POTTIER Matthew

Titre Professionnel Niveau III

Développeur Web / Web mobile

Session Avril 2021 / Février 2022

Mémoire professionnel



Table des matières

[1 – INTRODUCTION 3](#_Toc94103718)

[1.1– Introduction personnelle 3](#_Toc94103719)

[1.2– Introduction professionnelle 3](#_Toc94103720)

[1.3– Abstract (Personal introduction) 5](#_Toc94103721)

[2 – COMPÉTENCES COUVERTES 6](#_Toc94103722)

[2.1– Compétences Front-end 6](#_Toc94103723)

[2.2– Compétences Back-end 6](#_Toc94103724)

[3 – ANALYSE DU BESOIN 7](#_Toc94103725)

[3.1– Présentation de l’entreprise 7](#_Toc94103726)

[3.2– Objectifs et cibles 7](#_Toc94103727)

[3.3– Les fonctions du produit 7](#_Toc94103728)

[4 – SPÉCIFICATIONS FONCTIONNELLES 9](#_Toc94103729)

[4.1– Use Case 9](#_Toc94103730)

[4.2– Diagramme d’activité 10](#_Toc94103731)

[4.3– Diagramme de séquence 11](#_Toc94103732)

[4.4– Maquettage 12](#_Toc94103733)

[5 – CONCEPTION DE LA BASE DE DONNÉES 16](#_Toc94103734)

[5.1– MCD / MLD 16](#_Toc94103735)

[5.2– Création de la base de données 20](#_Toc94103736)

[6 – ARBORESCENCE 21](#_Toc94103737)

[7 – OUTILS UTILISÉS 22](#_Toc94103738)

[8 – INTÉGRATIONS DES FONCTIONNALITÉS 24](#_Toc94103739)

[8.1– Page de connexion 24](#_Toc94103740)

[8.2– Ajout d’un support 30](#_Toc94103741)

[8.3– Intégration Twitch et Responsive 35](#_Toc94103742)

# 1 – INTRODUCTION

## 1.1– Introduction personnelle

Je m’appelle Matthew POTTIER, j’ai 28 ans et j’habite sur Montpellier depuis maintenant 4 ans. Je suis originaire de Chartres (28) où j’ai passé un quart de ma vie avant de déménager dans le sud (tout d’abord à Béziers puis à Montpellier). J’ai toujours baigné dans le domaine informatique, avec un père à jour sur les dernières technologies, mais également dans le domaine des jeux-vidéo et c’est pour cela que je suis passionné par l’informatique et le gaming. J’ai d’ailleurs ouvert une chaîne Twitch il y a plusieurs années maintenant afin de partager cette passion.

J’ai suivi un cursus scolaire plutôt classique en faisant un Baccalauréat Économique & Social que j’ai validé avec une mention assez bien, puis n’ayant pas une idée claire sur quel métier je voulais faire j’ai fait une année de STAPS (Sciences et Techniques des Activités Physiques et Sportives) à l’université d’Orléans que je n’ai pas validé car les cours dispensés ne correspondaient pas à ce que je voulais faire.

Ne sachant toujours pas dans quel domaine m’orienter j’ai commencé à travailler dans la restauration rapide pendant environ 2 ans. Suite à ça j’ai eu plusieurs missions d’intérim puis j’ai suivi une formation professionnalisante d’agent de fabrication / peseur en usine qui m’a permis de travailler pendant la durée de cette formation dans une usine de cosmétique, formation au bout de laquelle j’ai décroché mon diplôme. J’ai ensuite continué les boulots d’intérim pendant un temps avant de me rendre compte que je ne trouvais toujours pas ce qui me passionnait, c’est donc à ce moment que j’ai décidé de déménager dans le sud afin d’y tenter ma chance.

J’ai commencé à travailler dans le domaine du support informatique, une porte d’entrée au monde numérique, pendant 2 ans également (mais en changeant d’entreprise au bout d’un an) et c’est au bout de ces deux ans que j’ai découvert l’Adrar ainsi que ses formations proposées dont celle qui touche désormais à sa fin, celle de développeur web et web mobile.

## 1.2– Introduction professionnelle

Dans le cadre de ma formation de développeur web / web mobile à l’Adrar de Montpellier se déroulant du 6 Avril 2021 au 11 Février 2022, j’ai développé un projet fil rouge centré sur ma passion du jeu-vidéo et sur ma chaîne Twitch. Le but de ce projet est de centraliser toutes les informations me concernant en regroupant un lecteur pour pouvoir regarder les directs quand je suis en ligne, un résumé de mon aventure sur Twitch avec mes habitudes de jeu et une bibliothèque donnant des informations sur les jeux auxquels j’ai joué avec mon avis.

Le projet est codé totalement en PHP orienté objet avec un modèle MVC et est hébergé sur la plateforme LWS. Ce projet a été mis en ligne dès le 2ème mois de formation car l’idée d’avoir un site lié à mon activité sur Twitch a germé à partir du moment où nous avons appris à faire nos premières pages en HTML/CSS.

L’inspiration principale pour mon site est celui de ZeratoR (zerator.com) car c’est une sommité dans le Twitch français et commençant à avoir quelques notions de codage à l’époque j’étais impressionné par le rendu de son site.

J’ai donc commencé à développer mon site en partant de rien, en utilisant les inspirations pour le côté visuel et en utilisant les exercices que l’on voyait en cours pour implanter de nouvelles fonctionnalités. Au départ il n’y avait que quelques pages basiques (Accueil, Présentation) mais la structure et le code couleur était déjà présent. J’ai tout d’abord ajouté à la page d’accueil le lecteur vidéo via les balises <iframe> et <script> (les deux méthodes sont disponibles dans mon code car le pare-feu de l’Adrar me bloquait le lecteur vidéo quand le script était utilisé) car c’est une des fonctionnalités principale de mon projet.

En écrivant le cahier des charges de ce projet, l’idée de la bibliothèque de jeux-vidéo était beaucoup plus (et même trop) ambitieuse. J’étais parti sur le principe de faire des animations 3D pour se déplacer entre les constructeurs, les consoles et les jeux correspondant chacun à une pièce, puis pour chaque jeu que la jaquette soit un modèle 3D qui s’ouvre une fois que l’on clique dessus.

Finalement l’idée sera mise en suspens pour quelque chose de plus accessible en termes de temps : un carrousel avec l’image du jeu.

Plusieurs fonctionnalités auront des améliorations dans le futur, notamment la bibliothèque avec l’idée précédemment citée ou encore une page de recherche de jeu regroupant tous les constructeurs, supports, jeux avec des filtres disponibles pour affiner la recherche.

## 1.3– Abstract (Personal introduction)

My name is Matthew POTTIER, I’m 28 years old and I live in Montpellier since 2018. I’m from Chartres (28) where I lived for a quarter of my life before moving in the south (first in Béziers and then in Montpellier). I always was surrounded by digital with my dad following all the new technologies, but also surrounded by videogames and that’s why I’m passionate by gaming and digital. I also created a Twitch channel few years ago from now so I can share this passion with other.

I did a pretty usual scholarship by doing a Baccalauréat Économique & Social that I passed with good grades, thus since I wasn’t sure what I wanted to do I did a first year in STAPS at Orleans’ university that I didn’t complete because it didn’t match with my expectations.

Since I still didn’t know in which domain I wanted to go, I started to work in a fast food for nearly 2 years. Therefor I did some temporary jobs and I followed a learning to become a Manufacturing process operator / weigher who got me into working in a cosmetic factory in the same time of that learning. In the end I got the diploma. After this experience I did some more temporary jobs before finding out that I will not get what I’m really looking for, that’s when I convinced myself to move in the south to try.

I started to work in the digital domain with Helpdesk for two years (but changing company in between) and after this two years I discovered Adrar and their learning, especially this one who comes to its end.

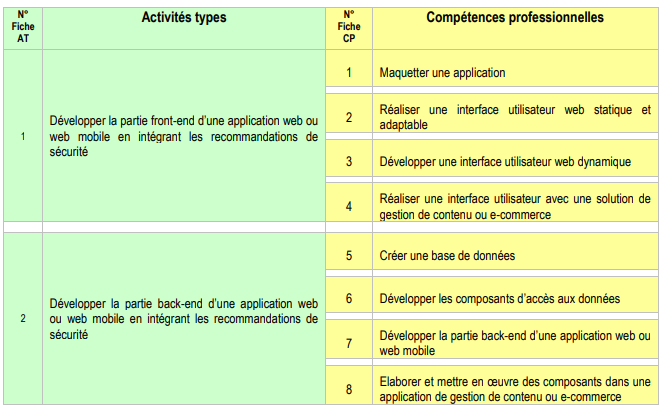
# 2 – COMPÉTENCES COUVERTES

## 2.1– Compétences Front-end

* Maquetter une application
* Réaliser une interface utilisateur web statique et adaptable
* Développer une interface utilisateur web dynamique

## 2.2– Compétences Back-end

* Créer une base de données
* Développer les composants d’accès aux données
* Développer la partie back-end d’une application web ou web mobile



# 3 – ANALYSE DU BESOIN

## 3.1– Présentation de l’entreprise

* Matsashi
* Domiciliation : 175 Avenue du Professeur Étienne Antonelli 34070 Montpellier
* Téléphone : 06 43 87 99 82
* Courriel : [contact@matsashi.fr](mailto:contact@matsashi.fr)
* Représentée par : Matthew POTTIER
* Histoire de la marque : S’étant lancé dans l’aventure du stream sur Twitch depuis plusieurs années, Matthew POTTIER a voulu avoir une marque avec une identité visuelle propre et permettant, si la chance lui sourit, d’être reconnu sur Internet.

## 3.2– Objectifs et cibles

L’objectif est d’avoir un site regroupant toutes les informations concernant le contenu de la marque « Matsashi ». De ce fait il faudra plusieurs pages comportant une retransmission de la chaîne, une bibliothèque qui contient tous les jeux possédés/joués, une page contact avec un formulaire ainsi qu’une partie biographique sur la page d’accueil.

Le site arbore une charte graphique propre ainsi qu’un logo déjà établi. Les couleurs principales sont en lien avec le violet, le noir et le jaune pour coller au logo et à la charte graphique de Twitch.

La tranche d’âge visée est les enfants/adolescents/jeunes adulte (donc entre 8 ans et 25 ans). Cependant une partie minoritaire vise également les personnes un peu plus âgées (entre 25 ans et 50 ans).

## 3.3– Les fonctions du produit

Le projet a énormément évolué depuis sa création, à tel point que le cahier des charges fait à l’époque n’est plus du tout d’actualité. Je l’ajouterais cependant dans le Dossier Professionnel pour ceux qui voudrait voir son évolution.

A l’heure actuelle le contenu du site est disponible qu’en une seule langue (français) et donc ne nécessite pas d’interface différente. Il n’y a également pas de newsletter pour le moment, mais l’intégration de cette dernière pour notifier de l’ajout d’un jeu ou d’un commentaire est une idée d’amélioration du projet.

La principale fonctionnalité du projet réside dans la bibliothèque qui va regrouper toutes les consoles et les jeux auxquels j’ai joué, avec pour chaque jeu plusieurs informations affichés dans un menu à onglets réunis par thème (« informations du jeu », « temps de jeu & taux de complétion », « vidéos & clips » et « avis »).

