

# Dossier d'analyse des besoins

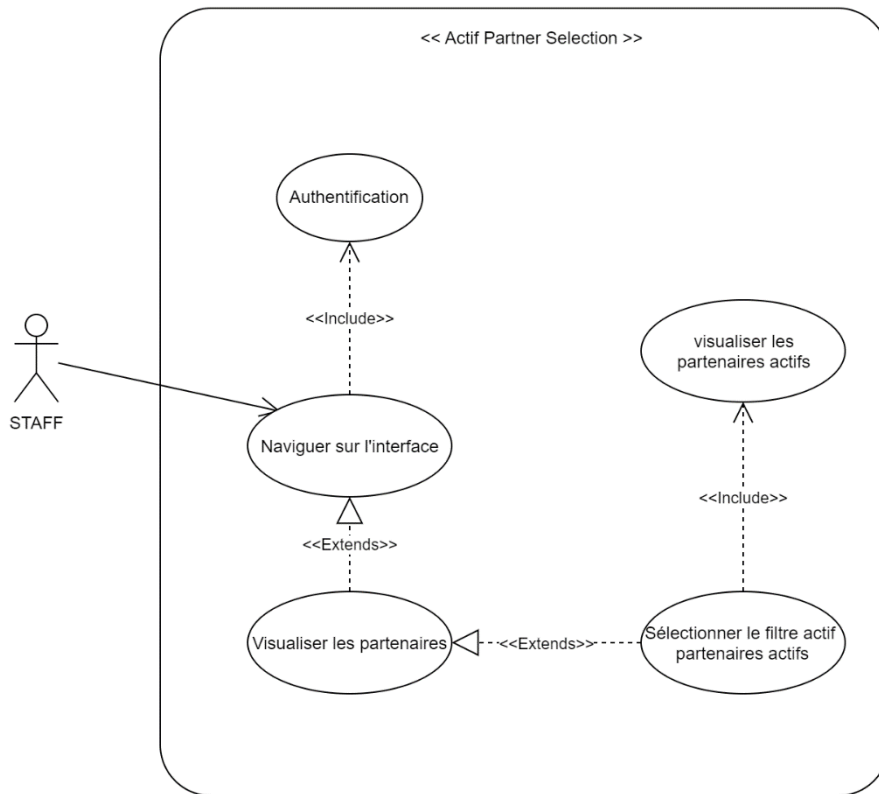
## Table des matières

I.	Diagramme de langage de modélisation unifié (UML) .....	3
1.	Diagramme de cas d'utilisation.....	3
2.	Diagramme de séquence .....	4
3.	Diagramme Merise .....	5
II.	Stack technique et Architecture logicielle.....	5
1.	Stack technique <b>Front-End</b> .....	5
2.	Stack technique <b>Back-End</b> .....	6
3.	Architecture logicielle .....	7
III.	Sécurité .....	8
1)	Formulaires .....	8
2)	Mot de passe.....	8
3)	Modification des structures/franchises .....	8
4)	Accès à la plateforme .....	9
5)	Base de données.....	9

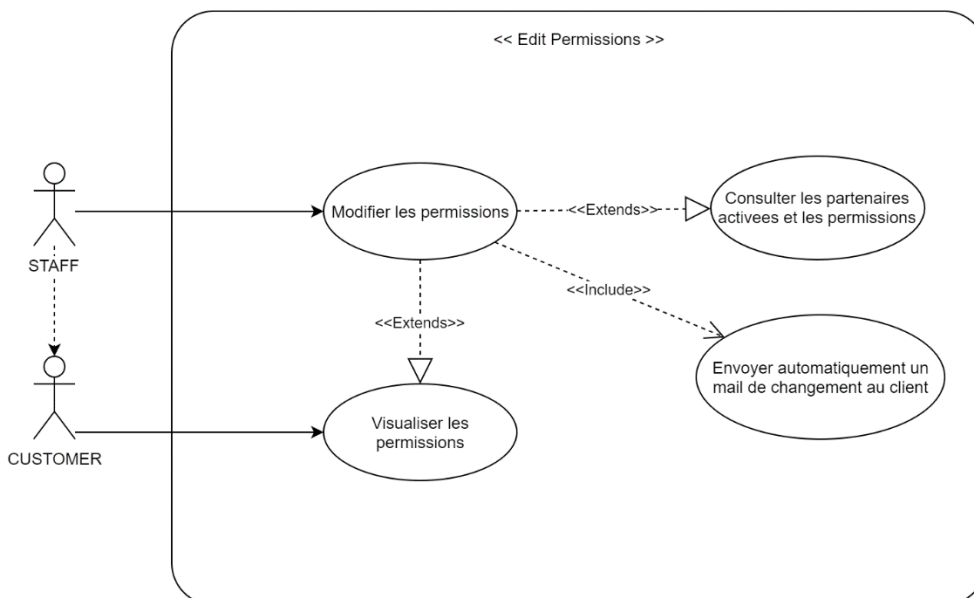
# I.Diagramme de langage de modélisation unifié (UML)

