

PPE : GSB frais



Ressources et documents du projet

L'application est accessible à l'adresse suivante : <https://gsb.ostyl.fr/>

- [Fiche de situation professionnelle](#) (PDF)
- [Code source](#) (GitHub repository)
- [Accès à la base de données du projet en ligne](#) (phpMyAdmin)
- [Base de données pour installation en local](#) (fichier SQL et instructions d'installation en local)
- [Documentation technique](#) (générée avec phpDocumentor)

Documents fournis :

- [Contexte de l'entreprise GSB](#) (fichier Word)
- [Cahier des charges](#) (fichier Word)
- [Code et base de données fournis](#) (fichier .zip)

Compétences associées

- A1.1.1 , Analyse du cahier des charges d'un service à produire
- A1.2.1 , Élaboration et présentation d'un dossier de choix de solution technique
- A1.2.2 , Rédaction des spécifications techniques de la solution retenue
- A1.4.1 , Participation à un projet
- A2.3.2 , Proposition d'amélioration d'un service
- A4.1.2 , Conception ou adaptation de l'interface utilisateur d'une solution applicative
- A4.1.3 , Conception ou adaptation d'une base de données
- A4.1.6 , Gestion des environnements de développement et de test
- A4.1.8 , Réalisation des tests nécessaires à la validation d'éléments adaptés ou développés
- A4.2.3 , Réalisation des tests nécessaires à la mise en production d'éléments mis à jour

Présentation du projet

Contexte de la situation professionnelle

Le laboratoire Galaxy Swiss Bourdin (GSB) est issu de la fusion entre le géant américain Galaxy (spécialisé dans le secteur des maladies virales dont le SIDA et les hépatites) et le conglomérat européen Swiss Bourdin (travaillant sur des médicaments plus conventionnels), lui-même déjà union de trois petits laboratoires.

Description du besoin

Le suivi des frais est actuellement géré de plusieurs façons selon le laboratoire d'origine des visiteurs. On souhaite uniformiser cette gestion.

L'application doit permettre d'enregistrer tout frais engagé, aussi bien pour l'activité directe (déplacement, restauration et hébergement) que pour les activités annexes (événementiel, conférences, autres), et de présenter un suivi daté des opérations menées par le service comptable (réception des pièces, validation de la demande de remboursement, mise en paiement, remboursement effectué).

Diagramme des cas d'utilisation

L'application est fournie avec les cas d'utilisations liés aux visiteurs ; la partie comptable est à développer selon les cas d'utilisations suivants :

