Documentation Site Web Ligue 1

Sommaire

[Documentation Site Web Ligue 1 1](#_Toc164933453)

[Sommaire 1](#_Toc164933454)

[1. Introduction 3](#_Toc164933455)

[1.1. Différence entre Frontend et backend 3](#_Toc164933456)

[2. Prérequis 4](#_Toc164933457)

[3. Installation et configuration 4](#_Toc164933458)

[4. Structure du projet 5](#_Toc164933459)

[4.1. Présentation rapide du modèle MVC 5](#_Toc164933460)

[4.2. Détails du dossier src 5](#_Toc164933461)

[4.2.1. Détails du dossier model 5](#_Toc164933462)

[4.2.1.1. La classe Article 5](#_Toc164933463)

[4.2.1.2. La classe Club 6](#_Toc164933464)

[4.2.1.3. La classe Commentary 7](#_Toc164933465)

[4.2.1.4. La classe User 9](#_Toc164933466)

[4.2.2. Détails du dossier view 10](#_Toc164933467)

[4.2.2.1. La classe V\_Acceuil 10](#_Toc164933468)

[4.2.2.2. La classe V\_Connexion 10](#_Toc164933469)

[4.2.2.3. La classe V\_inscription 11](#_Toc164933470)

[4.2.2.4. La classe V\_listeclub 11](#_Toc164933471)

[4.2.3. Details du dossier control 11](#_Toc164933472)

[4.2.3.1. La classe C\_Acceuil 11](#_Toc164933473)

[4.2.3.2. La classe C\_Connexion 12](#_Toc164933474)

[4.2.3.3. La classe C\_Inscription 12](#_Toc164933475)

[4.2.3.4. La classe C\_Liste\_Club 12](#_Toc164933476)

[4.2.4. Details du dossier script 13](#_Toc164933477)

[4.2.4.1. La classe checked\_form 13](#_Toc164933478)

[4.2.4.2. La classe validate 13](#_Toc164933479)

[4.2.4.3. La classe same\_password 14](#_Toc164933480)

[4.2.4.4. La classe before\_submit 14](#_Toc164933481)

[4.2.4.5. La classe selectALLClubs 15](#_Toc164933482)

[4.2.5. Détails du dossier img 15](#_Toc164933483)

[4.2.6. Détails du dossier Css 15](#_Toc164933484)

[5. Fonctionnement du site web 15](#_Toc164933485)

[6. Contact 16](#_Toc164933486)

[7. Conclusion 16](#_Toc164933487)

1. Introduction

Lors de ma 2ᵉ année de BTS SIO SLAM, nous avons étudié les langages Web (HTML, CSS, PHP, JavaScript), mais nous nous sommes particulièrement intéressés au côté backend de la programmation au travers d’un projet Web. Ce projet consiste à créer un site internet de foot sur la ligue 1, comportant différentes pages et interrogeant une Base de données PostgreSQL.

Nous avons travaillé sous le modèle MVC qui est une bonne façon de travailler, car elle permet de répartir les logiques de code dans différentes classes.

* 1. Différence entre Frontend et backend

Frontend : Imaginez que vous êtes dans un restaurant. Le frontend serait la salle de restaurant où vous vous asseyez, le menu que vous parcourez, et le serveur qui prend votre commande. En termes de développement web, le frontend est tout ce que l’utilisateur voit et avec quoi il interagit sur le site web. Cela comprend la conception, la mise en page, les couleurs, les boutons, les formulaires et tout autre élément visuel. Les technologies couramment utilisées dans le développement frontend comprennent HTML, CSS et JavaScript.

Backend : En revenant à notre analogie du restaurant, le backend serait la cuisine, où le chef prépare votre repas, et tous les processus qui se déroulent en coulisses pour s’assurer que votre repas arrive à votre table comme vous l’avez commandé. Dans le développement web, le backend fait référence à tout ce qui se passe en coulisses d’un site web. Cela comprend la gestion des bases de données, la logique métier (comme les calculs), la sécurité, l’authentification, et tout ce qui est nécessaire pour traiter les demandes de l’utilisateur et renvoyer les bonnes données. Les technologies couramment utilisées dans le développement backend comprennent Node.js, Ruby, Python, PHP, Java, et d’autres.

