DOSSIER DE PROJET

|  |  |
| --- | --- |
| *Développeur web et web mobile* | Stage effectué chez Altameos Multimédia 11/07/2022 au 16/09/2022  Jonathan OPIGEZ  *AFPA Evreux 2021/2022* |

Introduction :

Dans ce présent dossier, je vais vous présenter le projet qui m’a été confié pendant mon stage au sein de l’entreprise Altameos Multimédia.

Deux parties, bien distinctes, vont être abordées : un site web qui contient plusieurs pages statiques, un formulaire d'inscription et de connexion. Un back-office qui permet une gestion des membres, des prestations, des devis et des factures. Le tout avec un accès "superadmin" avec des droits spécifiques et un accès à une catégorie statistique.

Compétences couvertes par le projet :

Voici la liste des compétences classées par activité que j’ai pu acquérir lors de mon projet. Elles sont tirées du référentiel de certification du titre professionnel de développeur web et web mobile.

Activité type n°1 « Développer la partie frontend d’une application web ou web mobile en intégrant les recommandations de sécurité » :

- Maquetter une application,

- Réaliser une interface utilisateur web statique et adaptable,

- Développer une interface utilisateur web dynamique.

Activité type n°2 « Développer la partie backend d’une application web ou web mobile en intégrant les recommandations de sécurité » :

- Développer les composants d’accès aux données,

- Développer la partie backend d’une application web ou web mobile.

Résumé :

Ce stage a été réaliser à 90% en autonomie. Mon tuteur m’a laissé le choix du design et de manier les fonctionnalités à ma guise. La demande initiale était de développer un outil de gestion pour des devis/factures. Il devait offrir la possibilité de créer un devis pour les clients. Celui-ci se transformera en facture s’il est validé par l’entreprise. Plusieurs niveaux d’administration allant de 0 à 3 (0 étant le niveau superAdmin et 3 le niveau d’administration le plus faible). Cet outil a été réaliser principalement pour tester mes compétences. Le fait d’être en autonomie m’as obligé à structurer mon travail et à procéder étape par étape.

Cela m’a aussi permis de laisser parler ma créativité.

Le stage a été effectuer en télétravail.

Nous nous sommes très rapidement mis d’accord sur une totale souplesse au niveau horaires je me tenais à disposition pendant mes heures de travail.

Je faisais mon point le vendredi et le lundi matin avec mon tuteur pour échanger sur le projet. Je lui montrais mon avancement et , quelque fois, il me donnait plus de précision sur les méthodes sans me fournir la solution.

Le cahier des charges :

J’ai exposé plusieurs idées qui correspondaient aux diverses pistes/demandes que l’on avait évoquées. J’ai donc trouvé une direction à prendre.

Lors des 2 premières semaines je travaillais sur un design mais celui-ci ne me convenait pas. J’ai décidé de m’inspirer de Dashboard conçu par des professionnels du design pour les réadapter.

J’avais le choix d’utilisé des Framework j’ai donc utilisé Bootstrap ainsi que Jquery pour le front-end. Cependant pour la partie back-end, j’ai choisi de ne pas en utilisé. Afin de me former au maximum sur PHP/Mysql et aiguiser mes compétences.

L’outil devait répondre à plusieurs besoins :

- s’inscrire et se connecter

- créer des devis.

- système d’administration.

- gestion des membres et des administrateurs.

- niveaux d’administration.

- informations utiles lié à l’entreprise.

- stats de l’outil.

Sommaire

L’environnement de travail

Page 4-5

Partie front end

Généralités du projet

Page 6-7

Page connexion et inscription

Page 8-13

La barre de navigation

Page 13-15

Les infos utiles

Page 16

Gestion des membres

Page 17-20

Gestion devis/factures

Page 21-22

Partie backend

La base de données

Page 23-24

L’inscription

Page 25-27

Connexion

Page 28-29

Dashboard admin

Page 30-33

Dashboard client

Page 34-42

Recherche anglais/Améliorations

Page 43-46

Conclusion

Synthèse

Page 47

Conclusion

Page 48

L’environnement de travail :

Pour la partie backend, j’ai décidé de partir sur du PHP « from scratch », j’ai préféré ne pas utiliser Symfony car je ressentais le besoin de pratiquer PHP pure.



J’ai travaillé sur Visual studio code pour le développement du code.

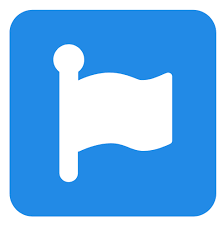
VS code possède un nombre important d’extensions qui facilite la vie.

HTML 5, CSS, JS et Version de PHP : 8.1.6



Avant de le déployer sur serveur, j’ai utilisé un serveur interne (XAMPP. PHP myadmin version 5.2)





Kit d’icônes : FONTAWESOME et Bootstrap



Libraire ‘frontend’ :







Création du MLD interface simple et efficace, possibilité de convertir (avec correction manuelle sur certains paramètres) en SQL.

Partie front end

Le projet était à construire de A à Z. De la modélisation à la réalisation.

Pour le design j’avais une totale liberté. Plusieurs essais ont été réalisés pour aboutir sur un design sobre et coloré. *Voir annexe 1*.

Pour le client :

Une interface claire et intuitive était indispensable, rien de pire que d’arriver sur une page et d’être submerger d’informations en tous genre. J’ai codé une barre de navigation, un menu simple sur le coter contenant les différentes pages accessibles. Les informations de la personne connecter se situe dans le haut de cette « <navbar> ». Dans le contenu de la page, j’ai créé un bouton « créer un devis » et une section pour afficher les informations/services de l’entreprise. *Voir annexe 2*.

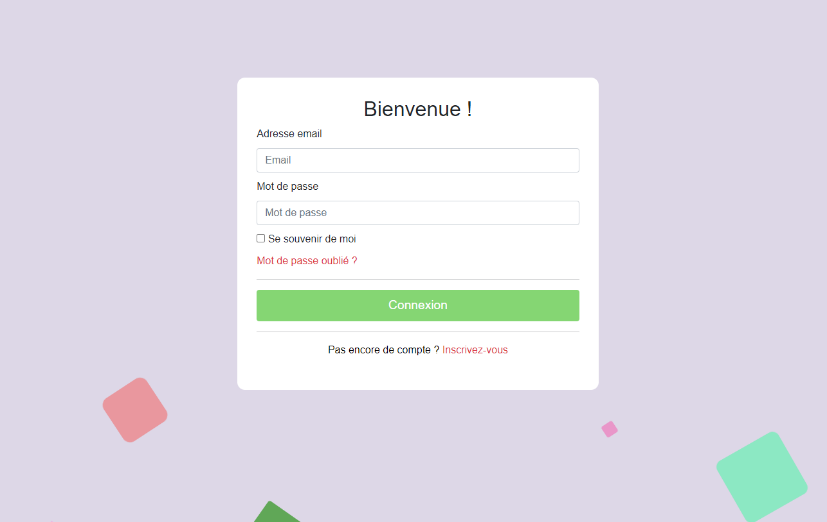
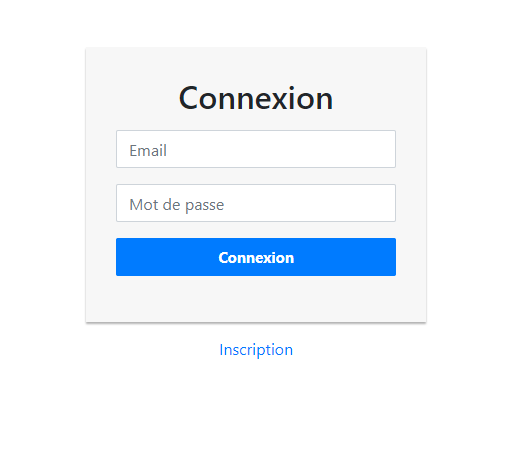
Pour les administrateurs :

Les administrateurs ont accès à un lien de plus dans le menu. Il s’agit du lien « gestion des membres ». Ils n’ont pas la possibilité de créer des devis. En revanche, ils peuvent ajouter une prestation selon leurs niveaux d’autorisation.

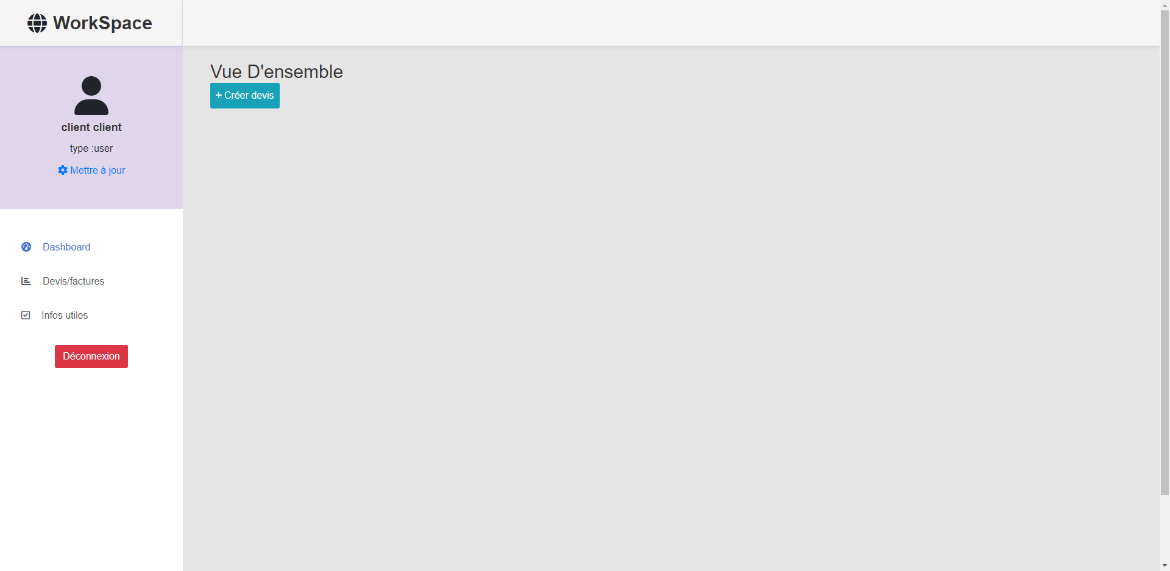
Pour le Superadmin :

Le superadmin a, lui, accès à toute une partie chiffres/stats. En plus, il a la possibilité de créer une prestation. Il peut aussi créer ou supprimer des administrateurs. *Voir annexe* 3

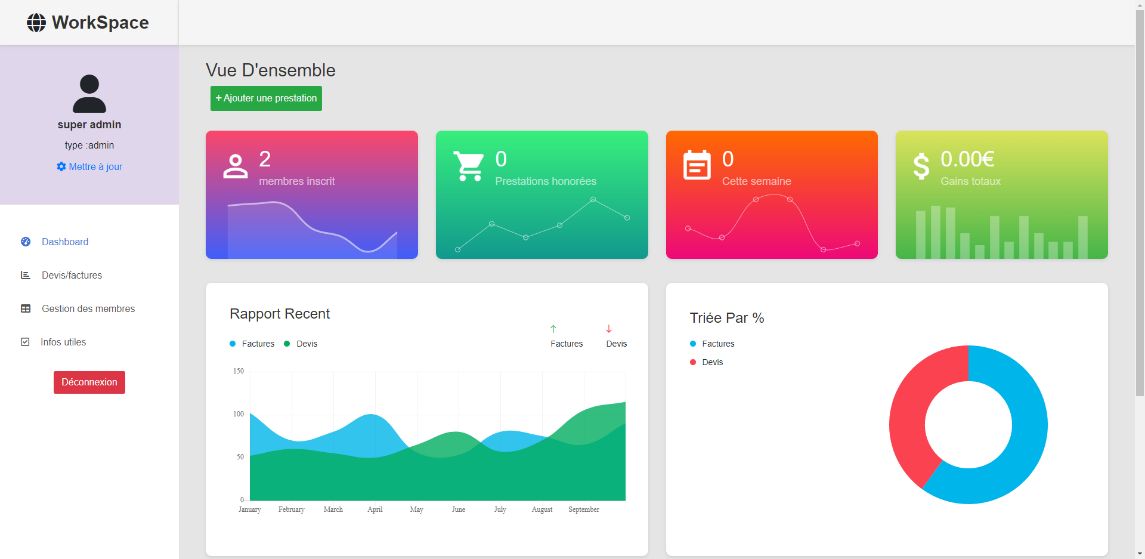
Annexe 1 :

Design 1 : Design 2 : 

