PPE 3.3

GSB - Gestion des Professionnels de santé
(GSB-AppliFrais-MVC)

Auteur : Maxime Lalange – Laura Sampaolo
Date de création : 21/12/2018
Version : 1.0.0

SOMMAIRE

Contexte de la situation professionnelle

Le besoin

Le suivi des frais est actuellement géré de plusieurs façons selon le laboratoire d'origine des visiteurs. Le laboratoire souhaite uniformiser cette gestion. L'application doit permettre d'enregistrer tout frais engagés, aussi bien pour l'activité directe (déplacement, restauration et hébergement) que pour les activités annexe (événementiel, conférence, 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).

Les objectifs / L'existant

Architecture
L'application respectera l'architecture des scripts fournis concernant la gestion de l'enregistrement des frais engagés par les visiteurs.

Ergonomie
Les pages fournies ont été définies suite à une consultation. Elles constituent une référence ergonomique. Des améliorations ou variations peuvent être proposées.

Codage
Le document "ApplisWeb-NormesDevelpt" présente des règles de bonnes pratiques de développement utilisées par le service informatique de GSB pour encadrer le développement d'applications en PHP et en faciliter la maintenance ; les deux applications fournies (GSB-AppliFrais et GSB-AppliFrais-MVC) s'efforcent de les mettre en œuvre.
Les éléments à fournir devront respecter le nommage des fichiers, variables et paramètres, ainsi que les codes couleurs et la disposition des éléments déjà fournis.

Environnement
Le langage de script côté serveur doit être le même que celui utilisé dans les pages fournies. L'utilisation de bibliothèques, API ou Frameworks est à l'appréciation du prestataire.

Modules
L'application présente deux modules :

- Enregistrement et suivi par les visiteurs (code fourni)
- Enregistrement des opérations par les comptables

Documentation
La documentation devra présenter l'arborescence des pages pour chaque module, le descriptif des éléments, classes et bibliothèques utilisées, la lise des Frameworks ou bibliothèque externes utilisés.

Responsabilités
Le commanditaire fournira à la demande toute information sur le contexte nécessaire à la production de l'application.
Le commanditaire fournira une documentation et des sources exploitables pour la phase de test (base de données, modélisation, ..)
Le prestataire est à l'initiative de toute proposition technique complémentaire. Le prestataire fournira un système opérationnel, une documentation technique permettant un transfert de compétences et un mode opératoire propre à chaque module.

Environnement de développement

IDE mis en œuvre

Sublime Text
Sublime Text est un éditeur de texte générique codé en C++ et Python, disponible sur Windows, Mac et Linux. Le logiciel a été conçu tout d'abord comme une extension pour Vim, riche en fonctionnalités.

IntelliJ
IntelliJ IDEA est un IDE Java commercial développé par JetBrains. Il est fréquemment appelé par le simple nom d'« IntelliJ », « IDEA » ou « IDJ ».

Netbeans
NetBeans est un environnement de développement intégré, placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2. 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

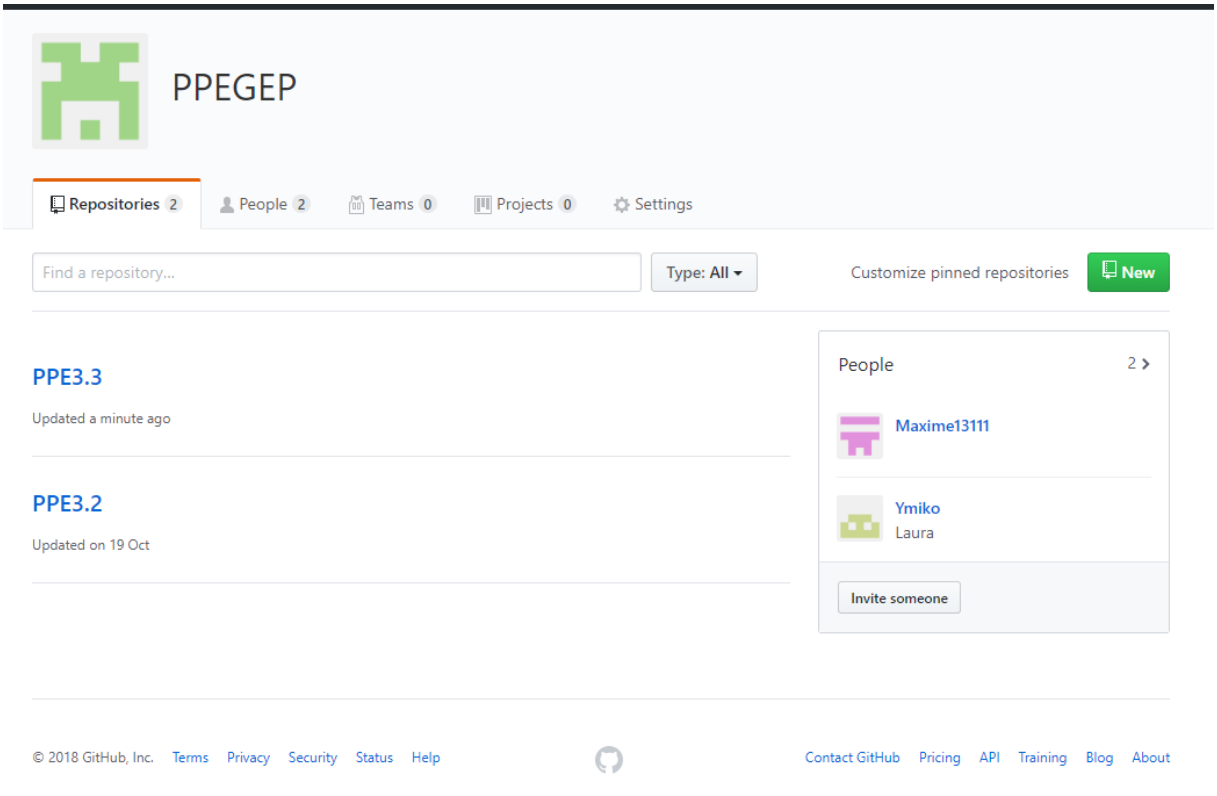
Langages utilisés

PHP
PHP: Hypertext Preprocessor, plus connu sous son sigle PHP , est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