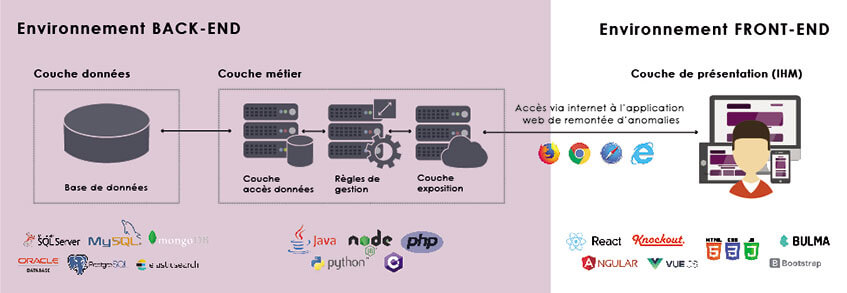


Figure 1 : schéma représentatif de la différence entre backend et frontend

La différence principale entre le frontend et le backend réside donc dans leur rôle et leur interaction avec l’utilisateur. Le frontend est tout ce que l’utilisateur voit et interagit, tandis que le backend est tout ce qui se passe en coulisses pour rendre cela possible.

1. Prérequis

Le site internet a été développé en utilisant les langages de programmation principaux du web, à savoir PHP, HTML et JavaScript. Actuellement, le site fonctionne en local car il n’est pas encore hébergé sur un serveur web. Pour la gestion des données, nous utilisons une base de données PostgreSQL, également en local.

Le code source du site est disponible sur GitHub. Pour y accéder, veuillez cliquer [ici](https://github.com/Paul-berne/Site-foot).

1. Installation et configuration

Pour pouvoir accéder au site web et profiter de celui-ci, je vais vous présenter comment pouvoir faire tourner un site web multi page sur une machine sur Microsoft Visual Studio, via une commande.

Vous devez avoir télécharger PHP préalablement puis avoir mis dans les variables d’environnement le path vers le dossier de PHP que vous venez de télécharger.

Ensuite il vous suffit de lancer Visual studio et d’ouvrir le projet, pour ensuite dans le terminal entrer cette commande : « PHP -S localhost :3000 -t src »

