RAPPORT DE PROJET : EMA

Emergency Managing Assistant

Equipe n°2 :

BONAMY Alexis (LD)

BORDET Marie (LD)

CARDOT Clément (LD)

CHAMOULEAU Aurélie (LD)

MARTIN-SEVESTRE Titouan (CSS)

ORSONI Nicolas (CSS)

Table des matières

Présentation de l'infrastructure

Lors de la conception de l'infrastructure afin de supporter le projet Emergency Managing Assistant (E.M.A), nous avons utilisé la solution de l'Entreprise Hashicorp : Vagrant. Cette dernière nous a permis de mettre en place plusieurs machines virtuelles. En effet nous avons donc trois machines virtuelles supportant les trois modules de notre infrastructure suivants : Le serveur reverse Proxy, Le serveur web Apache 2 et pour finir un serveur contenant la base de données MySQL.

Nous allons donc aborder la mise en place de chaque serveur.

-Le serveur web est donc le cœur notre projet car il permet de supporter le client web. Situer à l'adresse IP 192.168.56.80 comme tous les autres serveur ce dernier est basé sur la distribution Debian 11. De plus nous avons installé apache 2 et l'extension PHP. Par la suite nous avons automatiser le téléversement du site web sur ce serveur avec Vagrant.

- La base de données est hébergé sur une VM distinct du site web. Il est installé sur la VM Maria DB ainsi que apache2, l'extension PHP et phpMyAdmin pour faciliter la gestion de la base de données et vérifier son bon fonctionnement. Une sauvegarde de la base de données est faite chaque heure afin de garder une base de données de secours en cas de problème sur la VM. Le site phpMyAdmin est accessible sur l'adresse IP 192.168.56.81.

-Le serveur reverse proxy est l'écran de protection du serveur web avec l'aide d'apache2 et de ses modules proxy, proxy pass et proxy reverse pass permettant de renvoyer les informations contenues dans le serveur web. Ce dernier permet de protéger l'architecture de notre site web. La connexion au site web se fait donc sur l'adresse IP 192.168.56.82 même si le port 80 du serveur web est toujours accessible.

Le point principal d'amélioration est la sécurité. En effet, à l'heure actuelle, il est possible de se connecter à la base de données et au site web directement sans passer par le proxy. Il faut donc fermer les ports d'accès à ces 2 VM pour bloquer les communications. Il faut cependant garder les communications venant du proxy sur certain port précis et entre les 2 VM pour conserver l'accès à la base de données pour le site web.

Pour la base de données, il faut retirer le service apache2 ainsi que phpMyAdmin. Nous avons fait le choix de le conserver pour le moment afin de simplifier le travail de développement du site web.

Pour le proxy, il faut générer un certificat SSL afin de permettre la connexion en HTTPS. De plus, il faut rediriger les connexions au port 3306 sur la base de données pour permettre l'utilisation de l'application Java.

Présentation du site web

La partie web est utilisée exclusivement par les *utilisateurs*, admin ou pas, mais il n'y a aucun moyen d'exercer des fonctionnalités propres aux admins via cette interface. En revanche, il sera possible de modifier/ajouter des tickets *événements* et de visualiser les informations clés des *événements*, *effectifs*, *véhicules* et *infrastructures*. On retrouvera ces fonctionnalités dans le diagramme ci-dessous :