Les informations du jeu seront :

* Nom du jeu
* Support du jeu
* Genre du jeu
* Date de sortie
* Éditeur / Développeur
* Mode(s) (Solo et/ou multijoueur)

La catégorie « temps de jeu & taux de complétion » regroupera :

* Le temps (en jeu et/ou réel) passé sur le jeu
* Le taux de complétion (complétion en jeu et/ou complétion des trophées/succès)
* Dans le cas échéant, un message disant qu’il n’y a pas d’informations sur ce jeu

La catégorie « vidéos & clips » regroupera :

* Un lien vers les vidéos du jeu de Matsashi (s’il y en a)
* Un extrait du jeu de Matsashi (s’il y en a)
* Dans le cas échéant, un message disant qu’il n’y a pas encore de vidéos sur ce jeu

La catégorie « avis » comportera :

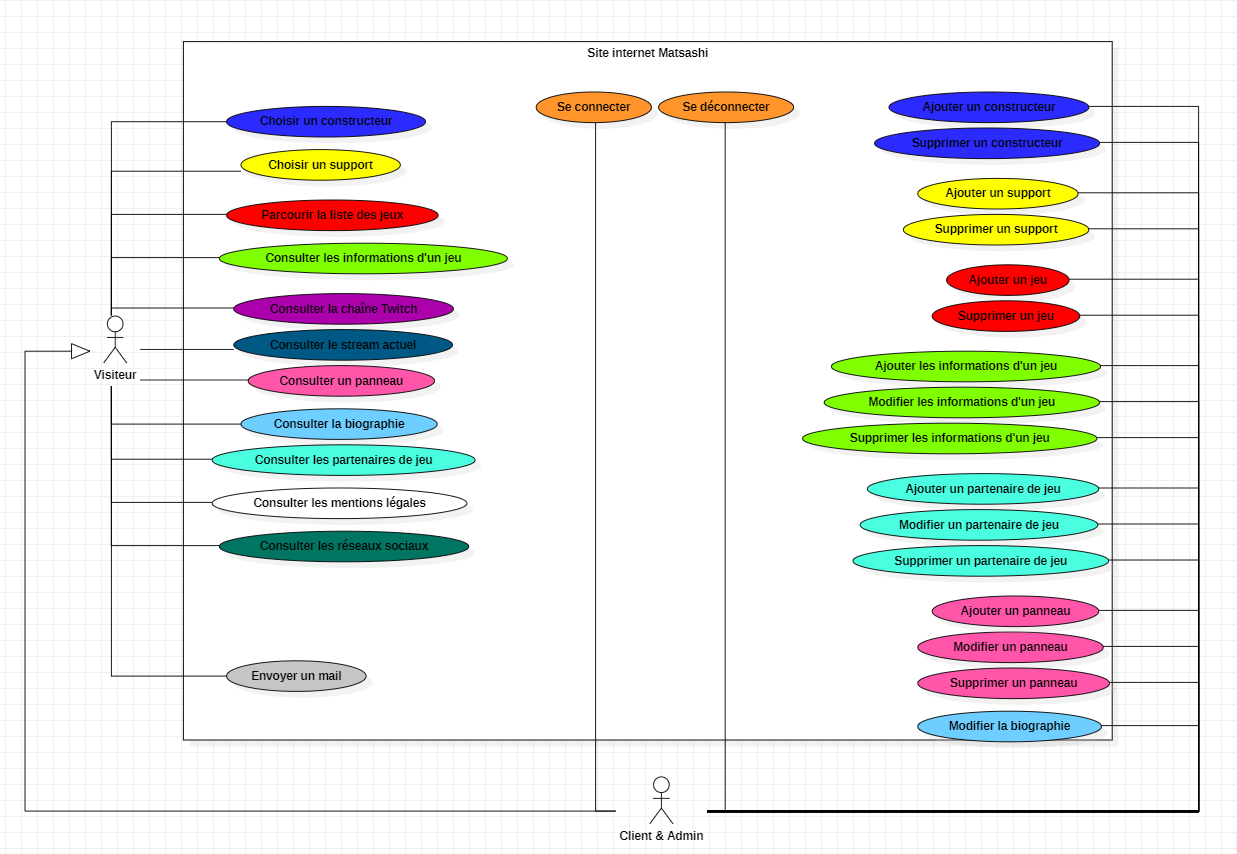
* Un texte résumant le ressenti de Matsashi concernant le jeu (potentiellement différent selon le support sur lequel le jeu a été joué)
* Un commentaire final (pour recommander ou non le jeu)
* Dans le cas échéant, un message disant qu’il n’y a pas d’avis sur ce jeu

Une fonctionnalité de ce type demande une grosse partie de back-end qui sera gérée par la partie Administrateur du site. L’accès à la page d’administration ne se fait que via la barre d’adresse, aucun lien sur le site ne permet d’y accéder. Les admin sont choisis directement par moi-même et la création des accès se fait seulement via la base de données (aucune page de création d’utilisateur n’existe). En supplément il y a une page de connexion afin que même si un utilisateur trouve le lien il ne puisse pas accéder à la partie administration et faire des modifications.

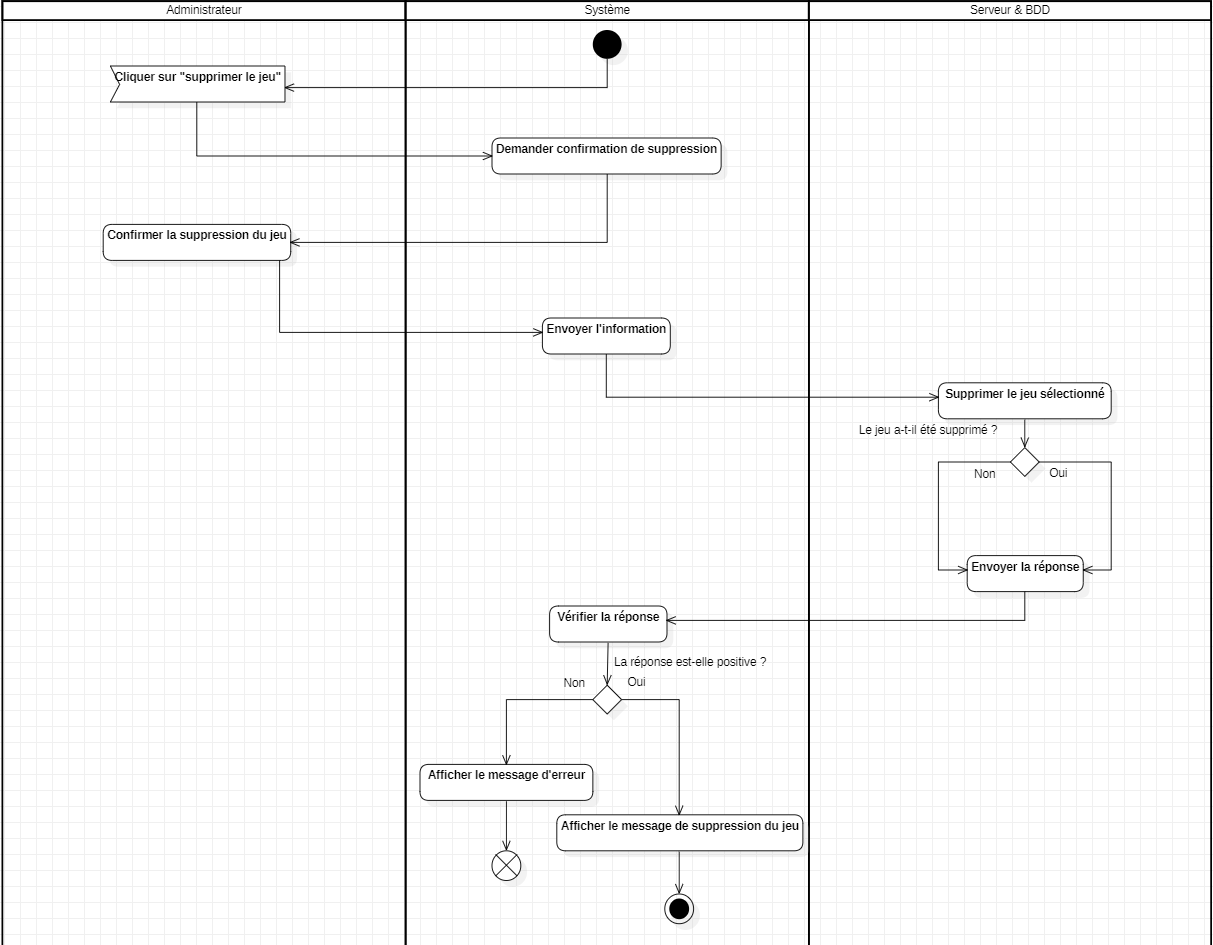
Dans cette partie admin les principales fonctionnalités sont l’ajout/modification des jeux et des supports.