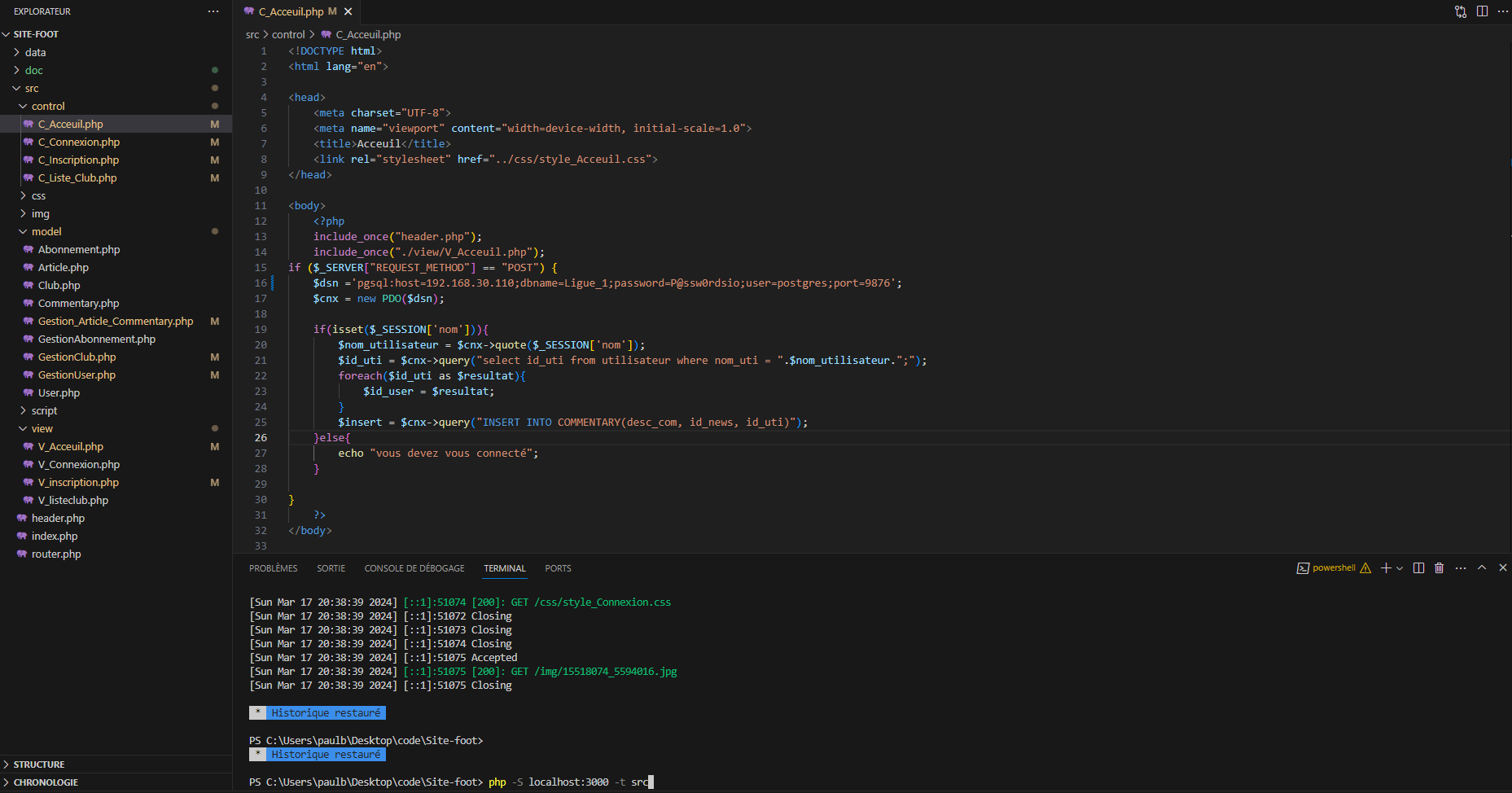


Figure 2 : Screenshot de Visual Studio

Ensuite vous pouvez accéder au site internet via le lien « localhost :3000 ».

1. Structure du projet

Maintenant nous verrons toute la structure du code derrière le site internet.

* 1. Présentation rapide du modèle MVC

Le modèle MVC est composé de 3 grands packages le modèle, la vue et le contrôleur, chacun a son rôle dans une application, voici leurs définitions respectives :

**Modèle** : Dans le modèle MVC, le “Modèle” représente les données et les règles métier de votre application. Il interagit avec la base de données pour créer, lire, mettre à jour et supprimer des informations.

**Vue** : La “Vue” est responsable de la présentation des données à l’utilisateur. Elle prend les données du modèle et les affiche d’une manière que l’utilisateur peut comprendre.

**Contrôleur** : Le “Contrôleur” agit comme un intermédiaire entre le modèle et la vue. Il traite les entrées de l’utilisateur, interagit avec le modèle pour obtenir ou modifier des données, et met à jour la vue en conséquence.

* 1. Détails du dossier src

Le dossier src comporte tout le code du projet et permet d’accéder à toute la logique et le fonctionnement du site internet.