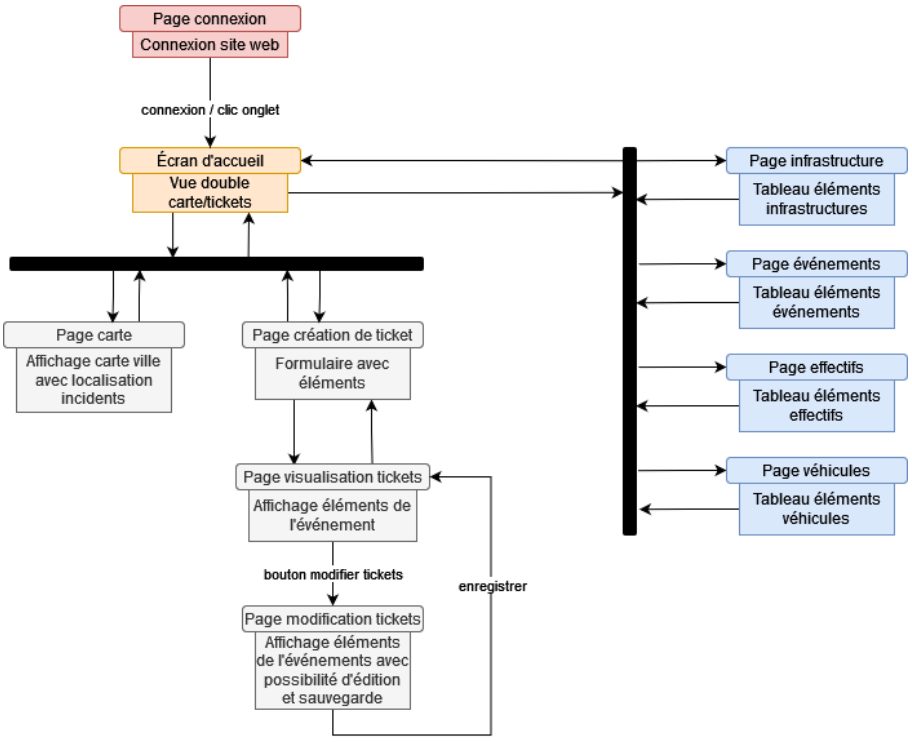

Figure 1 - Diagramme d'état-transition du client WEB

Par défaut, *l'utilisateur* sera sur la page connexion. Une fois connecté il aura accès à l'écran d'accueil ci-dessous :

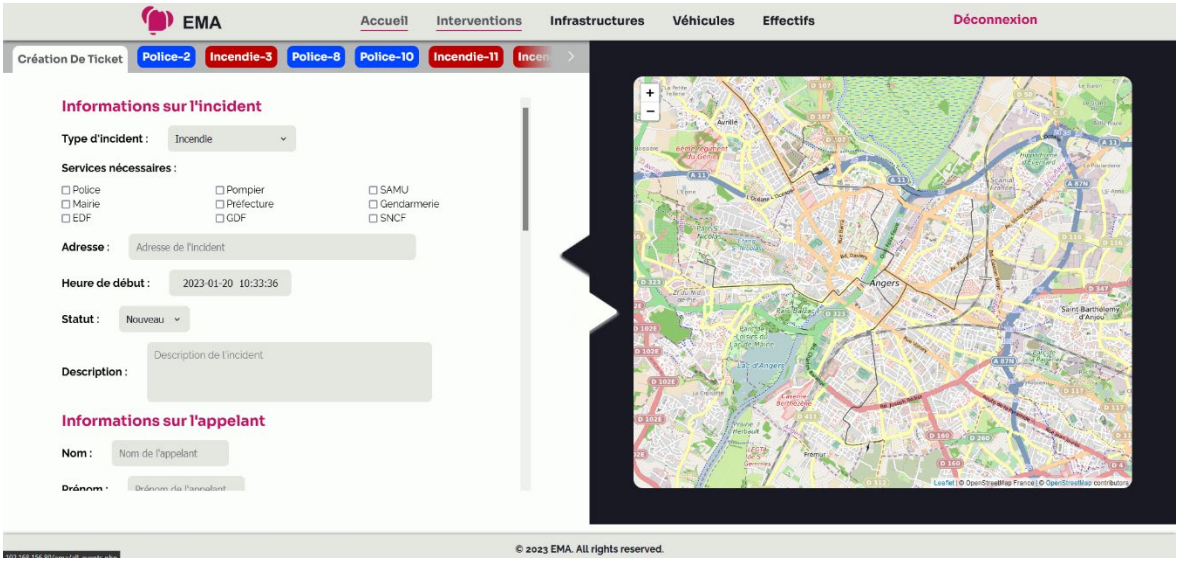

Figure 2 - Capture d'écran de la page d'accueil du client WEB

Cette page est à regarder de pair avec le diagramme d'état transition. Dans la barre de navigation on retrouvera les onglets permettant de passer d'une page à l'autre (*symbolisée par la moitié droite du diagramme*). Ces pages affichent les données essentielles sous forme de tableau. Une amélioration qui aurait pu être apportée à cette partie aurait été l'affichage sous forme de cartes, de taille variable en fonction de la quantité de données à montrer. Enfin, une fonction filtre a été envisagée pour n'afficher que certains types de tickets (*par statut, par type, etc.…*).

Sur la page d'accueil, en plus de cette barre de navigation, on va retrouver deux éléments visibles sur la capture d'écran, à savoir la page carte et la page *événements*. Les deux sont séparés mais peuvent être agrandis en cliquant sur les flèches centrales.