# 4 – SPÉCIFICATIONS FONCTIONNELLES

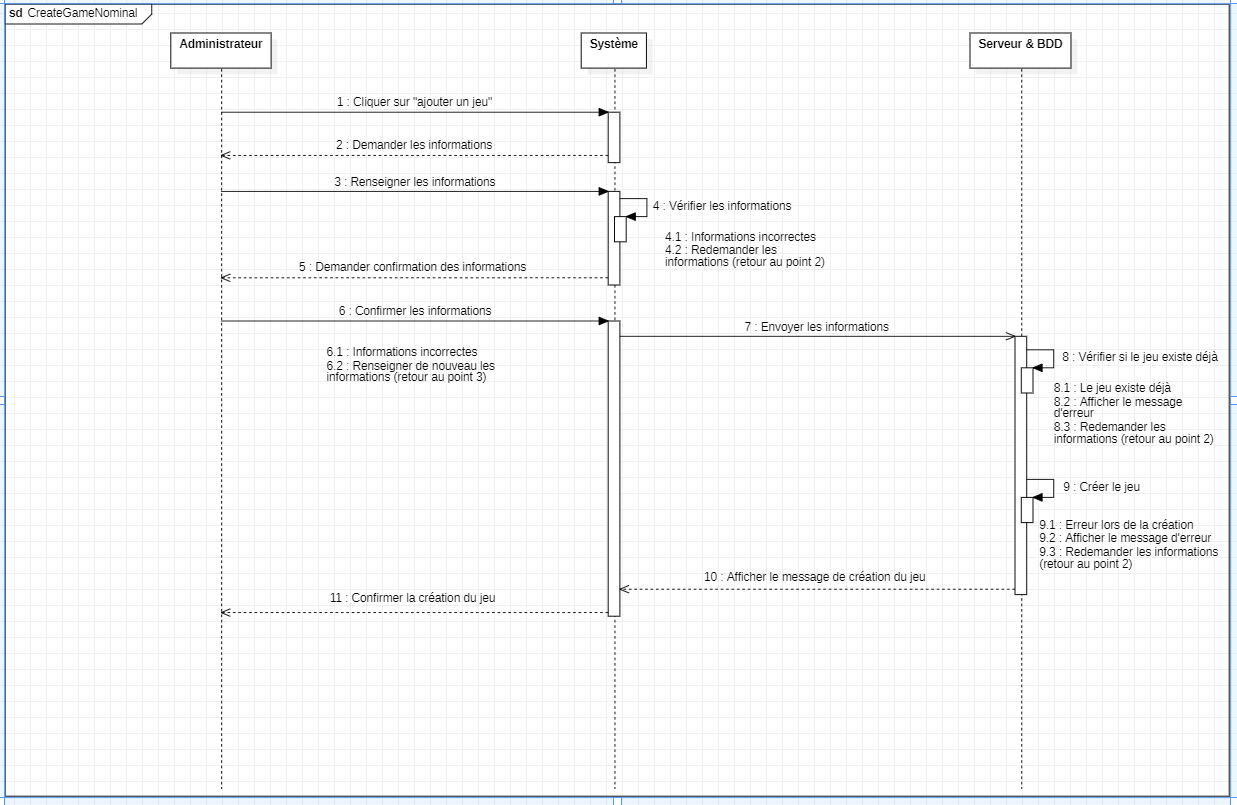
## 4.1– Use Case



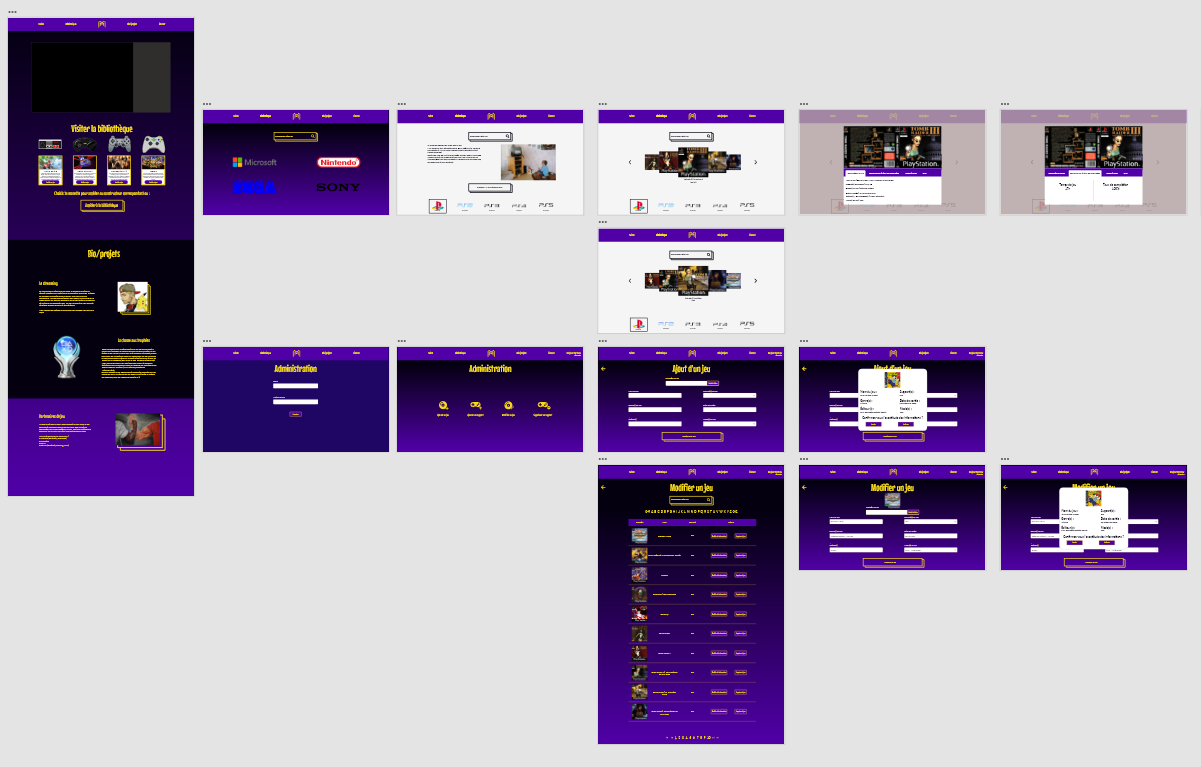
## 4.2– Diagramme d’activité

Diagramme d’activité sur la suppression d’un jeu

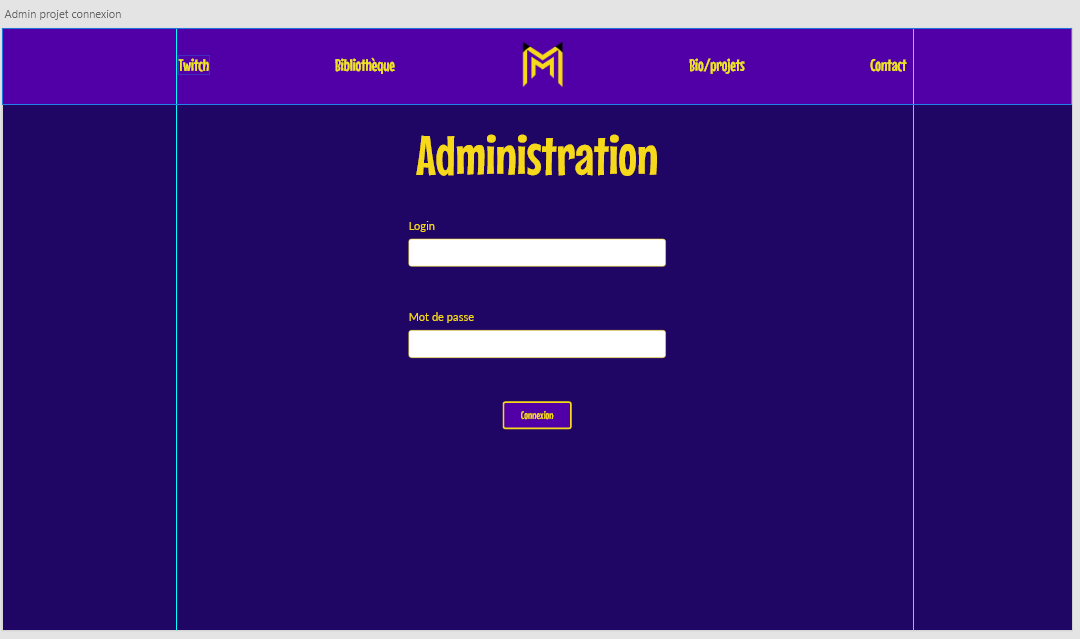
## 4.3– Diagramme de séquence

Diagramme de séquence sur l’ajout d’un jeu

## 4.4– Maquettage

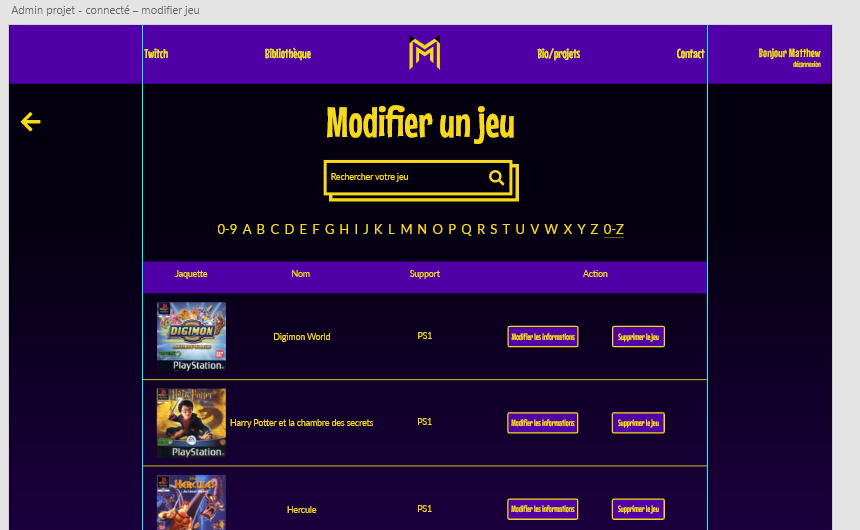


Maquette de la page de connexion :



Maquette de la page d’ajout de jeu :

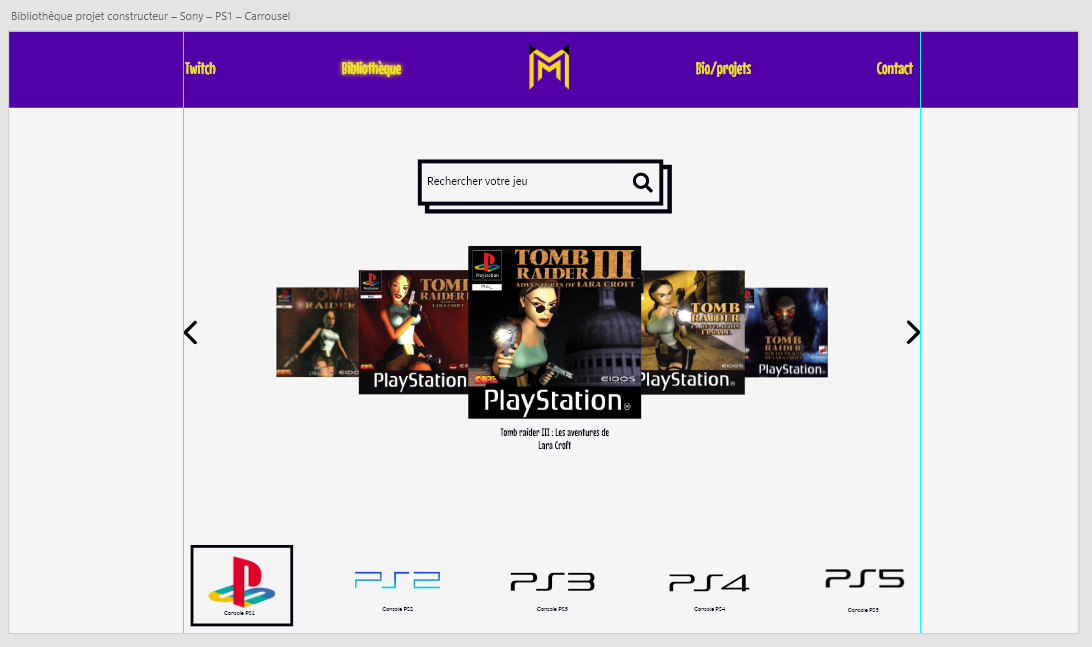
Maquette de la page de liste de jeu :



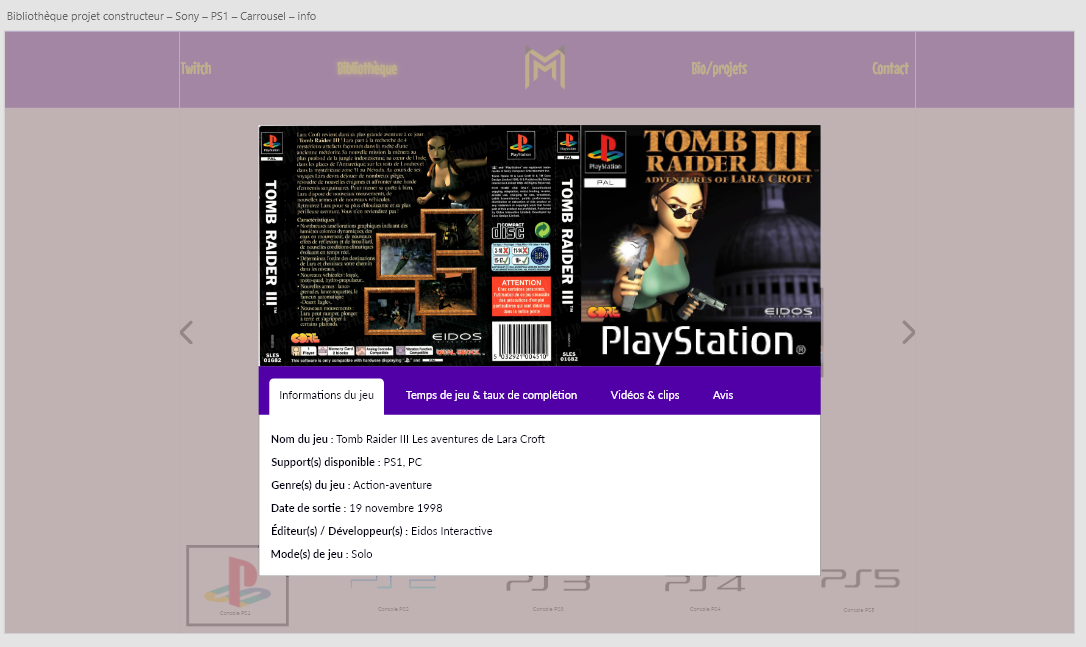
Maquette de la page d’une console dans la bibliothèque :