Méthode de travail en projet
La méthode de travail en projet fonctionne sur une répartition des tâches et du travail équitablement entre les collaborateurs. Une mise au point est faite sur le travail fourni et chacun explique sa production ainsi que les améliorations apportées. Si une des personnes du projet pense d’une information qu’elle est inutile, celui-ci procède à un discours argumenté et justifié.

Environnement de travail collaboratif

Ce projet a été conçu en mode collaboratif, partagé sur la plateforme GitHub.



Github est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de version Git. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet.

Modélisation fonctionnelle

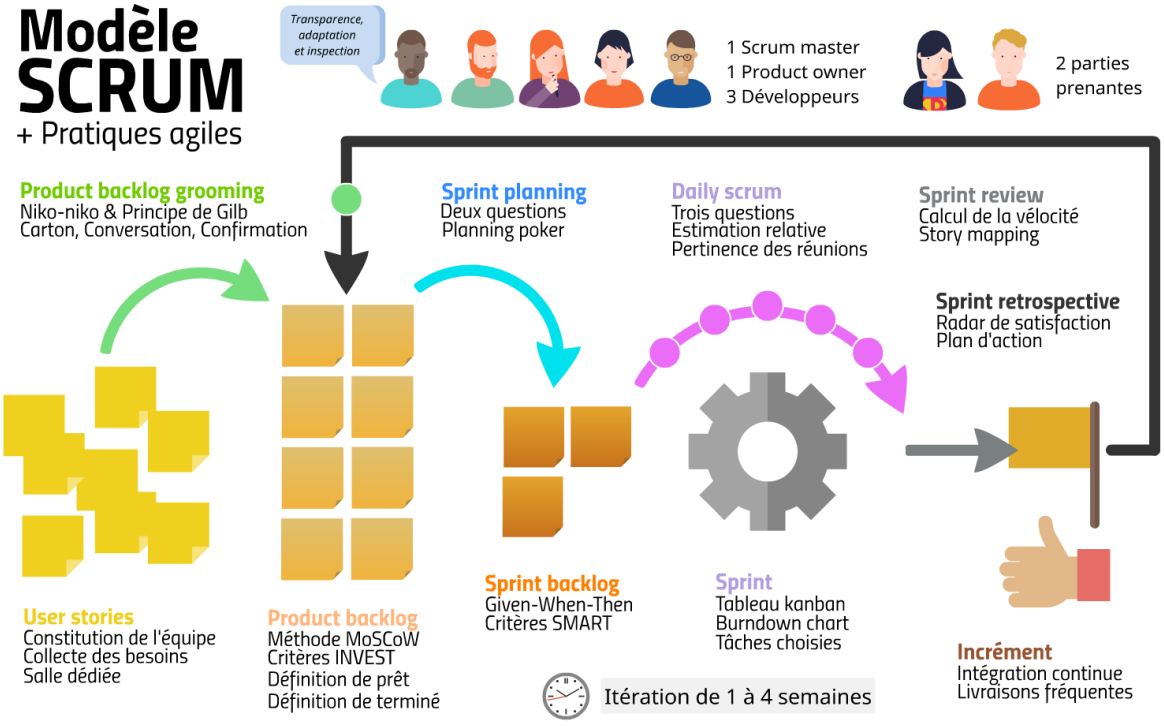
Méthodologie mis en œuvre / Expression des besoins / Modélisation fonctionnelle en UML

Présentation SCRUM

Scrum est un schéma d'organisation de développement de produits complexes. Il est défini par ses créateurs comme un « cadre de travail holistique itératif qui se concentre sur les buts communs en livrant de manière productive et créative des produits de la plus grande valeur possible ».

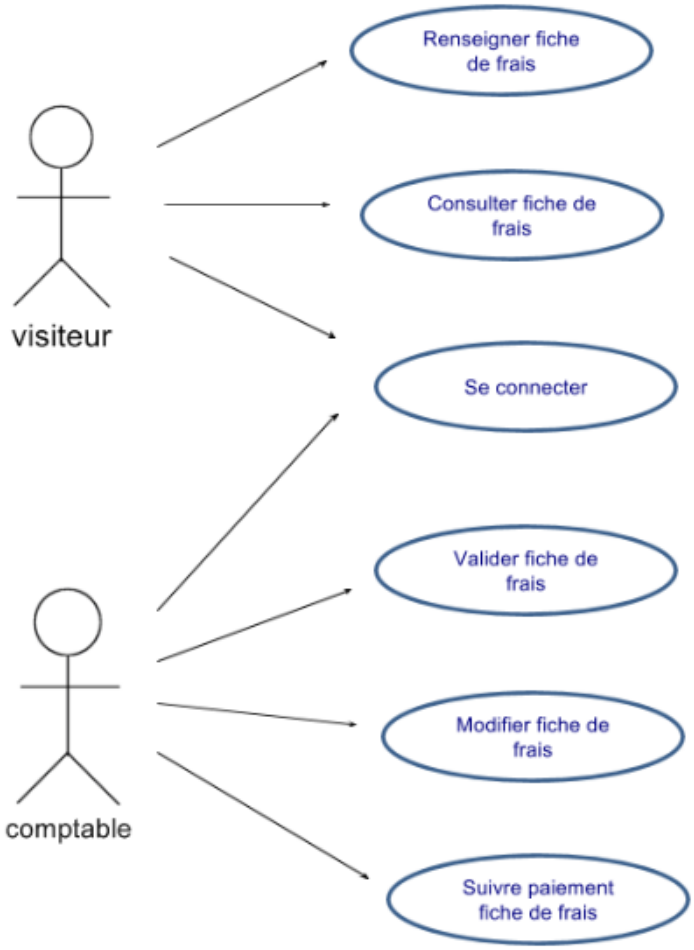
Présentation méthode AGILE

La méthode agile est une méthode de gestion et de développement de projets ou programmes informatiques. Elle vise à satisfaire les besoins du client au terme du contrat de développement. Elle fonctionne sur la base de l'itératif et l'incrémental.