Sur la partie carte on pourra retrouver un plan de la ville ainsi que les incidents déclarés via un élément de signalisation permettant de visualiser la localisation précise des *interventions*. Par ailleurs, nous souhaitions afficher les *véhicules* en récupérant les coordonnées GPS des transports en commun Irigo pour les afficher comme s'il s'agissait de véhicule d'intervention dans un contexte de simulation. Nous avons réussi à obtenir les coordonnées de ces *véhicules,* à les appliquer dans la base de données sur les *véhicules* que nous avons créés et à les récupérer dans la partie web, mais nous avons manqué de temps pour les afficher sur la carte.

Sur la partie ticket d'*événement* on aura par défaut l'interface permettant de créer les *événements*. Cette interface permet de rentrer les informations principales à savoir :

- Le type d'incident,
- L'adresse,
- L'heure de début,
- Une description de l'incident,
- Les coordonnées de l'*appelant*,
- L'affectation d'*effectifs* et des *véhicules* à la nouvelle mission.

Une fois l'*événement* créé il apparaît dans les onglets de la partie supérieure de la page tickets, et il est alors possible en cliquant dessus d'en obtenir les détails. L'*utilisateur* pourra modifier les éléments

de ce ticket à loisirs. Il est également à noter que la modification d'un ticket n'a pas été totalement terminée : la modification des affectations des effectifs et véhicules n'est pas intégralement implémentée. Enfin, l'*utilisateur* pourra clôturer le ticket ce qui le marquera comme terminé.

Présentation de l'application JAVA

L'application Java se lance avec une page de connexion. Il faut rentrer son adresse mail et son mot de passe pour se connecter et accéder au reste de l'application.

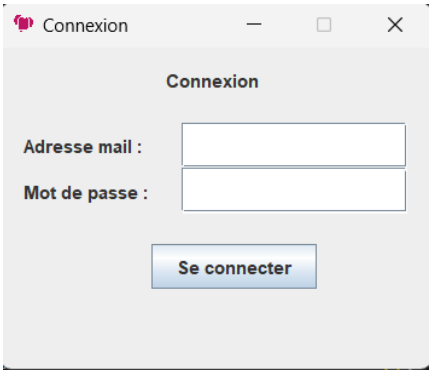

Figure 3 - Interface de connexion du client JAVA

Dans l'application nous retrouvons plusieurs onglets :

Users **Callers** **Rescuers** **Infrastructures** **Vehicles** **Events** **Statistics**

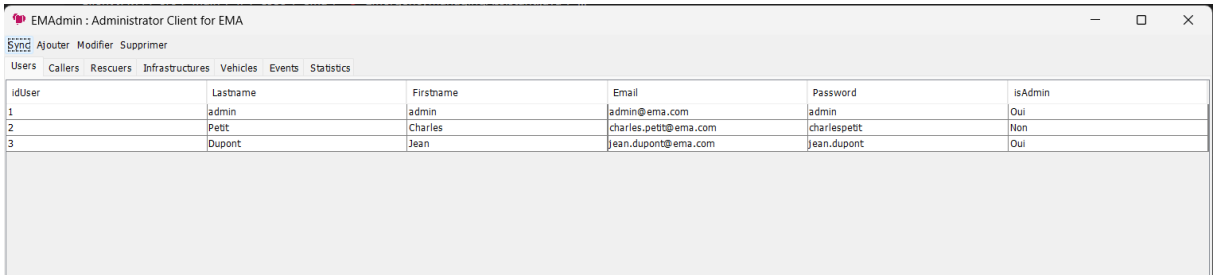

Figure 4 - Interface client JAVA

Dans tous les onglets excepté "Statistics", on retrouve un tableau avec toutes les entrées de la base de données. Des boutons permettent de :

- Modifier une ou plusieurs information(s)
- Supprimer une ligne de données
- Ajouter une entrée à la base de données

Ses fonctions ne sont pas disponibles pour les données de la table "Event". En effet, les *évènements* sont des données à risque et pour des raisons de sécurité, quand un *évènement* est passé comme terminé, il devient figé et il ne peut plus être modifié.

Lorsque l'on veut modifier une information, une fenêtre complémentaire apparaît avec tous les champs qui peuvent être modifiés.