## 1. Diagramme de cas d'utilisation

### Diagramme de cas d'utilisation de sélection des partenaires actifs

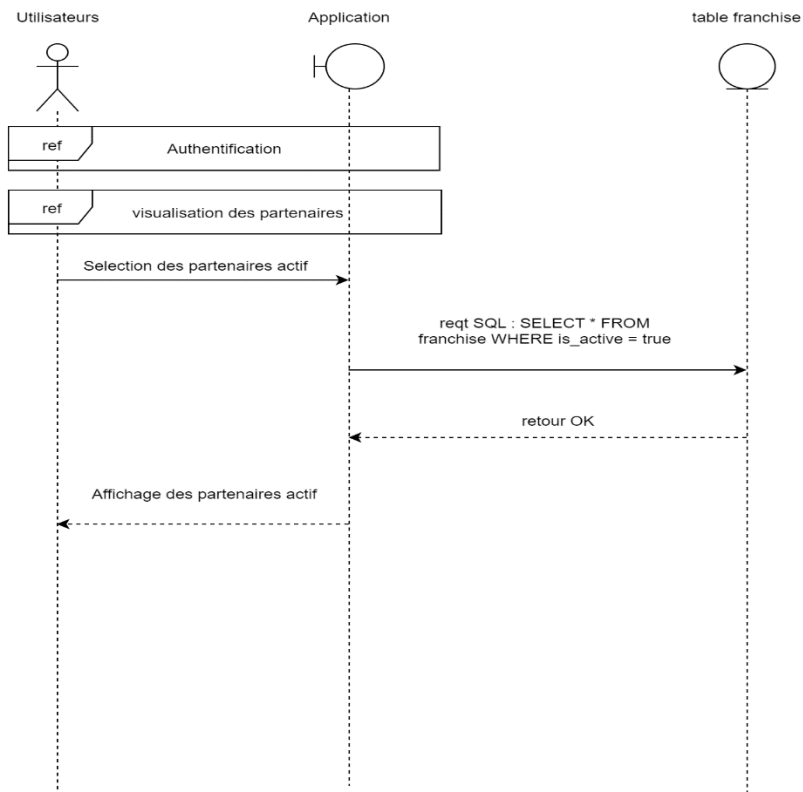


### Diagramme de cas d'utilisation de modification