- Validation d'une fiche de frais
- Suivi du paiement des fiches de frais

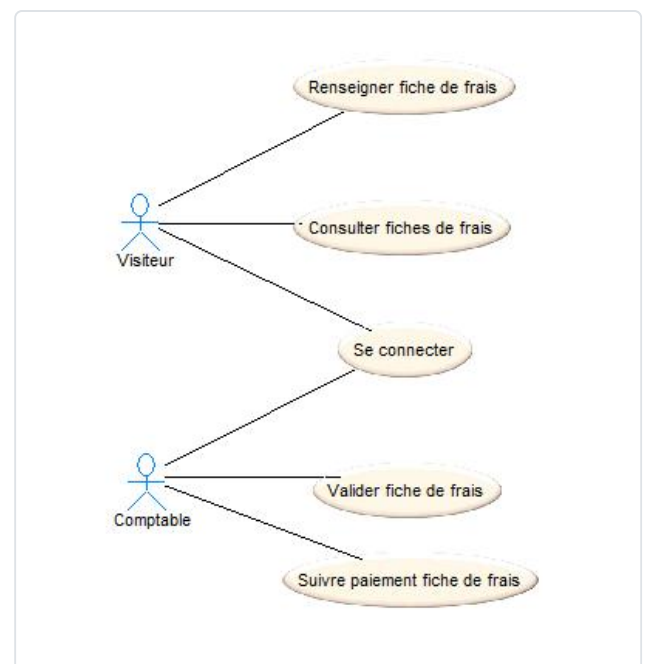


Diagramme des cas d'utilisation

Préparation de l'environnement de développement

Installation de WampServer

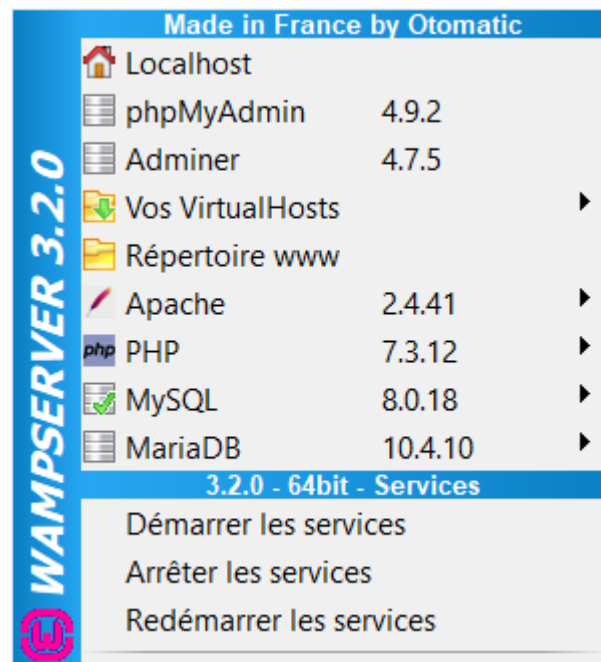
WampServer est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement des scripts PHP. C'est un environnement comprenant trois serveurs, un interpréteur de script, ainsi que phpMyAdmin pour l'administration Web des bases MySQL.

Installation de l'IDE NetBeans et création d'un nouveau projet

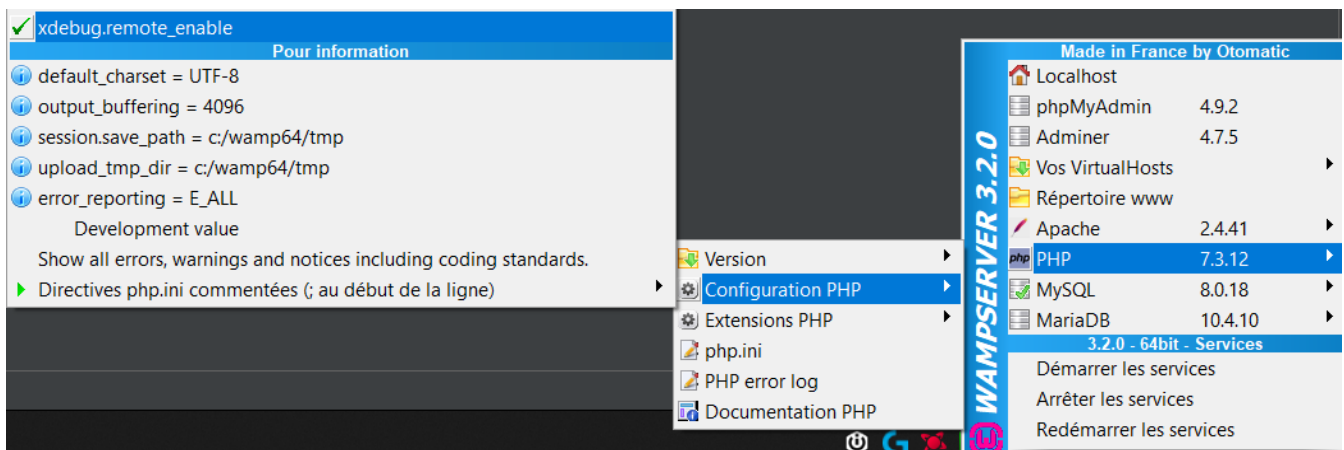
NetBeans est un environnement de développement intégré open source. En plus de Java, NetBeans permet la prise en charge native de divers langages tels le C, le C++, le JavaScript, le XML, le Groovy, le PHP et le HTML, ou d'autres par l'ajout de greffons.

Activation de Xdebug

Xdebug est une bibliothèque logicielle, extension pour PHP apportant des fonctions de débogage et de profilage.