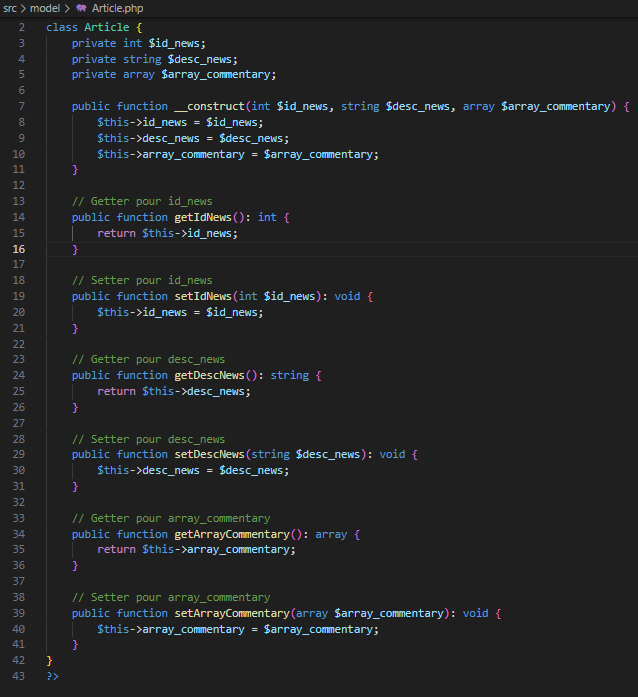
* + 1. Détails du dossier model

Le dossier model comme expliquer précédemment comporte toutes les classes Modèle, c’est-à-dire que les classes modèle vont interagir avec la base de données, il y’aura des classes qui ont des paramètres représentant des tables sur la base de données.

* + - 1. La classe Article

Elle représente un article avec des commentaires. Voici les détails :

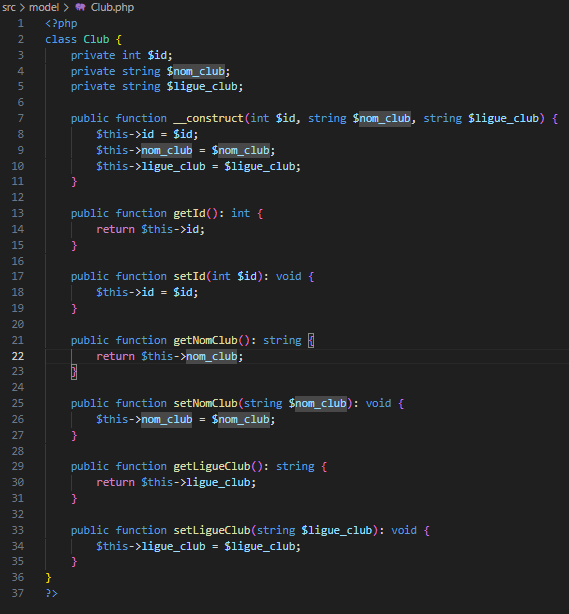
* Propriétés : La classe a trois propriétés privées :
  + $id\_news : Un entier qui représente l’identifiant unique de l’article.
  + $desc\_news : Une chaîne de caractères qui représente la description de l’article.
  + $array\_commentary : Un tableau qui contient les commentaires associés à l’article.
* Constructeur : Le constructeur de la classe prend trois paramètres ($id\_news, $desc\_news, $array\_commentary) qui sont utilisés pour initialiser les propriétés de l’objet lors de sa création.
* Getters : Les méthodes getIdNews(), getDescNews() et getArrayCommentary() sont des getters. Elles permettent d’accéder aux valeurs des propriétés privées de l’objet.
* Setters : Les méthodes setIdNews(), setDescNews() et setArrayCommentary() sont des setters. Elles permettent de modifier les valeurs des propriétés privées de l’objet.



* + - 1. La classe Club

Elle représente un club de football dans une application web. Voici les détails :

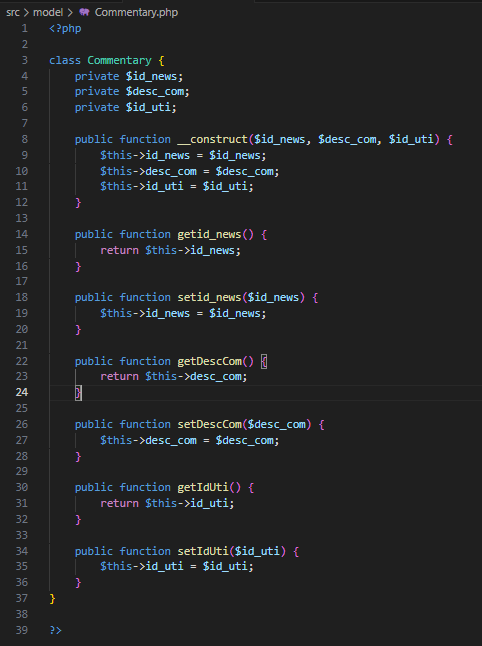
* Propriétés : La classe a trois propriétés privées :
  + $id : Un entier qui représente l’identifiant unique du club.
  + $nom\_club : Une chaîne de caractères qui représente le nom du club.
  + $ligue\_club : Une chaîne de caractères qui représente la ligue dans laquelle le club joue.
* Constructeur : Le constructeur de la classe prend trois paramètres ($id, $nom\_club, $ligue\_club) qui sont utilisés pour initialiser les propriétés de l’objet lors de sa création.
* Getters : Les méthodes getId(), getNomClub() et getLigueClub() sont des getters. Elles permettent d’accéder aux valeurs des propriétés privées de l’objet.
* Setters : Les méthodes setId(), setNomClub() et setLigueClub() sont des setters. Elles permettent de modifier les valeurs des propriétés privées de l’objet.



