandée
- Tenter de se connecter à la BDD grâce à la librairie Psycopg2, vérifier que la connexion peut s'établir avant de passer à la suite en exécutant le script
- Écrire dans un fichier main.py un code capable de récupérer l'action utilisateur et d'effectuer une action spécifique selon la commande rentrée
- Créer un fichier de vues contenant deux méthodes, une pour l'affichage des messages et une pour l'écriture d'un message. Pour l'instant, ces commandes ne font rien de particulier si ce n'est un print d'un message de test
- Vérifiez que lorsque l'utilisateur demande à voir les message ou à écrire un message ce soit la bonne méthode de la vue qui est appelée dans le fichier main.py
- Créer un fichier de modèle contenant deux méthodes, une pour récupérer tous les messages en base de données et une pour enregistrer un message en base de données. Attention cette méthode attend au moins deux arguments, le nom de l'auteur et le contenu de son message
- Appeler chaque méthode du modèle dans la méthode correspondante de la vue et vérifier son fonctionnement
- Mettre en forme l'affichage des messages
- Mettre en forme les inputs pour récupérer les informations nécessaires à la rédaction d'un nouveau message

Exercice 2 : un espace utilisateur

Dans cet exercice, vous allez simuler le fonctionnement d'un espace privé auquel les utilisateurs accèdent grâce à leur compte personnel protégé par mot de passe. Vous allez ainsi travailler les requêtes en bases de données, mais aussi la gestion des mots de passe.

L'application présentera les fonctionnalités suivantes :

- Une page d'accueil, qui permet à l'utilisateur de choisir entre se connecter avec un compte existant ou se créer un compte
- Pour se créer un compte l'utilisateur a accès à une série de prompts demandant les informations nécessaires. Une fois les informations rentrées, celles-ci sont enregistrées en base de données.
- Si le pseudo ou le mail rentrés par l'utilisateur sont déjà pris on demande à l'utilisateur d'en entrer un nouveau
- Pour se connecter l'utilisateur doit rentrer le pseudo et le mot de passe choisis lors de la création du compte. Si les identifiants sont corrects, l'utilisateur accède alors à la page de son profil, autrement on lui affiche un message d'erreur.
- Sur sa page de profil l'utilisateur peut choisir de supprimer son compte, ses informations sont alors effacées de la base de données. Il peut également choisir de mettre à jour une information en indiquant la colonne à modifier et sa nouvelle valeur
- L'utilisateur peut quitter l'application à tout moment en tapant une commande de votre choix

Spécifications techniques :
- Les mots de passe sont cryptés en base de données, autrement-dit ils ne sont pas stockés en clair, ils sont illisibles en l'état par un être humain

Cette base de données ne contiendra qu'une seule table : users, dont voici la structure.

id (integer, clé primaire, non nul)
name (varchar maximum 50 caractères, non nul)
firstname (varchar maximum 50 caractères, non nul)
pseudo (varchar maximum 50 caractères, unique, non nul)
email (varchar maximum 250 caractères, unique, non nul)
age (integer, non nul)
password (varchar de minimum 64 caractères, non nul)

NB : nous ne nous intéressons pas ici à la notion de salage de mot de passe mais il s'agit d'un élément très important de la sécurité d'une application. Vous pouvez lire cet article pour en savoir plus : https://auth0.com/blog/adding-salt-to-hashing-a-better-way-to-store-passwords/

Si vous souhaitez aller plus loin, vous pouvez également rajouter des vérifications de sécurité. Par exemple :
- S'assurer que les chaînes de caractères pour le nom et le prénom ne contiennent pas de caractères spéciaux
- S'assurer que l'âge est bien un nombre
- S'assurer que l'adresse mail est une adresse mail valide
- Saler le mot de passe