Starting modules... Apache NetBeans IDE 12.0



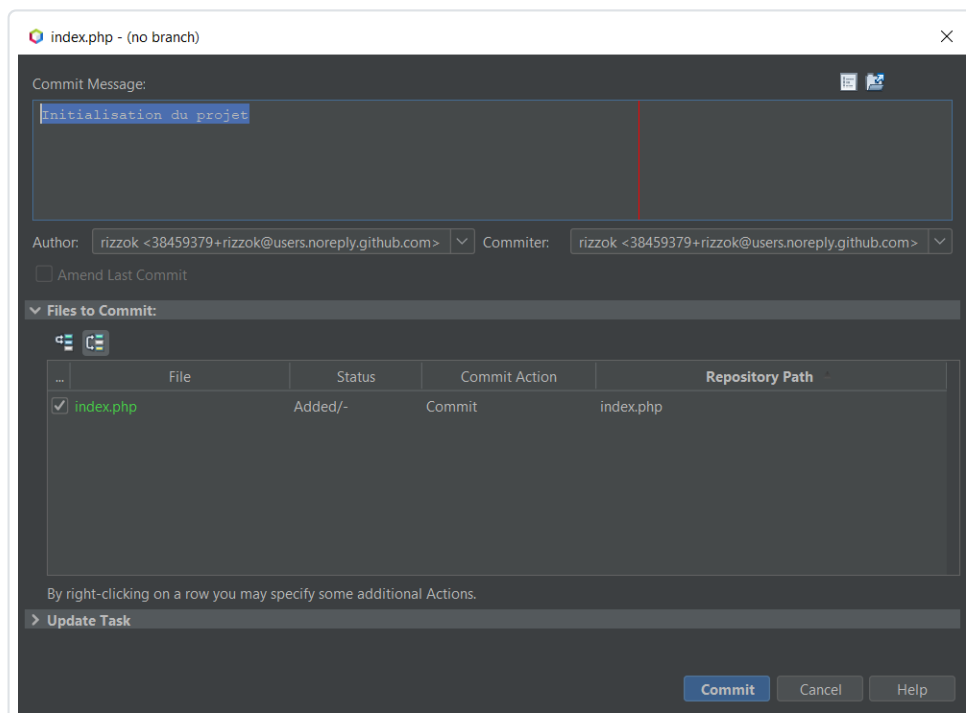
Activation de Xdebug

Initialisation de Git et création du repository sur GitHub

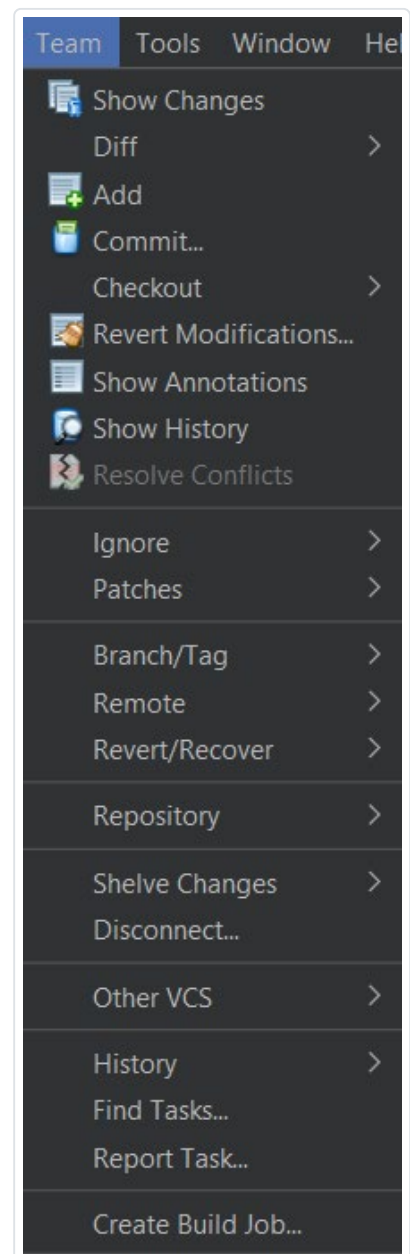
Git est un logiciel de gestion de versions décentralisé.

GitHub est un service web d'hébergement utilisant Git.

Pour plus de détails, rendez-vous sur la page que j'ai créé dédiée à ce sujet : [Outils de versioning : Git et GitHub](#)