* + - 1. La classe Commentary

La classe s’appelle Commentary. Voici les détails :

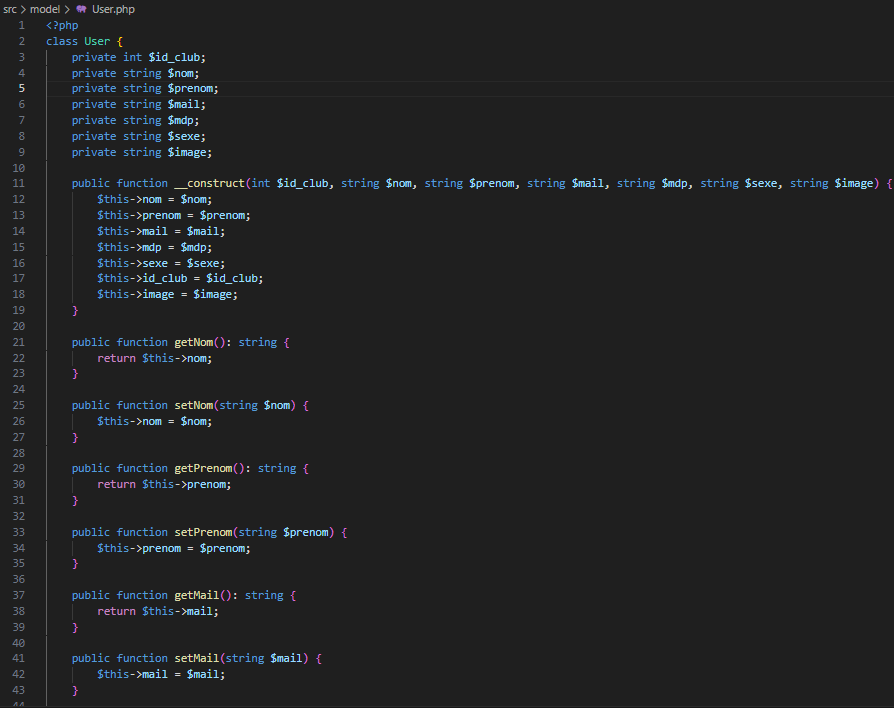
* Propriétés : La classe a trois propriétés privées :
  + $id\_news : Il s’agit de l’identifiant de l’article auquel le commentaire est associé.
  + $desc\_com : Il s’agit de la description du commentaire, c’est-à-dire le texte du commentaire lui-même.
  + $id\_uti : Il s’agit de l’identifiant de l’utilisateur qui a posté le commentaire.
* Constructeur : Le constructeur de la classe prend trois paramètres ($id\_news, $desc\_com, $id\_uti) qui sont utilisés pour initialiser les propriétés de l’objet lors de sa création.
* Getters : Les méthodes getid\_news(), getDescCom() et getIdUti() sont des getters. Elles permettent d’accéder aux valeurs des propriétés privées de l’objet.
* Setters : Les méthodes setid\_news(), setDescCom() et setIdUti() sont des setters. Elles permettent de modifier les valeurs des propriétés privées de l’objet.



