



Exercice PHP – Gestionnaire de profil utilisateur (sans base de données)



Objectifs

- Maîtriser les formulaires en PHP
- Gérer des fichiers (uploads, JSON)
- Utiliser les sessions pour gérer la connexion
- Sécuriser les données utilisateurs



Consignes

1. Page de connexion (`login.php`)

- Un formulaire avec un **champ "identifiant"** (ex : "jdupont").
- À la soumission :
 - Vérifier que l'identifiant est alphanumérique.
 - Sauvegarder l'identifiant dans la **session**.
 - Rediriger vers `profil.php`.

2. Page de gestion de profil (`profil.php`)

- Vérifier que l'utilisateur est bien connecté via la session.
- Afficher un **formulaire** avec :
 - Prénom (input text)
 - Bio (textarea)
 - Photo de profil (fichier image)
- À la soumission :
 - Sécuriser les champs (`htmlspecialchars`)
 - Valider et uploader la photo (images uniquement, max 2 Mo)
 - Sauvegarder les données dans un fichier `users.json` (format donné ci-dessous)

3. Format du fichier `users.json`

Voici un exemple de format possible pour le json:

```
{
  "jdupont": {
    "prenom": "Jean",
    "bio": "Développeur web junior.",
    "photo": "../uploads/jdupont.jpg"
  },
  "amartin": {
    "prenom": "Alice",
    "bio": "Étudiante en web mobile.",
    "photo": "../uploads/amartin.png"
  }
}
```

```
}  
}
```

4. Améliorations Bonus (facultatives)

Ces éléments permettent de renforcer la sécurité et de préparer les apprenants à des concepts plus avancés :

- **Ajout d'un mot de passe à la connexion**
 - Ajouter un champ "mot de passe" dans `login.php`.
 - Enregistrer le mot de passe haché avec `password_hash()` dans `users.json`.
 - Vérifier le mot de passe avec `password_verify()` lors de la connexion.
- **Protection contre les attaques CSRF**
 - Générer un token CSRF et le stocker en session.
 - Ajouter un champ caché contenant ce token dans tous les formulaires.
 - Vérifier que le token soumis correspond à celui de la session.
- **Ajout d'un captcha simple**
 - Générer un code (texte ou calcul simple) stocké en session.
 - L'utilisateur doit saisir le résultat dans un champ du formulaire.
 - Comparer les valeurs lors de la soumission.
- **Validation côté serveur + client**
 - Côté serveur :
 - Vérifier les types MIME des fichiers (`mime_content_type()`).
 - Vérifier les formats des emails (`filter_var()` avec `FILTER_VALIDATE_EMAIL`).
 - Côté client :
 - Ajouter des attributs `required`, `pattern`, etc. dans le HTML.
 - Ajouter de la validation JavaScript pour guider l'utilisateur.
- **Journalisation des connexions**
 - À chaque connexion réussie, écrire une ligne dans un fichier `log.txt` :
 - date et heure
 - identifiant
 - IP de l'utilisateur (`$_SERVER['REMOTE_ADDR']`)