



## **Dossier de projet : création d'une application de suivi logistique d'armoires pharmaceutiques**



**Chloé GUILLET**

Titre RNCP niveau 6 (bac +3)

Conceptrice développeuse  
d'applications

Année 2024



## Sommaire

### Table des matières

|   |    |
|---|----|
| I. Liste des compétences du référentiel couvertes par le projet .....     | 4  |
| A. Présentation du référentiel .....                                      | 4  |
| B. Compétences du référentiel couvertes par le projet .....               | 5  |
| II . Résumé du projet en anglais .....                                    | 8  |
| III . Cahier des charges.....   | 9  |
| A. Explication du besoin.....   | 9  |
| B. Architecture et diagrammes.....  | 9  |
| IV . Gestion de projet.....   | 11 |
| A. Méthode Agile .....  | 11 |
| B. Versioning du code.....  | 13 |
| V . Spécifications fonctionnelles du projet.....                          | 15 |
| A. Responsivité .....   | 15 |
| B. Scan de QR codes .....   | 16 |
| C. Upload de photos et stockage dans la base de données .....             | 18 |
| D. Génération de statistiques à partir des données de l'application ..... | 19 |
| VI . Spécifications techniques du projet.....                             | 20 |
| A. Langages de programmation et outils utilisés .....                     | 20 |
| B. Architecture technique.....  | 21 |
| C. Base de données.....   | 22 |
| D. Sécurité.....  | 22 |
| E. Tests et validations.....  | 23 |
| F. Evolution du projet dans le temps .....                                | 23 |
| VII . Réalisations du candidat .....                                      | 24 |
| A. Base de données.....   | 24 |
| B. Back-end.....  | 26 |
| C. Front-end.....   | 33 |
| D. Documentation.....   | 35 |
| E. Déploiement.....   | 35 |
| F. Difficultés rencontrées .....  | 35 |
| G. Conclusion .....   | 35 |



|   |    |
|---|----|
| VIII . Présentation du jeu d'essai .....  |    |
| A. Introduction .....   |    |
| B. Données en entrée .....  | 36 |
| C. Sécurisation des données en entrée .....   | 37 |
| D. Résultats attendus .....   | 38 |
| E. Résultats obtenus .....  | 39 |
| F. Conclusion .....   | 39 |
| IX. Description de la veille effectuée pendant le projet sur les vulnérabilités de sécurité ..... | 39 |
| A. Explication de la veille : .....   | 39 |
| B. Description de la veille réalisée : .....  | 39 |
| C. Impact de la veille sur la sécurité de l'application : .....                                   | 40 |
| X. Description d'une situation de recherche .....   | 41 |
| A. Contexte .....   | 41 |
| B. Problématique .....  | 41 |
| C. Recherche .....  | 41 |
| D. Mise en application .....  | 44 |
| E. Discussion .....   | 44 |
| F. Résultats obtenus .....  | 44 |
| G. Retour d'expérience .....  | 44 |
| Annexes .....   | 46 |
| Annexe I : Diagramme de déploiement .....   | 47 |
| Annexe II : Diagramme de classe de l'application représentant uniquement les models .....         | 48 |
| Annexe III : Documentation de Code Igniter concernant les recommandations de sécurité .....       | 49 |
| Annexe IV : MCD final réalisé avec Looping .....  | 50 |
| Annexe V : Fichier de configuration upload.php .....  | 51 |



## I. Liste des compétences du référentiel couvertes par le projet

### A. Présentation du référentiel

#### **CCP1 : Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité**

- ☐ Maquetter une application (obligatoire)
- ☐ Développer une interface utilisateur de type desktop
- ☐ Développer des composants d'accès aux données (obligatoire)
- ☐ Développer la partie front-end d'une interface utilisateur web (obligatoire)
- ☐ Développer la partie back-end d'une interface utilisateur web (obligatoire)

#### **CCP2 : Concevoir et développer la persistance des données en intégrant les recommandations de sécurité**

- ☐ Concevoir une base de données
- ☐ Mettre en place une base de données (obligatoire)
- ☐ Développer des composants dans le langage d'une base de données

#### **CCP3 : Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité**

- ☐ Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
- ☐ Concevoir une application (obligatoire)
- ☐ Développer des composants métier (obligatoire)
- ☐ Construire une application organisée en couches (obligatoire)
- ☐ Développer une application mobile
- ☐ Préparer et exécuter les plans de tests d'une application
- ☐ Préparer et exécuter le déploiement d'une application

#### Compétences transverses :

- ☐ Utiliser l'anglais dans son activité professionnelle en conception et développement d'applications
- ☐ Actualiser et partager ses compétences en conception et développement d'applications

Figure 1: Extrait du référentiel de compétence pour le titre de Concepteur développeur. Les compétences rouges doivent être obligatoirement couvertes par le stage

Au cours de ce projet, j'ai travaillé sur un large panel de tâches liées au développement ce qui m'a permis de valider plusieurs compétences du référentiel lié au titre de Concepteur développeur d'applications, notamment toutes les compétences obligatoires du référentiel (voir la figure 1). J'ai effectué les tâches suivantes :

- Conception de la base de données (MCD MLD) et conception de l'application sur un modèle MVC (Model View Controller)
- Base de données
- Back-end : réalisation des différents CRUD, créations de fonctionnalités supplémentaires pour les utilisateurs, tests fonctionnels
- Front-end : réalisation de l'interface utilisateur et appel des données issues du back-end dans le front-end.



## B. Compétences du référentiel couvertes par le projet

### 1. Maquetter une application

Afin de pouvoir réaliser les maquettes de l'application, je me suis appuyée sur le cahier des charges afin de pouvoir lister l'ensemble des fonctionnalités qu'elle doit posséder.

A partir des informations obtenues, j'ai élaboré un diagramme des cas d'utilisations (X) qui m'a permis d'avoir une vue d'ensemble de ce que l'utilisateur doit pouvoir faire sur cette application et à la suite de ça, j'ai pu commencer à élaborer mes maquettes avec Canva.

### 2. Développer des composants d'accès aux données

J'ai participé à la mise en place de l'ensemble des CRUD en PHP avec Code Igniter 3, qui permettent d'interagir avec les données dans la base.

Après chaque CRUD mis en place, des tests avec des données fictives sont effectués pour vérifier que ce dernier fonctionne correctement.

### 3. Développer la partie front-end d'une interface utilisateur web

Le front-end a été réalisé en HTML/CSS avec le framework Bootstrap. L'utilisation de ce dernier a été un gain de temps considérable car l'une des fonctionnalités de Bootstrap est gérer automatiquement la responsivité, ce qui était l'une des demandes principales du client.

### 4. Développer la partie back-end d'une interface utilisateur web

Le back-end comporte l'ensemble des CRUDs réalisés, la gestion des droits des utilisateurs, les validations de formulaires et divers controllers en plus pour gérer par exemple la page d'accueil, la connexion des utilisateurs, et autres fonctionnalités importantes.

### 5. Concevoir une base de données

Après avoir établi le diagramme UML des cas d'utilisations, il était plus aisé de voir quelles tables allaient devoir être créées pour faire une base de données correspondante à la demande du client. Un MCD (Modèle conceptuel de données) a donc été élaboré en premier, suivi après d'un MLD (Modèle logique de données).

Lors de la conception de la base de données, un point très important est apparu : la base de données doit pouvoir évoluer en fonction des besoins du client sans qu'il soit nécessaire de tout changer. Il a donc fallu réfléchir à comment lier les tables entre elles pour qu'elles puissent fonctionner indépendamment le plus possible les unes des autres pour que l'application puisse continuer à fonctionner en cas de problème technique, ou que si le client souhaite ajouter une nouvelle table, cela ne perturbe pas l'application entière.

### 6. Mettre en place une base de données



La base de données est mise en place sur Php My Admin et est entièrement gérée dessus, La connexion à cette dernière est gérée par un fichier PHP contenant les informations nécessaires à l'établissement de cette dernière et le driver MySQLi. Toutes les interactions avec les données contenues dans la base sont ensuite gérées par des CRUD spécifiques à chaque table.

### 7. Développer des composants dans le langage d'une base de données

L'ensemble des requêtes faites à la base de données sont faites en SQL dans les fichiers PHP en utilisant le query builder du PHP.

### 8. Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement

Le projet est entièrement géré sur GitLab, que ça soit pour le versioning du code ou pour travailler en agilité sur des itérations d'une semaine. Le logiciel Fork a également été utilisé pour faciliter la gestion des processus de versioning.

### 9. Concevoir une application

Pour ce qui est de la conception, j'ai élaboré le MCD sous la supervision des membres de l'équipe et ai participé à la construction des diagrammes UML (cas d'utilisation, classe et déploiement).

### 10. Développer des composants métier

Au sein de l'application, j'ai pu développer plusieurs composants métiers en PHP en autonomie avec l'approbation de mon tuteur.

### 11. Construire une application organisée en couches

L'application est entièrement basée sur le design pattern MVC (Model View Controller) auquel ont été rajouté des fonctionnalités supplémentaires pour les besoins du client ou faciliter son élaboration. Ce design pattern est très courant dans le développement car il permet une structuration simple et efficace des applications ce qui facilite le débogage et la maintenabilité dans le temps de ces dernières.

### 12. Préparer et exécuter les plans de tests d'une application



Des tests fonctionnels et techniques ont été réalisés au fur et à mesure du développement des composants de l'application. Le code était également relu et validé par des pairs avant chaque merge sur la branche master (code review).

### 13. Préparer et exécuter le déploiement d'une application

L'application est déployée à travers GitLab et GitLab Runner. Afin d'expliquer le fonctionnement du déploiement, j'ai construit un diagramme (voir annexe I) qui permet de comprendre comment se passe le déploiement de l'application.



## II . Résumé du projet en anglais

The purpose of the project was to create a logistics application that would enable employees to handle the delivery of medicine cabinets and monitor their progress.

To develop the application, I collaborated with a team consisting of myself, a senior developer, and a lead developer. The client aimed to optimize the delivery process to ensure that all deliveries could be completed within a day. Consequently, it was decided to utilize QR codes to obtain information about the location, medicine cabinet, and the vehicle used for delivery.

The delivery personnel only need to specify whether they are loading or unloading the vehicle and then scan the QR codes to input delivery-related information. The application also provides a feature for the delivery personnel to report incidents through a form and upload photos of any issues.

Thanks to the application, the client will receive updates on deliveries and will be able to generate various statistics based on the data.

Throughout the project, I was responsible for developing specific features, such as XXX.

This experience allowed me to acquire proficiency in a new programming language, understand a different work methodology, observe project management and deployment processes, and further improve my ability to collaborate within a team.





### III . Cahier des charges

#### A. Explication du besoin

Le besoin du client ici est d'avoir une application permettant à ses employés de livrer des armoires pharmaceutiques sur ses différents sites, et d'avoir un suivi et une trace de ces livraisons. Il souhaite également pouvoir dresser des statistiques concernant les livraisons (ex : nombre de livraisons sur une période donnée, nombre de livraisons par jour, nombre d'incidents recensés sur telle ou telle période, ...)

Afin de gagner du temps au niveau des livraisons (un des besoins du client), un système de QR code permet à l'employé de scanner les informations liées à sa livraison afin que celles-ci soient rentrées dans l'application. L'employé doit juste avant de scanner les différents codes choisir l'option de chargement ou de déchargement. En cas d'incidents au cours des processus de livraison, les employés peuvent également les signaler avec l'application via un formulaire avec si besoin l'ajout de photos pour illustrer le problème.

L'application est capable d'évoluer dans le temps afin de s'adapter au besoin du client puisqu'il est possible de modifier ou de supprimer les informations de l'application liées aux sites, aux armoires transportées, etc .... Le client peut également au fur et à mesure de ses besoins, ajouter de nouveaux objets à transporter, de nouveaux véhicules, sites de livraisons.

#### B. Architecture et diagrammes

Pour comprendre le besoin du client dans son ensemble, j'ai commencé par lire le cahier des charges qu'il a fourni à l'entreprise. Suite à la lecture de ce document, j'ai réalisé pour commencer les diagrammes de cas d'utilisation (voir les diagrammes 1.1, 1.2 et 1.3).

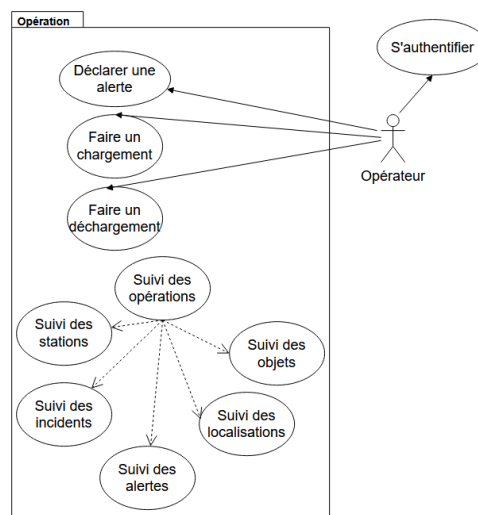


Diagramme 1.1 : Diagramme de cas d'utilisation pour les opérateurs

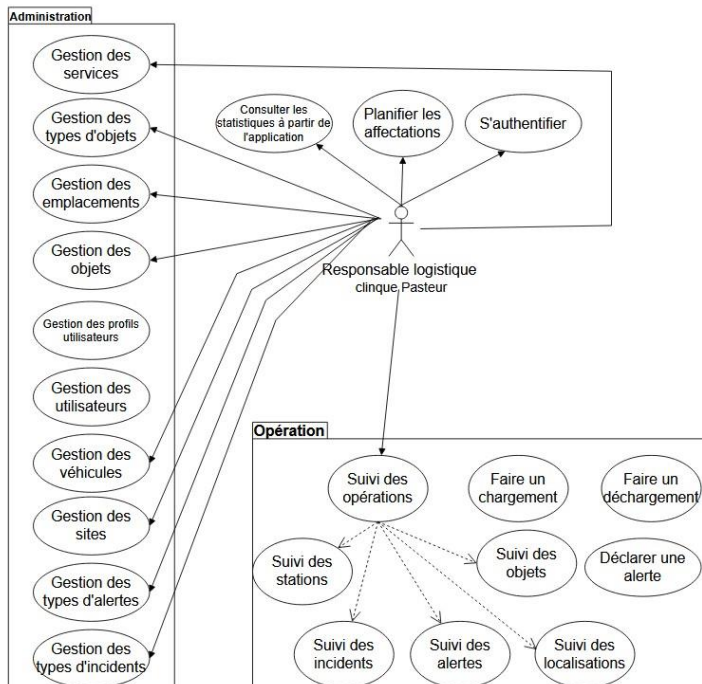


Diagramme 2.2 : Diagramme de cas d'utilisation pour le responsable logistique de la clinique Pasteur

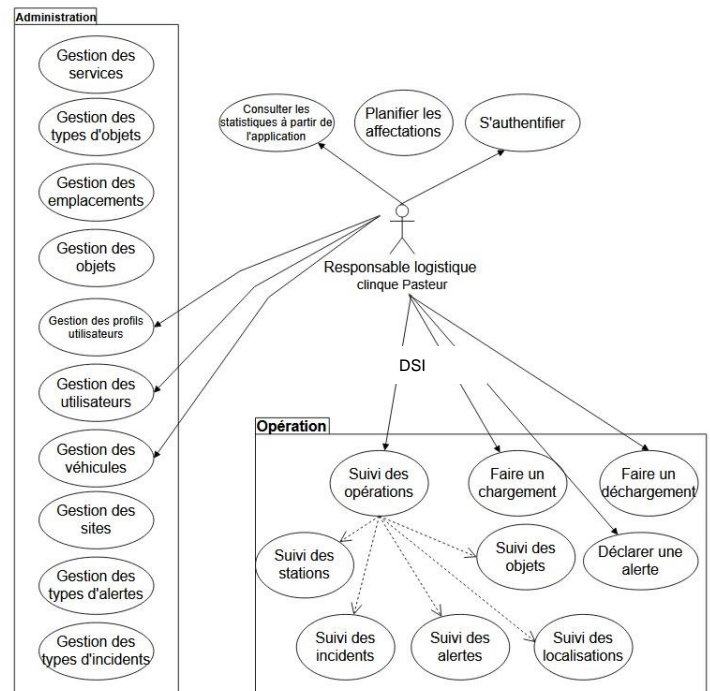


Diagramme 4.3 : Diagramme de cas d'utilisation pour le responsable logistique de DSI

Après ce diagramme j'ai ensuite dessiné le diagramme d'activité de l'application (voir le diagramme 2). Ce diagramme permet de visualiser ce que doit faire l'application et dans quel ordre les processus doivent se dérouler.

