Babillard de PostIts

A. Préparation

1. Transformez les fichiers HTML en fichiers PHP. Valider le fonctionnement de l'application en ouvrant les pages dans votre navigateur (le serveur se trouve à l'adresse: http://localhost:8000/)
2. Dans PhpMyAdmin (http://localhost:8080), importez - à la base `mydatabase` - le fichier `db/db-structure.sql` puis le fichier `db/db-data.sql`
3. Créez un fichier `config.php` qui démarre la session et crée la connexion à la base de données en PDO. Pour vous connecter à la base de données, utilisez le code suivant:

```php
//Configuration et connexion à la base de données
$host = 'db';
$db   = 'mydatabase';
$user = 'user';
$pass = 'password';
$charset = 'utf8mb4';

$dsn = "mysql:host=$host;dbname=$db;charset=$charset";
$options = [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES   => false,
];

try {
    $pdo = new PDO($dsn, $user, $pass, $options);
} catch (\PDOException $e) {
    throw new \PDOException($e->getMessage(), (int)$e->getCode());
}
```

4. Inclure le fichier `config.php` dans toutes les pages de l'application.

B. Requêtes SQL

1. Créez un fichier `test.php` qui affiche (en format textuel) le titre et le contenu du PostIt ayant pour identifiant 1.
2. Modifiez le fichier `test.php` pour qu'il insère un nouveau PostIt dans la base de données à partir de données (de votre choix) préalablement stockées dans des variables.

C. Création de compte

1. Modifiez le fichier `nouvelUsager.php` pour que les données du formulaires soient envoyées à la même page en utilisant la méthode `POST`.
2. Au début du fichier, ajoutez le code nécessaire au traitement des données reçues avec la méthode `POST` pour insérer le nouvel usager dans la table `users`. Le mot de passe ne doit pas être stocké en clair dans la base de données. Utilisez la fonction PHP `passwordHash` pour encoder le mot de passe. Par exemple:

   ```
   // Création d'un hash du mot de passe en utilisant l'algorithme de hachage par défaut
   $passwordHash = password_hash($password, PASSWORD_DEFAULT);
   ```

3. Si le nom d'utilisateur existe déjà, l'insertion de l'usager echéoura (contrainte d'unicité). Traitez cette erreur en affichant un message d'erreur dans la balise `div` prévue pour cet effet. Le nom d'usager (erronné) déjà saisi doit apparaitre dans le `input` correspondant.
4. Si l'ajout du nouvel usager a fonctionné, rediriger vers la page d'authentification. Pour rediriger vers une page, ajoutez une en-tête à la réponse HTTP avec la fonction `header`:

   ```
   header("Location: page_de_redirection.php");
   exit;
   ```

D. Authentification

1. Modifiez la page de login pour traiter les données d'authentification reçues par la méthode `POST`
2. Si le couple nom d'utilisateur et mot de passe sont valides, ajouter le champs `usager` à la session et rediriger l'usager vers la page d'acceuil de l'application (`index.php`)
3. Si l'authentification échoue, affichez un message d'erreur dans la balise `div` prévue à cet effet et pré-remplir le nom d'usager reçu dans le champs texte correspondant.

E. Page du babillard

1. Modifiez vos scripts pour que la page `index.php` ne s'affiche que si la personne est authentifiée. Si ce n'est pas le cas, l'usager doit être redirigé vers la page d'authentification.
2. Présentement, les PostIts affichés sont prédéfinis dans une variable Javascript `postIts`. Modifiez le code de sorte que le contenu de cette variable provienne de la base de données et que seuls les PostIts de la personne authentifiée soient affichés.