des permissions

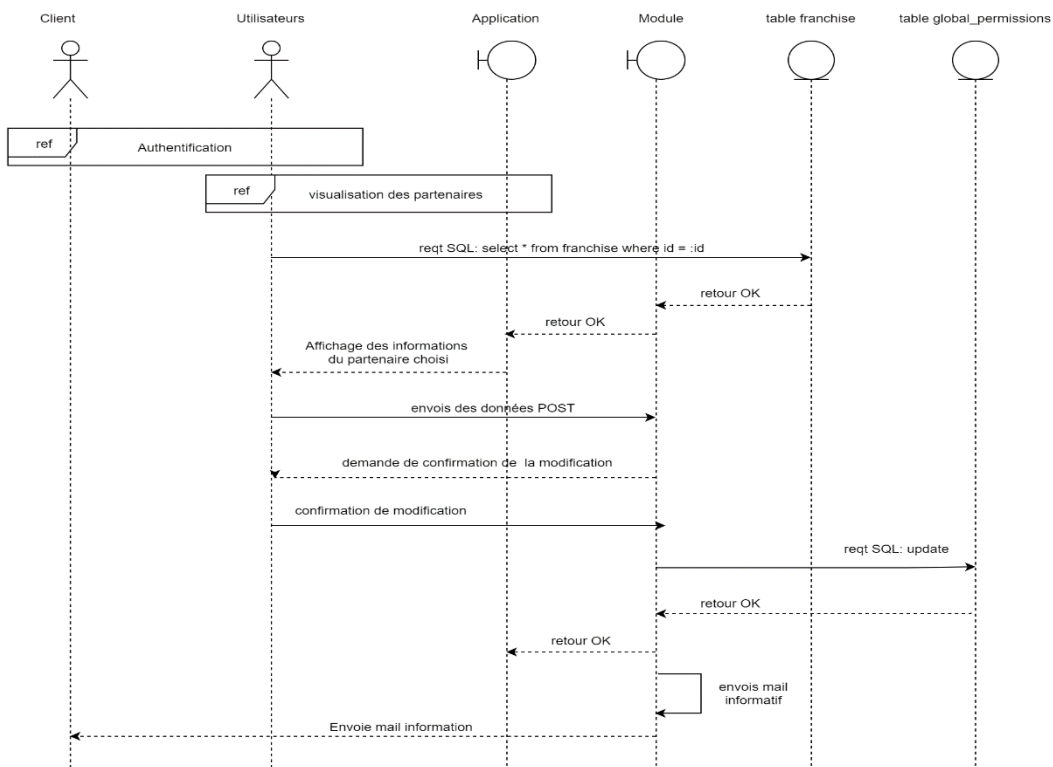


## 2. Diagramme de séquence

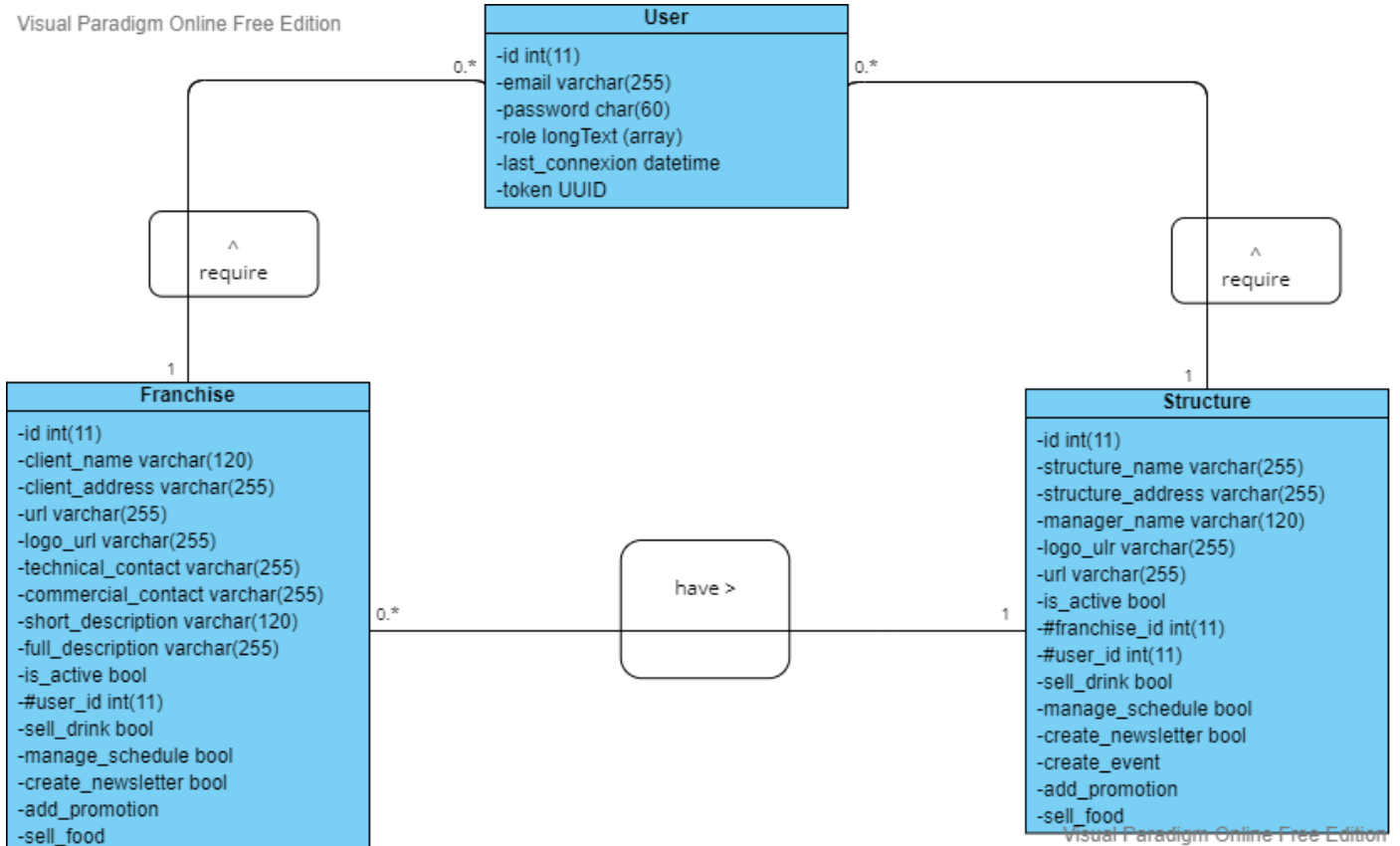
### Diagramme de séquence de sélection des partenaires actifs



