BDW1 - TP 4
Une mini messagerie instantanée en PHP

UCBL - Département Informatique de Lyon 1 – Printemps 2018

1 Objectif et organisation

1.1 Fonctionnalités

Lors de cette séance de TP, vous allez réaliser une application de messagerie instantanée (nommée *chat* par la suite) reposant sur PHP, une base de données MySQL et HTML5/CSS3. Cette application permet à un utilisateur de s'inscrire via un formulaire (pseudo/mot de passe) et de se connecter par la suite. Une fois connecté, l'utilisateur peut voir les derniers messages postés sur le fil de discussion global et d'en envoyer de nouveaux. L'utilisateur peut se déconnecter à tout moment et revenir sur la page d'accueil.

1.2 Découpage

Afin d'implémenter les différentes fonctions nécessaires au fonctionnement du chat et l'affichage des pages, des squelettes de fichiers vous sont donnés.

Les fichiers **bd.php**, **discussion.php** et **utilisateur.php** contiennent les différentes fonctions à implémenter pour enregistrer et extraire des informations de la base de données qui seront nécessaires pour le fonctionnement de l'application. Vous complèterez les fonctions au fur et à mesure du TP en respectant le format des résultats attendus tel qu'indiqué en en-tête de chaque fonction.

Les fichiers **index.php**, **inscription.php** et **chat.php** sont les pages de l'application qui seront affichées dans le navigateur. Ces pages seront complétées au fur et à mesure du TP en parallèle des fonctions des fichiers mentionnés précédemment.

1.3 Schéma de la base de données

L'application utilisera une base de données importable via le fichier **chat.sql**.

utilisateur(*pseudo, mdp, couleur, etat*)
message(*auteur, date, heure, valeur, type*)

Chaque utilisateur est identifié par un pseudo unique, un mot de passe, une couleur pour son affichage dans le fil de discussion et son état (connected/disconnected).

Un message est identifié par un auteur étant un pseudo d'utilisateur, une date (jour de l'année) et une valeur étant le contenu du message. Un typ *e est également associé à chaque message en fonction de son contenu.

1.4 Références et support

Pour les principales fonctions PHP nécessaires pour le TP :

— `http://php.net/manual/fr/langref.php`

— `http://php.net/manual/fr/book.mysqli.php`

Pour la structure de documents HTML5/CSS3 :

— `https://www.w3schools.com/html/html5_intro.asp`

— `https://www.w3schools.com/css/`

2 Connexion à la base de données

Afin de pouvoir mettre à jour des informations de la base de données MySQL, nous allons mettre en place 3 fonctions nous permettant de créer une connexion à la base de données, exécuter une requête SELECT et exécuter une requête de mise à jour (INSERT/UPDATE/DELETE).

TODO : Importer le fichier de création de la base données "chat" via phpmyadmin.

TODO : Entrer les identifiants de connexion à la base de données "chat" dans le fichier **bd.php**.

TODO : Implémenter les fonctions getConnection, executeQuery et executeUpdate du fichier **bd.php** tel qu'indiqué en en-tête.

3 Inscription et connexion d'un utilisateur

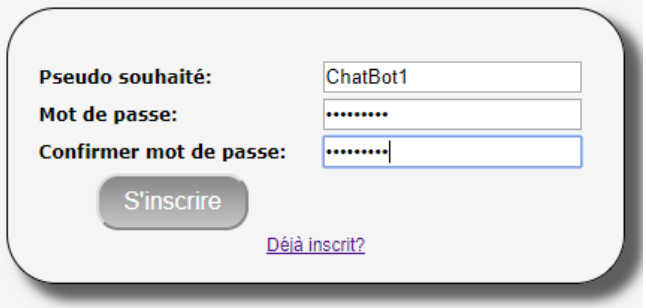

FIGURE 1 – Page d'inscription

3.1 Gestion de l'inscription

Maintenant que vous disposez de méthodes pour se connecter à la base de données et exécuter des requêtes, nous allons nous intéresser à l'inscription des utilisateurs.

TODO : Dans le fichier **utilisateur.php**, implémenter les fonctions checkAvailability et register tel qu'indiqué en en-tête.

TODO : Dans le fichier **inscription.php**, démarrer une session (cf. session PHP) et importer une fois (require_once) les fichiers **bd.php** et **utilisateur.php**.

TODO : Dans le fichier **inscription.php**, créer un formulaire HTML permettant d'entrer un pseudo souhaité, un mot de passe (non visible) et de retaper le mot de passe (non visible également) pour validation comme illustré sur l'exemple. Ce formulaire pourra être validé via un bouton et sera traité par la page **inscription.php** elle-même. La soumission se fera via méthode POST.

