Introduction Approfondie à PHP pour le Développement Web

Abdelweheb GUEDDES

11 avril 2025

Séance 8 : Sessions et Cookies en PHP

Introduction aux Sessions

Concept des Sessions Les sessions en PHP sont un mécanisme permettant de conserver des données *côté serveur* tout au long de la navigation d'un utilisateur sur votre site web. Contrairement aux cookies qui sont stockés chez le client, les sessions gardent les informations sensibles (comme l'ID utilisateur, les droits d'accès) sur le serveur, améliorant la sécurité. Elles permettent de maintenir un état (par exemple, le statut de connexion de l'utilisateur, les articles dans son panier).

Fonctionnement des Sessions

1. Lorsqu'un script PHP appelle `session_start()` pour la première fois pour un visiteur ou si aucun cookie de session valide n'est envoyé par le navigateur, PHP génère un identifiant de session unique (Session ID ou SID).
2. Par défaut, cet ID est envoyé au navigateur sous forme de cookie (nommé généralement `PHPSESSID`). Le navigateur renverra ce cookie à chaque requête ultérieure vers le même domaine. PHP peut aussi être configuré pour passer l'ID dans l'URL, mais c'est moins sécurisé et déconseillé.
3. Sur le serveur, PHP crée un fichier (ou utilise un autre mécanisme de stockage configuré) associé à cet ID de session pour y stocker les données.

4. Vous pouvez lire et écrire des données dans la session en utilisant le tableau superglobal `$_SESSION`. Ces données sont sérialisées et enregistrées dans le fichier de session à la fin de l'exécution du script.
5. Lors des requêtes suivantes du même utilisateur, si le cookie de session est présent et valide, `session_start()` récupère l'ID, ouvre le fichier de session correspondant et désérialise les données dans le tableau `$_SESSION`, rendant les informations précédentes disponibles.
6. Pour mettre fin à une session (déconnexion), on utilise généralement `session_unset()` pour effacer les variables de session, puis `session_destroy()` pour supprimer le fichier de session côté serveur. Il est aussi recommandé de supprimer le cookie de session côté client.

Manipulation des Sessions

Démarrer ou Reprendre une Session La fonction `session_start()` doit être appelée **au tout début** de votre script PHP, avant toute sortie HTML ou texte (espaces, lignes vides, `echo`, etc.), sauf si la bufferisation de sortie est activée. Elle initialise la session ou reprend une session existante basée sur l'ID reçu (cookie ou URL).

```
<?php
// TOUT EN HAUT DU FICHIER, AVANT TOUT AUTRE CODE OU HTML
session_start();
?>
<!DOCTYPE html>
<html>
<head>
    <title>Page avec Session</title>
</head>
<body>
    <?php // Le reste du code PHP peut venir ici ?>
</body>
</html>
```

Création/Modification de Variables de Session Une fois la session démarrée, utilisez le tableau superglobal `$_SESSION` (qui est un tableau associatif) pour stocker ou modifier des données.

```
<?php
session_start();

```

```php
// Stocker des informations
$_SESSION['utilisateur_id'] = 123;
$_SESSION['utilisateur_nom'] = "ali";
$_SESSION['role'] = "admin";
$_SESSION['derniere_visite'] = time(); // Timestamp de la dernière activité

// Modifier une valeur
$_SESSION['role'] = "moderateur";

// Ajouter un élément à un tableau en session
if (!isset($_SESSION['panier'])) {
    $_SESSION['panier'] = []; // Initialiser si n'existe pas
}
$_SESSION['panier'][] = 'Produit A'; // Ajoute un produit

echo "Variable 'utilisateur_nom' enregistrée.";
?>
```

Lecture de Variables de Session Accédez aux variables via leurs clés dans le tableau `$_SESSION`. Il est prudent de vérifier leur existence avec `isset()` avant de les utiliser.

```php
<?php
session_start();

if (isset($_SESSION['utilisateur_nom'])) {
    echo "Bonjour, " . htmlspecialchars($_SESSION['utilisateur_nom']) . " ! <br>";
} else {
    echo "Bonjour, visiteur anonyme !<br>";
}

if (isset($_SESSION['role']) && $_SESSION['role'] == "admin") {
    echo "Vous disposez des droits d'administrateur.";
}

if (isset($_SESSION['panier'])) {
    echo "Votre panier contient : " . count($_SESSION['panier']) . " article(s).";
}
?>
```