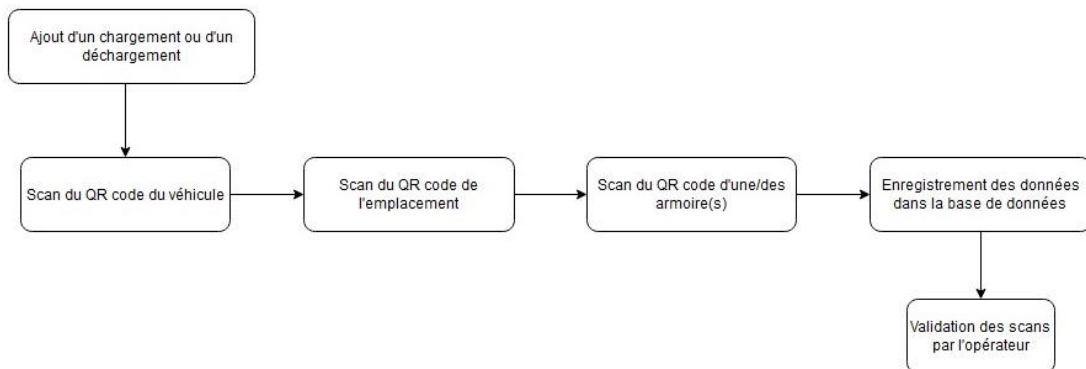


Diagramme 3 : Diagramme d'activité de l'application



## IV . Gestion de projet

### A. Méthode Agile

Pour travailler sur ce projet, l'équipe a utilisé la méthode Agile à travers les outils proposés par GitLab.

Chaque jour, l'équipe faisait un petit point rapide sur l'avancée du projet : ce qui est en cours, ce qui a été fini la journée précédente, les difficultés rencontrées et si la personne concernée nécessite de l'aide et ce qui reste à faire. Après ce point, les personnes qui avaient fini leur tâche précédente, prenaient un ticket avec une nouvelle tâche à faire pendant la journée. L'ensemble des tickets a été défini lors de la conception de

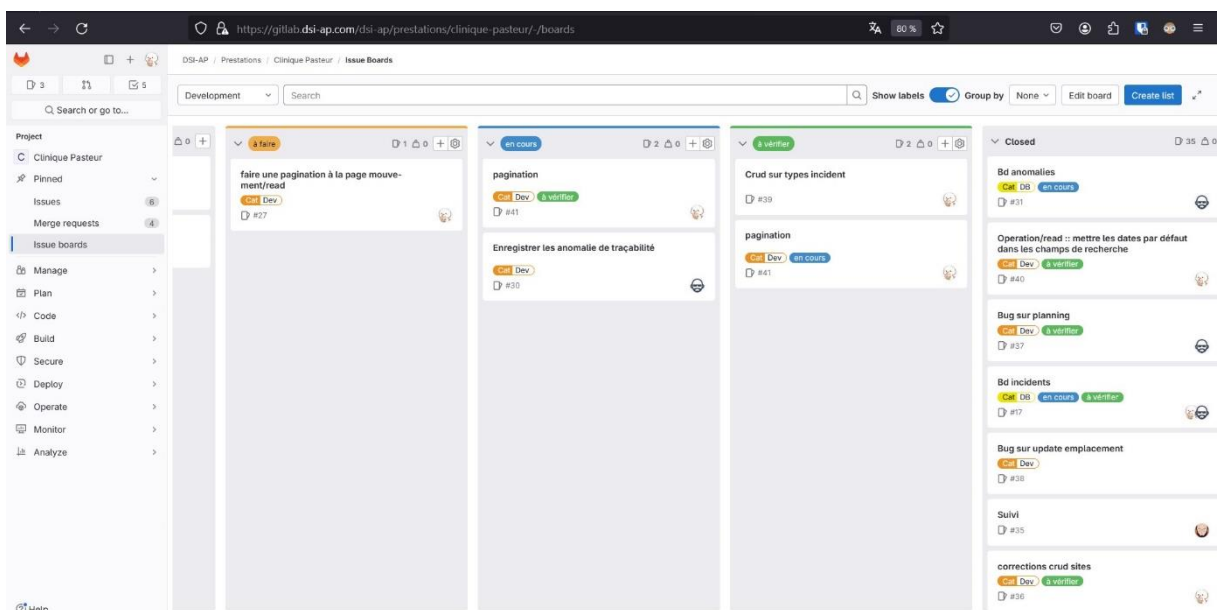


Figure 2 : Capture d'écran de GitLab en vue Kanban (Issue boards) avec des tickets et les différentes catégories d'accomplissement pour faciliter leur gestion

l'application après que nous ayons délimité le travail à faire. Le projet pouvant évoluer à tout moment selon le besoin du client ou simplement lorsqu'on se rendait que quelque chose en plus était nécessaire (ex : gestion de la connexion des utilisateurs), un nouveau ticket était créé.

GitLab permet de suivre qui fait quoi, les tickets qui doivent être fait, ceux à vérifier etc... grâce à une vue de type kanban (correspond à l'Issue board). Cela permet sur un projet une collaboration optimisée, une communication facilitée, et une autonomie plus aisée pour les membres de l'équipe.



vue / Issues / #41

## pagination

Open Issue created 2 days ago by Ramdane

Create merge request

Drag your designs here or click to upload.

Child items 0 Show labels Add

No child items are currently assigned. Use child items to break down this issue into smaller parts.

Linked items 0 Add

Link issues together to show that they're related or that one is blocking others. [Learn more.](#)

Related merge requests 1

When this merge request is accepted, this issue will be closed automatically.

Draft: Resolve "pagination" 138

### Activity

Sort or filter

- Ramdane assigned to @cgulet 2 days ago
- Ramdane added [Dev](#) label 2 days ago
- Ramdane changed Iteration to ITERATION PASTEUR Jan 1, 2024 - Jan 14, 2024 2 days ago
- Ramdane added [à faire](#) label 2 days ago
- Chloé GUILLET created branch [41-pagination](#) to address this issue 2 days ago
- Chloé GUILLET mentioned in merge request 138 2 days ago

Figure 3 : Vue du ticket pagination. Elle permet de voir ce qui a été fait et par qui sur ce ticket, de donner des informations supplémentaires et de créer la merge request.

Le développeur qui travaille sur le projet choisit un ticket à faire dans la colonne « à faire », il peut alors choisir de se l'assigner à lui seul ou avec d'autres personnes, ou de l'assigner à quelqu'un d'autre qui travaille sur le projet. Il déplace ensuite le ticket dans « en cours ». Quand le ticket est dans cette colonne, le développeur doit créer une merge request qui va faire apparaître une nouvelle branche liée à ce ticket qui sera supprimée une fois qu'elle est mergée avec la branche master. Quand le ticket est fini, il passe dans la colonne « à vérifier », là une code review est effectuée par un pair qui valide ensuite le ticket et la merge request avant de clore le ticket.

Clinique Pasteur / Merge requests / 138

## Draft: Resolve "pagination"

Open Chloé GUILLET requested to merge [41-pagination](#) into [master](#) 2 days ago

Overview 0 Commits 1 Pipelines 2 Changes 0

Closes #41

0 Assignees None - assign yourself Edit

0 Reviewers None - assign yourself Edit

Labels [Dev](#) [à faire](#) Edit

Milestone None Edit

Time tracking No estimate or time spent

1 Participant

Pipeline #16931 passed Pipeline passed for [cf3796e5](#) on [41-pagination](#) 2 days ago

Approval is optional

Merge blocked: Select Mark as ready to remove it from Draft status. [Mark as ready](#)

Merge details

- 1 commit and 1 merge commit will be added to master (squashes 1 commit).
- Source branch will be deleted.
- Closes issue [#41](#)

Activity

All activity 1

- Chloé GUILLET added [Dev](#) [à faire](#) labels 2 days ago
- Chloé GUILLET added 1 commit 2 days ago
- [cf3796e5](#) - #41 pagination OK

Compare with previous version

Preview

Figure 4 : Vue du ticket pagination. Lorsque du travail a été pushé sur la branche et que le ticket est fini, il doit être approuvé par un autre membre de l'équipe pour que le merge soit fait



Chaque ticket choisi est lié à une itération (souvent appelée sprint) qui permet de voir le travail accompli en une semaine et de juger si le projet sera fini à temps ou non. Ici, les itérations sont d'une durée allant de une à deux semaines.

## B. Versioning du code

Concernant le versioning du code, il est réalisé sur GitLab dans un espace de dépôt créé spécifiquement pour le projet. En plus de GitLab est utilisée une application qui s'appelle Fork, qui est un gestionnaire de Git et qui permet de grandement simplifier la gestion du versioning.

**C Clinique Pasteur** 🔒

🔔 134 Commits 7 Branches 0 Tags 1 Environment

Merge branch '46-remonter-les-anomalies' into 'master' **\*\*\***  
Ramdane authored 4 days ago ✓ b37c38b1

master clinique-pasteur / +

History Find file Edit Code

README CI/CD configuration Wiki

| Name               | Last commit                        | Last update |
|--------------------|------------------------------------|-------------|
| application        | Resolve "remonter les anomalies"   | 4 days ago  |
| assets             | save                               | 4 days ago  |
| public_html        | save                               | 4 days ago  |
| system             | Resolve "pagination"               | 1 week ago  |
| .gitignore         | yarn                               | 1 month ago |
| .gitlab-ci.yml     | deploy test                        | 2 weeks ago |
| README.md          | init                               | 1 month ago |
| composer.json      | init                               | 1 month ago |
| docker-compose.yml | Resolve "Création des chargements" | 1 month ago |
| migrate.sh         | migrate pas complet                | 2 weeks ago |
| package.json       | #12 debut planning                 | 1 month ago |
| webpack.config.js  | Resolve "Création des chargements" | 1 month ago |

README.md

Figure 5 : Dépôt GitLab associé au projet. Il permet de voir l'ensemble des fichiers du projet, les membres travaillant dessus et leur activité.



Fork est une application entièrement consacrée à la gestion de Git : on peut y lier un repository d'un projet et gérer les fetch, les checkout, les pull et push sur le repository.... Il est également possible de voir chaque commit associé à sa branche et ce que son auteur y a fait. Ce type d'outil permet une gestion simplifiée du versioning et de limiter les erreurs d'utilisation pouvant par exemple conduire à des conflits lors de merge.



Figure 6: Logo de Fork

Sa liaison avec l'IDE et GitLab permet de voir en temps réel comment avance la gestion du projet, qui a fait quoi, qui travaille sur quel ticket, qui est à l'origine de telle branche....

Les intérêts de travailler avec ces outils sont nombreux pour une équipe de développeurs :

- Travail sur une fonctionnalité sans impacter le reste de l'équipe en cas de problème technique
- Travail à plusieurs sur le même projet sans conflits lors des merges
- Possibilité de retourner à un état précédent un commit, un merge ou autre posant des problèmes
- Suivi facilité de l'avancée du projet avec la construction d'un « arbre » Git comportant les branches, leurs commits et leurs contenus
- Développer de manière plus sécurisée en ne mettant sur la branche principale des fonctionnalités qui marchent
- Enregistrement temporaire de modifications pouvant être importantes (= stash en git)

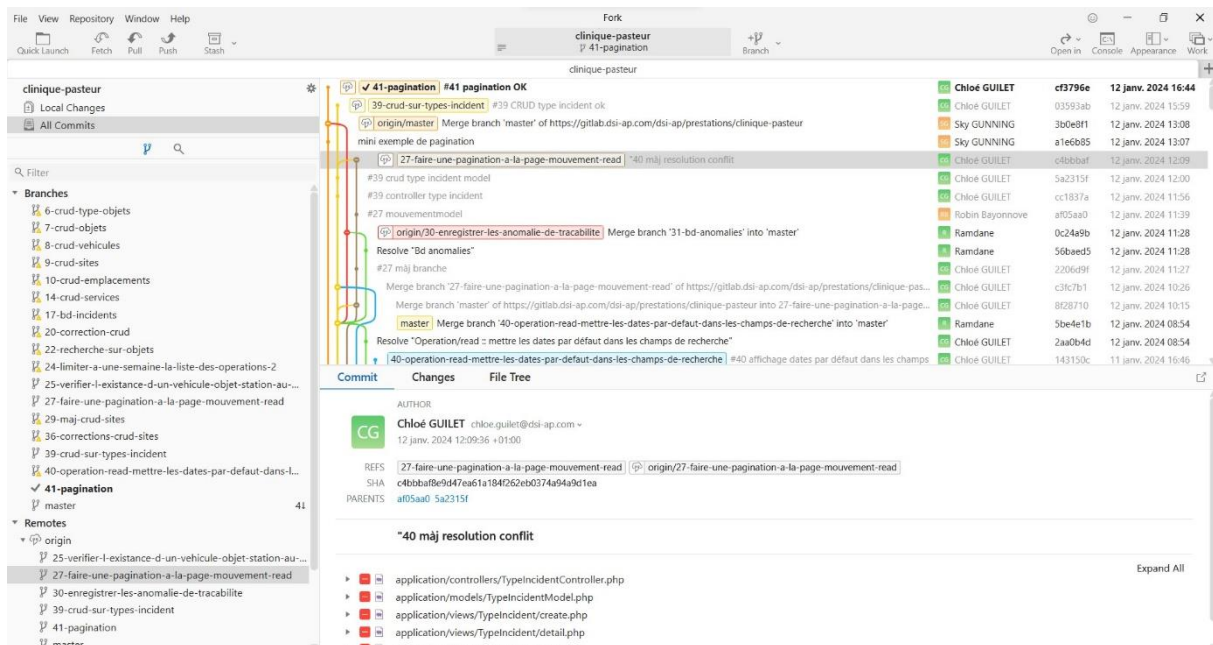


Figure 7: Capture d'écran du logiciel Fork



## V . Spécifications fonctionnelles du projet

### A. Responsivité

La responsivité de l'application n'était pas un point explicitement demandé par le client, mais au vu de l'utilisation de l'application (logistique et traçabilité des livrables/livraisons), ainsi que des utilisateurs principaux de l'application (opérateurs logistique), il était évident que l'application devait être responsive.

