Nicolas Gomes

Lycée Turgot  2021-2022

Documentation technique ProjEt gsb visite



Sommaire

1. Présentation et contexte du projet
2. Eléments à réaliser
3. Model Conceptuel de données (MCD)
4. Diagramme de classe
5. Model Relationnel de données (MLR)
6. Structure du projet
7. Arborescence du site
8. Configuration de la connexion à la base de données
9. Explication du code complexe
10. Interface du site
11. Présentation et contexte du projet

Le projet a été développé avec les technologies suivantes : HTML, CSS, JS, PHP et MySQL.

Il a été réalisé avec l’IDE Visual Studio et UWamp comme serveur WEB. La base de données a été créée sous phpMyAdmin.

L'activité commerciale d'un laboratoire pharmaceutique est principalement réalisée par les visiteurs médicaux. En effet, un médicament remboursé par la sécurité sociale n’est jamais vendu directement au consommateur mais prescrit au patient par son médecin.

Toute communication publicitaire sur les médicaments remboursés est d'ailleurs interdite par la loi. Il est donc important, pour l’industrie pharmaceutique, de promouvoir ses produits directement auprès des praticiens.

L'activité des visiteurs médicaux consiste à visiter régulièrement les médecins généralistes, spécialistes, les services hospitaliers ainsi que les infirmiers et pharmaciens pour les tenir au courant de l’intérêt de leurs produits et des nouveautés du laboratoire.

Pour affiner la définition des objectifs et l’attribution des budgets, il sera nécessaire d’informatiser les comptes rendus de visite.

L'activité est composée principalement de visites : réalisées auprès d’un praticien et on souhaite en connaître la date, le motif (6 motifs sont fixés au préalable), et savoir, pour chaque visite, les médicaments présentés et le nombre d’échantillons offerts. Le bilan fourni par le visiteur (le médecin a paru convaincu ou pas, une autre visite a été planifiée…) devra aussi être enregistré.

1. Eléments à réaliser

Pour gérer les rapports de visite

Le visiteur demande à créer un nouveau rapport de visite

Le système retourne un formulaire avec la liste des médecins et des champs de saisie

Le visiteur sélectionne un médecin à partir de son début de nom, sélectionne la date et remplit les différents champs, sélectionne les médicaments et les quantités offertes et valide

Le système enregistre le rapport

Le visiteur demande à modifier un rapport

Le système retourne un formulaire avec une date à sélectionner

Le visiteur sélectionne la date

Le système retourne les rapports que le visiteur a effectués à cette date

Le visiteur sélectionne un rapport de visite

Le système retourne les informations déjà saisies concernant le motif et le bilan

Le visiteur modifie les informations

Le système enregistre les modifications

Pour gérer les médecins

Le visiteur demande à voir les informations concernant un médecin

Le système retourne un formulaire avec un champ de recherche du médecin

Le visiteur sélectionne un médecin à partir de son début de nom et valide

Le système retourne les informations personnelles concernant ce médecin

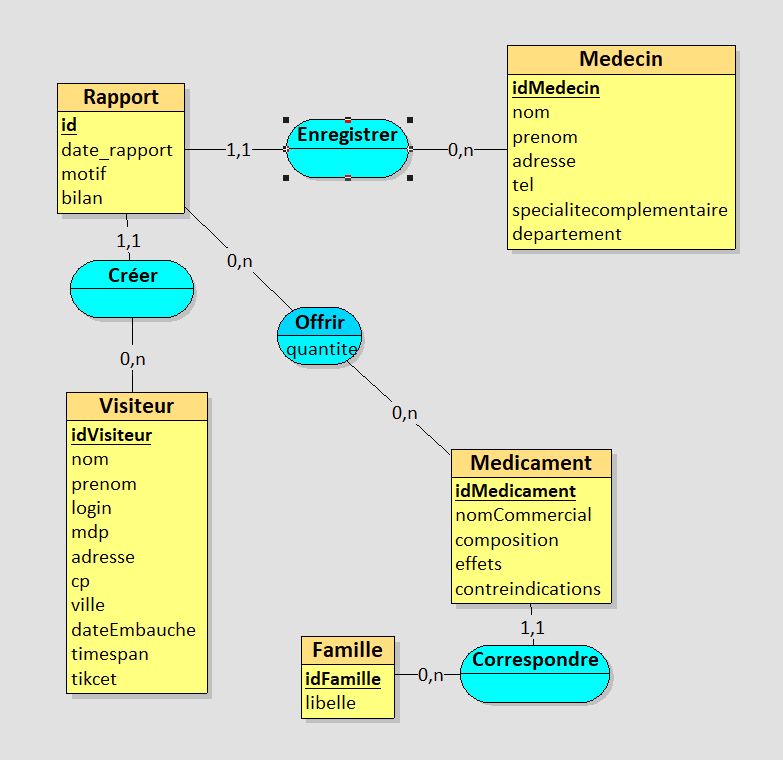
Le visiteur demande à voir tous les anciens rapports de visite concernant ce médecin