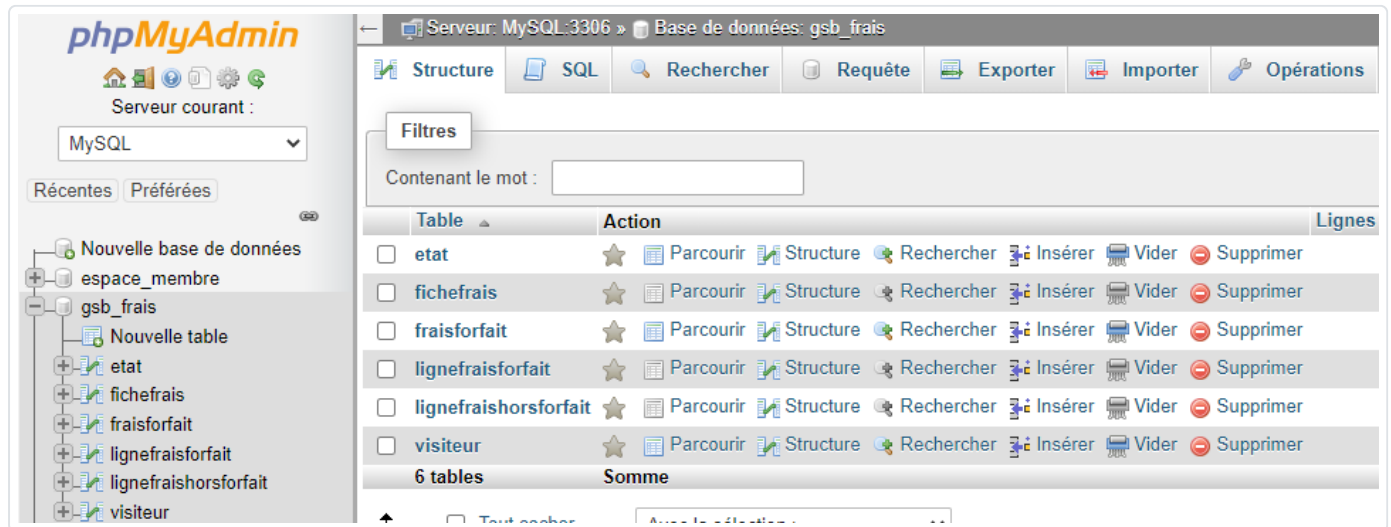
Premier commit sur GitHub



Outils liés à Git sur NetBeans

Import et modifications de la base de données

Import de la base de donnée, fichier .sql fourni avec l'application.



Import de la base de données

Pour pouvoir répondre aux besoins du cahier des charges et au différents cas d'utilisation décrits, j'ai renommé la table "visiteur" en "utilisateur" et j'y ai ajouté un champ booléen "estcomptable", qui est à 0 (faux) si l'utilisateur est un visiteur et à 1 (vrai) si c'est est un comptable.

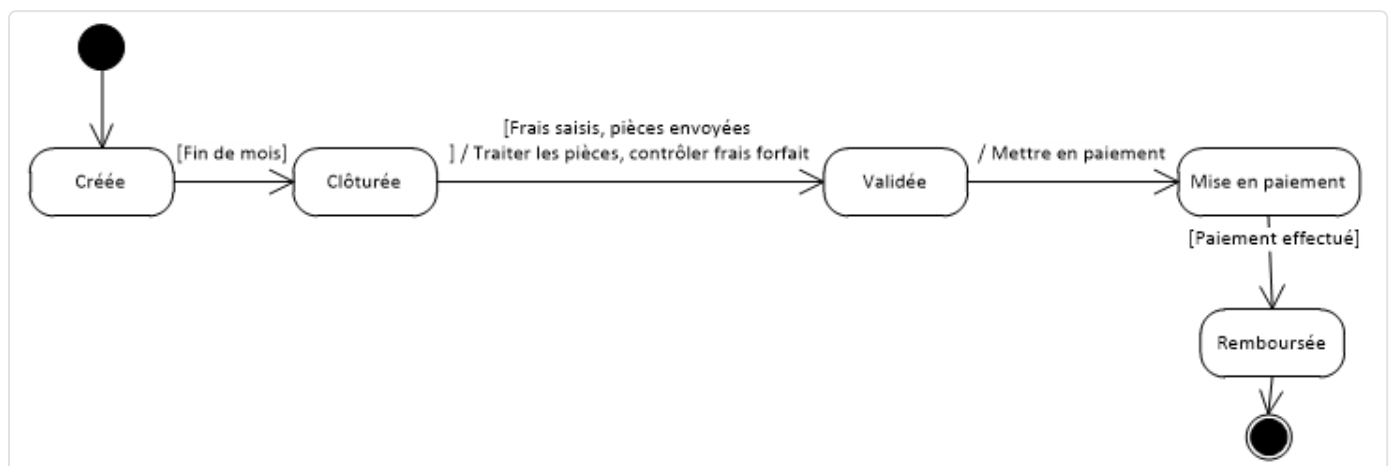
J'ai ensuite ajouté de 2 nouvelles lignes à la table "utilisateur", qui correspondent à l'ajout de 2 comptables, la valeur "estcomptable" est donc à 1 (true).

J'ai également ajouté à la table "etat", une ligne ayant les valeurs suivantes :

id = 'MP'