Le délai étant court, la solution la plus rapide et la plus simple a été d'utiliser Bootstrap qui permet de gérer automatiquement la responsivité. Dans sa configuration de base, propose six breakpoints par défaut modifiables selon les besoins.

```
$grid-breakpoints: (  
  xs: 0,  
  sm: 576px,  
  md: 768px,  
  lg: 992px,  
  xl: 1200px,  
  xxl: 1400px  
);
```

```
// No media query necessary for xs breakpoint  
@include media-breakpoint-up(sm) { ... }  
@include media-breakpoint-up(md) { ... }  
@include media-breakpoint-up(lg) { ... }  
@include media-breakpoint-up(xl) { ... }  
@include media-breakpoint-up(xxl) { ... }
```

Code 1: Exemple des paramètres de Bootstrap gérant la responsivité. Un smartphone correspond à small (sm) en moyenne et un écran d'ordinateur à extra large (xl).

Pour l'application, les résolutions définies par défaut nous convenaient et elles n'ont donc pas été modifiées.

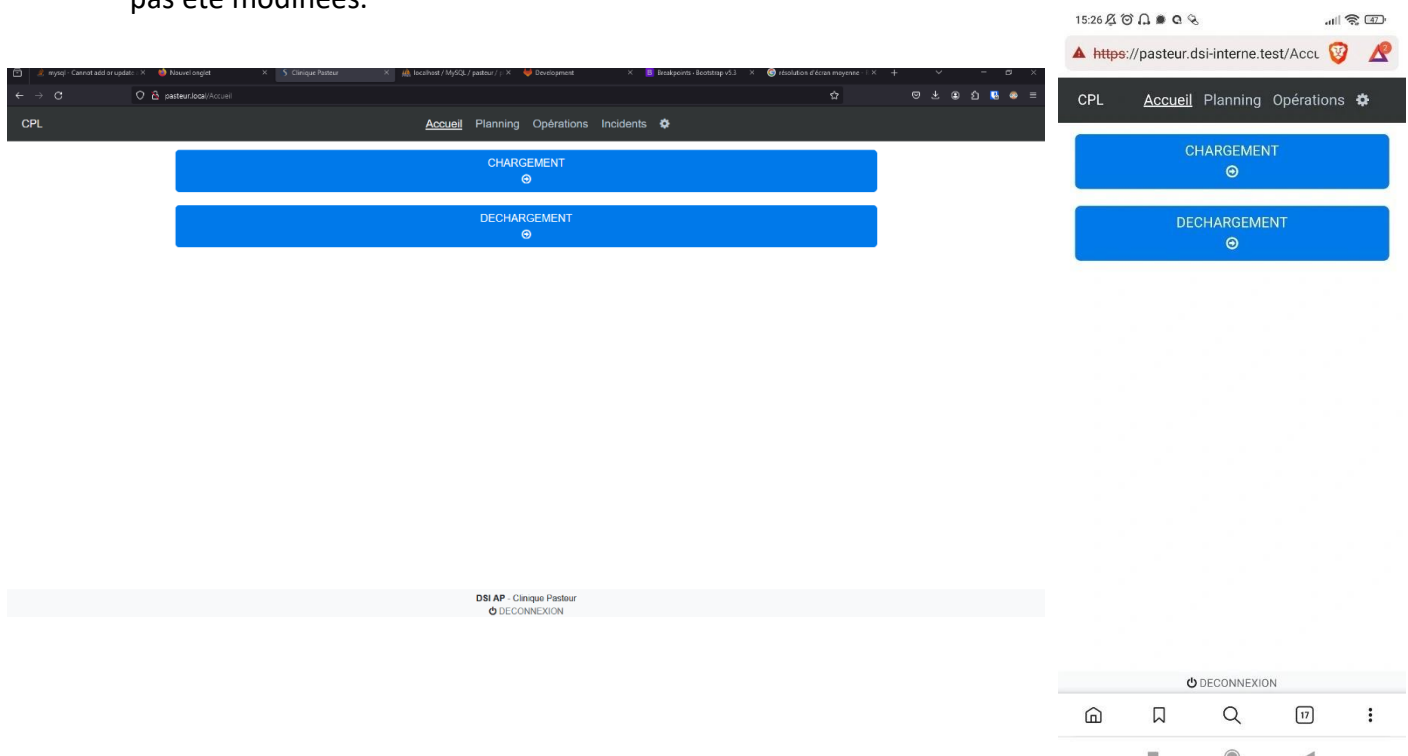


Figure 8 : Capture d'écran de la page d'accueil de l'application sur ordinateur puis sur smartphone





## B. Scan de QR codes

Le scan de QR codes est un des points clés de l'application puisque c'est dessus que s'appuie la relation entre le réel et l'enregistrement dans la base de données.

Les QR codes lorsqu'ils sont scannés par téléphone vont permettre de récupérer les données propres à l'objet scanné (ex : immatriculation, code de l'armoire, nom de la station, etc) et de les enregistrer dans la base de données grâce aux CRUDs et à travers les opérations de chargement et déchargement accessibles sur l'écran d'accueil de l'application.

Cette fonctionnalité a été codée en JavaScript par un autre membre de l'équipe. Il faut au préalable importer la classe Qr scanner qui permet de se connecter à la caméra de l'appareil utilisé dans le but de scanner.

```
function startLoading() {  
  if ($('#scan_vehicule').length) {  
    console.log('reset loading sequence');  
    loading['vehicule'] = false;  
    loading['emplacement'] = false;  
    loading['objets'] = [];  
    scanActif = '';  
  
    console.log('show div scan vehicule');  
    $('#step_vehicule').show();  
  
    console.log('start vehicule scanning');  
    vehiculeScanner();  
  }  
  else {  
    loading['emplacement'] = false;  
    loading['objets'] = [];  
    scanActif = '';  
  
    $('#step_emplacement').show();  
  
    emplacementScanner();  
  }  
}
```

*Code 2: Fonction startLoading qui permet de déterminer quels scans doivent être lancés.*

Cette fonction permet de déterminer, selon le fait qu'un véhicule soit présent ou non dans l'opération à mener, les scans à réaliser. Dans le cas où le champ véhicule est requis, la fonction lancera en premier le scanner du véhicule (vehiculeScanner()) auquel cas la fonction lancera le scan de l'emplacement en premier (emplacementScanner()).





```
function vehiculeScanner() {
    scanActif = 'vehicule';

    console.log('Start vehicule scanner');
    $('#scan_vehicule').show();

    videoElem = document.getElementById('scan_vehicule');
    //window.scrollTo(0,document.body.scrollHeight);

    scanner = new QrScanner(
        videoElem,
        result => {

            console.log(result.data);

            if (result !== null && result.data !== null && result.data.length === 8 && result.data.charAt(0) == 'V') {
                if (!vehicles[result.data]) {
                    alert('Le véhicule n\'existe pas dans la liste des véhicules');
                }
                else {
                    loading['vehicule'] = result.data;
                    console.log(vehicles[result.data].immatriculation);

                    playSound('success');
                    scanner.stop();
                    scanner.destroy();
                    scanner = null;

                    $('#scan_vehicule').hide();
                    $('#step_vehicule h3').hide();
                    $('#step_vehicule hr').hide();
                    $('#vehicule_data_input').val(result.data);
                    //$('#vehicule_data').html('<b>Véhicule</b> ' + vehicles[result.data].immatriculation + ' <a class="btn btn-danger btn-sm removeVehicule"><i class="fa fa-trash"></i></a>');
                    $('#vehicule_data').html('<b>Véhicule</b> ' + vehicles[result.data].immatriculation);

                    emplacementScanner();
                }
            }
        }, {
            highlightScanRegion: true,
        });
    scanner.start();
}
```

Code 3 : Fonction emplacementScanner()

Cette fonction vérifie que le véhicule scanné est valide, si c'est le cas elle met l'interface utilisateur à jour et les données du véhicule sinon elle affiche une alerte à l'utilisateur. Une fois ces étapes faites, elle arrête le scanner, le détruit et le remet sur une valeur nulle. Les données sont ensuite liées au véhicule pour que ça puisse être transmis au back-end et à la base de données. La numérisation du véhicule est terminée, la fonction va terminer en lançant le scan des emplacements en appelant la fonction dédiée.



## C. Upload de photos et stockage dans la base de données

L'upload est utilisé dans la gestion des incidents. Ce n'était pas un point explicitement demandé par le client mais celui-ci souhaitait avoir la meilleure traçabilité possible sur le déroulement des livraisons et des opérations gravitant autour (chargement/déchargement). L'équipe a choisi de rajouter la possibilité de téléverser des photos pour que le client puisse comprendre les problèmes rencontrés par son équipe et avoir une idée de comment il peut les résoudre, cela permet aussi aux équipes de justifier davantage les retards pouvant avoir lieu.

```
//gestion de l'upload des fichiers, le fichier est stocké dans le repo stockage/incidents
$this->load->config('upload', TRUE);
$config = $this->config->item('upload');
$this->load->library('upload', $config);
if (isset($_FILES) && isset($_FILES['photos_incident']) && isset($_FILES['photos_incident']['name']) && !empty($_FILES['photos_incident']['name'])) {
    $name_file_upload = $_FILES['photos_incident']['name'];
    $name_file_upload = strtolower($name_file_upload);
    //récupération du nom du fichier et de sa taille en kb
    $size_file_upload = round(CURLINFO_SIZE_UPLOAD / 1024, 2);
    //envoi des infos du fichier dans la table
    if ($this->upload->do_upload('photos_incident')) {
        $data = $this->upload->data();
        $fileInfo = array(
            'nom_photo' => $name_file_upload,
            'date_upload_photo' => date('Y-m-d H:i:s'),
            'taille_photo' => $size_file_upload,
            'extension_photo' => strtolower(pathinfo($name_file_upload, PATHINFO_EXTENSION)),
            'nom_repertoire_upload' => NOM_REPERTOIRE_UPLOAD_INCIDENTS
        );
        //création du fichier dans la table et return de l'id du fichier
        $id_photo = $this->PhotoModel->insert_fichier($fileInfo);
        //rename des fichiers selon le modèle suivant : photo_id du fichier
        $this->PhotoModel->update_name_photo($id_photo);
        $this->IncidentModel->insert_incident_photo($id_photo, $insert);
    }
}
if ($insert) {
    $this->redirectWithSuccess("INCIDENT ENREGISTRE");
}
```

Code 4 : Extrait de la fonction contenant la gestion du téléversement

On charge dans un premier la librairie et le fichier de configuration de l'upload fournis par PHP et Code Igniter. La fonction vérifie ensuite qu'un fichier est bien chargé et non vide dans le champ du téléversement, puis elle récupère les informations du fichier (nom, taille, extension, date du téléversement), avant de les envoyer pour qu'ils soient insérés dans la base de données. Après avoir inséré les informations dans la table, la photo est renommée et les ID sont insérés dans la table de jointure entre photo et incident. Si l'ensemble de ces opérations s'est bien déroulé, la fonction redirige sur la page read avec un message de succès.

```
<div class="mb-3">
    <label for="photos_incident" class="form-label">Photos de l'incident</label>
    <?php echo form_upload("photos_incident", set_value("photos_incident", '', false), " class='form-control' rows='3'"); ?>
</div>
```

Code 5 : Définition du champ pour le téléversement côté vue. Les valeurs contenues dans ce champ sont récupérées pour être utilisées dans le contrôleur

## D. Génération de statistiques à partir des données de l'application

Cette partie est entièrement gérée par un autre membre de l'équipe spécialisée dans le traitement et l'analyse de données. Le but de cette fonctionnalité est de traiter les données obtenues à partir de l'application pour améliorer le processus de livraison, en se basant sur les problèmes que peuvent rencontrer les employés afin de les éviter au mieux possible.

A partir de ces données, des cas d'études sont réalisés puis transmis au client afin qu'il puisse prendre les mesures nécessaires. Ces cas sont créés à partir d'ETL (Extract Transformed Load) qui sont les données venant directement de l'application.

Ces données sont ensuite stockées dans des data warehouses (entrepôts de données), qui au sein d'une entreprise contiennent généralement les bases de données correspondant à chaque service de l'entreprise (ex : comptabilité, administration, ressources humaines, etc). Les données sont centralisées pour les rendre plus accessibles en cas de besoin.

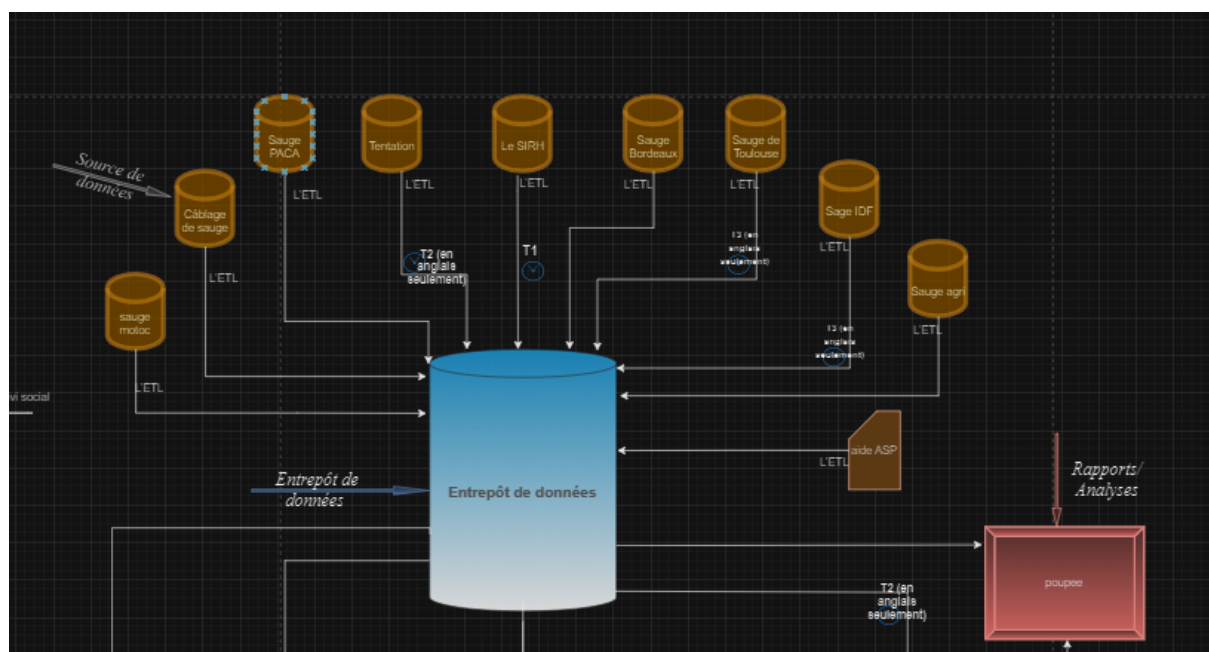


Figure 9 : Exemple de data warehouse de DSI

Dans ce cas, les données de la logistique vont être extraites afin de réaliser des KPI (Key Performance Indicator) qui permettent de savoir ce qui est améliorable ou de soulever des problèmes devant la hiérarchie afin d'améliorer les conditions de travail des employés (ex : dépassement des quotas car dépassement financier).