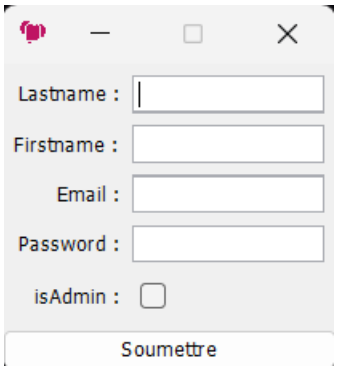

Figure 5 - Interface du formulaire utilisateurs du client JAVA

La synchronisation à la base de données se fait à chaque modification depuis l'application Java (modification, suppression ou ajout) mais lorsqu'un changement est effectué depuis le client Web, il n'y a pas de synchronisation automatique sur l'application Java. Pour pallier ce problème, nous avons mis en place une fonctionnalité de synchronisation à intervalle régulier mais cette fonction n'est pas encore stable. C'est pour cela que nous avons ajouté un bouton "Sync" qui permet de synchroniser l'application Java à la base de données.

L'onglet "Statistics" comprend quant à lui deux diagrammes. Un diagramme en secteurs qui représente la répartition des différents types d'*évènements* (Incendie, Secours à la personne, Police, Secours routier et Opérations Diverses). Le deuxième diagramme est un graphique représentant le nombre d'*évènements* par mois sur l'année en cours. On y retrouve le nombre d'*évènements* global mais aussi le nombre d'*évènements* nécessitait les pompiers, les forces de police et les urgences médicales.

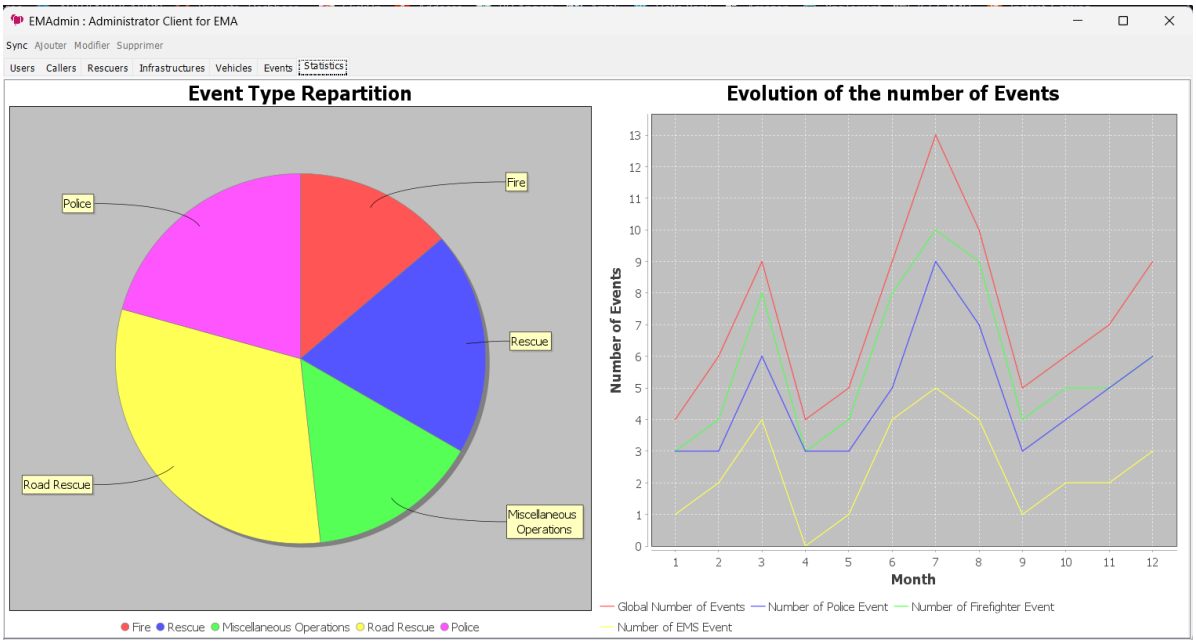

Figure 6 - Page statistiques du client JAVA

Pour la conception de l'application Java, nous avons d'abord crée un système de base de données locale avec la création d'une classe pour chaque table de la base de données. Ces classes reprennent la même structure que la base de données et un connecteur JBDC permet de faire le lien entre les deux

bases de données. Grâce à cette connexion, nous sommes capables de récupérer les informations, de les modifier, d'en ajouter et d'en supprimer.

Ensuite, nous avons fait la page de connexion et les pages principales de l'application. Puis, nous avons fait le lien entre les deux parties pour que la partie principale de l'application ne s'affiche que lorsque l'utilisateur s'est connecté et que l'application a vérifié que c'était un administrateur.