Le système retourne tous ses rapports

Le visiteur demande à modifier certains champs concernant des informations du médecin

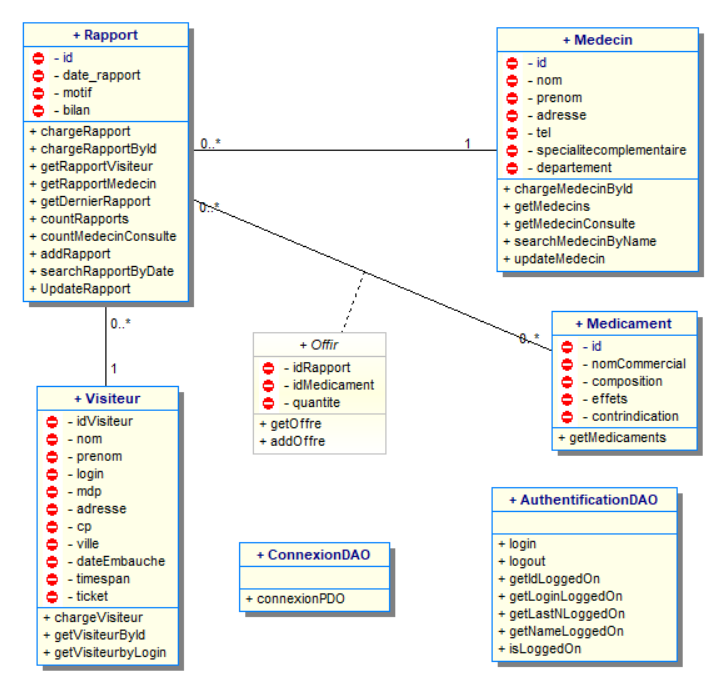
Le système enregistre ces modifications.

1. Modèle Conceptuel de données (MCD)



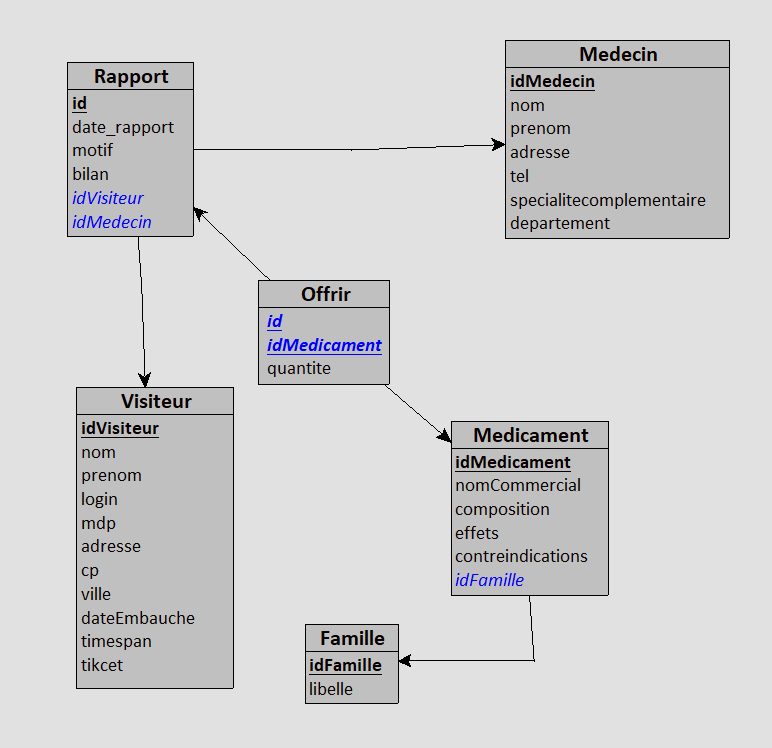
Ici l’entité Rapport recevra par la suite la clé primaire de l’entité visiteur qui sera une clé étrangère. De même avec l’entité médecin. Comme les cardinalités entre rapport et médicament sont 0,n elle deviendra une table par la suite avec une clé primaire composé (idMedicament, IdRapport). Pour l’entité Médicament, elle recevra par la suite la clé primaire de famille qui sera une clé étrangère. On peut voir toute ces modifications dans la partie V, le modèle relationnel de données.

