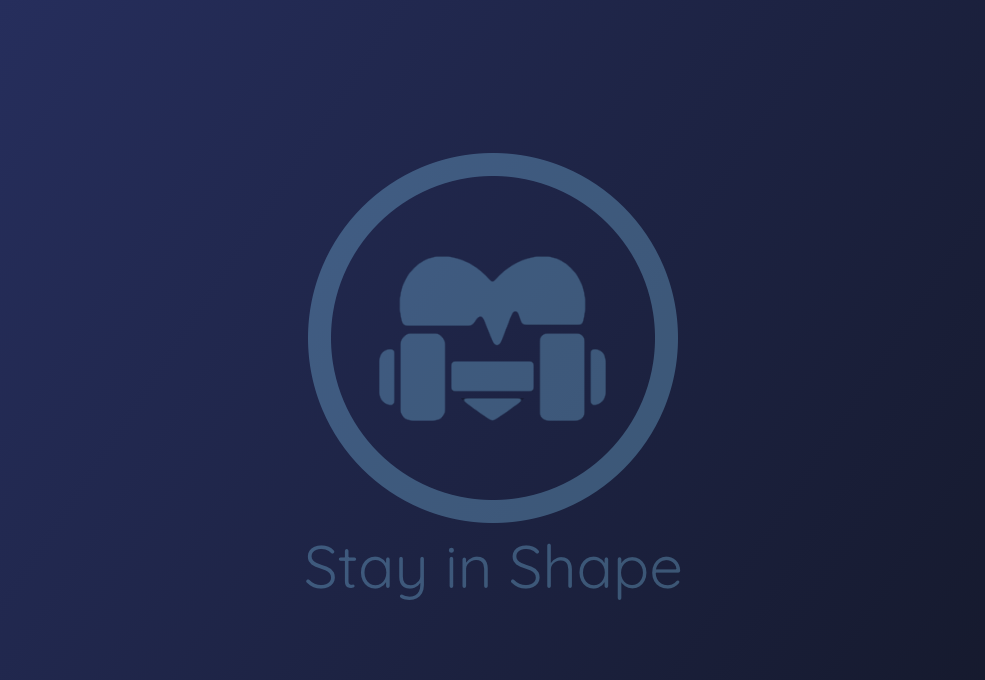
Mesures et bonnes pratiques de sécurité

Laignel Camille

Développeur Web et Web mobile

Application Stay in Shape



Sécurisation des sessions :

Afin de me prémunir d'une faille de type session hijacking qui consiste pour un attaquant à deviner (ou bruteforce) les identifiants de session, j'ai adopté plusieurs stratégies permettant d'empêcher de nombreuses attaques potentielles.

a. Génération de l’id de session :

J'ai commencé, pour empêcher le bruteforce de session en générant des identifiants de session de 256 bits, de manière aléatoire et cryptographiquement sécurisé.

Pour cela, j'ai utilisé la fonction random\_bytes(32) qui me permet de générer 32 octets de manière aléatoire et sécurisée.

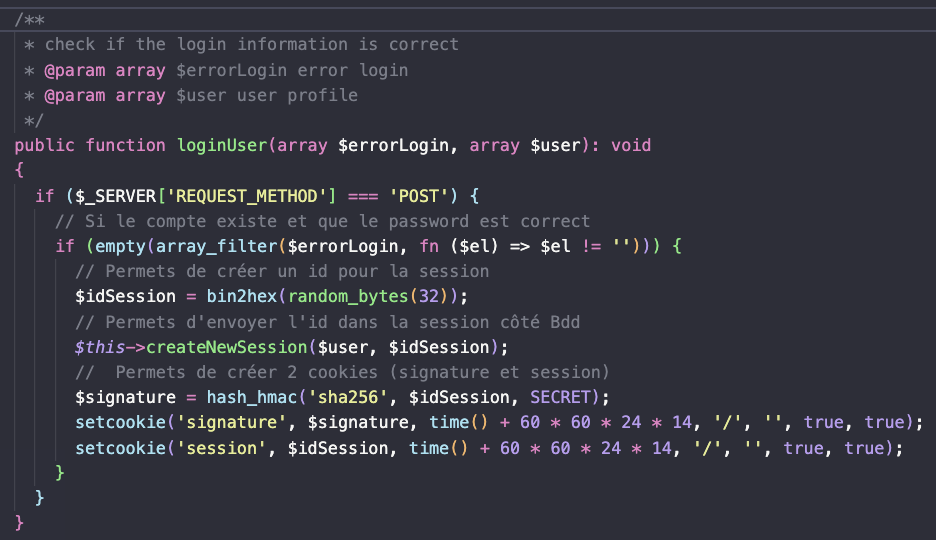
Ensuite afin de pouvoir le stocker dans la base de données j'ai effectué une conversion en hexadécimal à l'aide de la fonction bin2hex(random\_bytes(32)).

b. Signature des cookies :

Ici, j'ai décidé de signer les cookies en utilisant l'algorithme HS256 à l'aide de la fonction hash\_hmac() qui prend en paramètres l'algorithme utilisé pour le hachage, l'identifiant de la session que je souhaite protéger, ainsi qu'une clé secrète personnellement définie. Cette clé secrète ne devra pas être divulguée, et devra être d'une longueur conséquente ( minimum 256 bits ), afin de protéger au maximum celle-ci d'une attaque de types bruteforce.