Présentation UML

Le Langage de Modélisation Unifié, de l'anglais Unified Modeling Language, est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet.



Nouvelle version

PROJET : Application web de gestion des frais	**Description cas d'utilisation**

Nom cas d'utilisation : Se connecter
Acteur déclencheur : Visiteur médical ou Comptable
Pré conditions : Néant
Post conditions : L'utilisateur est reconnu visiteur médical ou comptable
Scénario nominal : • 1 – Le système affiche un formulaire de connexion. • 2 – L'utilisateur saisit son login et son mot de passe et valide. • 3 – Le système contrôle les informations de connexion, informe que le profil Visiteur ou Comptable est activé, et maintient affichée l'identité du visiteur médical / comptable connecté.
Exceptions : • 3 – a : Le nom et / ou le mot de passe n'est pas valide. 3 – a.1 Le système en informe l'utilisateur. Retour à l'étape 1 • 4 – L'utilisateur demande à se déconnecter. • 5 – Le système déconnecte l'utilisateur.
Contraintes :
Questions ouvertes :

PROJET : Application web de gestion des frais	**Description cas d'utilisation**

Nom cas d'utilisation : Renseigner fiche de frais
Acteur déclencheur : Visiteur médical
Pré conditions : Visiteur médical authentifié.
Post conditions : Néant.
Scénario nominal : • 1 – L'utilisateur demande à saisir un ou plusieurs frais pour le mois courant. • 2 – Le système retourne les frais actuellement saisis – éléments forfaitisés et hors forfait – pour le mois courant. • 3 – L'utilisateur modifie une ou des valeurs des frais au forfait et demande la validation. • 4 – Le système enregistre cette ou ces modifications et retourne ces valeurs à jours. • 5 – L'utilisateur ajoute un nouveau frais hors forfait en renseignant les différents champs – date d'engagement, libellé, montant – et valide. • 6 – Le système enregistre la ligne de frais hors forfait.
Exceptions : • 2.a- C'est la première saisie pour le mois courant. Si ce n'est pas encore fait, le système clôt la fiche du mois précédent et crée une nouvelle fiche de frais avec des valeurs initialisées à 0. Retour à 3. • 4.a. Une valeur modifiée n'est pas numérique : le système indique 'Valeur numérique attendue '. Retour à 3. • 6.a Un des champs n'est pas renseigné : le système indique : 'Le champ date (ou libellé ou montant) doit être renseigné'. • 6.b La date d'engagement des frais hors forfait est invalide : le système indique 'La date d'engagement doit être valide'. Retour à 5. • 6.c La date d'engagement des frais hors forfait date de plus d'un an. Le système indique 'La date d'engagement doit se situer dans l'année écoulée'. Retour à 5. • 7. L'utilisateur sélectionne un frais hors forfait pour suppression. • 8. Le système enregistre cette suppression après une demande de confirmation.
Contraintes :

PROJET : Application web de gestion des frais	**Description cas d'utilisation**

Nom cas d'utilisation : Consulter mes fiches de frais
Acteur déclencheur : Visiteur médical
Pré conditions : Visiteur médical authentifié
Post conditions : néant
Scénario nominal : • 1 – L'utilisateur demande à consulter ses frais. • 2 – Le système invite à sélectionner un mois donné. • 3 – L'utilisateur sélectionne un mois donné, puis valide. • 4 – Le système affiche l'état de la fiche de frais avec la date associée, les éléments forfaitisés – quantité pour chaque type de frais forfaitisé - et non forfaitisés – montant, libellé et date d'engagement - existant pour la fiche de frais du mois demandé.
Exceptions :
Contraintes :
Questions ouvertes : La sélection d'un mois sera facilitée par l'IHM. Il est possible de proposer les mois pour lesquels le visiteur médical connecté dispose d'une fiche de frais. On pourra se restreindre à remonter jusqu'au début de l'année civile précédente.