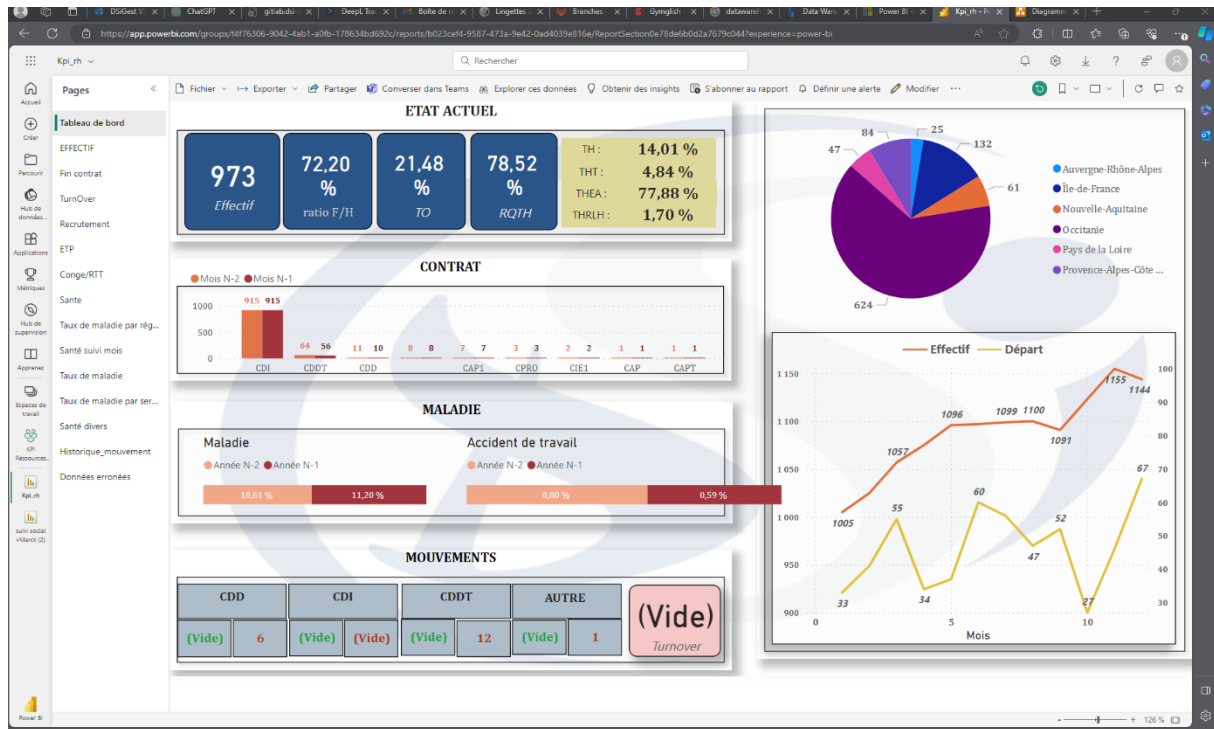


Figure 10 : Exemple de KPIs réalisés pour le service des ressources humaines de DSI

## VI . Spécifications techniques du projet

### A. Langages de programmation et outils utilisés

Le back-end est développé en PHP avec le framework Code Igniter 3. Le PHP a été choisi avant tout parce que c'est dans ce langage que l'équipe excelle, mais aussi parce que c'est un langage léger et riche en fonctionnalités, qui permet de développer des petites applications rapidement en plus d'être facile à prendre en main pour des débutants. De plus sa documentation est très large et facile à comprendre et le langage profite d'une très grande communauté active. Quant à Code Igniter, c'est un framework puissant qui consomme peu de ressource tout en proposant un large panel de fonctionnalités, avec une documentation également facile à prendre en main et une configuration minimale ce qui permet de commencer le travail plus rapidement.

Figure 8 : Logo de PHP

Figure 9 : Logo de Code Igniter



Pour ce qui est du front-end, il a été développé en JavaScript pour



gérer les interactions de l'utilisateur avec l'interface de l'application et faire le lien avec la partie back-end. Le reste est en HTML/CSS pour la mise en forme (d'autant plus qu'il est possible d'intégrer ces langages dans des fichiers PHP ce qui rend le projet plus simple à comprendre car les fichiers sont moins nombreux) couplé avec le framework Bootstrap qui permet de faciliter la mise en forme du CSS avec des templates déjà fait et personnalisables sur le site du framework. De plus Bootstrap gère automatiquement la responsivité (on peut également paramétrer les valeurs de base pour nos besoins), point qui était demandé par le client.

Figure 10 : Logo (de gauche à droite) de CSS, HTML, JS



Figure 11 : Logo de Bootstrap

Enfin, nous avons utilisé Wamp64 qui est une plateforme de développement web se basant sur les serveurs d'Apache2 permettant de travailler sur des applications en PHP et MySQL. Il intègre également PHPMyAdmin qui est un système de gestion de base de données (SGBD) que nous utilisons également dans ce projet.



Figure 12 : Logo de Wamp

## B. Architecture technique

Le projet s'appuie sur l'architecture MVC : Model View Controller (voir le diagramme 7). Ce design pattern est simple à utiliser et très fréquent dans le développement, il a en outre plusieurs avantages reconnus :

- La séparation des responsabilités entre les trois éléments du design pattern facilite la compréhension du code, sa construction et sa maintenabilité dans le temps.
- Du fait que les responsabilités soient séparées, il est plus aisé de faire évoluer l'application en modifiant ou en ajoutant de nouvelles fonctionnalités sans que cela impacte de manière négative le reste de l'application.
- Le fait d'isoler les composants permet de les réutiliser plus facilement dans d'autres endroits de l'application ou pour d'autres projets.
- Il est également plus facile de tester les composants du fait qu'ils soient isolés

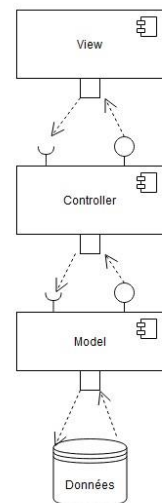


Diagramme 3 : Diagramme représentant l'architecture MVC

Si on applique le diagramme 5 à notre application pour voir le résultat on obtient le diagramme de classe suivant (le diagramme de classe complet est disponible en annexe II, pour des questions lisibilité, seul l'ensemble des modèles figure dessus) :

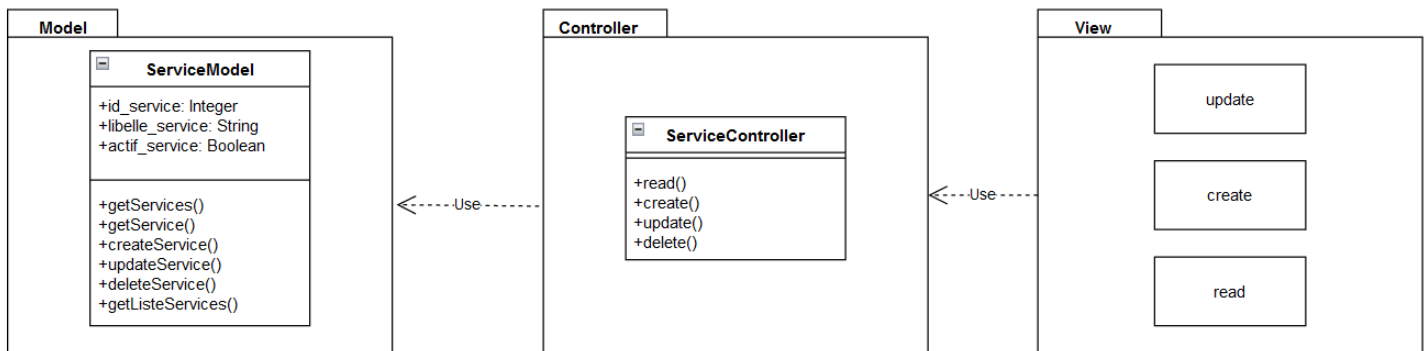


Diagramme 5 : Diagramme de classe représentatif de l'organisation de l'application selon le design pattern MVC

### C. Base de données

La base de données de ce projet a été conçue avec du SQL et le logiciel MySQL Workbench. Pour ma part j'ai commencé à travailler sur la base de données avec le logiciel Looping car c'était un outil que je connaissais, l'ayant vu en formation, mais je suis passée pour la suite du projet sur MySQL Workbench qui est le logiciel utilisé par l'équipe.



Figure 13 : Logo de MySQL



Figure 14 : Logo de Looping



Figure 16 : Logo de PHPMyAdmin



Figure 15 : Logo de MySQL Workbench

### D. Sécurité

L'un des grands avantages de Code Igniter est que le framework gère déjà une quantité de features non négligeable en termes de sécurité qui fait que les développeurs en ont peu à développer. Voici une liste de ce que gère Code Igniter :

- Gestion des URIs : les caractères autorisés au sein des URIs sont volontairement restreints pour éviter le passage de données malveillantes à l'application
- Désactivation de l'affichage des erreurs : le fait d'afficher ces erreurs à la vue de tous, peut communiquer des informations sensibles à de potentiels hacker concernant l'application. Ces erreurs peuvent être réaffichées en changeant le booléen concerné lorsque l'application est en cours de développement.
- Filtres XSS et CSRF : le premier filtre permet de détecter des scripts communs utilisés par exemple en JavaScript ou d'autres langages qui permettent de détourner des cookies ou le

- Alpha-numeric text (latin characters only)
- Tilde: ~
- Percent sign: %
- Period: .
- Colon: :

Figure 11 : Liste des caractères autorisés dans les URIs d'après la documentation officielle de Code Igniter.

- Space



code de l'application. Le second permet de filtrer les requêtes de

l'utilisateur pour en extraire les requêtes non légitimes ou non voulues par l'utilisateur, il propose également de générer des tokens propres à chaque utilisateur pour permettre d'identifier l'origine des requêtes.

- Hashage des mots de passe : PHP fournit cette fonction sur ses versions supérieures à la 5.5, mais Code Igniter en propose une également ce qui permet d'utiliser des versions plus anciennes. La fonction de hashage du framework se base sur l'algorithme BCrypt qui est plus récent. La documentation fournit par ailleurs une liste de recommandation d'usage à suivre ou ne pas suivre pour sécuriser son mot de passe (voir annexe III).
- Echappement des données : Code Igniter gère automatiquement cette fonctionnalité lorsque l'on utilise les fonctions de requêtes fournies par le framework. L'échappement consiste à rendre les caractères potentiellement dangereux dans une requête (ex : ; # -- / ') inoffensifs afin d'éviter les injections SQL.

## E. Tests et validations

Des tests fonctionnels sont réalisés au fur et à mesure que l'application est développée et des tests chez le client en conditions réelles pour étudier tous les cas de figures ont été réalisés à des étapes clés du développement. J'ai par exemple testé au cours du développement :

- L'ensemble des fonctions de mes CRUDs
- L'affichage de la pagination
- L'upload des fichiers et vérifier l'insertion des données dans la base de données

## F. Evolution du projet dans le temps

Le projet a démarré début décembre 2023 pour être livré le 29 janvier 2024. La date limite étant courte, l'équipe n'a livré que l'essentiel pour que l'application soit fonctionnelle et permette au client de travailler.

Il est certain que dans les mois qui suivent la livraison de l'application, l'équipe de DSI fournissent de nouvelles fonctionnalités à son client telles que parmi celles évoquées au moment de l'écriture de ce rapport :

- Une documentation à destination des utilisateurs
- La possibilité de faire un historique au sein de l'application de ce qui a été enregistré et de le rendre visible aux responsables logistiques
- Une responsivité adaptée aux tablettes
- La gestion des incidents (création, affichage du détail et des photos liées)
- Si le client est satisfait, il est possible à terme qu'il fasse une demande pour étendre ce système à plusieurs de ses services





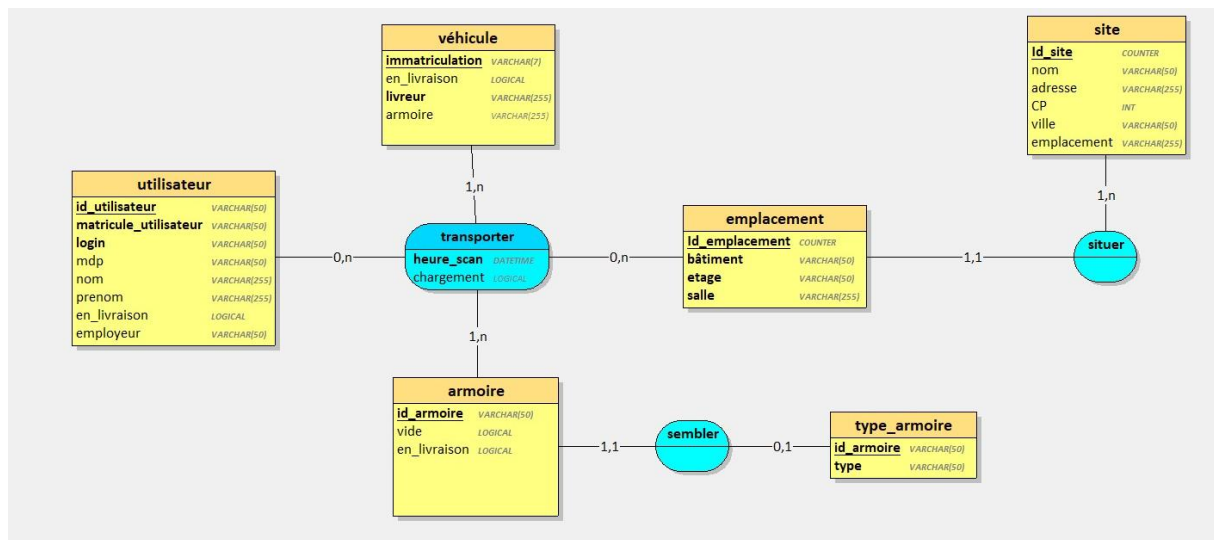
## VII . Réalisations du candidat

Au cours du projet, j'ai pu réaliser plusieurs tâches en autonomie que je vais présenter ici en plusieurs parties :

- Base de données
- Back-end
- Front-end

### A. Base de données

#### 1) Apport personnel à la conception



Dans le cadre de ce projet, j'ai d'abord travaillé sur la conception de la base de données avec Looping pour proposer une première version du MCD (Modèle Conceptuel de Données), ce dernier permet de visualiser les données et les relations entre elles. C'est une étape importante de la conception car elle permet de clarifier et simplifier la compréhension d'un projet de rendre l'organisation de ce dernier plus accessible à des personnes qui ne font pas de développement, et de servir de repère dans l'évolution du projet

Par la suite, ce MCD a évolué et de nouvelles tables sont apparues pour répondre de manière plus précise aux besoins du client, pour au final donner ce MCD. Pour des questions de lisibilité, seule une partie du MCD est visible ici, le MCD entier est disponible à l'annexe IV.



