Manipulation des Cookies et Sessions en PHP

Par Robert DIASSÉ

Introduction

Lorsqu'on développe une application web, il est souvent nécessaire de mémoriser des informations sur les utilisateurs entre différentes pages. PHP propose deux mécanismes principaux :

1. **Les Cookies** : Stockent des informations sur l'ordinateur du client.
2. **Les Sessions** : Stockent des informations côté serveur, associées à un utilisateur via un identifiant unique.

Dans ce cours, nous allons voir comment utiliser **les cookies et les sessions** avec **PDO** pour assurer la connexion et la gestion des utilisateurs de manière sécurisée.

I. Les Cookies en PHP

1. Définition et Utilité des Cookies

Un **cookie** est un petit fichier stocké sur l'ordinateur du client via son navigateur. Il permet de **mémoriser des données entre plusieurs visites**.

Utilisations courantes :

- Garder les préférences d'un utilisateur (exemple : thème sombre ou clair).
- Stocker une session de connexion persistante (exemple : "Se souvenir de moi").
- Suivre l'activité d'un utilisateur (exemple : panier d'achat en e-commerce).

2. Création d'un Cookie en PHP

On utilise la fonction `setcookie()` pour créer un cookie.

```php
<?php
// Créer un cookie qui expire dans 1 heure
setcookie("nom_utilisateur", "Robert", time() + 3600, "/");
?>
```

Explication du Code :

- `setcookie("nom_utilisateur", "Robert", time() + 3600, "/")` crée un cookie nommé `nom_utilisateur` avec la valeur `"Robert"`.
- `time() + 3600` définit l'expiration du cookie dans une heure.
- `"/"` signifie que le cookie est disponible sur l'ensemble du site.

---

3. Récupérer un Cookie

Les cookies sont stockés dans `$_COOKIE`, un tableau associatif.

```php
<?php
if (isset($_COOKIE["nom_utilisateur"])) {
    echo "Bonjour, " . $_COOKIE["nom_utilisateur"];
} else {
    echo "Aucun cookie défini.";
}
?>
```

Explication du Code :

- `isset($_COOKIE["nom_utilisateur"])` vérifie si le cookie est défini.
- `$_COOKIE["nom_utilisateur"]` permet d'accéder à la valeur du cookie.

---

4. Supprimer un Cookie

On supprime un cookie en définissant une **date d'expiration passée**.

```php
<?php
setcookie("nom_utilisateur", "", time() - 3600, "/"); // Expire immédiatement
?>
```

Explication du Code :

- `time() - 3600` met une expiration passée, ce qui force la suppression du cookie.
- `""` remplace la valeur du cookie pour éviter toute récupération ultérieure.

II. Les Sessions en PHP

1. Définition et Utilité des Sessions

Contrairement aux cookies, les **sessions stockent les informations côté serveur**. Elles permettent :

- Stockage sécurisé des données sensibles (exemple : identifiants).
- Suivi d'un utilisateur sans exposer ses informations au navigateur.

---

2. Démarrer une Session

Avant d'utiliser une session, on doit **toujours** l'initialiser avec `session_start()`.

```php
<?php
session_start(); // Toujours en début de script
$_SESSION["nom"] = "Robert";
echo "Nom stocké en session : " . $_SESSION["nom"];
?>
```

Explication du Code :

- `session_start()` démarre une session PHP.
- `$_SESSION["nom"] = "Robert"` stocke `"Robert"` dans la session.
- `$_SESSION["nom"]` permet d'afficher la valeur stockée.

---

3. Récupérer une Variable de Session

```php
<?php
session_start();
if (isset($_SESSION["nom"])) {
    echo "Bonjour, " . $_SESSION["nom"];
} else {
    echo "Session non définie.";
}
?>
```

Explication du Code :

- `isset($_SESSION["nom"])` vérifie si la session existe.
- `$_SESSION["nom"]` permet d'afficher la valeur stockée.

---

4. Détruire une Session

On supprime une session en appelant `session_destroy()`.

```php
<?php
session_start();
session_destroy(); // Supprime toutes les variables de session
header("Location: index.php"); // Redirige l'utilisateur
?>
```

Explication du Code :

- `session_start()` doit être appelé avant de manipuler la session.
- `session_destroy()` supprime toutes les variables de session.
- `header("Location: index.php")` redirige l'utilisateur après suppression de la session.

III. Connexion Sécurisée avec PDO et Sessions

1. Connexion à la Base de Données avec PDO (`db.php`)

```php
<?php
function getConnection() {
    try {
        $pdo = new
PDO("mysql:host=localhost;dbname=gestion_utilisateurs;charset=utf8", "root", "", [
            PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
            PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
        ]);
        return $pdo;
    } catch (PDOException $e) {
        die("Erreur de connexion : " . $e->getMessage());
    }
}
?>
```

Explication du Code :

- `new PDO("mysql:host=localhost;dbname=gestion_utilisateurs;charset=utf8", "root", "")` établit une connexion avec MySQL.
- `PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION` active les exceptions en cas d'erreur.
- `PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC` récupère les données sous forme de tableau associatif.

---

2. Formulaire de Connexion avec PDO (`login.php`)

```php
<?php
session_start();
```

```php
require 'db.php';

if (isset($_POST['login'])) {
    $pdo = getConnection();
    $username = trim($_POST['username']);
    $password = trim($_POST['password']);

    $stmt = $pdo->prepare("SELECT * FROM utilisateurs WHERE username = ?");
    $stmt->execute([$username]);
    $user = $stmt->fetch();

    if ($user && password_verify($password, $user['password'])) {
        $_SESSION['username'] = $user['username'];
        setcookie("utilisateur", $user['username'], time() + 3600, "/");
        header("Location: dashboard.php");
        exit;
    } else {
        echo "Identifiants incorrects.";
    }
}
?>
```

Explication du Code :

- Vérifie si le formulaire est soumis (`isset($_POST['login'])`).
- Récupère les informations de l'utilisateur via PDO.
- `password_verify($password, $user['password'])` vérifie le mot de passe.

---

3. Page Protégée (`dashboard.php`)

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

echo "Bienvenue, " . $_SESSION['username'];
?>

<a href="logout.php">Déconnexion</a>
```

---

4. Déconnexion (`logout.php`)

```php
<?php
session_start();
```

```php
session_destroy();
setcookie("utilisateur", "", time() - 3600, "/");
header("Location: login.php");
exit;
?>
```

Résumé du Projet

Connexion sécurisée avec sessions et cookies
Gestion des utilisateurs en base de données
Mémorisation de l'utilisateur avec les cookies
Sécurisation des mots de passe avec `password_hash()`
