

Documentation technique

authentification, session et gestion des acces administrateurs

BINVIGNAT Louise | E4 | Session 2019

Sommaire

[Cahier des charges 3](#_Toc10352064)

[Connexion au serveur 3](#_Toc10352065)

[Architecture MVC 3](#_Toc10352066)

[Routeur 3](#_Toc10352067)

[Navigation 5](#_Toc10352068)

[Cookies 6](#_Toc10352069)

[Model 7](#_Toc10352070)

[View 7](#_Toc10352071)

[Controller 7](#_Toc10352072)

Important : Ce document n’est pas finalisé et peut être modifié à tout moment

# Cahier des charges

Le cahier des charges exige un portail d’authentification pour le back-office, où l’utilisateur sera obligé de se connecter pour aller plus loin. La connexion (si réussie) va générer des cookies pour maintenir l’utilisateur connecté.

Il est aussi imposé que seuls les admins aient accès à l’écriture des données. Les autres utilisateurs du back-office ne pourront qu’accéder aux données en lecture, et les pages de modifications de données leurs seront donc interdites.

# Connexion au serveur

L’adresse IP du serveur de production est **172.17.21.14**.

# Architecture MVC

L’architecture du site est basée sur le modèle MVC (*Model View Controller*).

La récupération des actions se fera via la requête $\_GET. Les composants seront détaillés dans leurs parties respectives de la documentation.

## Routeur

Ici, le routeur de l’application est **index.php**. Ce dernier reçoit les requêtes $\_GET[‘action’] et exécute une série de tests.

L’existence de la requête dans l’URL sera d’abord vérifiée comme ceci :

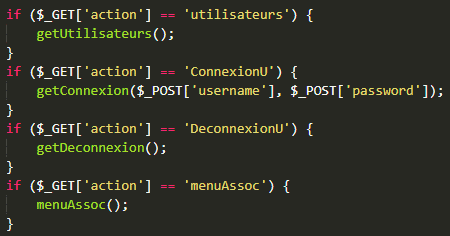
*index.php l.5*



**Si la condition est validée** :

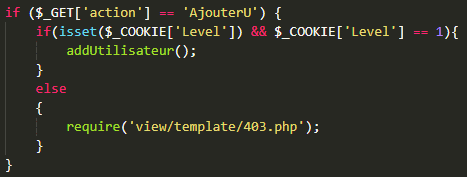
L’application sera confrontée à une série de if et de if/else pour chaque action existante. Ici, on vérifie le contenu de l’URL. L’application exécutera alors l’action correspondante au contenu de l’URL.

*index.php l.7*



Pour les actions qui tentent de rediriger vers une page dédiée à l’écriture de données, une vérification de cookies va se faire. Les cookies seront détaillés dans la section « Cookies » de la documentation. On utilisera ici le cookie stockant le rang de l’utilisateur, pour vérifier si ce dernier est administrateur ou non. Si oui, l’action s’effectuera normalement avec l’appel du contrôleur. Si non, l’utilisateur sera renvoyé sur **403.php**.

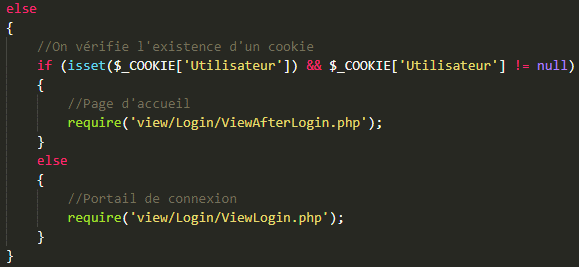
*index.php l.20*



**Si la condition n’est pas validée** :

Le routeur vérifiera l’existence d’un cookie utilisateur. Si ce dernier existe et si son contenu n’est pas vide, alors il sera renvoyé à la page d’accueil (**ViewAfterLogin.php**). Sinon, il sera renvoyé au portail (**ViewLogin.php**).