Suppression de Variables de Session Utilisez `unset()` pour supprimer une variable spécifique de la session.

```php
<?php
session_start();

// Supprimer une seule variable
unset($_SESSION['role']);

// Vérifier si elle existe encore (elle ne devrait plus)
if (!isset($_SESSION['role'])) {
    echo "La variable de session 'role' a été supprimée.";
}
?>
```

Destruction Complète d'une Session Pour terminer une session (déconnexion), suivez ces étapes :

1. Démarrez la session avec `session_start()` (obligatoire pour manipuler la session courante).
2. Supprimez toutes les variables de session avec `session_unset()`.
3. Détruisez la session côté serveur avec `session_destroy()`. Cela supprime le fichier de session.
4. (Recommandé) Supprimez le cookie de session côté client pour éviter qu'il ne soit renvoyé inutilement.

```php
<?php
session_start(); // 1. Démarrer la session

session_unset(); // 2. Supprimer toutes les variables de session

session_destroy(); // 3. Détruire la session côté serveur

// 4. Supprimer le cookie de session côté client (optionnel mais recommandé)
if (ini_get("session.use_cookies")) {
    setcookie(session_name(), '', time() - 42000, // Expiration dans le passé
    );
}

echo "Session détruite et cookie supprimé.";
```

```php
// Souvent suivi d'une redirection vers la page de connexion ou d'accueil
// header('Location: login.php');
// exit();
?>
```

Introduction aux Cookies

Concept des Cookies Les cookies sont de petits fichiers textes (ou des données) stockés par le navigateur web sur l'ordinateur de l'utilisateur, à la demande d'un serveur web. Ils permettent de conserver des informations sur le visiteur entre différentes visites ou pages vues (par exemple, préférences de langue, identifiant pour se souvenir de la connexion, suivi basique). Les cookies ont une date d'expiration et sont renvoyés automatiquement par le navigateur au serveur à chaque requête vers le domaine concerné.

Fonctionnement des Cookies

1. Le serveur envoie un cookie au navigateur via l'en-tête HTTP `Set-Cookie`.
2. Le navigateur enregistre ce cookie (nom, valeur, date d'expiration, domaine, chemin, etc.).
3. Lors des visites ultérieures sur le même domaine/chemin, avant la date d'expiration, le navigateur inclut automatiquement les cookies correspondants dans l'en-tête HTTP `Cookie` de la requête envoyée au serveur.
4. Le serveur peut alors lire ces informations via la superglobale `$_COOKIE`.

Attention : Les données des cookies sont stockées chez l'utilisateur et peuvent être lues ou modifiées par lui. Ne stockez jamais d'informations sensibles (mots de passe, etc.) directement dans les cookies.

Manipulation des Cookies

Création/Modification d'un Cookie Pour créer ou modifier un cookie, utilisez la fonction `setcookie()`. Elle doit être appelée **avant toute sortie HTML**, tout comme `session_start()`. Syntaxe de base : `setcookie(nom, valeur, expiration, chemin, domaine, secure, httponly)`

— `nom` (string) : Le nom du cookie (obligatoire).

— `valeur` (string) : La valeur du cookie. Généralement encodée si elle contient des caractères spéciaux. (obligatoire).
— `expiration` (int, timestamp) : Date d'expiration du cookie (en secondes depuis l'Epoch Unix). Si 0 ou omis, le cookie expire à la fin de la session du navigateur (cookie de session). Utiliser `time() + nb_secondes`.
— `chemin` (string, optionnel) : Chemin sur le serveur où le cookie sera disponible (par défaut '/').
— `domaine` (string, optionnel) : Domaine pour lequel le cookie est valide (par défaut, le domaine courant).
— `secure` (bool, optionnel) : Si `true`, le cookie ne sera transmis que sur une connexion HTTPS.
— `httponly` (bool, optionnel) : Si `true`, le cookie ne sera pas accessible via JavaScript (`document.cookie`), aidant à prévenir les attaques XSS. Fortement recommandé.

```php
<?php
 // Cookie valable 1 an
 $nom_cookie = 'langue';
 $valeur_cookie = 'fr';
 $expiration = time() + 365 * 24 * 3600; // timestamp dans 1 an
 $chemin = '/'; // Disponible sur tout le site
 $domaine = ''; // Domaine courant (ou spécifiez ex: '.example.com')
 $secure = false; // Mettre true si site en HTTPS uniquement
 $httponly = true; // Recommandé

 setcookie($nom_cookie, $valeur_cookie, $expiration, $chemin, $domaine, $secure, $httponly);

 // Cookie qui expire à la fermeture du navigateur
 setcookie('visite_session', 'oui', 0, '/', '', false, true);

 // Modification d'un cookie existant (même nom, chemin, domaine)
 setcookie('langue', 'en', $expiration, $chemin, $domaine, $secure, $httponly);
?>
<!DOCTYPE html>
<html>
<head><title>Cookies</title></head>
<body>
```

```php
<?php
echo "Cookie 'langue' envoyé/mis à jour (valeur 'en').";
?>
</body>
</html>
```