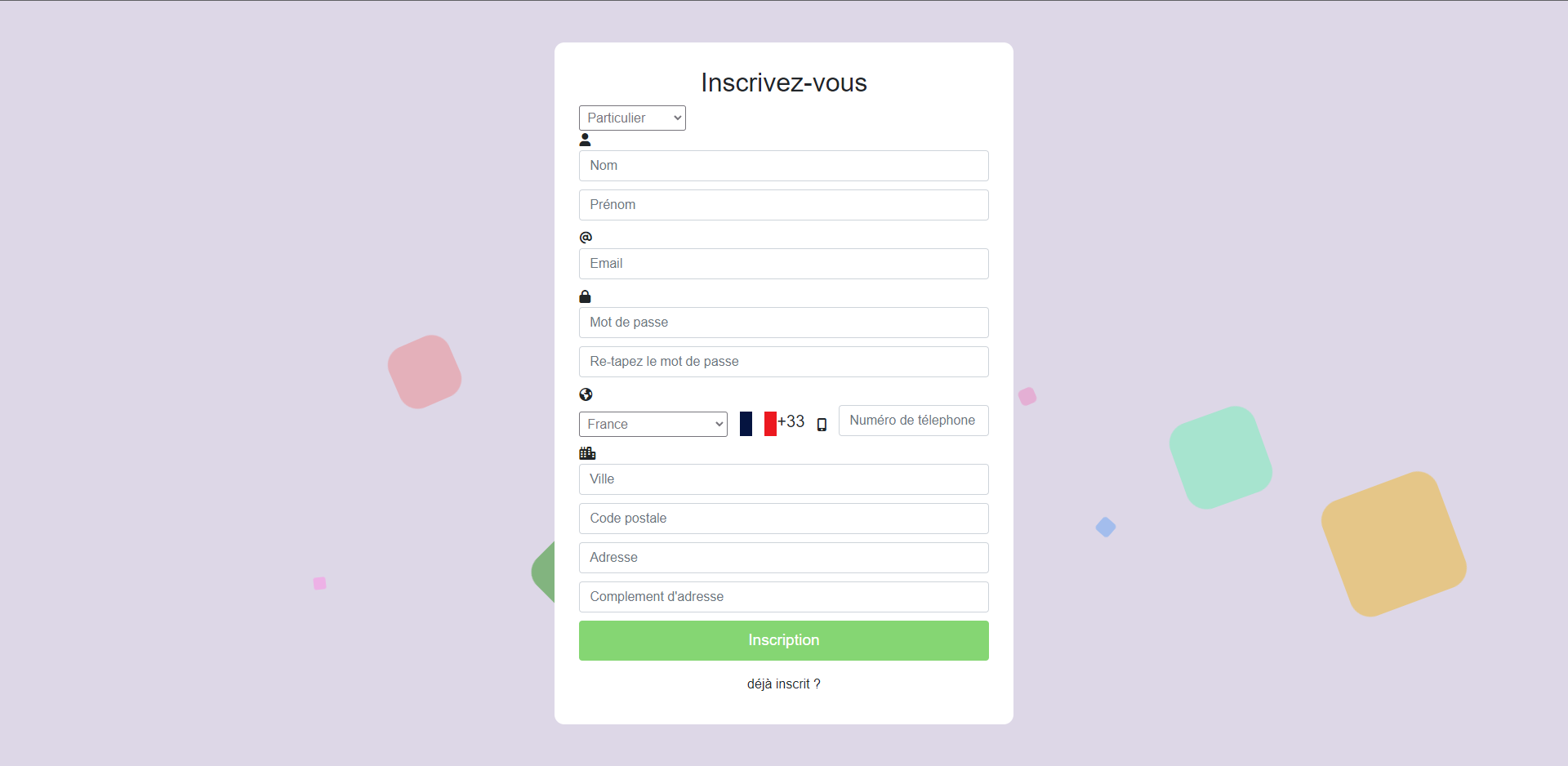
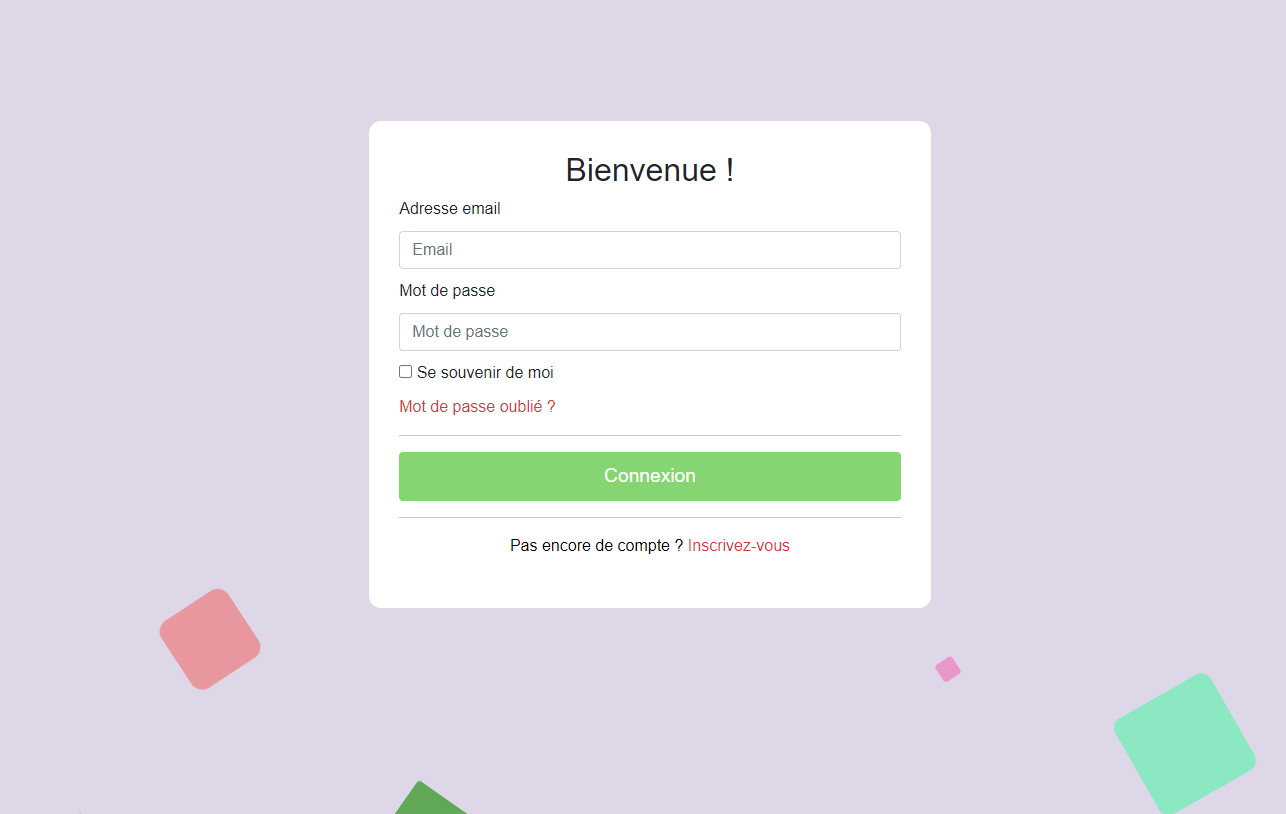
Annexe 2 :



Annexe 3 :



Pages connexion et inscription :



Ici les pages de connexion et d’inscription. Un formulaire html basique mis en forme à l’aide de Bootstrap (utilisation de différentes classes tel que « container » pour centrer les formulaires, les class « btn » pour le style des boutons ainsi que les couleurs bootstrap).

Fonctionnalité de la page inscription :

Mise place d’une fonction pour ajouter un champ selon le type de client. Si c’est un particulier qui s’inscrit, uniquement les champs nom et prénom seront affiché. *Voir annexe 1*

Si c’est un professionnel, un champ « Nom de l’entreprise » s’ajoute au formulaire.

Mise en place d’une autre fonction qui va récupérer le pays sélectionner et afficher le drapeau ainsi que le code de téléphone qui correspond au pays (exemple +33 pour la France) *Voir annexe 2*

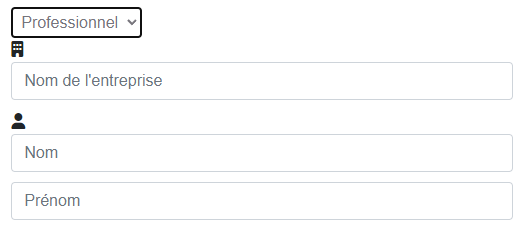
Chemin vers un dossier contenant tous les drapeaux en format SVG. Chaque drapeau est nommé avec le nom du pays en ALPHA2 (exemple : France = FR).

*Annexe 1 :*

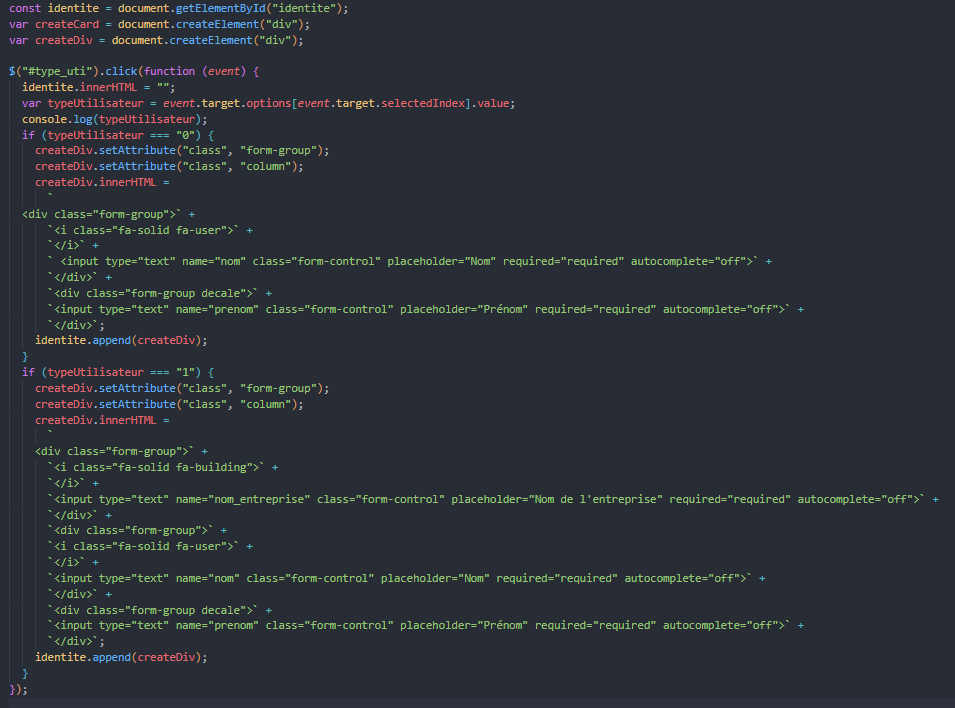
Formulaire particulier :



Formulaire professionnel :



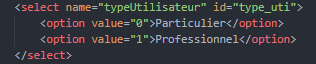
**Fonction pour choisir le type d’utilisateur :**



Fonction simple qui récupère la valeur de l’option choisi dans le select et affiche en conséquence les champs correspondants.

Si la valeur = 0 : Particulier

Si la valeur = 1 : Professionnel

**

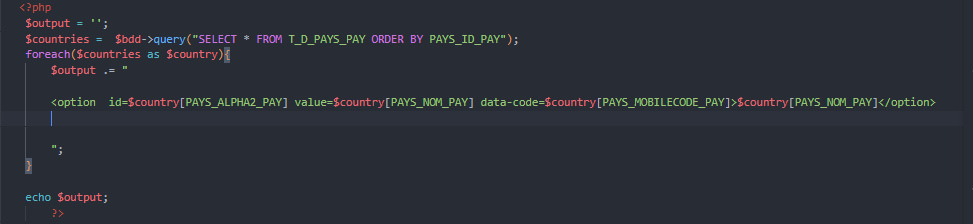
*Annexe 2 :*



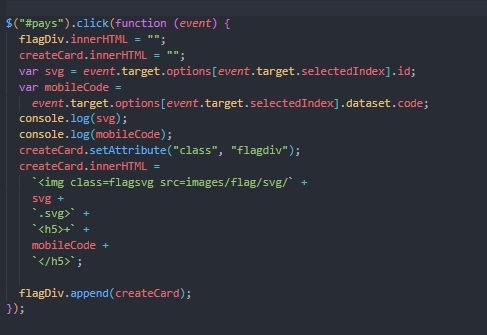
**Fonction qui va chercher le drapeau correspondant au pays et le code de téléphone :**

Cette fonction marche en parallèle avec une requête SQL. J’ai rajouté une table dans ma BDD qui récence tous les pays ainsi que leurs codes de téléphone. Grâce à une boucle PHP je peux attribuer à chaque option un id qui va être égale au nom du pays en ALPHA2 (France = FR) et un attribut data-code qui va correspondre au code de téléphone.

**La boucle PHP :**



A partir de là il ne me reste plus qu’as récupérer ces informations lors de la sélection et de manipuler le DOM en conséquence. Je déclare une variable SVG qui va prendre l’id du pays sélectionné (exemple « France » le chemin serait images/flag/svg/France.svg ) et ainsi pouvoir créer le chemin vers un dossier qui contiens tous les drapeau et afficher celui qui correspond.



Pareil pour le code du téléphone on va chercher l’attribut « data-code » et le stocker dans une variable « mobileCode » et appeler cette variable dans un <h5>.

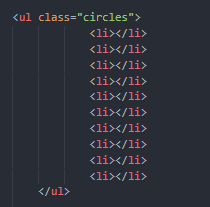
CSS Connexion/inscription

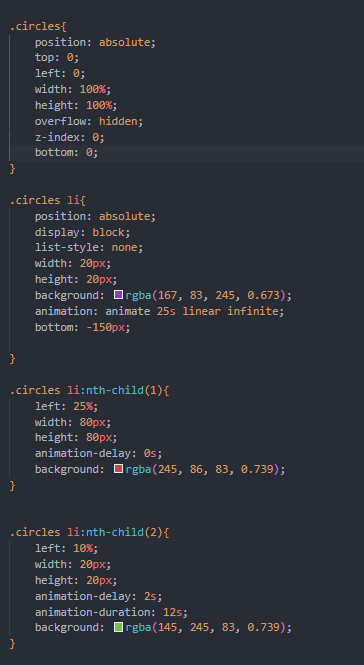
Le style de ces pages est principalement géré par bootstrap, les ajustements sont directement dans le header des pages concernées.

**Animation du background :**

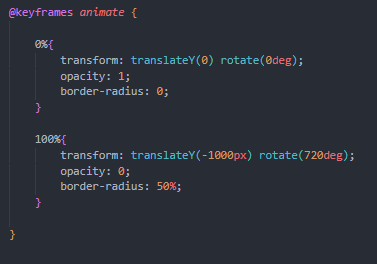
L’animation en arrière-plan est mise en place grâce à des balises <li> contenu dans une balise <ul> avec la class « circles » pour lui donner des propriétés css. Propriété définie dans un fichier css a part nommée cubes.css

**Balises HTML :** **Code CSS :**





**Animation keyframes :**



**Explication du code CSS :**

-Position « absolute » pour ne pas impacter les autres balises et pouvoir placer librement notre élément avec les propriétés (left, right etc…)

-Hauteur et largeur a 100% pour prendre toute la fenêtre.

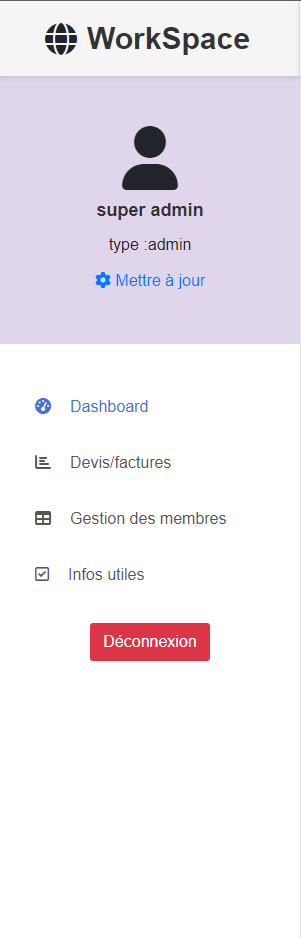
-Overflow hidden pour cacher la bar de scroll

-z-index a 0 pour pouvoir le placer en arrière-plan

Ensuite on se sert de la propriété :nth-child() pour sélectionné les enfant de la balise <ul> pour donner des propriété unique tel que des couleurs et tailles différentes.

On y ajoute aussi un effet de délais avec la propriété « animation-delay » pour créer cet effet de cascade.

La barre de navigation :



La barre de navigation est composée de plusieurs section :

* Une première qui contient le logo de l’entreprise
* Une seconde qui contient les informations de la personne connecter
* Une troisième qui contient les liens vers les différentes pages ainsi que le bouton déconnexion

Nous l’intégrons a toutes les pages grâce à un « include » :



**Le style :**

La majorité du style a été gérer par le thème de base ainsi que les class Bootstrap,

La barre de navigation a été volontairement positionné à gauche en static pour une meilleure navigation.

La mise en place d’une variable $page qui identifie la page sur laquelle l’utilisateur navigue actuellement et attribut au lien qui correspond dans la barre de navigation une class qui lui donne une couleur bleue pour aider l’utilisateur à se repérer lors de la navigation entre les différentes pages.

