



644 L'OCCITANE - BÂTIMENT ARIZONA A,  
31670 LABÈGE  
[HTTPS://WWW.FITTINGBOX.COM](https://www.fittingbox.com)

ADRAR Digit@l Academy  
Pôle Numérique  
11 avenue de l'Europe  
31520 Ramonville



**Meyelook®**  
formes & couleurs

**Mémoire professionnel**

**DJEDOUI Jamila | DEVELOPPEUSE LOGICIEL | Carbonne 2018/2019**

**SKY IS THE LIMIT...**

# **TABLE DES MATIERES**

Remerciements .....	4
Introduction .....	5
Abstract .....	7
I – Présentation et gestion de projet.....	8
1 : Présentation de Fitting Box : L'entreprise, ses talents .....	8
2 : Présentation de l'application .....	9
3 : Analyse de l'existant .....	12
A] Page d'accueil de l'application MeyeLook.....	12
B] Qualification d'une monture, partie gauche de l'écran .....	13
C] Qualification d'une monture, partie droite de l'écran .....	15
4 : Cahier des charges : .....	16
A] MeyeLook, analyse et INTRODUCTION : .....	16
B] Description des missions : .....	16
C] Planification des priorités des MISSIONS :.....	17
5 : Planification des livrables et tests.....	18
II – Spécifications fonctionnelles.....	19
1 : Use cases .....	19
A] Connexion de l'utilisateur.....	19
B] Les actions de l'utilisateur .....	20
2 : Diagramme de séquence .....	20
A] Ajout ou mise à jour des montures .....	21
B] Supprimer une monture .....	22
3 : Maquettage.....	23
A] Connexion de l'utilisateur.....	23
B] Modal pour suppression de monture .....	24
4 : Fonctionnalités diverses.....	25
A] Suppression du menu de gestion de profil de l'utilisateur.....	25
B] Centrer le message d'accueil et le login de l'utilisateur .....	26

<b>III – Conception .....</b>	<b>27</b>
1 : Connexion de l'utilisateur .....	27
2 : Attribution du style à une monture .....	28
3 et 4 : MCD et MLD .....	29
<b>IV – Arborescence et Architecture .....</b>	<b>31</b>
1 : Arborescence .....	31
2 : Architecture .....	31
A] Architecture initiale .....	32
B] Nouvelle architecture .....	33
<b>V – Spécifications Techniques .....</b>	<b>34</b>
1 : Gestion des fichiers : les étapes de réalisations .....	34
A] Ajout ou mise à jour des montures .....	34
B] Supprimer une monture .....	37
C] Centrer le message d'accueil et le login de l'utilisateur .....	39
2 : Environnement technique et équipements .....	40
3 : La connexion à la base de données.....	44
<b>VI– Spécifications diverses .....</b>	<b>46</b>
1 : Environnement de tests.....	46
2 : Difficultés rencontrées.....	47
<b>Conclusion .....</b>	<b>48</b>
<b>Annexes .....</b>	<b>50</b>
<b>Glossaire .....</b>	<b>64</b>

# REMERCIEMENTS

- Je tiens à remercier chaleureusement l'entreprise **FittingBox** au complet !

L'équipe produit qui m'a accueillie : **Laurent FEDOU** le chef de l'équipe, disponible et à l'écoute durant mes missions. Bien sûr toute l'équipe : **Guillaume, Diane, Nizar, Adrien, Bastien, Yannick, Olivier, Florian, Marwen, Emma** toujours disponibles et compétents dans leur travail, ils ont souvent été d'une aide inestimable et m'ont prêté une oreille des plus attentives !

Des remerciements sincères également à **Aurélie CROQUIN** de **FittingBox** pour sa confiance et son appui.

- Je cite l'école du numérique l'**ADRAR**.

Pour son accompagnement et sa confiance quant à ma candidature dans son intégralité, **M. CHRETIENNE**, les formateurs, les coordinateurs.

Plus particulièrement **Mme Florence CALMETTES** pour ses conseils, sa patience et ses encouragements tout au long de la formation.

- La région Occitanie également,

Elle m'a permis d'accéder au cursus de formation sur le nouveau site de Carbone et m'a donné accès à la formation professionnelle que je recherchais dans mon secteur géographique.

## INTRODUCTION

Le projet est mené dans une entreprise appelée '**FittingBox**'.

Ils se spécialisent dans le développement d'applications pour des clients professionnels exclusivement opticiens.

Personnellement, j'ai travaillé sur un outil développé précédemment appelé '**Meyelook**' qui devait être amélioré et sécurisé. Il s'agit d'une application de qualification de lunettes par catalogue.

Via un Web service, les opticiens définissent une sélection de lunettes appropriées en fonction de certains paramètres. Celles-ci tiennent compte d'éléments tels que le visage, la couleur des yeux, la couleur des cheveux, le type de peau, etc. Cet outil est ce que l'on peut appeler : le petit plus 'Stylisme' côté opticiens.

J'étais principalement en charge de maintenir les fonctionnalités, d'ajouter une sécurisation et de mettre à jour l'outil pour qu'il soit efficace et utile. Le client était satisfait du résultat. Même si j'ai travaillé seule, j'ai réussi à gérer des tâches particulières telles que le chargement des fichiers, la mise à jour des données des montures, l'affichage des lunettes et du catalogue, mener certaines réflexions sur la base de données et ainsi atteint les objectifs fixés. J'ai aussi rédigé la documentation quant à mes actions.

Personnellement, j'ai beaucoup aimé travailler sur ce projet. En tant que stagiaire, j'ai eu l'opportunité de travailler avec des développeurs, des designers, des référents qualité... et les clients de l'entreprise.



## **ABSTRACT**

This project consists in presenting the main mission during my internship in a company called 'FittingBox'.

They specialize in developing applications for professional clients who are opticians exclusively.

Personally, I worked on a tool developed earlier called 'Meyelook' which required to be improved and secured among other things. It consists on an application of eyeglass qualification categorized by catalogue.

It has a Web service that allows the opticians to define a selection of appropriate glasses according to some parameters. These take into account elements such as their face, eye and hair colour, type of skin, and so on. This tool is what we can call: the little 'Stylish' extra provided by Opticians to users. I was mainly in charge to maintain functionalities, add security and upgrade the tool so that it is efficient and useful. The client was satisfied with the result. Even though I worked alone, I managed to handle issues such as file loading, frame data updating, glass and catalogue displaying, some thinking on the database and so reached the goals. I've also documented my tasks.

Personally, I enjoyed very much working on this project. As an intern, I had the opportunity to take part in a project, to work with developers, designers, quality referents... and the customers of the company.

# ***I – PRÉSENTATION ET GESTION DE PROJET***

## ***1 : PRÉSENTATION DE FITTING BOX : L'ENTREPRISE, SES TALENTS***

*L'entreprise voit le jour en 2006 lorsque M. Benjamin HAKOUN et M. Ariel CHOUKROUN relèvent le pari de réinventer le miroir traditionnel de l'opticien pour créer le 1er miroir digital. Couplée à une technologie de pointe et intuitive, cette innovation a pour ambition de transformer la manière dont exercent les professionnels de l'optique en les propulsant dans l'ère digitale.*

*FittingBox est composé de 53 employés répartis entre Miami (Floride) et Labège (Toulouse). 50% sont des femmes et l'âge moyen est de 33 ans. L'entreprise possède la plus grande base de données de montures numériques au monde (plus de 62 000 montures).*

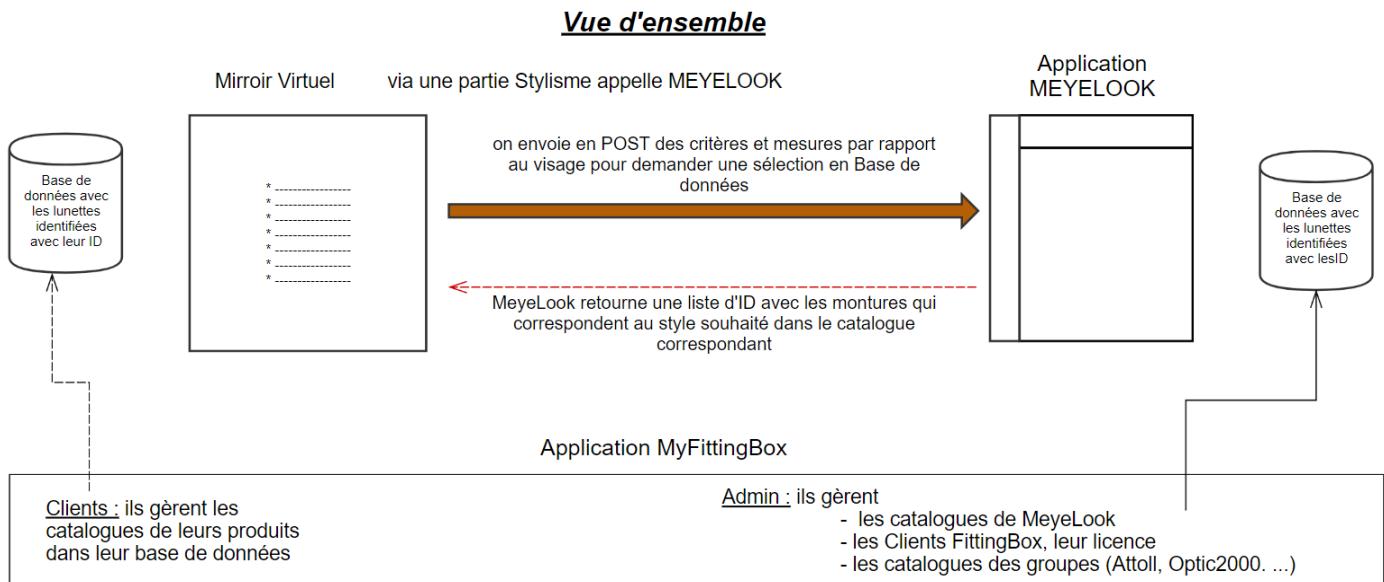
*FittingBox commercialise des outils interactifs d'aide à la vente en ligne et en magasin, pour les professionnels de l'optique. Leurs solutions enrichissent l'expérience d'achat des clients en les aidant à trouver la monture idéale. Ainsi, opticien, e-commerçant, fabricant ou représentant d'une marque, la gamme de produits offre des solutions pour les magasins ou les sites e-commerce.*

<https://www.fittingbox.com/fr/about-us>



## 2 : PRESENTATION DE L'APPLICATION

[DOC] Meyelook Schéma Global Explicatif



Elle a été éditée par un prestataire de service et l'application a gagné les locaux de Fitting Box à la suite de son rachat en août 2018. L'application n'est actuellement pas stable et nécessite un rafraîchissement par le lancement de la première version propre à Fitting Box qui sera proposée dans les mois à venir.

### Les clients, qui sont-ils ?

Ce sont les opticiens qui utilisent le « Miroir Virtuel » dans leur enseigne. Pour eux tout est transparent ils détiennent une liste de montures disponibles dans leur base de données. Leur travail sera de proposer les montures à **leurs** clients en fonction de leur vue et de leurs envies. Lorsque ceux-ci le souhaitent, ils peuvent demander un avis Styliste pour faire leur choix. C'est là que MeyeLook entre en jeu via le Miroir Virtuel.

### Fonctionnement général :

L'application « **Miroir Virtuel** » lancé, plusieurs prises de mesures et d'informations sont enregistrés telle la forme du visage, la couleur de la peau, la couleur des cheveux, des yeux... Mais aussi en fonction de l'activité que le client final fera avec la monture ou encore le style voulu : vintage, sport...

## Accueil Application Miroir Virtuel avec la partie « Stylisme » en bas qui appelle Meyelook

Langue : FRANÇAIS ▾

FEMME



OPTIQUE

HOMME



OPTIQUE

OU



SOLAIRE

OU



SOLAIRE

En **5 minutes** découvrez votre **vrai style** de lunettes



VISAGISME



Répondez aux questions de notre **conseiller visagiste**  
et découvrez le type de monture qui vous va le mieux...

Powered by **FittingBox**



Une fois toutes les informations recueillies, Meyelook propose une liste de montures correspondantes.

L'application fait le tri dans sa base de données avec les paramètres qui lui sont demandés et renvoie une liste de monture qui correspondent aux attentes du client. Les montures sont identifiées par un identifiant (ID) qui est unique et identique dans les catalogues des clients qui font appel à cette application. Evidemment, les retours de Meyelook se font selon un ou plusieurs catalogue(s) défini(s) utilisé(s) par les clients (opticiens).

Comme indiqué dans le schéma, « **MeyeLook** » est une application qui est appelée via un *web service*.

Les utilisateurs de l'application sont les membres de l'entreprise **FittingBox**.

Ils peuvent, via l'*interface* de gestion des montures *qualifier*, ajouter ou supprimer des montures. Les montures sont photographiées ajoutées aux données pour être utilisées dans l'application.

Les photographies de face et de profil des montures sont utilisées dans l'application afin d'en montrer un aperçu.

Meyelook est le projet de gestion de montures.

### **3 : ANALYSE DE L'EXISTANT**

La page d'accueil se décompose en trois parties principales : **le bandeau header**, **le menu avec les onglets** et **le « corps »** de la page.

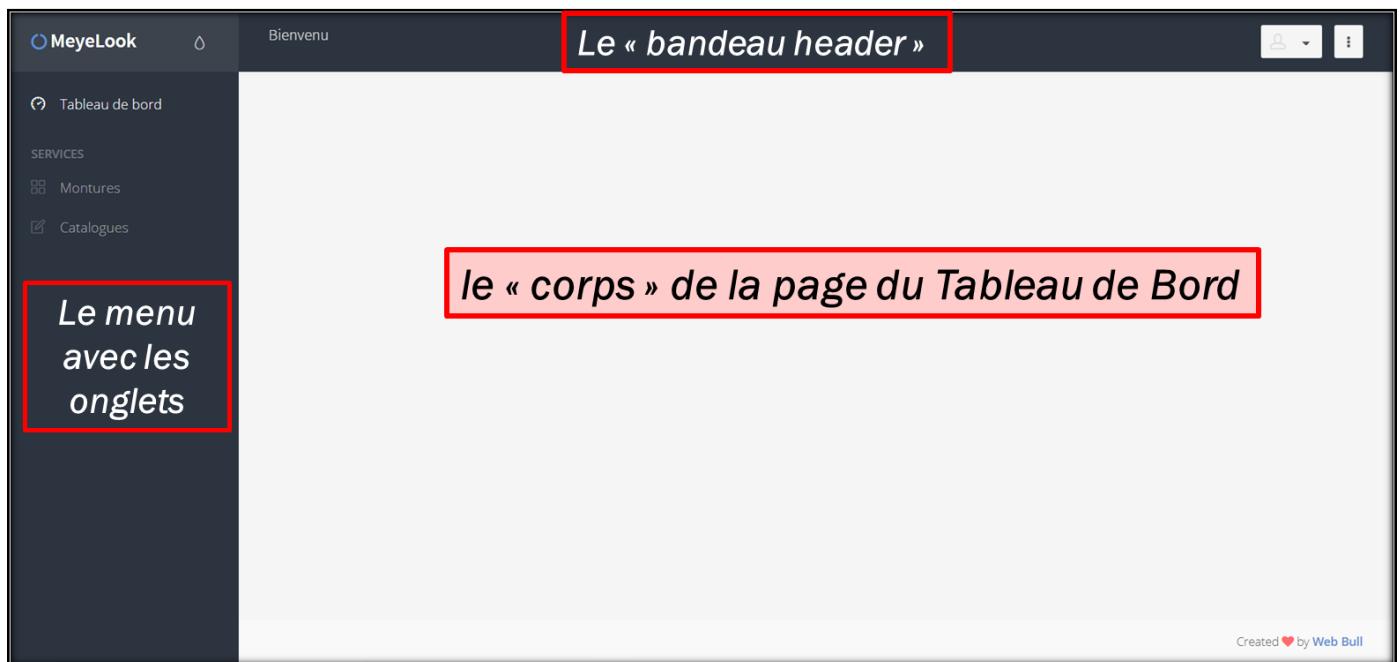
**Le « bandeau header »** est composé de 2 parties : un message de bienvenue à gauche et une liste à droite. Cette liste contient un petit menu déroulant login et profil pour gérer son profil ainsi qu'une icône de masquage de menu.

**Le menu** se situe à gauche sur l'écran et contient plusieurs éléments.

- Tout en haut, au même niveau que le header, on peut cliquer sur le nom de l'application pour retourner à la page d'accueil à tout moment.
- En dessous, le titre « Tableau de bord »
- Vient ensuite les onglets « **SERVICES** » qui sont **Montures** et **Catalogues** et que nous les détailleront plus tard.

Dans le « **corps** » de la page du **Tableau de Bord**, qui est la plus grande partie, on peut afficher les informations que l'on souhaite en cliquant sur l'onglet du menu SERVICES.

### **A) PAGE D'ACCUEIL DE L'APPLICATION MEYELOOK**



Maintenant que nous avons décortiqué la page d'accueil, nous pouvons nous attarder sur les fonctionnalités des éléments.

A noter que les plus importants restent les onglets MONTURES et CATALOGUES.

## **B1 QUALIFICATION D'UNE MONTURE, PARTIE GAUCHE DE L'ECRAN**

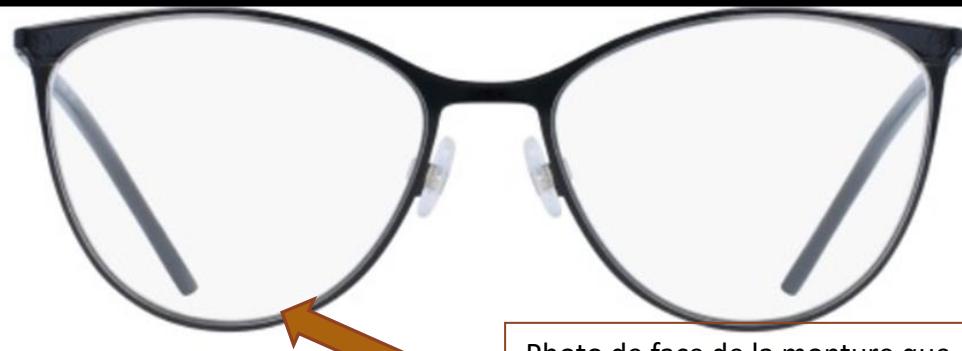


Photo de face de la monture que l'on souhaite qualifier



Photo de trois quarts de la monture que l'on souhaite qualifier

La température et l'intensité de la monture nous donne une information sur la couleur de la monture. Ici : une température froide et une intensité sombre.

Couleur dominante



Température : Froide  
Intensité : Sombre

Couleur secondaire



Les pipettes permettent de retrouver facilement les couleurs dominantes de la monture, en fonction du clic la couleur est affichée dans le rectangle en dessus. La pipette sélectionnée est sur fond vert lorsqu'elle est active. Dans ce cas, la monture est d'une couleur qui varie entre le noir et l'anthracite.

Catalogue

fittingbox



Marque

Marc Jacobs

Modèle

MARC 41

Genre

Mixte



Gamme de prix

Milieu de gamme



Style

Classique



Boxing

54



Hauteur du verre

36



Nez

17



Longueur branches

140



Forme

Rectangle adouci



Matière

Plastique



Montage

Percée



Type de tenons

Déporté



Position des tenons

Bas



Plaquette

Monture "légère"

Monture "solide"

Verres Progressifs

Précédente

Mettre à jour

Suivante

## **CJ QUALIFICATION D'UNE MONTURE, PARTIE DROITE DE L'ECRAN**

### **Légende**



*On peut modifier les informations concernant :*

- La forme,
- Le montage,
- Le type de tenons,
- Leur position,
- La matière de la monture,
- Le genre,
- Le style,
- Les catalogues,
- Et la gamme de prix

Via des menus déroulants, les valeurs sont déjà entrées.



On saisit les informations directement au clavier

Grâce aux inputs de type « checkbox switch » on enclenche ou non les autres caractéristiques



Modèle de la monture. Juste l'information, on ne peut pas modifier.



La marque est reportée ici. Juste l'information on ne peut pas modifier



Lorsque l'on clique sur « Mettre à jour », les informations sont enregistrées dans la base de données via une requête. On peut passer à la monture suivante ou revenir à la précédente.