## 2) Réalisation d'une migration

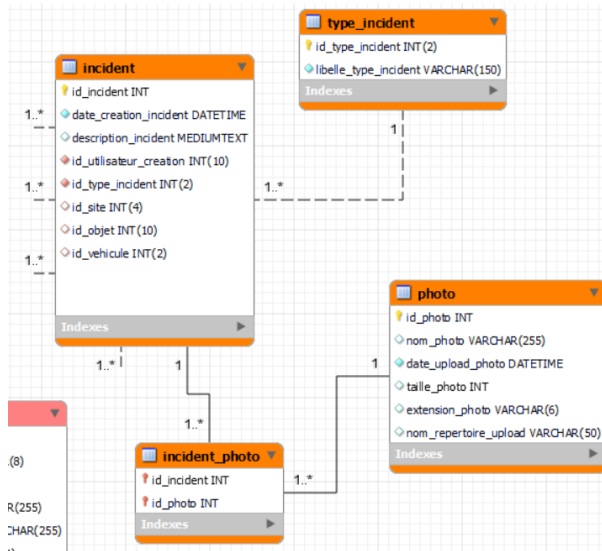


Figure 12: Tables ajoutées lors de la migration

Parmi les modifications qui ont été apportées à la base de données, il y a l'ajout des tables photos, incident, type\_incident et incident\_photo. J'ai été chargée concernant cette modification de faire la migration.

Pour cela j'ai généré les scripts SQL grâce à MySQL Workbench en ciblant les tables concernées, j'ai ensuite créé un fichier de migration sur VS Code avec la commande suivante (à faire en étant dans le répertoire public\_html) : `php index.php matches create:migration incidents`

Puis j'ai ensuite inséré l'ensemble des requêtes dans ce fichier avant de lancer la migration avec la commande suivante : `php index.php matches do:migration`

```

6 class Migration_Incidents extends CI_Migration
7 {
8     protected $db;
9     public function __construct()
10     {
11         parent::__construct();
12         $this->db = $this->load->database('default', TRUE);
13     }
14     public function up()
15     {
16         echo __FUNCTION__ . ' :: ' . __CLASS__ . ' : START' . "\n";
17
18         $this->db->query("CREATE TABLE IF NOT EXISTS `photo` (
19             `nom_photo` VARCHAR(255) NULL DEFAULT NULL,
20             `date_upload_photo` DATETIME NOT NULL,
21             `taille_photo` INT NULL DEFAULT NULL,
22             `extension_photo` VARCHAR(6) NULL DEFAULT NULL,
23             `nom_repertoire_upload` VARCHAR(50) NULL DEFAULT NULL,
24             PRIMARY KEY (`id_photo`))
25             ENGINE = InnoDB;");
26
27         $this->db->query("CREATE TABLE IF NOT EXISTS `type_incident` (
28             `id_type_incident` INT(2) NOT NULL AUTO_INCREMENT,
29             `libelle_type_incident` VARCHAR(150) NOT NULL,
30             PRIMARY KEY (`id_type_incident`))
31             ENGINE = InnoDB;");
32     }
33 }

```



Figure 13 : Exemples de requêtes SQL réalisées dans le cadre de la migration



## B. Back-end

L'objectif ici est de développer des fonctionnalités pour enrichir l'application et plus maniable pour les opérateurs.

Les fonctionnalités proposées ici ont été étudiées dans le but d'apporter quelque chose au client lors de l'utilisation de l'application (ergonomie, utilisation rapide, lisibilité, ...). Il ne s'agit pas seulement d'avoir un rendu esthétique mais d'améliorer l'application pour le client pour livrer un produit fonctionnel et de qualité.

### 1) Réalisation de CRUD

#### a) Exemples de fonction typique

Au cours de ce projet j'ai été amenée à réaliser de nombreux CRUD (Create Read Update Delete) afin de pouvoir lier l'application à la base de données et de gérer les données à l'intérieur de la base grâce à des fonctions : on peut donc les afficher (read), en créer de nouvelles (create), les mettre à jour (update), ou les supprimer (delete).

Un CRUD en se basant sur une architecture MVC comporte au moins trois fichiers : un model, qui va servir à récupérer les données de la base, un controller qui va permettre de faire le lien entre le model et la vue, et enfin la vue (détaillée dans la partie C. Front-end) qui elle permet d'afficher les données reçues à l'utilisateur.

```
//Récupération des données de la base
public function getEmplacements($activated_status = -1)
{
    //Requête à la base de données
    $this->db->select("emplacement.*, site.id_site, site.libelle_site, site.verification_dechargement_planning"
        ->from("emplacement")
        ->join('site', 'emplacement.id_site=site.id_site');

    // afficher les emplacements par leur statut activé et/ou désactivé
    if ($activated_status > -1) {
        $status = ($activated_status == 1) ? 1 : 0;
        $this->db->where('actif_emplacement', $status);
    }
    $this->db->order_by('libelle_emplacement', 'ASC');

    //Récupération du résultat dans une variable qui sera retournée
    $resultat = $this->db->get()->result();

    return secureHtml($resultat);
}
```

Figure 14 : Exemple d'une fonction de récupération de données avec filtration selon un statut actif ou non dans le model de la table emplacement



```
//Fonction d'affichage des données à l'utilisateur
public function read()
{
    //Filtration des données par la présence du statut actif
    $activated_status = $this->input->post('activated_status');
    if ($activated_status == '') {
        $activated_status = -1;
    }

    $this->data["activated_status"] = $activated_status;
    //Utilisation de la fonction du model pour obtenir les données
    $this->data["enregistrements"] = $this->EmplacementModel->getEmplacements($activated_status);
    //Envoie des données à la vue
    $this->view();
}
```

Figure 15 : Exemple de fonction dans le contrôleur de la table emplacement permettant de récupérer les données du modèle pour les envoyer à la vue

### b) Configuration du framework

Code Igniter est un framework nécessitant peu de configuration pour être utilisé, cependant, dans l'intérêt de simplifier le travail et de réduire les erreurs dans le code, il peut être intéressant de configurer quelques points supplémentaires soi-même. Code Igniter possède un répertoire config regroupant une large quantité de fichiers permettant de définir des paramètres (ex : routage, bibliothèques, droits, constantes, ...).

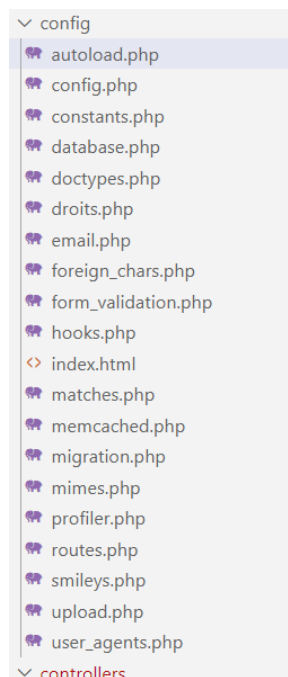


Figure 16 : Vue d'ensemble du répertoire config de Code Igniter



Dans le cas des CRUD, j'ai été amené à configurer :

- Le chargement automatique de librairies (ex : database nécessaire pour faire les requêtes)

```
$autoload['libraries'] = array('database', 'form_validation', 'pagination', 'session', 'hashing');
```

Figure 17 : Configuration du fichier autoload avec ajout des librairies à charger

- La mise en place des droits d'accès à la vue du CRUD et l'autorisation de créer, mettre à jour ou supprimer les données

```
$controleur->Site = "read;create;update;delete";  
$controleur->Objet = "read;create;update;delete;qrcodes";
```

Figure 18 : Configuration du fichier droits avec ajout des opérations possibles sur les tables Objets et Sites

- La gestion de la page dans les sous-menus pour que les utilisateurs puissent y accéder

```
$menu->administration->sousMenu[0] = "Emplacements; Emplacement/read";  
$menu->administration->onglet[0]["Emplacement"] = "Emplacements; Emplacement/read; Liste des emplacements";
```

Figure 19 : Suite de la configuration du fichier droits avec indication du nom et du chemin de la page

## 2) Ajout d'une pagination

La gestion de la pagination est un peu particulière ici puisqu'elle dépend des résultats de recherche renvoyé par une fonction du fichier MouvementModel : si les résultats ne sont pas assez nombreux, elle ne doit pas apparaître. La pagination est intégrée à la fonction read puisqu'elle doit agir directement sur l'affichage des données reçues de la base.

La pagination requiert quelques paramètres de base mais il est possible d'en rajouter d'autres si besoin :

- Offset : indique que la pagination doit commencer à partir de l'objet X, ici nous choisissons 0 pour initialiser la variable pour la suite des opérations
- Le nombre d'objets affichés par page : ici défini sous la forme d'une constante (NB\_PAR\_PAGE\_DEFAULT\_MOUVEMENT). Si cette limite est dépassée, une nouvelle page est créée
- Le chargement de la librairie de pagination qui est essentielle



```
public function read()
{
    $this->data['sites'] = $this->SiteModel->getListSites();
    $this->data['objets'] = $this->ObjetModel->getListObjets();
    $this->data['emplacements'] = $this->EmplacementModel->getListEmplacements();
    $this->data['types_objet'] = $this->TypeObjetModel->getListTypesObjet();

    $type_mouvement = $this->input->post('type_mouvement');
    $this->data["type_mouvement"] = $type_mouvement;

    $search = array (
        'date_debut' => $this->input->post('start_date'),
        'date_fin' => $this->input->post('end_date'),
        'id_objet' => $this->input->post('id_objet'),
        'id_emplacement' => $this->input->post('id_emplacement'),
        'id_site' => $this->input->post('id_site'),
        'type_mouvement' => $type_mouvement,
        'id_type_objet' => $this->input->post('id_type_objet'),
        'id_utilisateur' => $this->input->post('id_utilisateur'),
    );

    $offset = 0;
    if (!empty($this->input->get('offset'))) {
        $offset = $this->input->get('offset');
    }
    $resultats = $this->MouvementModel->search($search, self::NB_PAR_PAGE_DEFAULT_MOUVEMENT, $offset);

    $this->data['enregistrements'] = $resultats['enregistrements'];
    $nbTotalResultats = $resultats['total'];
    $this->load->library('pagination');

    $this->pagination->initialize(set_pagination_config(base_url('/Mouvement/read/'), $nbTotalResultats, self::NB_PAR_PAGE_DEFAULT_MOUVEMENT));
    $this->data['page_number'] = ($offset == 0) ? 1 : ($offset / self::NB_PAR_PAGE_DEFAULT_MOUVEMENT) + 1;
    $this->data['pagination'] = $this->pagination->create_links();

    $this->data['nbResultats'] = self::NB_PAR_PAGE_DEFAULT_MOUVEMENT;

    $this->view();
}
```

Code 6 : Fonction read() du MouvementController contenant la pagination

Ensuite j'ai initialisé et récupérer dans l'URL (en vérifiant qu'il est présent est non nul) l'offset pour le mettre à jour avec les paramètres de la requête. J'appelle la fonction search qui a été modifiée au niveau des paramètres pour la pagination et je récupère les valeurs qu'elle retourne que je passe sous forme d'un tableau. Je charge ensuite la librairie de pagination qui contient les fonctions essentielles pour réaliser cette dernière : initialize() qui permet de construire et configurer la pagination et create\_links() qui va créer les liens pour chaque page générée sur ce modèle-là :

[http://pasteur.local/Mouvement/read/?recherche=&start\\_date=2024-01-12&end\\_date=2024-02-01&id\\_utilisateur=0&id\\_objet=&id\\_emplacement=&id\\_site=&type\\_mouvement=0&id\\_type\\_objet=&valider=Rechercher&offset=30](http://pasteur.local/Mouvement/read/?recherche=&start_date=2024-01-12&end_date=2024-02-01&id_utilisateur=0&id_objet=&id_emplacement=&id_site=&type_mouvement=0&id_type_objet=&valider=Rechercher&offset=30)

Sur ce lien, la requête est construite autour des paramètres passés à la fonction search() (daté de début, date de fin, les identifiants, etc) et on retrouve tout à la fin l'offset qui est à 30 sur la deuxième page.

