Sécurisation des routes

Nous voilà sur la dernière partie de ce cours. Nous allons, dans ce chapitre, mettre en place le nécessaire pour permettre à notre boutique en ligne de protéger les pages d'administration en fonction du rôle de chaque utilisateur.

Sur ce principe, un utilisateur ne doit pas pouvoir accéder à la gestion des produits. Seul un administrateur doit être capable de manipuler les produits.

Pour commencer, nous allons nous rendre dans notre fichier de configuration des routes. Au travers de celui-ci, nous avions défini une nomenclature permettant de matcher le nom d'une page entrée dans l'URL avec le nom d'un contrôleur ainsi que de sa fonction.

Pour rappel :

```
...
'home' => 'homeController',
...
```

Explications :

- « home » étant le nom de la route à entrer dans l'URL, par exemple index.php?page=home.
- « homeController » correspond au nom de la fonction et du contrôleur que nous souhaitons renvoyer à l'utilisateur.

Pour pouvoir sécuriser les routes, nous allons continuer sur cette nomenclature, mais en ajoutant un élément supplémentaire : le niveau d'autorisation.

De cette manière, nous allons pouvoir compléter les différentes routes avec l'id autorisé à accéder à la page. Ce qui nous donne l'exemple suivant :

```
...
'home' => 'homeController:0',
...
```

Pour mieux comprendre, nous allons utiliser le caractère « : » pour découper notre chaine de caractères en deux. Le premier morceau sera le nom du contrôleur à afficher et le deuxième morceau l'id permettant d'autoriser l'accès à la page.

À savoir que nous devons fixer des règles concernant le niveau d'accès. Voici un tableau indiquant ces règles :

Nom du rôle	Id	Détails
Visiteur	0	Accès aux pages ne nécessitant pas d'authentification.
Utilisateur	1	Accès aux pages des visiteurs et celles qui nécessitent une authentification, mais qui ne sont pas destinées à l'administration du site.

Administrateur	2	Accès aux pages de rôle d'utilisateurs et aux pages d'administration.

Pour la mise en place de la fonctionnalité, nous avons simplement besoin de modifier notre fichier « router.php » : *(attention, vous avez certainement apporté des modifications lors des exercices de début de cours, adaptez donc le code présenté à la structure du vôtre)*

```
...
$routeParameters = explode(':', $route);

$controller = ucfirst($routeParameters[0]);
$access = $routeParameters[1] ?? 0;

if ($access != 0 ) {
    if (!isset($_SESSION['auth']) || $_SESSION['auth']['role'] < $access) {
        $controller = "HomeController";
    }
}
require_once 'controller/' . $controller . '.php';

return $controller;
...
```

Explications :

- $routeParameters = explode(':', $route); → Explode est une fonction PHP permettant de scinder une chaine de caractères en fonction d'un caractère définis. Dans le cas présent, nous demandons à PHP de découper la valeur contenue dans notre tableau de routes à chaque fois qu'il rencontre le caractère « : ». À savoir que la fonction « explode() » prend en premier paramètre le caractère séparateur et en deuxième la chaine de caractères à découper. Cette fonction ressort la chaine de caractère sous la forme d'un tableau associatif.
- $controller → nous stockons la première valeur du tableau qui correspond bien au nom du contrôleur à renvoyer. Nous en profitons également de rajouter la majuscule nécessaire pour qu'il puisse trouver le fichier du contrôleur.
- $access = $routeParameters[1] ?? 0; → Nous avons ici une condition ternaire. C'est-à-dire que s'il n'existe pas de deuxième valeur dans le tableau, la permission par défaut pour la page sera de « 0 ». Dans le cas où elle existe, « $access » prendra donc la valeur.
- if ($access != 0) { → si le niveau de rôle n'équivaut pas à zéro, cela implique qu'il est nécessaire d'effectuer une vérification d'accès à la page. Dans le cas où la valeur est égale à 0, la page est accessible sans vérification.
- if (!isset($_SESSION['auth']) || $_SESSION['auth']['role'] < $access) { → la deuxième condition se charge de vérifier que l'utilisateur doit être authentifié et qu'il possède le rôle nécessaire pour accéder à la page. Dans le cas où l'utilisateur n'est pas

authentifié et qu'il ne possède pas le rôle, nous écrasons la variable « $controller ». Cette nouvelle est alors celle menant vers la page d'accueil.

Travail à faire :

- Protéger l'ensemble des pages du site e-commerce, tout limitant au bon rôle
- Tester l'accès à vos pages

Informations :

Pour tester vos pages avec un administrateur, vous devez vous rendre dans PhpMyAdmin et modifier le rôle de votre utilisateur. Cliquez sur le bouton « Éditer » de la ligne de l'utilisateur à modifier.

Puis dans la ligne « idRole » sélectionnez l'id numéro 2 qui correspond au rôle d'administrateur.

Pour que la modification soit effective, vous devez vous déconnecter et vous reconnecter.

Il nous reste plus qu'un souci de protection à régler. En effet, lorsque nous sommes authentifiés sur le site, nous pouvons accéder à la page de connexion et d'inscription. Or un utilisateur authentifié ne doit pas pouvoir enregistrer un nouveau compte ni se connecter à nouveau. Puisque c'est une particularité, nous allons ajouter une redirection vers la page d'accueil lorsque l'utilisateur est authentifié et qu'il tente d'accéder à l'une de ces deux pages.

Pour cela, modifions le contrôleur des deux pages et ajoutons le code suivant :

```
if (isset($_SESSION['auth'])) {
    header('Location: index.php');
}
```

Ce code est à placer en première ligne de la fonction du contrôleur. Elle permet simplement de vérifier si l'utilisateur est authentifié. Si c'est le cas, alors il est redirigé vers la page d'accueil.

Maintenant, vous êtes capable de créer un système d'authentification complet, manipuler des enregistrements en base de données et sécuriser votre application. Pour clôturer ce dernier chapitre, vous allez mettre en pratique tout ce que nous avons vu.

Travail à faire :

- Réaliser une page d'administration des utilisateurs. Cette page doit permettre à un administrateur de modifier, supprimer, ajouter et consulter un utilisateur. Attention, lors de la consultation d'un utilisateur, personne (même un administrateur) ne doit pouvoir consulter le mot de passe. Pour valider la modification d'un utilisateur, l'administrateur devra valider deux fois le mot de passe.
- Réaliser une page profil permettant à un utilisateur authentifié de modifier les informations de son profil. L'utilisateur ne doit pas pouvoir modifier son adresse mail. Lorsqu'il modifie son profil, il est important de lui demander à nouveau son mot de passe actuel. De même, quand il modifie son mot de passe, il sera nécessaire qu'il le confirme deux fois.
- Permettre à l'utilisateur de désactiver son compte

Une fois que vous avez terminé ce chapitre, envoyez le code du site sur le répertoire Github.