## **4 : CAHIER DES CHARGES :**

### **A) MEYELOOK, ANALYSE ET INTRODUCTION :**

Les utilisateurs de MeyeLook sont les intervenants de FittingBox qui seront amenés à utiliser l'application à divers degrés. Les clients qui feront appel à Meyelook même indirectement sont les opticiens et professionnels de l'optique tous secteurs confondus, aussi bien dans les cabinets que dans les magasins ou encore via des site e-commerce.

#### MeyeLook Pourquoi ?

L'interface permet une gestion de la base de données en temps réel. La mise à jour, l'ajout ou la suppression des montures dont dispose les clients de FittingBox via une application qui s'appelle « Miroir Virtuel ». La gestion des montures devient plus aisée.

#### Pour qui ?

Tout d'abord pour les intervenants directs de FittingBox mais aussi indirectement pour les clients de FittingBox qui utilisent l'application « MyFittingBox » dont ils disposent d'une licence.

#### L'application :

Meyelook a gagné les locaux de FittingBox en août 2018 où elle sera revue et remise au goût du jour par l'équipe produit. Actuellement une version qui s'appelle V4 est celle utilisée en production. Je suis amenée à proposer la Version Alpha 1.0 pour les changements à mettre en place depuis la reprise de l'application chez FittingBox.

#### L'équipement des clients :

Les clients qui feront appel à Meyelook disposent d'une tablette de type IPad et qui propose l'option « Visagisme ».

### **B) DESCRIPTION DES MISSIONS :**

- 1- Dresser un schéma explicatif global (présent page 10)
- 2- Présenter l'*arborescence* actuelle de l'application afin de cerner l'étendue du projet et des changements à apporter en vue d'une *architecture MVC*
- 3- Permettre l'ajout et la mise à jour des catalogues de montures
- 4- Sécuriser l'accès aux seuls administrateurs authentifiés
- 5- Rendre fonctionnelle la suppression avec confirmation à l'utilisateur
- 6- Optimiser la connexion à la base de données et supprimer les répétitions dans les fichiers
- 7- Suppression du profil sur le tableau de bord de l'interface et remplacement par le bouton « DECONNEXION ».
- 8- Permettre l'exportation de catalogue vers un fichier Excel
- 9- Documentation
- 10- Réflexion autour de la base de données, ajouter des données en maintenant les fonctionnalités du WS.

## **C) PLANIFICATION DES PRIORITES DES MISSIONS :**

### Les axes prioritaires :

Dans cette catégorie, nous classerons dans cet ordre :

- ➔ L'analyse de l'application existante avec l'élaboration d'un schéma global explicatif ;
- ➔ La documentation de l'application ;
- ➔ L'ajout ou modification de monture afin que cela soit de nouveau opérationnel.
- ➔ La **sécurisation** de l'accès à l'application via un formulaire de connexion.

### Les axes secondaires :

- ➔ L'export de catalogue ;
- ➔ Présenter l'architecture et l'arborescence de l'application ;
- ➔ Optimiser le code pour supprimer les répétitions relatives à la connexion à la base de données ;
- ➔ Supprimer le profil de l'utilisateur depuis Meyelook ;

### Les optionnels :

- ➔ Autoriser la suppression d'une monture via une confirmation à l'utilisateur ;
- ➔ Analyse d'une nouvelle proposition intégrant de nouvelles tables en base de données pour l'ajout de catalogue dans un premier temps.
- ➔ Compléter l'exportation du catalogue en fonction du pourcentage de la qualification de la monture (100% étant attribué pour une qualification de monture complété entièrement).

## **5 : PLANIFICATION DES LIVRABLES ET TESTS**

Après une période d'adaptation et de prise en main de l'équipement et des outils, je m'approprie les logiciels, le repository de l'application sur mon poste. Je teste de fonctionnement actuel sur ma machine.

Je démarre la première version de 'Meyelook' Version Alpha 1.0 pour FittingBox.

### Les axes prioritaires

15/01/2019 : Analyse du contenu de l'application, les possibilités, les actions, les erreurs. Reports... Schéma explicatif global du fonctionnement de l'application. Soumis à l'approbation du chef de l'équipe produit.

29/01/2019 : Gestion des erreurs lors de l'ajout des montures via le fichier Excel. Test sur machine du bon fonctionnement.

05/02/2019 : Sécurisation de la connexion à l'application.

### Les axes secondaires

07/02/2019 : Exportation de catalogue pour toutes les montures.

15/02/2019 : Nouvelle architecture de l'application « Meyelook ».

19/02/2019 : Optimisation de la connexion en base de données.

20/02/2019 : Suppression du profil de l'utilisateur sur l'application.

### Les axes optionnels

26/03/2019 : Lors d'une suppression d'une monture, confirmer à l'utilisateur.

05/03/2019 : Analyse base de données et intégration du style.

15/03/2019 : Documentation du travail effectué durant la période de stage.

27/06/2019 : Déploiement en environnement de teste de la version Alpha 1.0 de MeyeLook

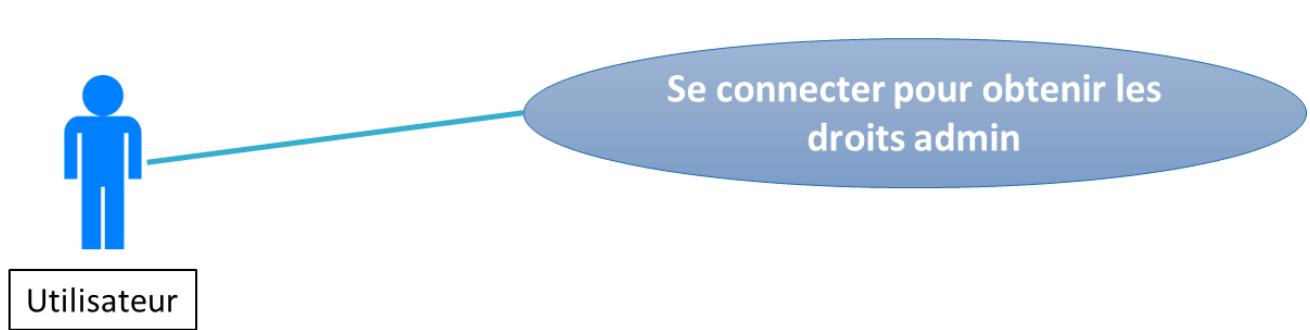
## **II – SPÉCIFICATIONS FONCTIONNELLES**

Au travers de « use cases » et de diagrammes de séquence, j’expliquerai les étapes qui permettent l’exécution d’un code en fonction de ce que l’on veut faire.

MeyeLook permet aux utilisateurs d’interagir avec des données enregistrées sur un serveur.

### **1 : USE CASES**

#### **AJ CONNEXION DE L’UTILISATEUR**

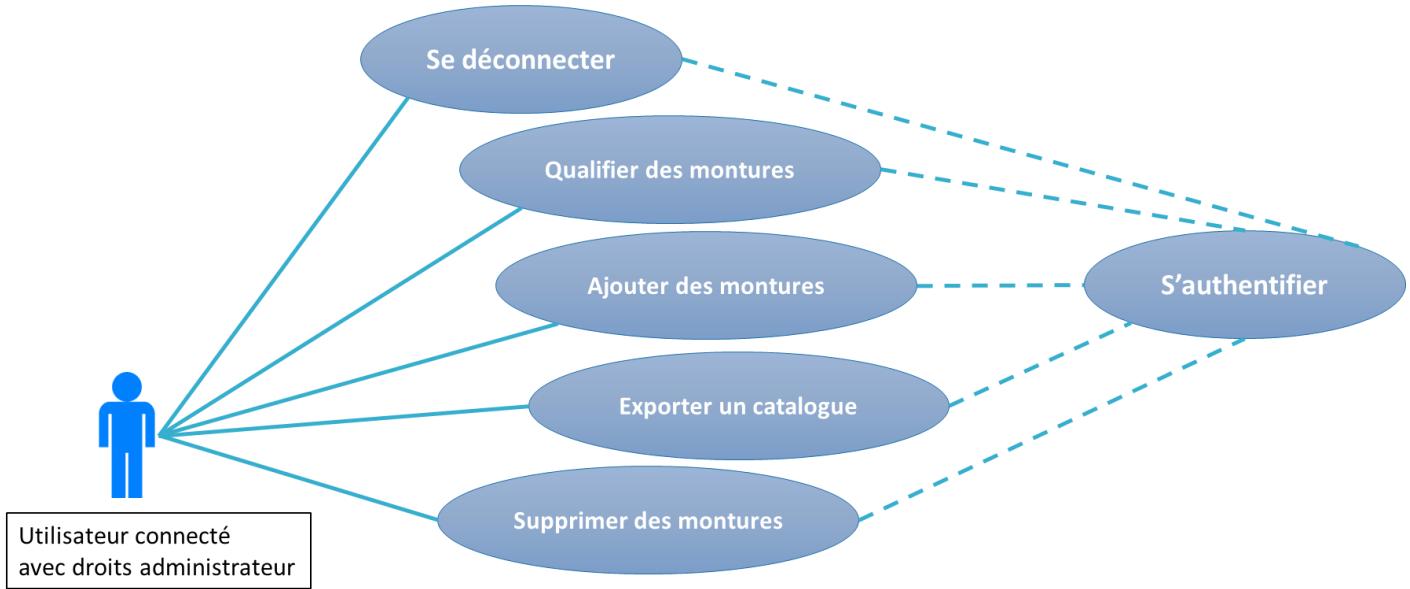


Les personnes amenées à intervenir sur « **MeyeLook** » sont des utilisateurs et membres de FittingBox. Ils ont le droit d’administrateur et peuvent effectuer toutes les modifications qu’ils souhaitent. Ils gèrent les clients de l’application MyFittingBox et les *licences* associées. Aujourd’hui, l’application n’est pas sécurisée. On souhaite établir un accès limité aux seuls personnes authentifiées par login et mot de passe.

Les identifiants de connexion sont fournis par le service interne qui gère de manière générale tout ce qui concerne les problèmes de connexion, session... Si l’on ne dispose pas des codes, il suffira de faire une demande ou *ticket*. Il en est de même pour les pertes et oubli d’identifiants.

Un fois l’utilisateur connecté, il a les pleins pouvoirs ! Il est administrateur de l’application et peut faire toutes les modifications ou suppression qu’il souhaite.

## **B) LES ACTIONS DE L'UTILISATEUR**



L'utilisateur connecté peut agir sur l'ensemble des stocks de montures. Il est administrateur.  
Listons les possibilités, il peut :

- Fermer sa session.
- Visualiser l'ensemble des montures.
- Télécharger un fichier pour inclure de nouvelles montures ou alors effectuer des mises à jour concernant les montures existantes en base données.
- Exporter un catalogue, c'est-à-dire récupérer dans un fichier toutes les montures étant enregistrées sur un catalogue bien défini et sélectionné en amont.
- Visualiser les catalogues.
- « *Qualifier* » des montures (modifier certaines informations).
- Supprimer des montures

## **2 : DIAGRAMME DE SEQUENCE**

J'ai choisi de présenter 2 diagrammes de séquence : « **AJOUT / MISE A JOUR DES MONTURES** » et « **SUPPRIMER UNE MONTURE** » qui me paraissent intéressants.

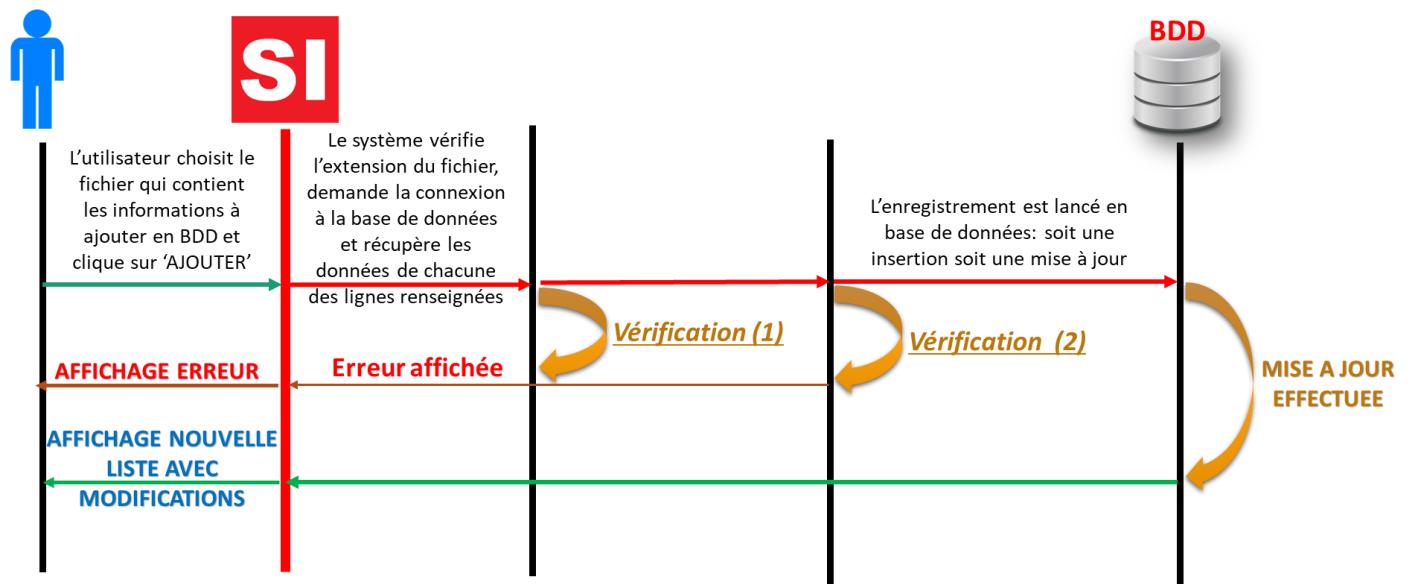
Je joins, en annexes, les diagrammes de séquence « **CONNEXION DE L'UTILISATEUR** » et « **EXPORTER UN CATALOGUE** ».

*On ne vérifie pas les identifiants de l'utilisateur car ils sont vérifiés lors de l'accès à l'application et il a un droit d'administrateur une fois connecté.*

## AJOUT OU MISE A JOUR DES MONTURES

Dans le corps de la page du Tableau de bord, l'administrateur clique sur l'icône « **outils** » qui ouvre une nouvelle fenêtre avec la possibilité soit d'ajouter un catalogue soit d'exporter un catalogue.

Avec '*Choose File*', on sélectionne le fichier qui contient nos informations d'ajout ou de mise à jour de montures et on clique sur *Ajouter*.



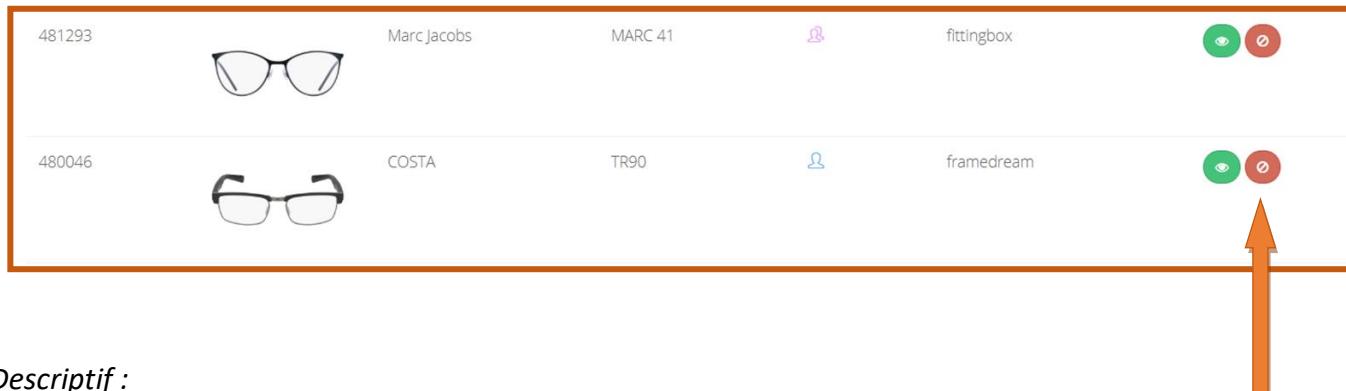
Vérification (1): 1-1: Type de Fichier non valide

1-2: Connexion à la base de données impossible ou connexion à MeyeLook impossible (licence indisponible)

Vérification (2): 2-1: Erreur Requête (ou code)

## B1 SUPPRIMER UNE MONTURE

Via le tableau de bord, il est tout à fait possible de gérer la suppression d'une monture mais ce n'était pas fonctionnel. Pour réintroduire la suppression, bien entendu avec confirmation demandée à l'utilisateur, je choisis de mettre en place l'apparition d'une modal tout comme cela est fait pour les catalogues.



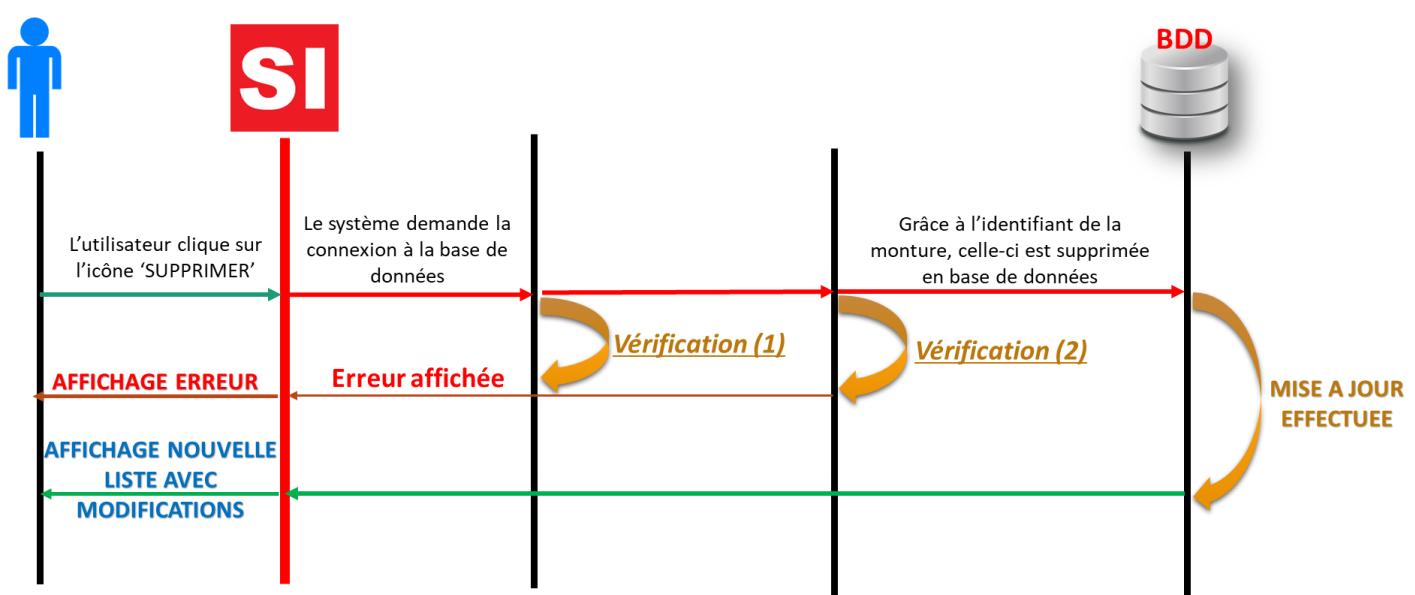
### Descriptif:

Je suis sur la page des montures et je dispose de deux petites icônes à droite de chaque monture.

La première icône verte nous mène sur la page de description de la monture sélectionnée (la page de qualification).

La seconde nous propose de supprimer la monture associée.

Au clic sur l'icône « supprimer », la monture sélectionnée est supprimée de la base de données.