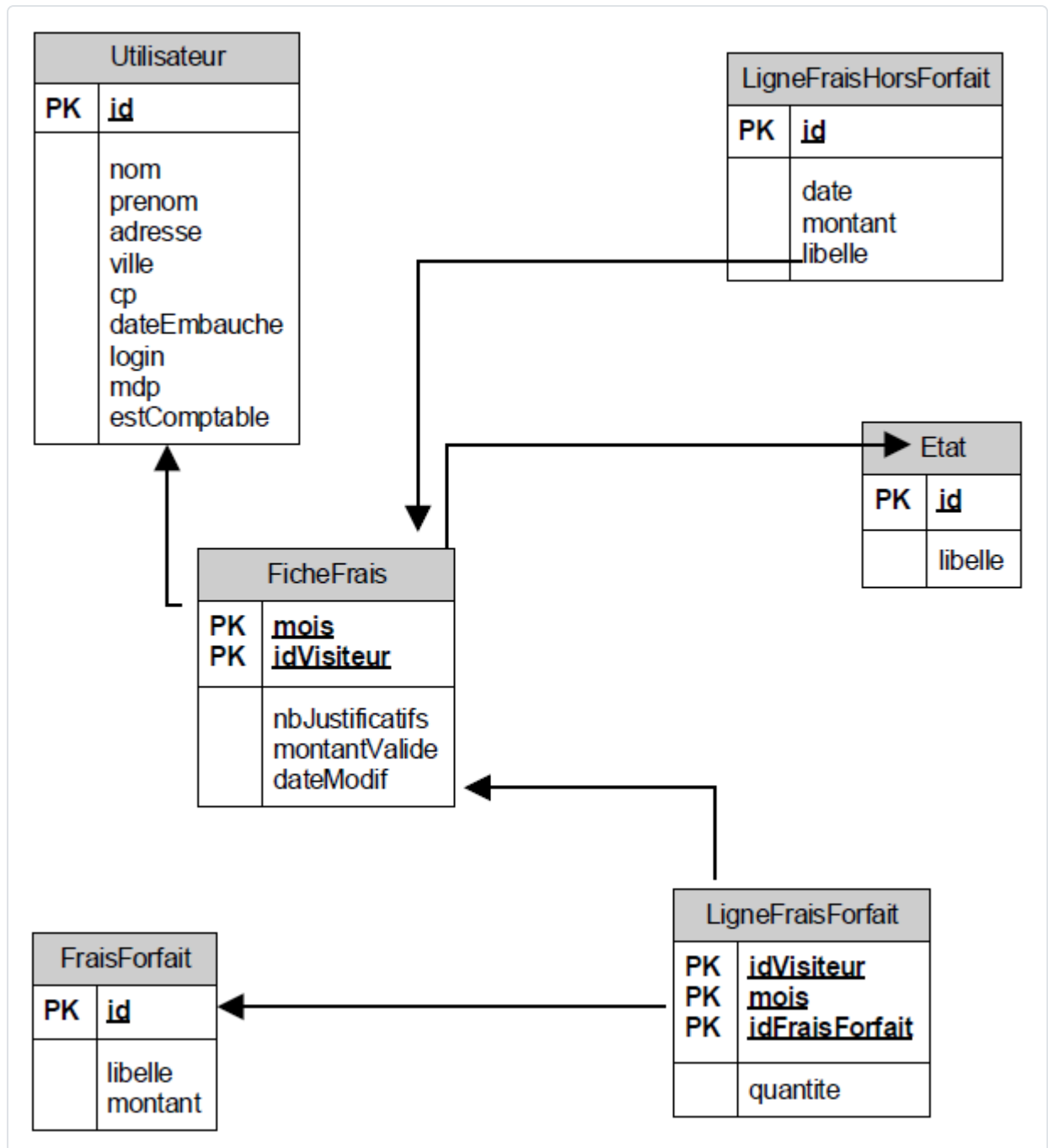
libelle = 'Mise en paiement'

Cette modification à pour but de répondre aux différents états décrits dans le schéma suivant, visible dans le cahier des charges :



Différents états d'une fiche de frais

Modèle entité-association de la base de données après modification :