* + - 1. La classe User

La classe s’appelle User. Voici les détails :

* Propriétés : La classe a sept propriétés privées :
  + $id\_club : Il s’agit de l’identifiant du club auquel l’utilisateur est associé.
  + $nom : Il s’agit du nom de l’utilisateur.
  + $prenom : Il s’agit du prénom de l’utilisateur.
  + $mail : Il s’agit de l’adresse mail de l’utilisateur.
  + $mdp : Il s’agit du mot de passe de l’utilisateur.
  + $sexe : Il s’agit du sexe de l’utilisateur.
  + $image : Il s’agit de l’image de profil de l’utilisateur.
* Constructeur : Le constructeur de la classe prend sept paramètres ($id\_club, $nom, $prenom, $mail, $mdp, $sexe, $image) qui sont utilisés pour initialiser les propriétés de l’objet lors de sa création.
* Getters : Les méthodes getNom(), getPrenom(), getMail(), getMdp(), getSexe(), getId\_club() et getImage() sont des getters. Elles permettent d’accéder aux valeurs des propriétés privées de l’objet.
* Setters : Les méthodes setNom(), setPrenom(), setMail(), setMdp(), setSexe(), setId\_club() et setImage() sont des setters. Elles permettent de modifier les valeurs des propriétés privées de l’objet



* + 1. Détails du dossier view

Le dossier View comporte les Classe PHP qui sont le frontend des pages du site. Elles sont là pour contenir le code de ce que l’utilisateur voit.

* + - 1. La classe V\_Acceuil

La classe V\_Acceuil est une vue qui vas contenir le contenu de la page d’accueil lorsqu’une personne se connecte.

Elle contient :

* Un paragraphe qui présente le site web et propose a l’utilisateur de se connecter ou de s’inscrire
* 4 articles les plus récents avec leurs commentaires respectifs

Depuis cette page nous pouvons donc accéder aux derniers articles ainsi nous pouvons aussi publier un commentaire.

* + - 1. La classe V\_Connexion

Cette classe V\_connexion est très simpliste, car elle contient simplement des zones de texte avec login et mot de passe pour que l’utilisateur se connecte avec un bouton se connecter.

* + - 1. La classe V\_inscription

Cette classe V\_inscription permet à un utilisateur de se créer un compte, en entrant ses informations personnelles, tel que :

* Son nom
* Son prénom
* Son adresse électronique
* Son mot de passe et la confirmation de celui-ci
* Son sexe
* Une image de profil
* Le championnat qui l’intéresse
* Son club favori

Pour ensuite la validé via le bouton « Validez »

* + - 1. La classe V\_listeclub

Cette classe V\_listeclub, est une page qui vas contenir tous les clubs présents dans la base de données sous forme d’un tableau avec une colonne nom du club et une colonne championnat.

Cette page est utile pour pouvoir s’informer sur les clubs présents dans quel championnat.

* + 1. Details du dossier control

Le dossier control comporte toutes les classes qui ont de la logique de traitement de donnée et qui utilisent les classes de vue.

* + - 1. La classe C\_Acceuil

La classe C\_Acceuil vas contenir la vue de la classe accueil et tout le traitement qu’il y a derrière :

* Inclut le fichier “header.php” qui contient le code pour l’en-tête du site.
* Inclut le fichier “V\_Acceuil.php” qui contient le code pour la vue d’accueil du site.
* Vérifie si la méthode de requête est POST, ce qui signifie qu’un formulaire a été soumis.
* Établit une connexion à une base de données PostgreSQL.
* Si une session est en cours et qu’un nom d’utilisateur est défini, le code fait plusieurs choses :
  + Il récupère l’identifiant de l’utilisateur à partir de la base de données.
  + Il insère un nouveau commentaire dans la base de données.
* Si aucune session n’est en cours, le code affiche un message indiquant à l’utilisateur qu’il doit se connecter.
  + - 1. La classe C\_Connexion

La classe C\_Connexion vas contenir la vue de la classe Connexion mais elle va avoir toute la logique derrière les interactions que fait l’utilisateur :

* Inclut le fichier de vue ainsi que le header
* récupère l’adresse électronique et le mot de passe du formulaire. Le mot de passe est haché avec MD5 pour des raisons de sécurité.
* exécute une requête SQL pour sélectionner l’utilisateur correspondant à l’adresse électronique et au mot de passe fournis.
* Si un utilisateur correspondant est trouvé (c’est-à-dire si le nombre de lignes retournées par la requête est 1), le code fait plusieurs choses :
  + Il récupère les informations de l’utilisateur de la base de données et les stocke dans des variables de session.
  + Il redirige l’utilisateur vers la page “Connexion.php”.
* Si aucun utilisateur correspondant n’est trouvé, le code affiche un message d’erreur.
  + - 1. La classe C\_Inscription