PROJET : Application web de gestion des frais	**Description cas d'utilisation**

Nom cas d'utilisation : Valider fiche de frais
Acteur déclencheur : Comptable
Pré conditions : Utilisateur Comptable authentifié Toutes les fiches du mois qui vient de s'achever sont clôturées par un script. Le comptable dispose de tous les éléments pour évaluer la conformité des frais forfaitisés.
Post conditions : néant
Scénario nominal : 1. L'utilisateur demande à valider les fiches de frais 2. Le système propose de choisir le visiteur et le mois concernés 3. L'utilisateur sélectionne les informations et valide 4. Le système affiche le détail de la fiche de frais – frais forfaitisés et hors forfait 5. L'utilisateur actualise les informations des frais forfaitisés. 6. Le système indique que la modification a été prise en compte et affiche les informations actualisées. 7. L'utilisateur demande la suppression des lignes de frais hors forfait non valides 8. Le système modifie le libellé en ajoutant en début le texte « REFUSE : ». 9. L'utilisateur valide la fiche. 10. Le système passe la fiche à l'état « Validée » et met à jour la date de modification de la fiche.
Exceptions : • 4-a : Aucune fiche de frais n'existe, le système affiche: "Pas de fiche de frais pour ce visiteur ce mois". Retour au 2. • 7.a : L'utilisateur demande le report des frais hors forfait pour lesquels une facture acquittée n'a pas été reçue dans les temps. • 8.a : Le système ajoute la ligne hors forfait dans la fiche du mois suivant et la supprime de la fiche courante. Si la fiche du mois suivant n'existe pas, le système génère une nouvelle fiche pour le visiteur en cours de traitement et pour le mois suivant. Cette nouvelle fiche a des valeurs à 0 pour les frais forfaitisés et est dans l'état « Saisie en cours ». Retour au 9 • 8.b : le texte ainsi complété dépasse la taille maximale du champ « libelle » : le texte est tronqué par la fin au nombre de caractères du champ « libelle »

PROJET : Application web de gestion des frais	**Description cas d'utilisation**
Nom cas d'utilisation : Suivre le paiement fiche de frais	
Acteur déclencheur : Comptable	
Pré conditions : Utilisateur Comptable authentifié	
Post conditions : néant	

Scénario nominal :

1. L'utilisateur demande à suivre le paiement les fiches de frais.
2. Le système propose de choisir une fiche de frais parmi celles à valider et mises en paiement.
3. L'utilisateur sélectionne les informations et valide.
4. Le système affiche le détail de la fiche de frais – frais forfaitisés et hors forfait.
5. L'utilisateur « Met en paiement » la fiche de frais.
6. Le système modifie l'état de la fiche à « Mise en paiement » et met à jour la date de modification.

Exceptions :

4-a. Aucune fiche de frais n'existe, le système affiche: "Pas de fiche de frais pour ce visiteur ce mois". Retour au 2
5.a. L'utilisateur indique que la fiche a été effectivement payée
6.a. Le système modifie l'état de la fiche à « Remboursée » et enregistre la date de modification

Architecture applicative

Présentation MVC

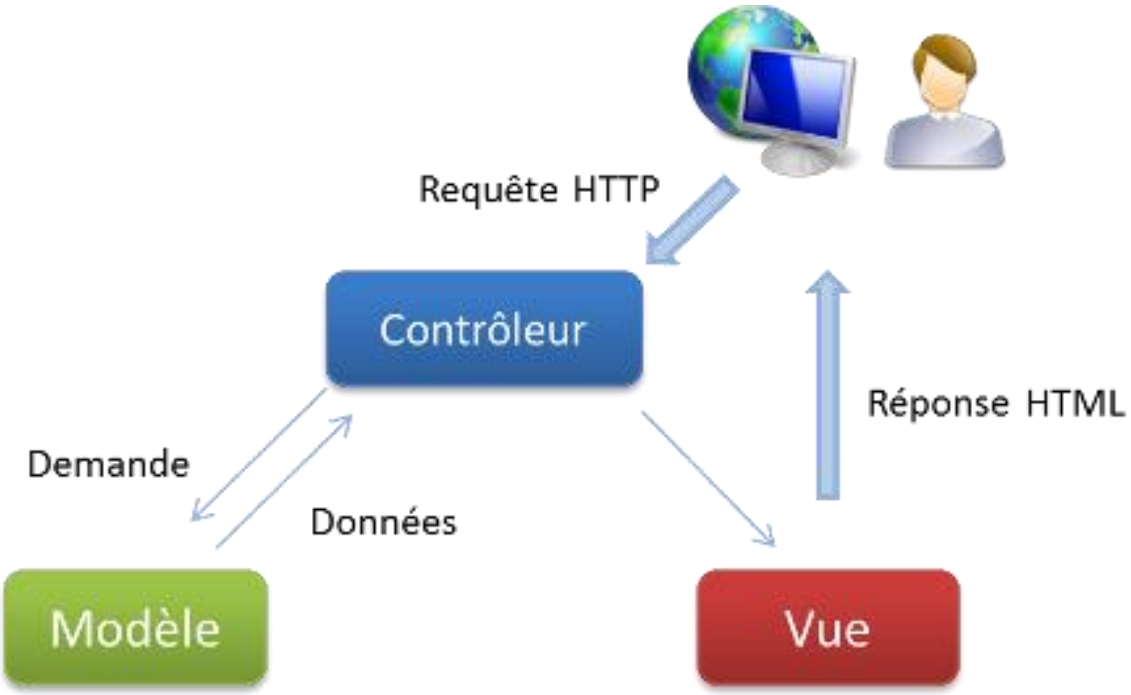
Modèle-vue-contrôleur ou MVC est un motif d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs

Répartition des rôles

Le **modèle** représente les données et les règles métiers. C'est dans ce composant que s'effectuent les traitements liés au cœur du métier.