Maquette de la page de carrousel de jeu d’une console :

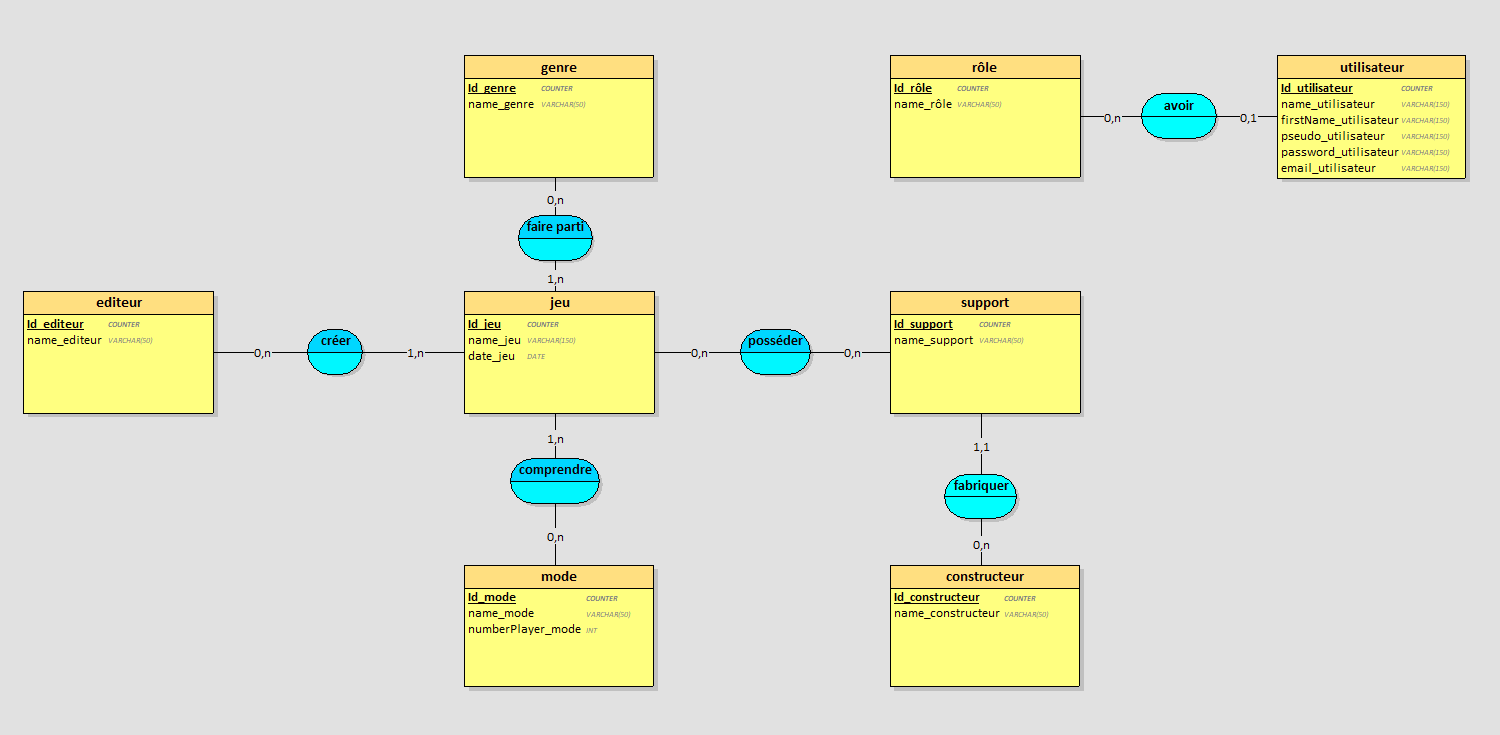


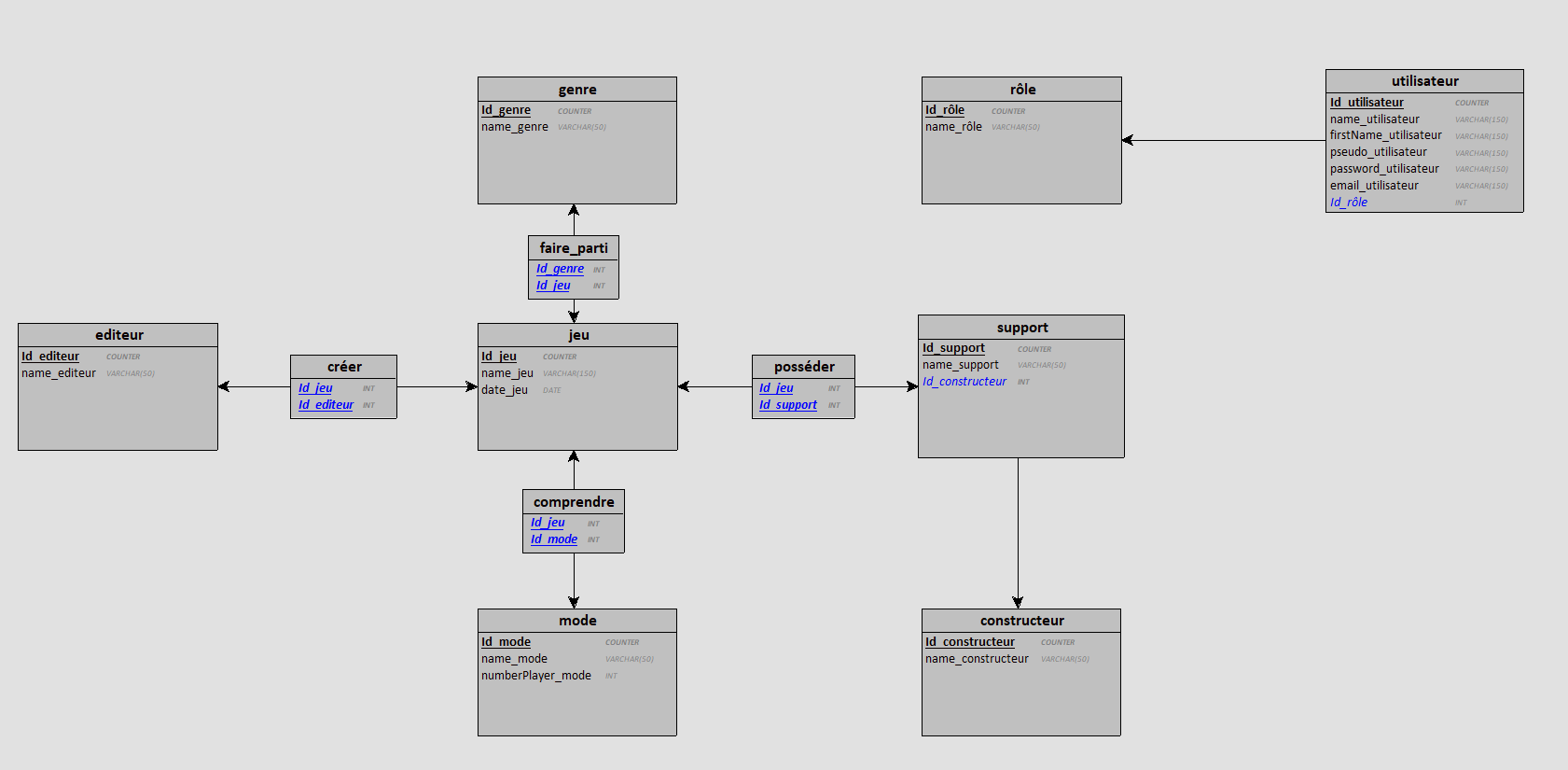
Maquette de la page d’information de jeu :



# 5 – CONCEPTION DE LA BASE DE DONNÉES

## 5.1– MCD / MLD





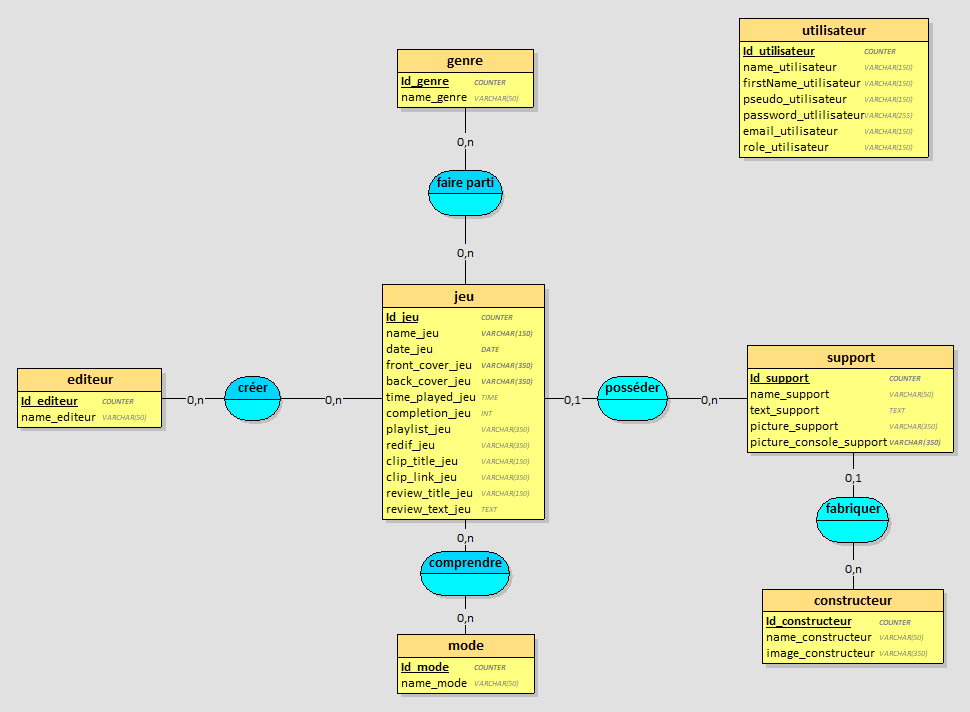
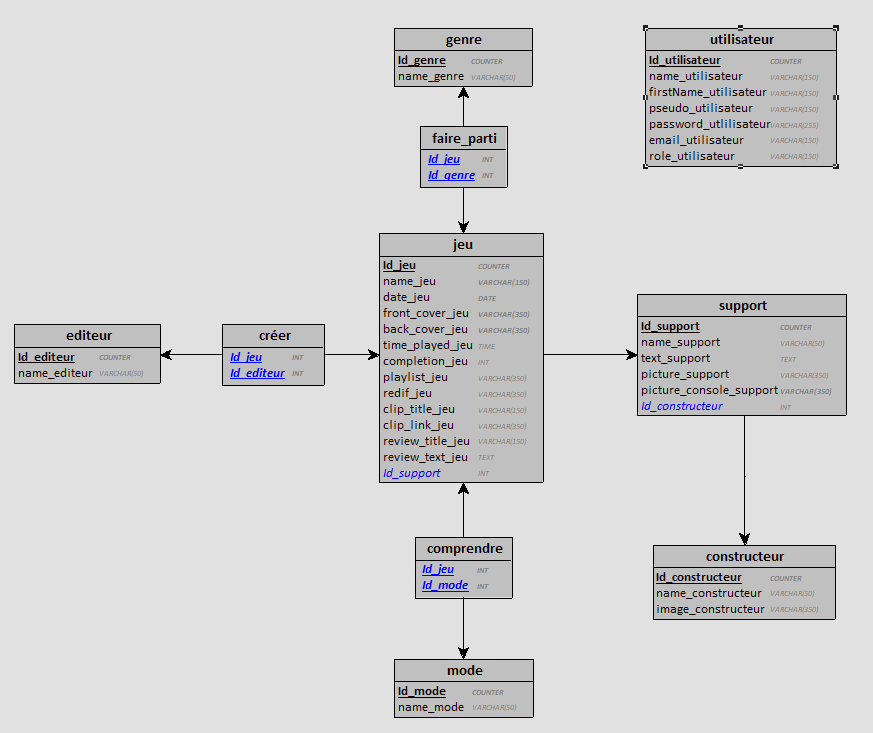
Étant donné que mon projet a beaucoup évolué depuis le début, la conception de la base de données a également changé.