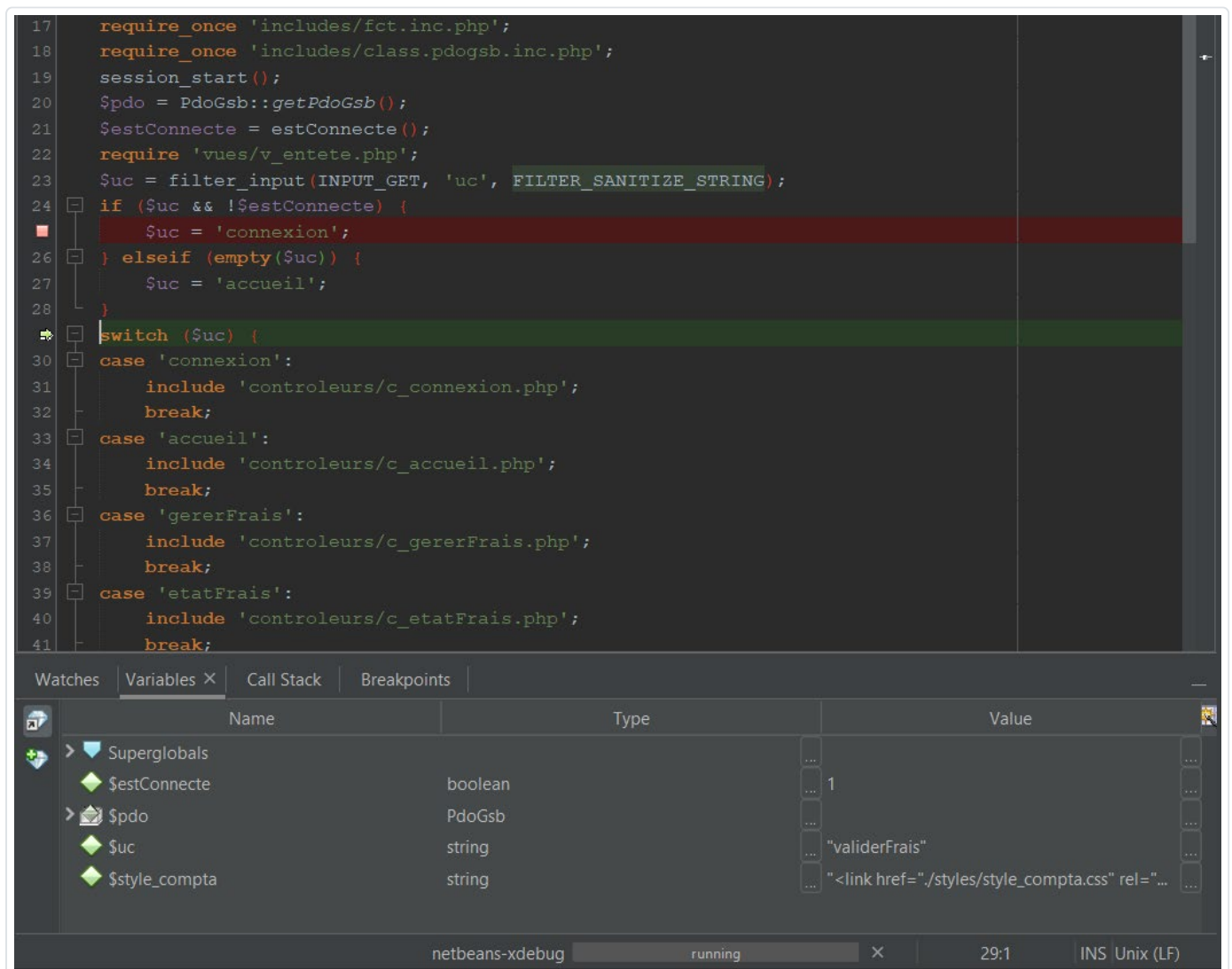


Modèle entité-association

Utilisation de Xdebug

L'extension Xdebug apporte de nombreuses fonctions de débogage et de profilage, comme :

- l'exécutions du code PHP pas à pas,
- la visualisation des variables et leurs assignations à un moment donné,
- les traces d'appels (stack trace en anglais) qui représentent les piles d'exécution,
- ou encore le profilage du code (analyse de l'exécution du code afin de connaître son comportement).



Exemple d'utilisation de Xdebug

Projet respectant l'architecture MVC

Le projet de base est fourni en respectant l'architecture MVC ou Modèle-Vue-Contrôleur. J'ai donc pris soin de continuer à respecter ce modèle qui profite à l'organisation du projet parce qu'elle permet de séparer les fichiers dans des répertoires différents selon leur utilisation.

En effet, nous avons d'une part le(s) fichier(s) "modèle", qui contient les données et la logique en rapport avec les données : validation, lecture, enregistrement.

La "vue" correspond à la partie visible d'une interface graphique. Une vue contient des éléments visuels ainsi que la logique nécessaire pour afficher les données provenant du modèle.

Le "contrôleur" est un module qui traite les actions de l'utilisateur et modifie les données du modèle et de la vue.

Voici un exemple de vue qui, comme on peut le voir, est un mélange de HTML et de PHP, qui se sert à la fois des données transmises par le modèle et le contrôleur :

```

5      * @category   PPE
6      * @package    GSB
7      * @author     Kévin RIZZO
8      */
9      ?>
10     <div class="row">
11         <div class="col-md">
12             <form action="index.php?uc=validerFrais&select=voirFicheDeFrais"
13                 method="post" class="form-inline m-20">
14                 <div class="form-group mr-10">
15                     <label for="lstVisiteur">Choisir le visiteur :</label>
16                     <select id="lstVisiteur" name="lstVisiteur"
17                         class="form-control">
18                         <?php foreach ($lesVisiteurs as $key => $unVisiteur): ?>
19                             <?php $nom = $unVisiteur['nom'];
20                             $prenom = $unVisiteur['prenom']; ?>
21                             <?php if ($key == $visiteurASelectionner): ?>
22                                 <option selected value="<?=$key ?>">
23                                     <?=$nom . ' ' . $prenom ?> </option>
24                             <?php else: ?>
25                                 <option value="<?=$key ?>">
26                                     <?=$nom . ' ' . $prenom ?> </option>
27                             <?php endif; ?>
28                         <?php endforeach; ?>
29                     </select>
30                 </div>
31                 <div class="form-group mr-10">
32                     <label for="lstMois" accesskey="n">Mois : </label>
33                     <select id="lstMois" name="lstMois" class="form-control">
34                         <?php foreach ($lesMois as $unMois): ?>
35                             <?php $mois = $unMois['mois'];
36                             $numAnnee = $unMois['numAnnee'];
37                             $numMois = $unMois['numMois']; ?>
38                             <?php if ($mois == $moisASelectionner): ?>
39                                 <option selected value="<?=$mois ?>">
40                                     <?=$numMois . '/' . $numAnnee ?> </option>
41                             <?php else: ?>
42                                 <option value="<?=$mois ?>">
43                                     <?=$numMois . '/' . $numAnnee ?> </option>
44                             <?php endif; ?>
45                         <?php endforeach; ?>
46                     </select>
47                 </div>
48                 <input id="ok" type="submit" value="Voir cette fiche"
49                     class="btn btn-success" role="button">
50             </form>
51         </div>
52     </div>
53