1. Diagramme de classe



Pour des raisons de visibilités et clarté du diagramme, les get des attributs privés de chaque classe n’ont pas été écrites.

1. Modèle relationnel de données (MLR)

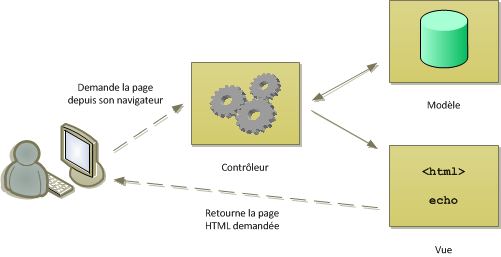


1. Structure du projet

Le projet est réalisé avec le modèle MVC. La structure est composée de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.



* Un modèle (dossier Model) contient les données à afficher. C’est ici qu’on va définir nos classes métiers, techniques et d’accès aux données (DAO). 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 contrôleur. On y trouve donc entre autres les requêtes SQL.
* Une vue (dossier View) contient la présentation de l'interface graphique. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple la liste des médecins.
* Un contrôleur (dossier Controller) contient la logique concernant les actions effectuées par l'utilisateur. C'est l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP.

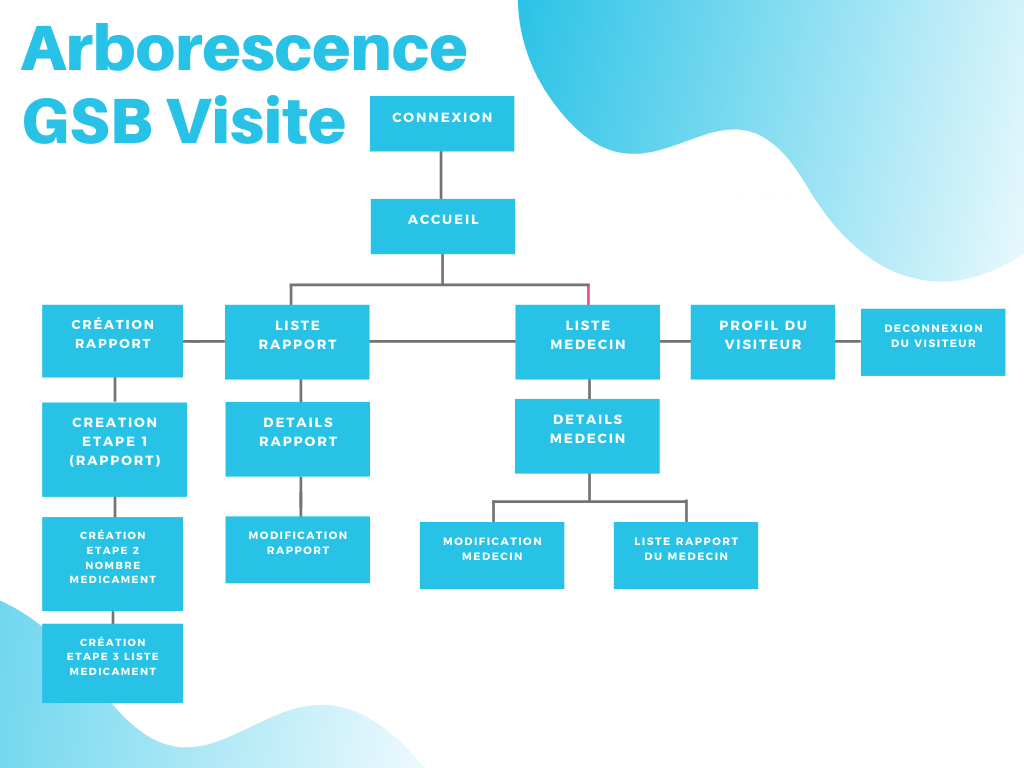


La requête du visiteur arrive au contrôleur et celui-ci lui retourne la vue

On dispose par la suite d’un dossier CSS qui contient notre css (design du site web) et un dossiers images regroupant les différent logo, icones et images du site.

Une image contenant texte

Description générée automatiquement

1. Arborescence du site web
2. Configuration de la connexion à la base de données

La base de données est créée sous PhpMyAdmin. Le code ci-dessous ce trouve dans le dossier Model et le nom du fichier est ConnexionDAO.php

Une image contenant texte

Description générée automatiquement

Ici on va créer la classe ConnexionDAO et la fonction connexionPDO qui va nous permettre de nous connecter à la base de données. La fonction est statique cela va nous permettre d'y accéder sans avoir besoin d'instancier la classe à chaque fois. La fonction possède quatre variables : $login et $mdp qui vont permettre de se connecter à la phpMyAdmin et $bd qui correspond au nom de la base de données.

1. Explication du code
2. Le contrôleur principal

Le contrôleur principal est un contrôleur avec une fonction du même nom qui sert à définir toutes les actions que le visiteur va pouvoir faire et l’amener au bon contrôleur qui lui renverra la bonne vue. Ici les contrôleurs sont repartis en 3 parties :

