**Projet réalisé par Sextius Sullivan, Vasone Antoine et Dauw Marc**

**Documentation Technique**

**Table des matières :**

1. Arborescence Utilisateur (User, Médecin, Administrateur) .........................................
2. Explication des dossiers et fichiers (.php)…...................................................................
3. FORM.............................................................................................................................

* Ajout\_medecin\_rdv\_form.php …......................................................................
* Annulation\_form.php.........................................................................................
* Connexion\_form.php.........................................................................................
* Deconnexion\_form.php......................................................................................
* Inscription\_admin.php …....................................................................................
* Inscription\_form.php …......................................................................................
* inscription\_medecin.php …................................................................................
* modification\_form.php …...................................................................................
* Password\_form.php ….......................................................................................
* prise\_rendez\_vous.php …..................................................................................
* Verify.php ….......................................................................................................

1. Include............................................................................................................................

* Footer.php …......................................................................................................
* Head.php …........................................................................................................
* Header.php ….....................................................................................................
* Import.php …......................................................................................................
* Page.php ….........................................................................................................
* Script.php ….......................................................................................................

1. Manager........................................................................................................................

* Explication technique du Model MCV...............................................................
* Manager.php.....................................................................................................

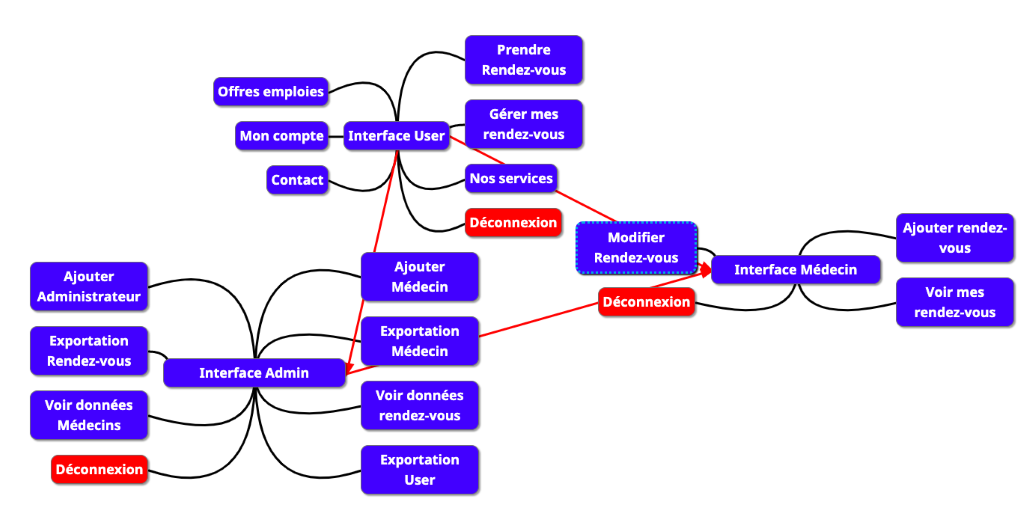
1. Modal............................................................................................................................
2. Model............................................................................................................................
3. Traitement …................................................................................................................

* Traitement suppression utilisateur….................................................................
* Traitement exportation de données..................................................................

1. View …............................................................................................................................

* Interface Admin (Index.php) …...........................................................................
* Interface Médecin (Index.php) ….......................................................................
* Interface User (Index.php) …..............................................................................

1. Conclusion.....................................................................................................................
2. **Arborescence Utilisateur (User, Médecin, Administrateur)**



1. **Explication technique des dossiers et fichiers du site**

Dans le projet on accède aux pages du site dans le dossier *View* en allant dans le fichier qui sont situé à la racine du projet ce qui permet à l’utilisateur d’accéder directement au site.

Tous les traitements se font dans deux dossiers, le dossier FORM et le dossier traitement. Tous les modals sont dans le dossiers *Modal* ce qui permet d’afficher une multitude de fenêtres avec la bibliothèque Bootstrap. Toutes les importations permettant le style Bootstrap du Site sont faites dans le dossier *Include*.

Toutes les images du site sont contenues dans le dossier *Img*, et le manager qui permet toutes les fonctionnalités précises du site sont dans le dossier *manager*

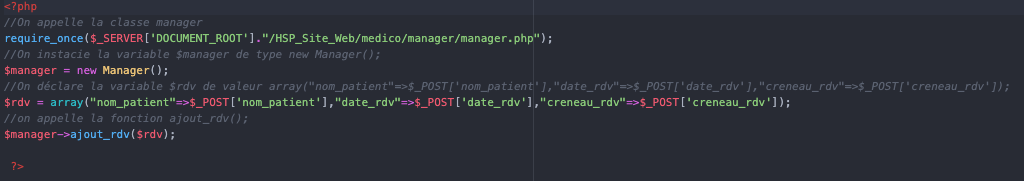
Les attributs, les getter et setter sont contenus dans le dossier *Model*.

Tous ce qui permet l’envoie de mail est contenu dans le dossier *Vendor* en utilisant PHP Mailer.

1. **Dossier Form**

Le dossier *Form* est l’un des dossiers les plus importants du site il permet le traitement de tous les formulaires mais aussi d’appeler les méthodes de la classe situées dans le manager.

**Ajout\_medecin\_rdv\_form.php**



Ce fichier permet d’appeler la classe *manager.php* dans le dossier *Manager* on instancie la classe afin d’appeler la fonction *ajout\_rdv* qui permet l’ajout de rendez-vous dans l’interface Médecin on stocke les POST envoyer depuis le formulaire d’ajout de rendez-vous dans un tableau que l’on appelle *$rdv*, puis on traite ce tableau de données en paramètre dans la fonction *ajout\_rdv*.

**Annulation\_form**



On appelle la classe Manager avec la fonction PHP *require\_once* puis on déclare la variable manager de type new Manager() on exécute la requête SQL qui permet d’afficher les rendez-vous par rapport à l’Id de l’utilisateur stocké dans la variable *$my\_key* qui est une variable de type *$\_GET.*

**Connexion\_form**

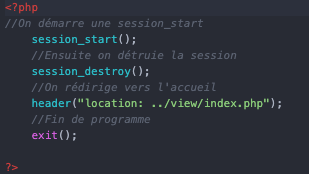


On échappe directement les caractères avec la fonction *real\_escape\_string* qui permet d’échapper les valeurs de type *$\_POST* puis on les crypte avec la méthode *md5()* ensuite on exécute la requête SQL qui permet d’afficher les utilisateurs en fonction de leurs mails.