```

Fichier vue

Exemple de fichier contrôleur :

```
17 $idVisiteur = $_SESSION['idVisiteur'];
18 $mois = getMois(date('d/m/Y'));
19 $numAnnee = substr($mois, 0, 4);
20 $numMois = substr($mois, 4, 2);
21 $action = filter_input(INPUT_GET, 'action', FILTER_SANITIZE_STRING);
22 switch ($action) {
23     case 'saisirFrais':
24         if ($pdo->estPremierFraisMois($idVisiteur, $mois)) {
25             $pdo->creeNouvellesLignesFrais($idVisiteur, $mois);
26         }
27         break;
28     case 'validerMajFraisForfait':
29         $lesFrais = filter_input(INPUT_POST, 'lesFrais', FILTER_DEFAULT, FILTER_FORCE_ARRAY);
30         if (lesQteFraisValides($lesFrais)) {
31             $pdo->majFraisForfait($idVisiteur, $mois, $lesFrais);
32             $messageAlert = 'La modification des éléments forfaitisés à bien été prise en compte';
33         } else {
34             ajouterErreur('Les valeurs des frais doivent être numériques');
35             include 'vues/v_erreurs.php';
36         }
37         break;
38     case 'validerCreationFrais':
39         $dateFrais = filter_input(INPUT_POST, 'dateFrais', FILTER_SANITIZE_STRING);
40         $libelle = filter_input(INPUT_POST, 'libelle', FILTER_SANITIZE_STRING);
41         $montant = filter_input(INPUT_POST, 'montant', FILTER_VALIDATE_FLOAT);
42         valideInfosFrais($dateFrais, $libelle, $montant);
43         if (nbErreurs() != 0) {
44             include 'vues/v_erreurs.php';
45         } else {
46             $pdo->creeNouveauFraisHorsForfait(
47                 $idVisiteur,
48                 $mois,
49                 $libelle,
50                 $dateFrais,
51                 $montant
52             );
53             $messageAlert = 'La création de l\'élément hors forfait à bien été prise en compte';
54         }
55         break;
56     case 'supprimerFrais':
57         $idFrais = filter_input(INPUT_GET, 'idFrais', FILTER_SANITIZE_STRING);
58         $pdo->supprimerFraisHorsForfait($idFrais);
59         $messageAlert = 'La suppression de l\'élément hors forfait à bien été prise en compte';
60         break;
61 }
62 $lesFraisHorsForfait = $pdo->getLesFraisHorsForfait($idVisiteur, $mois);
63 $lesFraisForfait = $pdo->getLesFraisForfait($idVisiteur, $mois);
64 require 'vues/visiteur/v_listeFraisForfait.php';
65 require 'vues/visiteur/v_listeFraisHorsForfait.php';
```

Fichier contrôleur

Intallation du phpUnit et réalisation de tests

Initialisation de composer et installation de phpUnit :