* Les contrôleurs concernant les actions liés au visiteur (connexion, déconnexion, profil...) ;
* Les contrôleurs concernant les rapports (création, liste, fiche...) ;
* Les contrôleurs concernant les médecins (création, liste, fiche...) ;

Si l’action que le visiteur veut faire existe bien, celle-ci sera retourné, sinon c’est l’action par default qui le sera (ici l’action par default est celle de connexion).

Une image contenant texte

Description générée automatiquement

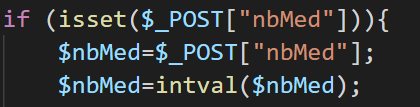
1. La création d’un rapport

Un Rapport est créé en 3 étapes. La première ou on rentre la date, le motif, le bilan et le medecin consulté. Cette fonction va permettre l’insertion des champs rentré juste avant dans la base de données :

Une image contenant texte

Description générée automatiquement

La seconde ou on rentre le nombre de médicaments offert. Dans la vue NbMed.php on rentrera un nombre (récupéré et stocké sur la variable $nbmed) et le contrôleur NewRapportMedicament ira la récupéré avec intval (retourne la valeur numérique entière équivalente d'une variable).



Enfin, la troisième ou on rentre le nom et la quantité des médicaments. On crée une variable erreur vide et 2 tableaux vide pour le médicament et la quantité. Si la valeur nbmed posté est rentré, on stocke la valeur dans la variable $nbmed (code au-dessus) et on créer une boucle for allant jusqu’à $nbmed. Si les médicament et quantité sont rentré par l’utilisateur on fait un array\_push sur les tableaux. Sinon la variable erreur sera égale à « Tous les médicaments ne sont pas rentrés »

Une image contenant texte

Description générée automatiquement

Par la suite, si la variable erreur est toujours vide, on définit la variable $rapport qui aura comme valeur l’id du dernier rapport (on récupère l’id du dernier rapport grâce à la fonction getDernierRapport). Avec une boucle for allant jusqu’au nombre d’éléments stocké dans le tableau des médicaments, on appelle la fonction addMedicament qui va ajouter les médicaments et la quantité. Sinon le visiteur reste sur la même page

Une image contenant texte

Description générée automatiquement

1. La connexion du visiteur

La connexion du visiteur est réalisée avec la fonction login du fichier authentification.php du dossier Model.