### Diagramme de séquence de modification des permissions



### 3. Diagramme Merise



## II. Stack technique et Architecture logicielle

### 1. Stack technique **Front-End**

Concernant ce projet, la stack technique que j'ai choisi pour mener à bien la réalisation des différents composants, contraintes techniques, éléments de sécurité et fonctionnalités est la suivante :



**HTML5** pour créer et représenter le contenu de mes pages web et leur structure.



**CSS3** pour la mise en forme des pages html.



**Javascript Vanilla** en tant que langage de programmation orienté front-end. J'ai choisi de ne pas utiliser de Framework front-end (React, VueJs, Angular) sur ce projet car ses besoins techniques n'en nécessitaient pas. De plus l'utilisation d'un Framework, alourdie de façon conséquente l'application surtout si son utilisation est minime et peu efficiente par rapport aux outils de base du langage.



Comme nous le verrons par la suite, ce projet est développé principalement en php sous le Framework Symfony, de ce fait l'utilisation du **moteur de template Twig** était essentiel.

On notera que je n'ai pas choisi d'utiliser de Framework CSS tel que Bootstrap ou un préprocesseur comme SASS sur ce projet pour les mêmes raisons que je n'ai pas souhaité inclure de Framework Javascript.

## 2. Stack technique **Back-End**



**PHP8** est le langage de programmation composant quasiment l'intégralité du projet. Ce langage est ancien et encore très utilisé donc sa documentation et les ressources liées à son utilisation sont régulièrement mise à jour.



J'ai beaucoup travaillé avec le Framework **Symfony** sur mes précédents projets, surtout concernant la sécurité, l'authentification et la gestion des utilisateurs. Ce Framework offre beaucoup d'outils simples correspondant parfaitement aux besoins du projet. De plus, sa documentation est facile d'utilisation et régulièrement mise à jour, tout comme les ressources en lien avec le Framework.



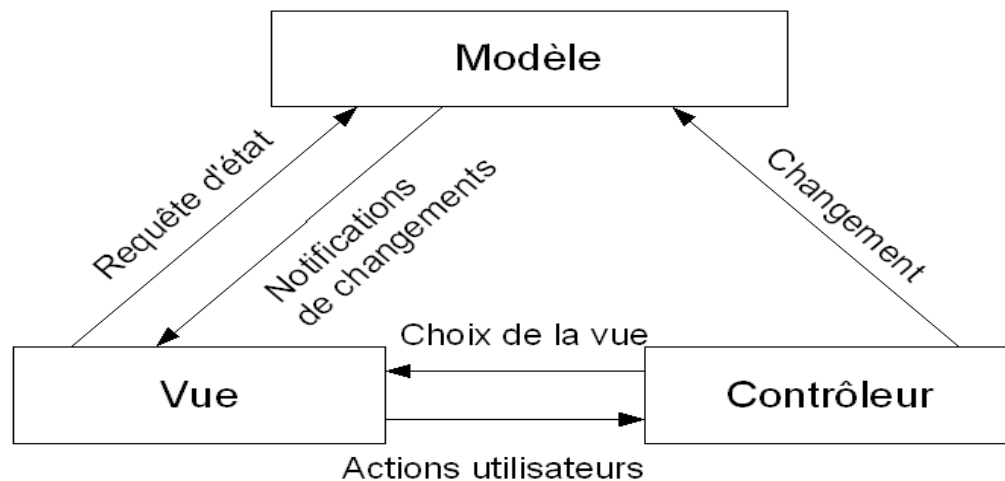
**SQL** est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles



**MariaDB** est un système de gestion de base de données (SGBD) efficace qui est régulièrement mis à jour. Le choix de son utilisation pour le projet est dans ce cas purement arbitraire, PostgreSQL aurait aussi parfaitement fonctionné.

### 3. Architecture logicielle

L'architecture logicielle la plus efficace avec les technologies utilisées et aussi une des plus utilisées sur une application de ce type est l'architecture Model View Controller (MVC). Cette architecture est inhérente au fonctionnement du Framework Symfony.