Vérification (1): 1-1: Connexion impossible

1-2: Connexion à la base de données impossible ou connexion à MeyeLook impossible (licence indisponible)

Vérification (2): 2-1: Erreur Requête (ou code)

### ***3 : MAQUETTAGE***

#### ***AJ CONNEXION DE L'UTILISATEUR***

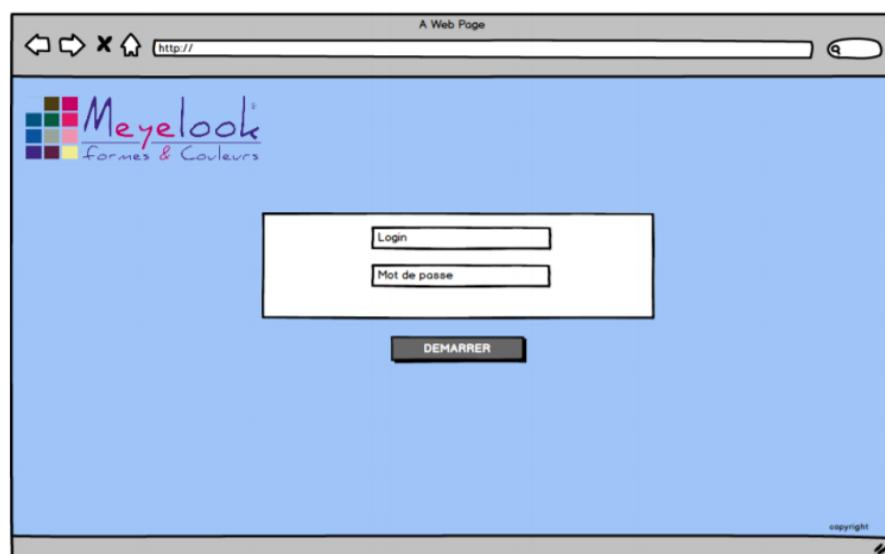
J'ai choisi de présenter une maquette réalisée à l'aide de « balsamiq » et en n'utilisant que l'image de l'application (MeyeLook) et les formulaires en m'aidant de la librairie « Bootstrap ». Les couleurs ne sont pas la priorité, je me contente des fonctionnalités de base de CSS de Bootstrap, nous pourrons y revenir plus tard pour plus personnaliser les couleurs.

#### **Maquette Page 1 ACCUEIL :**



Logo de l'application. Le bouton « M'IDENTIFIER » lance la page connexion avec un formulaire à compléter.

#### **Maquette Page 2 FORMULAIRE :**



- Logo de l'application plus petit format cliquable qui nous ramène à l'accueil. Le bouton « M'IDENTIFIER » lance la page connexion avec un formulaire à compléter.
- Champs « login » et « mot de passe » à remplir pour l'identification de l'utilisateur.
- Le bouton « DEMARRER » qui lance la vérification des identifiants puis, le cas échéant, la redirection vers le Tableau de bord.

## **B1 MODAL POUR SUPPRESSION DE MONTURE**

Pour gérer la suppression de la monture, nous devons pouvoir confirmer notre choix via un bouton « confirmer la suppression ». Pour cela, je m'inspire de la modal existante pour la suppression des catalogues afin de garder le même type de modal pour que le tout reste homogène.

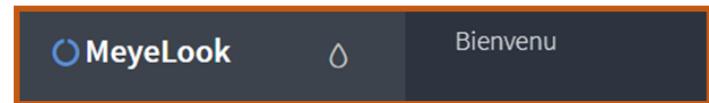


## **4 : FONCTIONNALITES DIVERSES**

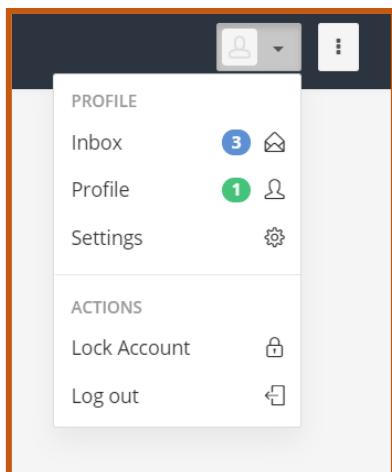
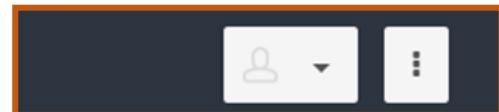
### **A) SUPPRESSION DU MENU DE GESTION DE PROFIL DE L'UTILISATEUR**

La première page du tableau de bord nous permet d'accéder à l'interface et nous propose le menu du profil représenté sous forme d'avatar. Le menu apparaît au clic sur l'icône.

Voici comment est composé le « header ». Un message de bienvenue et le nom de l'application à gauche



A droite, un menu d'identification présenté par un icône.



Quand on clique, on affiche le menu de gestion de profil avec la possibilité d'accéder aux mails de l'utilisateur, de faire des réglages pour changer l'image, la couleur...

L'application ne proposera plus ses fonctionnalités qui étaient prévues mais non opérationnelles. Les liens nous envoient vers d'autres pages qui n'existent pas donc vers une page du navigateur avec un message d'erreur « 404 Not Found ».

### **Not Found**

The requested URL was not found on this server.  
Apache/2.4.35 (Win64) PHP/7.2.10 Server at localhost Port 8080

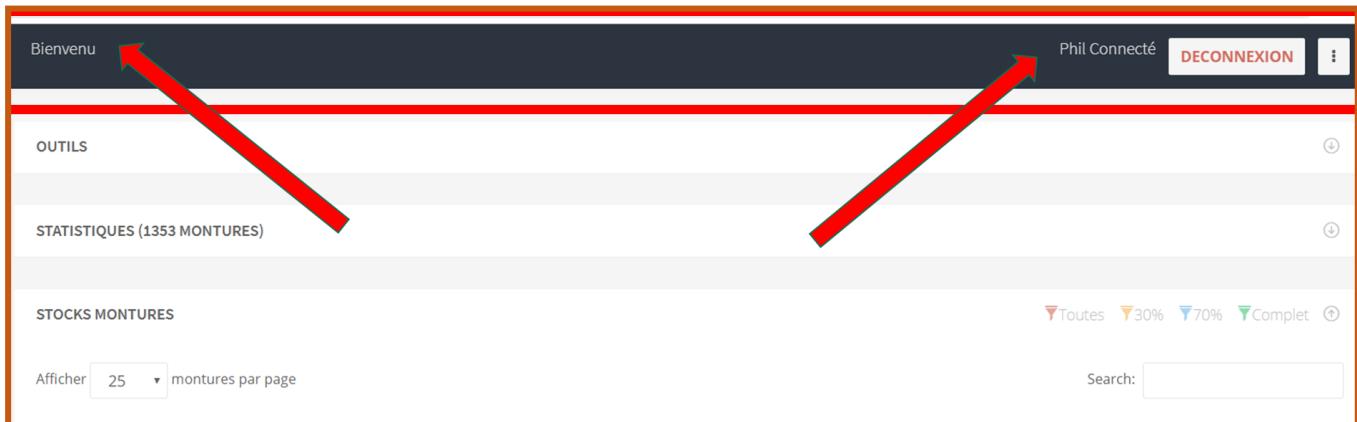
Sur l'application, on ne gèrera pas le profil de l'utilisateur ni même les identifiants donc le menu présent sera supprimé et remplacé par un seul bouton « DECONNEXION » qui permettra à l'utilisateur de se déconnecter.

Le rendu final ressemblera à ceci :



## **B1 CENTRER LE MESSAGE D'ACCUEIL ET LE LOGIN DE L'UTILISATEUR**

Le login est repris avec le mot 'Connecté' pour identifier l'utilisateur.



D'un point de vue esthétique, il vaut mieux ajouter quelques modifications et centrer le mot « bienvenue » après corrections (« bienvenue » au lieu de « bienvenu ») et le nom de l'utilisateur qui s'est authentifié !

De plus, au lancement, une animation doit se produire lorsque l'utilisateur arrive sur le tableau de bord mais elle n'apparaît pas, il faut donc la faire fonctionner.

Les animations ne fonctionnent pas car elles sont encapsulées dans des balises `<span>` et elles ne peuvent pas s'exécuter sur un élément `<span></span>` de par le fait que ces balises sont par défaut en « `display : inline` ». Je déplace l'animation vers la balise `<li>` qui est lié au login. Je choisis de placer mes éléments dans une « `<div></div>` ».

Pour le message de bienvenue, je rajoute un élément `<div>` dans lequel j'affecte l'animation du message et je garde mes éléments `<span>` à l'intérieur des balises `<div>`.

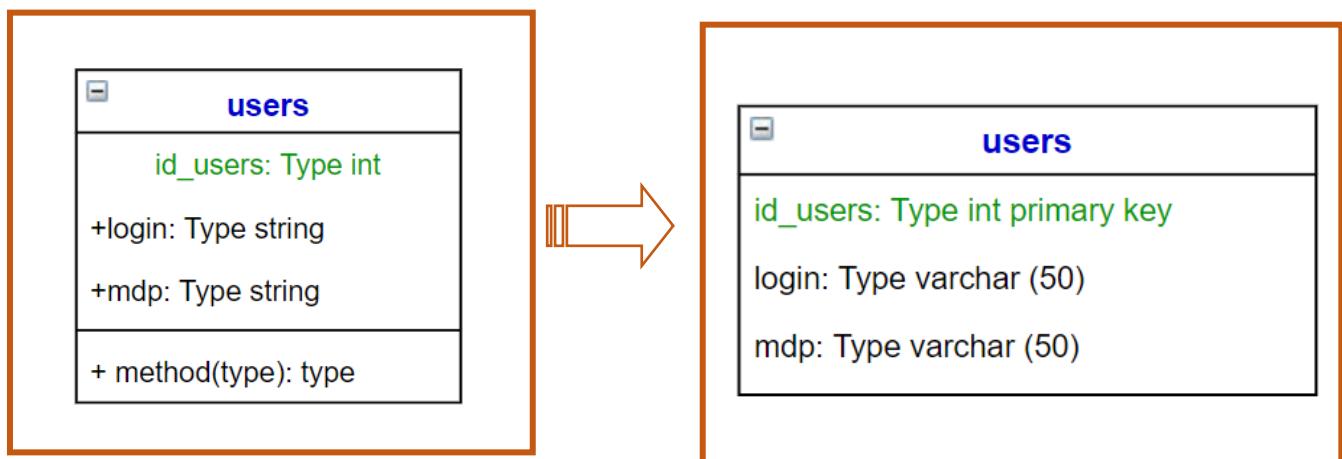
L'animation consiste à faire apparaître le mot « bienvenue » et le login de l'utilisateur avec un effet de « `zoomIn` », c'est-à-dire qu'il apparaît en petit puis grossit jusqu'à gagner sa place définitive.

## III – CONCEPTION

### 1 : CONNEXION DE L'UTILISATEUR

Pour pouvoir identifier un utilisateur, il doit exister dans la base de données. Je rajoute donc en base de données une table ‘**users**’ qui regroupera les champs id\_users, login et mot de passe. La typologie des 3 champs de cette table est respectivement ‘Integer’ pour l’id\_users (qui est clé primaire de la table et en auto-incrémentation), et ‘Chaîne de caractères’ pour les champs login et mdp.

J'y ai intégré 3 utilisateurs pour les premiers tests. Le mot de passe sera haché.



En base de données dans PhpMyAdmin,

	+ Options	← →	▼	id_users	login	mdp
<input type="checkbox"/>	Edit  Copy  Delete			1	Jamhs	81dc9bdb52d04dc20036dbd8313ed055
<input type="checkbox"/>	Edit  Copy  Delete			2	Phil	456c3816494a33a4b404583cc54ffcb7
<input type="checkbox"/>	Edit  Copy  Delete			3	jaja	bb0ed6ad56f41c6de469776171261226

## **2 : ATTRIBUTION DU STYLE A UNE MONTURE**

Actuellement, les paramètres styles sont stockés dans un fichier sous forme de tableaux tels que :

`$var_styles = array ("none", "Classique", "Fashion", "Vintage", "Sport", "Design", "Urbain")` qui regroupe les styles.

Pour les qualifier, le fichier tri les marques en fonction du style...

```
$marque_fashion = array (  
    "BAILA",    "ATTITUDEFASHION",    "LESPERSONNALITES",    "KARLLAGERFELD",    "RAYBAN",    "GUCCI",  
    "HUGOBOSS",  "PAULANDJOE",      "VOGUE",      "ESPRIT",      "CALVINKLEIN",    "BOSSORANGE",    "JUSTCAVALLI",  
    "ROCHAS",    "SONIARYKIEL",     "GUESS",     "SAINTLAURENT",  "CHRISTIANLACROIX", "CERRUTI",     "BENETTON",  
    "CKCALVINKLEIN", "KENZO", "NINARICCI"  
);  
  
$marque_vintage = array ("ZADIGETVOLTAIRE", "MEDLEY");  
  
$marque_sport = array ("RIPCURL", "LEVIS", "OXYDO", "CARRERA", "RAYBAN");  
  
$marque_design = array ("MODEINFRANCE", "PURE");
```

La réflexion s'oriente autour de l'intégration de ces paramètres en base de données et non plus en 'dur' dans un fichier. Ainsi, lorsque l'on voudra modifier ou ajouter des styles, ou des marques, ces modifications se feront directement en base de données. Le programme continuera de fonctionner comme actuellement. (Document en annexes « Intégration du style des montures ajoutées à l'aide d'un fichier »).

## **3 ET 4 : MCD ET MLD**

Je traiterai ici uniquement le cas de la marque des montures en fonction du style choisi, le travail autour de la base de données continuera pour inclure les autres caractéristiques (correction, attentes, activités...).

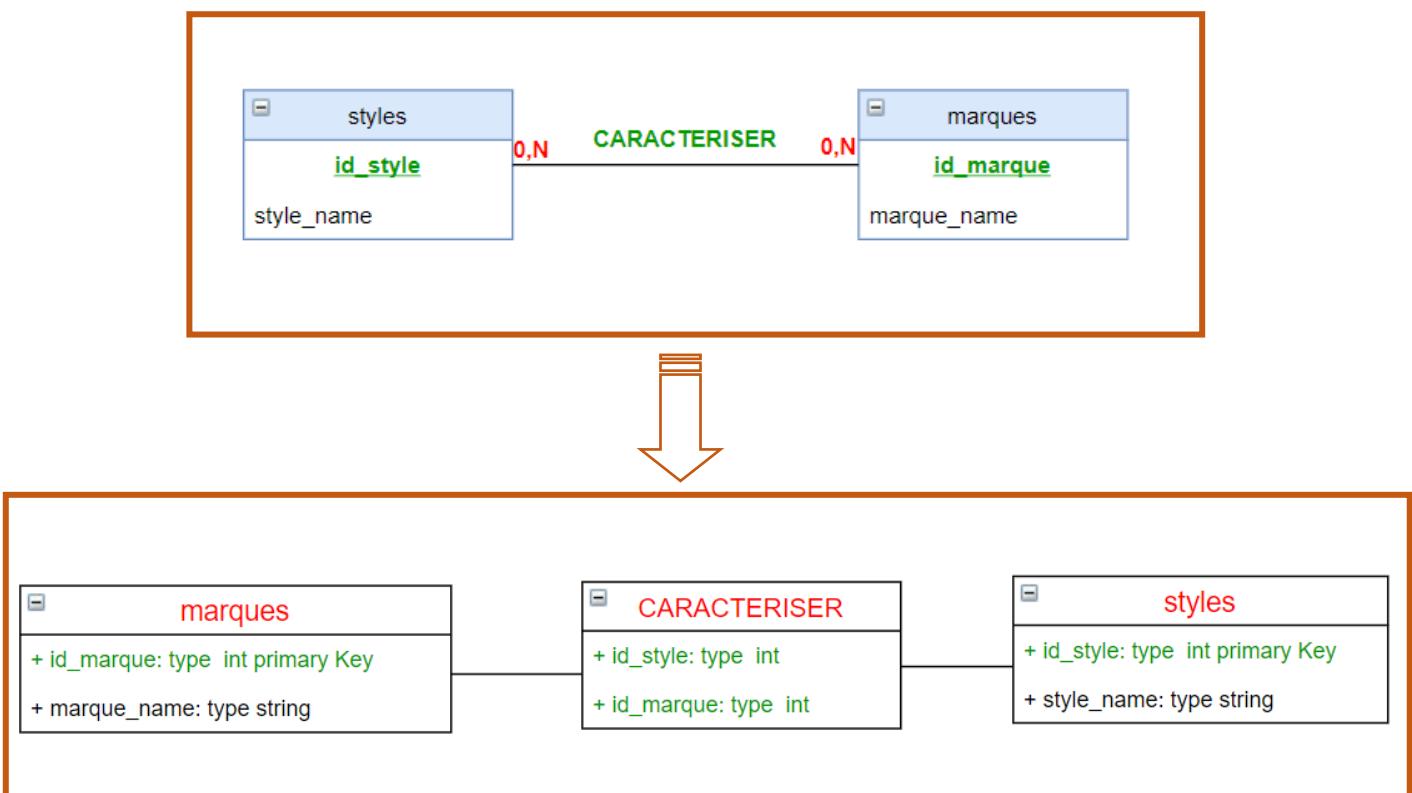
Un style regroupe plusieurs marques, par exemple pour le style « sport » on aura les marques :

« RIPCURL », « LEVIS », « OXYDO », « CARRERA », « RAYBAN ».

Pour le style fashion, on retrouve les marques :

« BAILA », « ATTITUDE FASHION », « LES PERSONNALITES », « KARL LAGERFELD », « RAY BAN », « GUCCI », « HUGO BOSS », « PAUL AND JOE », « VOGUE », « ESPRIT », « CALVIN KLEIN », « BOSS ORANGE », « JUST CAVALLI », « ROCHAS », « SONIA RYKIEL », « GUESS », « SAINT LAURENT », « CHRISTIAN LACROIX », « CERRUTI », « BENETTON », « CK CALVIN KLEIN », « KENZO », « NINA RICCI ».

Mais une marque peut être qualifiée par plusieurs styles comme dans ce cas « RAYBAN » appartient aussi bien au style sport qu'au style fashion.



Je vais donc intégrer les tables marques et styles. Je relie ces tables grâce à une table associative ‘caractériser’ à ma base de données.