Voici quelques diagrammes pour illustrer le projet Java :

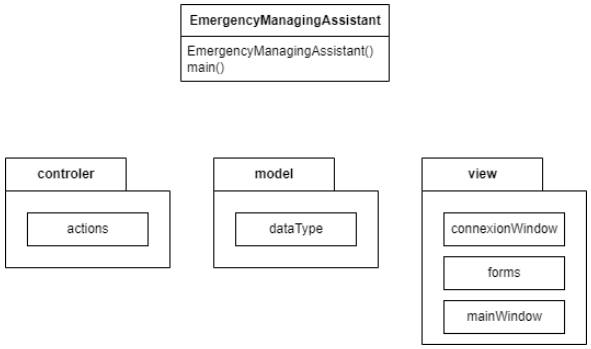

Figure 7 - Modélisation répertoires du client JAVA

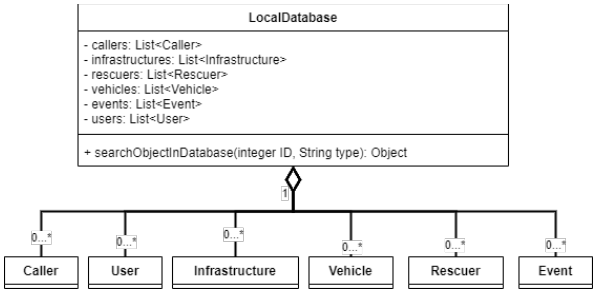

Figure 8 - Modélisation des classes « model » du client JAVA

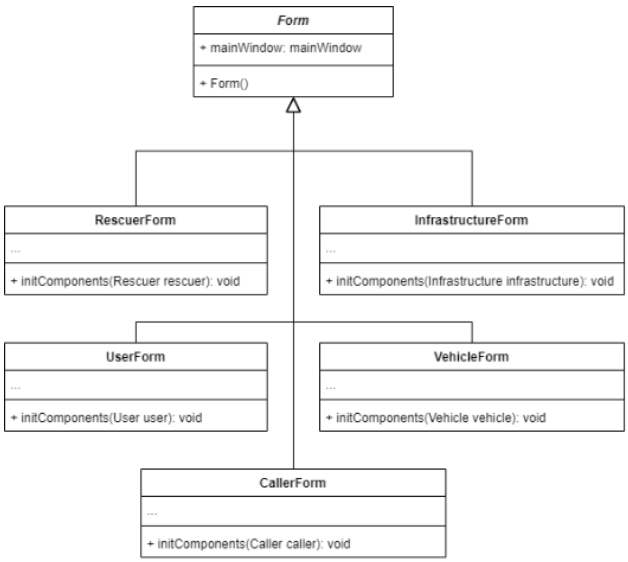

Figure 9 - Modélisation des Formulaires du client JAVA

Présentation de la base de données

Lors de la conception de la base de données, nous avons tout d'abord listé les différents éléments et acteurs interagissant avec notre système. On retrouve donc :

Les **Appelants** (Caller), correspondent aux personnes appelant les urgences pour signaler un incident.

Les **Utilisateurs** (User), correspondent aux opérateurs du Centre de Traitement des Appels et aux administrateurs du système.

Les **Infrastructures** (Infrastructure), correspondent aux différents centres de secours (commissariats, caserne de pompiers, hôpitaux...)

Les **Effectifs** (Rescuer), correspondent aux pompiers, policiers, ambulanciers et autres employés des services d'urgences.

- On peut affecter **des** *effectifs* à **un** *véhicule*
- On peut affecter **des** *effectifs* à **une** *infrastructure*

Les **Véhicules** (Vehicle), correspondent aux différents véhicules des différents services (véhicules léger, ambulances, fourgon pompe-tonne, hélicoptère)

- On peut affecter **des** *véhicules* à **une** *infrastructure*
- On peut affecter **des** *véhicules* à **une** *intervention*

Les **Interventions** (Event), correspondent aux différentes interventions des différents services (incendie, secours à la personne, secours routier, police …)

- On peut affecter **des** *interventions* à **un** *appelant*
- On peut affecter **des** *interventions* à **un** *utilisateur*