**Exemple pour la page Dashboard :**



**Mise en place des conditions en php :**



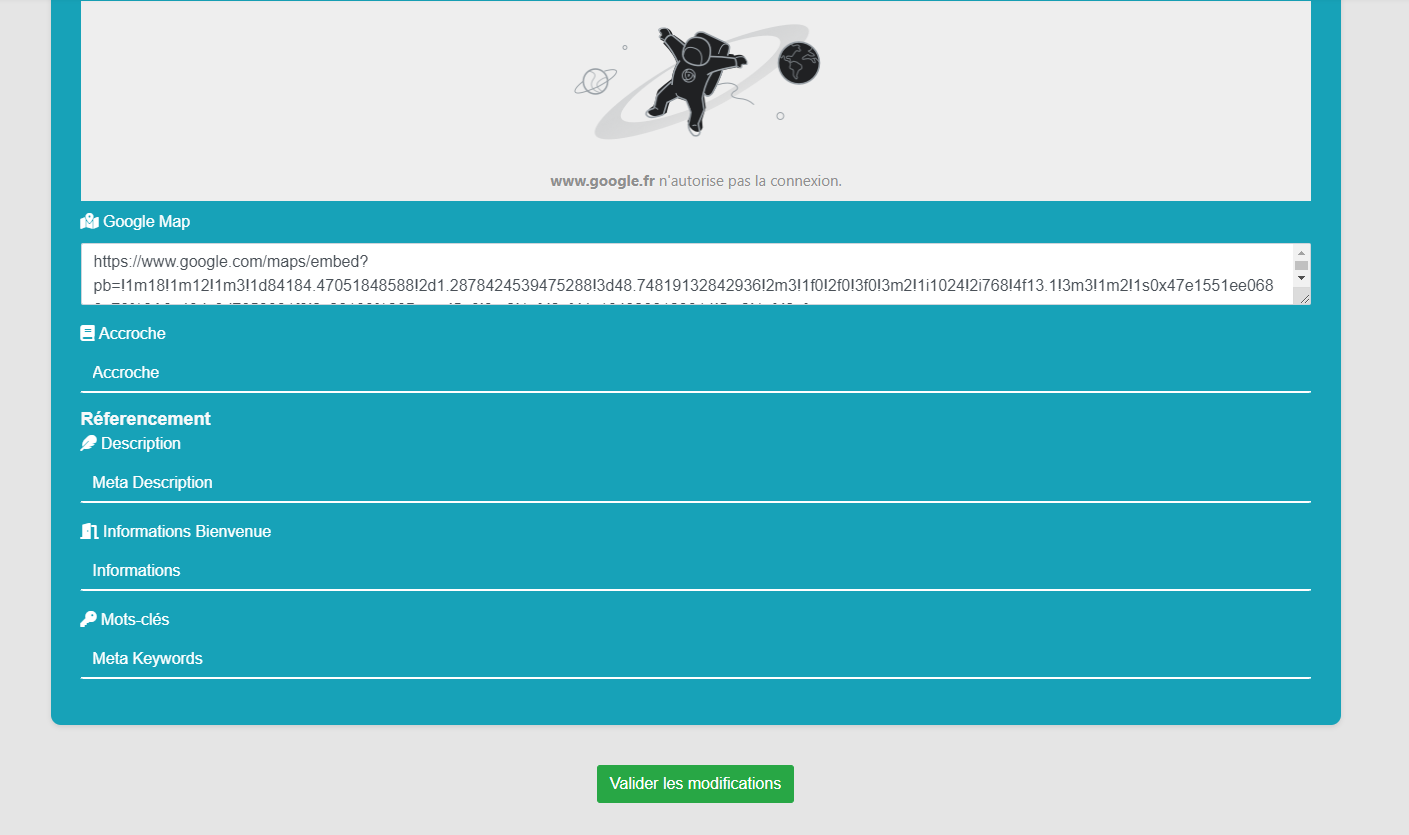
Les infos utiles :

La page infos utiles reprend les informations de l’entreprise.

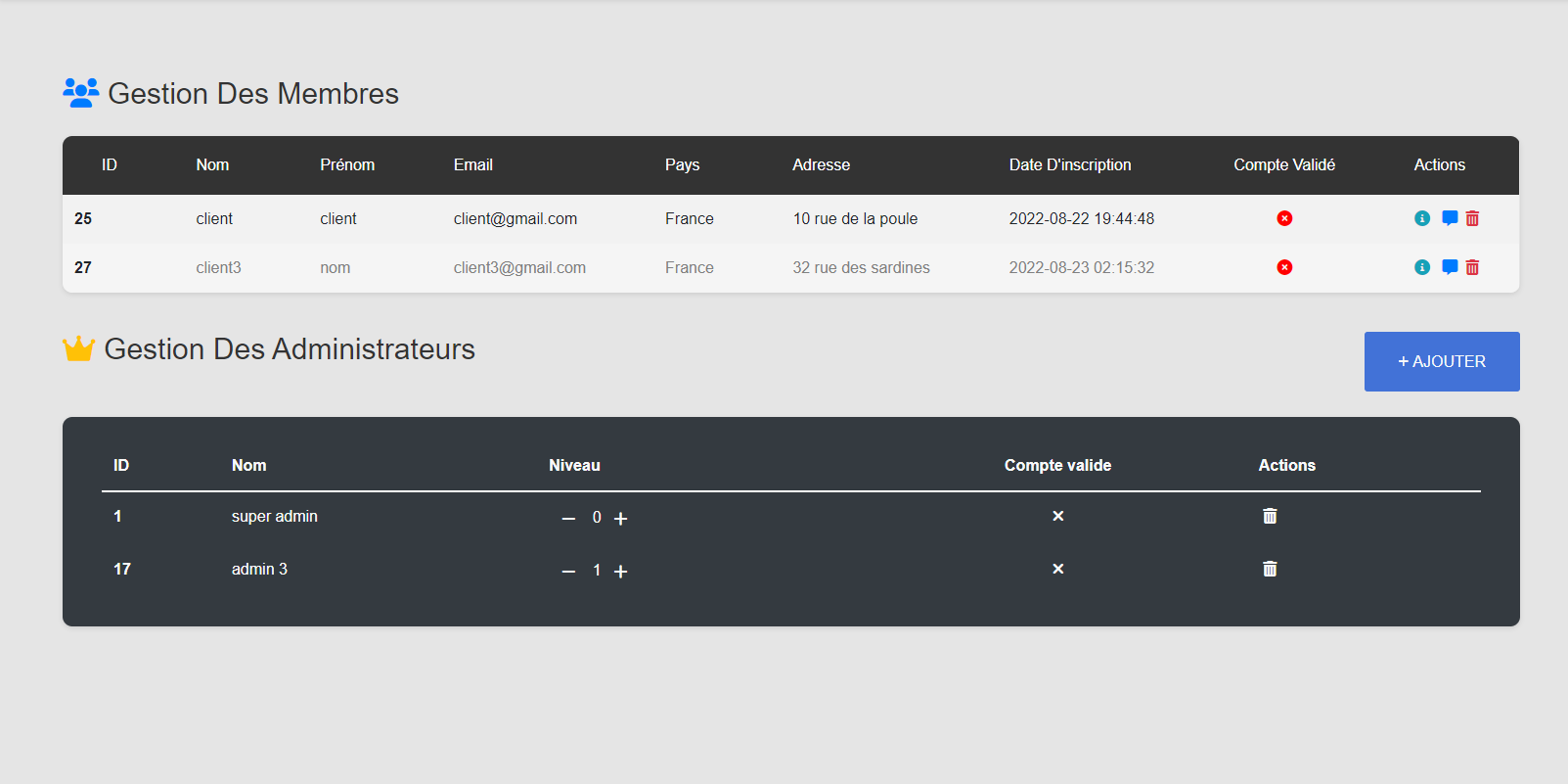
Les clients auront accès à l’information tel que le nom de l’entreprise, son adresse, son numéro de téléphone.

Mais aussi les réseaux sociaux, des informations de localisation etc…

L’administrateur lui pourras changer ses information et validé les changements aux cliques sur un bouton.



La gestion des membres :



Pour la gestion des membres j’ai décidé de représenter cela dans deux tableaux différents, l’un concerne les clients, l’autre les administrateurs.

**Les clients :**

Ils sont triés par ID dans le tableau nous avons accès à leurs information principal :

- l’identifiant - pays

- nom - adresse

- prénom - date d’inscription

- email - si le compte est valide ou non

**Les actions**:

- information (ouvre une modal avec toutes les informations du client) *voir annexe 1*

- pouvoir envoyer un message a ce client sur sa boite mail

- supprimé le client de la base de données *voir annexe 2*

**Les administrateurs :**

Ils sont triés de la même manière que les clients, cependant moins d’information sont nécessaire et d’autre apparaisse :

- l’identifiant - Niveau (allant de 0 à 3, étant le superAdmin et 3 le plus faible)

- nom - si le compte est validé

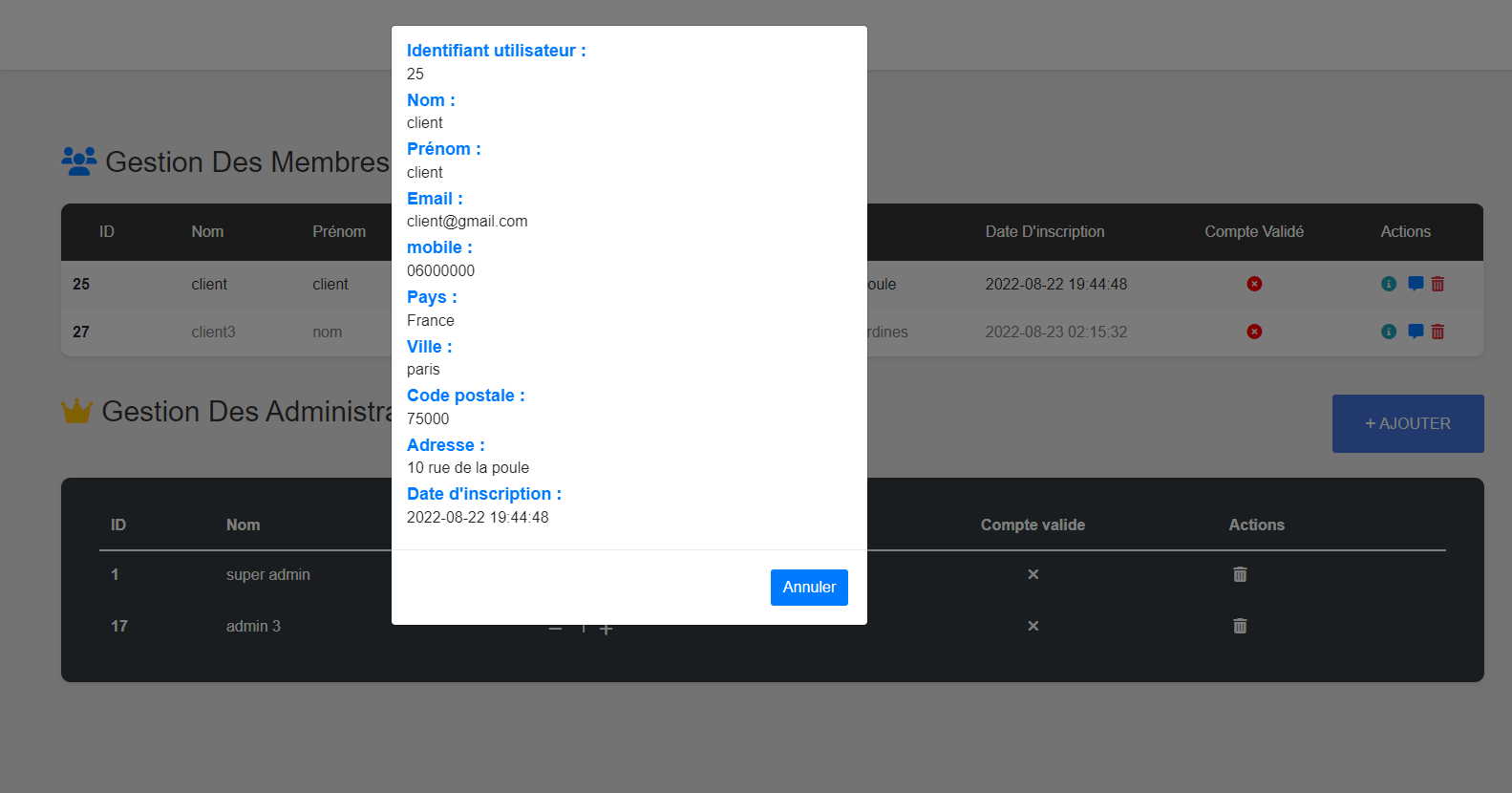
**Les actions**:

- pouvoir augmenter ou baisser le niveau de l’administrateur aux cliques sur les boutons plus ou moins dans le tableau

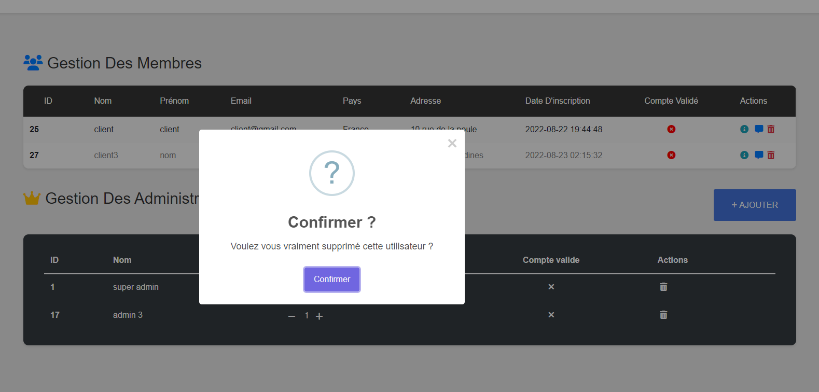
- supprimé l’administrateur de la base de données *voir annexe 3*

*Annexes :*

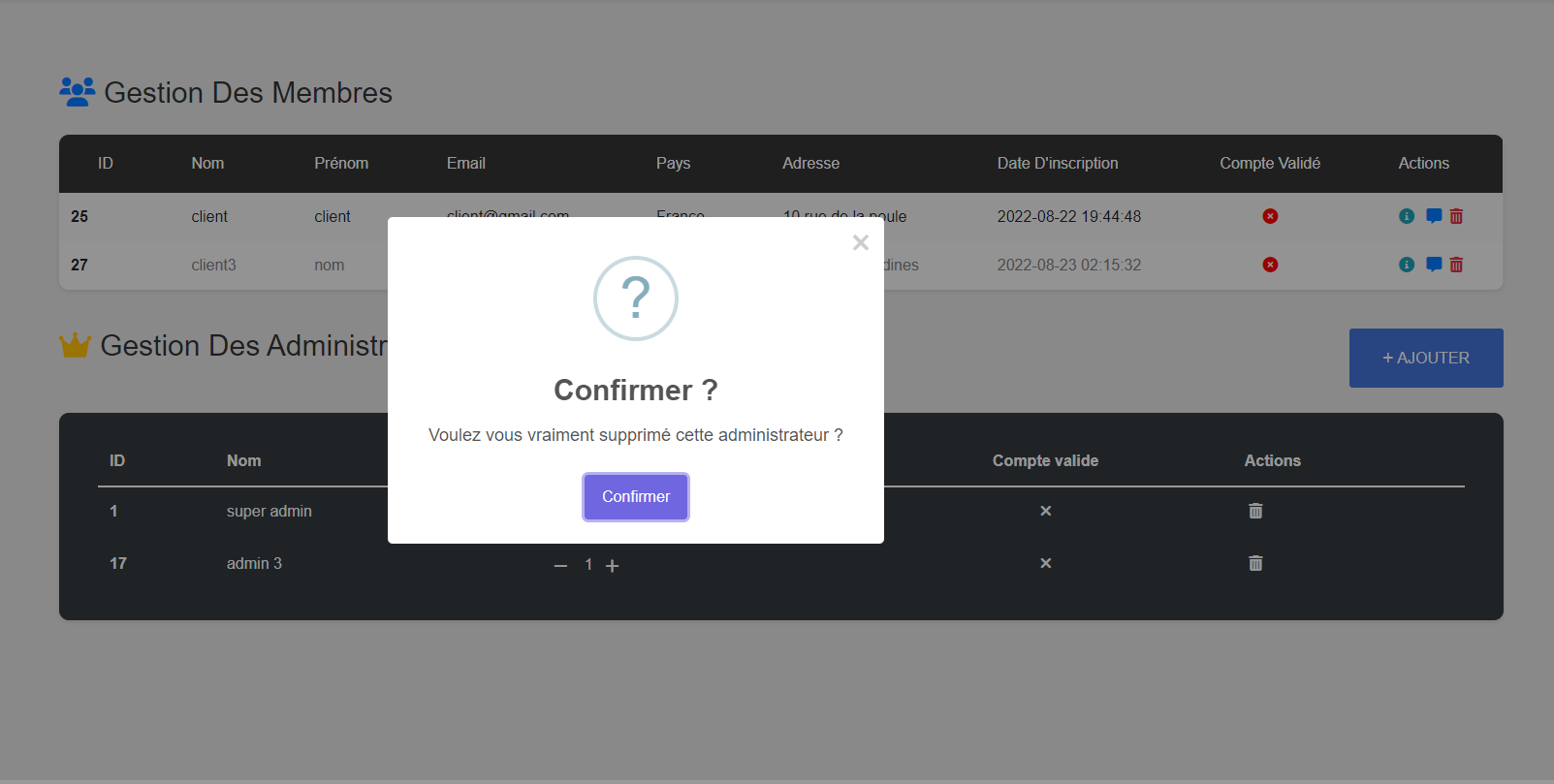
1.