Table ‘styles’

		+ Options			
		← T →			
			id_style	nom_style	
<input type="checkbox"/>		Edit			1 none
<input type="checkbox"/>		Edit			2 classique
<input type="checkbox"/>		Edit			3 fashion
<input type="checkbox"/>		Edit			4 vintage
<input type="checkbox"/>		Edit			5 sport
<input type="checkbox"/>		Edit			6 design
<input type="checkbox"/>		Edit			7 urbain
<input type="checkbox"/>		With selected:			

Tables ‘marques’

	+ Options	← T →		id_marque	name_marque
<input type="checkbox"/>		Edit			1 BAILA
<input type="checkbox"/>		Edit			2 ATTITUDE FASHION
<input type="checkbox"/>		Edit			3 LES PERSONNALITES
<input type="checkbox"/>		Edit			4 KARL LAGERFELD
<input type="checkbox"/>		Edit			5 RAY BAN
<input type="checkbox"/>		Edit			6 GUCCI
<input type="checkbox"/>		Edit			7 HUGO BOSS
<input type="checkbox"/>		Edit			8 PAUL AND JOE
<input type="checkbox"/>		Edit			9 VOGUE
<input type="checkbox"/>		Edit			10 ESPRIT
<input type="checkbox"/>		Edit			11 CALVIN KLEIN
<input type="checkbox"/>		Edit			12 BOSS ORANGE
<input type="checkbox"/>		Edit			13 JUST CAVALLI
<input type="checkbox"/>		Edit			14 ROCHAS
<input type="checkbox"/>		Edit			15 SONIA RYKIEL
<input type="checkbox"/>		Edit			16 GUESS

Table associative ‘caractériser’

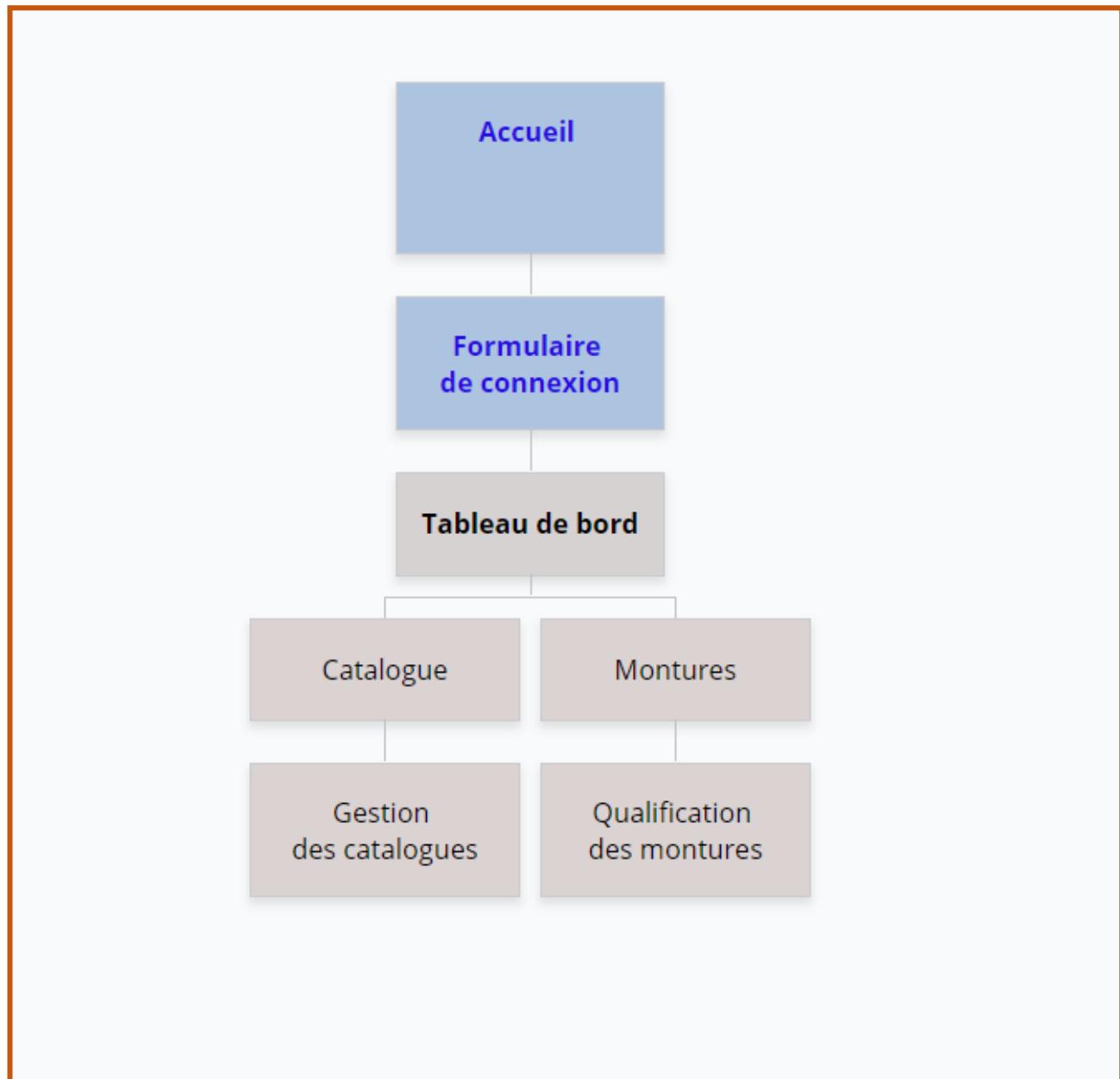
	+ Options	← T →		id_caracteriser	id_marque	id_style
<input type="checkbox"/>		Edit			1	1 3
<input type="checkbox"/>		Edit			2	2 3
<input type="checkbox"/>		Edit			3	3 3
<input type="checkbox"/>		Edit			4	4 3
<input type="checkbox"/>		Edit			5	5 3
<input type="checkbox"/>		Edit			6	6 3
<input type="checkbox"/>		Edit			7	7 3
<input type="checkbox"/>		Edit			8	8 3
<input type="checkbox"/>		Edit			9	9 3
<input type="checkbox"/>		Edit			10	10 3
<input type="checkbox"/>		Edit			11	11 3
<input type="checkbox"/>		Edit			12	12 3
<input type="checkbox"/>		Edit			13	13 3
<input type="checkbox"/>		Edit			14	14 3
<input type="checkbox"/>		Edit			15	15 3
<input type="checkbox"/>		Edit			16	16 3
<input type="checkbox"/>		Edit			17	17 3
<input type="checkbox"/>		Edit			18	18 3

## IV – ARBORESCENCE ET ARCHITECTURE

### 1 : ARBORESCENCE

Elle se présente simplement ainsi à présent que nous avons ajouté l'étape de connexion. Au lancement de l'application, il est indispensable de s'identifier. Après avoir cliqué sur « M'identifier », nous sommes dirigés vers la page de formulaire de connexion où nous sommes invités à saisir nos login et mot de passe.

Puis le tableau de bord s'ouvre et nous accédons aux onglets à gauche « Catalogue » et « Montures », chacun menant respectivement à la gestion des catalogues et à la qualification des montures.

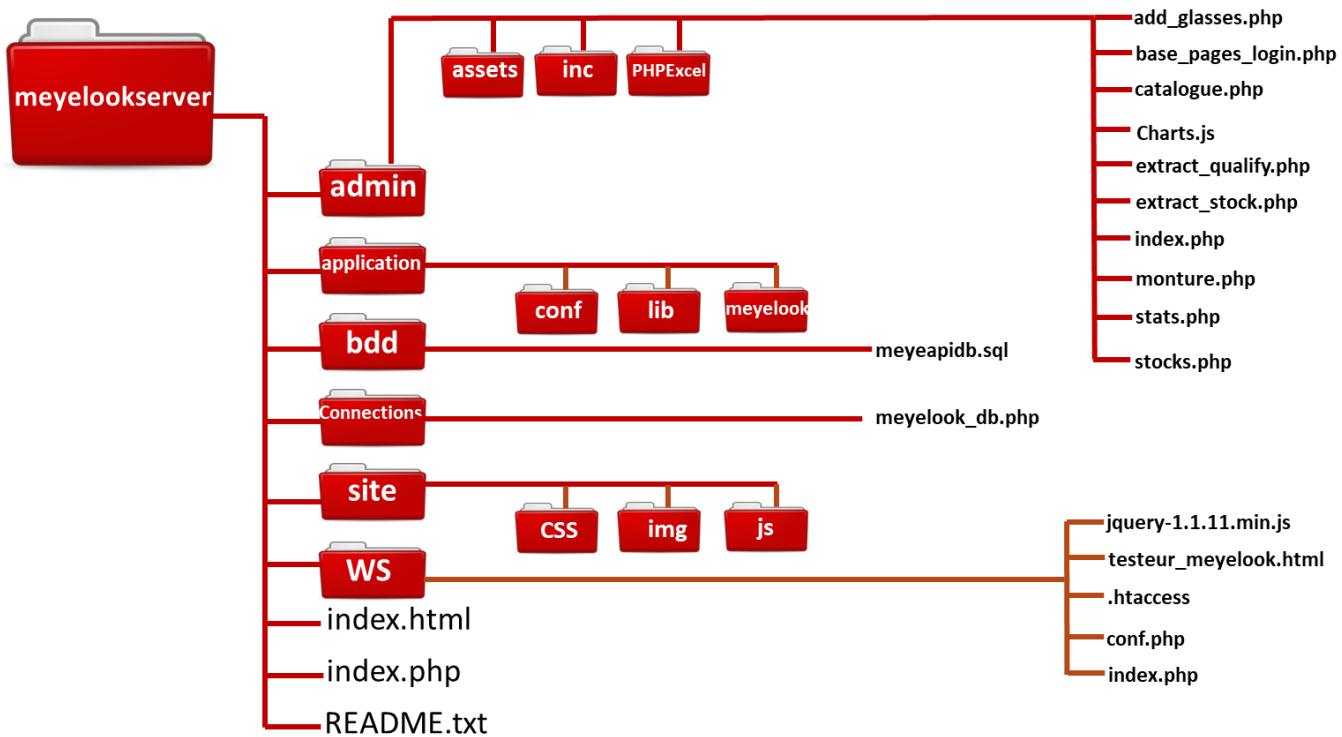


## 2 : ARCHITECTURE

### A] Architecture initiale

L'application existante est difficile à cerner. Les fichiers ne sont pas structurés et mélangés. Je fais un bilan de ce dont nous disposons et je change l'*architecture* pour atteindre une meilleure vision de l'application.

### Schéma de l'arborescence de l'application Meyelook



On retrouve donc à la racine **6 dossiers** :

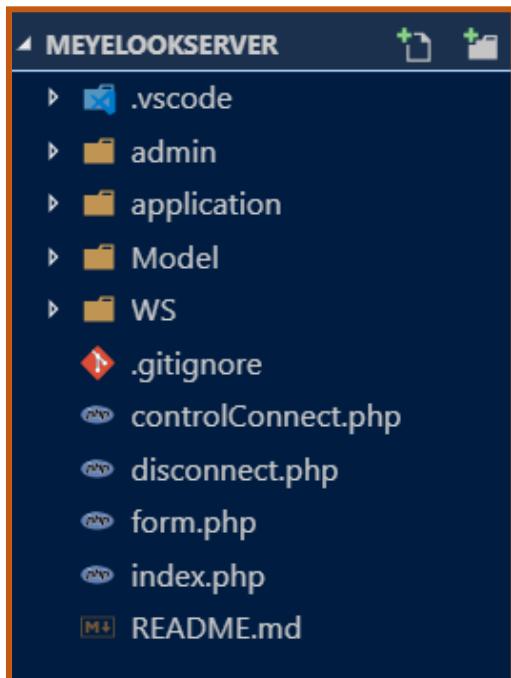
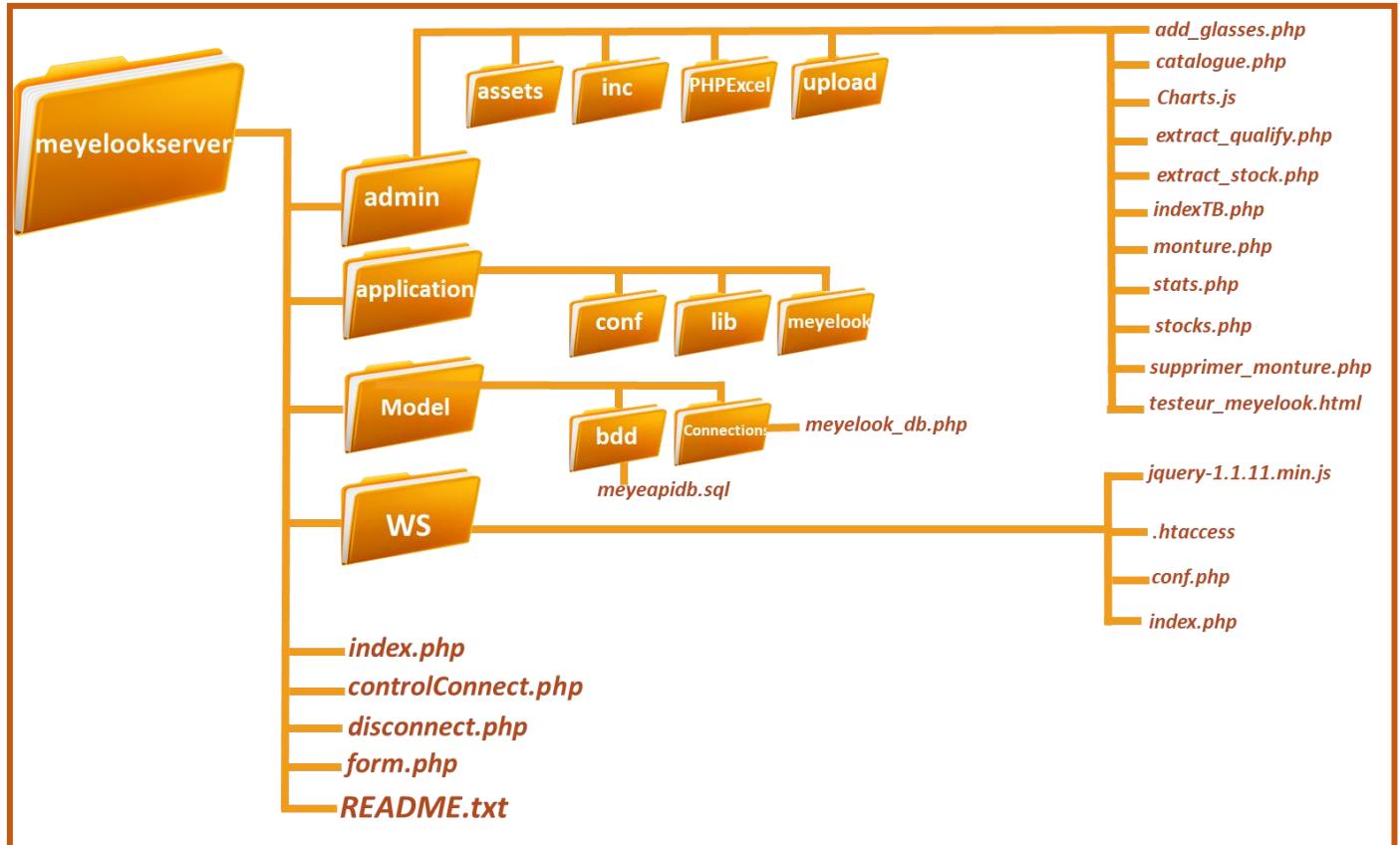
- Le dossier admin qui contient lui-même 3 dossiers et plusieurs sous niveaux que je détaillerais plus loin,
- Le dossier application avec 3 dossiers à l'intérieur,
- Le dossier bdd qui lui contient l'export de la base de données de la V6 de MeyeLook,
- Le dossier Connections qui établit la connexion à la base de données mais qui n'était pas exploité à son maximum,
- Le dossier site avec 3 dossiers,
- Le dossier WS par rapport au Web Service.

Et **3 fichiers** : « index.php », « README.txt » et « index.html ».

## **B) NOUVELLE ARCHITECTURE**

J'ai rassemblé tous les fichiers JavaScript, CSS ainsi que les images dans le dossier « admin/assets » qui est logiquement prévu pour cela, nous permettant d'avoir une structure plus claire.

Les modifications ont été effectuées dans les fichiers pour que les chemins soient respectés et que l'application continue de fonctionner.



Ainsi, lorsque nous voulons récupérer des informations ou ouvrir un fichier en particulier, il est aisément de le retrouver dans notre éditeur de texte après avoir ouvert notre dossier.

## V – SPÉCIFICATIONS TECHNIQUES

### 1 : GESTION DES FICHIERS : LES ETAPES DE REALISATIONS

#### A] Ajout ou mise à jour des montures

Que se passe-t-il ?

```
//Pour la feuil 1
if($sheet->getTitle() == 'Feuil1'){
    $glassesID_use = array();
    $glassesID_del = array();
    // On boucle sur les lignes
    foreach($sheet->getRowIterator() as $row) {
        $row_current = $row->getRowIndex();
        if($row_current>1){
            $A=$sheet->getCell("A".$row_current)->getValue();
            $E=$sheet->getCell("E".$row_current)->getValue();
            $F=$sheet->getCell("F".$row_current)->getValue();
            $G=$sheet->getCell("G".$row_current)->getValue();
            $H=$sheet->getCell("H".$row_current)->getValue();
            $J=$sheet->getCell("J".$row_current)->getValue();
            $K=$sheet->getCell("K".$row_current)->getValue();
            $L=$sheet->getCell("L".$row_current)->getValue();
            $M=$sheet->getCell("M".$row_current)->getValue();

            $marque_monture = $E;
```

Tout d'abord, le fichier chargé est vérifié. S'il ne s'agit pas d'un fichier Excel, alors l'IHM affiche un message d'erreur. Si l'extension du fichier est bonne, alors on lance une connexion à la base de données. Grâce à la bibliothèque PHPExcel, on récupère, le contenu des cellules dans le fichier pour les associer aux qualifications des montures.

```

// Traitement du style par marque
if(in_array($marque_monture, $marque_fashion)){
    $style_monture = "2";
}elseif(in_array($marque_monture, $marque_vintage)){
    $style_monture = "3";
}elseif(in_array($marque_monture, $marque_sport)){
    $style_monture = "4";
}elseif(in_array($marque_monture, $marque_design)){
    $style_monture = "5";
}else{
    $style_monture = "1";
}

// récupération deu GlassID de FBX
$id_fb_monture = intval($A);
// récupération du modèle
$modele_monture = $F;
// récupération de la référence couleur
$ref_coul_monture = $G;
// récupération de la boxing
$boxing_monture = intval($J);
// récupération de la marque
$taille_branche_monture = "NP";
// récupération du nez
$nez_monture = intval($K);
// récupération des branches
$branche_monture = intval($L);
// récupération du catalogue
$stock_monture = $M;
// récupération du sexe
if($H=="Male"){
    $sexe_monture = 2;
}elseif($H=="Female"){
    $sexe_monture = 1;
}else{$sexe_monture = 3;};
// Date de l'ajout de la monture
$date_monture = date("Y-m-d", strtotime("now"));

```

On stocke les valeurs dans des variables (la marque, le modèle, le style, le sexe (genre), la taille des branches... ainsi que la date d'enregistrement de la monture. On envoie une requête SQL vers la base de données pour procéder à l'insertion ou à la mise à jour des montures (si elle existe, on la modifie, sinon on la rajoute). Une fois le code exécuté, on retourne dans le tableau de bord avec un rafraîchissement de page qui prend en compte les modifications.

L'exécution de l'action va enregistrer le fichier concerné dans le dossier **admin**.



dans le dossier

## Résultats

Je crée un fichier Excel dans lequel j'enregistre des montures « factices » pour faire les tests.

ID	PHOTO	MARQUE	MODÈLE	SEXÉ	CATALOGUE	Outils
481293		Marc Jacobs	MARC 41		fittingbox	
480046		COSTA	TR90		framedream	

Sélection du fichier →

AJOUTER UN CATALOGUE

 ajoutmonturesfactices.xlsx 

Contenu du fichier

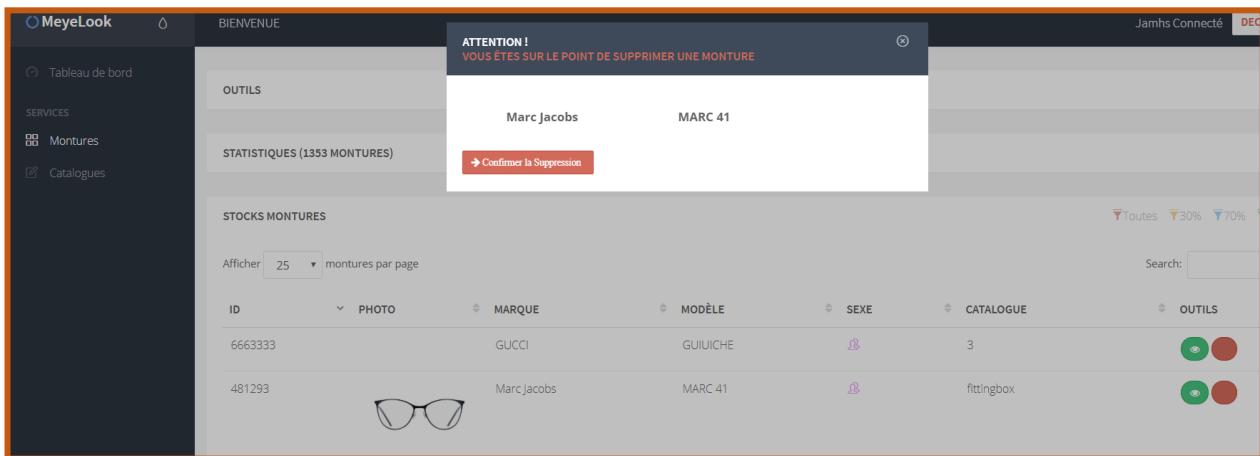
glassId	UPC	EAN	GTIN	Brand	Model	Color	Kind	Type	lensWidth	bridgeSize	templeSize	catal
6663333				GUCCI	GUIUCHE	2000	Unisex	Ophthalmic	52	16	140	fittin
9995666				CHANEL	CORICO	BK	Femme	Ophthalmic	55	0	0	fittin
3333333				RONALDO	SUPERCR7	BK	Homme		32	12	15	foot

IHM après prise en compte des nouveautés

ID	PHOTO	MARQUE	MODÈLE	SEXÉ	CATALOGUE	Outils
9995666		CHANEL	CORICO		fittingbox	
6663333		GUCCI	GUIUCHE		fittingbox	
3333333		RONALDO	SUPERCR7		foot2018	
481293		Marc Jacobs	MARC 41		fittingbox	
480046		COSTA	TR90		framedream	

## B1 SUPPRIMER UNE MONTURE

Une modal ? Comment ça ? Grâce à Bootstrap, je fais apparaître la modal qui lui demandera de confirmer la suppression à l'utilisateur. Je la réalise à l'image de la maquette proposée et validée par mon tuteur.