Au départ j’étais parti sur l’idée de faire une table « jeu » qui pouvait être liée à plusieurs consoles et de ce fait ce serait la table d’association entre « jeu » et « support » (la table « posséder ») qui regrouperait la plupart des entrées. Cependant après réflexion je me suis dit que ce serait plus logique et pratique de faire passer le support directement dans la table « jeu » car les informations d’un jeu pourront être différentes selon le support (par exemple la date de sortie pourrait ne pas être la même, le temps de jeu ou même mon avis).

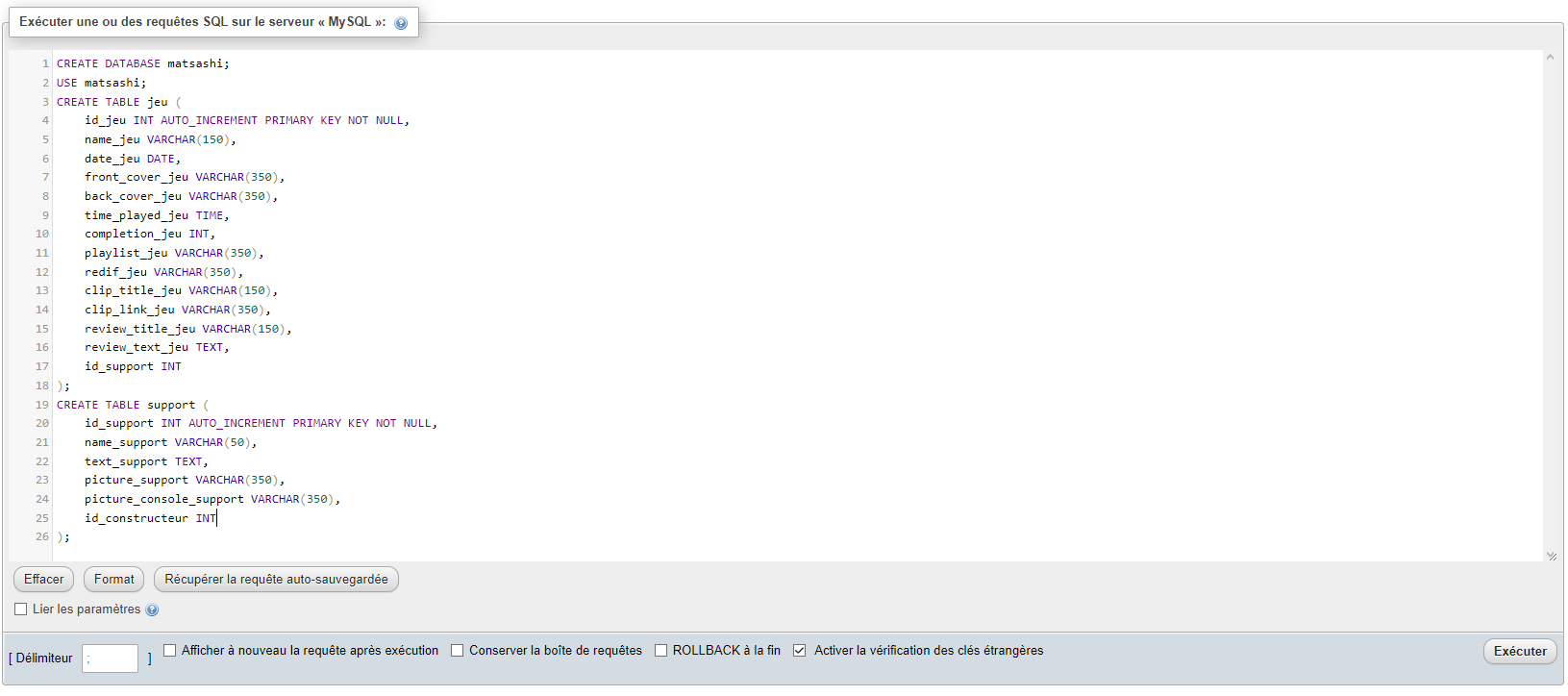
Autre modification : sur la 1ère version de la conception il était question de créer une table « rôle » afin d’attribuer aux utilisateurs la possibilité ou non de se connecter à l’espace administrateur. Mais finalement je n’ai pas prévu d’espace de connexion pour les utilisateurs basiques car ils n’ont aucune interaction possible avec le site, seulement les admins ont des accès de connexion. De ce fait la table « rôle » n’avait plus aucun intérêt et j’en ai fait un attribut dans la table « utilisateur ».

La table « mode » a eu différentes versions mais pour le moment je suis resté sur une version où chaque possibilité est listée directement dans la table sans utiliser la table d’association « comprendre », avec une cardinalité « 0,1 » entre « comprendre » et « jeu ». Étant donné qu’il y a peu de choix dans les modes et que mon code est prévu pour ajouter les jeux de cette façon je préfère le laisser comme tel afin d’éviter tout retard sur mes fonctionnalités, mais l’amélioration en « 0,n » afin d’utiliser la table d’association est prévu.

Les tables se sont étoffées d’attributs supplémentaires (principalement la table « jeu »), les cardinalités ont évolués afin d’éviter la suppression en cascade et voilà donc à quoi ressemble mes MCD / MLD :



## 5.2– Création de la base de données

En suivant les MCD / MLD, la création de la base de données a été plutôt rapide. Dans un 1er temps, vu que mon site était déjà hébergé sur LWS j’avais une base de données déjà créée (vide cependant). Je n’ai eu « que » les tables et les attributs à rajouter :

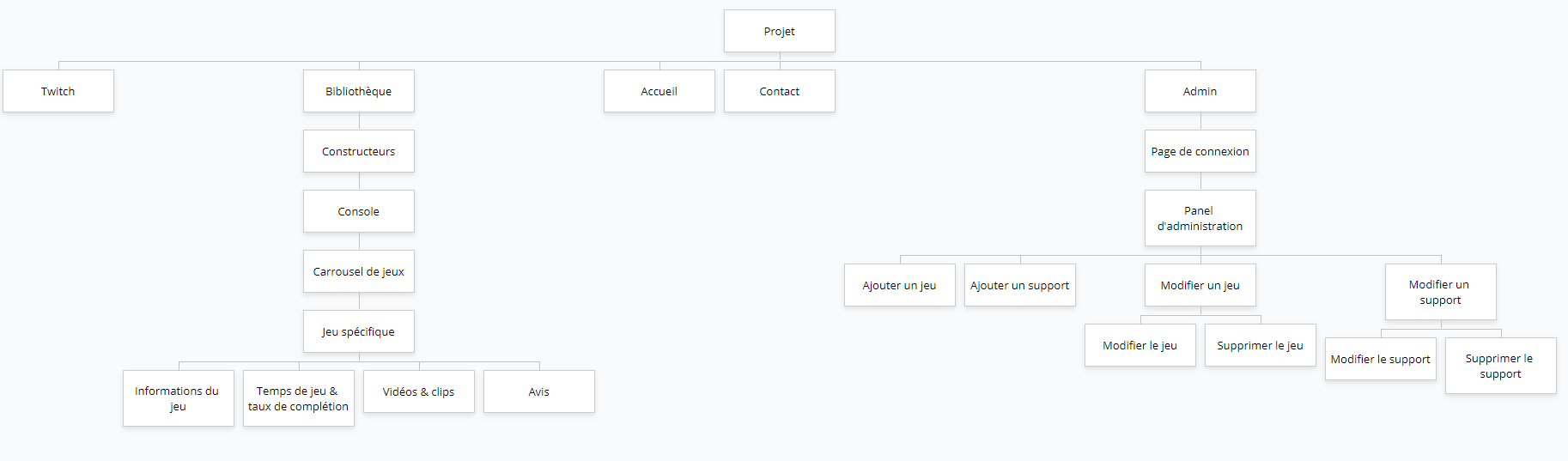
NB : J’ai rajouté la ligne de création de la base de données pour l’exemple, mais elle était déjà créée au moment de la structuration.

Après que les tables et les attributs aient été ajoutés, il a fallu également faire les clés étrangères afin de lier les tables entre elles :



La base de données était enfin prête !

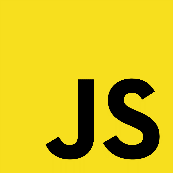
# 6 – ARBORESCENCE



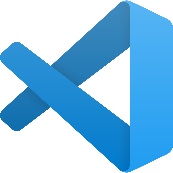
# 7 – OUTILS UTILISÉS



HTML 5 / CSS 3 / Bootstrap qui ont été utilisé pour la structure et le style des pages, et PHP qui est le langage de développement du projet.



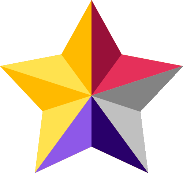
JavaScript qui a été utilisé pour mettre en surbrillance le lien de la page sur laquelle on se trouve dans le header.



VisualStudioCode qui est l’environnement de développement intégré (IDE) utilisé pour coder le projet.



Adobe Xd pour le maquettage du projet, version PC et version mobile.



StarUML et Looping qui ont été utilisé pour la conception (diagrammes et MCD / MLD).



WampServer qui a permis de créer un serveur local afin de faire les tests du projet avant la mise en prod, et phpMyAdmin qui contient la base de données.





LWS qui est le fournisseur de serveur sur lequel le projet a été mis en ligne, et Filezilla qui permet de transférer les fichiers directement sur le serveur via ftp.



PHP Mailer qui est l’outil permettant de gérer l’envoi de mail via formulaire sous PHP.