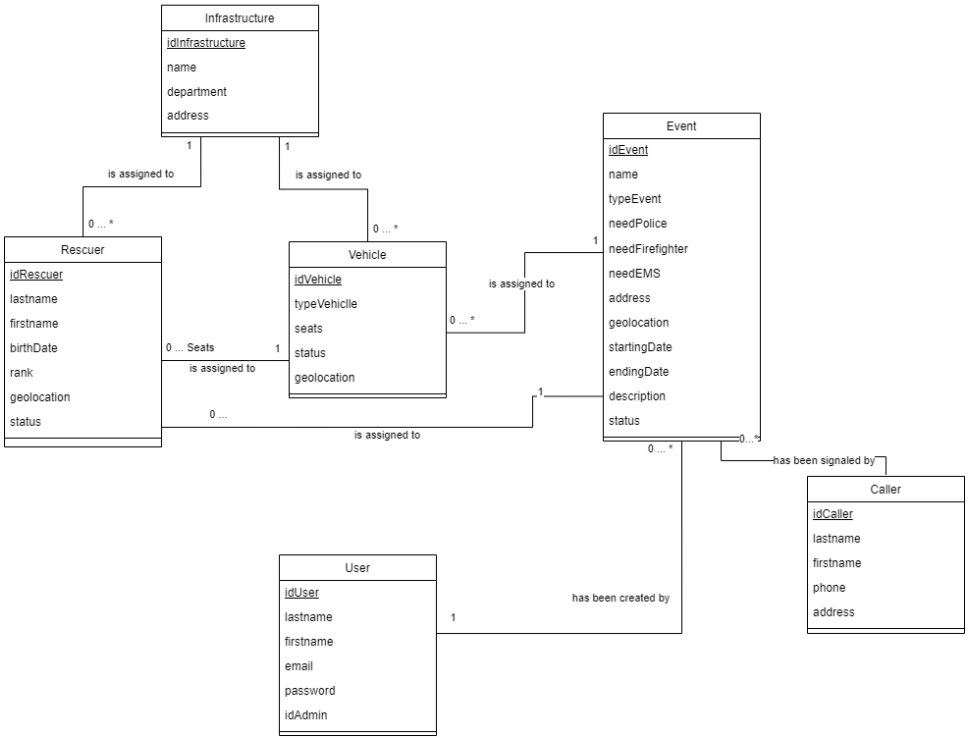

Figure 10 - Modélisation conceptuelle de la base de données

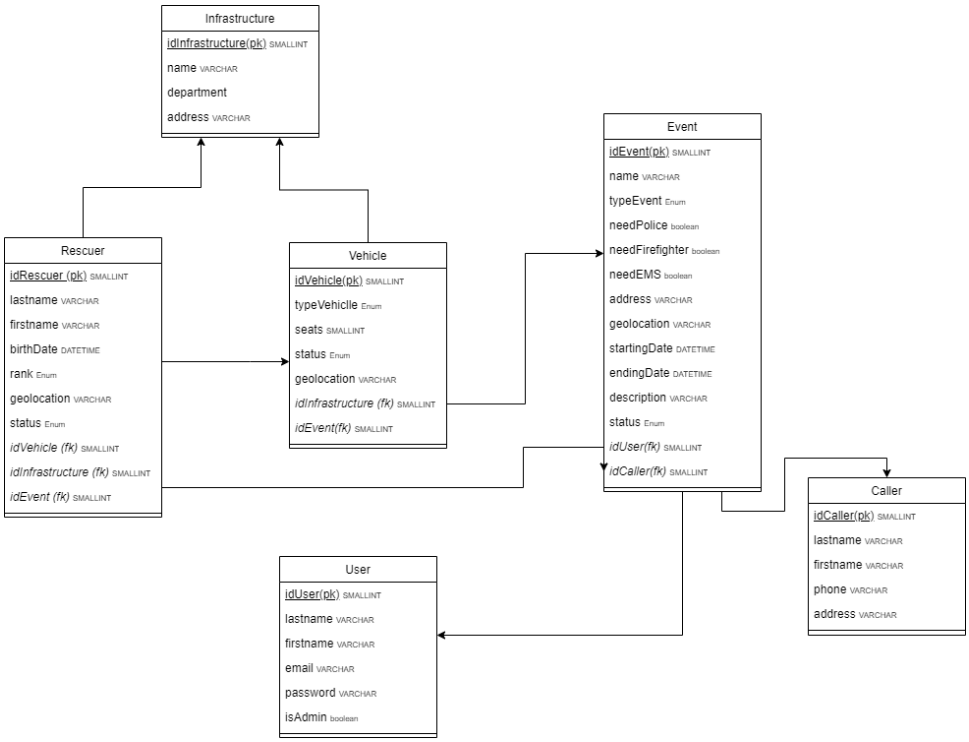

Figure 11 - Modélisation logique de la base de données

Architecture globale du projet

Notre projet se structure de la façon suivante :

- Un réseau externe composé du/des navigateur(s) web du/des client(s) et du/des client(s) Java.