Je

procède donc aux modifications nécessaires dans les fichiers « `admin\catalogue.php` » et « `admin\assets\css\oneui.css` ». L'ajout de la modal et de la requête SQL pour la suppression :

```
if(isset($_POST["form_type"])){
    if($_POST["form_type"]=="delete_monture"){
        $req_delete_stock_in_monture = $bdd->prepare('DELETE FROM montures WHERE id_fb_monture = ?');
        $req_delete_stock_in_monture->execute(array($_POST['monture_id']));
    }
}
```

Au clic, je récupère les informations que je souhaite faire afficher dans la modal.

```
$(document).on("click", ".option", function () {
    var myformInfos = $(this).attr('id');
    var arr = myformInfos.split(';');
    let monture_id = arr[0];
    $.get("getMontureByID.php", { id_fb_monture: monture_id }, function (dataMonture) {
        Obj_Monture = JSON.parse(dataMonture);
        console.log(Obj_Monture[0].marque_monture, Obj_Monture[0].modele_monture);
        $("input[type=text].monture_marque").val(Obj_Monture[0].marque_monture);
        $("input[type=text].monture_model").val(Obj_Monture[0].modele_monture);
        $("input[type=hidden].monture_id").val(Obj_Monture[0].id_fb_monture);
    });
});
```

J'ajoute également quelques informations à la génération de l'icône pour récupérer l'ID de la monture visée par la suppression. Pour cela, j'ajoute les lignes suivantes dans le fichier « `admin\inc\server_processing_monture.php` » :

```
tion( $d, $row ) { return '<div class="btn-group">
    <a href="monture.php?id_fb_monture=' . $d . '">
        <button class="btn btn-sm btn-success btn-rounded" type="button" data-toggle="tooltip" title="Visualiser la monture">
            <i class="fa fa-eye"></i>
        </button>
    </a>
    <a>
        <button class="btn btn-sm btn-danger btn-rounded" type="button" data-toggle="tooltip" title="Supprimer la monture">
            <i data-toggle="modal" data-target="#modal_delete_glasses" class="fa fa-ban option text-danger" style="cursor:pointer" id="'.$d.'"></i>
        </button>
    </a>
</div>';}
```

data-target dirige vers la modal qui aura pour ID `#modal_delete_glasses`.

```
<div class="form-group">
    <div class="col-xs-12">
        <div class="form-material floating">
            <input class="monture_marque mm" type="text" id="" name="monture_marque" value="" ></input>
            <input class="monture_model mm" type="text" id="" name="monture_model" value="" ></input>
            <input class="monture_id" type="hidden" id="" name="monture_id" value="" ></input>
            <input type="hidden" id="form_type" name="form_type" value="delete_monture" ></input>
        </div>
    </div>
<div class="form-group">
    <div class="col-xs-6">
        <button class="btn btn-sm btn-danger" type="submit" form="deleteMonture">
            <i class="text-uppercase"> confirmer la suppression</i>
        </button>
    </div>
    <!-- -->
    <div class="form-group" -->
        <div class="col-xs-6">
            <button class="btn btn-sm btn-success" type="submit" onclick='location.href="catalogue.php"' form="cancel">
                <i class="text-uppercase"> annuler</i>
            </button>
        </div>
    </div>
```

### Rendu final de la modal opérationnelle



## C] Centrer le message d'accueil et le login de l'utilisateur

Les fichiers affectés par des modifications sont :

« *admin/inc/views/base\_header.php* » et « *admin/assets/css/oneui.css* » pour le CSS.

Je rajoute aussi la redirection vers le fichier « *disconnect.php* ». Je modifie le contenu de la feuille de style associée afin de centrer et d'améliorer le visuel.

```
<span class="h5 text-white-op animated zoomIn">Bienvenu</span>
<ul class="nav-header pull-right">
    <li>
        <span class="h5 text-white-op animated zoomIn"><?php echo ($_SESSION['user'] . ' Connecté');?></span>
    </li>
    <li>
        <div class="btn-group">
            <button class="btn btn-default btn-image text-center">
                <a href="../disconnect.php" class="text-danger" id="deconnexion">DECONNEXION</a>
            </button>
        </div>
    </li>
    <li class="hidden-xs hidden-sm">
        <button class="btn btn-default" data-toggle="layout" data-action="sidebar_mini_toggle" type="button">
            <i class="fa fa-ellipsis-v"></i>
        </button>
    </li>
</ul>
```

Dans mon fichier « *admin/inc/views/base\_header.php* », à l'aide des « id » que j'ai ajouté, j'apporte les modifications dans la feuille de style associée « *admin/assets/css/oneui.css* ». Ces ajouts aligneront le texte et donneront le rendu souhaité sans affecter les animations.

```
<header id="header-navbar" class="content-mini d-flex content-mini-full">
    <!-- Header Navigation Right --&gt;

    &lt;div class="h5 text-white-op animated zoomIn"&gt;
        &lt;span id="bienvenue" class="text-uppercase"&gt;Bienvenue&lt;/span&gt;
        &lt;ul class="nav-header d-flex pull-right"&gt;
            &lt;li class="h5 text-white-op align-self-center animated zoomIn"&gt;
                &lt;span id="login"&gt;&lt;?php echo ($_SESSION['user'] . ' Connecté');?&gt;&lt;/span&gt;
            &lt;/li&gt;
        &lt;/ul&gt;
    &lt;/div&gt;</pre>
```

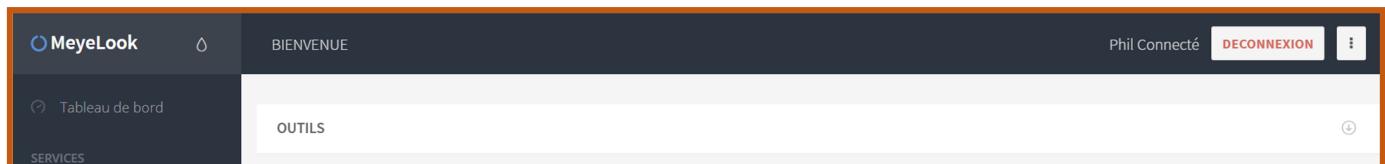
*admin/inc/views/base\_header.php*

```
689     }
690     #deconnexion{
691         font-weight: bolder;
692     }
693     .btn.btn-image {
694         position: relative;
695         padding-left: 10px;
696         padding-right: 10px;
697     }
```

```
148     color: #777;
149     font-size: 16px;
150     font-weight: 300;
151     line-height: 1.4;
152 }
153 #bienvenue, #login{
154     margin-top: 8px;
155     display: inline-block;
156 }
157
158 @media screen and (min-width: 768px) {
159     .page-heading small {
160         margin-top: 0;
161         display: inline;
162         line-height: inherit;
163     }
```

*admin/assets/css/oneui.css*

## Rendu Final



## **2 : ENVIRONNEMENT TECHNIQUE ET EQUIPEMENTS**

Le matériel proposé est un ordinateur de l'entreprise disposant d'une connexion WIFI et par câble au réseau internet et différents périphériques si besoin (imprimantes, scan...). Il est configuré avec une session à mon intention avec mot de passe et boîte mail dédiée.

Je dispose aussi d'un compte sur **Microsoft Teams**, une plateforme en ligne afin d'échanger avec l'ensemble des collaborateurs de l'entreprise y compris les membres de l'équipe au sein de laquelle j'évolue : « l'équipe produit ». Une adresse mail m'est affectée personnellement pour échanger avec les membres de FittingBox tous secteurs confondus. Je commence par télécharger et installer les logiciels et outils me permettant de travailler et de me connecter à mon nouveau réseau de collègues.

Première étape : récupérer le « repository » de l'application sur **GIT** afin d'accéder au code de l'application.

Pour travailler en local et ainsi pouvoir modifier et supprimer des données sans toucher à l'application existante, j'installe aussi **WAMP64**.

J'enregistre le dépôt GIT dans le répertoire www de Wamp. Mes collaborateurs utilisent **WEBSTROM** qui est un *IDE* sous License mais pour ma part, je travaillerai sur l'éditeur de texte **VISUAL STUDIO CODE**. Ce choix était cohérent puisque je connais déjà ce logiciel et un autre développeur de l'équipe travaille également avec. Grâce à cela, j'ai pu commencer plus rapidement les tâches qui m'étaient assignées.

En travaillant en PHP, la vérification des contenus de mes variables a souvent été fastidieux car cela impliquait la présence de `var_dump()` et autre `echo` au sein du code pouvant parfois être oublié(s) et n'affichant à l'écran les informations que lorsqu'il n'y avait pas de redirection vers d'autres page. J'ai appris à mieux utiliser **VisualStudioCode** avec l'extension **Xdebug** que j'ai donc installé.

**Sous Windows l'extension est active par défaut sur Wamp**

Version Apache : 2.4.35 - [Documentation](#)

Version de PHP : 7.2.10 - [Documentation](#)

Server Software : Apache/2.4.35 (Win64) PHP/7.2.10 - Port défini pour Apache : 80

Extensions Chargées :

apache2handler	bcmath	bz2	calendar	com_dotnet
Core	ctype	curl	date	dom
exif	fileinfo	filter	gd	gettext
gmp	hash	iconv	imap	intl
json	ldap	libxml	mbstring	mysqli
mysqlnd	openssl	pcre	PDO	pdo_mysql
pdo_sqlite	Phar	readline	Reflection	session
SimpleXML	soap	sockets	SPL	sqlite3
standard	tokenizer	wddx	xdebug	xml
xmlreader	xmlrpc	xmlwriter	xsl	Zend OPcache
zip	zlib			

Version de MySQL : 5.7.23 - Port défini pour MySQL : 3306 - Default DBMS - [Documentation](#)

Version de MariaDB : 10.3.9 - Port défini pour MariaDB : 3307 - [Documentation](#)

**Configuration Serveur**

Version Apache : 2.4.35 - [Documentation](#)

Version de PHP : 7.2.10 - [Documentation](#)

Server Software : Apache/2.4.35 (Win64) PHP/7.2.10 - Port défini pour Apache : 80

Extensions Chargées :

apache2handler	bcmath	bz2	calendar	com_dotnet
Core	ctype	curl	date	dom
exif	fileinfo	filter	gd	gettext
gmp	hash	iconv	imap	intl
json	ldap	libxml	mbstring	mysqli
mysqlnd	openssl	pcre	PDO	pdo_mysql
pdo_sqlite	Phar	readline	Reflection	session
SimpleXML	soap	sockets	SPL	sqlite3
standard	tokenizer	wddx	xdebug	xml
xmlreader	xmlrpc	xmlwriter	xsl	Zend OPcache
zip	zlib			

Version de MySQL : 5.7.23 - Port défini pour MySQL : 3306 - Default DBMS - [Documentation](#)

Version de MariaDB : 10.3.9 - Port défini pour MariaDB : 3307 - [Documentation](#)

**Outils**

- [phpinfo\(\)](#)
- [phpmyadmin](#)
- [Ajouter un Virtual Host](#)

**Vos Projets**

- [stage\\_Jamila](#)

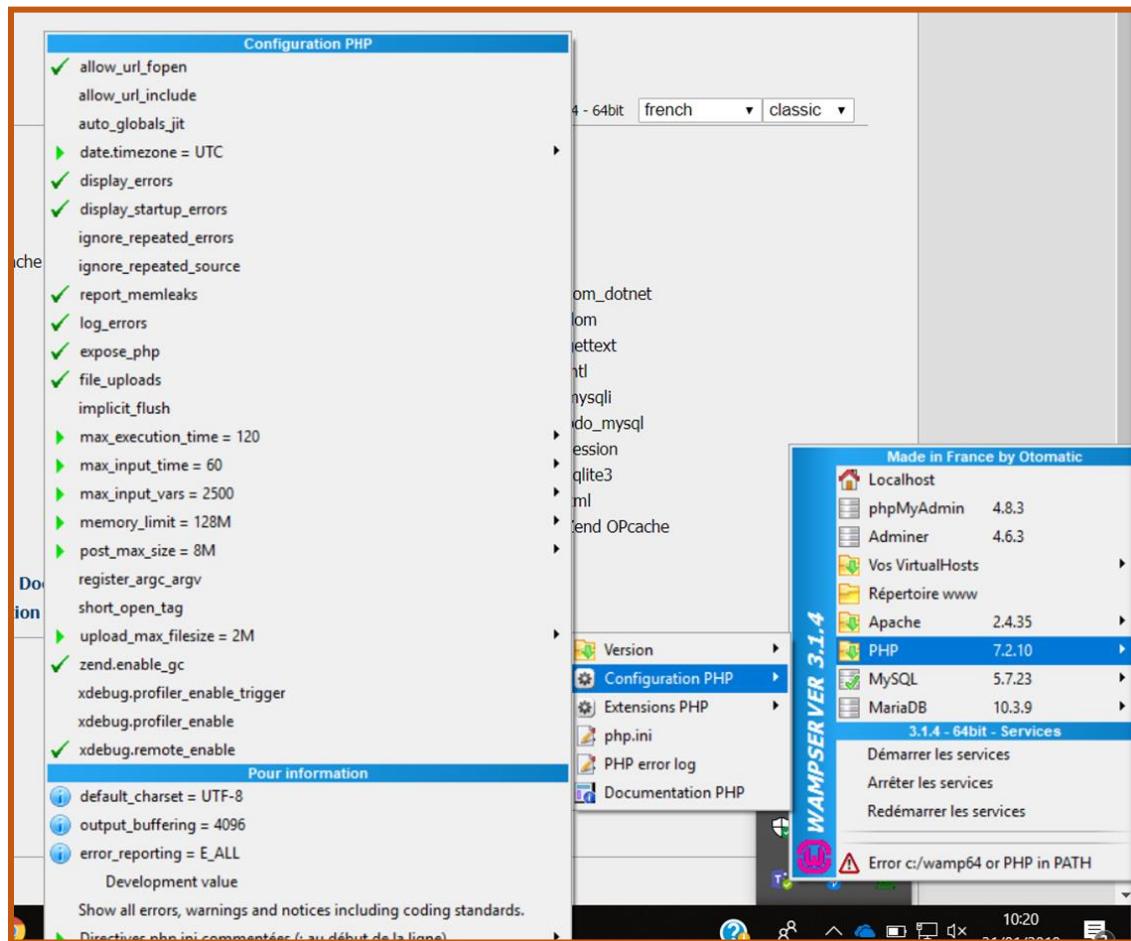
**Vos Alias**

- [adminer](#)
- [phpmyadmin](#)
- [phpsysinfo](#)

**Vos VirtualHost**

- [localhost](#)

Par défaut, la valeur de **xdebug\_remote\_enable** est à « **off** » on doit la passer sur « **on** » pour utiliser l'extension.



D'un clic gauche sur l'icône W de Wamp, je sélectionne la Configuration PHP et je clique sur Xdebug\_remote\_enable. Cette action relance WAMP.

**Xdebug\_remote\_enable** est sur **on** à présent. Il ne me reste qu'à profiter pleinement de l'extension **PHP Debug** sur Visual Studio Code.

xdebug.remote_autostart	Off	Off
xdebug.remote_connect_back	Off	Off
xdebug.remote_cookie_expire_time	0000	0000
<b>xdebug.remote_enable</b>	<b>On</b>	<b>On</b>
xdebug.remote_handler	dbgp	dbgp
xdebug.remote_host	localhost	localhost
xdebug.remote_log	no value	no value

On peut exploiter nos variables sans passer par les « var\_dump () » et autre « echo » habituels pas très pratiques pour cerner les problèmes de codes au risque de les oublier après.

Ce point a été important pour moi car j'ai appris à utiliser le débuggeur sur Visual Studio Code et il m'a permis d'étendre ma compréhension de cet éditeur et de ses fonctionnalités.

The screenshot shows the 'Variables' panel in Visual Studio Code. The title bar says '▼ VARIABLES'. Under the 'Locals' section, there are several variables listed with their values:

- \$A: 155218
- \$E: "Ray-Ban"
- \$F: "RB5187"
- \$G: "2000"
- \$H: "Unisex"
- \$J: 52
- \$K: 16
- \$L: 140
- \$M: "fittingbox"

Below the 'Locals' section, there are expanded sections for '\$bdd: PDO' and '\$extensions\_valides: array(3)'. The '\$bdd' section contains:

- \$boxing\_monture: 52
- \$branche\_monture: 140
- \$date\_monture: "2019-01-31"
- \$dbHost: "localhost"
- \$dbName: "meyearapidb\_"
- \$dbPass: ""
- \$dbUser: "root"
- \$dossier: "upload/"
- \$e: uninitialized
- \$extension\_upload: "xlsx"

The '\$extensions\_valides' section contains three arrays:

- \$glassesID\_del: array(0)
- \$glassesID\_in: array(560)
- \$glassesID\_use: array(0)

At the bottom of the list, \$id\_fb\_monture: 155218 is shown.

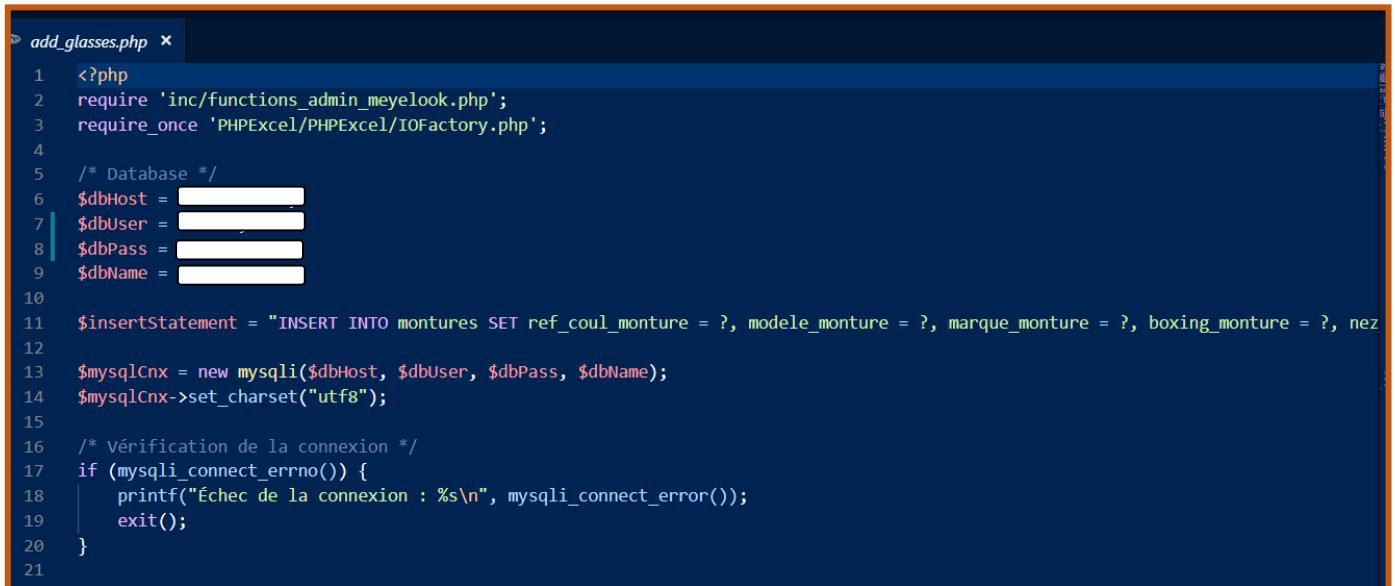
### **3 : LA CONNEXION A LA BASE DE DONNEES**

Au fil de l'exploitation des fichiers présents dans le dossier, j'ai pu constater que la connexion à la base de données était nécessaire à de nombreux fichiers afin de traiter les demandes des utilisateurs. Cela fonctionne sans problème pour le moment. Mais si l'on modifie le mot de passe ou l'identifiant, alors cette modification devra être faite aussi dans **TOUS** les fichiers qui inclus ces lignes. Un travail fastidieux !!! Et qui peut / doit être évité.

Pour cela, je dois cibler toutes les occurrences à cette demande de connexion où l'on saisit les identifiants et les remplacer par des variables qui stockeront les identifiants de connexion.

