**CORLER Damien – WEBO - Rapport 2 - 23/09/2017**

## COURS

- Utilisation des sessions (maitrisé)
- Utilisation de PhPmyadmin (maitrisé)
- Mise en garde contre les vols de sessions et les injections SQL (semi-découverte)

Découverte de @ devant les variables qui permet de rendre les messages d'avertissement muets.

## EXERCICE

L'un des exercices consistait à manipuler une des composantes de PHP : les sessions.

Les sessions constituent un moyen alternatif aux BDD, GET, POST, ... de conserver des variables et de les transmettre sur toutes les pages d'un site pendant toute la durée de la présence d'un visiteur.

En effet, sans l'un de ses moyens, dès que le visiteur charge une autre page, ces variables sont « oubliées ».

Dans cet exercice très simple, l'objectif était de présenter à l'utilisateur un formulaire qui comprend ces fonctionnalités : **se connecter**, **se déconnecter** et **rafraîchir la page**.

La première consiste à conserver un identifiant de l'utilisateur tout le long de sa visite, la seconde de le détruire et d'effacer toutes traces de son existence, et la dernière de rafraîchir les pages afin de vérifier le bon accomplissement des deux précédentes fonctions.

Pour débuter, il est nécessaire de faire un appel à la fonction "session_start()" de PHP afin de lui demander de mettre en place une session. Il est primordial d'effectuer cet appel le plus haut possible dans le code, et après avoir vérifié si une session n'est pas déjà en cours. Ceci étant fait sur chacune des pages du site, voici le descriptif des fonctions énoncés.

**Se connecter** : le bouton dédié sur le formulaire envoi une requête en GET avec un paramètre booléen "init". La fonction ainsi activée va déclarer une variable dîtes "de session" : celle si sera stockée, restera en mémoire et permettra par la suite aux autres pages de déterminer si l'utilisateur est connecté. Un ID de session est également généré et est lui aussi stocké en mémoire.

**Se déconnecter** : afin de déconnecter l'utilisateur, il est nécessaire d'effacer toutes traces d'une session. Activée par un paramètre booléen GET "destroy", la fonction va effectuer ces actions : tout d'abord, nous détruisons toutes les variables de session, dont celle qui stipule que l'utilisateur est connecté. Ensuite, l'objectif est de retrouver et de détruire le "cookie" que la session PHP génère afin de stocker des données : ce cookie est une composante à ne surtout pas oublier, car est un possible vecteur d'usurpation d'identité (vol de session). Afin de le rendre irrécupérable, nous altérons ses données et nous le définissons périmé en le paramétrant avec une large marge horaire afin de contrer les attaques temporelles. La session est désormais détruite, son ID est également régénéré.

**Rafraîchir** : le bouton rafraîchir consiste tout simplement à appeler une page sans aucun paramètres GET afin d'effectuer une vérification du bon accomplissement des deux autres fonctions. Si l'utilisateur est connecté, l'ID doit rester le même : à l'inverse, à charge rechargement, l'ID doit être unique.

## CONCLUSION

Peu de découvertes pour ma part, j'ai déjà manipulé à de nombreuses fois les sessions au travers de divers projets en entreprise. J'ai apprécié le rappel par rapport aux menaces et aux potentiels vecteurs d'attaques.