- Un réseau interne composé des trois serveurs : Reverse proxy, Web et Base de données.

Tout d'abord pour l'utilisation web un navigateur se connecte à l'adresse IP : 192.168.56.82 qui est l'adresse de notre serveur proxy. Ce dernier va renvoyer les informations du site web situé à l'adresse IP : 192.168.56.80. Pour finir, dans le cadre de son fonctionnement le serveur web communique avec la base de données situé à l'adresse IP : 192.168.56.81.

Dans le cadre de l'utilisation du client Java ce dernier communique avec la base de données situé à l'adresse IP : 192.168.56.81.

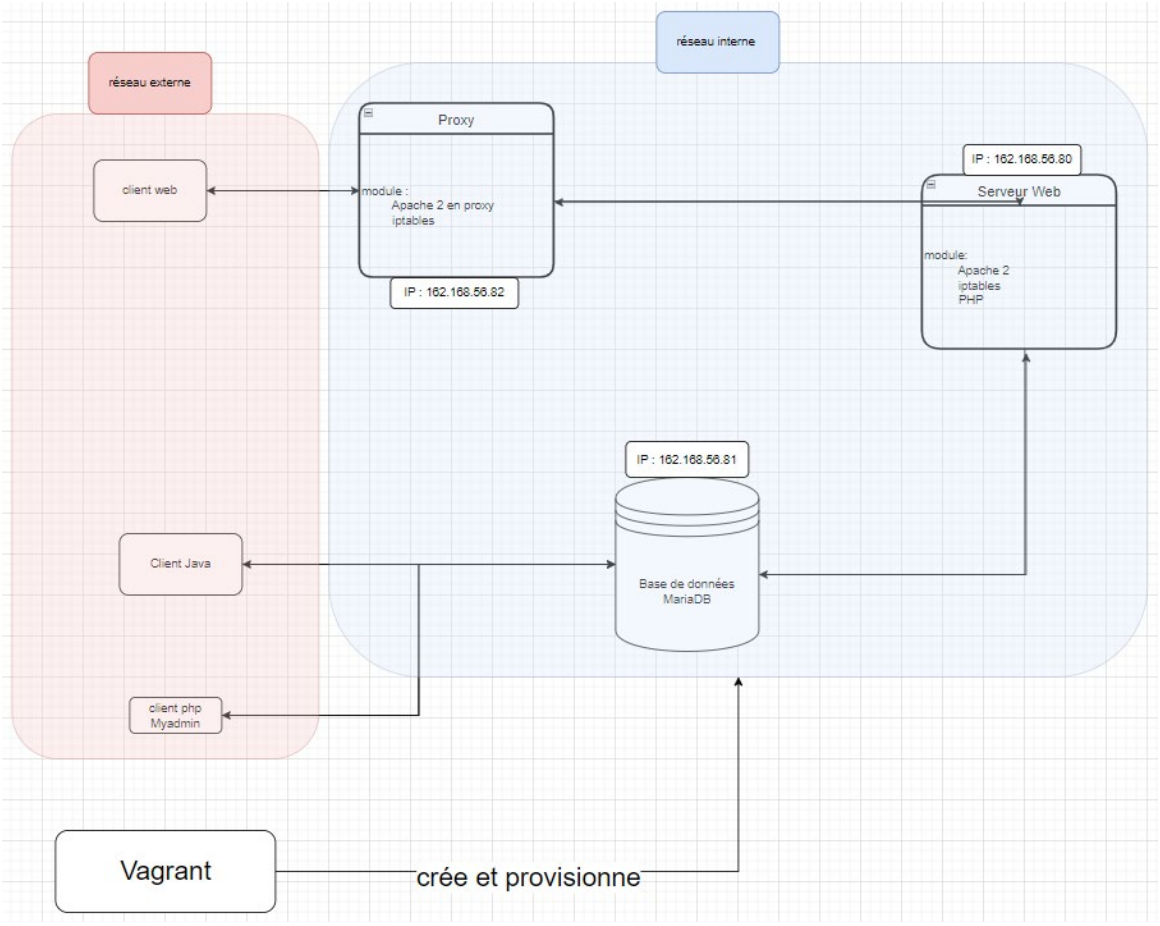

Figure 12 - Modélisation de l'architecture globale de l'infrastructures d'EMA

Scénario de Démonstration

Scénario de démonstration WEB

1. Connexion au site WEB (login : *charles.petit@ema.com*, password : *charlespetit*)
2. Page d'accueil, vérification de présence des éléments suivants :
    a. Onglets Interventions, Infrastructures, Véhicules et Effectifs
    b. Bouton Déconnexion
    c. Sur moitié gauche : création et visualisation des tickets avec un onglet pour la création de tickets
    d. Sur moitié droite : carte ville
    e. Entre les deux parties : flèches permettant l'agrandissement des deux parties
3. Interface création de ticket :
    a. Affichage des différents attributs
    b. Auto-complétion des adresses saisies (*ESEO Angers*)
    c. Heure de début saisie et mise à jour automatiquement
    d. Remplissage classique des autres éléments
    e. Enregistrement du ticket
    f. Apparition d'une pin à l'emplacement de l'ESEO sur la carte
    g. Apparition d'un onglet supplémentaire avec les informations du nouvel événement
    h. Dans le nouvel onglet, les informations correspondent à celles saisies
4. Modification de ticket :
    a. Modification de certains attributs
    b. Enregistrement des modifications
    c. Modifications enregistrées affichées
5. Onglets Interventions, Infrastructures, Véhicules et Effectifs
    a. Affichage des données avec l'ensemble des paramètres
    b. Les données sont conformes aux modifications réalisées précédemment
6. Déconnexion

Scénario de démonstration JAVA

1. Lancement du client via EmergencyMananingAssistant.java