2.



3.



**Framework SweetAlert2 :**

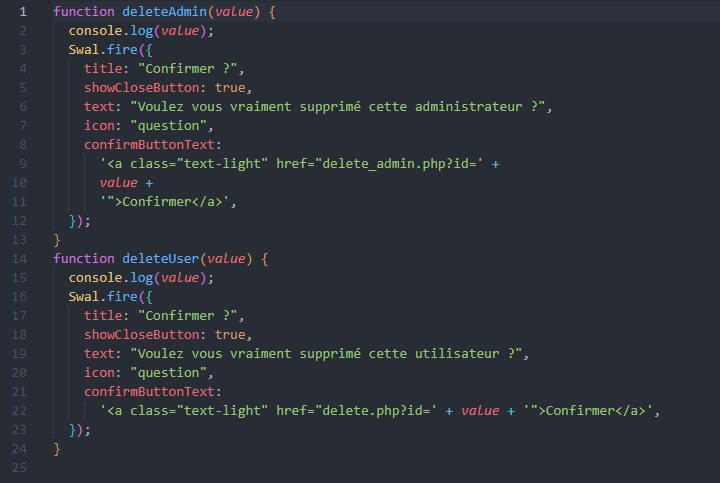
Les modal sont géré avec le framework sweetAlert2. Celui-ci propose plusieurs solutions ergonomiques pour styliser les pages sans trop de difficulté.

<https://sweetalert2.github.io/>

Pour cela nous devons juste importer le framework de la même manière que bootstrap



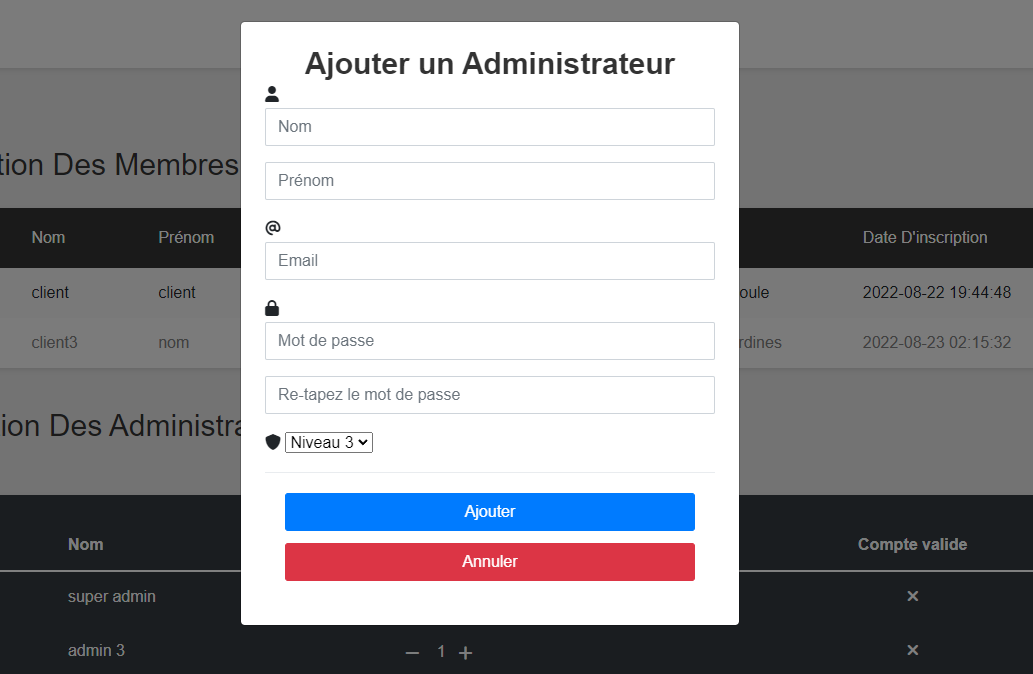
Une fois ceci effectuer nous avons juste à déclarer des alertes de la manière suivante :



Ici la fonction deleteAdmin va appeler une alerte « Swal.fire »

Ensuite nous avons juste à changer les paramètres selon ce que l’on désire dans la fenêtre.

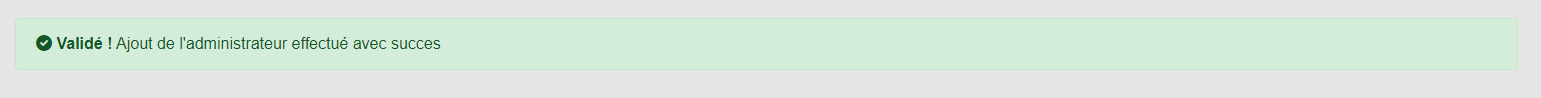
**Ajouter un administrateur :**

Création d’un bouton qui ouvre une modal contenant un formulaire pour l’ajout d’un administrateur

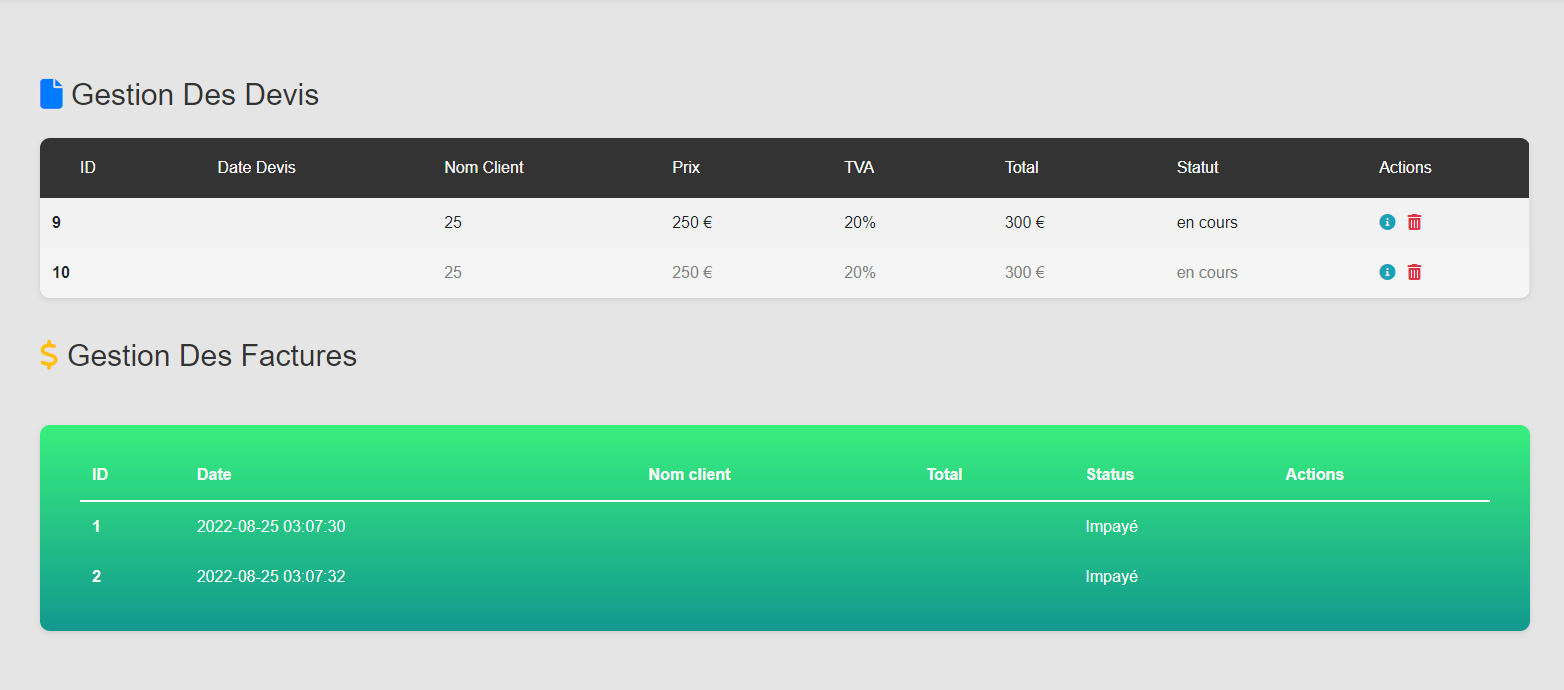
Dans ce formulaire nous renseignons le nom prénom de l’administrateur, son adresse email et nous lui définissons un mot de passe (celui-ci pourra être changer par l’administrateur par la suite).

Nous définissons aussi son niveau d’administration de 1 à 3

Une fois le clique effectuer sur le bouton « Ajouter » une notification apparait nous confirmant l’ajout de l’administrateur.



La gestion Devis/factures :



De la même manière que la gestion des membres, la gestion des devis/factures sont représenté par deux tableaux.

**Les devis :**

Ils sont triés par ID dans le tableau nous avons accès à leurs information principal :

- l’identifiant - date du devis

- nom du client

- prix - TVA

- Total - le statut (en cours, validé, refusé)

**Les actions**:

- information (ouvre une modal avec toutes les information).

- supprimé le devis.

**Les factures :**

Ils sont triés par ID dans le tableau nous avons accès à leurs information principal :

- l’identifiant - date de la facture

- nom du client - total

- prix - TVA

- Total - le statut (impayé, payé, refusé, partiellement payé)

**Les actions**:

- information (ouvre une modal avec toutes les information).

- supprimé la facture.

- bouton de modification (changer le statut de la facture)

Partie back-end

* 1. Présentation :

Pour la partie BACKEND, je n’ai eu pour instruction unique qu’elle soit le plus simple et lisible possible.

J’ai choisi de partir sur du PHP « from scratch ». Laravel aurait, peut-être, été plus pragmatique, mais je ressens le besoin de mieux connaître PHP.

Je suis satisfait d’avoir opté pour cette solution. Je pourrais, à présent, aborder un Framework plus sereinement.

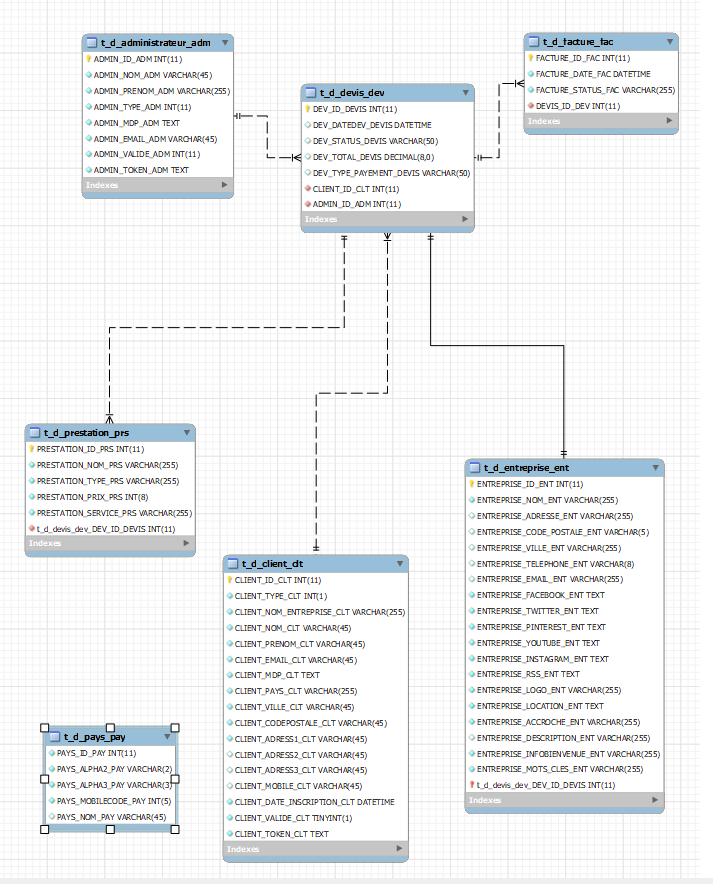
* 1. La base de données

Au premier jour de mon stage nous avons établi les besoins, j’ai donc imaginé les tables qui composerai ma base de données.

**Les tables imaginées :**

* Client - Devis
* Admin - Factures
* Entreprise
* Prestations

J’avais désormais de quoi travaillé et commencer à structurer ma base de données.



Voici les tables que composent la base de données.

Pays n’a pas de liaison avec d’autres tables (ajouter plus tard)

La table client est relié à la table devis grâce à une foreignKey

Et la table devis est relié à la table facture a son tour par une foreignKey

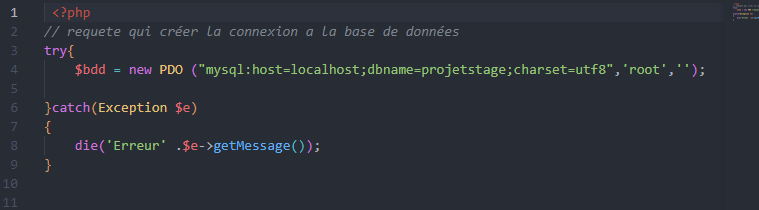
La table entreprise est relié à la table devis pour récupérer les informations de l’entreprise lors de l’édition des devis.

