
TP : Création d'une Interface Web et Gestion des Utilisateurs

Dans ce tutoriel, nous allons mettre en place un **système d'inscription et de connexion sécurisé** en PHP. Nous utiliserons **PDO (PHP Data Objects)**, une manière simple et efficace de communiquer avec la base de données.

1. Organisation du Projet

Avant de commencer, structurons notre projet :

```
/projet_pdo
├── /css
│   └── style.css      → Styles de l'interface
├── /includes
│   └── config.php     → Connexion à la base de données
├── /pages
│   ├── register.php  → Page d'inscription
│   ├── login.php     → Page de connexion
│   └── dashboard.php → Interface après connexion
├── index.php         → Page d'accueil
└── logout.php        → Déconnexion
```

Cette structure permet de séparer les fichiers et de rendre le code plus organisé.

2. Connexion à la Base de Données

Nous allons stocker la connexion dans un fichier **config.php** pour pouvoir la réutiliser dans toutes nos pages.

📌 Fichier : **/includes/config.php**

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "testdb"; // Nom de la base de données
```

```
try {  
    $pdo = new PDO("mysql:host=$servername;dbname=$dbname", $username,  
$password);  
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
    $pdo->exec("SET NAMES utf8");  
} catch (PDOException $e) {  
    die("Erreur de connexion : " . $e->getMessage());  
}  
?>
```

✓ Explication :

- **PDO** permet de se connecter à la base sans besoin de connaître SQL.
- **try...catch** permet de capturer les erreurs et d'éviter que le site plante.
- **On encode les données en UTF-8** pour éviter les problèmes d'affichage.

3. Création de la Page d'Inscription

Nous allons créer un formulaire pour que les utilisateurs puissent s'inscrire.

📁 Fichier : **/pages/register.php**

```
<?php  
include '../includes/config.php';  
  
$message = "";  
  
// Vérifier si le formulaire est soumis  
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    $name = htmlspecialchars($_POST['name']);  
    $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);  
    $password = password_hash($_POST['password'], PASSWORD_DEFAULT); // Hachage  
sécurisé  
  
    // Vérifier si l'email existe déjà  
    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");  
    $stmt->execute([$email]);  
  
    if ($stmt->rowCount() > 0) {  
        $message = "Cet email est déjà utilisé.";  
    } else {  
        $stmt = $pdo->prepare("INSERT INTO users (name, email, password) VALUES (?, ?,  
?);");
```

```
if ($stmt->execute([$name, $email, $password])) {  
    $message = "Inscription réussie ! <a href='login.php'>Connectez-vous</a>";  
} else {  
    $message = "Erreur lors de l'inscription.";  
}  
}  
}  
?>
```

```
<!DOCTYPE html>  
<html lang="fr">  
<head>  
    <title>Inscription</title>  
    <link rel="stylesheet" href="../css/style.css">  
</head>  
<body>  
    <h2>Inscription</h2>  
    <form method="post">  
        <input type="text" name="name" placeholder="Nom" required><br>  
        <input type="email" name="email" placeholder="Email" required><br>  
        <input type="password" name="password" placeholder="Mot de passe" required><br>  
        <button type="submit">S'inscrire</button>  
    </form>  
    <p><?php echo $message; ?></p>  
</body>  
</html>
```

✓ Explication :

- L'utilisateur remplit un formulaire.
- Le mot de passe est **sécurisé avec password_hash()**.
- On vérifie si l'email existe déjà pour éviter les doublons.

4. Création de la Page de Connexion

Après l'inscription, l'utilisateur doit pouvoir se connecter.

📌 Fichier : **/pages/login.php**

```
<?php  
include '../includes/config.php';  
session_start();
```

```
$message = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
    $password = $_POST['password'];

    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
    $stmt->execute([$email]);
    $user = $stmt->fetch(PDO::FETCH_ASSOC);

    if ($user && password_verify($password, $user['password'])) {
        $_SESSION['user'] = $user['name'];
        header("Location: dashboard.php");
        exit();
    } else {
        $message = "Email ou mot de passe incorrect.";
    }
}