Si la session n’est pas déclarée, on l’initialise. On récupère le login du visiteur avec la fonction getVisiteurByLogin et on stocke la valeur dans la variable $util. Si le mot de passe de la base de données est égal au mot de passe donné, la session est créée, sinon on prévient l’utilisateur qu’il y a une erreur

Une image contenant texte

Description générée automatiquement

1. La liste, la fiche et la modification d’un medecin
2. La liste des médecins

Dans la vue MedecinView.php du dossier view, on affiche tous les médecins. Le contrôleur appelle la fonction searchMedecinByName. Dans la vue une boucle for est créer pour les afficher (uniquement leur nom et leur spécialité(s’ils n’ont pas de spécialité on affiche « ce medecin n’a pas de spécialité »)).

Une image contenant texte

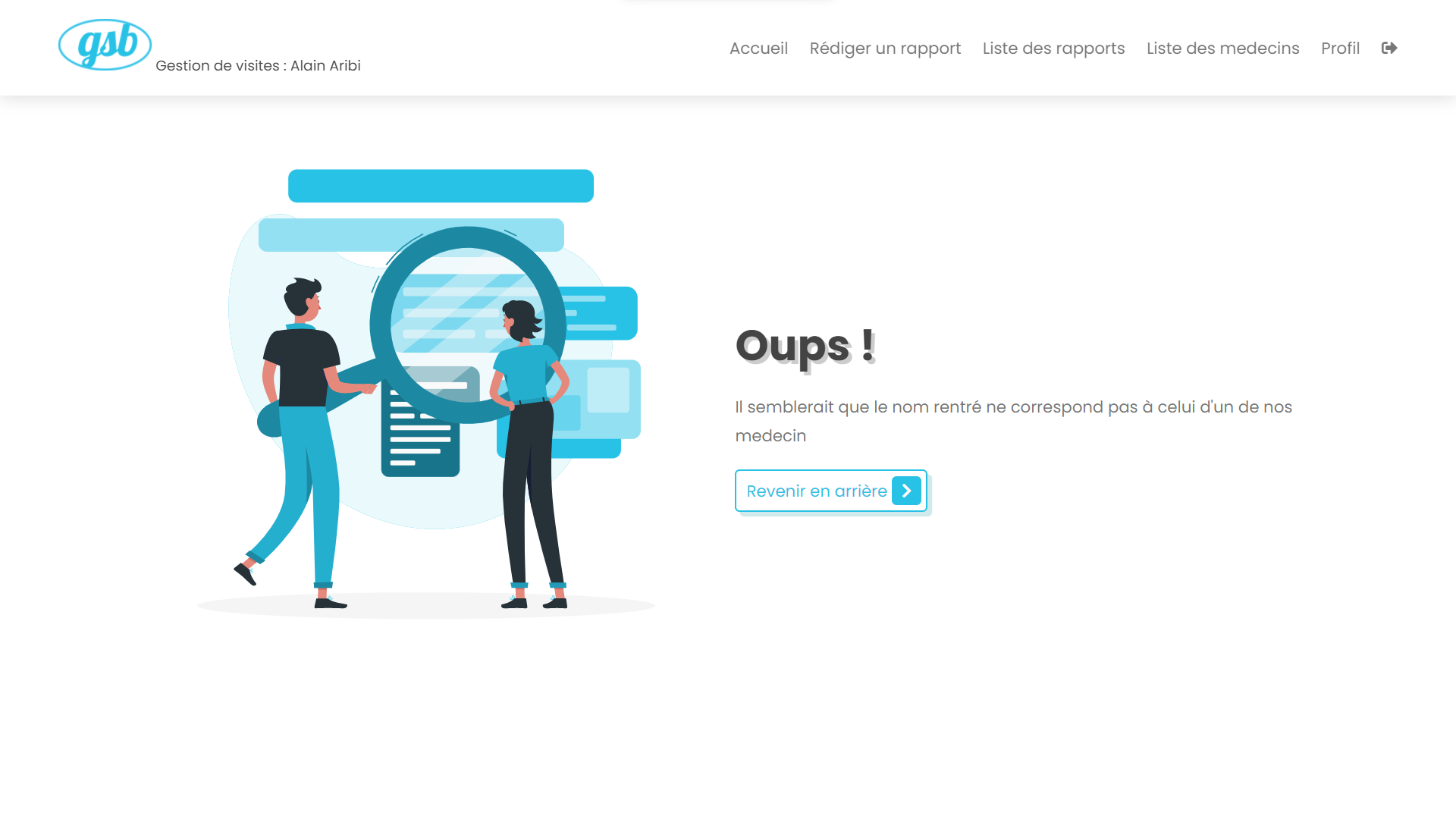
Description générée automatiquement

La recherche elle, appelle l’action SearchMedecin qui correspond au contrôleur searchMedecinByName. Si le nom est rentré on stocke celui-ci dans la variable nom et on vérifie à l’aide de la fonction SearchMedecinByName($nom) si $nom existe dans la base de données. Si le nom n’est pas rentré ou ne correspond pas à celui d’un des médecins de la base de données, la variable $lemedecin est null et on inclut une vue d’erreur. Sinon on affiche le résultat.

Une image contenant texte

Description générée automatiquement

Vue d’erreur :



1. La fiche d’un médecin

Dans la vue de la liste des médecins, il y a un bouton Voir Plus. Celui-ci nous ramène à l’action FicheMedecin (et on récupère l’id du médecin grâce à la fonction getId() de la classe Medecin.php (dossier model)).

<?php echo "<a href='./?action=FicheMedecin&id=".$lemedecin[$i]->getId(). "'class='btn'>"?> Voir Plus<span class="fas fa-chevron-right"></span> </a>