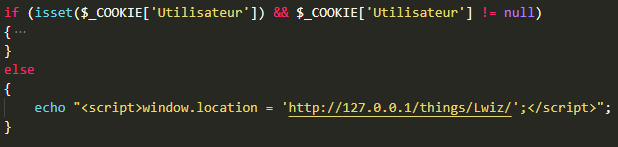
*index.php l.58*



## Navigation

Après authentification, l’utilisateur sera renvoyé dans la page d’accueil (**ViewAfterLogin.php**). Un **header.php** sera chargé sur toutes les pages du dossier View. Cet header sert de navigateur pour l’utilisateur, mais aussi de contrôle de cookie. Etant donné que le header sera présent sur toutes les pages (excepté le portail), ce sera donc lui qui contrôlera l’existence du cookie Utilisateur. Si ce dernier n’est pas présent (cookie expiré par exemple), l’utilisateur sera renvoyé au portail de connexion.

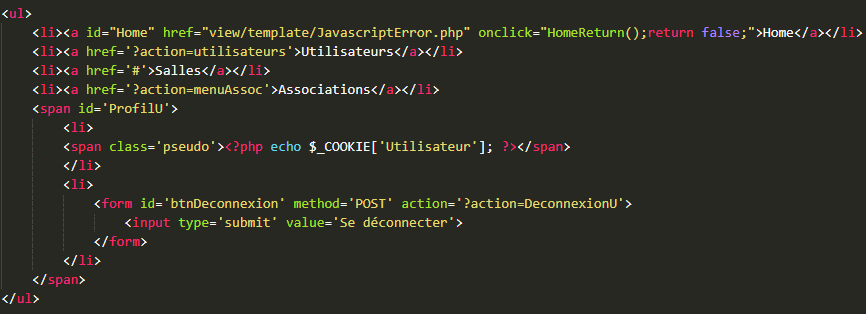
*index.php l.4 puis l.29*



Ce navigateur possède 4 boutons sur la gauche : Home, Utilisateurs, Salles et Associations. Sur la droite, on retrouve le nom de l’utilisateur (via $\_COOKIE[‘Utilisateur’]) et un bouton « Se déconnecter ».

La navigation se fait via les actions. Etant donné que le bouton Home n’est pas supposé posséder d’action, un script est nécessaire pour pouvoir atteindre cette page.

*Header.php l.9*



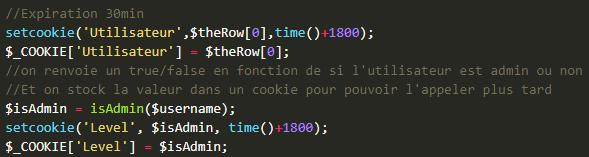
Ici « HomeReturn() ; » renvoie à un script stocké dans **Redirection.js**, qui renvoie tout simplement l’utilisateur à la page d’accueil pour permettre les cas où l’URL possède une action, d’effacer cette dernière et charger la page. Le return false ; qui le suit empêche la page de prendre en compte le lien du href si HomeReturn() fonctionne. Si ce n’est pas le cas, on redirige l’utilisateur vers **JavascriptError.php** pour lui indiquer que son Javascript est désactivé.

# Cookies

Les cookies servent à stocker les données d session de l’utilisateur. Ici, 2 cookies sont utilisés : Un pour stocker le nom du compte de l’utilisateur, un autre pour stocker le rang de l’utilisateur. Ce sont ces cookies qui permettront à l’utilisateur de rester connecté. Les cookies sont réglés sur 30min, après ça, ils expireront automatiquement et l’utilisateur devra se reconnecter à nouveau.

Les cookies seront générés dans **fonctions.php**, une fois que le contenu des requêtes contrôlé.

controller.php l.78



$\_COOKIE[‘Utilisateur’] contiendra donc le nom de l’utilisateur (string) et $\_COOKIE[‘Level’] un booléen dont 1 signifie que l’utilisateur est administrateur. On pourra les appeler partout dans n’importe quelle page de l’application tant qu’ils existent.

# Model

## login()

TODO

## logout()

TODO

## isAdmin()

TODO

# View

## Login (dossier)

TODO

## Template (dossier)

TODO

## HomeReturn()

TODO

# Controller

## getConnexion()

TODO

## getDeconnexion()

TODO

## get403()

TODO