Les prestations sont reliées au devis pour récupérer les différentes informations

**Appel de la base de données :**

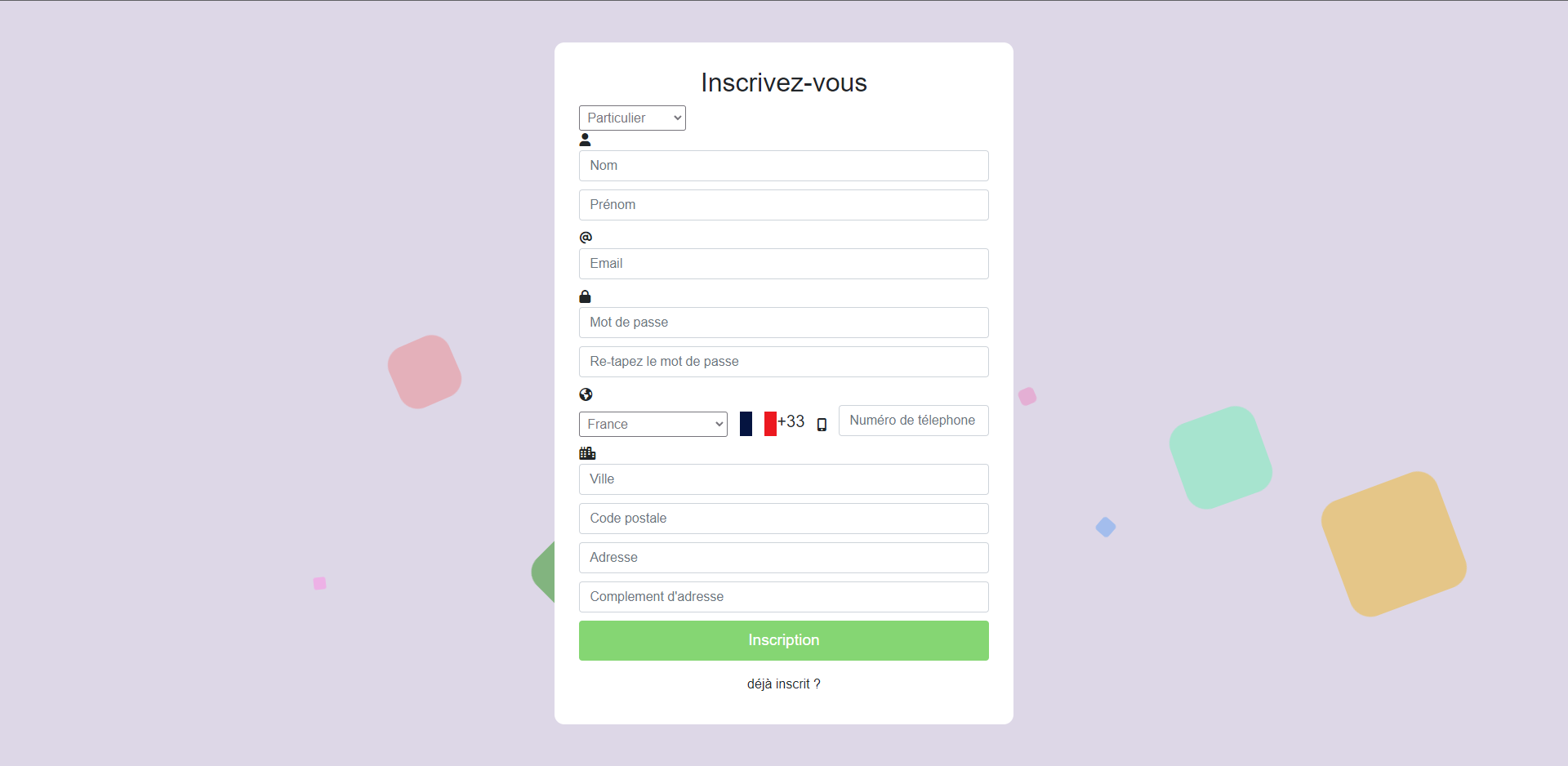
**Pour commencer nous devons appeler la base de données et l’inclure dans les pages php.**

Requête qui créer la connexion :

****

Inclure la BDD dans chaque page ou c’est nécessaire :.

    L’inscription :



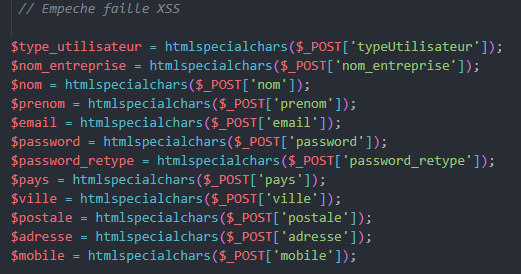
*le formulaire d’inscription est protege par des expressions regulieres, htmlspecialchars dans les input et une requete preparee.*

Nous pouvons désormais renvoyer le formulaire vers une page de traitement PHP qui va effectuer les requêtes d’insertion.



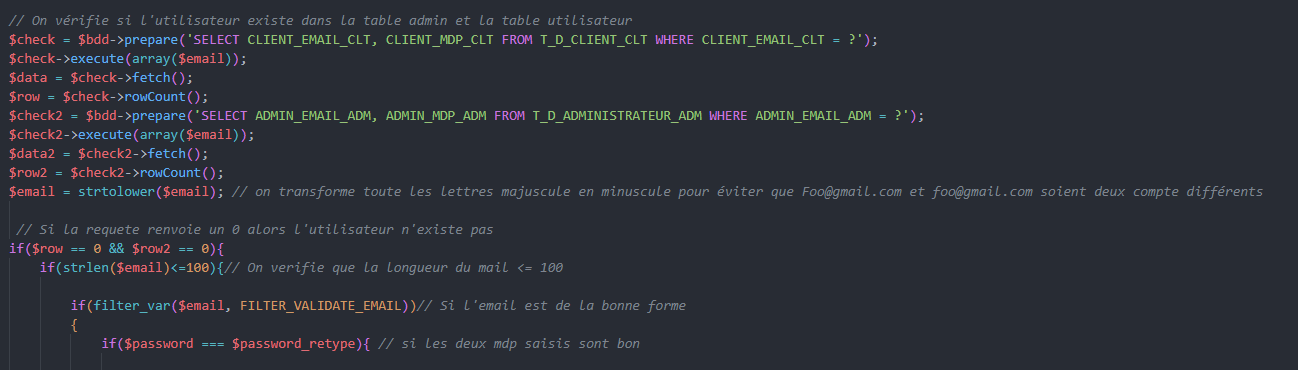
Chaque valeur envoyer par le Post sera stocker dans une variable avec la méthode ->

HTMLspecialchars ce qui empêche les failles SQL en vérifiant qu’il n’y a pas de balise ou tous autre caractère inapproprié.



**Ensuite nous procédons à plusieurs vérifications :**

* Que l’utilisateur n’existe pas déjà dans la table
* Que le mail n’est pas trop long
* Que le format de l’email est correct
* Que les deux mots de passe saisi correspondent



A chaque vérification une erreur correspond.

L’utilisateur existe :



Email trop long :



Format email incorrect :

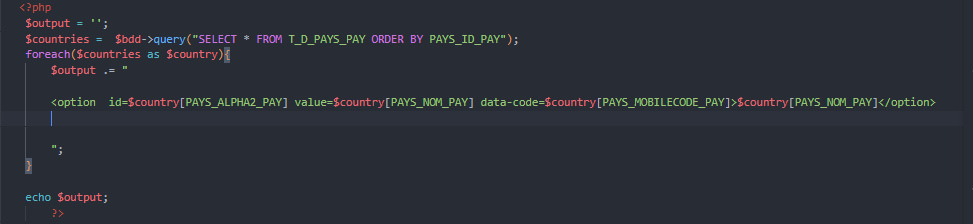


Mot de passe qui ne correspond pas :



**Les Drapeaux :**

Pour mettre en place le fait qu’un drapeau suivi du code téléphone correspondant au pays s’affiche après l’avoir choisi dans le select. Il a fallu que je trouve une table qui renseigne tous les pays ainsi que leurs codes de téléphone, une fois trouver, j’avais moyen de boucler dans cette table avec un foreach pour ressortir toutes les options possibles.



Chaque option prend comme id la colonne PAYS\_ALPHA2\_PAY qui correspond à « FR » pour France par exemple.

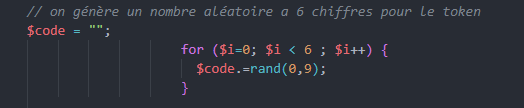
Comme valeur le nom du pays donc France pour la France.

On ajoute une propriété data-code qui correspond au code de téléphone.

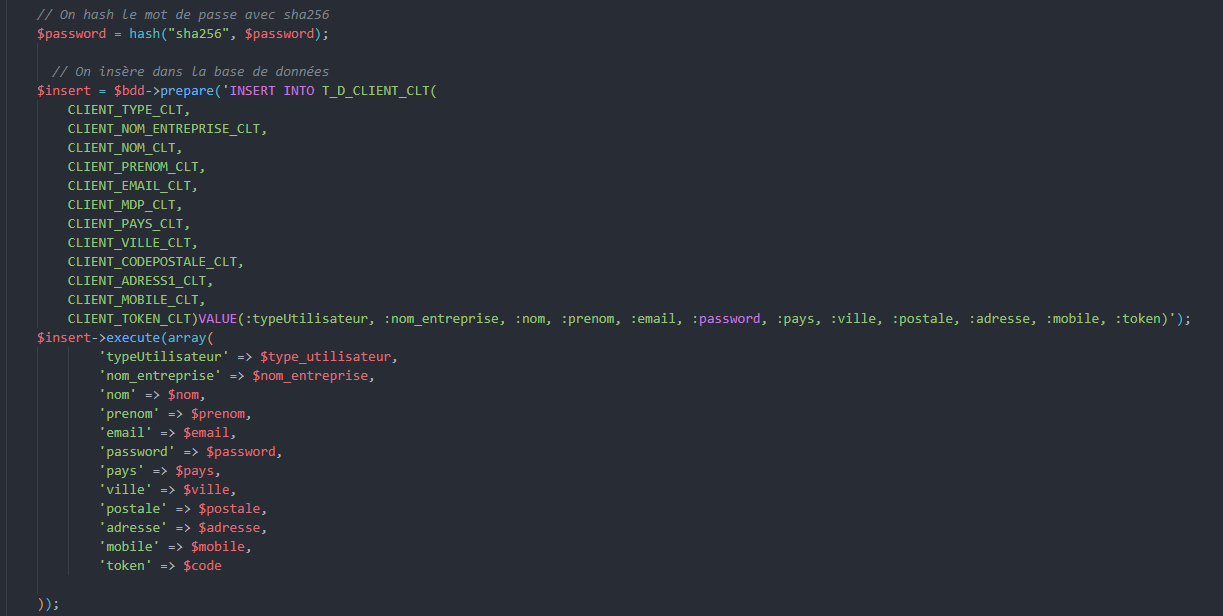
**La valeur par défaut est la France**.

Une fois toutes ces vérifications effectuer on Hash le mot de passe, ici en Sha256.

On définit un token pour la sécurité et pouvoir l’utiliser a différents endroits (changement de mot de passe, confirmation par mail etc…)



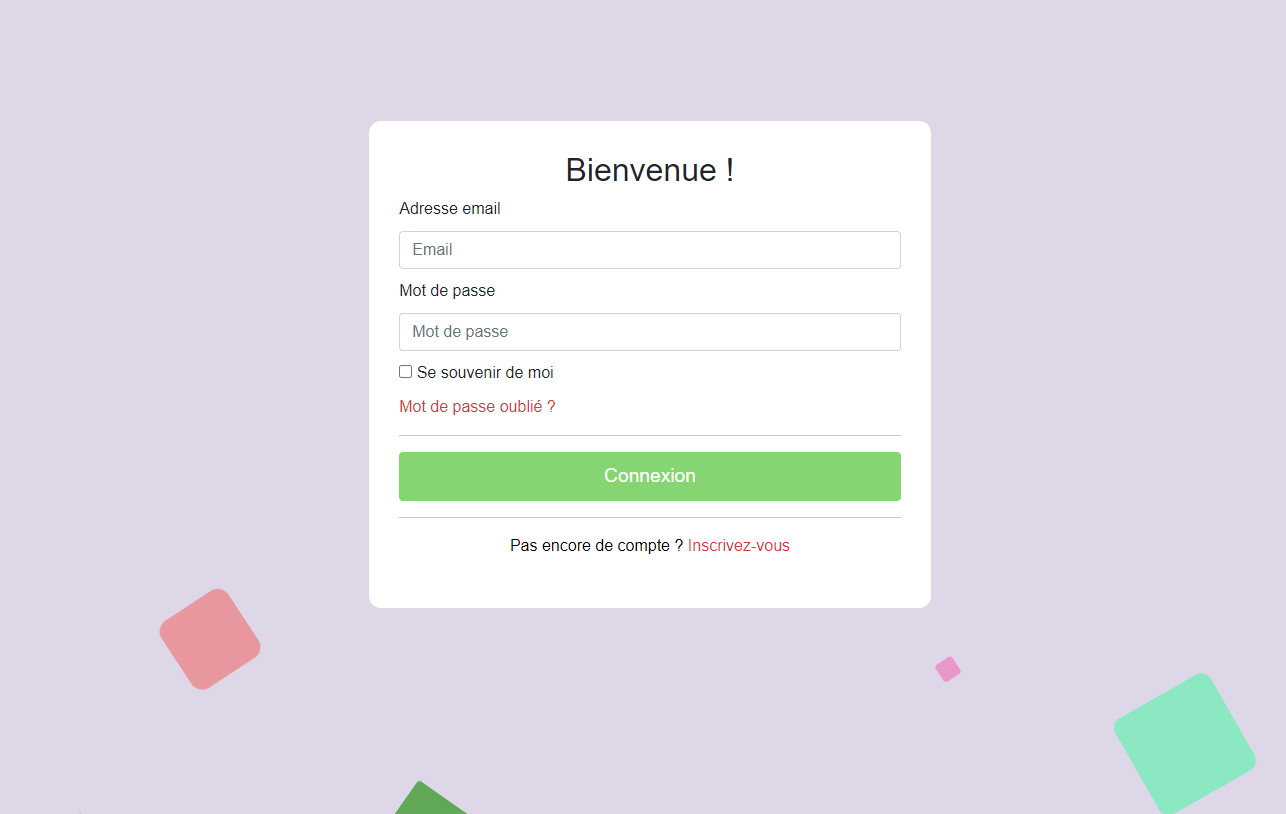
Puis on insère dans la base de données.



Une fois l’insertion effectuer on renvoie vers la page de connexion et on met fin au script

.