L’action ficheMedecin nous ramène au controleur FicheMedecin.php qui va récupérer l’id ($\_GET). On créer une variable $lemedecin avec la fonction chargeMedecinById($id) qui va créer l’objet Medecin (nom, prénom, adresse, tel...).

Une image contenant texte, capture d’écran, intérieur

Description générée automatiquement

Enfin dans la vue, on fera appel aux différentes méthodes de la classe médecin pour pouvoir afficher les informations du médecin concerné :

Une image contenant texte

Description générée automatiquement

1. La modification d’un médecin

Lorsque l’utilisateur sera sur la fiche d’un médecin, il aura un bouton qui lui permettra de modifier les informations de ce dernier. Il s’agit du même fonctionnement que le bouton pour accéder à la fiche mais cette fois, l’action appelé ne sera pas la fiche mais la modification.

<?php echo "<a href='./?action=UpdateMedecin&id=".$unMedecin->getId(). "'class='btn'>"?> Modifier les informations du docteur <?= $unMedecin->getNameMe();?><span class="fas fa-chevron-right"></span> </a> <br>

Le contrôleur retournera une vue qui affiche la même chose que la fiche mais les balise <p> ont été remplacé par des imput comme valeurs le résultat des get. Il est donc possible de modifier. L’enregistrement se fait du côté du contrôleur. On vérifier que les champs du formulaire son bien déclaré. Si oui, on stock chacune des valeurs dans des variables et on appelle la fonction UpdateMedecin qui va effectuer la modification.

Une image contenant texte, capture d’écran, moniteur, intérieur

Description générée automatiquement

Fonction Update :

Une image contenant texte

Description générée automatiquement

Si la modification est bien effectuée, on renvoie au visiteur une vue qui indique le succès de la modification.

Pour les rapports, la logique du code et des fonctions est pratiquement équivalente à ceux des médecins.