GlooMaps qui est un outil en ligne entièrement gratuit permettant de faire l’arborescence du projet, et Twitch Developers qui donne les outils nécessaires pour l’intégration du lecteur Twitch.



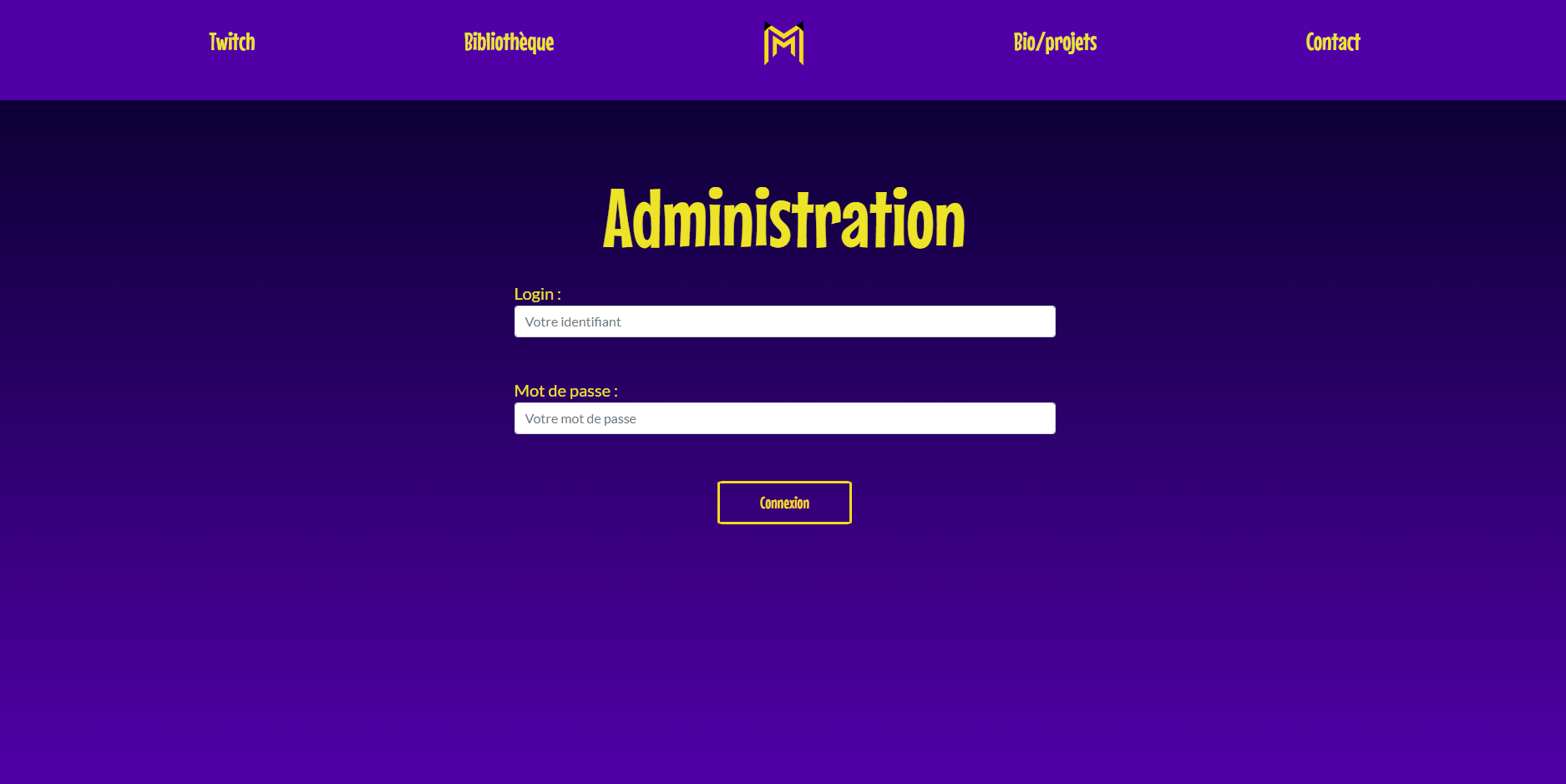
Et enfin GitHub qui a servi à sauvegarder le projet en ligne afin de pouvoir le récupérer sur différents postes de travail.

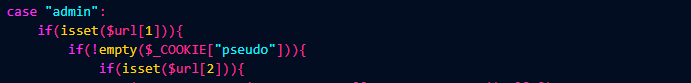
# 8 – INTÉGRATIONS DES FONCTIONNALITÉS

## 8.1– Page de connexion

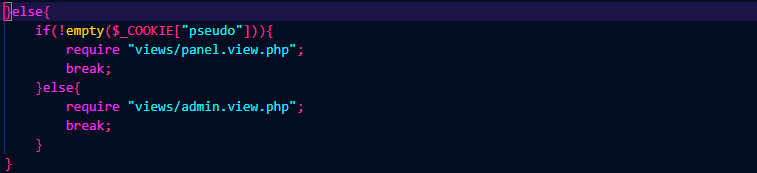
L’une des fonctionnalités clés du projet est la consultation de jeux, dans la bibliothèque, avec plusieurs informations différentes sur ces derniers ainsi que mes impressions et/ou des liens vers des moments particuliers lorsque j’y ai joué.

Cependant pour avoir cette fonctionnalité et étant donné que ma base de données est structurée de façon à renseigner le support de jeu pour le créer, il faut auparavant faire la page d’ajout de support, mais également faire toute la partie « administration » du site qui sera nécessaire afin seulement certaines personnes sélectionnées puissent faire les modifications sur le site et dans la base de données.

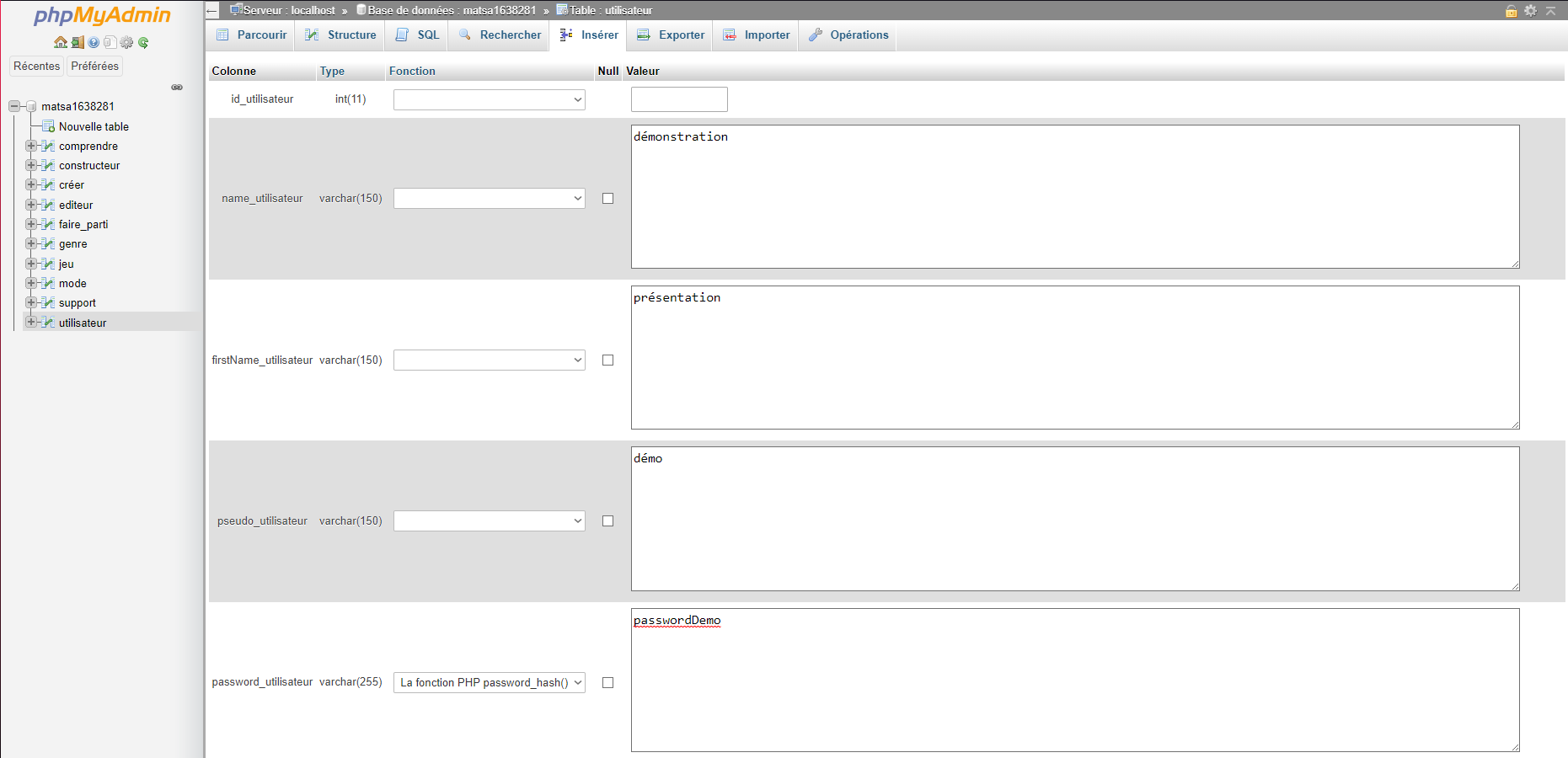


Tout d’abord cette page n’est accessible qu’en rentrant une adresse spécifique dans la barre d’adresse, aucun lien sur le site n’y emmène. Pour y accéder il faut renseigner l’adresse « matsashi.fr/admin » et ensuite c’est le router de la page index.php qui va s’occuper de rediriger l’utilisateur selon certaines conditions. Il va en premier lieu vérifier s’il y a un autre $\_GET[‘page’] dans l’adresse derrière le « admin » et si c’est le cas il va vérifier que les cookies de session existent bien (afin d’éviter que des personnes non connectées puissent accéder au panneau d’administration). Enfin si la personne est déjà connectée et possède des cookies, le router va vérifier s’il existe un deuxième $\_GET[‘page’] derrière le premier, ce dernier étant le chemin le plus long possible dans la partie admin.