La pagination est ensuite appelée sur la vue simplement à travers : `<?php echo $pagination; ?>`



Opérations [Objets](#) [Stations](#) [Localisations](#) [Incidents](#) [Alertes](#)

Recherche globale

Du 12/01/2024 Au 01/02/2024 Opérateur

Objet Station Site

Type mouvement Type objet

Tous

Rechercher

< 1 2

| Date             | Objet     | Station     | Type       | Véhicule  | Opérateur           |
|------------------|-----------|-------------|------------|-----------|---------------------|
| 16/01/2024 14:49 | Armoire 2 | Ancien CLIC | Chargement | ZE-145-HS | AIT SLIMANI Ramdane |
| 16/01/2024 14:49 | Armoire 4 | Ancien CLIC | Chargement | ZE-145-HS | AIT SLIMANI Ramdane |

Figure 20 : Résultat de l'affichage de la pagination sur la vue associée à la table Mouvement

### 3) Téléversement de fichier et affichage des fichiers téléversés

Cette fonctionnalité permet de téléverser des photos pour documenter un incident qui peut arriver lors du travail d'un employé. Il y a donc trois tables en jeu pour cette fonctionnalité :

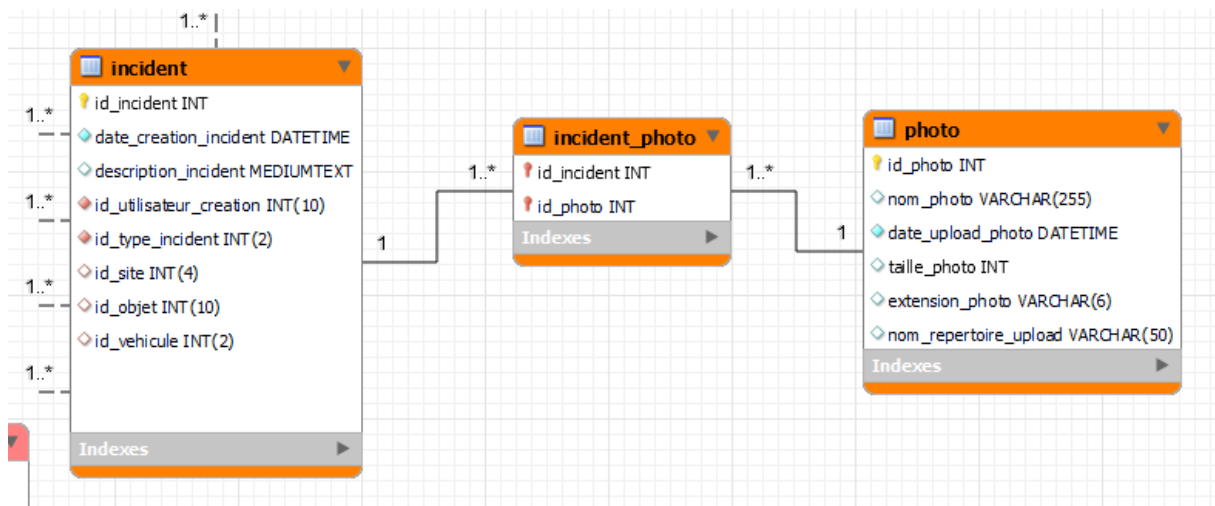


Figure 21 : Tables concernées par la fonctionnalité de téléversement



- Incident : c'est à partir de la vue de cette table que se fait le téléversement du fichier.
- Incident\_photo : permet de faire le lien entre les tables photo et incident. Elle permet d'obtenir toutes les photos pouvant être liées à un incident car il peut y'en avoir plusieurs pour un incident.
- Photo : elle contient les informations liées à chaque téléversement de fichier. Les fichiers, eux, sont stockés dans un répertoire de l'application.

```
//return de l'id de l'incident après création
$insert = $this->IncidentModel->createIncident($data);

//gestion de l'upload des fichiers, le fichier est stocké dans le repo stockage/incidents
$this->load->config('upload', TRUE);
$config = $this->config->item('upload');
$this->load->library('upload', $config);
if (isset($_FILES) && isset($_FILES['photos_incident']) && isset($_FILES['photos_incident']['name']) && !empty($_FILES['photos_incident']['name'])) {
    $name_file_upload = $_FILES['photos_incident']['name'];
    $name_file_upload = strtolower($name_file_upload);
    //récupération du nom du fichier et de sa taille en kb
    $size_file_upload = round(CURLINFO_SIZE_UPLOAD / 1024, 2);
    //envoi des infos du fichier dans la table
    if ($this->upload->do_upload('photos_incident')) {
        $data = $this->upload->data();
        $fileInfo = array(
            'nom_photo' => $name_file_upload,
            'date_upload_photo' => date('Y-m-d H:i:s'),
            'taille_photo' => $size_file_upload,
            'extension_photo' => strtolower(pathinfo($name_file_upload, PATHINFO_EXTENSION)),
            'nom_repertoire_upload' => NOM_REPERTOIRE_UPLOAD_INCIDENTS
        );
        //création du fichier dans la table et return de l'id du fichier
        $id_photo = $this->PhotoModel->insert_fichier($fileInfo);
        //rename des fichiers selon le modèle suivant : photo_id du fichier
        $this->PhotoModel->update_name_photo($id_photo);
        $this->IncidentModel->insert_incident_photo($id_photo, $insert);
    }
}
if ($insert) {
    $this->redirectWithSuccess("INCIDENT ENREGISTRE");
}
```

Code 7 : Code gérant le téléversement des fichiers dans la fonction create()

Le téléversement des fichiers nécessite le chargement de la bibliothèque upload, dans ce cas je m'appuie en plus sur un fichier de configuration contenant des constantes déjà définies pour la gestion des téléversements (voir annexe V). Avant de lancer le téléversement, des multiples vérifications sont faites pour s'assurer que quelque chose est bien récupéré. Je récupère ensuite les informations utiles au remplissage de la table photo avant faire le téléversement du contenu associé au champ photo\_incident qui permet d'envoyer les informations à la table. Je récupère ensuite l'ID du fichier versé avec celui de l'incident auquel il est lié pour remplir la table incident\_photo. Le nom du fichier téléversé est modifié pour plus de clarté et de facilité en cas de recherche par les utilisateurs.



## Ajouter un incident

[Retour](#)

\* Type d'incident

-- Choisissez un type --

\* Description de l'incident

Charger une photo

Parcourir... Aucun fichier sélectionné.

Prendre une photo

Activer la caméra

Ceci concerne un objet ?

-- Choisissez un objet --

Ceci concerne un véhicule ?

-- Choisissez un véhicule --

Ceci concerne un site ?

-- Choisissez un site --

Enregistrer

Figure 23 : Rendu de la page de création d'incidents avec le champ de téléversement

## Détail de l'incident

[Retour](#)

[Modifier](#)

Incident saisi le 02/02/2024 à 20:24

par GUILLET Chloé


|                             |  |
|-----------------------------|--|
| Type de l'incident :        | Emplacement inaccessible   |
| Description de l'incident : | Test   |
| Photo :                     |  |

Figure 22 : Rendu de la création d'un incident avec affichage des photos





## C. Front-end

L'objectif ici était de rendre l'interface utilisateur claire et lisible pour les utilisateurs. Pour cela, l'équipe a choisi de partir sur un design minimaliste et épuré, ne gardant ainsi que le strict nécessaire pour l'utilisation de l'application.

### 1) Affichage de CRUD

Lorsque le controller est fait il faut renvoyer les données vers la vue, et encore une fois Code Igniter simplifie grandement les choses puisque si les fichiers sont placés dans le bon répertoire (ex : model, controller, view) avec des noms contenant une partie commune, le framework fait automatiquement le lien entre ces fichiers. Il ne reste plus qu'à faire la mise en forme selon les besoins.

```
<table id="tab_utilisateur" class="table table-bordered mt-3">
  <thead>
    <tr>
      <th>ID</th>
      <th>Type d'alerte</th>
      <th>Statut</th>
      <th></th>
    </tr>
  </thead>
  <tbody>
    <?php
    foreach ($enregistrements as $i => $ligne) {
      ?>
      <tr>
        <td><?php echo $ligne->id_type_alerte; ?></td>
        <td><?php echo $ligne->libelle_type_alerte; ?></td>
        <td><?php echo ($ligne->actif_type_alerte == 1) ? '<span style="color:green">Activé</span>' : '<span style="color:grey">Désactivé</span>' ?></td>
        <td class="d-flex justify-content-evenly align-items-center">
          <?php if (isAuthorized('TypeAlerte/update')) : ?>
            <a href="<?php echo base_url('TypeAlerte/update/' . $ligne->id_type_alerte) ?>" id="a_update_dans_read" data-toggle="tooltip" title="Modifier" class="btn btn-sm">
              <i class="fa fa-edit"></i>
            </a>
          <?php endif; ?>
          <?php if (isAuthorized('TypeAlerte/delete')) : ?>
            <a href="<?php echo site_url('TypeAlerte/delete/' . $ligne->id_type_alerte); ?>" onclick="return confirm('Êtes-vous sûr de vouloir supprimer le type alerte ?');">
              <i class="fa fa-trash text-danger"> <?php echo (isset($ligne->controle) ? ' title="Impossible de supprimer une ligne quand un contrôle associé existe."' : ' ' ?>
            </a>
          <?php endif; ?>
        </td>
      </tr>
    <?php
    }
  </tbody>
</table>
```

Code 8 : Mise en forme d'un CRUD pour les types d'alertes

#### Type d'alerte

Ajouter +




| ID | Type d'alerte                               | Statut |   |
|----|---|--------|---|
| 1  | Chargement sans déchargement                | Activé |   |
| 2  | Déchargement sans chargement                | Activé |   |
| 3  | Déchargement dans une station non planifiée | Activé |   |
| 4  | Opération non planifiée                     | Activé |   |
| 5  | Dernière opération est un chargement        | Activé |   |

Figure 24 : Affichage de la page liée à la fonction read() pour les types d'alertes

### 2) Création de formulaires



Beaucoup de ces formulaires servent en fait à créer des éléments pour les insérer dans la base de données, ils sont liés directement aux fonctions create() des CRUDs. A chacun de ces formulaires est associé une vérification des champs avec la génération de messages d'erreurs si le champ ne correspond pas au format demandé ou est vide.

The figure shows two examples of a web form for adding a new vehicle. The form has three main fields: 'Immatriculation' (license plate), 'Modèle du véhicule' (vehicle model), and 'Statut actif' (active status). The 'Statut actif' field has a checkbox that is checked. The 'Immatriculation' field has a red border and a red error message: 'Le format de l'immatriculation n'est pas valide (Format : AB-123-CD ou 1234-AB-56)'. The 'Modèle du véhicule' field has a red border and a red error message: 'Le champ modele\_vehicule est obligatoire.' (The vehicle model field is mandatory). A 'Valider' (Validate) button is at the bottom of the form.

Figure 25 : Exemples de tentative de soumission d'un formulaire avec des champs erronés

Pour imposer une validation sur un champ il faut le demander dans le controller et ensuite dans le fichier de configuration form\_validation indiquer ce que l'on souhaite concernant le champ (obligatoire, caractère, etc) dans le cas des véhicules (voir code 9), je demande des caractères alphanumériques, à ce que les champs soient obligatoires et dans le cas de l'immatriculation je demande en plus une vérification à ce qu'elle corresponde bien au format de ma regex et à ce qu'elle soit unique dans la base de données. Si l'ensemble de ces conditions sont remplies alors que le véhicule est enregistré dans la base de données.

```
"Vehicule/create" => array(
    array(
        "field" => "immatriculation",
        "label" => "Immatriculation",
        "rules" => "trim|required|callback_check_immatriculation|is_unique[vehicule.immatriculation]"
    ),
    array(
        "field" => "modele_vehicule",
        "label" => "modele_vehicule",
        "rules" => "trim|required"
    )
),
```

Code 9 : Exemple de paramètres pour une form validation dans le cas des véhicules



### 3) Fonction de scan des QR codes

Cette fonction entièrement écrite en JS, permet à l'utilisateur de scanner des QR codes en utilisant la caméra de son téléphone quand il appuie sur le bouton lié à cette fonction. Ces informations sont ensuite récupérées pour lui permettre de faire le mouvement associé et les données sont ensuite renvoyées à la base de données.

Le code de cette fonction est expliqué dans la partie V. B. Scan de QR Codes.

### D. Documentation

La documentation à destination des utilisateurs concernant ce projet n'existe pas au moment où ce rapport est écrit, mais comme dit dans la partie VI. F., elle fera partie des améliorations à venir après la mise en production du projet.

Il existe quand même au sein du code des commentaires pour en faciliter la compréhension et au début des fichiers pour communiquer des informations sur ces derniers, mais il n'y a pas plus pour le moment.

### E. Déploiement

A chaque fois que je finissais de coder pour un ticket, je le mettais dans la colonne « à vérifier » pour qu'un des développeurs de l'équipe le vérifie et valide la merge request associée au ticket. Le code était ensuite poussé sur la branche principale et le reste de l'équipe pouvait faire un pull à partir de Fork pour avoir leur branche master à jour.



Figure 26 : Exemple de merge de la branche sur Fork suite à la validation de la merge request du ticket 42

### F. Difficultés rencontrées

La plus grande difficulté que j'ai pu rencontrer au cours de ce projet a été de le faire avec un langage et pleins de nouveaux outils que je ne connaissais pas.

Afin de les maîtriser suffisamment pour pouvoir les utiliser pleinement pour ce projet, j'ai occupé les deux premières semaines de mon stage à chercher des vidéos, des tutoriaux en ligne et à les appliquer pour comprendre ces outils. Bien qu'au bout de ces deux semaines j'avais assez de connaissances pour commencer à pratiquer, il n'en reste pas moins que j'étais loin de les maîtriser et qu'au cours du projet j'ai pu rencontrer quelques difficultés face à certaines tâches (ex : pagination).

Néanmoins le fait de devoir apprendre de nouvelles technologies de zéro et de continuellement devoir chercher à pallier mon manque de connaissance m'ont permis de renforcer mes capacités d'adaptation à un environnement nouveau et à ne pas hésiter à demander de l'aide plus souvent à mes collègues.

### G. Conclusion

J'ai beaucoup appris au cours de ce projet à travers les différentes activités auxquelles j'ai participé. Là où j'ai sans doute le plus appris est dans le back-end puisque PHP était un langage complètement



nouveau pour moi de même que le framework utilisé. J'ai aussi appris à travailler en Agile de manière claire et plus formalisée et j'ai pu renforcer ma capacité à m'adapter à un environnement totalement inconnu. Cette expérience m'a beaucoup apporté sur le plan professionnel et je suis certaine que toutes les compétences que j'ai pu acquérir ou perfectionner me seront utiles au cours de ma carrière.

## VIII . Présentation du jeu d'essai

### A. Introduction

Dans cette partie, je vais présenter le jeu d'essai utilisé pour évaluer la fonctionnalité clé de l'application qui est le scan de QR code, pour la saisie et l'enregistrement d'un mouvement dans la base de données tel qu'un opérateur en condition réelle serait amené à le faire. Ce jeu d'essai est composé de données fictives correspondant le plus possible à la réalité du client.

### B. Données en entrée

Pour réaliser le test, je dispose de QR codes (voir la figure 27) représentant chacun un type d'objet pouvant être demandé au cours d'un scan. Il suffit ensuite d'aller sur l'écran d'accueil de l'application (figure 28) pour lancer le scan, soit en choisissant chargement soit déchargement. Pour le test avec le jeu d'essai, j'ai choisi de tester le chargement car celui-ci permet en plus de scanner les véhicules, je réaliserai un test rapide du déchargement pour vérifier que celui-ci génère bien les erreurs attendues en cas de non-respect du planning défini par le responsable logistique.