Si le tableau de la requête SQL existe alors on déclare les variables *$id, $nom, $prenom, $addr, $ville, $mail, $mutuelle, $verified, $medecin et $admin* puis on vérifie la valeur de la variable *$verified* si elle est égale à 1 alors on stocke la valeur de toutes les variables du tableau de la requête SQL dans une session

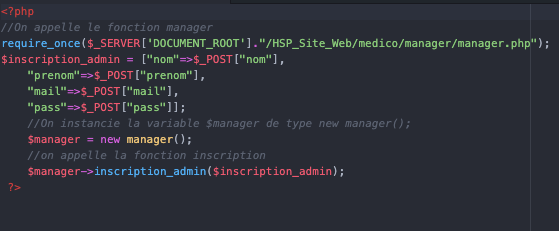
Ensuite on vérifie si l’utilisateur en question est un médecin, un admin ou un patient. Si cet utilisateur est un patient alors on redirige la personne en question, vers le profil médecin, le profil admin ou le profil utilisateur, si la condition globale n’est pas vérifiée alors on redirige vers la page avec une clé puis on affiche une erreur.

**Deconnexion\_form**



On démarre une session puis on la détruit si elle existe, puis on redirige vers la page d’accueil

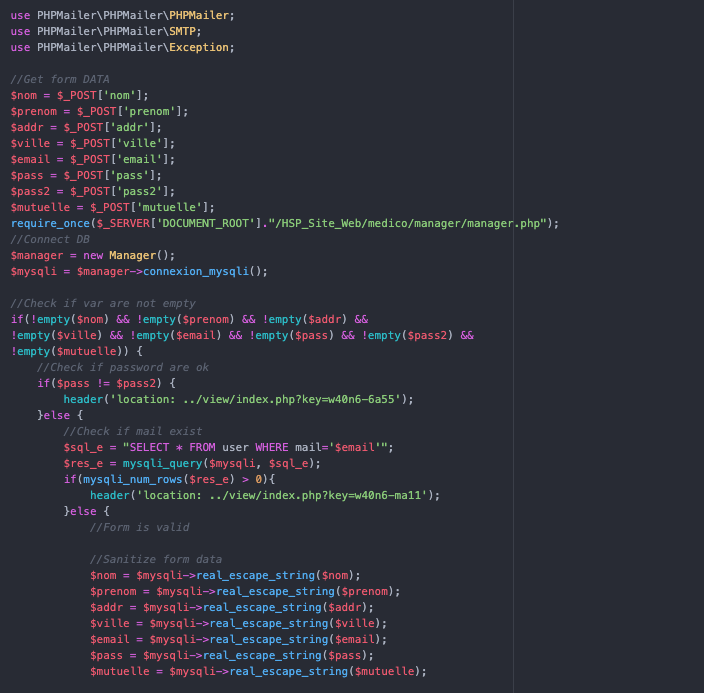
**Inscription\_admin.php**



On appelle la classe manager, puis déclare la variable *$inscription\_admin* de type *array()*, on déclare la variable *$manager* de type new Manager () ;

Puis on appelle depuis la classe manager la fonction *inscription\_admin*

**Inscription\_form.php**

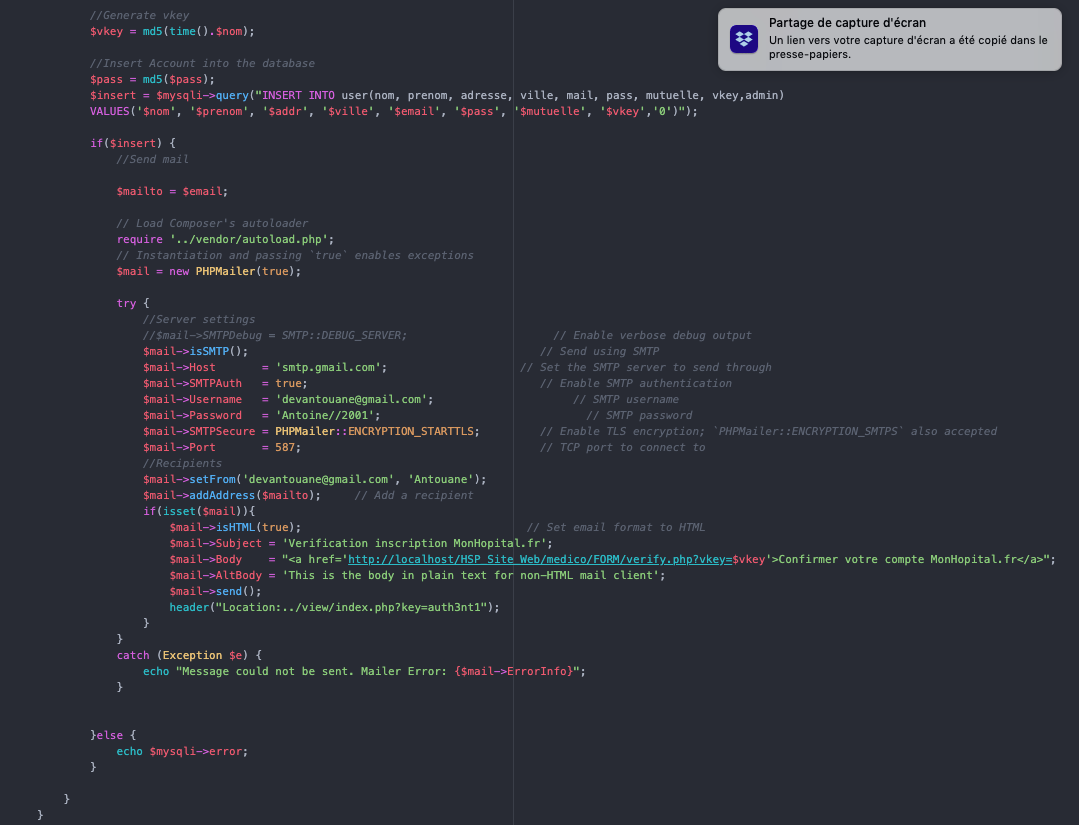


On appelle avec la méthode *use PHP* les dossiers PHP Mailer SMTP et EXception, puis on déclare les variables *$nom, $prenom, $addr, $ville, $email, $pass, $pass2* puis on appelle la classe manager à la racine du projet.