Ainsi, lorsque l'on modifiera les codes d'accès à la base de données dans PHPMyAdmin par exemple, il suffira de les enregistrer dans ce fichier qui sera inclus dans tous les fichiers qui nécessiteront la connexion. De cette manière, le code est opérationnel et il devient plus facile à gérer du côté de la connexion à la base de données.

*Des illustrations Avant-Après :*



```
add_glasses.php *  
1 <?php  
2 require 'inc/functions_admin_meyelook.php';  
3 require_once 'PHPExcel/PHPExcel/IOFactory.php';  
4  
5 /* Database */  
6 $dbHost = [REDACTED];  
7 $dbUser = [REDACTED];  
8 $dbPass = [REDACTED];  
9 $dbName = [REDACTED];  
10  
11 $insertStatement = "INSERT INTO montures SET ref_coul_monture = ?, modele_monture = ?, marque_monture = ?, boxing_monture = ?, nez  
12  
13 $mysqlCnx = new mysqli($dbHost, $dbUser, $dbPass, $dbName);  
14 $mysqlCnx->set_charset("utf8");  
15  
16 /* Vérification de la connexion */  
17 if (mysqli_connect_errno()) {  
18     printf("Échec de la connexion : %s\n", mysqli_connect_error());  
19     exit();  
20 }  
21
```

On retrouve ces lignes dans les fichiers :

- « admin/add\_glasses.php »,
- « admin/supprimer\_monture.php »,
- « admin/extract\_stock.php »,
- « controlConnect.php »,
- « admin/extract\_qualify.php »,
- « admin/inc/server\_processing\_monture.php »,
- Et « admin/monture.php ».

J'ai donc créé un fichier « dans un dossier « Model » où je précise les lignes et j'inclus le fichier dans tous les autres.

Je crée le fichier « Model/ Connections/meyelook\_db.php ». Celui-ci contiendra les lignes suivantes.

```

// SQL server connection information
$sql_details = array(
    'user' => $dbUser,
    'pass' => $dbPass,
    'db'   => $dbName,
    'host' => $dbHost
);

/* Database les paramètres */
$dbHost = [REDACTED];
$dbUser = [REDACTED];
$dbPass = [REDACTED];
$dbName = [REDACTED];

$mysqlCnx = new mysqli($dbHost, $dbUser, $dbPass, $dbName);
$mysqlCnx->set_charset("utf8");

/* Vérification de la connexion */
if (mysqli_connect_error()) {
    printf("Échec de la connexion : %s\n", mysqli_connect_error());
    exit();
}
?>

```

The diagram illustrates the flow of variables from the \$sql\_details array to the \$mysqlCnx object. Arrows point from each key-value pair in the array to its corresponding variable (\$dbHost, \$dbUser, \$dbPass, \$dbName) in the code block. A large bracket groups these four variables, which are then used to create a new mysqli object.

Je profite de l'opportunité pour enregistrer mes identifiants dans des variables pour une utilisation ultérieure. Ainsi dans d'autres fichiers qui incluront « Model/Connections/meyelook\_db.php », l'utilisation de ces variables sera tout à fait possible et même plus simple et nous évitera des erreurs.

J'inclus le fichier créé dans tous les fichiers qui nécessiteront une connexion en base de données.

Cela donnera finalement, par exemple, dans le fichier « admin/catalogue.php » :

```

<?php
session_start();
if (!$_SESSION['loggedIn']){
    // If not logged in
    header('Location : ../form.php');
}
require 'inc/functions_admin_meyelook.php';
require_once 'PHPExcel/PHPExcel/IOFactory.php';
require '../Model/Connections/meyelook_db.php';

```

A red arrow points to the line 'require '../Model/Connections/meyelook\_db.php';' in the code editor, indicating where the database connection file is included.

# VI- SPÉCIFICATIONS DIVERSES

## 1 : ENVIRONNEMENT DE TESTS

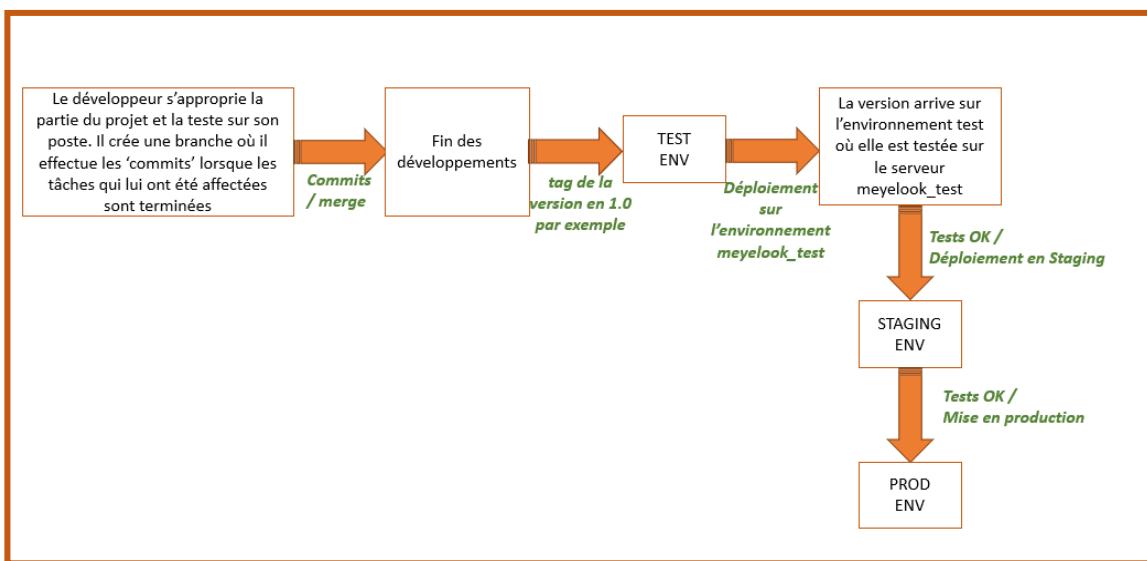
Après de nombreuses erreurs et essais, les modifications sont enregistrées dans le dossier en localhost. Lorsque je jugeais mon travail suffisamment avancé, avec la validation de mon tuteur, j'effectuais un enregistrement dans la branche GIT qui m'était assignée.

```
Jamila.djedoui@LAP-SOFT-36 MINGW64 /c/wamp64/www/stage_Jamila/repository/meyelookserver (jamila)
$ git add *

Jamila.djedoui@LAP-SOFT-36 MINGW64 /c/wamp64/www/stage_Jamila/repository/meyelookserver (jamila)
$ git commit -m "ajout modal en suppression et fin de stage"
[jamila fd38703] ajout modal en suppression et fin de stage
18 files changed, 441 insertions(+), 104 deletions(-)
create mode 100644 admin/add_glasses_copie.php
create mode 100644 admin/functions_styles.php
create mode 100644 admin/getMontureByID.php
delete mode 100644 admin/supprimer_monture.php
create mode 100644 admin/upload/ajoutmonturesfactices.xlsx

Jamila.djedoui@LAP-SOFT-36 MINGW64 /c/wamp64/www/stage_Jamila/repository/meyelookserver (jamila)
```

La chronologie du travail du développeur ressemble au schéma ci-après :



L'application sera « mergée » sur release\_deploy avec la version Alpha 1.0 que j'ai démarré et sera déployée en environnement de test une fois que la base de données sera entièrement remaniée. Cela est prévu pour fin 2019.

## ***2 : DIFFICULTES RENCONTREES***

Il est de coutume, en développement, d'être soumis à des tests avant la validation des modifications. Pour ce qui est des divers tests effectués, je citerai particulièrement l'apparition de la modal. Comme cette action n'était pas possible avant mon intervention j'ai commencé par proposer une version qui englobait une requête SQL afin de supprimer la monture. Cette dernière était supprimée directement sans confirmation demandée à l'utilisateur.

Mon tuteur m'a donc demandé d'intégrer une demande de confirmation et d'annulation car si l'on voulait supprimer une monture cela ne se ferait que si l'utilisateur le décide vraiment et que ce ne puisse pas être une erreur de frappe. Pour la suppression, j'ai opté pour l'apparition de la modal avec un bouton « Confirmer la suppression ».



Ma première proposition fonctionnait mais elle n'était finalement pas suffisante et je devais faire des modifications. En effet, il est préférable d'avoir un bouton « Annuler » pour le cas où l'action devait être avortée. J'ai donc proposé une modal proposant les deux « CONFIRMER » et « ANNULER ». L'annulation ne fait que recharger la page active.

En laissant l'opportunité à l'utilisateur de faire son choix et cela n'étant effectif qu'au clic de la souris, il devra être sûr de sa sélection car la suppression est irréversible et définitive.

Souvent les premières solutions proposées ne sont que des prototypes et il faut les retoucher et les remanier pour arriver à cerner le but final du client et lui proposer enfin une version qui lui conviendra.

Finalement, nous opterons pour celle-ci, plus complète :



**WHATEVER  
THE PROBLEM  
BE PART  
OF THE SOLUTION**

## **CONCLUSION**

Pour tout ce qui est du domaine technique, il m'a été confié l'*analyse* d'un produit existant et fonctionnel qu'il faut repenser et retravailler. Cette partie a été pour moi la plus longue et la plus délicate étant donné que je devais m'approprier le projet et bien le comprendre pour intervenir.

Il a été évident que la gestion de la documentation pour l'équipe fut importante et ce pour permettre d'expliquer les actions que j'ai entreprises et dans quels buts je les ai accomplies, afin que le rafraîchissement de l'application se poursuive après mon départ.

En définitive, ce stage a été la chance pour moi de plonger dans le vif du sujet pour m'immerger au maximum dans l'entreprise. Je reste quelque peu étrangère aux projets avec deadlines cependant, grâce aux réunions d'équipe où j'ai été conviée, j'ai pu entrevoir les bienfaits de la méthode *Agile*.

Il reste beaucoup à faire sur le projet, notamment en base de données. J'ai eu beaucoup de plaisir à accomplir ma mission et à réaliser toutes les tâches qui m'ont été fixées.

Ce fut une étape cruciale, une immersion dans mon futur métier. J'y ai appris également beaucoup sur le travail en équipe, que l'on avance ensemble et chacun apporte par ses actions et par son savoir. C'est un échange perpétuel entre partenaires vers la réalisation du projet et ses améliorations.

Je ressors de cette mission enrichie et sereine. J'y ai gagné un peu plus de confiance en moi.

Je sais à présent mettre en pratique mes capacités, mes compétences. Je peux mieux utiliser mes savoirs et échanger avec mes collègues.

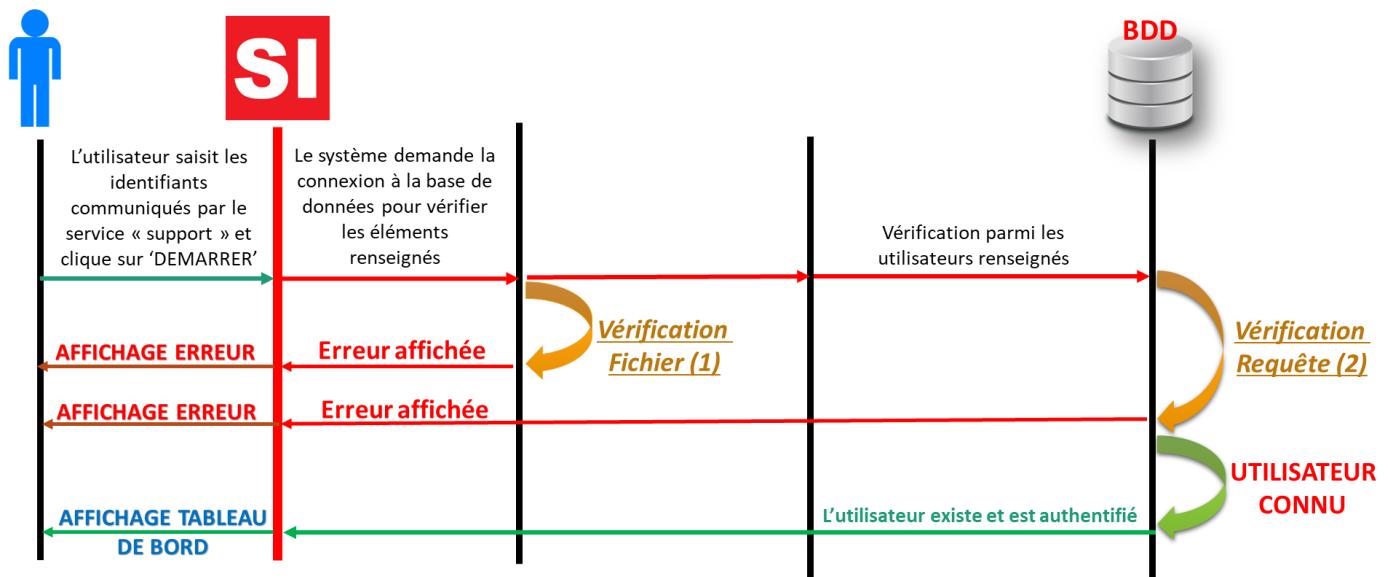
Mon projet est de continuer vers une seconde formation à niveau plus élevé pour parfaire mes compétences et être plus efficace avec un contrat de professionnalisation de *Conceptrice Développeuse Informatique* si l'avenir me le permet.

## ANNEXES

### Diagramme de Séquence "Connexion de l'utilisateur"

L'administrateur clique sur le bouton « DEMARRER » : le système vérifie les informations saisies dans le formulaire de connexion. Si la connexion est impossible ou si ceux-ci sont incorrects, un message d'erreur s'affiche soit en précisant l'erreur de connexion soit en invitant l'utilisateur à renouveler sa saisie.

Lorsque la connexion est établie alors le tableau de bord s'affiche.



Vérification (1): 1-1: Les champs ne sont pas renseignés  
                   1-2: Connexion impossible  
                   1-3: Les champs sont erronés

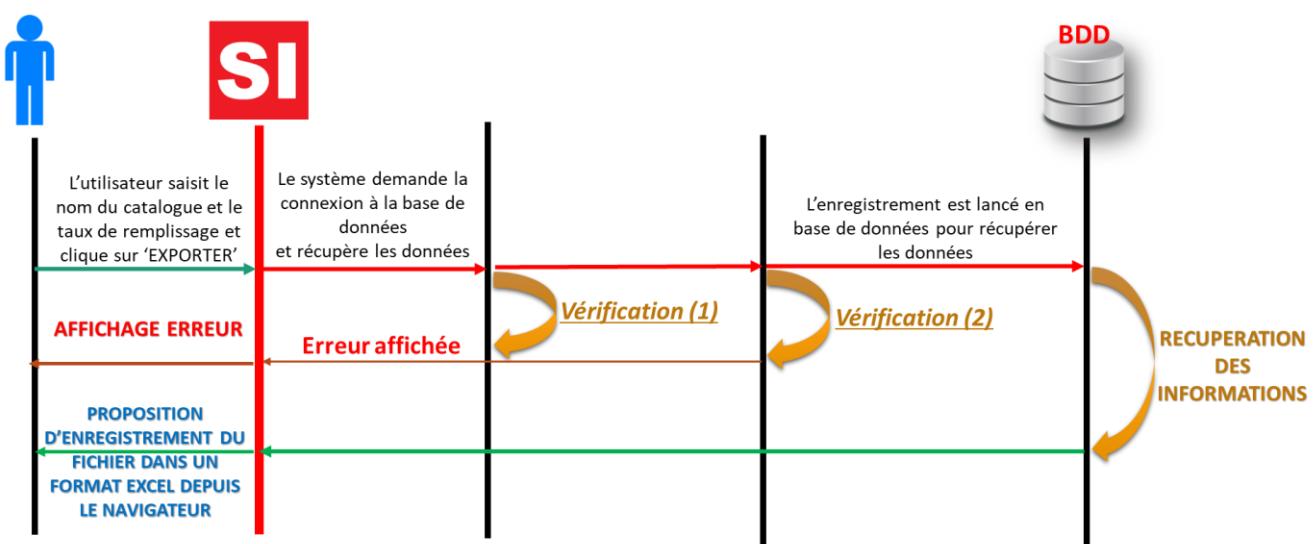
Vérification (2): 2-1: L'utilisateur n'a pas de droit et ne peut être autorisé à se connecter

### Diagramme de Séquence "Exporter un catalogue"

Cette fois, c'est la seconde partie de la fenêtre qui nous intéresse. L'utilisateur choisit dans la liste déroulante le catalogue qu'il veut exporter. Le deuxième onglet noté « A faire » reste pour le moment non terminé (elle permettra de cibler les montures des catalogues, celle dont les qualifications sont complètes ou renseignées à 30%, à 60%...). Je ne suis pas amenée à gérer cela pour l'instant, elle fera partie des tâches optionnelles.

The screenshot shows a user interface with two main sections. On the left, there is a section titled 'AJOUTER UN CATALOGUE' containing a 'Choose File' button with 'No file chosen' and an 'Ajouter' button. On the right, there is a section titled 'EXPORTER UN CATALOGUE' containing dropdown menus for 'Tous' and 'A faire' and a 'Exporter' button. The 'EXPORTER UN CATALOGUE' section is highlighted with a thick orange border.

Lorsque l'on clique sur *Exporter*, les montures du catalogue choisi sont récupérées et stockées dans un fichier Excel que l'on pourra enregistrer par la suite pour utilisation ultérieur.



**Vérification Fichier (1):** 1-1: Type de langage non valide  
1-2: Connexion à la base de données impossible

**Vérification Requête (2):** 2-1: Erreur Requête

## Etapes de réalisation “ Connexion de l'utilisateur ”

### **Comment générer l'accès ?**

Je crée les fichiers **form.php**, **controlConnect.php** et **disconnect.php** pour pouvoir gérer les connexions à l'application.

- **form.php** générera une page contenant un formulaire de saisie pour pouvoir entrer ses informations,
- **controlConnect.php** est un fichier qui vérifie les informations saisies, ouvre la session et redirige l'utilisateur vers l'interface lorsqu'il est identifié.
- **disconnect.php** redirige l'utilisateur vers la page d'identification (accueil) et ferme la session.

### Les fichiers affectés par les modifications sont :

- « **Index.php** »,
- « **admin/index.php** » qui lance le Tableau de bord devient « **admin/indexTB.php** »,
- « **admin/inc/config.php** » pour remplacer **index.php** par **indexTB.php**
- « **admin/inc/views/base\_header.php** ».

Afin que l'utilisateur puisse ouvrir la page et être authentifié, j'ai également ajouté le démarrage d'une session et la vérification de connexion dans les fichiers :

- ✓ « **admin\add\_glasses.php** »,
- ✓ « **admin\catalogue.php** »,
- ✓ « **admin\extract\_qualify.php** »,
- ✓ « **admin\extract\_stock.php** »,
- ✓ « **admin\indexTB.php** »,
- ✓ « **admin\monture.php** »,
- ✓ « **admin\stocks.php** »,
- ✓ « **admin\supprimer\_monture.php** »
- ✓ « **controlConnect.php** ».

```
<?php  
session_start();  
if (!$_SESSION['loggedIn']){  
    // If not logged in  
    header('Location : ../form.php');  
}
```

```
// Code ( Accès autorisé )  
$_SESSION['user'] = $login;  
$_SESSION['loggedIn'] = true;  
header("Location: admin/indexTB.php");  
}
```

Dans le fichier **controlConnect.php**, je stocke dans la variable `$_SESSION['loggedIn']` la valeur ‘true’ (vrai). Ainsi, avant l'ouverture des fichiers affectés par l'accès à MeyeLook, si cette valeur n'est pas égale à ‘true’ alors nous sommes redirigés vers le formulaire où nous devrons saisir nos identifiants.