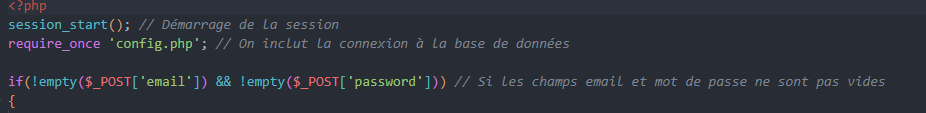
Connexion :



*le formulaire de connexion est aussiprotege par des expressions regulieres, htmlspecialchars dans les input et une requete preparee.*

Pour commencer nous devons démarrer une session et initialiser la connexion a la base de données

Ensuite nous vérifions si les champs input n’ont pas été envoyer vide, si c’est le cas on met fin au script sinon on passe à la suite…

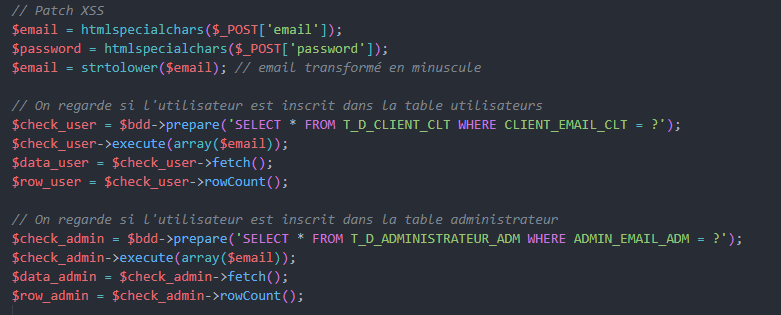


**Comme pour l’inscription nous stocker les valeurs dans des variable avec la methode ->**

**HTMLspecialchars**

**Puis on passe aux vérifications :**

Nous vérifions d’abord si l’email et le mot de passe rentré se trouve dans la table Client ou la table Administrateur.



Les variables $row\_admin ou $row\_user est supérieur a 0 alors l’utilisateur existe dans la table.

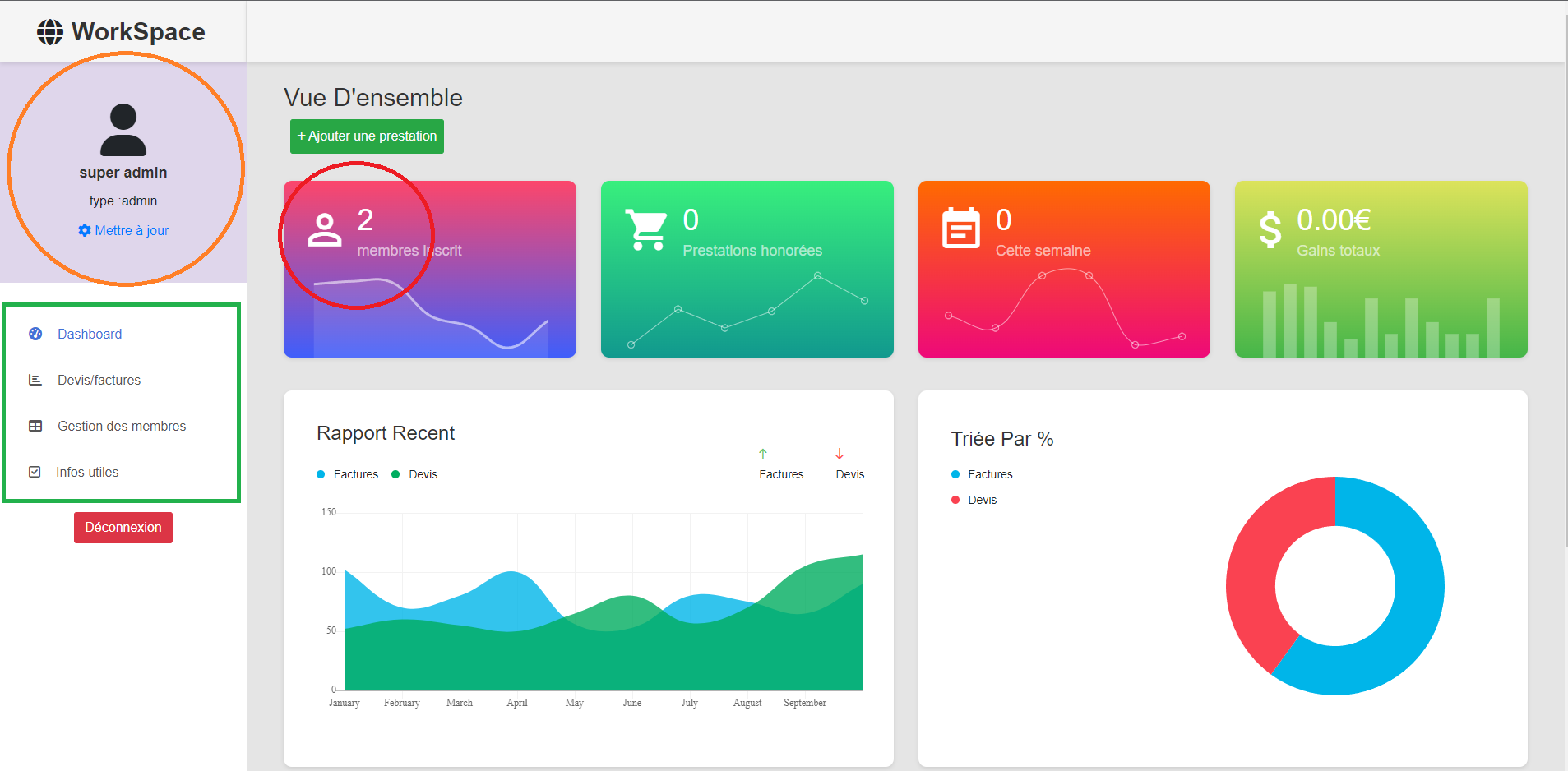
Alors on rentre dans la boucle, on vérifie le format de l’email, puis si le mot de passe correspond.

On créer ensuite la Session qui va stocker toutes les informations de l’utilisateur.

Puis lui assigner un rôle **! Très important pour gérer les droits !**



Dashboard admin :

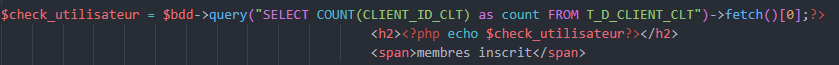


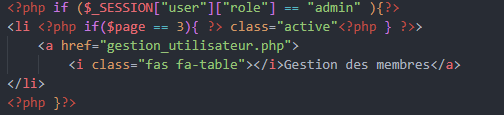
**Dashboard coté administrateur :**

Plusieurs requêtes sont mises en place sur cette page.

1. Récupérer les informations de la personne connecter
2. Récupérer le nombre de client inscrit dans la base de données
3. Afficher les liens dans la barre de navigation selon le type d’utilisateur.







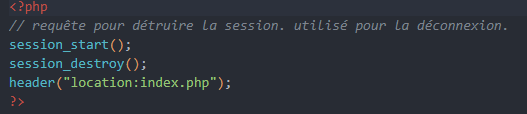
**Le bouton déconnexion :**

Pour mettre en place la déconnexion, une simple balise <a> avec comme propriété href la page deconnexion.php

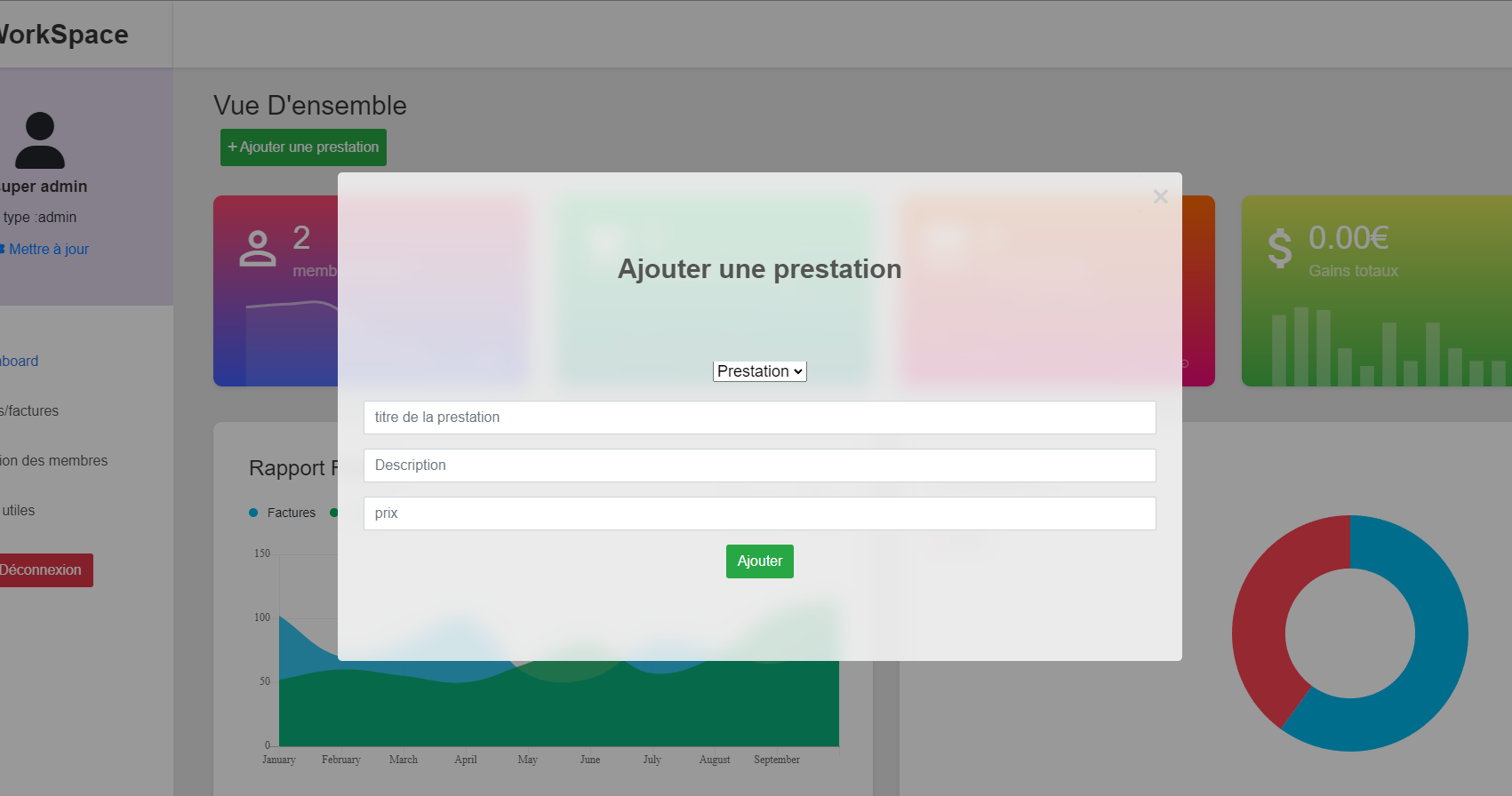
La fonction session\_destroy() comme son nom l’indique détruit la session actuel

Ensuite nous renvoyons vers la page de connexion.

**Deconnexion.php :**



**Ajouter une prestation :**

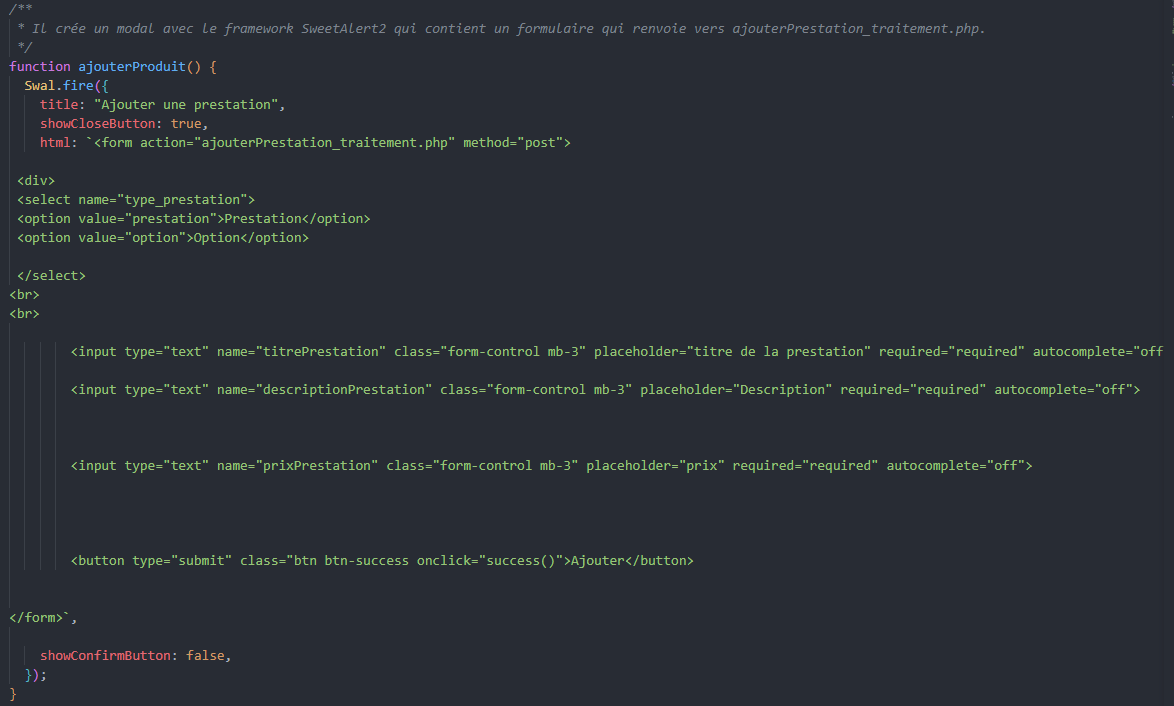
****

Les administrateurs peuvent ajouter des prestations.

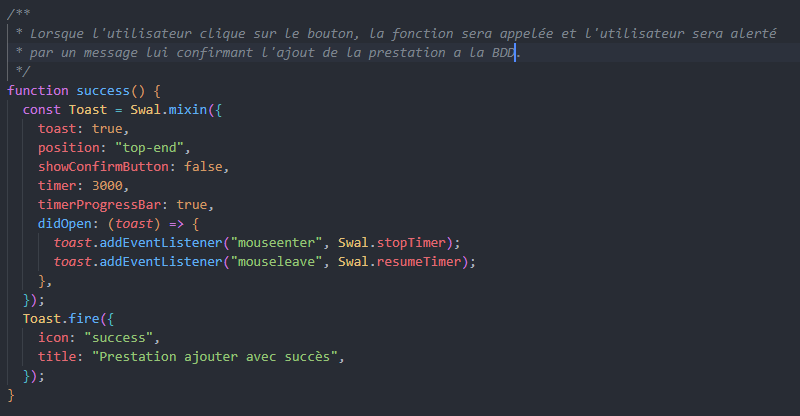
Lors du clique sur le bouton en question, une modal s’ouvre avec un formulaire, celui-ci renvoie vers une page php qui s’occupe d’insérer dans la table prestation les information saisies.