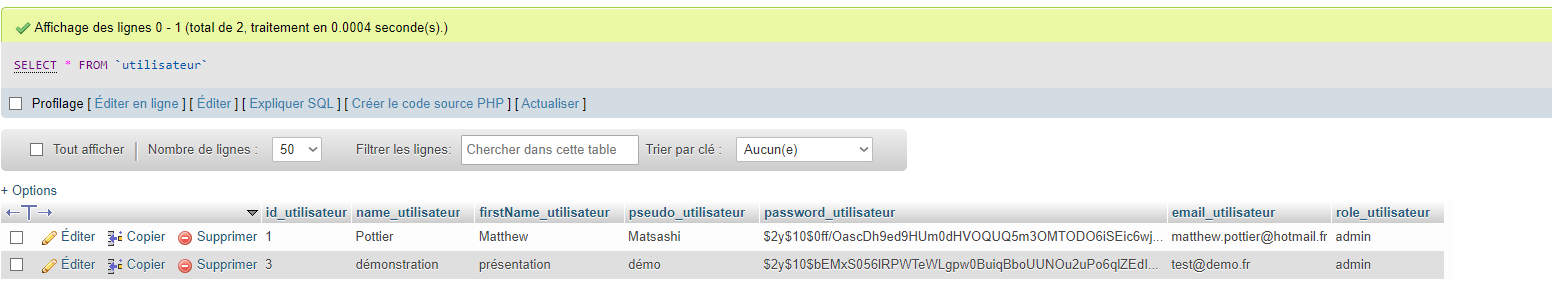
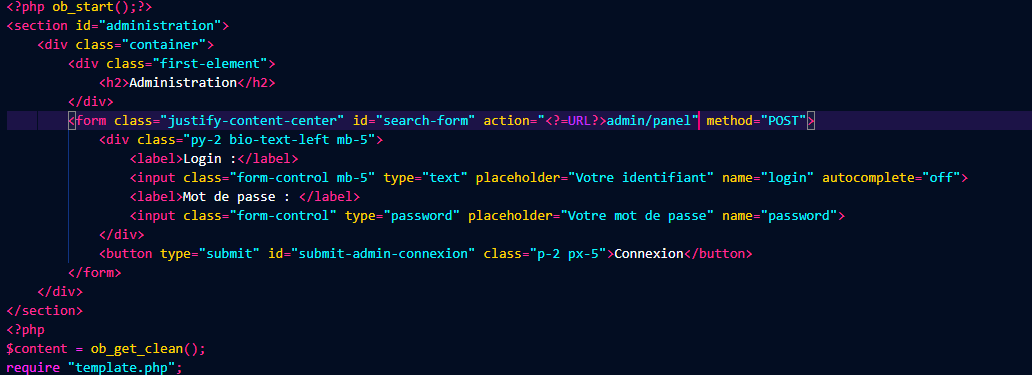
Dans le cas où aucune de ces conditions n’est remplie, le router va faire une dernière vérification : il va voir si des cookies de connexion sont présents afin que si la personne s’est déjà connectée elle ne soit pas envoyée sur la page de connexion mais directement sur le panneau d’administration. Dans le cas contraire elle sera bien redirigée vers la page de connexion.

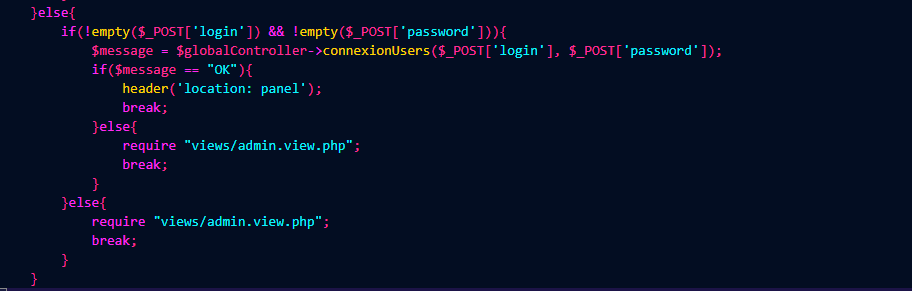


Lorsqu’un utilisateur arrive sur la page de connexion, il doit remplir le formulaire avec ses identifiants précédemment fournis. Ces identifiants de connexion sont créés directement dans la base de données, aucune page ne permet la création d’utilisateur pour le moment. Lors de la création de ces identifiants, le mot de passe est crypté via Bcrypt grâce à une fonction de PHPMyAdmin qui s’appelle « La fonction PHP password\_hash() » (fonction disponible que depuis la mise à jour 5.1 de PHPMyAdmin).

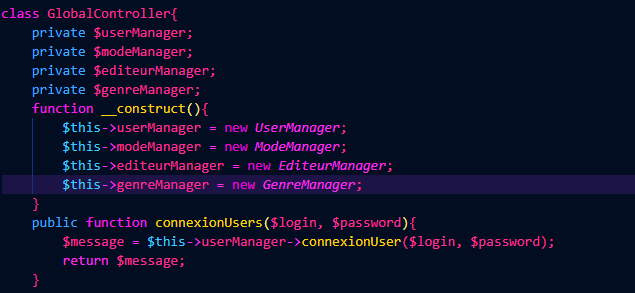


Une fois la requête d’insertion envoyée et faite, le mot de passe ressort bien en crypté dans la base de données.

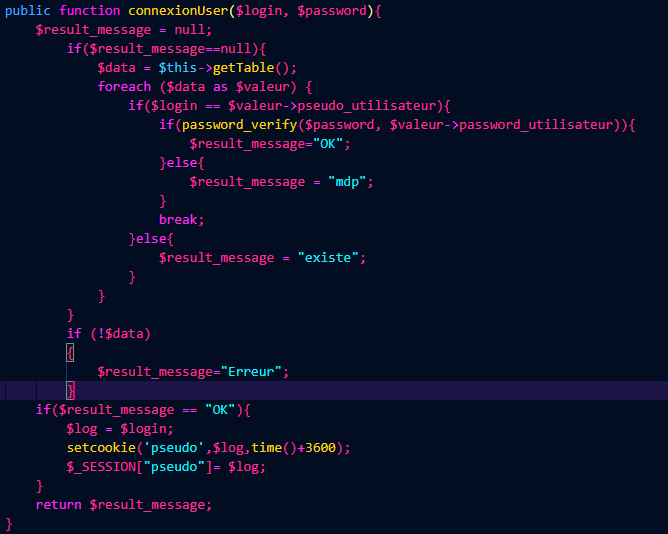
Du côté du code le traitement du formulaire va se faire avec le router et des fonctions, il n’y a pas de page dédiée au traitement. L’action du formulaire envoie donc sur la route « admin/panel » avec une méthode POST, le router va diriger l’action vers une condition où le POST existe et les vérifications d’identifiants vont être faites via le Global Controller.



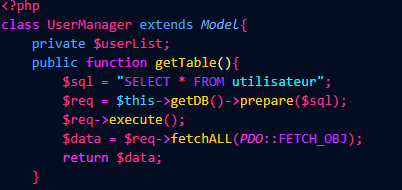
La fonction connexionUsers() va récupérer le POST du login et du mot de passe puis les transférer dans la fonction connexionUser() du User Manager pour être traité.



UserManager :

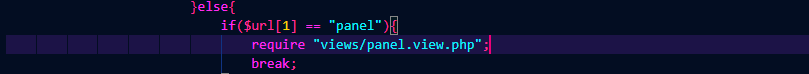


Dans la fonction connexionUser() de l’UserManager, cette dernière va tout d’abord récupérer la table « utilisateur » de la base de données via la fonction getTable() qui se trouve dans le UserManager.



A partir de là nous allons vérifier pour chaque itération de la table « utilisateur » si le login transmis via le POST[‘login’] correspond au pseudo enregistré dans la base de données. Une fois que le POST[‘login’] et la valeur de la BDD sont identiques nous utilisons la fonction native de PHP password\_verify() qui permet de comparer un mot de passe crypté avec un mot de passe non crypté (dans le cas présent vérifier le POST[‘password’] avec celui de la table « utilisateur »). Si le mot de passe correspond on ajoute le message « OK » dans la variable $result\_message et sinon on y ajoute le message « mdp », dans les deux cas un break est fait ensuite afin de ne pas continuer à chercher un login identique. Si jamais aucune itération ne match le login du POST, un autre message est ajouté au $result\_message. Enfin si tout s’est bien passé et que le message dans la variable correspond à « OK » la fonction créer un cookie de connexion, une session puis retourne le message présent dans la variable.

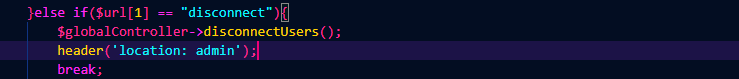
De retour au router, on vérifie une dernière fois que tout s’est bien passé en regardant le message résultant de la fonction. S’il corresponde à « OK » alors on utilise un header pour rediriger l’utilisateur vers le panneau d’administration, le router faisant alors de nouveau des vérifications de cookie et de $url[1].



A contrario si le message n’est pas « OK » le router va renvoyer l’utilisateur sur la page de connexion.

Après s’être connecté il fallait ensuite faire en sorte que l’admin puisse se déconnecter à tout moment. J’ai donc ajouté dans le html une condition afin que si des cookies de session existent, un message de bienvenu avec le pseudo ainsi qu’un lien de déconnexion apparaissent.



Quand tel est le cas le lien renvoie sur la route « admin/disconnect » qui va ensuite être géré par le router de l’index.

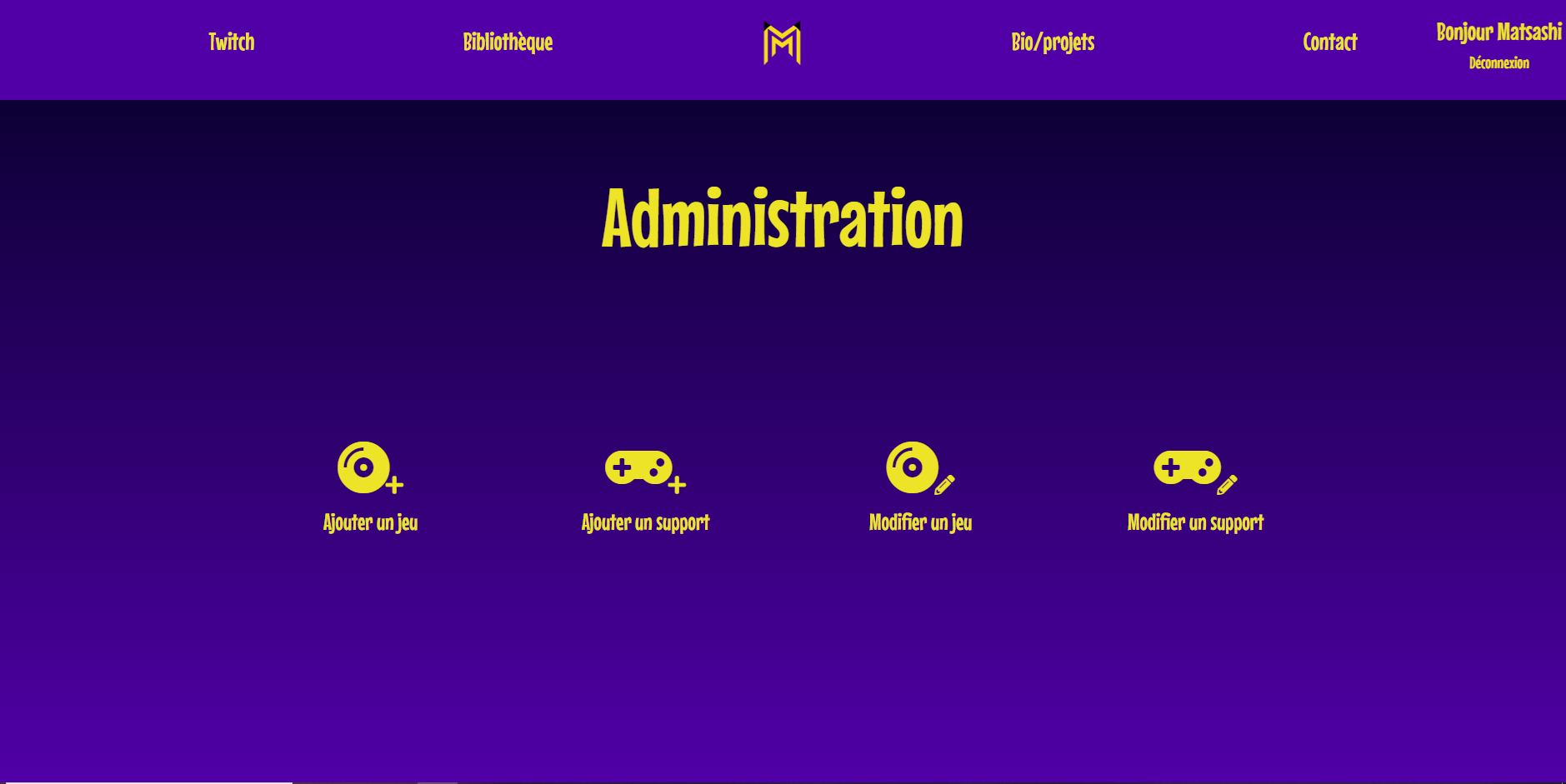
La fonction disconnectUsers() va ensuite prendre le relai via le GlobalController.