Note : La valeur d'un cookie n'est disponible dans `$_COOKIE` qu'à la requête *suivante*, pas immédiatement après `setcookie()`.

Lecture d'un Cookie Les cookies envoyés par le navigateur sont disponibles dans le tableau associatif superglobal `$_COOKIE`. Vérifiez toujours leur existence avec `isset()` avant utilisation.

```php
<?php
// Ce code lira les cookies définis lors d'une requête PRECEDENTE

if (isset($_COOKIE['langue'])) {
    $langue_preferee = htmlspecialchars($_COOKIE['langue']); // Sécurité !
    echo "Votre langue préférée (lue depuis le cookie) est : " . $langue_preferee ;
} else {
    echo "Le cookie 'langue' n'a pas été trouvé.";
}

if (isset($_COOKIE['visite_session'])) {
     echo "<br>Cookie de session 'visite_session' détecté.";
}
?>
```

Suppression d'un Cookie Pour supprimer un cookie, vous devez utiliser `setcookie()` avec le même nom, chemin et domaine que lors de sa création, mais en lui donnant une date d'expiration *dans le passé* et éventuellement une valeur vide.

```php
<?php
 // Assurez-vous d'utiliser les mêmes paramètres (chemin, domaine) que lors de la création
 $nom_cookie = 'langue';
 $chemin = '/';
 $domaine = '';

```

```php
// Mettre une date d'expiration passée (ex: il y a une heure)
setcookie($nom_cookie, '', time() - 3600, $chemin, $domaine);

echo "Tentative de suppression du cookie '$nom_cookie'. Il disparaîtra à la prochaine requête.";

// Pour vérifier après rechargement de la page :
// if (!isset($_COOKIE['langue'])) { echo "Cookie supprimé."; }
?>
```

Exercice Pratique Guidé (TP8)

Prérequis

Pour réaliser cet exercice, vous devez disposer d'une base de données et y avoir créé une table `utilisateurs`. Assurez-vous également que l'extension PDO pour votre type de base de données (ex : `pdo_mysql`) est activée dans votre configuration PHP (`php.ini`).

Structure de la Table `utilisateurs` (Exemple) Créez une table avec au moins les colonnes suivantes dans votre base de données (nommez la base `cours_php` par exemple) :

```
CREATE TABLE utilisateurs (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(100) NOT NULL,
    email VARCHAR(150) NOT NULL UNIQUE,
    mot_de_passe VARCHAR(255) NOT NULL, -- Important : pour MDP HACHÉS !
    age INT NULL
);
```

Insertion d'un Utilisateur de Test Insérez au moins un utilisateur dans cette table. **Crucial :** Ne stockez JAMAIS un mot de passe en clair. Utilisez la fonction `password_hash()` de PHP pour le hacher avant de l'insérer.