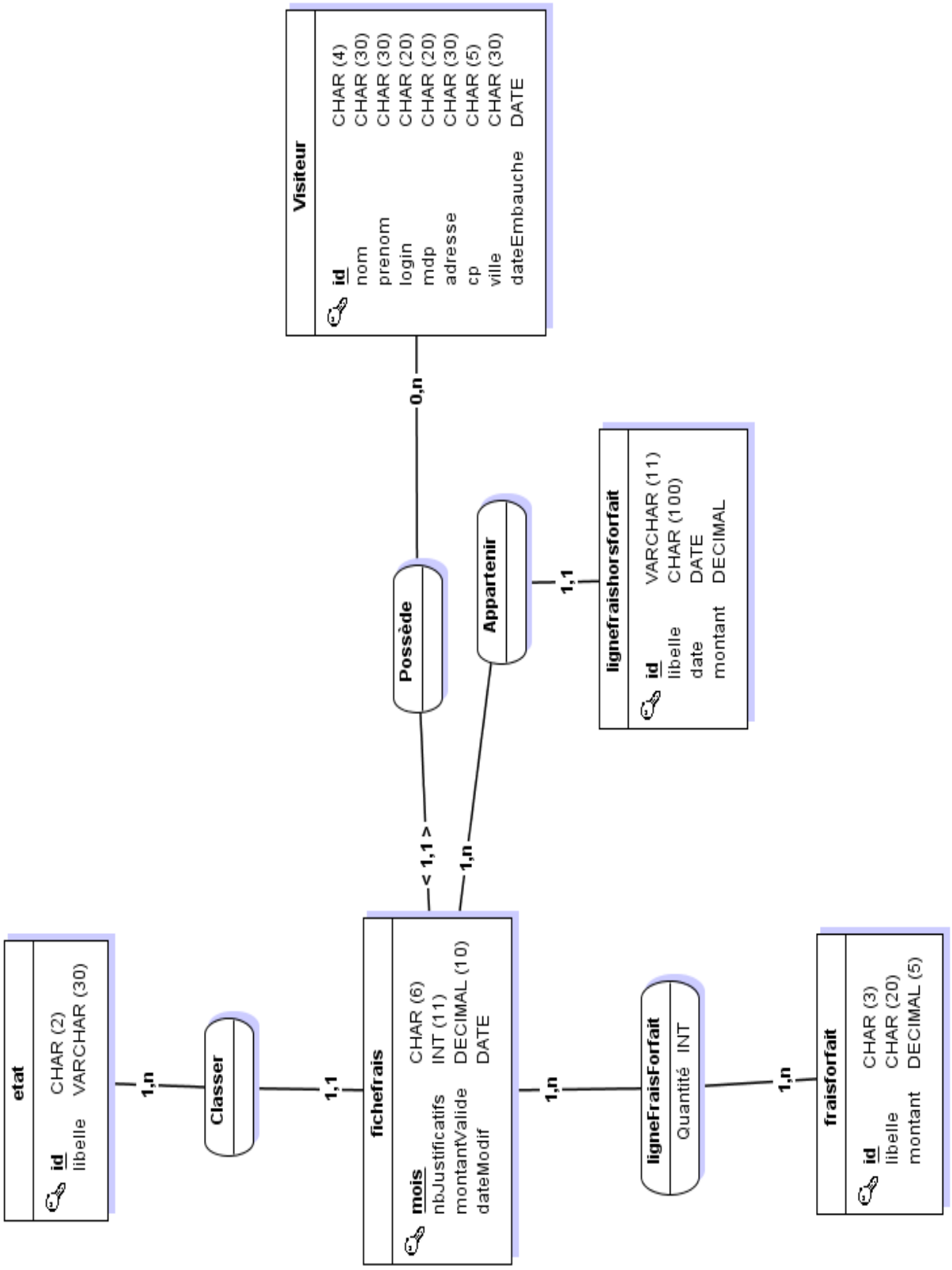
La **vue** correspond à l'IHM. Elle présente les données et interagit avec l'utilisateur. Dans le cadre des applications Web, il s'agit d'une interface HTML, mais n'importe quel composant graphique peut jouer ce rôle.

Le **contrôleur**, quant à lui, se charge d'intercepter les requêtes de l'utilisateur, d'appeler le modèle puis de rediriger vers la vue adéquate. Il ne doit faire aucun traitement. Il ne fait que de l'interception et de la redirection.