## Etapes de réalisation "Exporter un catalogue"

### Comment ça marche ?

Comme pour l'ajout de catalogue, on utilise aussi la librairie PHPExcel. On utilise '\t' (tabulation) pour représenter le changement de cellule

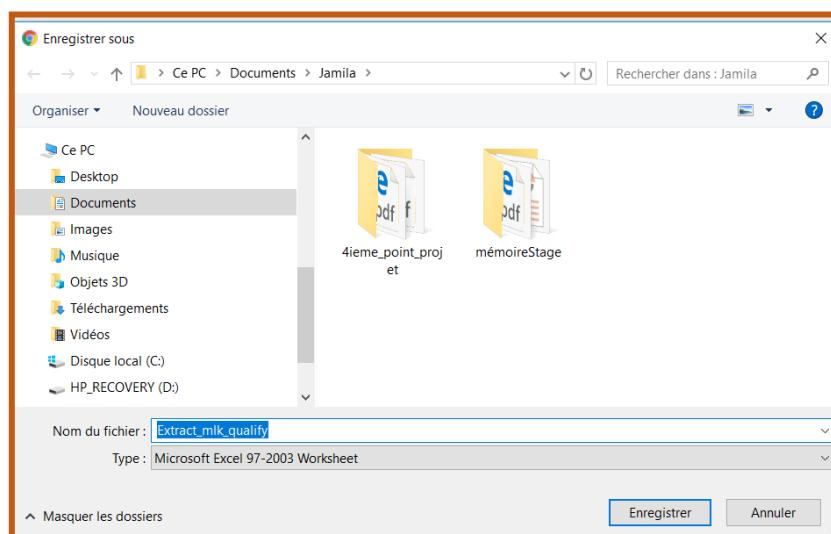
```
    $export = $bdd->query($select);

    $fields = $export->columnCount();
    $header = '';
    $data = '';

    for ( $i = 0; $i < $fields; $i++ ){
        $header .= $export->getColumnMeta($i)[ 'name' ] . "\t";
    }
    while( $row = $export->fetch() ){
        $line = '';
        for ( $i = 0; $i < $fields; $i++ ){
            if ( ( !isset( $row ) ) || ( $row == "" ) ){
                $value = "\t";
            }
            else {
                $value = str_replace( "'", "''" , $row[$i] );
                $value = "'" . $row[$i] . "'\t";
            }
            $line .= $value;
        }
        $data .= trim( $line ) . "\n";
    }
    $data = str_replace( "\r" , "" , $data );
    if ( $data == "" ){
        $data = "\n(0) Records Found!\n";
    }
    header("Content-type: application/octet-stream");
    header("Content-Disposition: attachment; filename=Extract_mlk_qualify.xls");
    header("Pragma: no-cache");
    header("Expires: 0");
    print "$header\n$data";           ///////////// Enregistrement à déterminer par l'utilisateur     ///////////
}

?>
```

Finalement, nous sommes invités à choisir l'endroit où nous enregistrons notre fichier Excel pour sa future utilisation.



Je peux renommer mon fichier comme je le souhaite et le sauvegarder dans le dossier de mon choix.

### Intégration du style des montures ajoutées à l'aide d'un fichier

En PHP, je récupère toutes les marques liées aux différents styles dans un tableau. Lorsque nous voudrons ajouter des marques définies par un style, nous pourrons le faire dans la base de données directement.

Requête SQL :

```
SELECT marques.name_marque FROM marques JOIN caracteriser ON marques.id_marque = caracteriser.id_marque WHERE id_style = ?;
```

Dans notre fichier « admin/add\_glasses.php », lorsque nous voulions récupérer les marques définies par style, nous avions précédemment :

```
if ($_FILES['fichier']['error'] == 0) {  
  
    $marque_fashion = array("BAILA", "ATTITUDEFASHION", "LESPERSONNALITES", "KARLLAGERFELD", "RAYBAN",  
        "GUCCI", "HUGOBOSS", "PAULANDJOE", "VOGUE", "ESPRIT", "CALVINKLEIN",  
        "BOSSORANGE", "JUSTCAVALLI", "ROCHAS", "SONIARYKIEL", "GUESS", "SAINTLAURENT",  
        "CHRISTIANLACROIX", "CERRUTI", "BENETTON", "CKCALVINKLEIN", "KENZO", "NINARICCI");  
  
    $marque_vintage = array("ZADIGETVOLTAIRE", "MEDLEY");  
  
    $marque_sport = array("RIPCURL", "LEVIS", "OXYDO", "CARRERA", "RAYBAN");  
  
    $marque_design = array("MODEINFRANCE", "PURE");
```

Les informations entrées en 'dur'

```
// Traitement du style par marque  
if(in_array($marque_monture, $marque_fashion)){  
    $style_monture = "2";  
}elseif(in_array($marque_monture, $marque_vintage)){  
    $style_monture = "3";  
}elseif(in_array($marque_monture, $marque_sport)){  
    $style_monture = "4";  
}elseif(in_array($marque_monture, $marque_design)){  
    $style_monture = "5";  
}else{  
    $style_monture = "1";  
}
```

Je n'ai qu'à reporter mes informations grâce aux fonctions que j'ai créé dans le fichier « admin/functions\_styles.php » inclus dans « admin/add\_glasses.php ».

```
if ($_FILES['fichier']['error'] == 0) {  
    $marque_classic = getBrandsOfStyle(getIDStyles('classique'));  
    $marque_fashion = getBrandsOfStyle(getIDStyles('fashion'));  
    $marque_vintage = getBrandsOfStyle(getIDStyles('vintage'));  
    $marque_sport = getBrandsOfStyle(getIDStyles('sport'));  
    $marque_design = getBrandsOfStyle(getIDStyles('design'));  
    $marque_urbain = getBrandsOfStyle(getIDStyles('urbain'));  
    $marque_none = getBrandsOfStyle(getIDStyles('none'));
```

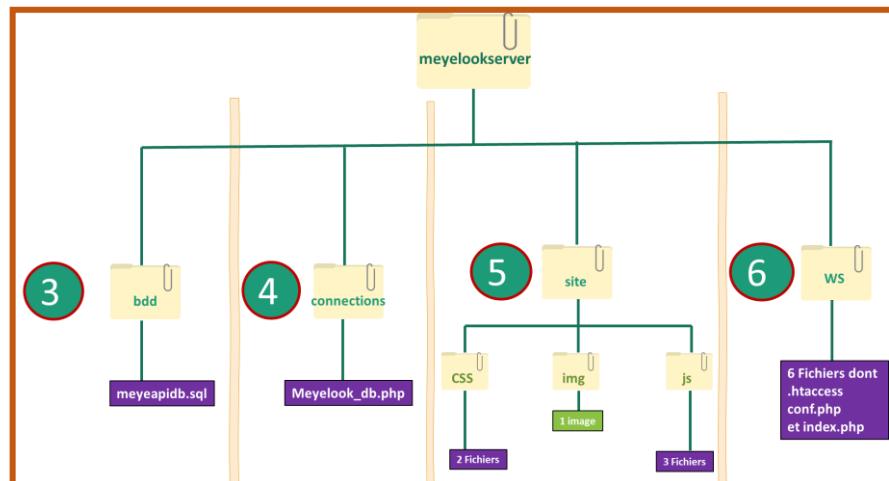
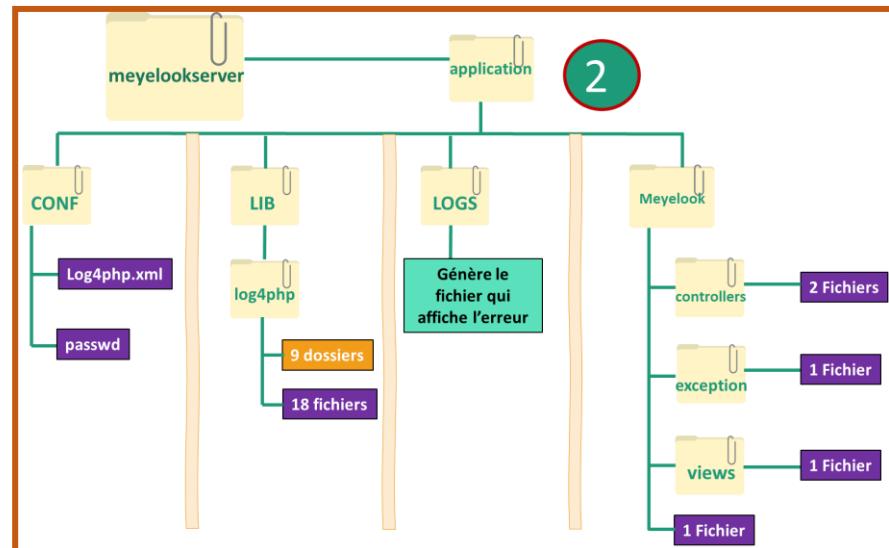
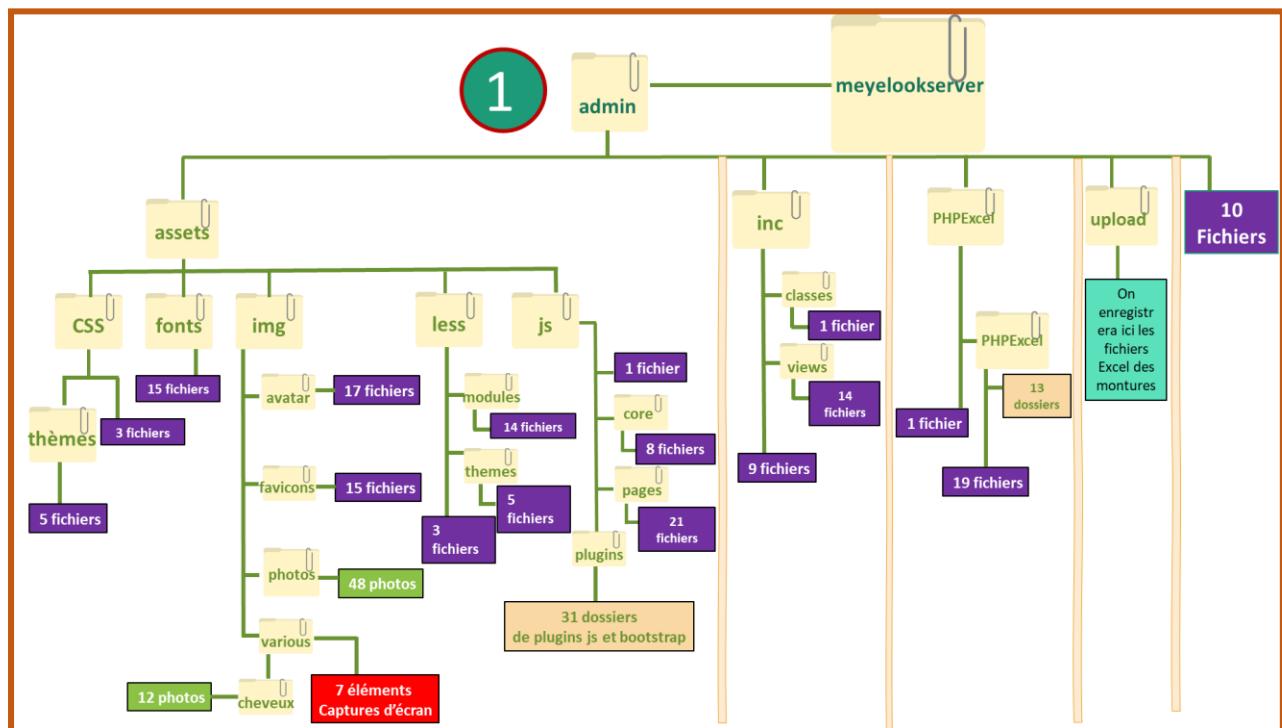
Une des fonctions utilisées dans le fichier « admin/functions\_styles.php » :

```
// Traitement du style par marque
if(in_array($marque_monture, $marque_classic)){
    $style_monture = getIDStyles('classique');
}elseif(in_array($marque_monture, $marque_fashion)){
    $style_monture = getIDStyles('fashion');
}elseif(in_array($marque_monture, $marque_vintage)){
    $style_monture = getIDStyles('vintage');
}elseif(in_array($marque_monture, $marque_sport)){
    $style_monture = getIDStyles('sport');
}elseif(in_array($marque_monture, $marque_design)){
    $style_monture = getIDStyles('design');
}elseif(in_array($marque_monture, $marque_urbain)){
    $style_monture = getIDStyles('urbain');
}elseif(in_array($marque_monture, $marque_none)){
    $style_monture = getIDStyles('none');
}
// Sur l'interface, il suffit de cliquer sur le style pour enregistrer la modification //
```

<?php

```
///////////je recupere les ID des styles avec le n
function getIDStyles($name_style){
    $styles = array();
    $sql_style = 'SELECT * FROM styles WHERE name_style = ?;';
    $bdd = connectBDD();
    $req_style = $bdd->prepare($sql_style);
    $req_style->execute(array($name_style));
    $req_style->setFetchMode(PDO::FETCH_OBJ);
    while ($resultat = $req_style->fetch()) {
        array_push($styles, $resultat);
    }
    echo ($styles[0]->id_style);
    return $styles[0]->id_style;
}
```

## L'architecture initiale plus en détail...



## Documentation

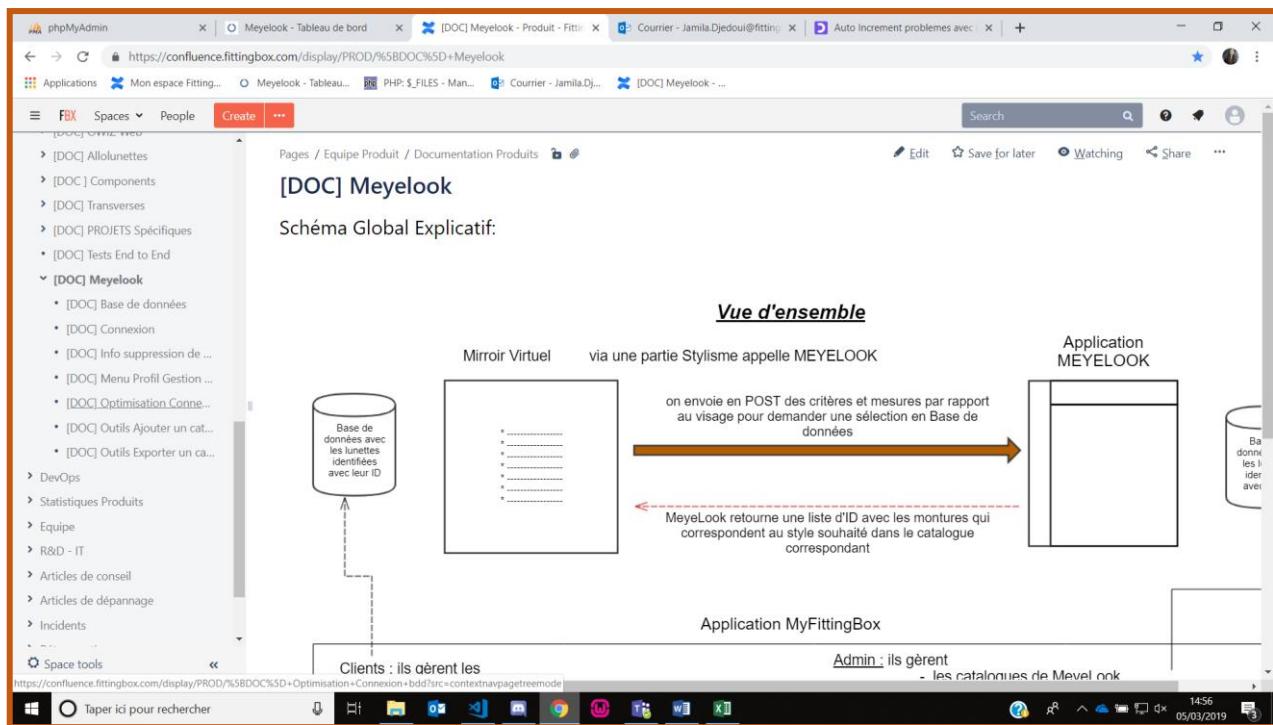
Etant donné que le projet sera repris après mon départ, il est important de documenter les missions et les tâches que j'ai menées. Ainsi, dans « CONFLUENCE », dans l'espace produits, une partie traitant des documentations y est dédiée. J'ai pu y créer le dossier « DOC MeyeLook » et y reporter tout mon travail.

The screenshot shows a Confluence page titled "Mon espace FittingBox". On the left, there is a sidebar with a "Produit" section highlighted. The main content area displays news items such as "La boîte à idées!" and "Fais jouer ton réseau!". Below the news, there is a section titled "Mes Tâches" which lists various tasks like "Articles de dépannage", "FAQ", and "How-to articles". The bottom of the screen shows a Windows taskbar with icons for various applications.

The screenshot shows a Confluence page titled "Equipe Produit". The sidebar has a "Produit" section. The main content area features a diagram titled "GAMME DE PRODUITS OWIZ" showing "ONLINE" and "RETAIL" categories with sub-products like "OWIZ Web Optiso", "OWIZ Plugin OANDO", "OWIZ Social", "OWIZ Apps OWIZ", "OWIZ Mirror OWIZ\*", and "OWIZ Street OWIZ\*". To the right, there is a table comparing "NOUVEAUX NOMS 2015" with "ANCIENS NOMS". The table includes entries like "OWIZ Web" (Site de l'Opticien (SDLO) / OPTISSO), "OWIZ Plugin" (FitWall / OANDOPLAY), and "VTO Frame" (eGlasses Photo / eGlasses Live / eGP / eGB...). The bottom of the screen shows a Windows taskbar.

Screenshot of a web browser showing a Confluence page titled "Documentation Produits". The page lists various documentation items under categories like "Archives", "Backends", "FitMix", etc. The interface includes a sidebar with navigation links, a search bar, and a toolbar with edit, save, and share options.

## [DOC] Meyelook: Schéma Global Explicatif



## Arborescence et Architecture

Pages / ... / [DOC] Meyelook

### [DOC] Arborescence et Architecture

Voici un essai d'arborescence de l'application telle que je la laisse aujourd'hui.

The screenshot shows a file tree for a project named 'meyelook'. The tree includes several sub-directories like 'Model', 'View', and 'WS', along with various PHP files and configuration files. Below the tree, a screenshot of a web browser displays a user interface with tabs for 'Dashboard', 'Connexion', 'Statistiques', and 'Profil'. A message 'Aucun résultat' is visible on the screen.

Like Be the first to like this

No labels

## Base de données

Pages / ... / [DOC] Meyelook

### [DOC] Base de données

Le fichier "add\_glasses.php" nous permet d'ajouter des montures ou quand celles-ci existent de les mettre à jour. Ce travail ne concerne finalement que ce fichier (lié à l'insertion et la mise à jour des montures grâce à un fichier Excel).

Aussi il y a dans ce fichiers des lignes en 'dur' pour entrer des marques dans des tableaux de styles.

En effet, pour chaque style, on choisit de stipuler ici les noms des marques... exemple

```
$marque_fashion = array("BAILA","ATTITUDEFASHION","LESPERSONNALITES","KARLLAGERFELD","RAYBAN",
"GUCCI","HUGOBOSS","PAULANDJOE","VOGUE","ESPRIT","CALVINKLEIN",
"BOSSORANGE","JUSTCAVALLI","ROCHAS","SONIARYKIEL","GUESS","SAINTLAURENT",
"CHRISTIANLACROIX","CERRUTI","BENETTON","CKCALVINKLEIN","KENZO","NINARICCI");
```

\$marque\_vintage = array("ZADIGETVOLTAIRE","MEDLEY");

\$marque\_sport = array("RIPCURL","LEVIS","OXYDO","CARRERA","RAYBAN");

\$marque\_design = array("MODEFRANCE","PURE");

Sachant qu'une marque peut correspondre à plusieurs styles, j'ai donc mené une réflexion autour d'un ajout en base de données afin de ne plus saisir ces informations directement dans le fichier mais en base de données. Il en ressort 3 tables la table 'marques', la table 'styles' et une table associative que j'ai appellé 'caractériser'.

Je n'ai qu'à reporter mes informations grâce aux fonctions que j'ai créé dans le fichier « admin/functions\_styles.php » (fichier inclus dans « admin/add\_glasses.php » naturellement). Et en générant les requête SQL le rendu est les même.

Like Be the first to like this

No labels

## Connexion

The screenshot shows a Confluence page titled "[DOC] Connexion". The page content discusses adding three pages to the "meyelookserver" directory and modifying the "index.php" file to include user authentication. It lists files modified under "Fichiers impactés:" and files added under "Fichiers ajoutés:". A note at the bottom states that users are redirected to a login form if they are not authenticated.

Comme l'on souhaite que l'accès à l'application soit sécurisé, j'ai ajouté trois pages dans le dossier meyelookserver et modifier le fichier index.php. Désormais, on n'accède plus au tableau de bord directement mais à une page de connexion en cliquant sur "m'identifier".  
Une fois les login et mot de passe authentifiés on accède au tableau de bord.