TODO : Dans le fichier **inscription.php**, lorsque le formulaire est soumis, récupérer le pseudo souhaité, le mot de passe et la confirmation de mot de passe. Si le mot de passe n'est pas égal à la confirmation, renvoyer un message d'erreur. Dans le cas contraire, si le pseudo est disponible, enregistrer l'utilisateur dans la base puis renvoyer le vers la page de connexion. Si le pseudo est déjà utilisé, renvoyer un message d'erreur.

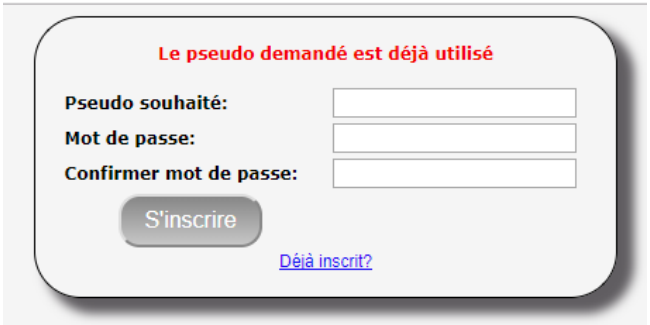

FIGURE 2 – Mot de passe non confirmé

TODO : En regardant dans la base de données, on s'aperçoit que les utilisateurs sont enregistrés avec leurs mots de passe en clair ce qui constitue une faille de sécurité majeure. Modifier la récupération du mot de passe dans le fichier **inscription.php** pour que le mot de passe soit crypté (hachage cryptographique MD5) dans la base de données.

3.2 Gestion de la connexion

Maintenant que des utilisateurs peuvent s'enregistrer, passons à la connexion.

TODO : Dans le fichier **utilisateur.php**, implémenter les fonctions getUser et setConnected tel qu'indiqué en en-tête.

TODO : Dans le fichier **index.php**, mettre en place un formulaire permettant d'entrer un login et un mot de passe (non visible). Ce formulaire pourra être soumis via un bouton de connexion comme illustré. Ce formulaire sera traité par la page **index.php** elle-même et soumis via méthode POST. De plus, rajouter un lien hypertexte permettant d'accéder à la page d'inscription.

FIGURE 3 – Page de connexion

TODO : Dans le fichier **index.phgp**, démarrer une session et importer une fois les fichiers **bd.php** et **utilisateur.php**.

TODO : Dans le fichier **index.php**, lorsque le formulaire est soumis, récupérer le pseudo et le mot de passe crypté. Si l'utilisateur existe, enregistrer le pseudo dans une variable de session (par ex. pseudo), changer l'état de l'utilisateur en "connected" et renvoyer l'utilisateur vers la page **chat.php**. Sinon, renvoyer un message d'erreur comme illustré ci-dessous.

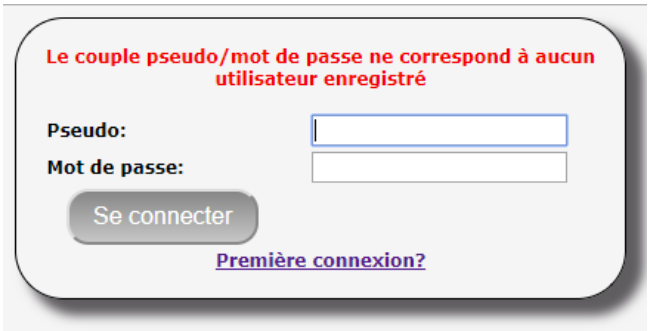

FIGURE 4 – Page de connexion

4 Affichage de l'historique

FIGURE 5 – Page principale du chat

4.1 Récupération des utilisateurs connectés

Une fois l'utilisateur connecté, nous allons lui fournir des informations sur l'activité récente.

TODO : Dans le fichier **utilisateur.php**, implémenter la fonction getConnectedUsers tel qu'indiqué en en-tête.

TODO : Dans le fichier **chat.php**, démarrer une session et importer une fois les fichier **bd.php**, **utilisateur.php** et **discussion.php**.

TODO : Dans le fichier **chat.php**, afficher un message de bienvenue pour l'utilisateur enregistré en variable de session dans une div.

TODO : Dans le fichier **chat.php**, récupérer la liste des utilisateurs connectés et afficher dans la div créée précédemment, la liste des utilisateurs connectés. Un tableau HTML permettra d'afficher un utilisateur par ligne.

4.2 Récupération et affichage des derniers messages

Passons désormais à l'affichage des messages dans le fil de discussion. Lorsque l'utilisateur se connecte, il serait intéressant d'afficher les derniers messages par ordre chronologique pour qu'il

puisse reprendre la conversation en cours. Toutefois, on ne va pas récupérer tous les messages de la base mais uniquement les n pus récents.

TODO : Dans le fichier **discussion.php**, fixer une valeur pour la variable historySize et implémenter la fonction getHistory tel qu'indiqué en en-tête.