Exemple de script PHP (à exécuter une fois) pour insérer un utilisateur de test :

```php
<?php
// --- Configuration et Connexion PDO (Adaptez !) ---
$dsn = 'mysql:host=localhost;dbname=cours_php;charset=utf8';
$db_user = 'votre_utilisateur_bdd'; // Remplacez par votre user BDD
$db_pass = 'votre_mot_de_passe_bdd'; // Remplacez par votre mot de passe BDD
$options = [ PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION ];

try {
    $pdo = new PDO($dsn, $db_user, $db_pass, $options);
} catch (PDOException $e) {
    die('Erreur de connexion BDD : ' . $e->getMessage());
}

// --- Données du nouvel utilisateur ---
$nomUser = "Ali Ben Salah";
$emailUser = "ali.bensalah@test.com";
$ageUser = 25; // L'âge peut être NULL si non fourni
$motDePasseClair = "motdepasse123"; // Le mot de passe choisi par l'utilisateur

// --- Hachage du mot de passe ---
$motDePasseHache = password_hash($motDePasseClair, PASSWORD_DEFAULT);

// --- Insertion en base via Requête Préparée ---
try {
    $sql = "INSERT INTO utilisateurs (nom, email, mot_de_passe, age)
            VALUES (:nom, :email, :mdp, :age)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([
        ':nom' => $nomUser,
        ':email' => $emailUser,
        ':mdp' => $motDePasseHache,
        ':age' => $ageUser
    ]);
    echo "Utilisateur '" . htmlspecialchars($nomUser) . "' inséré avec succès (si email non existant).";

} catch (PDOException $e) {
    // Gérer l'erreur (ex: email déjà existant)
    if ($e->getCode() == 23000) { // Code d'erreur pour violation de contrainte d'unicité (approx)
```

```php
        echo "Erreur : L'email '" . htmlspecialchars($emailUser) . "' existe déjà.";
    } else {
        echo "Erreur lors de l'insertion : " . $e->getMessage();
    }
}

$pdo = null; // Fermer la connexion
?>
```

Assurez-vous que cet utilisateur existe avant de continuer le TP.

Objectif du TP

Créer une page PHP nommée `tp8.php` qui simule la récupération d'un utilisateur depuis la base de données et l'utilisation de ses informations dans une session et un cookie. La page devra :

1. Se connecter à la base de données `cours_php` via PDO.
2. Récupérer les informations (id, nom, age) d'un utilisateur spécifique (par exemple, celui avec l'email `ali.bensalah@test.com`) depuis la table `utilisateurs`. *Note : Pour cet exercice, nous ne vérifions pas le mot de passe, nous simulons juste qu'il est correct.*
3. Si l'utilisateur est trouvé :
   - Démarrer une session PHP.
   - Enregistrer l'ID, le nom et l'âge de l'utilisateur récupéré dans des variables de session (`$_SESSION`).
   - Afficher un message de bienvenue utilisant le nom et l'âge stockés en session.
   - Créer un cookie nommé `couleurPreferee` avec une valeur de votre choix (ex : 'Bleu') et une expiration de 30 minutes.
   - Afficher la valeur du cookie `couleurPreferee` si elle existe.
   - Afficher un lien permettant de "se déconnecter" (qui détruira la session).
4. Si l'utilisateur n'est pas trouvé, afficher un message d'erreur.
5. Gérer la déconnexion : si un paramètre `action=logout` est présent dans l'URL, détruire la session et afficher un message de confirmation.

Instructions

1. Créez le fichier `tp8.php`.
2. Au début du fichier, mettez en place la connexion PDO à votre base de données `cours_php` (avec gestion des erreurs `try...catch`). Utilisez les identifiants que vous avez configurés (`votre_utilisateur_bdd`, `votre_mot_de_passe_bdd`). Définissez les options PDO recommandées (`PDO::ATTR_ERRMODE`, `PDO::ATTR_DEFAULT_FETCH_MODE`).
3. Ajoutez la logique pour gérer la déconnexion : vérifiez `isset($_GET['action'])` et si sa valeur est `'logout'`. Si oui, démarrez la session (`session_start()`), détruisez-la (`session_unset()`, `session_destroy()`), supprimez le cookie de session (voir exemple plus haut), supprimez aussi le cookie `couleurPreferee`, affichez un message et terminez le script avec `exit()`.
4. Définissez l'email de l'utilisateur à rechercher (ex : `$email_a_chercher = 'ali.bensalah@test.com';`).
5. Préparez et exécutez une requête SQL `SELECT id, nom, age FROM utilisateurs WHERE email = :email` en utilisant cet email. Utilisez `bindParam` ou passez un tableau à `execute`.
6. Récupérez le résultat avec `$statement->fetch()`.
7. Vérifiez si `$utilisateur` contient des données (n'est pas `false`).
8. **Si oui** (`if ($utilisateur)`) :
   — Appelez `session_start()` **ici**.
   — Stockez les données dans `$_SESSION` : `$_SESSION['user_id'] = $utilisateur['id'];`, `$_SESSION['user_nom'] = $utilisateur['nom'];`, etc. Gérez le cas où l'âge est `NULL`.
   — Affichez le message de bienvenue en utilisant `htmlspecialchars()` sur les données de session.
   — Utilisez `setcookie()` pour créer le cookie `couleurPreferee` (valeur 'Bleu', expiration dans 30 min, httponly).
   — Vérifiez `isset($_COOKIE['couleurPreferee'])` et affichez sa valeur si présente (elle ne sera visible qu'au prochain chargement de page).
   — Affichez le lien `<a href="tp8.php?action=logout">Se déconnecter</a>`.
9. **Sinon** (`else`) :
   — Affichez un message indiquant que l'utilisateur n'a pas été trouvé.

