

Mini-Projet: Création d'un système de gestion des commandes

Les besoins fonctionnelles:

La première question que nous devons nous poser: ***qu'est-ce que nous souhaitons faire concrètement ?***
Cela nous permettra de définir ce que nous souhaitons éviter d'avoir à concevoir.

Dans un premier temps, nous aurons besoin d'un espace utilisateurs. Vous devriez donc savoir qu'un espace utilisateurs nécessite au minimum les éléments suivants :

- Une page de connexion
- Une page de déconnexion
- Une page d'inscription (Ajout de nouveaux utilisateurs)

Ensuite nous pourrons ajouter d'autres pages, par exemple pour afficher et modifier son profil de utilisateur.

Pour le côté gestion des commandes, nous aurons besoin des éléments suivants:

- Une page d'ajout des produits.
- Une page affichant tous les produits.
- Une page d'ajout pour passer des commandes.
- Une page affichant toutes les commandes.
- Une page d'ajout des clients.
- Une page affichant tous les clients.
- Une page d'accueil fournissant un menu pour accéder au différentes pages.

Pour commencer, nous allons créer la table MySQL qui stockera les utilisateurs du système.

La table des utilisateurs:

La table **Users** a quatre champs :

- Un pseudonyme.
- Un mot de passe.
- Une adresse e-mail.
- Une date d'inscription.

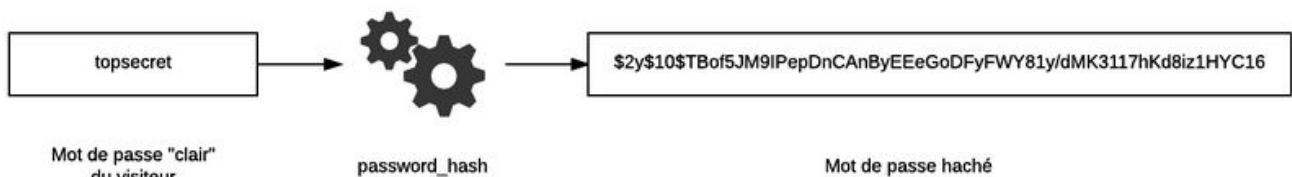
Ces champs sont résumés sur la figure suivante qui présente la table une fois créée sous phpMyAdmin.

Champ	Type
<u>id</u>	int(11)
pseudo	varchar(255)
pass	varchar(255)
email	varchar(255)
date_inscription	date

Un de ces champs mérite une attention particulière : celui qui stocke le mot de passe.

Il faudrait hacher le mot de passe avant de le stocker dans la base.

Le Hachage est une fonction qui transforme n'importe quel texte en un nombre hexadécimal qui représente le mot de passe mais qui est illisible, comme le montre la figure suivante.



Pour hacher un mot de passe, il existe plusieurs fonctions qui se basent sur des algorithmes différents.

La bonne fonction à utiliser est [password_hash](#), qui va choisir pour vous le meilleur algorithme.

La particularité du hachage est qu'il fonctionne dans un seul sens : il est impossible de retrouver le mot de passe d'origine une fois qu'il a été haché. De plus, un *hash* (nom donné à la version hachée du mot de passe) est unique : il correspond à un et un seul mot de passe.

Vous stockerez la **version hachée du mot de passe**, qui sera donc passé à la moulinette par la fonction [password_hash](#). Lorsqu'un utilisateur voudra se connecter, il vous enverra son mot de passe que vous hacherez à nouveau et que vous comparerez avec celui stocké dans la base de données.

Si les deux mots de passe hachés sont identiques, alors cela signifie que l'utilisateur a entré le bon mot de passe.

Réalisation des pages principales de l'espace:

Nous avons déterminé un peu plus tôt la liste des pages dont nous avons besoin au minimum pour gérer nos utilisateurs :

- Ajout d'utilisateur.
 - Connexion ;
 - Déconnexion.
1. **La page d'inscription** est en général constituée de quatre champs :
 - Pseudonyme souhaité.
 - Mot de passe.
 - Confirmation du mot de passe (pour éviter les erreurs de saisie).

- Adresse e-mail.

La figure suivante présente un formulaire basique d'ajout d'utilisateur.