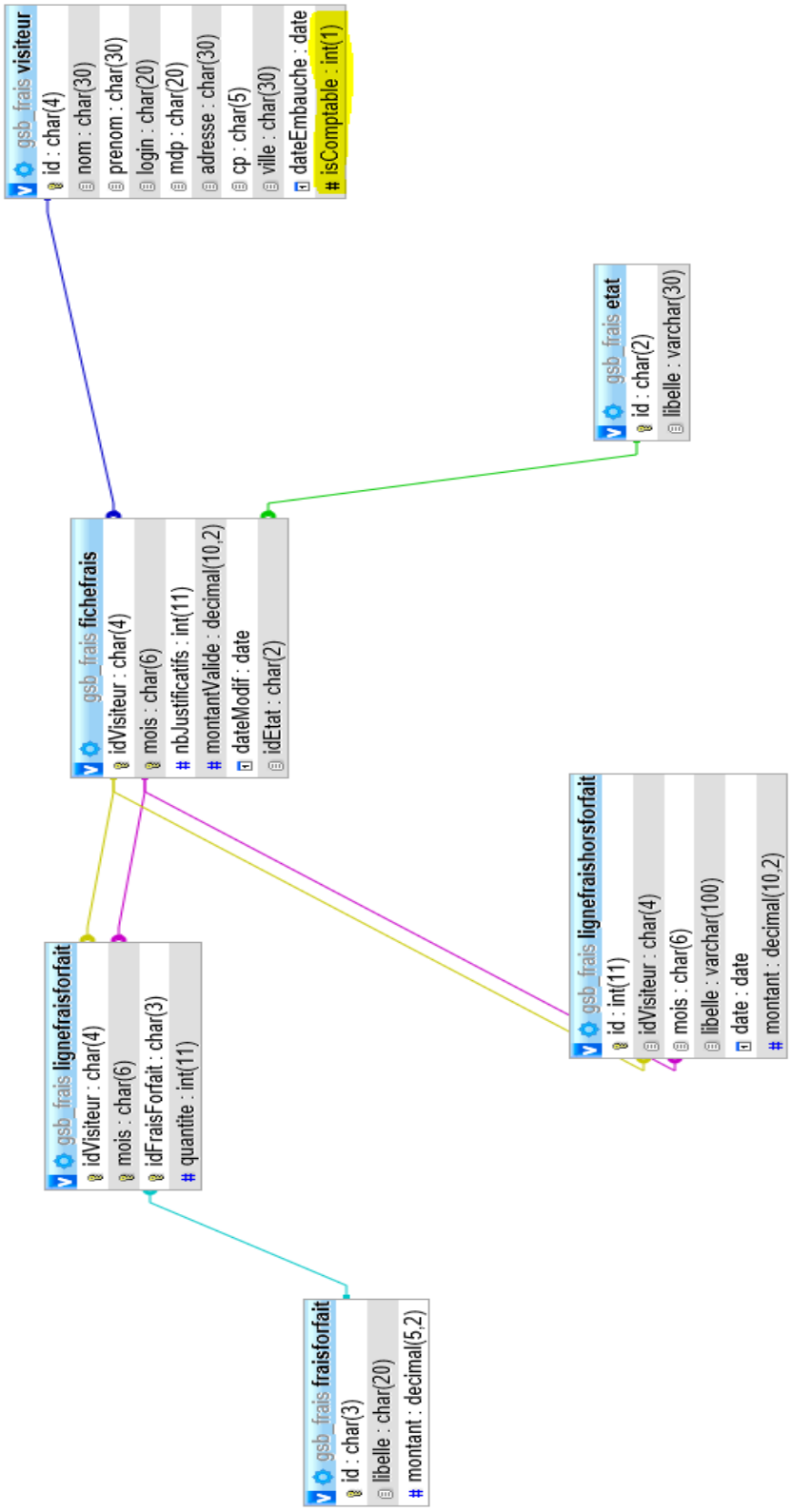


Persistance des données

MCD avant



MLD après


gsb_frais visiteur
id : char(4)
nom : char(30)
prenom : char(30)
login : char(20)
mdp : char(20)
adresse : char(30)
cp : char(5)
ville : char(30)
dateEmbauche : date
isComptable : int(1)
gsb_frais etat
id : char(2)
libelle : varchar(30)
gsb_frais fichefrais
idVisiteur : char(4)
mois : char(6)
nbJustificatifs : int(11)
montantValide : decimal(10,2)
dateModif : date
idEtat : char(2)
gsb_frais lignefraisforfait
idVisiteur : char(4)
mois : char(6)
idFraisForfait : char(3)
quantite : int(11)
gsb_frais lignefraishorsforfait
id : int(11)
idVisiteur : char(4)
mois : char(6)
libelle : varchar(100)
date : date
montant : decimal(10,2)
gsb_frais fraisforfait
id : char(3)
libelle : char(20)
montant : decimal(5,2)