Figure 27 : Exemple de QR code utilisé dans le jeu d'essai

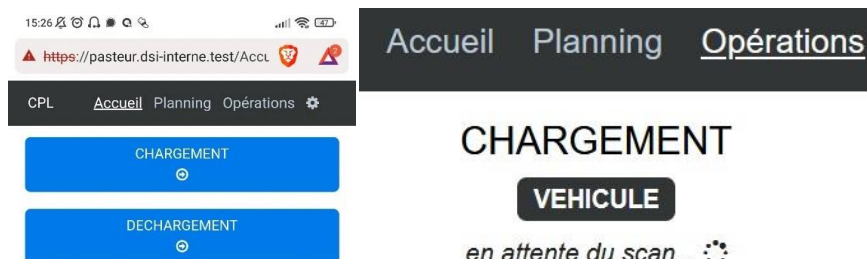


Figure 29 : Ecran de scan des véhicules



Figure 28 : Ecran d'accueil de l'application

Figure 30 : Suite et fin du chargement avec l'ajout des objets à charger



Une fois le chargement lancé, le scan s'ouvre et l'utilisateur apparaît sur cette fenêtre. Les scans s'effectuent dans un ordre précis et l'utilisateur ne peut pas y déroger car l'application vérifie à chaque fois ce que l'utilisateur lui soumet et ne lui permet pas d'accéder à la suite si le QR code n'est pas bon. Une fois le scan complet du chargement fini, l'utilisateur arrive sur l'écran suivant qui lui signifie que l'opération est bien terminée et enregistrée (figure 31). Si on souhaite vérifier que l'opération existe bien, on peut aller dans le sous menu Opérations dans l'onglet Opérations qui répertorie toutes les opérations existantes (figure 32).

## Chargement enregistré !

|          |                                     |
|----------|-------------------------------------|
| Véhicule | BB-999-ZZ                           |
| Station  | Ancien CLIC                         |
| Objets   | Armoire 2<br>Armoire 3<br>Armoire 4 |

Fermer

Figure 31 : Ecran du chargement terminé

### Suivi des opérations

**Opérations** Objets Stations Localisations Incidents Alertes

Recherche globale

Du 29 / 01 / 2024 Au 05 / 02 / 2024 Opérateur

Objet Station Site

Type mouvement Tous Type objet

Rechercher

| Date             | Objet     | Station     | Type       | Véhicule  | Opérateur    |
|------------------|-----------|-------------|------------|-----------|--------------|
| 05/02/2024 10:12 | Armoire 3 | Ancien CLIC | Chargement | BB-999-ZZ | GUILET Chloé |

Figure 32 : Vue du contenu de l'onglet Opérations. Ici l'opération de chargement a bien été enregistrée

## C. Sécurisation des données en entrée

En terme de sécurité, plusieurs choses ont été mises en place pour s'assurer que les QR codes scannés soient exacts :

- Une vérification entre les informations contenues dans le QR code et les données de la table est effectuée afin de vérifier que l'opérateur respecte bien l'ordre des objets à scanner
- 
-



- Chaque objet possède un code unique qui est lui est associé avec une nomenclature particulière, cela empêche l'utilisateur de scanner en double un objet lors d'une opération et d'en scanner un qui n'existe pas dans la base de données
- La vérification de la correspondance des données du scan avec celles de la table permet aussi d'éviter dans le cas où un utilisateur malveillant posséderait un QR code avec des données pouvant correspondre au formalisme utilisé de scanner des données inexistantes et de les enregistrer dans la base de données.
- Pour ce point, il s'agit plus d'une sécurité sur le terrain que concernant réellement l'application, mais le nombre d'objets pouvant être scanné est limité à 3 pour éviter la surcharge des véhicules et compromettre la sécurité des employés.
- Pour une traçabilité optimale, en cas de non-respect du planning et des affectations lors des déchargements des armoires à la clinique, un message est affiché à l'employé (voir figure 33) et une alerte est générée dans l'application (voir figure 34)



Figure 33 : Exemple de message d'alerte envoyé à l'employé en cas d'erreur

## Alertes

Opérations Objets Stations Localisations Incidents Alertes

| Date de création | Type alerte                                 | Object concerné | Type de mouvement | Station   | Opérateur en service |
|------------------|---|-----------------|-------------------|-----------|----------------------|
| 05/02/2024 10:55 | Déchargement dans une station non planifiée | Armoire 3       | Déchargement      | fond USIC | GUILET Chloé         |

Figure 34 : Vue de l'onglet Alertes. L'alerte suite au non-respect du planning a bien été générée sur la page

## D. Résultats attendus

Si le test se déroule bien, je m'attends pour commencer, à ce que le scan ne prenne pas en compte les QR codes présentés lorsqu'ils ne correspondent pas au type d'objet demandé par l'application, et qu'il ne réagisse pas non plus fasse à un QR code présentant des données fictives.

Ensuite je m'attends à ce que le scan puisse être complet avec à la fin l'obtention l'écran de confirmation que le scan est terminé quand on lui soumet les bons QR codes et dans l'ordre demandé.

Pour finir afin d'être sûre que tout s'est bien déroulé, je m'attends à ce que l'opération soit visible dans la base de données dans la table mouvement, et qu'elle puisse être consultée dans le sous menu Opération et l'onglet Opérations (voir figure 32).



## E. Résultats obtenus

D'après les figures 29 à 34, on peut voir que le test avec le jeu d'essai s'est bien passé : on a bien la confirmation de l'enregistrement du chargement et celui-ci apparaît bien dans le suivi des opérations.

Face à des QR codes contenant des données inconnues de la base, l'application ne réagit pas et il en va de même face à des QR codes connus mais ne correspondant pas aux attentes de l'application au moment donné.

Enfin pour finir, en cas de déchargement ne respectant pas le planning, il y a bien une erreur qui s'affiche et en cas de confirmation du déchargement malgré l'avertissement à l'employé, on a bien une alerte qui s'affiche dans le sous menu Opérations, onglet Alertes (voir figure 34)

## F. Conclusion

De nombreuses sécurités sont mises en place pour garantir l'intégrité de l'application, mais aussi pour protéger ses utilisateurs. Cela permet d'éviter l'écriture de données erronées dans la base, de garantir la sécurité des employés en évitant la surcharge des véhicules et de permettre une traçabilité optimale des livraisons et des livrables.

Cela permet de mettre en avant l'importance d'une bonne sécurité sur l'application pour garantir son intégrité mais aussi celle de ceux qui l'utilisent au quotidien. Si ce système est amené à être appliqué à d'autres secteurs d'activité du client, on peut supposer que la sécurité actuelle va s'adapter et évoluer pour coller au mieux à cette nouvelle activité.

## IX. Description de la veille effectuée pendant le projet sur les vulnérabilités de sécurité

### A. Explication de la veille :

La veille sur les vulnérabilités de sécurité a été une composante essentielle du projet, visant à maintenir l'application développée à jour et protégée contre les menaces potentielles. Cette veille a été effectuée de manière régulière tout au long du projet afin de s'assurer que les dernières vulnérabilités étaient identifiées et traitées de manière appropriée.

### B. Description de la veille réalisée :

Afin de maintenir une veille active sur les vulnérabilités de sécurité, nous nous sommes concentrés sur trois sources d'informations fiables et pertinentes.

Tout d'abord, nous consultons régulièrement le site de l'OWASP (Open Web Application Security Project) qui fournit des ressources et des guides de bonnes pratiques en matière de sécurité. Nous nous référons notamment à leur fameux "OWASP Top 10" qui recense les dix principales vulnérabilités à prendre en compte lors du développement d'applications web.





De plus au sein de notre entreprise, nous avons



régulièrement des formations sur la cybersécurité et sur les bonnes

pratiques en entreprise concernant la sécurité informatique. Ces formations sont importantes pour être à jour sur les bonnes pratiques face aux nouvelles menaces qui peuvent apparaître et ne pas oublier les bonnes pratiques de base.

Enfin, je suis également divers comptes (l'ANSSI, The Cyber Security Hub et IT-Connect) sur LinkedIn qui sont des comptes spécialisés dans la détection de failles et vulnérabilités et leurs gestions. Ils publient régulièrement des posts faisant état des dernières failles de sécurité découvertes ou de l'actualité cyber. Cela me permet de rester informée les dernières actualités, avancées et découvertes.

En m'appuyant sur ces 3 sources d'informations différentes, j'ai pu rester informée sur l'actualité de la sécurité informatique et la cybersécurité afin de toujours avoir les meilleures pratiques possibles sur ces domaines pour garantir la sécurité de ce que je développe.

### C. Impact de la veille sur la sécurité de l'application :

Grâce à cette veille constante sur les vulnérabilités de sécurité, nous avons pu intégrer des mesures de protection supplémentaires dans l'application. Cela inclut la mise en place de mécanismes d'authentification renforcés, la mise à jour régulière des bibliothèques et des frameworks utilisés, ainsi que la réalisation de tests de sécurité périodiques pour détecter et corriger les éventuelles vulnérabilités.

En suivant les dernières tendances et les meilleures pratiques en matière de sécurité, nous avons pu minimiser les risques d'exploitation des vulnérabilités connues. Cette veille constante a joué un rôle essentiel dans la protection des données et la prévention des attaques potentielles.





## X. Description d'une situation de recherche

### A. Contexte

Dans le métier de développeur, nous sommes souvent amenés à faire des recherches pour nous aider à résoudre des problèmes courants inhérents à ce métier (ex : message d'erreur que l'on ne connaît pas, besoin technique, etc). De plus les langages sont tellement nombreux et différents aujourd'hui, qu'il est impossible de les maîtriser parfaitement, il est donc important et courant que les développeurs fassent des recherches pour les aider dans leurs missions.

### B. Problématique

Le projet a été entièrement fait en PHP pour le back end qui est un langage que je ne connaissais pas avant. J'ai eu pour mission au cours de ce projet de réaliser une pagination et afin de m'aider dans cette tâche j'ai réalisé des recherches sur Internet.

### C. Recherche

La plupart du contenu publié sur Internet dans le domaine du développement est en anglais, il est donc plus judicieux d'effectuer ses recherches en anglais afin d'obtenir davantage de contenu pertinent. Sur la figure 27, les premiers liens donnés par le navigateur concernent directement la documentation officielle du framework, suivie de blogs de développeurs. Mais avec la richesse d'internet aujourd'hui, il est aisé d'obtenir différents types de supports susceptibles de convenir davantage à une population variée (ex : vidéo, blogs, documentation officielle, ...).



how to make pagination codeigniter

Vidéos Images Actualités Livres Finance

Environ 356 000 résultats (0,18 secondes)





-  **CodeIgniter**  
<https://codeigniter.com/libraries> · Traduire cette page
- Pagination Class — CodeIgniter 3.1.13 documentation**  
Simply **create** a new file called **pagination.php**, add the **\$config** array in that file. Then save the file in **application/config/pagination.php** and it will ...
-  **CodeIgniter**  
<https://www.codeigniter.com/pagi...> · Traduire cette page
- Pagination — CodeIgniter 4.4.4 documentation**  
28 déc. 2023 — **Adding WHERE ...** If you want to add **WHERE** conditions, you can specify conditions directly: // In your Controller. **\$model = new \App\Models\ ...**
-  **Guru99**  
<https://www.guru99.com/codeign...> · Traduire cette page
- Pagination in Codeigniter with Step by Step Example**  
11 nov. 2023 — We will start with creating the database and inserting some dummy records in it. It will be a single table database with 50 records on it. Run ...
-  **Cloudways**  
<https://www.cloudways.com/blog> · Traduire cette page
- Use Pagination in CodeIgniter in your projects**  
8 déc. 2021 — **Pagination in CodeIgniter** is a great technique for displaying a large number of

Figure 28 : Résultats de la recherche "How to make pagination codeigniter"



Avec la révolution de l'IA, il peut être intéressant aussi de parfois s'appuyer sur ChatGPT non pas en tant que réponse seule mais en tant que

complément de réponse ou en tant qu'aide après un échec dans une mise en application des précédents résultats de recherches. Dans le cas de ma pagination, j'ai trouvé dans la documentation des éléments que je ne comprenais pas bien comme la notion de l'uri segment et comment il était utilisé ici j'ai donc fait appel à ChatGPT pour m'aider à comprendre ce point davantage.



**You**

C'est quoi l'uri segment



**ChatGPT**

L'URI (Uniform Resource Identifier) segment est la partie d'une URL qui correspond à une section spécifique de la hiérarchie de la ressource. Dans le contexte de Codelgniter, l'URI segment est utilisé pour déterminer quelle partie de l'URL contient le numéro de la page lors de la pagination.

Par exemple, considérez l'URL suivante :

```
ruby Copy code  
  
http://exemple.com/controller/methode/3
```

Dans cette URL, "controller" est le contrôleur, "methode" est la méthode du contrôleur, et "3" est le troisième segment de l'URI. Dans le cas de la pagination, le segment de l'URI est utilisé pour indiquer le numéro de la page que l'utilisateur souhaite afficher.

Si, dans votre configuration de pagination, vous avez `'uri_segment' => 3`, cela signifie que le numéro de page sera extrait du troisième segment de l'URI. Vous devrez donc utiliser `$this->uri->segment(3)` pour récupérer ce numéro de page dans votre contrôleur.

Notez que le comptage des segments commence à partir de 1. Ainsi, `$this->uri->segment(1)` renverra le premier segment ↓ l'URI, `$this->uri->segment(2)` le

Figure 29 : Utilisation de ChatGPT pour comprendre une notion importante en pagination



## D. Mise en application



J'ai appliqué les instructions données par la documentation officielle en prenant en compte le code déjà existant.

Il a donc fallu commencer par paramétrer la pagination, je me suis servie de ChatGPT pour mieux comprendre à quoi pouvait servir les différents paramètres demandés pour faire le tri entre ceux dont j'avais besoin et ceux qui étaient superflus.

Une fois les paramètres configurés, j'ai appliqué le reste des instructions données par la documentation en essayant de les faire correspondre au reste du code.

## E. Discussion

En tant que développeuse, j'aime ne pas me baser sur une seule source d'information lorsque que je fais des recherches, que ça soit pour avoir des informations sur un outil, une erreur ou autre ou que ça soit pour avoir un aperçu de la méthode à appliquer.

Dans ce cas-ci, je me suis appuyée en premier lieu sur la documentation, mais cela ne m'a pas empêché d'aller consulter des blogs de développeurs pour voir si ces derniers ont rencontré les problèmes que j'ai pu rencontrer, si ils avaient une autre manière de faire plus pertinente vis-à-vis du fait que je devais faire une pagination se basant sur les résultats d'une recherche par l'utilisateur.

ChatGPT a également été utilisé en renfort des informations que j'ai pu trouver, afin de m'aider à déboguer le code, mais aussi pour mieux comprendre les informations que je pouvais trouver sur Internet, notamment sur la documentation car même si celle de Code Igniter est plutôt bien faite, il y a parfois quelque passage difficile à comprendre que cela soit au niveau des concepts ou à cause de la barrière de la langue.

## F. Résultats obtenus

A l'aide des différentes ressources trouvables sur Internet (ChatGPT et la documentation officielle) et avec un peu d'aide de mes collègues, j'ai réussi à mettre en place la pagination demandée.

## G. Retour d'expérience

En tant que développeur, on est très souvent amené à faire des recherches et je pense qu'il est important de ne pas hésiter à les faire en anglais pour commencer même si on ne maîtrise pas bien la langue car il existe aujourd'hui suffisamment d'outils pour passer outre cette barrière.