Si les variables ne sont pas vides alors on vérifie si les deux mots de passe sont corrects lors de l’inscription puis on redirige vers la page d’accueil.

Sinon on déclare les variables *$sql, $res* puis on exécute cette requête en vérifiant si elle existe et on redirige.

Sinon on déclare les variables *$nom, $prenom, $addr, $ville, $email, $pass, $mutuelle*, puis on échappe leur caractère avec la méthode *real\_escape\_string* et *Mysqli*.



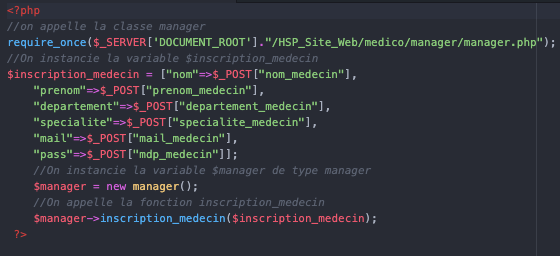
Si l’utilisateur est inscrit alors on lui envoie un mail en vérifiant si la requête est bien exécutée puis on appelle la méthode STMP qui permet l’envoie de mail. Depuis la méthode STMP on déclare les variables host *STMAuth, Username ,Password, STMPSecure et Port*.

Puis on envoie le mail avec la méthode setFrom qui permet d’y attribuer le mail, le mot de passe du compte de la personne qui enverra le message à l’utilisateur inscrit.

Puis on affiche le corps du mail avec les variables de la méthode isHTML(), *$mail->object,$mail->body* puis on redirige vers la page d’accueil l’utilisateur en question

Sinon on affiche une erreur générée par *$msqli.*

**inscription\_medecin.php**



On appelle la classe manager puis on déclare la variable *$inscription\_medecin* de type array() puis on stocke la valeur des inputs récupérés dans un tableau associatif, avec la méthode *$\_POST*, puis on déclare la variable *$manager* de type new Manager() puis on appelle la méthode *inscription\_medecin* qui prends en paramètre le tableau *$inscription\_medecin.*

**Modification\_form.php**



On démarre une session puis on appelle la classe manager, on déclare la variable *$manager* de type new Manager (), puis on appelle la méthode *connexion\_mysqli*,

On déclare les variables *$nom, $prenom, $adresse, $ville, $mutuelle*.

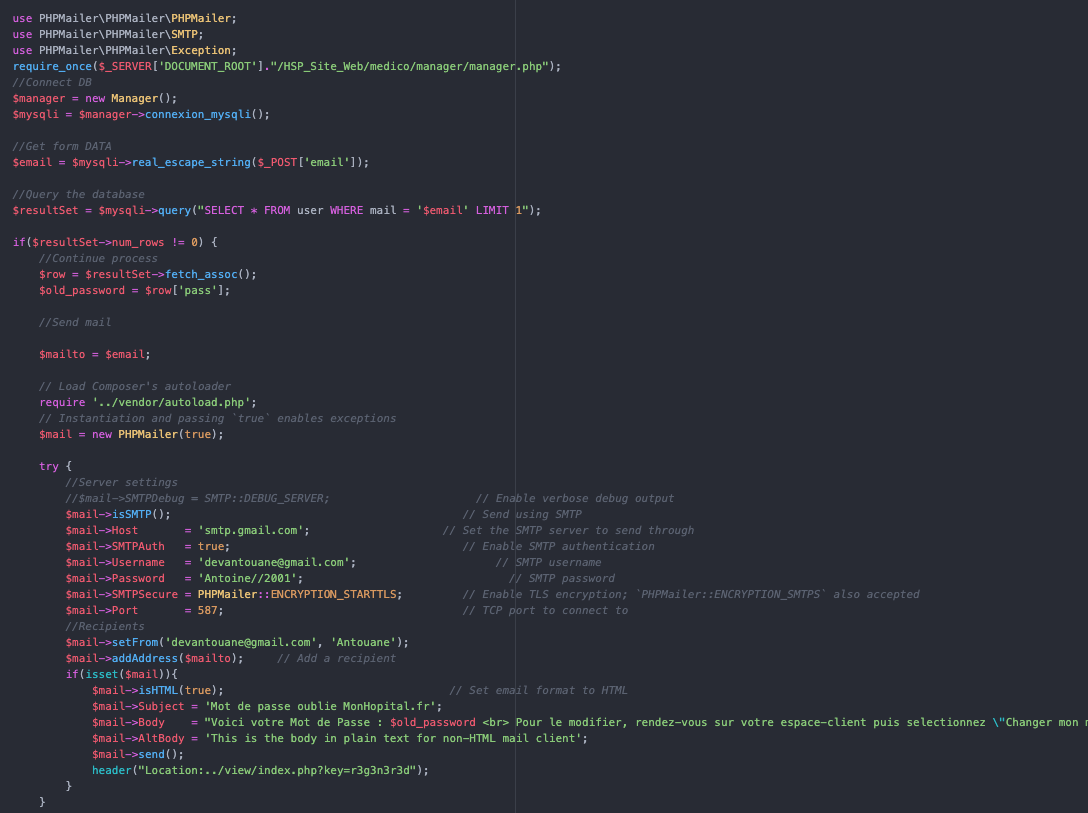
Puis on échappe les caractères avec la méthode *real\_escape\_string* des variables que l’on a déclaré. On déclare la variable *$user\_id* qui permet de stocker l’id de la session de l’utilisateur.

On exécute la requête SQL qui permet de modifier les informations de l’utilisateur par rapport à son ID.

Si la requête existe alors on redirige l’utilisateur vers le traitement de déconnexion

Sinon on affiche une erreur *$mysqli*.