À cet effet, une fois la signature générée, il suffit de créer 2 cookies, un contenant l'identifiant de session et l'autre la signature.

Lors de la connexion d’un utilisateur, si l’identifiant et mot de passe s'avère correct alors les 2 cookies précédemment vus seront créés, puis vérifiés à l'aide de la fonction hash\_equals() qui permet simplement de vérifier la signature du cookie. Si l'identifiant de session dans le cookie a été modifié ou qu'il n'a pas été créé par le serveur, il ne sera pas validé et nous rejetterons la connexion.



c. Réutilisation des sessions

Lorsqu'un utilisateur quitte l'application sans se déconnecter, il pourra accéder à celle-ci durant 14 jours.

Pour mettre en place cette fonctionnalité, j'ai utilisé les cookies précédemment créés.

En effet, lorsque l'utilisateur entrera l'adresse du site, une première vérification sera effectuée, si le cookie signature et le cookie identifiant de session sont bien présents, alors la fonction hash\_equals() se lance en prenant en paramètre le hash ainsi que la signature. Cette fonction vérifie la signature du cookie.

Si l'identifiant de session dans le cookie a été modifié ou qu'il n'a pas été créé par le serveur, il ne sera pas valide et nous rejetterons la connexion.

Une image contenant texte

Description générée automatiquement

Nettoyages des inputs :

a. Faille XSS

Afin de me prévenir des attaques des cross-site-scripting qui permettent d’injecter du contenu dans une page, j'ai effectué un nettoyage de l'ensemble des entrées utilisateur au sein du site. À ce titre, j'ai principalement utilisé les fonctions filter\_input\_array() et filter\_var() avec des filtres tels que FILTER\_SANATIZE\_EMAIL, FILTER\_SANATIZE\_SPECIAL\_CHARS ainsi que des filtres de validation tels que FILTER\_VALIDATE\_EMAIL.

Une image contenant texte

Description générée automatiquement

Modification d’URL :

Afin de protéger le site, des actions menées directement sur l’URL, les méthodes appartenant aux controller qui utilisent les données stockées dans la barre de recherche du navigateur, se verront dotées de plusieurs vérifications supplémentaires qui, dans le cas d’un échec, lèveront une erreur à l’aide de "throw new exception", ce qui génèrera une page d’erreur invitant l’utilisateur à retourner sur la page principale du site.

J’ai effectué ce type de contrôle afin d’éviter toute modification intempestive lors de l’utilisation de cette application.



Sécurisation de dossier secure

Le dossier secure contient le fichier secureData.php qui regroupe l’ensemble des données sensibles du site (connexion à la base de données, identifiant et mot de passe de l’adresse e-mail utiliser par phpMailer).

Afin de sécuriser au maximum les données contenues dans ce fichier, j’ai décidé d’interdire l’accès de ce fichier à l’aide d’un identifiant, et d’un mot de passe.

En effet pour accéder à ce fichier il faudra détenir le mot de passe nécessaire à son ouverture.

Cette option est proposée par hostinger.

Lorsque l’on configure cette option, hostinger ajoute deux fichiers dans le dossier à protéger.

* .htaccess  : ce fichier permet de sécuriser un dossier et d’indiquer au serveur où se situe le pseudo est le mot de passe nécessaire à l’ouverture de celui-ci. Ainsi, le serveur saura quels sont les utilisateurs autorisés à accéder à la zone sécurisée.
* .htpasswd : ce fichier comporte les informations avec les noms des utilisateurs qui peuvent accéder au dossier. Pour chaque utilisateur, il y a un mot de passe associé. Le mot de passe est alors crypté.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Configuration du serveur :

Avant le déploiement du site, j'ai dû accéder au fichier php.ini afin d'effectuer quelques modifications.

Concernant les sessions, j’ai modifié :

* session.gc\_maxlifetime = 1 209 600 (afin de spécifier la durée de vie des données sur le serveur).
* session.use\_strict\_mode =1 (afin d’utiliser le mode d’identifiant de session strict).
* session.use\_only\_cookies=1 (permets de spécifier si le module doit utiliser seulement les cookies pour stocker les id des sessions afin de protéger l’URL).
* session.cookie\_lifetime = 1 209 600 (spécifie la durée de vie d’un cookie).
* session.cookie\_httponly=1 (protection contre les attaques xss, cookie non accessible par Js).
* session.cookie\_secure=1 (permet d’émettre un cookie que sur des connexions sécurisées).
* display\_errors=OFF (ne pas afficher les erreurs à l’utilisateur).
* log\_errors=ON (permets l’enregistrement des erreurs rencontré lors de la navigation sur le site).
* allow\_url\_fopen=OFF (évite de récupérer des données à partir d’emplacements distants via FTP ou HTTP).
* file\_uploads=ON (permets de télécharger des fichiers via HTTP).
* upload\_max\_filesize=2M (permets de définir une taille maximum de fichier à télécharger).