TODO : Dans le fichier **chat.php**, créer une div dans laquelle vous appellerez du code PHP. Ce code récupèrera l'historique des messages et l'affichera tel que le message le plus ancien sera en haut de la page et le plus récent en bas de la page tel qu'illustré au-dessus. Chaque message sera affiché sur une ligne commençant par le pseudo de l'utilisateur, le contenu du message, la date et l'heure en italique.

5 Envoi d'un message et rafraichissement

L'historique des messages étant maintenant récupérable et affichable, il ne reste plus qu'à pouvoir ajouter des messages et rafraichir la page manuellement.

TODO : Dans le fichier **discussion.php**, implémenter la fonction submitMessage tel qu'indiqué en en-tête.

TODO : Dans le fichier **chat.php**, créer un formulaire HTML qui sera traité par la page **chat.php** elle-même et soumis via méthode POST. Ce formulaire contiendra un champ texte perGrâce aux balises natives d'HTML5, il est possible de faire de même pour des vidéos. mettant de saisir un nouveau message et un bouton 'Envoyer' permettant de soumettre le formulaire.

TODO : Dans le fichier **chat.php**, lorsque le formulaire est soumis, récupérer le message et le pseudo de l'utilisateur enregistré en variable de session et enregistrer le nouveau message dans la base de données.

Voilà, nous avons un mini-chat fonctionnel mais peu ergonomique car l'historique ne se met à jour qu'à l'envoi d'un nouveau message, ce qui est peu pratique pour discuter. En PHP, nous allons proposer un bouton de rafraichissement manuel.

TODO : Dans le fichier **chat.php**, rajouter un formulaire HTML qui sera traité par la page **chat.php** elle-même et sera soumis via méthode POST. Ce formulaire contiendra un unique bouton de soumission 'Rafraichir'. Lors de la soumission du formulaire, l'utilisateur sera redirigé vers la page **chat.php**.

6 Déconnexion

Il ne reste plus qu'à permettre aux utilisateurs de se déconnecter.

TODO : Dans le fichier **utilisateur.php**, implémenter la fonction setDisconnected tel qu'indiqué en en-tête.

TODO : Dans le fichier **chat.php**, rajouter un formulaire HTML qui sera traité par la page **chat.php** elle-même et sera soumis via méthode POST. Ce formulaire contiendra également un unique bouton de soumission 'Se déconnecter'. Lors de la soumission du formulaire, l'état de l'utilisateur sera modifié dans la base de données, toutes les variables de session seront supprimées, puis l'utilisateur sera redirigé vers la page de connexion.

7 Pour aller plus loin

Bravo, vous avez réalisé votre application fonctionnelle de chat en PHP avec support d'une base de données. De nombreux axes permettent d'améliorer cette application : le design, l'envoi de messages multimédias, l'ergonomie et la gestion de la navigation...

Voici quelques suggestions pour améliorer le chat :

7.1 Le design

TODO : Dans le fichier **utilisateur.php**, implémenter la fonction getUserColor tel qu'indiqué en en-tête. Dans le fichier **chat.php**, pour chaque message de l'historique, récupérer la couleur associée à l'auteur et rajouter 'style="color :*la couleur récupérée en PHP*"' à la balise contenant l'auteur du message.

TODO : Créer un fichier **style.css** dans le dossier contenant les pages de votre application. Importer ce fichier en temps que feuille de style CSS dans les fichiers **index.php**, **inscription.php** et **chat.php**.

TODO : Encapsuler, lorsque ce n'est pas le cas, les formulaires de connexion et d'inscription dans des balises div de class 'login'.

TODO : Dans le fichier **style.css**, définir les propriétés suivantes pour les balises div de classe login : une bordure de type 'solid', un padding de 2% et le texte aligné au centre.

TODO : Dans le fichier **chat.php**, encapsuler les div dans un tableau de telle sorte que la div contenant le message de bienvenue soit sur la gauche et la div contenant le chat soit sur la droite.

TODO : Attribuer à la div de gauche la classe 'left' et la div contenant le chat la classe 'right'. Rajouter dans le fichier **style.css** les instructions pour que ces div aient une bordure, un padding de 2em et un couleur de fond différente.

Cela définit un style de base que vous êtes libre de modifier et enrichir selon vos goûts.

7.2 L'envoi et l'affichage d'images

Nous allons proposer aux utilisateurs d'envoyer des images accessibles sur Internet en ne sauvegardant que le lien vers l'image. Cela suppose une connexion Internet active pour afficher l'image.

TODO : Dans le fichier **discussion.php**, implémenter la fonction submitImage tel qu'indiqué en en-tête.

TODO : Dans le fichier **chat.php**, lors de l'affichage de l'historique, distinguer le cas où le message est de type texte et image. Dans le second cas, encapsuler le lien de l'image (ou du gif) dans une balise img dont la longueur et de la largeur seront fixes (par exemple 100 et 100) pour afficher l'image dans le cadre de la page.

Grâce aux balises natives d'HTML5, il est possible de faire de même pour des vidéos.