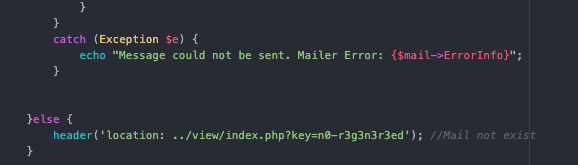
**Password\_form.php**



On utilise la méthode use de la PHP qui permet d’utiliser les dossiers de PHP Mailer, on déclare la variable $manager de type new Manager();

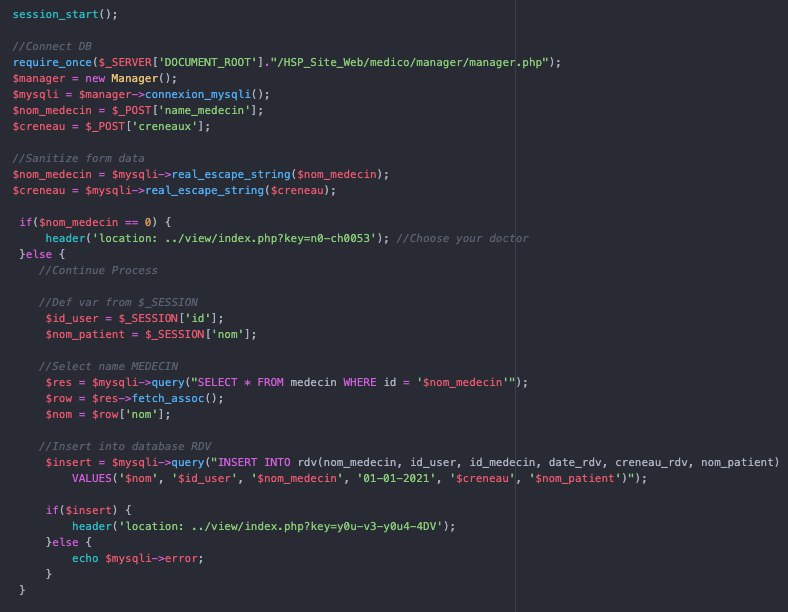
Puis on appelle la fonction *connexion\_mysqli*.

Ensuite on échappe les caractères du mail reçu avec la méthode *$\_POST.*



Sinon on redirige vers une erreur si le mail ne s’envoie pas, sinon on redirige vers la page d’accueil avec en lien une clé envoyé en *$\_GET.*

**prise\_rendez\_vous.php**



On appelle la classe manager avec la fonction require\_once puis on déclare la variable *$manager* de type new Manager(), on déclare la variable *$mysqli* en appelant la méthode *connexion\_mysqli()*, puis on déclare la variable *$nom\_medecin, $creneau* qui ont pour valeur les $\_POST, puis on échappe leur caractère avec la méthode real\_escape\_string,

Si le nom du médecin existe alors on redirige.

Sinon on déclare les variables *$id\_user, $nom\_patient* qui ont pour valeur les informations de session de l’utilisateur

Puis on affiche les informations du médecin par rapport à son *$nom\_medecin*.

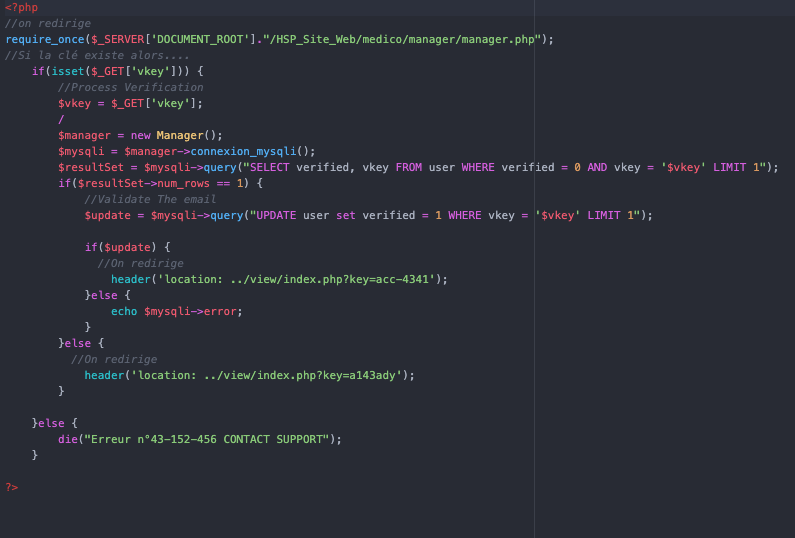
On déclare la variable $nom qui a pour valeur *$row[‘nom’]*.

Puis on insert les données des rendez-vous dans la table rdv.

Si le tableau existe alors on redirige vers la page d’accueil.

Sinon on affiche une erreur *$mysqli*

**Verify.php**



On appelle la classe manager depuis la racine du projet, si la clé est vérifiée alors on déclare la variable *$vkeyh*, puis on déclare la variable $manager de type new Manager(), puis on déclare la variable *$mysqli* en appelant la méthode *connexion\_mysqli()*,

On exécute la requête qui permet de vérifier l’existence des clés, puis si la variable *$resultSet* existe alors on modifie sa valeur par rapport à la clé que l’on a déclarée en haut de code.

Si la requête existe alors on redirige sinon on affiche une erreur *$sqli*

Sinon on redirige vers la page d’accueil

Sinon on affiche une erreur globale.

1. **Include**

Le dossier *include* contient toutes les pages qui permettent les grandes fonctionnalités du site avec la barre de navigation ainsi que le pied de page et le lien script javascript Bootstrap et CSS Bootstrap.

1. **Manager**

Le dossier Manager contient la classe Manager qui contient les méthodes globales du site,



Les méthodes *connexion\_bd* et *connexion\_mysqli* permettent de se connecter à la BDD, en déclarant la variable *$bdd* de type new PDO() on retourne sa valeur. On déclare la variable *$mysqli* de type msqli\_connect() puis on retourne sa valeur.



On déclare la variable *$msqli* en appelant la fonction de la classe manager avec $this et on déclare les variables *$nom,$prenom, $departement, $specialite, $mail, $pass*. On déclare la variable *$pass* de valeur md5(time().nom).

On déclare la variable $insert qui permet d’insérer les informations dans la table médecin.

Si le tableau existe alors on redirige l’utilisateur vers la page d’accueil.

Sinon on redirige vers l’index avec une erreur en clé.