Un formulaire d'ajout d'utilisateur avec quatre champs de saisie. Les étiquettes sont : 'Pseudo', 'Mot de passe', 'Retapez votre mot de passe' (sur deux lignes) et 'Adresse email'. Chaque étiquette est alignée à gauche de son champ de saisie correspondant.

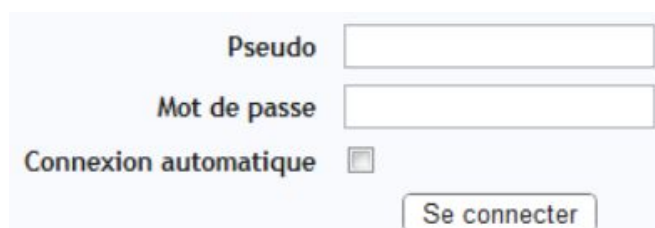
Avant d'enregistrer l'utilisateur dans la base de données, il faudra penser à faire un certain nombre de vérifications.

- Le pseudonyme demandé par l'utilisateur est-il libre ? S'il est déjà présent en base de données, il faudra demander au visiteur d'en choisir un autre.
- Les deux mots de passe saisis sont-ils identiques ? S'il y a une erreur, il faut inviter le visiteur à rentrer à nouveau le mot de passe.
- L'adresse e-mail a-t-elle une forme valide ? Vous pouvez utiliser une expression régulière pour le vérifier.

Si toutes ces conditions sont remplies, on peut insérer l'utilisateur dans la base de données.

2. La page de connexion :

Habituellement, on demande au moins le pseudonyme (ou *login*) et le mot de passe de l'utilisateur.

Un formulaire de connexion avec deux champs de saisie : 'Pseudo' et 'Mot de passe'. En dessous, il y a une case à cocher pour 'Connexion automatique'. À droite de la case à cocher se trouve un bouton 'Se connecter'.

La page qui reçoit les données du formulaire de connexion **doit vérifier le mot de passe en le comparant à celui stocké dans la base** avec la fonction [`password_verify`](#). Cette fonction va en fait hasher le mot de passe de l'utilisateur qui vient de se connecter et le comparer à celui qui était stocké en base de données.

S'il existe un utilisateur qui a le même pseudonyme et le même mot de passe haché, alors on autorise la connexion et on peut créer les variables de session. Sinon, on renvoie une erreur indiquant que le pseudonyme ou le mot de passe est invalide.

3. La page de déconnexion :

Au bout d'un certain temps d'inactivité, la session du membre est automatiquement détruite et il se retrouve déconnecté. S'il charge à nouveau une page du site, il apparaîtra donc déconnecté.

Si la déconnexion est automatique au bout d'un certain temps (*timeout*), il faut quand même proposer un lien de déconnexion. La page de déconnexion devra supprimer le contenu de `$_SESSION`, mettre fin au système de sessions (en appelant `session_destroy()`)

Les tables des produits, Clients et Commandes:

Utilisez les tables que vous avez créé lors du dernier travail **DB_MGF**.

Créez des pages pour

- Ajout des produits (*Formulaire*).
- Affichant tous les produits.
- Ajout pour passer des commandes (*Formulaire*).
- Afficher toutes les commandes.
- Ajout des clients (*Formulaire*).
- Afficher tous les clients.
- D'accueil fournissant un menu pour accéder au différentes pages après connexion d'un utilisateur.

Format pour rendre le mini-projet:

Le projet devra être rendu le **Mercredi 20 Mai minuit au plus tard** (*tout dépassement aura un impact négatif sur la note*).

Pour rendre le projet, envoyez un message électronique à t.aitbaha@uiz.ac.ma.

Un seul message par projet (par groupe).

Le sujet du message sera du type "Mini-Projet Tec-Web de *noms des étudiants*"

Le corps du message devra rappeler les **noms des étudiants du binôme**.

Le fichier **zip** attaché au message devra comporter tout votre travail sur ce projet (*Codes+Base de données+Rapport*).

Évitez d'envoyer plusieurs messages pour un même projet. Seule la première version envoyée sera prise en compte pour la correction.

Il sera tenu le plus grand compte de :

1. Bonne architecture de l'application.
2. La lisibilité du code.
3. L'extensibilité du code..
4. L'ergonomie (*faites tester par une personne étrangère au développement...*). L'utilisateur ne doit pas avoir besoin d'aller lire le manuel de l'utilisateur que de façon exceptionnelle.
5. La conformité du projet au format demandé.