**La modal est toujours gérer avec le framework sweetAlert2 !**

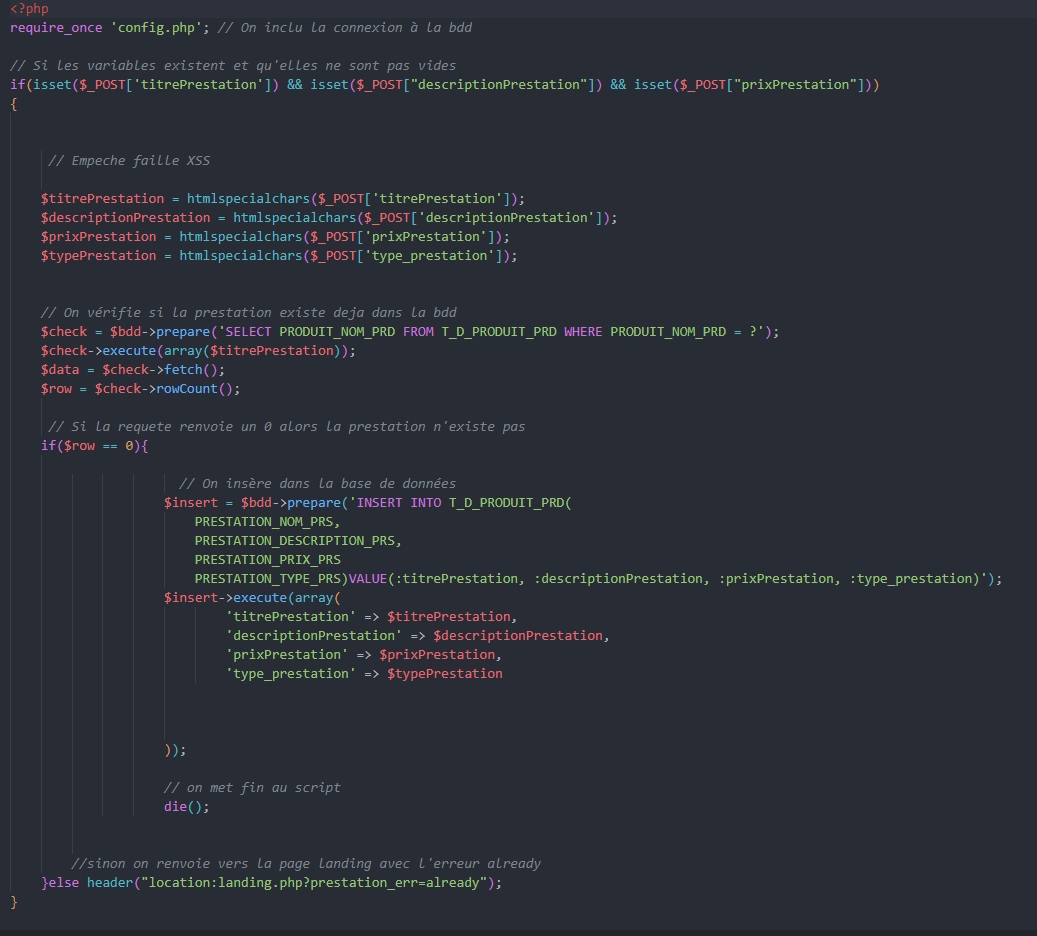
**Fonctions JS pour l’ouverture de la modal avec le formulaire :**

****

**Fonctions JS pour faire apparaitre un pop-up de confirmation d’ajout :**

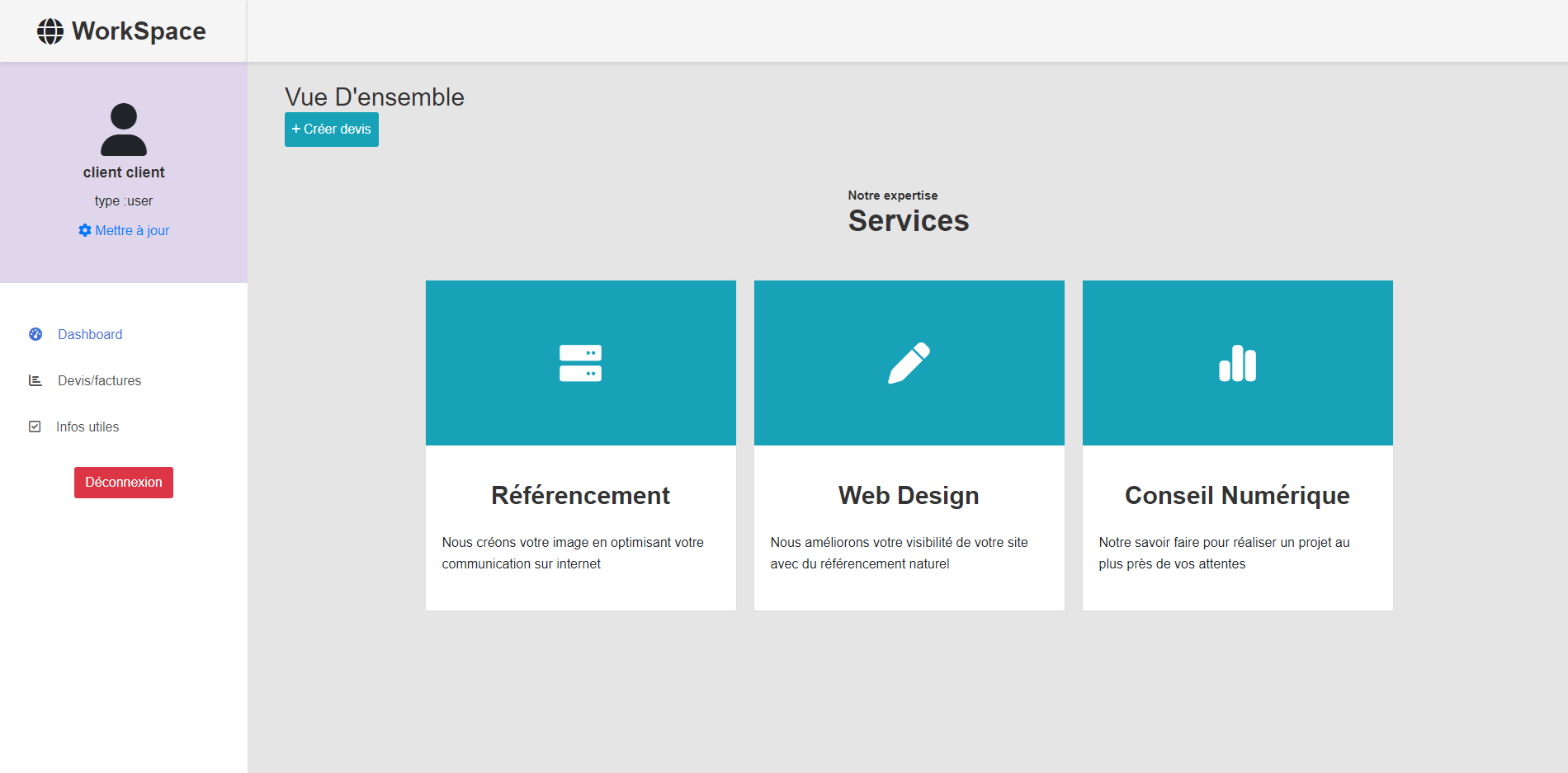


**La requête qui insert dans la base de données :**

****

1. On inclut la base de données
2. On stock les valeurs dans des variables avec la méthode HTMLspecialchars pour les failles SQL
3. On vérifie si la prestation n’existe pas déjà dans la BDD
4. On insère dans la BDD
5. On met fin au script

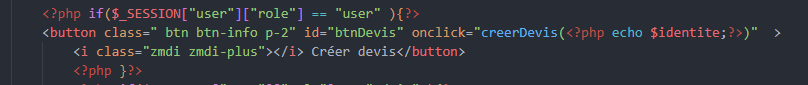
Dashboard client :



**Dashboard coté client :**

Le client a la possibilité de créer des devis. Le bouton ouvre une modal contenant un formulaire avec une barre de progression.

**La modal est toujours gérer avec le framework sweetAlert2 !**

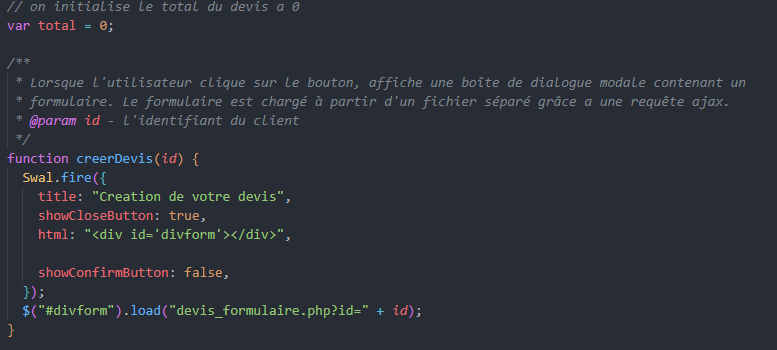


Le bouton possède une fonction au clique avec comme paramètre l’id du client connecter actuellement.



Il est important de passé l’id de l’utilisateur car celui-ci va nous permettre de relié le devis a son créateur beaucoup plus facilement

**La fonction JS qui gère la modal:**



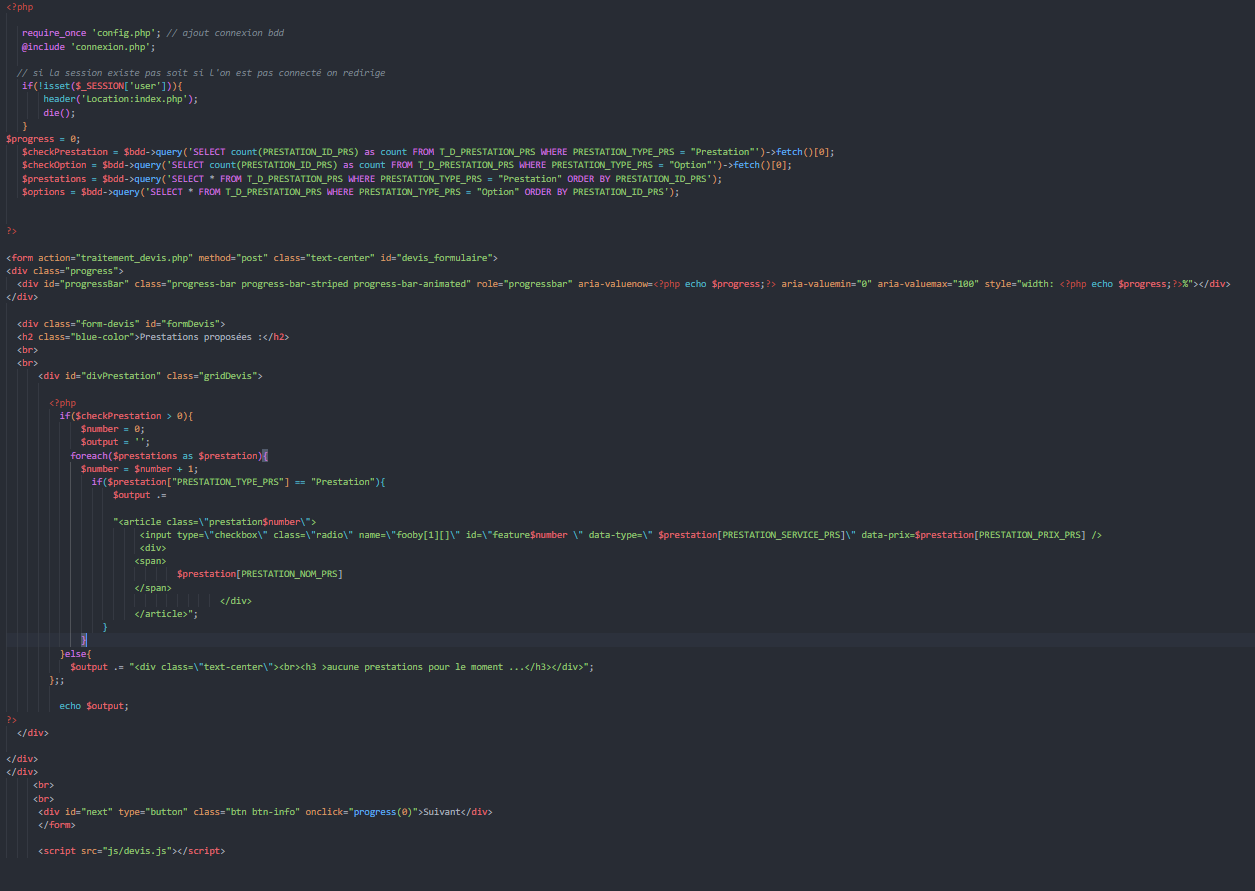
Pour commencer on initialize le total du devis a 0.

Ensuite on ouvre la modal avec le framework SweetAlert2

A l’intérieur de cette modal on créer une div avec comme ID “divForm”

Grâce a une requête ajax on ajoute a cette div le formulaire contenu dans la page “devis\_formulaire.php avec comme Valeur passé dans le get l’id de l’utilisateur.

**Formulaire php**:



**Explication du code JS :**

Tous d’abord nous allons vérifier s’il existe des prestations ou des options dans la BDD.

Ensuite s’il existe des prestations/options, pour chaque nous allons créer un input de type checkbox

Pour pouvoir leurs données un id unique nous déclarons une variable $number qui va s’incrémenter à chaque boucle.



Nous allons passer aussi plusieurs valeurs en data comme le type de service que cela concerne (maintenance, Sav, conception…) mais aussi le prix de la prestation/option.



**Le bouton suivant dispose d’une fonction au clique nommé « progress() »**

****

Le paramètre de cette fonction va servir à définir la progression de la barre.

**La fonction progress() :**

# 

A chaque clique sur le bouton suivant on incrémente la valeur de « pourcentage de 25 » la barre de progression avance donc de 25% à chaque fois

**Présentation du système de devis :**



En premier lieu vous devez choisir le type de prestation qui vous intéresse.

Chacune d’entre elle donne accès à des options différentes.

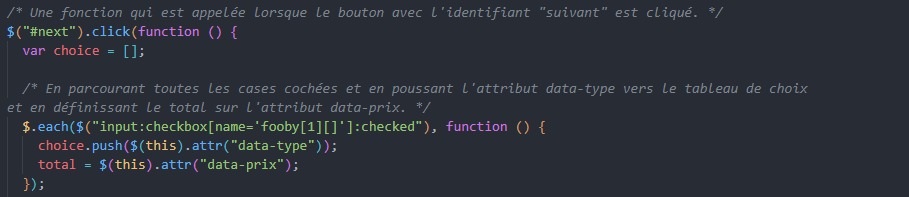
**Il était important de veillez à ce qu’un seul checkbox ne soit cocher à la fois !**

Pour cela j’ai mis en place une fonction qui va regrouper les inputs de type checkbox par nom dans une variable et vérifie s’il y en a une de cocher si oui décocher les autres



Pour donner accès aux bonnes options selon la prestation choisie j’ai décidé de passé par la vérification de l’attribut data-type de pousser cette valeur dans un tableau pour pouvoir la comparé.

Je récupère aussi au même moment l’attribut data-prix et je l’ajoute au total.



Une fois ceci fait, je peux passer à la vérification pour afficher les options correspondantes



****

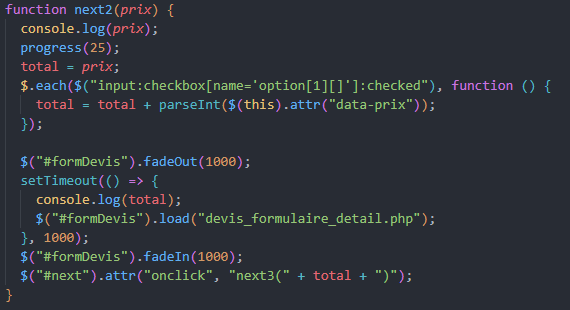
**Explication :**

Pour chaque possibilité nous avons une requête ajax pour afficher le formulaire depuis une page php.

Celle-ci est entouré de fonction JS fadeIn fadeOut pour un rendu plus ergonomique.