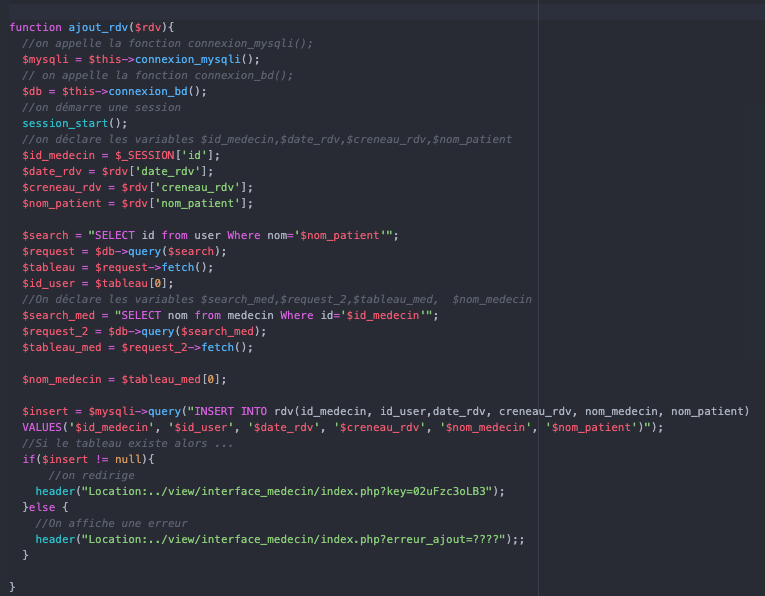


On déclare la variable *$msqli* en appelant la fonction de la classe manager avec *$this* et on déclare les variables *$nom,$prenom, $departement, $specialite, $mail, $pass, $vkey* . On déclare la variable *$pass* de valeur md5(time().nom).

On déclare la variable $insert qui permet d’insérer les informations dans la table user

Si le tableau existe alors on redirige l’utilisateur vers la page d’accueil.

Sinon on redirige vers l’index avec une erreur en clé.

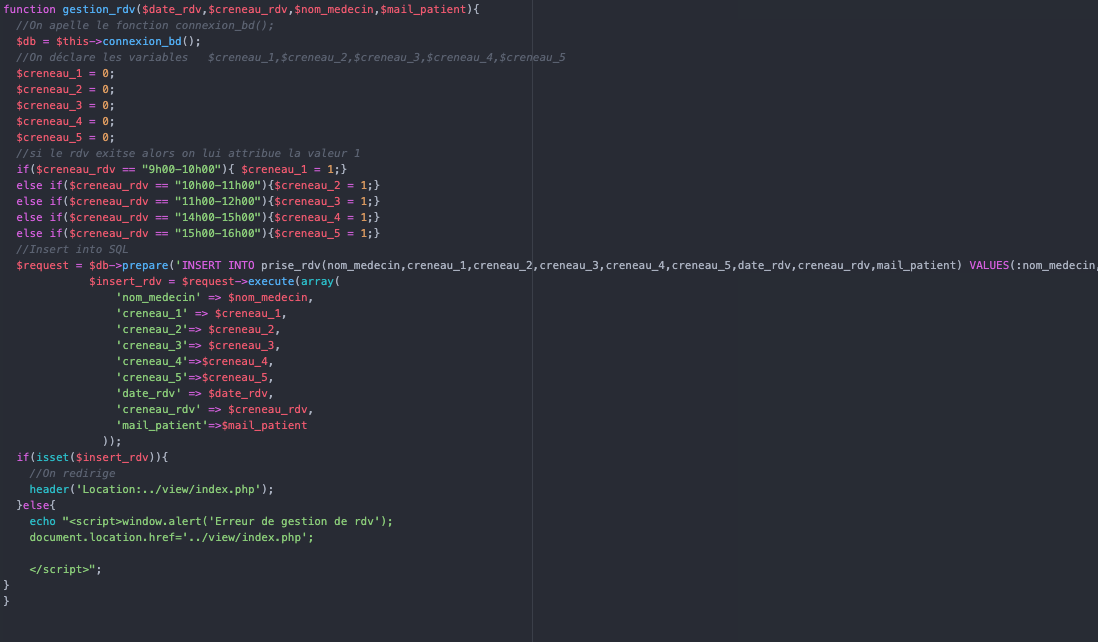


On déclare la variable *$myslqi* en appelant la méthode *connexion\_bd()* qui permet de se connecter à la BDD. Puis on démarre une session. On déclare les variables *$id\_medecin, $date\_rdv, $creneau\_rdv, $nom\_patient* qui ont pour valeur l’index du tableau $rdv.

On déclare la variable *$search* qui permet d’afficher l’id de l’utilisateur, on déclare les variables *$id\_user* de valeur *$taleau[0]*.

Puis on insère les informations du rendez-vous dans la table rdv.

Si la variable $insert existe alors on redirige le médecin dans la page d’accueil de l’interface médecin, sinon on redirige vers la page d’accueil avec une erreur en clé.



On déclare les variables *$creneau\_1, $creneau\_2, $creneau\_3, $creneau\_4 et $creneau\_5* qui ont pour valeur 0, si le *$creneau\_rdv* existe alors on initialise l’une de ces variables à 1.

Puis on insert les informations dans la table prise de rendez-vous.

Si *$insert\_rdv* existe alors on redirige l’utilisateur dans la page d’accueil du site, sinon on affiche une erreur.



On déclare la variable *$db* en appelant la fonction *connexion\_bd()*

Puis on déclare la variable *$suppresion* qui permet de supprimer le rdv par rapport à la valeur de la variable *$delete* reçu en paramètre de la méthode query().

On déclare la variable $tableau qui permet de mettre dans un tableau les valeurs de la requête SQL.

Si le tableau existe alors on redirige vers la page d’accueil sinon, on redirige vers la page d’accueil avec une erreur envoyé en lien $\_GET.



On déclare la variable *$db* en appelant la fonction *connexion\_bd()*

Puis on déclare la variable *$suppresion* qui permet de supprimer le rdv dans l’interface administrateur par rapport à la valeur de la variable *$delete* reçu en paramètre de la méthode.

On déclare la variable $tableau qui permet de mettre dans un tableau les valeurs de la requête SQL.

Si le tableau existe alors on redirige vers la page d’accueil sinon, on redirige vers la page d’accueil avec une erreur envoyé en lien *$\_GET.*



On déclare la variable *$db* en appelant la fonction *connexion\_bd()*

Puis on déclare la variable $suppression qui permet de supprimer l’utilisateur par rapport à la valeur de la variable *$delete* reçu en paramètre de la méthode.

On déclare la variable $tableau qui permet de mettre dans un tableau les valeurs de la requête SQL.

Si le tableau existe alors on redirige vers la page d’accueil sinon, on redirige vers la page d’accueil avec une erreur envoyé en lien *$\_GET*.