Extrait de code

```sql
-- Export de la structure de la base pour gsb_frais
CREATE DATABASE IF NOT EXISTS gsb_frais /*!40100 DEFAULT CHARACTER SET latin1 */;
USE gsb_frais;

-- Export de la structure de la table gsb_frais. etat
CREATE TABLE IF NOT EXISTS etat (
  id char(2) NOT NULL,
  libelle varchar(30) DEFAULT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Export de données de la table gsb_frais.etat : ~4 rows (environ)
/*!40000 ALTER TABLE etat DISABLE KEYS */;
INSERT INTO etat (id, libelle) VALUES
    ('CL', 'Saisie clôturée'),
    ('CR', 'Fiche créée, saisie en cours'),
    ('RB', 'Remboursée'),
    ('VA', 'Validée et mise en paiement');
/*!40000 ALTER TABLE etat ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. fichefrais
CREATE TABLE IF NOT EXISTS fichefrais (
  idVisiteur char(4) NOT NULL,
  mois char(6) NOT NULL,
  nbJustificatifs int(11) DEFAULT NULL,
  montantValide decimal(10,2) DEFAULT NULL,
  dateModif date DEFAULT NULL,
  idEtat char(2) DEFAULT 'CR',
  PRIMARY KEY (idVisiteur,`mois`),
  KEY idEtat (idEtat),
  CONSTRAINT fichefrais_ibfk_1 FOREIGN KEY (idEtat) REFERENCES etat (id),
  CONSTRAINT fichefrais_ibfk_2 FOREIGN KEY (idVisiteur) REFERENCES utilisateur (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Export de données de la table gsb_frais.fichefrais : ~0 rows (environ)
/*!40000 ALTER TABLE fichefrais DISABLE KEYS */;
/*!40000 ALTER TABLE fichefrais ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. fraisforfait
CREATE TABLE IF NOT EXISTS fraisforfait (
  id char(3) NOT NULL,
  libelle char(20) DEFAULT NULL,
  montant decimal(5,2) DEFAULT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
-- Export de données de la table gsb_frais.fraisforfait : ~4 rows (environ)
/*!40000 ALTER TABLE fraisforfait DISABLE KEYS */;
INSERT INTO fraisforfait (id, libelle, montant) VALUES
    ('ETP', 'Forfait Etape', 110.00),
    ('KM', 'Frais Kilométrique', 0.62),
    ('NUI', 'Nuitée Hôtel', 80.00),
    ('REP', 'Repas Restaurant', 25.00);
/*!40000 ALTER TABLE fraisforfait ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. lignefraisforfait
CREATE TABLE IF NOT EXISTS lignefraisforfait (
  idVisiteur char(4) NOT NULL,
  mois char(6) NOT NULL,
  idFraisForfait char(3) NOT NULL,
  quantite int(11) DEFAULT NULL,
  PRIMARY KEY (idVisiteur,`mois`,`idFraisForfait`),
  KEY idFraisForfait (idFraisForfait),
  CONSTRAINT lignefraisforfait_ibfk_1 FOREIGN KEY (idVisiteur, mois) REFERENCES fichefrais (idVisiteur, mois),
  CONSTRAINT lignefraisforfait_ibfk_2 FOREIGN KEY (idFraisForfait) REFERENCES fraisforfait (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Export de données de la table gsb_frais.lignefraisforfait : ~0 rows (environ)
/*!40000 ALTER TABLE lignefraisforfait DISABLE KEYS */;
/*!40000 ALTER TABLE lignefraisforfait ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. lignefraishorsforfait
CREATE TABLE IF NOT EXISTS lignefraishorsforfait (
  id int(11) NOT NULL AUTO_INCREMENT,
  idVisiteur char(4) NOT NULL,
  mois char(6) NOT NULL,
  libelle varchar(100) DEFAULT NULL,
  date date DEFAULT NULL,
  montant decimal(10,2) DEFAULT NULL,
  PRIMARY KEY (id),
  KEY idVisiteur (idVisiteur,`mois`),
  CONSTRAINT lignefraishorsforfait_ibfk_1 FOREIGN KEY (idVisiteur, mois) REFERENCES fichefrais (idVisiteur, mois)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Export de données de la table gsb_frais.lignefraishorsforfait : ~0 rows (environ)
/*!40000 ALTER TABLE lignefraishorsforfait DISABLE KEYS */;
/*!40000 ALTER TABLE lignefraishorsforfait ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. typeutilisateur
CREATE TABLE IF NOT EXISTS typeutilisateur (
  id varchar(1) NOT NULL,
  libelle varchar(50) DEFAULT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Export de données de la table gsb_frais.typeutilisateur : ~2 rows (environ)
/*!40000 ALTER TABLE typeutilisateur DISABLE KEYS */;
INSERT INTO typeutilisateur (id, libelle) VALUES
    ('1', 'Visiteur'),
    ('2', 'Comptable');
/*!40000 ALTER TABLE typeutilisateur ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. utilisateur
CREATE TABLE IF NOT EXISTS utilisateur (
  id char(4) NOT NULL,
  id_type varchar(1) NOT NULL,
  nom char(30) DEFAULT NULL,
  prenom char(30) DEFAULT NULL,
  login char(20) DEFAULT NULL,
  mdp char(20) DEFAULT NULL,
  adresse char(30) DEFAULT NULL,
  cp char(5) DEFAULT NULL,
  ville char(30) DEFAULT NULL,
  dateEmbauche date DEFAULT NULL,
  PRIMARY KEY (id),
  KEY id_type (id_type),
  CONSTRAINT FK_utilisateur_typeutilisateur FOREIGN KEY (id_type) REFERENCES typeutilisateur (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

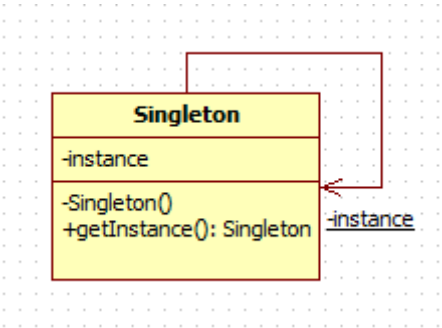
```sql
-- Export de données de la table gsb_frais.typeutilisateur : ~2 rows (environ)
/*!40000 ALTER TABLE typeutilisateur DISABLE KEYS */;
INSERT INTO typeutilisateur (id, libelle) VALUES
    ('1', 'Visiteur'),
    ('2', 'Comptable');
/*!40000 ALTER TABLE typeutilisateur ENABLE KEYS */;

-- Export de la structure de la table gsb_frais. utilisateur
CREATE TABLE IF NOT EXISTS utilisateur (
  id char(4) NOT NULL,
  id_type varchar(1) NOT NULL,
  nom char(30) DEFAULT NULL,
  prenom char(30) DEFAULT NULL,
  login char(20) DEFAULT NULL,
  mdp char(20) DEFAULT NULL,
  adresse char(30) DEFAULT NULL,
  cp char(5) DEFAULT NULL,
  ville char(30) DEFAULT NULL,
  dateEmbauche date DEFAULT NULL,
  PRIMARY KEY (id),
  KEY id_type (id_type),
  CONSTRAINT FK_utilisateur_typeutilisateur FOREIGN KEY (id_type) REFERENCES typeutilisateur (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Accès aux données

Présentation PDO

PHP Data Objects (PDO) est une extension définissant l'interface pour accéder à une base de données avec PHP.
PDO constitue une couche d'abstraction qui intervient entre l'application PHP et un système de gestion de base de données tel que MySQL, PostgreSQL ou MariaDB par exemple.

Présentation Design Pattern Singleton



Le **singleton** est un patron de conception (**design pattern**) dont l'objectif est de restreindre l'instanciation d'une classe à un seul objet (ou bien à quelques objets seulement). Il est utilisé lorsqu'on a besoin exactement d'un objet pour coordonner des opérations dans un système.