La classe C\_Inscription vas contenir la vue de la classe Inscription mais elle va avoir toute la logique derrière les interactions que fait l’utilisateur :

* Inclut la vue correspondante
* Vérifie si la méthode de requête est POST, ce qui signifie qu’un formulaire a été soumis.
* Si c’est le cas, le code fait plusieurs choses :
  + Établit une connexion à une base de données PostgreSQL.
  + Récupère les informations du formulaire d’inscription.
  + Si une image a été téléchargée, le code la déplace dans un répertoire spécifique et enregistre le chemin de l’image.
  + Crée un nouvel objet User avec les informations du formulaire et l’image téléchargée.
  + Envoie l’objet User à la base de données.
  + Affiche un message indiquant que la création du compte a réussi.
* Si la méthode de requête n’est pas POST, le code inclut le fichier “V\_inscription.php” qui contient probablement le code pour la vue de la page d’inscription.
  + - 1. La classe C\_Liste\_Club

La classe C\_Liste\_Club vas contenir la vue de la classe ListeClub mais elle va avoir toute la logique derrière les interactions que fait l’utilisateur :

* Inclus la vue correspondante ainsi que le header pour permettre a l’utilisateur de naviguer entre les pages
* Établit une connexion à une base de données PostgreSQL.
* Crée un nouvel objet GestionClub est utilisé pour gérer les opérations liées aux clubs.
  + Appelle la méthode getLIsteClub() de l’objet GestionClub pour obtenir une liste de clubs.

Cette page ne contient pas énormément de code mais permet simplement de montrer a l’utilisateur les clubs existant avec leurs championnats.

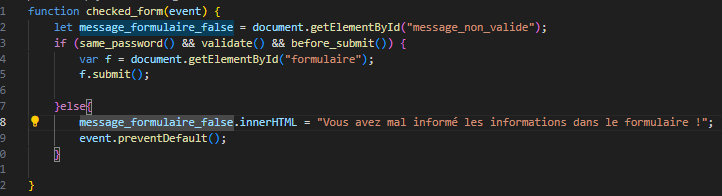
* + 1. Details du dossier script

Le dossier script contient un fichier script.js qui lui vas contenir du code javascript que le code du site va interroger.

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Ce langage permet de rendre vivant les pages et de les rendre dynamique et agréable pour l’utilisateur.

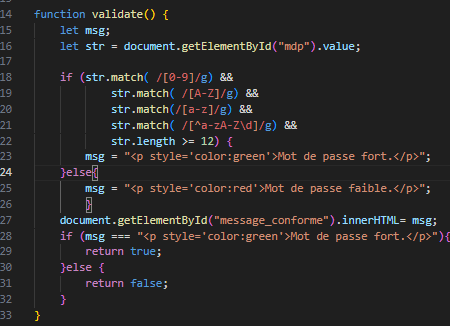
Le fichier comporte 5 classes.

* + - 1. La classe checked\_form



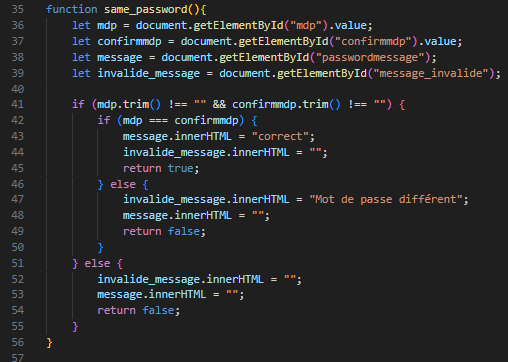
Cette fonction est appelée lors de la soumission du formulaire. Elle vérifie si les mots de passe correspondent, si le mot de passe est fort et si tous les champs nécessaires sont remplis en utilisant les fonctions same\_password(), validate() et before\_submit(). Si toutes ces conditions sont remplies, le formulaire est soumis. Sinon, un message d’erreur est affiché et la soumission du formulaire est empêchée.