10. À la fin du script (ou après l'utilisation de PDO), fermez la connexion : `$pdo = null;`.
11. Testez en chargeant `tp8.php`. Vérifiez l'affichage, la création du cookie (Outils de développement du navigateur > Application/Stockage > Cookies), puis cliquez sur le lien de déconnexion et rechargez la page pour voir l'effet.

Code Complet (Exemple pour `tp8.php`)

```php
<?php
// --- Configuration Base de Données (Adaptez !) ---
$dsn = 'mysql:host=localhost;dbname=cours_php;charset=utf8';
$db_user = 'votre_utilisateur_bdd';
$db_pass = 'votre_mot_de_passe_bdd';
$options = [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    // Erreurs en exceptions
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    // Résultats en tableau associatif
    PDO::ATTR_EMULATE_PREPARES   => false,
    // Vraies requêtes préparées
];

// --- Connexion PDO ---
try {
     $pdo = new PDO($dsn, $db_user, $db_pass, $options);
} catch (PDOException $e) {
     error_log("Erreur Connexion BDD: " . $e->getMessage());
    // Log l'erreur
     die("Erreur critique: Impossible de se connecter à la
    base de données."); // Message générique pour l'
    utilisateur
}

// --- Gestion de la Déconnexion ---
if (isset($_GET['action']) && $_GET['action'] === 'logout') {
    session_start(); // Nécessaire pour accéder et détruire
    la session
    session_unset(); // Efface les variables $_SESSION
    session_destroy(); // Détruit les données de session côté
    serveur

    // Supprimer le cookie de session côté client
```

```php
    if (ini_get("session.use_cookies")) {
        $params = session_get_cookie_params();
        setcookie(session_name(), '', time() - 42000,
            $params["path"], $params["domain"],
            $params["secure"], $params["httponly"]
        );
    }
    // Supprimer notre cookie personnalisé
    setcookie('couleurPreferee', '', time() - 3600, '/'); // Chemin '/' important

    // Message et fin du script
    echo "<p>Vous avez été déconnecté avec succès.</p>";
    echo '<p><a href="tp8.php">Retour</a></p>';
    $pdo = null; // Fermer connexion BDD
    exit();
}

// --- Tentative de récupération de l'utilisateur ---
$user_email_a_chercher = "ali.bensalah@test.com"; // Email de l'utilisateur à simuler

try {
    $sql = "SELECT id, nom, age FROM utilisateurs WHERE email = :email LIMIT 1";
    $statement = $pdo->prepare($sql);
    $statement->execute([':email' => $user_email_a_chercher]);
    $utilisateur = $statement->fetch(); // Récupère l'utilisateur ou false

    if ($utilisateur) {
        // --- Utilisateur trouvé : Démarrer la session et stocker les infos ---
        session_start(); // Démarrer la session ICI, APRES avoir trouvé l'utilisateur

        // Stocker les informations en session
        $_SESSION['user_id'] = $utilisateur['id'];
        $_SESSION['user_nom'] = $utilisateur['nom'];
        $_SESSION['user_age'] = $utilisateur['age'] ?? 'Non spécifié'; // Opérateur Null Coalescent (PHP 7+)

        // --- Affichage des informations de session ---
        echo "<h1>Bienvenue !</h1>";
```

```
64          echo "<p>Bonjour, " . htmlspecialchars($_SESSION['
    user_nom']) . ". ";
65          echo "Vous avez " . htmlspecialchars($_SESSION['user_
    age']) . " ans.</p>";
66          echo "<p>(ID Utilisateur : " . htmlspecialchars($_
    SESSION['user_id']) . ")</p>";
67
68          // --- Gestion du Cookie 'couleurPreferee' ---
69          $nom_cookie_couleur = 'couleurPreferee';
70          $valeur_cookie_couleur = 'Bleu'; // Valeur par défaut
71          $expiration_cookie = time() + 30 * 60; // Cookie
    valable 30 minutes
72          setcookie($nom_cookie_couleur, $valeur_cookie_couleur
    , $expiration_cookie, '/', '', false, true);
73
74          // Afficher la valeur du cookie (si déjà présent lors
     de cette requête)
75          if (isset($_COOKIE[$nom_cookie_couleur])) {
76              echo "<p>Votre couleur préférée (cookie) est : "
     . htmlspecialchars($_COOKIE[$nom_cookie_couleur]). "</p>"
    ;
77          } else {
78              // Le cookie vient d'être envoyé, il sera
    lisible au prochain chargement
79              echo "<p>Le cookie de couleur vient d'être dé
    fini à '".htmlspecialchars($valeur_cookie_couleur)."'.
    Rechargez la page pour le voir.</p>";
80          }
81
82          // --- Lien de Déconnexion ---
83          echo '<p><a href="tp8.php?action=logout">Se dé
    connecter</a></p>';
84
85      } else {
86          // --- Utilisateur non trouvé ---
87          echo "<h1>Accès Refusé</h1>";
88          echo "<p>Utilisateur avec l'email '" .
    htmlspecialchars($user_email_a_chercher) . "' non trouvé
    .</p>";
89          // Ici, on ne démarre pas de session et on ne crée
    pas de cookie lié à l'utilisateur
90      }
91
92  } catch (PDOException $e) {
93      error_log("Erreur Requête Utilisateur: " . $e->getMessage
```

```
    ()); // Log l'erreur SQL
94      echo "<p>Erreur lors de la récupération des informations
    utilisateur.</p>";
95      // Ne pas démarrer de session en cas d'erreur BDD
    critique
96  }
97
98  // Fermeture de la connexion PDO
99  $pdo = null;
100
101 ?>
102 <!DOCTYPE html>
103 <html lang="fr">
104 <head>
105     <meta charset="UTF-8">
106     <title>TP8 - Sessions et Cookies avec PDO</title>
107     <style>
108         body { font-family: sans-serif; padding: 1em; }
109         h1 { color: navy; }
110         a { color: firebrick; }
111     </style>
112 </head>
113 <body>
114     <hr>
115     <p><em>Fin du script PHP. Le contenu HTML ci-dessus a été
    généré dynamiquement.</em></p>
116 </body>
117 </html>
```