Mise en œuvre

```php
class PdoGsb{
        private static $serveur='mysql:host=localhost';
        private static $bdd='dbname=gsb_frais';
        private static $user='root' ;
        private static $mdp='' ;
        private static $monPdo;
        private static $monPdoGsb=null;
/**
 * Constructeur privé, crée l'instance de PDO qui sera sollicitée
 * pour toutes les méthodes de la classe
 */
        private function __construct(){
            PdoGsb::$monPdo = new PDO(PdoGsb::$serveur.';'.PdoGsb::$bdd, PdoGsb::$user, PdoGsb::$mdp);
            PdoGsb::$monPdo->query("SET CHARACTER SET utf8");
        }
        public function _destruct(){
            PdoGsb::$monPdo = null;
        }
/**
 * Fonction statique qui crée l'unique instance de la classe

 * Appel : $instancePdoGsb = PdoGsb::getPdoGsb();

 * @return l'unique objet de la classe PdoGsb
 */
        public  static function getPdoGsb(){
                if(PdoGsb::$monPdoGsb==null){
                        PdoGsb::$monPdoGsb= new PdoGsb();
                }
                return PdoGsb::$monPdoGsb;
        }
```

Déploiement, mise en production de l'application

VM Windows server

Une machine virtuelle (anglais virtual machine, abr. VM) est une illusion d'un appareil informatique créée par un logiciel d'émulation. Le logiciel d'émulation simule la présence de ressources matérielles et logicielles telles que la mémoire, le processeur, le disque dur, voire le système d'exploitation et les pilotes, permettant d'exécuter des programmes dans les mêmes conditions que celles de la machine simulée.

WampServer

WampServer est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans avoir à se connecter à un serveur externe) des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.

Apache

Le logiciel Apache est un serveur HTTP en Open Source utilisé principalement sur les hébergements Internet en Linux, bien qu'il soit également utilisable en Windowsm Unix ou OS X. C'est actuellement le plus utilisé sur le WEB. Différentes fonctionnalités sont implantées comme la possibilité d'utiliser un seul serveur Internet pour héberger plusieurs sites.

SQL

SQL est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles

Tests fonctionnels

N°	Action	Résultat attendu	OK/KO
1	Le Comptable saisit son identifiant [$login] et son mot de passe [$mdp] puis clique sur bouton « Valider ».	Le système actualise la page d'accueil et le menu apparait.	KO
2	Le Comptable sélectionne « Validation fiches de frais »	Le système fait apparaitre la liste des visiteurs médicaux inscrit en base.	KO
3	Le Comptable sélectionne un visiteur et clique sur le bouton « Valider ».	Le système fait apparaître la liste du/des mois par année(s) pour lequel/lesquels le visiteur médical à créer une/des fiche(s) de frais.	KO
4	Le Comptable sélectionne un mois par année et clique sur le bouton « Valider ».	Le système actualise la page et retourne les tableaux des frais forfaits et hors forfaits du mois par année concerné pour le visiteur concerné.	KO
5	Le Comptable saisi le nombre de justificatifs reçu dans [$nbjustificatifs].	Le système retourne le nombre déjà enregistré de justificatif et retourne la saisie du comptable.	KO
6	Le Comptable vérifie le montant calculé par l'application et « Valide »	Le système indique si le montant total est à jour par rapport aux derniers enregistrements du visiteur donné. Le système enregistre les données dans la base de données, et indique « Le montant est à jour. »	KO
7	Le Comptable valide un frais hors forfait en cliquant sur « Valider ce frais »	Le système met à jour la base de données et ajoute à « libelle » la mention « ACCEPTE – »	KO
8	Le Comptable refuse un frais hors forfait en cliquant sur « Refuser ce frais »	Le système met à jour la base de données et ajoute à « libelle » la mention « REFUSE – »	KO
9	Le Comptable sélectionne « Validation fiches de frais ».	Le système fait apparaitre la liste de sélection du visiteur concerné	KO
10	Le Comptable sélectionne un visiteur et clique sur le bouton « Valider ».	Le système fait apparaître la liste de sélection du mois concerné	KO
11	Le Comptable sélectionne le mois et clique sur le bouton « Valider ».	Le système actualise la page et retourne l'état de frais en détail du mois concerné pour le visiteur et propose la validation du montant total.	KO
12	Le Comptable valide le montant total de frais en cliquant sur le bouton « Valider ».	Le système met à jour la base de données « libelle », « état », « montantValide » et actualise la page.	KO

Conclusion

Améliorations possibles - effectuées

Requêtes préparées

Une requête préparée, c'est en quelque sorte une requête stockée en mémoire (pour la session courante), et que l'on peut exécuter à loisir.

Bootstrap

Aujourd'hui, pour avoir un site web optimal, il faut qu'il soit responsive, c'est-à-dire modulable selon tous les types d'écrans (smartphone, tablette tactile, ordinateur, TV). L'enjeu est que la structure du site reste cohérente.

Requêtes asynchrones

Ajax permet de modifier partiellement la page affichée par le navigateur pour la mettre à jour sans avoir à recharger la page entière. Par exemple le contenu d'un champ de formulaire peut être changé, sans avoir à recharger la page avec le titre, les images, le menu, etc.

Ajax permet ainsi d'effectuer des traitements sur le poste client (avec JavaScript) à partir d'informations prises sur le serveur. Cela répartit la charge de traitement.

Auparavant, toutes les modifications de pages étaient faites sur le serveur ce qui nécessitait des échanges maintenant inutiles.

C'est une technique qui fait usage des éléments suivants:

- HTML.
- CSS (Cascading Style-Sheet) pour la présentation de la page.
- JavaScript
- XMLHttpRequest lit des données ou fichiers sur le serveur de façon asynchrone.
- -Si besoin, DOMparser intègre un document XML.
- PHP ou un autre langage de scripts peut être utilisé côté serveur.

Le terme "Asynchronous", asynchrone en français, signifie que l'exécution de JavaScript continue sans attendre la réponse du serveur qui sera traitée quand elle arrivera. Tandis qu'en mode synchrone, le navigateur serait gelé en attendant la réponse du serveur. Dynamic HTML est aussi un ensemble de techniques, qui comprend: HTML, CSS, JavaScript.

Cela permet de modifier le contenu d'une page selon les commandes de l'utilisateur, à partir de données préalablement fournies ou avec un texte tapé par l'utilisateur.

Ajax est DHTML plus l'objet XHR pour communiquer avec le serveur.