#### **Modèle :**

Noyau de l'application qui gère les données, permet de récupérer les informations dans la base de données, de les organiser pour qu'elles puissent ensuite être traitées par le contrôleur.

#### **Vue :**

Composant d'affichage de l'interface qui permet de présenter les données du modèle à l'utilisateur.

#### **Contrôleur :**

Composant responsable des prises de décision, gère la logique du code qui prend des décisions, il est l'intermédiaire entre le modèle et la vue.

#### **⊕ Avantages :**

- Meilleure organisation du code.
- Diminution de la complexité lors de la conception.
- Conception claire et efficace grâce à la séparation des données de la vue et du contrôleur.

#### **⊖ Inconvénients :**

- Augmentation de la complexité lors de l'implantation.
- Architecture complexe pour des petits projets.
- Le nombre important de fichiers représente une charge non négligeable dans un projet.

## III.Sécurité

### 1) Formulaires

Les formulaires composant l'application sont des formulaires qui n'interprètent pas les symboles spéciaux comme les chevrons (< >) et autres symboles pouvant permettre à un attaquant de procéder à l'injection de données, commande via les champs des formulaires.

Ensuite, les champs des formulaires comme les adresses électroniques, mot de passe... sont soumis à validation de leur type de donnée empêchant l'utilisateur de saisir une adresse électronique non valide, d'afficher en clair le mot de passe ou même d'importer un type de fichier non valide.

### 2) Mot de passe

Le mot passe d'un compte utilisateur est obligatoirement encrypté avant son insertion dans la base de données. Comme nous le savons un mot de passe quel qu'il soit ne doit jamais apparaître en clair en base de données.

Pour plus de sécurité chaque mot de passe est soumis à une validation de la robustesse de ce dernier. En effet chaque mot de passe doit respecter les règles de base suivantes concernant sa création :

- Le mot de passe doit contenir au **minimum 8 caractères**.
- Le mot de passe doit contenir **une ou plusieurs majuscules**.
- Le mot de passe doit contenir **des chiffres**.
- Le mot de passe doit contenir **un ou plusieurs caractères spéciaux**.

Lors de la création d'une franchise ou d'une structure l'utilisateur associé à cette entité reçoit une notification par mail contenant l'adresse électronique de connexion ainsi qu'un lien vers la page de modification de mot de passe contenant un token unique associé à son compte.

De plus, pour plus de sécurité, pour modifier son mot de passe l'utilisateur doit confirmer l'adresse électronique associé à son compte. A chaque changement de mot de passe un nouveau token est généré et est associé à l'utilisateur. Rendant ainsi le lien de changement de mot de passe inutilisable dès le premier changement de mot de passe.

### 3) Modification des structures/franchises

La modification des données concernant les structures et les franchises ne sont accessibles que par l'équipe de la marque. Lors d'une modification d'une ou plusieurs informations (nom, adresse, permissions) une fenêtre s'ouvre pour demander à l'utilisateur de valider ou non les changements apportés à l'entité en question.

Si l'utilisateur annule ses modifications, la fenêtre se ferme et aucun changement n'est effectué. Dans le cas contraire, les modifications vont s'effectuées directement et un mail va être envoyé au responsable de la franchise s'il s'agit d'une modification sur une franchise. Lors de modifications sur une structure le responsable de la franchise liée à la structure et le responsable de la structure vont être notifiés.



#### 4) Accès à la plateforme

Les utilisateurs faisant partie de l'équipe d'administration n'ont pas de restriction particulière concernant l'accès à la plateforme. Cependant les comptes administrateurs ne peuvent pas être créés depuis l'application. Uniquement un accès direct et une insertion directe en base de données le permettent.

Les comptes franchise et structure doivent avoir leur entité associée active pour qu'ils puissent se connecter à la plateforme. Dans le cas d'un utilisateur souhaitant se connecter et dont son entité est soit inactive soit inexistante il sera redirigé vers une page l'informant de l'inactivité de sa franchise/structure.

Les utilisateurs autres que les administrateurs ont accès aux informations de leur franchise ou structure seulement dans un cadre de lecture seule aucune modification n'est possible à part le changement de mot de passe du compte.

#### 5) Base de données

La base de données effectue automatiquement des backups de façon régulière. De plus, cette dernière est couplée à InnoDB qui est un moteur de stockage pour les SGBD comme MariaDB permettant de lire les données alors que les tables sont verrouillées.