    a. Lecture des logs (connexion à la BDD, récupération de tous les éléments)
    b. La fenêtre de connexion s'affiche
2. Essaie de connexion avec un utilisateur non-administrateur
    a. La connexion est refusée
3. Essaie de connexion avec un utilisateur administrateur
    a. La connexion est acceptée
    b. La fenêtre de connexion se ferme
    c. La fenêtre principale s'affiche
4. Présentation des différents onglets (Caller, Rescuer, Vehicle…)
5. Démonstration de la fonctionnalité d'ajout
    a. Ajout d'un nouvel utilisateur -> Utilisateur ajouté dans la BDD
    b. Ajout d'un nouveau véhicule -> Véhicule ajouté dans la BDD
6. Démonstration de la fonctionnalité de modification
    a. Modification d'une infrastructure -> Infrastructure modifié dans la BDD
    b. Modification d'un effectif -> Effectif modifié dans la BDD
7. Démonstration de la fonctionnalité de suppression
    a. Suppression d'un appelant -> Appelant supprimé dans la BDD
    b. Suppression d'une intervention -> Bouton d'action grisé car action interdite
8. Présentation de l'onglet Statistics
    a. Affichage d'un diagramme en secteur et d'un graphique d'évolutions des interventions
9. Présentation du bouton de Synchronisation manuelle
10. Présentation du script python permettant d'émuler la position des véhicules

Planning Détaillé

Partie Infrastructure

Tâche	Séance	Développeur
Prise en main de Vagrant	6	Nicolas & Titouan
Mise en place du serveur Web	7	Nicolas
Mise en place base de données + sauvegarde	7	Titouan
Mise en place du Proxy Pass Reverse	8	Nicolas & Titouan
Debug	9	Nicolas & Titouan
Tentative ip tables	10	Nicolas & Titouan

Partie Web

Tâche	Séance	Développeur
Maquette HTML/CSS	4-6	Aurélie & Alexis
PHP avec la BDD	7	Aurélie & Alexis
Fonction de création d'un Event	8	Aurélie
Intégration Open Street Map	9	Aurélie
Visualisations Events, Infrastructures, Effectifs et Véhicules	8-9	Alexis
Debug	10	Aurélie & Alexis

Partie Java

Tâche	Séance	Développeur
Classes orientées objets	6	Clément & Marie
Connecteurs JAVA/BDD	7	Clément
ToolBox SQL	8	Clément
Fonctionalités BDD locale	8	Clément
IHM - Connexion	7 à 10	Marie
IHM – Navigation BDD	9 à 10	Clément
Debug	-	TODO