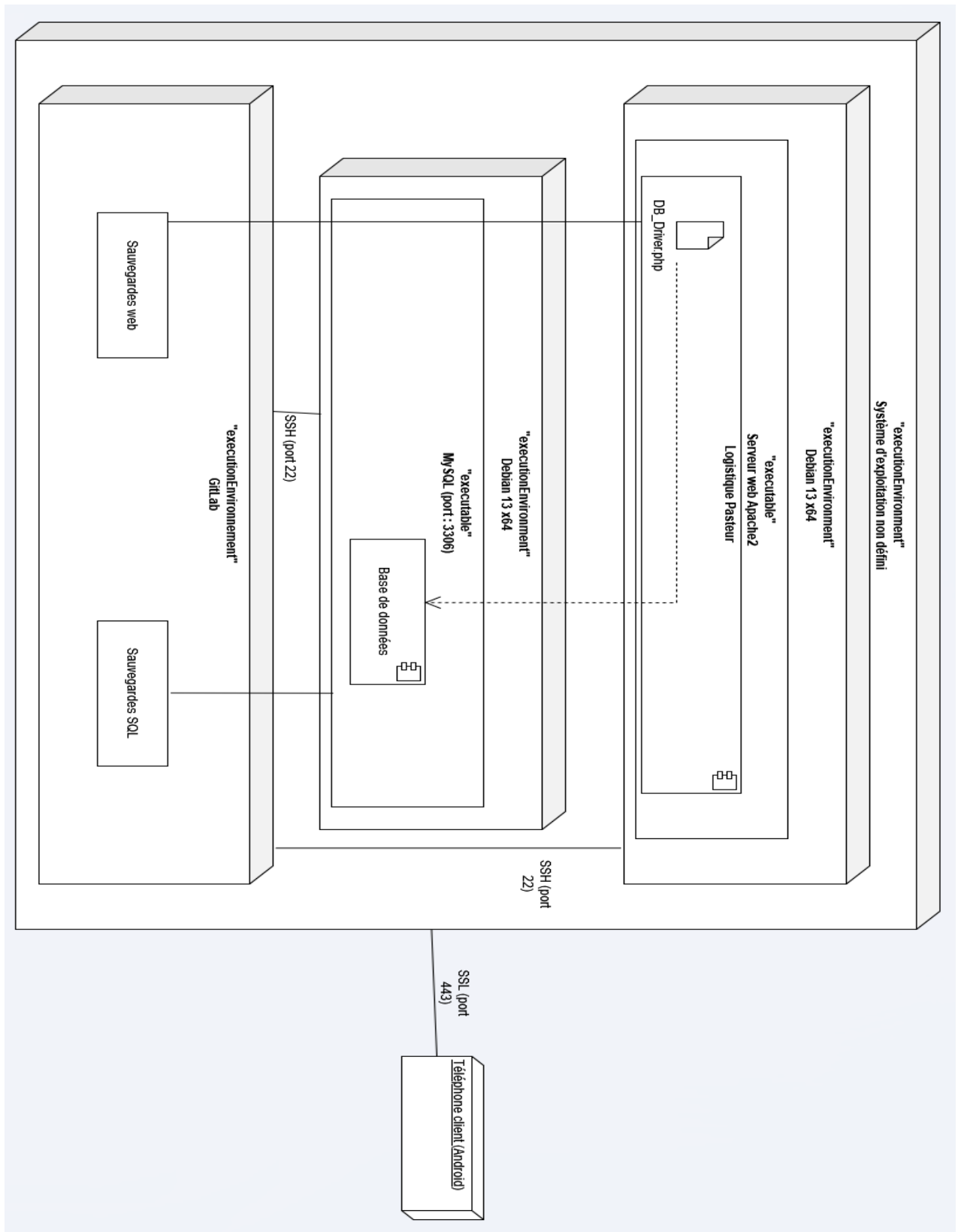
Il est important également de ne pas s'arrêter à la première adresse que l'on trouve sur Internet mais d'aller voir plusieurs ressources (blog, vidéos, documentation) afin d'avoir différents points de vue car cela peut aider à résoudre les problèmes rencontrés, en plus d'acquérir des connaissances supplémentaires.

Concernant ChatGPT, il est indéniable qu'il fait un outil de choix et de qualité aujourd'hui pour un développeur. Cependant, je pense qu'il est important de ne pas constamment l'utiliser en tant que premier outil car un développeur se doit de faire évoluer sa capacité à raisonner et à comprendre le langage qu'il utilise. Or je pense qu'utiliser ChatGPT en permanence ne permet pas à ses capacités de se développer pleinement, mais qu'il doit plutôt être utilisé quand ces capacités nous font défaut ou que ce que l'on trouve sur Internet n'est pas suffisant pour nous aider dans notre tâche. Pour moi ChatGPT est un outil complémentaire avant tout, qui doit être utilisé avec parcimonie.



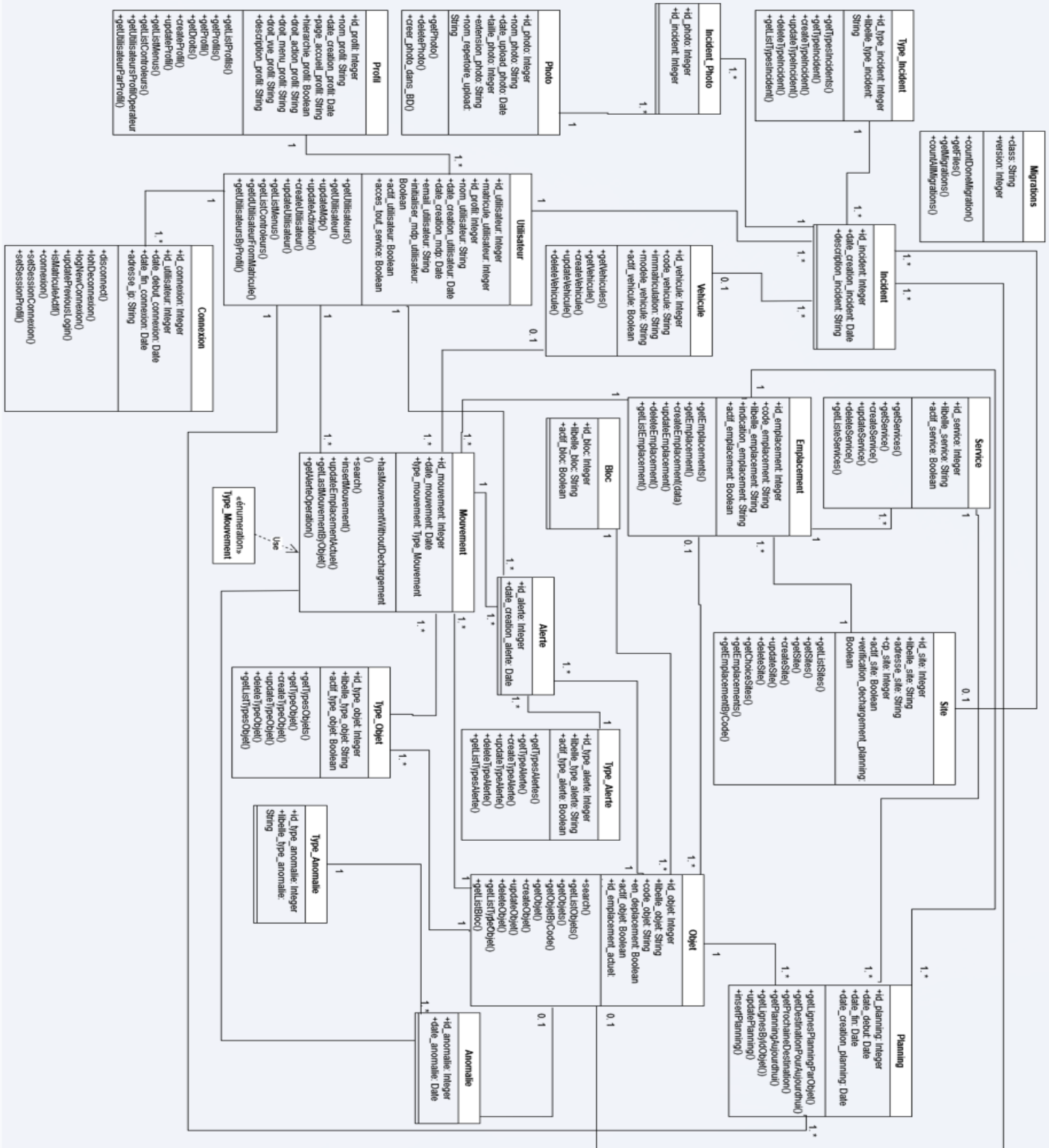
## Annexes

## Annexe I : Diagramme de déploiement





## Annexe II : Diagramme de classe de l'application représentant uniquement les models







## Annexe III : Documentation de Code Igniter concernant les recommandations de sécurité

### Password handling

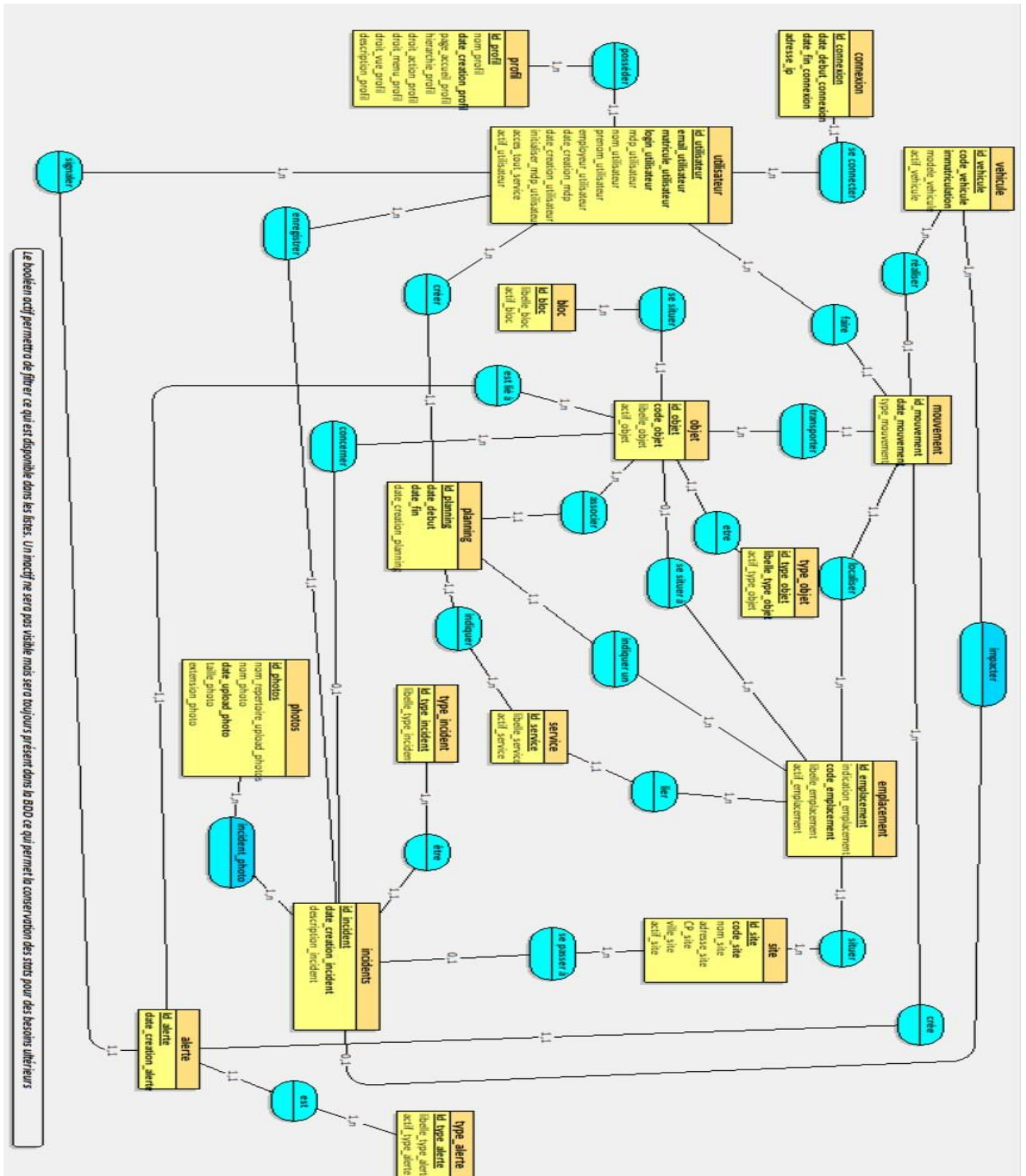
It is *critical* that you handle passwords in your application properly.

Unfortunately, many developers don't know how to do that, and the web is full of outdated or otherwise wrongful advices, which doesn't help.

We would like to give you a list of combined do's and don'ts to help you with that. Please read below.

- DO NOT store passwords in plain-text format.  
Always **hash** your passwords.
- DO NOT use Base64 or similar encoding for storing passwords.  
This is as good as storing them in plain-text. Really. Do **hashing**, not *encoding*.  
Encoding, and encryption too, are two-way processes. Passwords are secrets that must only be known to their owner, and thus must work only in one direction.  
Hashing does that - there's *no* un-hashing or de-hashing, but there is decoding and decryption.
- DO NOT use weak or broken hashing algorithms like MD5 or SHA1.  
These algorithms are old, proven to be flawed, and not designed for password hashing in the first place.  
Also, DON'T invent your own algorithms.  
Only use strong password hashing algorithms like BCrypt, which is used in PHP's own [Password Hashing](#) functions.  
Please use them, even if you're not running PHP 5.5+, CodeIgniter provides them for you.
- DO NOT ever display or send a password in plain-text format!  
Even to the password's owner, if you need a "Forgotten password" feature, just randomly generate a new, one-time (this is also important) password and send that instead.
- DO NOT put unnecessary limits on your users' passwords.  
If you're using a hashing algorithm other than BCrypt (which has a limit of 72 characters), you should set a relatively high limit on password lengths in order to mitigate DoS attacks - say, 1024 characters.

## Annexe IV : MCD final réalisé avec Looping





## Annexe V : Fichier de configuration upload.php

```
1  <?php
2  defined('BASEPATH') OR exit('No direct script access allowed');
3
4  // répertoire de destination des fichiers uploadés
5  $config['upload_path'] = REPERTOIRE_UPLOAD;
6
7  // Extensions autorisées
8  $config['allowed_types'] = "csv|jpg|png|jpeg";
9
10 // Taille maximale autorisée
11 $config['max_size'] = UPLOAD_MAX_FILE_SIZE;
12
13 // Booléen précisant si un fichier peut être écrasé
14 $config['overwrite'] = TRUE;
15
16 // Nombre de caractères maximal pour chaque nom de fichiers à uploader
17 //$config['max_filename'] = 30;
18
```