Lançant ainsi la fonction disconnectUser() du UserManager qui va détruire le cookie de session. Une fois ceci fait, le router renvoie l’utilisateur sur la page de connexion administrateur.

## 8.2– Ajout d’un support

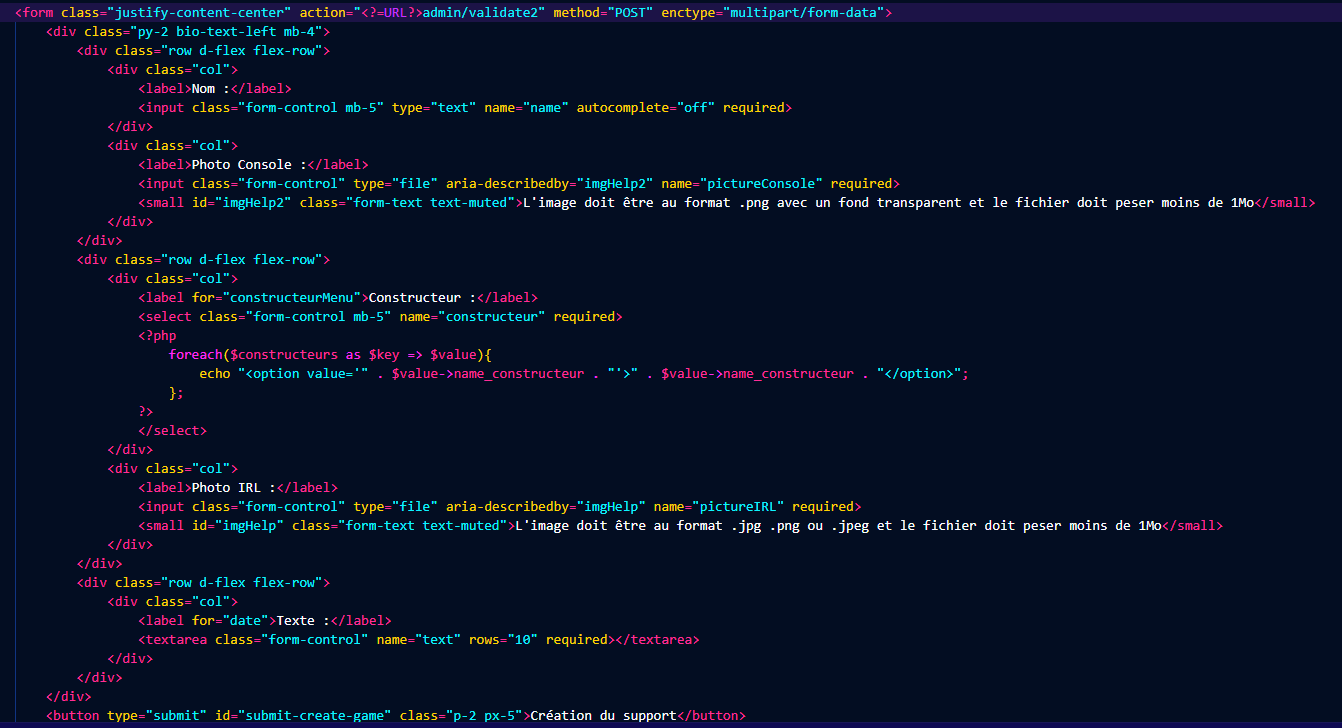
Maintenant que l’espace de connexion est fonctionnel l’utilisateur va pouvoir accéder au panneau d’administration, et de là il pourra aller vers la page d’ajout de support.



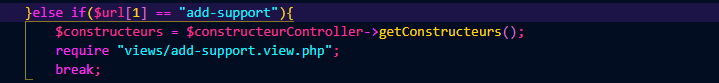


Une fois sur la page l’admin va devoir renseigner plusieurs champs afin de créer le support puis qu’il soit ajouté dans la base de données. Le formulaire comprend 5 champs :

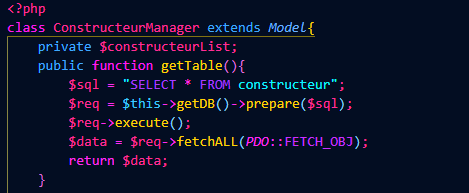
* Le nom du support
* La photo du support en format png avec fond transparent (qui sera ensuite utilisé dans la bibliothèque)
* Le nom du constructeur du support
* Une photo de moi représentant l’époque où j’ai eu cette console (qui sera également utilisé dans la bibliothèque par la suite)
* Le texte expliquant mon ressenti concernant cette console (lui aussi utilisé dans la bibliothèque)

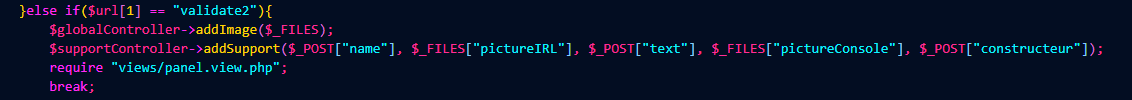
En ce qui concerne les champs où une photo est ajoutée, ce sont des input type files qui ont été utilisé et dans ce cas l’ajout de l’attribut « enctype=’multipart/form-data’ » est nécessaire afin de pouvoir gérer le transfert de fichiers via le formulaire.

Ces inputs sont également accompagnés d’attribut « aria-describedby » ainsi que de balises <small> afin de faire parvenir les restrictions d’envoi de fichier à l’utilisateur.

 Enfin le champ « constructeur » va récupérer les constructeurs présents dans la base de données pour les afficher sous forme d’option de la balise <select>. La récupération des données de la BDD se fait au préalable via le router qui lance la fonction getConstructeurs() du ConstructeurController.

Puis la fonction va lancer celle du ConstructeurManager prénommée getTable() qui va simplement récupérer les informations de la base de données et les retourner.



Une fois le formulaire complété et l’envoi des informations, ces dernières vont être transmises au router via l’action « < ?=URL ?>admin/validate2 » qui va ensuite lancer le processus d’ajout à la base de données.

Le router va tout d’abord lancer la fonction addImage() du GlobalController qui permet l’ajout des fichiers d’image dans les dossiers du projet.

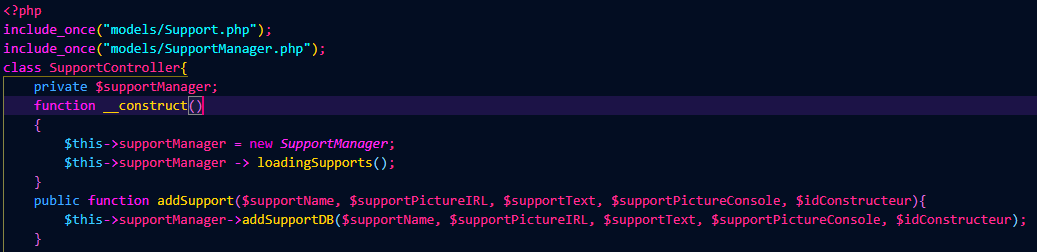
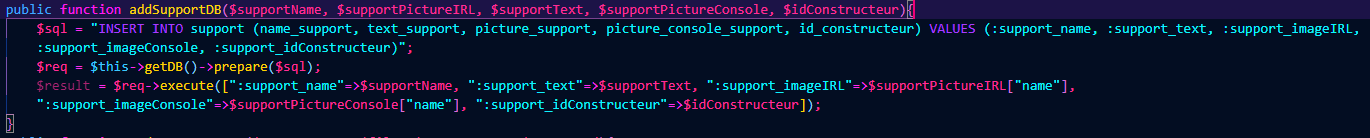


Cette fonction va tout d’abord vérifier si les champs des inputs « pictureIRL » et « pictureConsole » ne sont pas vides. Si tel est le cas la fonction renverra une exception qui stoppera tout le processus d’ajout, le router étant dans un try qui est suivi par un catch permettant de réceptionner les erreurs.

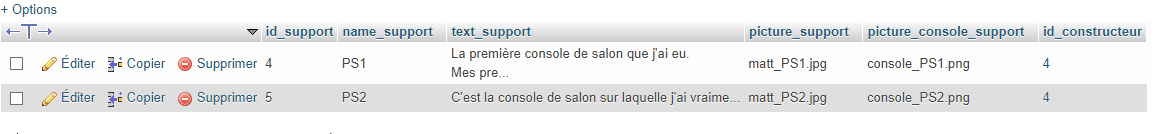
Dans le cas où les deux champs sont bien remplis, la fonction va stocker le nom de la 1ère photo dans une variable via la fonction native pathinfo() afin que par la suite nous puissions vérifier si l’extension du fichier correspond. Avant nous allons également vérifier si la taille du fichier ne dépasse pas 1Mo, dans le cas contraire nous attribuerons un string correspondant à l’erreur dans la variable $error\_message.

Si aucune erreur n’est à déplorer pendant les vérifications, la fonction va ensuite utiliser move\_upload\_file() qui est également une fonction native de PHP qui permet de sauvegarder le fichier envoyé dans les dossiers du projet (en précisant le nom du fichier en 1ère variable et le chemin du dossier dans le 2nd). Après cet ajout de fichier la fonction va de nouveau refaire toutes les vérifications précédentes pour le 2nd fichier envoyé et l’ajouter également aux dossiers du projet s’il correspond aux restrictions. Enfin la fonction retourne la variable $error\_message afin de la traiter plus tard.

De retour au router désormais c’est la fonction addSupport() du SupportController qui va être lancée afin de cette fois ajouter toutes les informations transmises dans la base de données (jusque là les ajouts n’étaient que locaux).

La fonction va directement lancer la fonction addSupportDB() du SupportManager.

La 1ère ligne exécutée va être l’attribution de la commande SQL qui va être envoyée à la BDD à un variable nommée $sql. L’attribution à cette variable va permettre par la suite d’utiliser la fonction getDB()->prepare() qui est héritée du Model dont le SupportManager est l’extension, cette fonction étant nécessaire pour éviter toute injection SQL dans la base de données via les inputs. Une fois la préparation de la requête effectuée, l’exécution de la requête peut être faite et ainsi les informations sont insérées dans la table « support ».

 Une fois l’ajout du support terminé, l’utilisateur est ensuite redirigé vers le panneau d’administration où il pourra continuer les ajouts nécessaires ou faire des modifications sur ceux existant.

## 8.3– Intégration Twitch et Responsive