A ce moment-là nous attribuons aussi au bouton next une nouvelle fonction nommée next2()

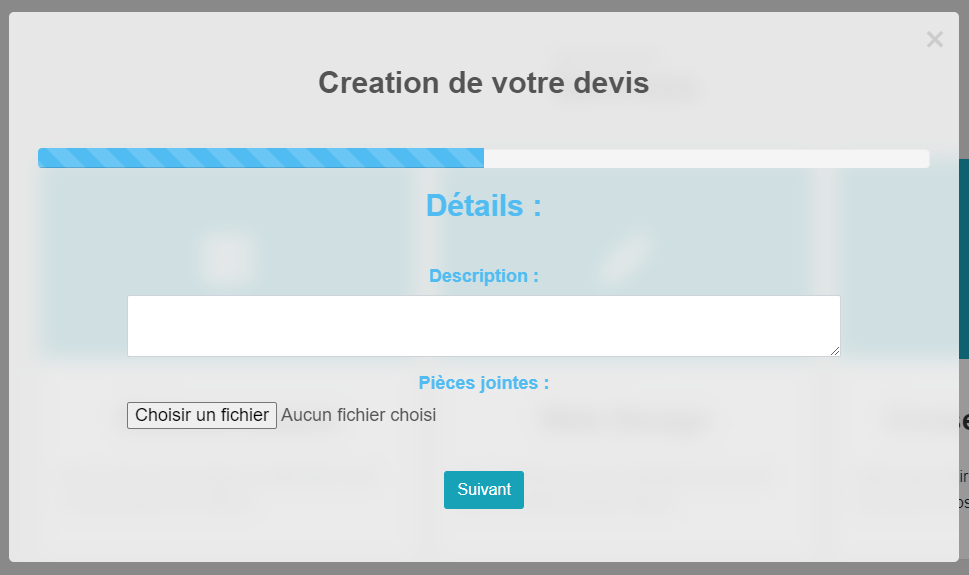
Avec comme paramètre le total du devis.



Cette fonction permet d’afficher la suite du devis de la même manière grâce à une requête ajax.

**Initialisé la barre de progression à 25% car déjà une étape de passé !**

A la fin de cette fonction nous changeons encore la fonction au clique du bouton par la fonction next3().



Cette fonction va afficher la dernière partie du devis avec l’estimation et les informations du client.

Puis envoyer les informations du formulaire dans une page de traitement php qui va insérer les valeurs dans la BDD





Recherche anglais :

Lors du développement j’étais en constante recherche de framework pour donner de l’esthétique et une identité à mon site.

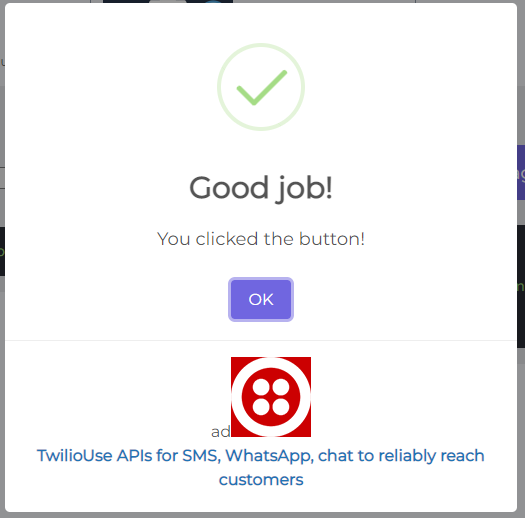
J’ai alors découvert le framework 

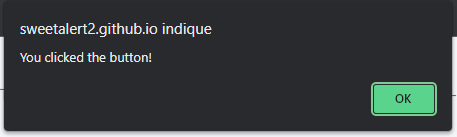
<https://sweetalert2.github.io/>

Le site étant en anglais, j’ai du me servir de mes compétences dans cette langue pour utilisé et remanier cette outils à mon envie.

Ce framework donne un nouvelle aspect au alertes Javascript et sont beaucoup plus esthétique.

**Alerte de base : Alerte SweetAlert2 :**



****

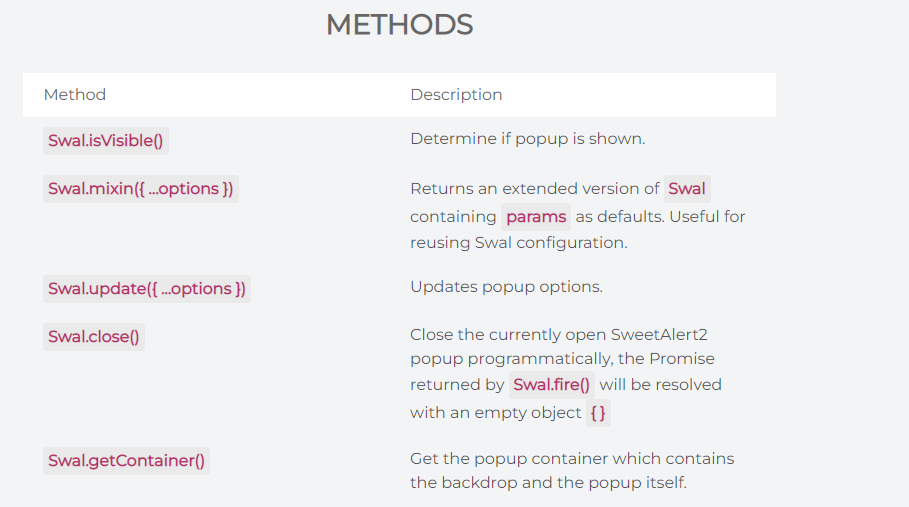
Pour commencer il me fallait savoir comment intégrer se framework je me rend donc dans l’onglet installation.



On nous explique qu’il suffit de référencer le plugin depuis une balise script, de la même manière qu’un fichier Javascript.

On peut aussi décider d’importer séparément le fichier CSS pour pouvoir modifier le style si on le désire.

En continuant de scroller je tombe sur différents exemples. 

Puis sur les différentes méthodes qui s’applique à ce plugin.

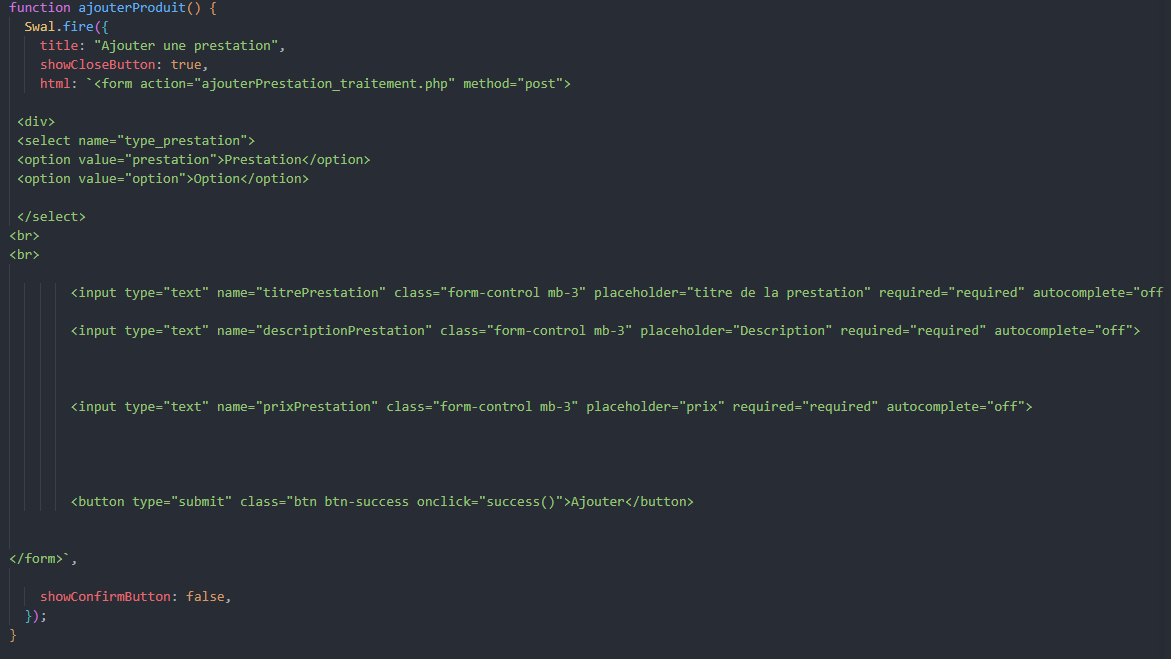
Par exemple :

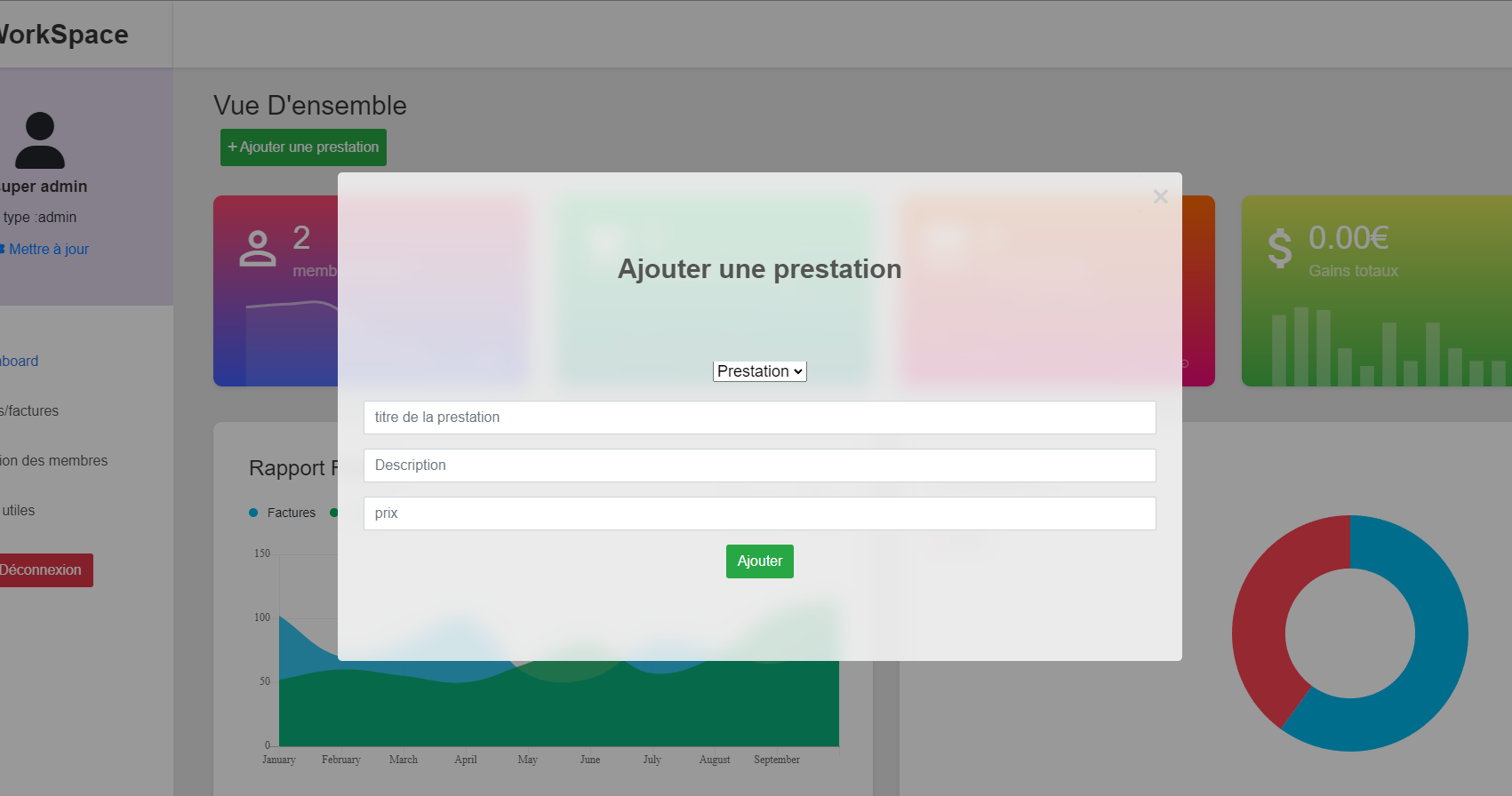
Swal.isVisible détermine si l’alerte est visible ou non

Swal.mixin{{…option}} retourne une version étendue du plugin, contenant des paramètres par défaut

J’ai compris sans trop de difficulté comment utilisé ce framework j’ai pu naviguer et chercher les informations au bon endroit. Je n’ai pas rencontré de problème particulier lors de la mise en place de mes alertes, j’y ai même inclus des formulaires

**Exemple d’utilisation du plugin :**

.

Ouvre une alerte contenant un formulaire pour l’ajout d’une prestation

Améliorations :

Étant un projet en cours, il y a beaucoup d’axes d’améliorations possibles.

Je prévois cela :

1. Mettre en place une confirmation par mail pour vérifier le compte
2. Construction de la catégorie « statistique ».
3. Gérer les envoie de messages aux utilisateurs.
4. Pouvoir attribuer un admin a un client pour s’occuper de leurs demande/réclamation par rapport à leurs devis.
5. Améliorer l’aspect esthétique de l’outil.

Synthèse :

J’ai , au maximum, tenté de rendre cet outil neutre et facile à retravailler pour l’adapter selon la demande. Ainsi je pourrais le proposer à différentes entreprises.

Au début de ce projet je ne pensais pas structurer mon travail de cette manière. Je voyais le back-end et le front-end comme deux blocs bien distincts. Cependant, j’ai pu me rendre compte , au cours de ce stage, qu’il est tout à fait possible de commencer par l’un ou par l’autre. Il est parfois nécessaire de les construire en simultané.

Cet outil saura répondre à plusieurs besoins. Une fois mis en place et opérationnel, il pourra servir à plusieurs types d’entreprises, notamment un développeur indépendant.

En effet, j’imagine travailler avec cet outil pour de futurs clients. Avoir un recueille de mes prestations et un œil sur les statistiques de mon entreprise me serait bénéfique.

Par la suite, je compte intégrer le calcul : des charges sociales, du chiffre d’affaires, et une gestion des factures classé par mois.

Conclusion :

Cette expérience m’as aussi permis de pratiquer le back-end, module sur lequel je rencontrais des difficultés et renforcer mes compétences dans ce dernier.

Mais aussi de confirmé mes compétences en design et en front-end de manière libre et se fut un grand plaisir.

Cette période en entreprise a été, pour moi, très formatrice. Elle m’a permis, de prendre conscience de l’indépendance d’un développeur et de sa capacité a recherché les solutions par ses propres moyens.

Il peut y avoir 1000 façons de résoudre un problème il faut juste être curieux et avoir soif de connaissance.

Stage effectué chez Altameos Multimédia 11/07/2022 au 16/09/2022

Jonathan OPIGEZ

*AFPA Evreux 2021/2022*

Pour finir je tiens à remercier l’entreprise AltameosMultimedia qui m’as permis d’effectuer ce stage dans de très bonne conditions, et de me laisser cette liberté qui m’as donné cette envie de me dépasser.

Je remercie également l’Afpa qui m’as accompagné durant 7 mois et grâce à qui mes compétences en tant que développeur se sont affutées.