* + - 1. La classe validate



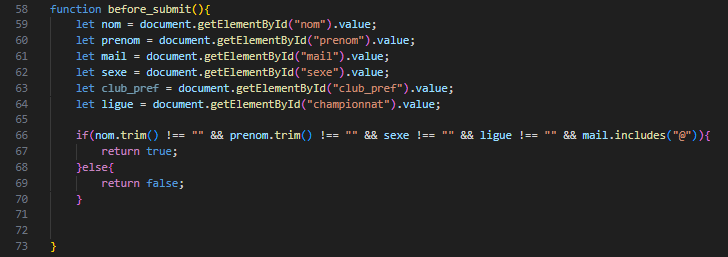
Cette fonction vérifie la force du mot de passe. Elle vérifie si le mot de passe contient au moins un chiffre, une lettre majuscule, une lettre minuscule, un caractère spécial et s’il a une longueur d’au moins 12 caractères. Si le mot de passe est fort, un message vert est affiché et la fonction retourne true. Sinon, un message rouge est affiché et la fonction retourne false.

* + - 1. La classe same\_password



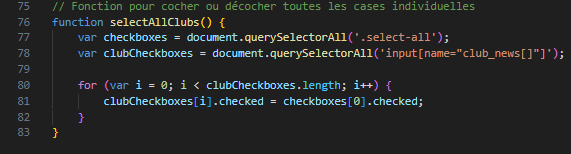
Cette fonction vérifie si le mot de passe et la confirmation du mot de passe correspondent. Si c’est le cas, un message “correct” est affiché et la fonction retourne true. Sinon, un message d’erreur est affiché et la fonction retourne false.

* + - 1. La classe before\_submit



Cette fonction vérifie si tous les champs nécessaires sont remplis et si l’adresse électronique contient un “@”. Si c’est le cas, la fonction retourne true. Sinon, elle retourne false.

* + - 1. La classe selectALLClubs



Cette fonction est utilisée pour cocher ou décocher toutes les cases de sélection des clubs. Si la case “Sélectionner tout” est cochée, toutes les cases individuelles sont cochées. Si la case “Sélectionner tout” est décochée, toutes les cases individuelles sont décochées.

* + 1. Détails du dossier img

Le dossier img contient toutes les photos de profil des utilisateurs avec pour nom à chaque fois :

*Nomutilisateur*\_*prenomutilisateur*\_*nomdelimageimporter*.jpg

* + 1. Détails du dossier Css

Le Css est utilisé en Web pour uniquement faire du visuel, Frontend, ce langage nous permet de mettre des couleurs sur les pages, gérer les positions des paragraphes, gérer le responsives.

Il est la pour rendre agréable la visite du site web pour l’utilisateur.

1. Fonctionnement du site web

Pour généraliser le Site web voici un résumé de toutes les fonctionnalités que nous offre le site web :

Lorsqu’un utilisateur visite le site, il peut s’inscrire en fournissant des informations telles que son nom, son prénom, son adresse électronique, son mot de passe, son sexe, son club préféré et une image de profil. Ces informations sont stockées dans un objet User et envoyées à la base de données.

Une fois inscrit, l’utilisateur peut se connecter. Le site vérifie les informations de connexion en les comparant aux données stockées dans la base de données. Si les informations sont correctes, l’utilisateur est connecté et une session est créée.

Le site propose également une liste de clubs que l’utilisateur peut consulter. Chaque club est représenté par un objet Club, qui contient des informations telles que l’identifiant du club, le nom du club et la ligue du club.

1. Contact

Si vous souhaitez me contacter :

Adresse électronique : [Paulberne@gmail.com](mailto:Paulberne@gmail.com)

Numéro de téléphone : 07 83 51 97 15

Ou via LinkedIn : <https://www.linkedin.com/in/paul-berne/>

1. Conclusion

Ce projet m’a permis d’approfondir mes connaissances en termes de Web car j’ai pu exploiter un grande partie des fonctionnalités qui s’offre a moi dans ce domaine.

J’ai pu intéragir avec une base de donnée, mettre en place un modèle MVC et prendre conscience des enjeux de la cybersécurité dans le Web avec l’importance de la place des vérifications dans celui-ci.

La base de données nous été fournis préalablement et en utilisant les préférences de l’utilisateur dans ses choix de clubs préférés.