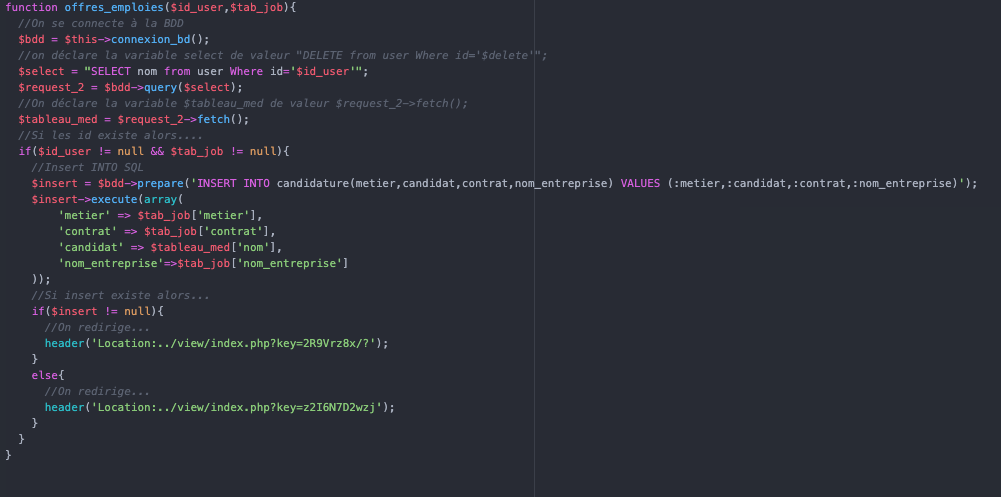


On déclare la variable $db en appelant la fonction connexion\_bd()

Puis on déclare la variable $suppression qui permet de supprimer l’utilisateur par rapport à la valeur de la variable $delete reçu en paramètre de la méthode.

On déclare la variable $tableau qui permet de mettre dans un tableau les valeurs de la requête SQL.

Si le tableau existe alors on redirige vers la page d’accueil sinon, on redirige vers la page d’accueil avec une erreur envoyé en lien *$\_GET.*



On déclare la variable *$bdd* en appelant la fonction *connexion\_bd()*

Puis on déclare la variable $select qui permet d’afficher les utilisateurs par rapport à son id.

On déclare la variable *$tableau\_med* avec la méthode *fetch()* PHP.

Si l’*id\_user et $tab\_job* existent alors on insert les données dans la table candidature sinon on redirige vers la page d’accueil avec une clé en lien GET. Sinon redirige l’utilisateur sur la page d’accueil avec une clé qui génère une erreur.

1. **Modal**

Le dossier Modal permet contient toutes les fenêtres Bootstrap qui permettent d’afficher les fonctionnalités du site, la connexion, la prise de rendez-vous, la modification d’informations, la candidature d’offre d’emplois ou encore les paramètres sur compte utilisateur.

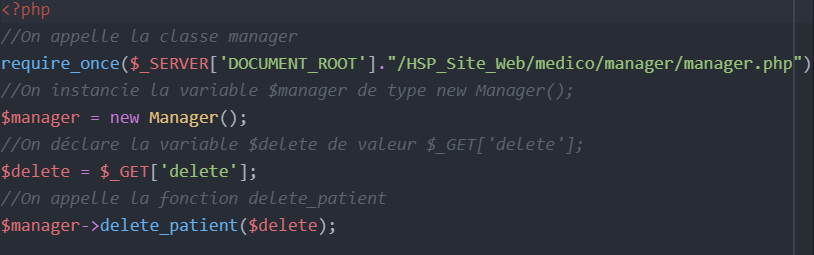
1. **Model**

Le model contient tous les Getter et Setter qui permette de retourner les valeurs des POST envoyés par l’utilisateur. Cette partie gère les *données* de votre site. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le manager.

1. **Traitement**

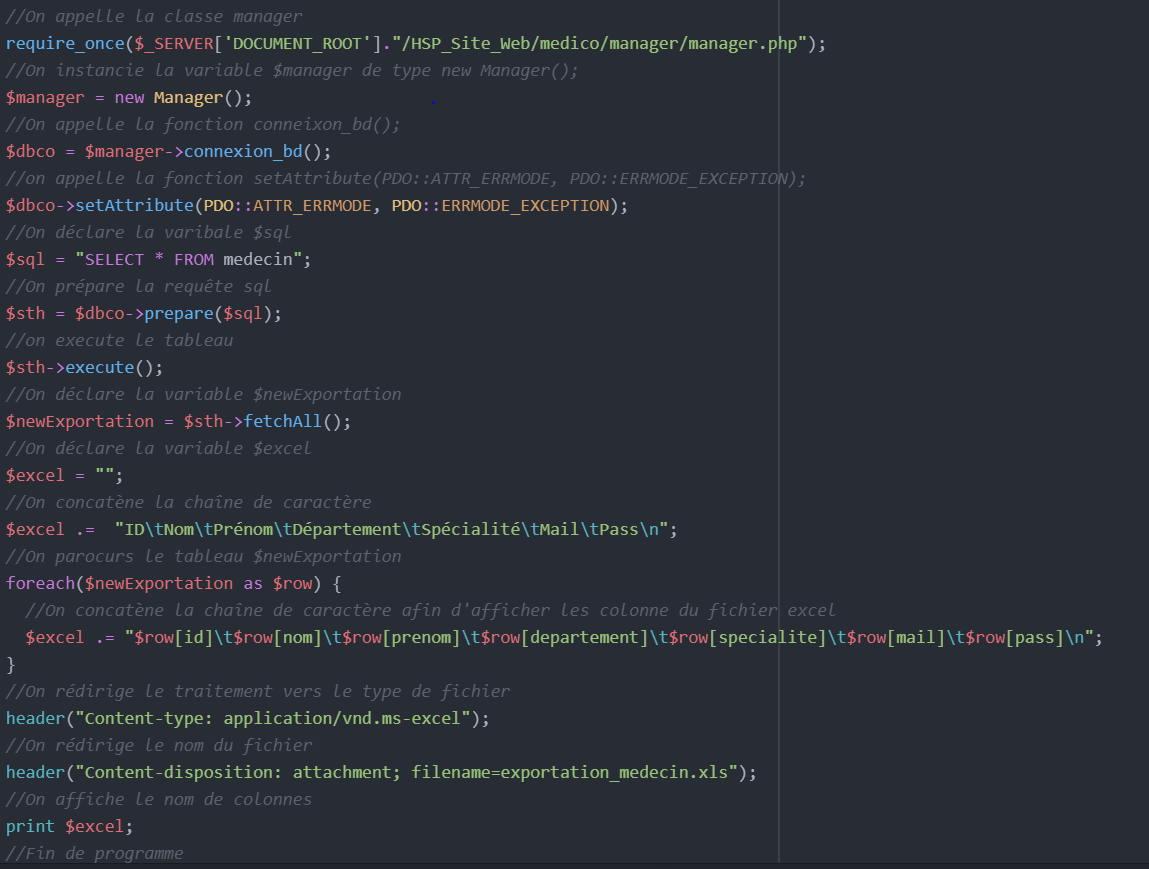
Le dossier traitement contient tous les traitements du site connexion, inscription, l’ajout de rendez-vous, l’insertion administrateur etc...