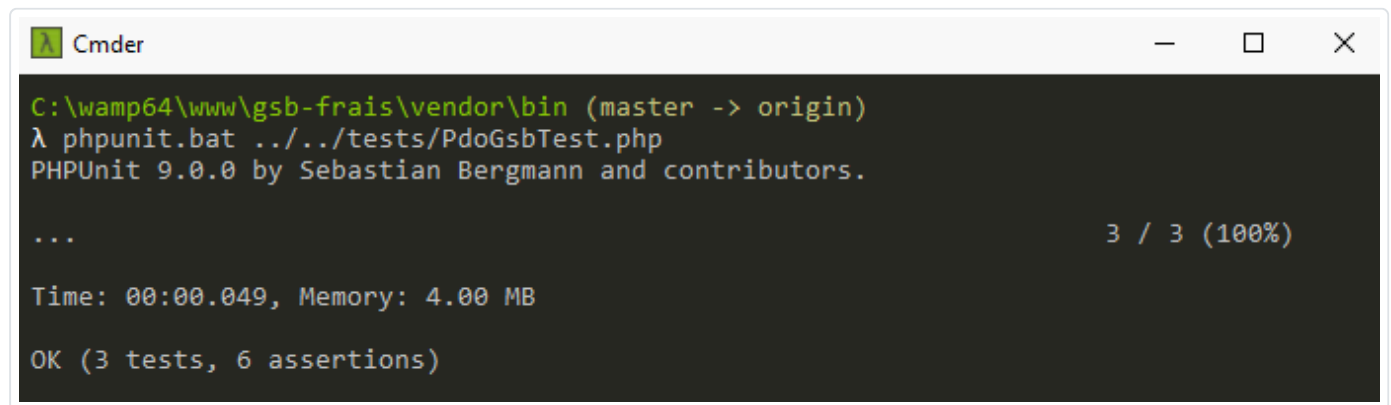
```
PS C:\wamp64\www\gsb-frais> composer init -n
PS C:\wamp64\www\gsb-frais> composer require --dev phpunit/phpunit ^9
```

Réalisation de tests sur la classe PdoGsb :

gsb-frais > tests >  PdoGsbTest.php > ...

```
1  <?php
2
3  namespace Tests\App;
4
5  use App\PdoGsb;
6  use PHPUnit\Framework\TestCase;
7
8  // session_start();
9
10 class PdoGsbTest extends TestCase {
11
12     public function testgetInfosVisiteurIsArrayOrNot()
13     {
14         $pdo = PdoGsb::getPdoGsb();
15
16         $this->assertIsArray($pdo->getInfosVisiteur('pbentot', 'doyw1'));
17         $this->assertIsNotArray($pdo->getInfosVisiteur('pbentot', 'azerty'));
18     }
19
20
21     public function testgetInfosVisiteurIdIsEqualOrNot()
22     {
23         $pdo = PdoGsb::getPdoGsb();
24         $infosVisiteur = $pdo->getInfosVisiteur('pbentot', 'doyw1');
25
26         $this->assertEquals('b13', $infosVisiteur['id']);
27         $this->assertNotEquals('a1', $infosVisiteur['id']);
28     }
29
30     public function testestPremierFraisMoisOrNot()
31     {
32         $pdo = PdoGsb::getPdoGsb();
33
34         $this->assertTrue($pdo->estPremierFraisMois('b13', '202010'));
35         $this->assertFalse($pdo->estPremierFraisMois('b13', '202009'));
36     }
37 }
```

Affichage des résultats :



```
C:\wamp64\www\gsb-frais\vendor\bin (master -> origin)
λ phpunit.bat ../../tests/PdoGsbTest.php
PHPUnit 9.0.0 by Sebastian Bergmann and contributors.

...
3 / 3 (100%)

Time: 00:00.049, Memory: 4.00 MB

OK (3 tests, 6 assertions)
```

Description de la mise en production

Pour rendre cette application disponible en ligne, voici les différentes manipulations effectuées :

1. Ajout d'un sous domaine "gsb.ostyl.fr" ayant un dossier racine différent de celui de mon nom de domaine principal "ostyl.fr".
2. Régénération du certificat SSL pour disposer d'une connection sécurisés sur le sous domaine "gsb".
3. Modification des informations de connexion à la base de données dans les fichiers `includes/class.pdogsb.inc.php` et `tests/gendatas/majGSB.php`.
4. Copie des dossiers et fichiers vers le répertoire "gsb", via le logiciel FileZilla.
5. Modification du fichier SQL avant l'import : suppression de la partie "Administration de la base de données", qui est utile en local uniquement.
6. Création d'une nouvelle base de données liée à l'hébergement web.
7. Import de la base de données via phpMyAdmin.
8. Lancement du script `gendatas/majGSB.php` permettant de générer des données fictives dans la base, puis suppression du dossier `gendatas`, afin que ce script ne soit pas exécuté à nouveau.
9. Création d'un fichier `.htaccess` permettant de rediriger automatiquement les utilisateurs vers le "https", et empêche l'accès aux répertoires et fichiers qui pourraient compromettre la sécurité de l'application.

Conclusion

Grâce à ce projet, je me suis rendu compte de l'importance de travailler sur un IDE avec des extensions comme Xdebug qui permettent de déboguer le code de manière efficace.

J'ai également pris conscience de la praticité de l'architecture MVC dans le développement web. Ce patron de conception permet de mieux s'y retrouver, les fichiers sont bien organisés dans différents répertoires correspondant aux différentes parties logiques.