À Compléter par l'Étudiant

Pour consolider vos connaissances, modifiez le code ***tp8.php*** *:*

1. Dans le bloc de déconnexion (`action=logout`), au lieu d'afficher un message, effectuez une **redirection** HTTP vers `tp8.php` (sans le paramètre `action`). Utilisez `header('Location: tp8.php');` suivi impérativement de `exit();`. Assurez-vous que cet appel à `header()` se fait avant toute sortie HTML.
2. Modifiez l'expiration du cookie `couleurPreferee` pour qu'il expire dans 1 heure en utilisant la fonction `mktime()`. Trouvez comment calculer le timestamp pour "maintenant + 1 heure" avec `mktime()`.
3. (Interface utilisateur) Si l'utilisateur est connecté (c'est-à-dire si `$utilisateur` a été trouvé), affichez un petit formulaire HTML (`<form method="post">`

... </form>) avec un champ de texte (<input type="text" name="nouvelleCouleur">) et un bouton de soumission (<button type="submit">Changer couleur</button>).

4. (Traitement du formulaire) Au début du script (après la connexion PDO mais avant la recherche utilisateur), vérifiez si le formulaire a été soumis (if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['nouvelleCo Si oui, récupérez la valeur de $_POST['nouvelleCouleur'], nettoyez-la (par exemple avec htmlspecialchars()), et utilisez setcookie() pour mettre à jour le cookie couleurPreferee avec cette nouvelle valeur (gardez la même expiration et les mêmes options). Affichez un message de confirmation. Adaptez ensuite l'affichage pour montrer la couleur actuelle du cookie.
5. (Avancé - Authentification réelle) Créez une page login.php avec un formulaire demandant email et mot de passe. Faites pointer ce formulaire vers tp8.php (ou une page dédiée process_login.php). Dans le script de traitement, récupérez l'email et le mot de passe POSTés ($_POST). Recherchez l'utilisateur par email dans la BDD (en récupérant aussi son mot_de_passe haché). Utilisez password_verify($motDePasseSoumis, $motDePasseHacheDeLaBDD) pour vérifier si le mot de passe est correct. **Seulement si** l'email existe ET que password_verify() renvoie true, alors démarrez la session et stockez les informations utilisateur. Sinon, affichez un message d'erreur ("Identifiants incorrects").