**Delete\_patient.php, Delete\_patient, Delete\_medecin,**

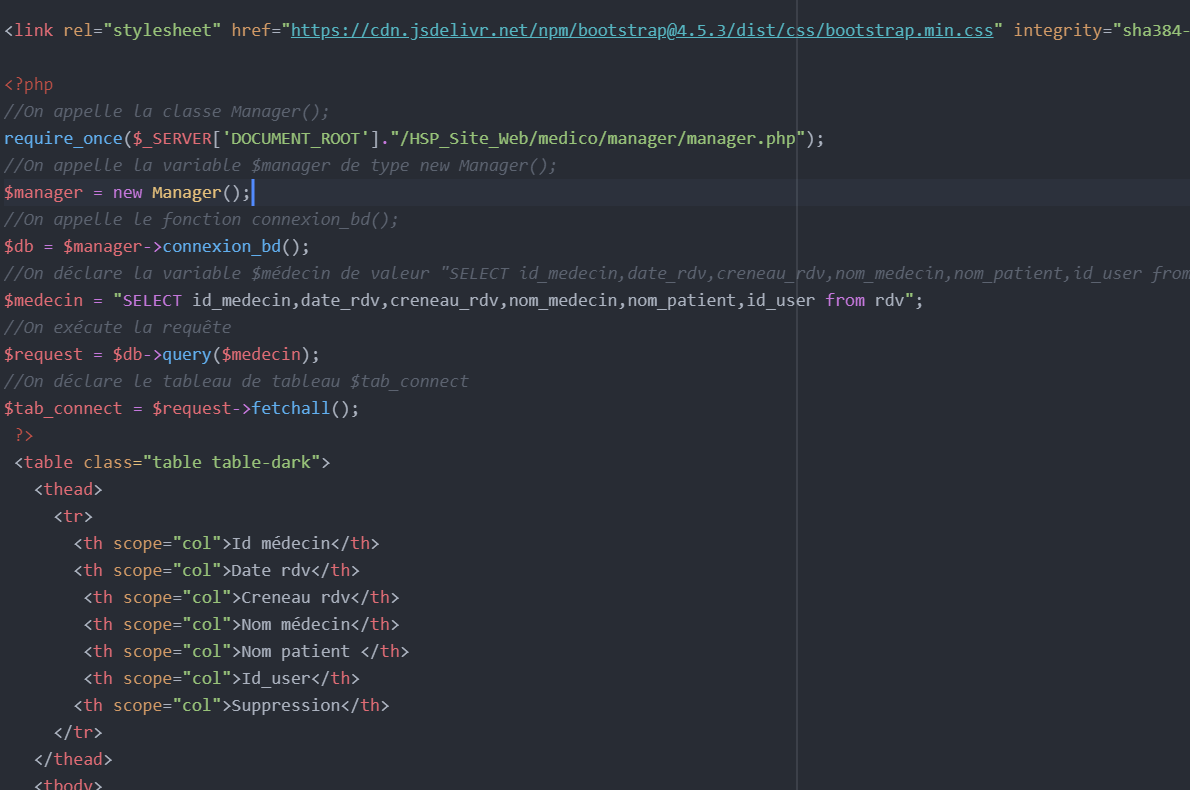


On appelle la classe manager, on déclare une variable $manager de type new Manager(), on déclare la variable *$delete* puis on appelle la méthode *delete\_patient*.

**Traitement Exportation des données**



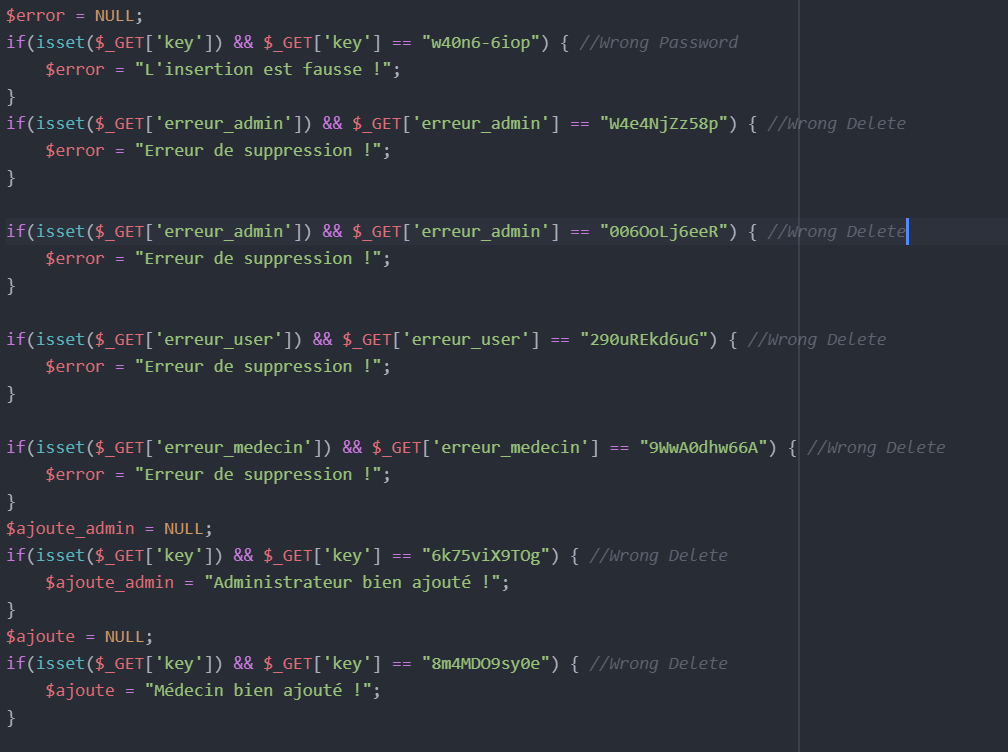
On appelle la classe manager puis on déclare une variable qui permet d’afficher les médecins ensuite on affiche le nom des colonnes avec les noms des médecins avec un foreach. On redirige l’écriture du fichier vers un fichier Excel.