?>

<!DOCTYPE html>
<html lang="fr">
<head>
    <title>Connexion</title>
    <link rel="stylesheet" href="../css/style.css">
</head>
<body>
    <h2>Connexion</h2>
    <form method="post">
        <input type="email" name="email" placeholder="Email" required><br>
        <input type="password" name="password" placeholder="Mot de passe" required><br>
        <button type="submit">Se connecter</button>
    </form>
    <p><?php echo $message; ?></p>
</body>
</html>
```

✓ Explication :

- L'utilisateur entre son email et son mot de passe.
- On vérifie le mot de passe avec **password_verify()**.
- Si c'est correct, il est redirigé vers son tableau de bord.

5. Création du Tableau de Bord

Une fois connecté, l'utilisateur doit accéder à une page protégée.

 **Fichier : /pages/dashboard.php**

```
<?php
session_start();
if (!isset($_SESSION['user'])) {
    header("Location: login.php");
    exit();
}
?>

<!DOCTYPE html>
<html lang="fr">
<head>
    <title>Tableau de Bord</title>
    <link rel="stylesheet" href="../css/style.css">
</head>
<body>
    <h2>Bienvenue, <?php echo $_SESSION['user']; ?> !</h2>
    <a href="../logout.php">Déconnexion</a>
</body>
</html>
```

 **Explication :**

- On vérifie si l'utilisateur est connecté.
- Si non, il est redirigé vers la page de connexion.

6. Gestion de la Déconnexion

 **Fichier : /logout.php**

```
<?php
session_start();
session_destroy();
header("Location: pages/login.php");
exit();
?>
```

✓ Explication :

- On **détruit la session** pour déconnecter l'utilisateur.

7. Gestion des Erreurs

Nous allons **ajouter un fichier de logs** pour enregistrer les erreurs.

📌 Modification du **config.php**

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "testdb";

try {
    $pdo = new PDO("mysql:host=$servername;dbname=$dbname", $username,
    $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $pdo->exec("SET NAMES utf8");
} catch (PDOException $e) {
    file_put_contents('error_log.txt', date('Y-m-d H:i:s') . " - Erreur PDO : " . $e->getMessage()
    . "\n", FILE_APPEND);
    die("Une erreur est survenue. Veuillez contacter l'administrateur.");
}
?>
```

✓ Explication :

- **Les erreurs sont enregistrées** dans **error_log.txt**.
- L'utilisateur **ne voit pas les détails techniques des erreurs**, ce qui est plus sécurisé.

🎯 Conclusion

Nous avons appris à :

- ✓ Créer une interface utilisateur en PHP
- ✓ Gérer un système de connexion sécurisé
- ✓ Protéger les données des utilisateurs

À Vous de Jouer !

Vous avez maintenant les bases pour gérer une interface utilisateur avec un système sécurisé d'inscription et de connexion.

Devoir à Rendre

Votre mission :

1 Ajoutez une fonctionnalité de récupération de mot de passe

- Indice : Utilisez une page où l'utilisateur entre son email.
- Envoyez un lien unique pour réinitialiser le mot de passe.
- Stockez un jeton temporaire dans la base de données.


2 Créez un espace administrateur

- Indice : Ajoutez une colonne **role** dans la table **users** (ex. **user** ou **admin**).
- Sécurisez l'accès avec une vérification du rôle après la connexion.
- Permettez à l'admin de voir et supprimer des utilisateurs.

Instructions de soumission

Chaque étudiant doit :

1. Soumettre un **rapport en PDF** contenant :
 - Captures d'écran du code PHP et des interfaces commentés.
2. Envoyer un e-mail avec l'objet : **Nom-Prénom-Filière - The Assignment**.
 - Exemple : **Qassimi-Sara-IRISI 1 - PHP BD**
3. Joindre une **pièce jointe** à l'e-mail.
4. Envoyer l'e-mail à : **sara.qassimi@uca.ac.ma**

 **Date limite** : La semaine prochaine.