Fichiers impactés :

- index.php;
- admin/index.php; Tableau de bord ici devenu indexTB.php
- admin/inc/config.php;
- admin/inc/views/base\_header.php;

Fichiers ajoutés : (à la racine du dossier pour gérer l'authentification)

- form.php;
- controlConnect.php;
- disconnect.php; destruction de la session qui se fera au clic sur DECONNEXION

Afin que l'utilisateur puisse ouvrir la page et être authentifié j'ai également ajouté le démarrage d'une session et la vérification des identifiants dans les fichiers:

```
admin/add_glasses.php  
admin/catalogue.php  
admin/extract_qualify.php  
admin/extract_stock.php  
admin/indexTB.php  
admin/monture.php  
admin/stocks.php  
admin/supprimer_monture.php  
controlConnect.php
```

Ajout de la vérification de l'authentification avant d'accéder au tableau de bord dans les fichiers concernés par l'authentification cités plus haut. Si les utilisateurs ne sont pas authentifiés ils sont redirigés vers le formulaire de connexion où ils sont invités à saisir leur login et mot de passe (fournis par le support) mais dans le cadre de ce travail j'ai ajouté une table 'users' dans la base données et les requêtes sont liées à cette table dans le fichier controlConnect.php.

## Info suppression de Monture

The screenshot shows a Confluence page titled "[DOC] Info suppression de Monture". The page content describes the addition of a modal dialog for confirming the deletion of a mounting. It includes a note about centering the modal using Bootstrap CSS and a screenshot of the modal interface.

Activation de la suppression de la monture au clic. Apparition de la modal associée pour confirmation. Ajout du bouton "ANNULER" qui recharge simplement la page. Un peu de css via bootstrap pour centrer.

Fichier ajouté:

```
"admin/getMontureById.php";
```

Fichiers impactés:

```
"admin/inc/server_processing_monture.php";  
"admin/catalogue.php";
```

Like Be the first to like this

No labels

Write a comment...

Powered by Atlassian Confluence 6.8.1 · Report a bug · Atlassian News

ATLASSIAN

## Menu Gestion du profil utilisateur

**[DOC] Menu Gestion du profil utilisateur**

En fin de compte il n'y aura que le bouton avec le lien qui retourne à la page d'accueil du site et le bouton pour le masquage de la barre de menu du tableau de gauche.

Je n'oublie pas de modifier le contenu de la feuille de style associé afin de centrer et d'améliorer le visuel.

Le login est repris avec le mot 'Connecté' pour identifier l'utilisateur mais ceci est un ajout de ma part (peut être supprimer).

**Fichiers impactés :**

- admin/inc/views/base\_header.php;
- admin/assets/css/oneui.css;

**Fichiers ajoutés :**

- disconnect.php;

Tout d'abord, les animations ne peuvent s'exécuter sur un élément <span></span> donc je déplace l'animation vers le <div> qui est lié au login. Pour le message de bienvenue je rajoute un élément <div> dans lequel j'affecte l'animation du message et je garde mes éléments <span> à l'intérieur des div.

```

<header id="header_main" class="content-mini d-flex content-mini-full">
    <div class="flex-grow-1 flex-align-items-center">
        <div id="blmname" class="text-uppercase blmname"/>
        <div class="flex-grow-1 flex-align-items-center flex-justify-end">
            <div id="login" class="text-align-right">
                <div>php echo ($SESSION['user']).' Connecté';</div>
                <div class="dropdown">
                    <button class="btn-default btn-img text-center">
                        <img alt="Logout icon" data-action="disconnect" id="deconnection"/>
                    </button>
                </div>
                <div class="hidden-xs hidden-sm">
                    <button class="btn-default" data-toggle="layout" data-action="sidebar_mini_toggle" type="button">
                        <i class="fa fa-ellipsis-v"></i>
                    </button>
                </div>
            </div>
        </div>
    </div>
</header>

```

fichier admin/inc/views/base\_header.php

## Optimisation Connexion bdd

**[DOC] Optimisation Connexion bdd**

Dans chaque fichier on retrouvait les lignes correspondantes à la connexion à la base de données et si on change une info on doit le faire dans tous les fichiers donc on utilise un include pour regrouper toutes les infos concernant la connexion à la base de données. Ainsi les infos sont à modifier à un endroit et 1 seulement.

**Fichiers impactés :**

- admin/add\_glasses.php;
- admin/supprimer\_monture.php;
- admin/extract\_stock.php;
- controlConnect.php;
- admin/extract\_qualify.php;
- admin/inc/server\_processing\_monture.php;
- admin/monture.php;
- Connections/meyelook\_db.php

Ainsi dans le fichier « admin/inc/server\_processing\_monture.php », on inclut le fichier « Connections/meyelook\_db.php » et on peut utiliser les variables du fichier ce qui nous épargnera de les modifier aussi ici au besoin.

```

/*
Database les paramètres */
$dbhost = "localhost";
$dbuser = "root";
$dbpass = "";
$dbname = "meyelookdb_";

$SQL_details = array(
    'host' => $dbhost,
    'pass' => $dbpass,
    'db' => $dbname,
    'host' => $dbhost
);

$mysqlcnx = new mysqli($dbhost, $dbuser, $dbPass, $dbName);
$mysqlcnx->set_charset("utf8");

/* Vérification de la connexion */
if (mysqli_connect_error()) {
    printf("Echec de la connexion : %s\n", mysqli_connect_error());
    exit();
}

```

## Outils Ajouter un catalogue

The screenshot shows a Confluence page with the following details:

- Page Title:** [DOC] Outils Ajouter un catalogue
- Page Content:** En cliquant, on sélectionne le fichier en format Excel que l'on souhaite afin d'ajouter ou de mettre à jour des montures. On appelle le fichier "add\_glasses.php" qui fera la requête en base de données.  
Fichier impacté:  
admin/add\_glasses.php;  
Le fichier add\_glasses.php contient des fonctions et méthodes qui sont obsolètes depuis PHP 5.0 et qui ont été supprimées sur PHP 7.0 il a donc été nécessaire d'apporter des modifications pour permettre l'ajout des montures en base de données ou leur modification.
- Page Navigation:** Shows a sidebar with navigation links like [DOC] OWIZ Selfie, [DOC] OWIZ Street, [DOC] OWIZ Stylist, etc., and a main content area with a comment section and Atlassian branding.

## Outils Exporter un catalogue

The screenshot shows a Confluence page with the following details:

- Page Title:** [DOC] Outils Exporter un catalogue
- Page Content:** Comme pour l'ajout de catalogue, des modifications ont été nécessaires dans le fichier admin/extract\_stock.php. L'extraction se fait correctement et on peut enregistrer le fichier où on le souhaite pour réutilisation. Reste à faire: Les filtres ne sont pas encore gérés pour l'exportation car au choix de l'utilisateur de cibler les montures qu'il souhaite afin d'avoir une liste selon ses besoin. A voir et à terminer!
- Page Navigation:** Shows a sidebar with navigation links like [DOC] Components, [DOC] Transverses, [DOC] PROJETS Spécifiques, etc., and a main content area with a comment section and Atlassian branding.

# GLOSSAIRE

(Sources Wikipédia)

**ID** : En informatique, les identifiants (ID pour **IDentifier** en anglais) sont des marqueurs lexicaux qui nomment des entités. Les identifiants sont pratiquement toujours utilisés par les systèmes de traitement de l'information. Identifier des entités permet de s'y référer, ce qui est essentiel pour tout type de traitement symbolique. Souvent de type entier.

**REQUETE** : Un **langage de requête** est un langage informatique utilisé pour accéder aux données d'une base de données ou d'autres systèmes d'information. Il permet d'obtenir les données vérifiant certaines conditions (on parle de critères de sélection), comme toutes les personnes qui habitent une ville donnée. Les données peuvent être triées, elles peuvent également être regroupées suivant les valeurs d'une donnée particulière (par exemple on va regrouper toutes les personnes qui habitent la même rue).

**BASE DE DONNEES** : permet de stocker et de retrouver l'intégralité de données brutes ou d'informations en rapport avec un thème ou une activité ; celles-ci peuvent être de natures différentes et plus ou moins reliées entre elles.

**WEB SERVICE** : C'est un protocole d'interface informatique de la famille des technologies web permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués. Il s'agit donc d'un ensemble de fonctionnalités exposées sur internet ou sur un intranet, par et pour des applications ou machines, sans intervention humaine, de manière synchrone ou asynchrone. Le protocole de communication est défini dans le cadre de la norme SOAP dans la signature du service exposé (WSDL). Actuellement, le protocole de transport est essentiellement HTTP.

**INTERFACE (OU IHM)** : Les **interactions Homme-machines** (IHM) définissent les moyens et outils mis en œuvre afin qu'un humain puisse contrôler et communiquer avec une machine. Les ingénieurs en ce domaine étudient la façon dont les humains interagissent avec les ordinateurs ou entre eux à l'aide d'ordinateurs, ainsi que la façon de concevoir des systèmes qui soient ergonomiques, efficaces, faciles à utiliser ou plus généralement adaptés à leur contexte d'utilisation.

**QUALIFER** : Apporter une description. Dans ce contexte, ce sera le fait de décrire la monture par la marque, le modèle, le genre (homme, femme ou mixte), la forme, les dimensions...

**ARCHITECTURE** : En informatique, **architecture** désigne la structure générale inhérente à un système informatique, l'organisation des différents éléments du système (logiciels et/ou matériels et/ou humains et/ou informations) et des relations entre les éléments. Cette structure fait suite à un ensemble de décisions stratégiques prises durant la conception de tout ou partie du système informatique, par l'exercice d'une discipline technique et industrielle du secteur de l'informatique dénommée elle aussi **architecture**, et dont le responsable est l'architecte informatique.

**ARBORESCENCE** : En informatique, l'arborescence désigne généralement une organisation des données en mémoire, de manière logique et hiérarchisée utilisant une structure algorithmique d'arbre. Cette organisation rend plus efficace la consultation et la manipulation des données stockées. Cette structure part d'une racine (le niveau 1 de l'arborescence). La racine peut contenir autant de **répertoires** (les branches) que nécessaire, chaque répertoire pouvant lui-même contenir autant de **sous-répertoires** que nécessaire, et ainsi de suite.

**MODELE MVC :** C'est un motif d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.

**CLOUD :** Le **cloud computing**, en français l'**informatique en nuage** ou **nuagique** ou encore l'**infonuagique** (au Québec), consiste à exploiter la puissance de calcul ou de stockage de serveurs informatiques distants par l'intermédiaire d'un réseau, généralement Internet. Les serveurs sont loués à la demande, le plus souvent par tranche d'utilisation, selon des critères techniques (puissance, bande passante, etc.), mais, également, au forfait. Le **cloud computing** se caractérise par sa grande souplesse : selon le niveau de compétence de l'utilisateur client, il est possible de gérer soi-même son serveur ou de se contenter d'utiliser des applicatifs distants en mode SaaS.

**LOCALHOST :** Toute machine disposant d'une pile TCP/IP fonctionnelle permet de s'adresser à *localhost*, même si cette machine n'est reliée à aucun réseau physique ou virtuel.

Il faut bien entendu que soit démarré au préalable le serveur approprié (par exemple un serveur web ou un serveur de base de données) sur un port préalablement convenu (respectivement le port 80 associé par défaut aux requêtes HTTP et le port 3306 sur lequel écoute par défaut un serveur MySQL).

**WAMP OU LAMP :** C'est un acronyme informatique signifiant : « Windows », « Apache », « MySQL », « PHP » dans la majorité des cas mais aussi parfois, « Perl », ou « Python ».

Il s'agit d'un néologisme basé sur LAMP. Les rôles de ces quatre composants sont les suivants :

- Apache est le serveur web « frontal » : il est « devant » tous les autres et répond directement aux requêtes du client web (navigateur) ;
- Le langage de script PHP sert la logique ;
- MySQL stocke toutes les données de l'application ;
- Windows assure l'attribution des ressources à ces trois composants.

Tous les composants peuvent être situés :

sur une même machine ; sur deux machines, généralement Apache et le langage de script d'un côté et MySQL de l'autre ; sur de nombreuses machines pour assurer la haute disponibilité (répartition de charge et/ou failover).

Néanmoins, l'architecture WAMP est le plus souvent utilisée pour développer des sites web sur une machine Windows. De ce fait, en général, tout se passe sur une même machine. La mise en production se fera généralement sur une architecture LAMP (ou XAMP, X désignant un système à base d'Unix).

**VISUAL STUDIO CODE :** Visual Studio Code est présenté lors de la conférence des développeurs Build d'avril 2015 comme un éditeur de code cross-platform, open source et gratuit, supportant une dizaine de langages.

Le code source est fourni sous la licence libre MIT (plus précisément la licence Expat) sur le site du projet sur Github. En revanche, l'exécutable est proposé sur le site officiel de Microsoft sous une licence privative.

**GIT :** Logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes.

**REPOSITORY**: dépôt ou référentiel (de l'anglais *repository*) est un stockage centralisé et organisé de données. Ce peut être une ou plusieurs bases de données où les fichiers sont localisés en vue de leur distribution sur le réseau ou bien un endroit directement accessible aux utilisateurs.

**UML**: (*Langage de Modélisation Unifié*, de l'anglais *Unified Modeling Language*) est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet.

**PHP**: *HyperText Preprocessor*, plus connu sous son sigle **PHP** (acronyme récursif), est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

**EXTENSION**: une série de commandes écrites et compressées en un seul fichier : elle est une sorte de liste de commandes se retrouvant directement incluses au programme.

Il ne faut pas confondre ce terme avec *plugin* (« module d'extension ») qui, lui, désigne un programme s'ajoutant à un logiciel ; il ne faut pas non plus confondre avec l'expression « extension de nom de fichier » désignant le format d'un fichier.

**SUPPORT**: Les actions d'assistance consistent à garantir que les utilisateurs d'un système puissent continuer à profiter de la disponibilité de l'ensemble de ses composants pour l'accomplissement de leurs tâches. L'assistance peut porter sur les applications, ou sur les composants techniques (plateformes informatiques, réseaux).

**SERVEUR**: Un serveur fonctionne en permanence, répondant automatiquement à des requêtes provenant d'autres dispositifs informatiques (les clients), selon le principe dit client-serveur. Le format des requêtes et des résultats est normalisé, se conforme à des protocoles réseaux et chaque service peut être exploité par tout client qui met en œuvre le protocole propre à ce service.

**LIBRAIRIE** : ou **bibliothèque logicielle** est une collection de routines, qui peuvent être déjà compilées et prêtées à être utilisées par des programmes. Les bibliothèques sont enregistrées dans des fichiers semblables, voire identiques aux fichiers de programmes, sous la forme d'une collection de fichiers de code objet rassemblés accompagnée d'un index permettant de retrouver facilement chaque routine. Le mot « librairie » est souvent utilisé à tort pour désigner une bibliothèque logicielle. Il s'agit d'un anglicisme fautif dû à un faux-amis (*library*).

**USE CASE**: En génie logiciel et en ingénierie des systèmes, un **cas d'utilisation** définit une manière d'utiliser le système et permet d'en décrire les exigences fonctionnelles. D'après Bittner et Spence, « Un cas d'utilisation, défini simplement, permet de décrire une séquence d'événements qui, pris tous ensemble, définissent un système faisant quelque chose d'utile ». Chaque cas d'utilisation contient un ou plusieurs scénarios qui définissent comment le système devrait interagir avec les utilisateurs (appelés acteurs) pour atteindre un but ou une fonction spécifique d'un travail. Un acteur d'un cas d'utilisation peut être un humain ou un autre système externe à celui que l'on tente de définir.

**TICKET** : Ou demandes d'assistance en informatique, est une demande faite par l'utilisateur d'une application informatique, concernant un incident rencontré, une évolution souhaitée, ou une mauvaise compréhension des règles de gestion.

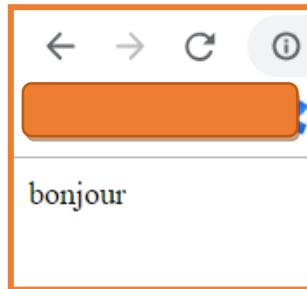
**Var\_dump ()**: `var_dump ()` affiche les informations structurées d'une variable, y compris son type et sa valeur. Les tableaux et les objets sont explorés récursivement, avec des indentations, pour mettre en valeur leur structure.

Ex : var\_dump(\$resultat) ;

```
C:\wamp64\www\stage_Jamila\repository\meyelookserver\admin\add_glasses.php:20
object(stdClass)[4]
  public 'id_monture' => string '1' (length=1)
  public 'photo_monture' => string 'M90284300001_PAULJOE_AZURE42-0-E108-49-17-135_2500x1400-face.jpg' (length=66)
  public 'photo_2_monture' => string 'M90284300001_PAULJOE_AZURE42-0-E108-49-17-135_2500x1400-gauche.jpg' (length=68)
  public 'modele_monture' => string 'AZURE42' (length=7)
  public 'marque_monture' => string 'Paul Joe' (length=8)
  public 'fournisseur_monture' => null
  public 'forme_monture' => string 'Papillon' (length=8)
  public 'teinte_monture' => string 'Chaudé' (length=6)
  public 'intensite_monture' => string 'Sombre' (length=6)
  public 'montage_monture' => string 'Cerclée' (length=8)
  public 'finition_mat_monture' => null
  public 'branche_monture' => null
  public 'taille_branches_monture' => string '135' (length=3)
  public 'tenon_monture' => string '2' (length=1)
  public 'position_tenon_monture' => string '3' (length=1)
  public 'position_pont_monture' => null
  public 'barre_monture' => null
  public 'plaquette_monture' => null
  public 'style_monture' => string '6' (length=1)
  public 'boving_monture' => string '49' (length=2)
  public 'largeur_verre_monture' => null
  public 'hauteur_verre_monture' => string '' (length=0)
  public 'verre_progressif_monture' => string 'Oui' (length=3)
  public 'nez_monture' => string '17' (length=2)
  public 'largeur_monture' => null
  public 'sexe_monture' => string '1' (length=1)
  public 'prix_monture' => string '0' (length=1)
  public 'matiere_monture' => string 'Plastique' (length=9)
  public 'poids_monture' => null
  public 'solidite_monture' => string 'Y' (length=1)
  public 'effet_monture' => null
  public 'couleur_monture' => null
  public 'color_monture' => null
  public 'date_monture' => string '2017-10-07' (length=10)
  public 'ref_coul_monture' => string 'E108' (length=4)
  public 'hex_couleur_monture' => string '#4a2e2b' (length=7)
  public 'rgb_couleur_monture' => string 'rgb(74, 46, 43)' (length=13)
  public 'h_couleur_monture' => string '6' (length=1)
  public 's_couleur_monture' => string '42' (length=2)
  public 'b_couleur_monture' => string '29' (length=2)
  public 'hex_couleur2_monture' => string '#B8353A' (length=7)
  public 'rgb_couleur2_monture' => string 'rgb(184, 53, 58)' (length=14)
  public 'h_couleur2_monture' => string '350' (length=3)
  public 's_couleur2_monture' => string '72' (length=2)
  public 'b_couleur2_monture' => string 'non' (length=3)
  public 'valid_monture' => string 'non' (length=3)
  public 'stock_monture' => string 'fittingbox' (length=10)
  public 'id_fb_monture' => string '455023' (length=6)
```

**Echo :** Commande UNIX (également présente avec MS-DOS) qui permet d'afficher une chaîne de caractères passée en paramètre sur le terminal (sortie standard). Cette commande est fréquemment utilisée dans les scripts shell et les programmes batchs pour indiquer textuellement un état du programme à l'écran ou dans un fichier.

Ex : echo 'bonjour' ; →



**IDE :** En programmation informatique, un **environnement de développement** est un ensemble d'outils qui permet d'augmenter la productivité des programmeurs qui développent des logiciels. Il comporte un **éditeur de texte** destiné à la programmation, des **fonctions** qui permettent, par pression sur un bouton, de démarrer le compilateur ou l'éditeur de liens ainsi qu'un **débogueur** en ligne, qui permet d'exécuter ligne par ligne le programme en cours de construction. Certains environnements sont dédiés à un langage de programmation en particulier.

## ACRONYMES

**HTML:** HyperText Markup Language

**PHP :** Personal Home Page de Hypertext Processor

**CSS :** Cascading Style Sheets

**WWW :** World Wide Web

**CMS:** Content Management System