On exécute la requête SQL dans la table rendez-vous puis on vérifie si la requête existe. On déclare un tableau avec la balise <table></table> puis on parcourt le tableau de la requête SQL qui permet d’afficher les rendez-vous ensuite on crée des cellules afin d’y mettre les index de ce tableau.

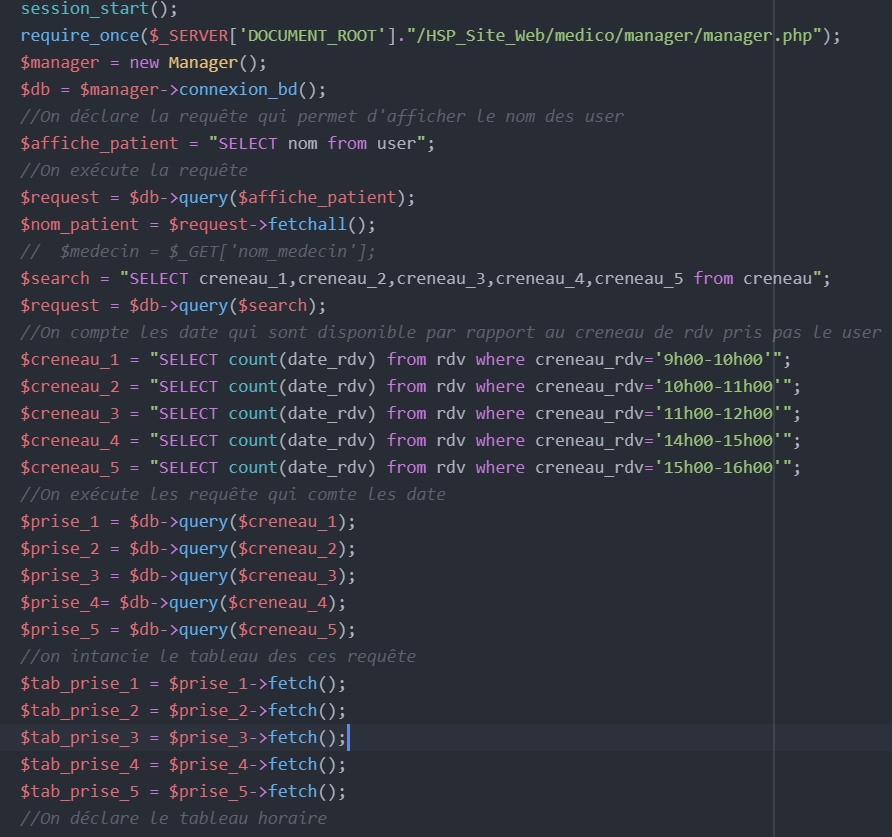
1. **View**

**Interface Admin**



Dans cet index, on effectue un peu de gestion de d’erreur qui est créée par la réception des clés retournés par les méthodes du manager.

**Interface médecin**

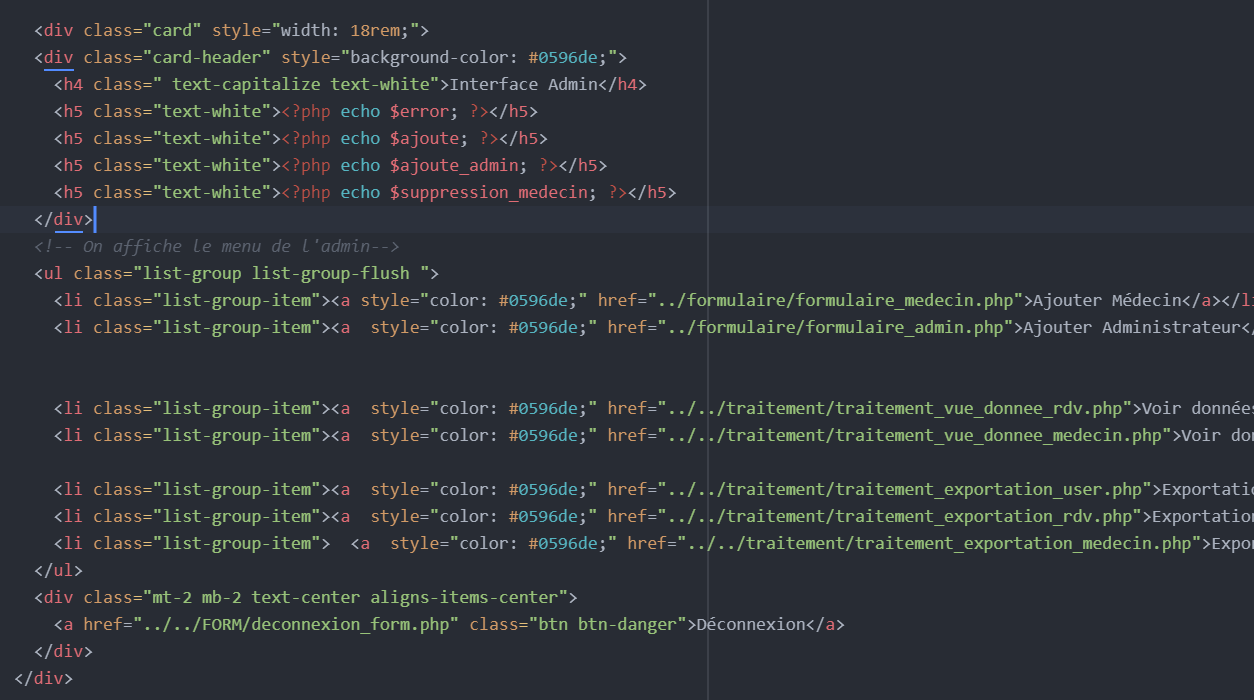


Afin d’afficher les rendez-vous en fonction des dates on effectue la même gestion d’erreur que sur la gestion de rendez-vous disponible sur l’interface utilisateur.

**Interface Admin**



**On effectue de la gestion d’erreur pour les afficher directement dans l’index de l’interface admin.**



Ensuite si les variables existent on les affiche, puis ont reparties les fonctionnalités dans des liste et liste Bootstrap.

**Conclusion**

Pour conclure ce projet m’a apporté de nouvelles notions en programmation c’est à dire la maîtrise du langage PHP en Programmation orientée objet. J’ai pu apprendre de nouvelles notions en programmation comme de l’algorithmie en javascript mais aussi la gestion de base de données en SQL avec des clés étrangères et primaires.