Rapport de Projet
PWEB 2022 : Application Full Stack
Casernes de pompiers de Paris

Yohan RUDNY 209

Antonin RASKOPF 209

Table des matières

Description de l'application

Intérêt collaboratif du projet

Dans le cadre de ce projet, nous avions le choix parmi une multitude de services à proposer qui pourraient être délivrés sous forme d'une carte interactive en ligne. L'objectif premier était pour nous **l'utilité autant publique que privée de cette application**. Cela veut dire que notre cahier des charges devait correspondre à celui d'une application développée à des fins collaboratives mais également pratique pour n'importe qui voudrait l'utiliser. Il se devait également d'inclure toutes les contraintes imposées par les consignes de base (Service backend, système de connexion/création de compte, écrans de profil etc…).

Après multiples réflexions et exploration parmi toutes les données qu'il était possible de traiter : Nous avons choisi de mettre en place **une carte répertoriant toutes les casernes de pompier de Paris et sa grande ceinture.** L'intérêt premier de ce service serait **d'accroitre la facilité de listage et d'accès à ces casernes** de pompiers pour les secouristes ayant besoin d'intervenir rapidement **mais aussi** pour tous usagers susceptibles de devoir se rendre à une caserne de pompier précise se situant dans la grande ceinture de Paris.

La collaboration est alors ici au cœur de notre service. Tous les usagers seront facilement identifiables car il est nécessaire de se créer un compte et de s'y connecter afin d'utiliser le service. Dans un contexte où beaucoup de secouristes d'une région donnée **doivent se rendre rapidement sur plusieurs sites différents** pour des interventions ou simplement pour de la logistique : Cette carte interactive se présente comme une solution beaucoup plus rapide et fiable que des services d'itinéraire classique comme Google Maps ou Waze.

De plus, étant donné que toutes les casernes présentes sont communes à tous les utilisateurs du site : Il est plus facile pour les pompiers de se repérer et de se donner rendez-vous avec d'autres secouristes à l'aide de cette carte que d'un autre service de géolocalisation sur internet.

API publique utilisée

Notre recherche de données de géolocalisation qui seraient intéressantes à exploiter s'est basée sur des sites officiels d'API tel que Open Data Paris (https://opendata.paris.fr) et Data Gouv (https://www.data.gouv.fr/).

Nous avons finalement trouvé des données au format **json** contenant les géolocalisations, les adresses et les secteurs de toutes les casernes de pompier de Paris sur **Data Gouv** (https://www.data.gouv.fr/fr/datasets/liste-des-casernes-a-paris-et-dans-les-departements-de-la-petite-couronne-551678/) et téléchargeables sur le site **Data Smartidf** (https://data.smartidf.services/explore/dataset/liste-des-casernes-a-paris-et-dans-les-departements-de-la-petite-couronne).

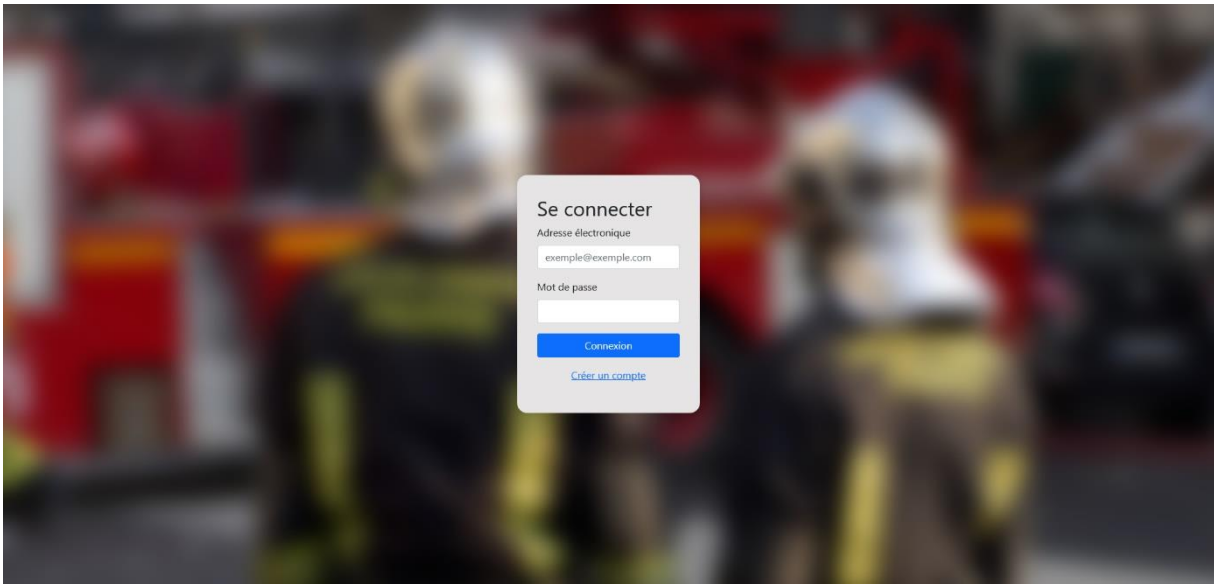
Cette liste contient 77 casernes de pompiers différentes situées dans la grande ceinture de Paris et un grand nombre d'informations différentes sur celles-ci qui pourront être efficacement exploitées dans le cadre de notre application collaborative.

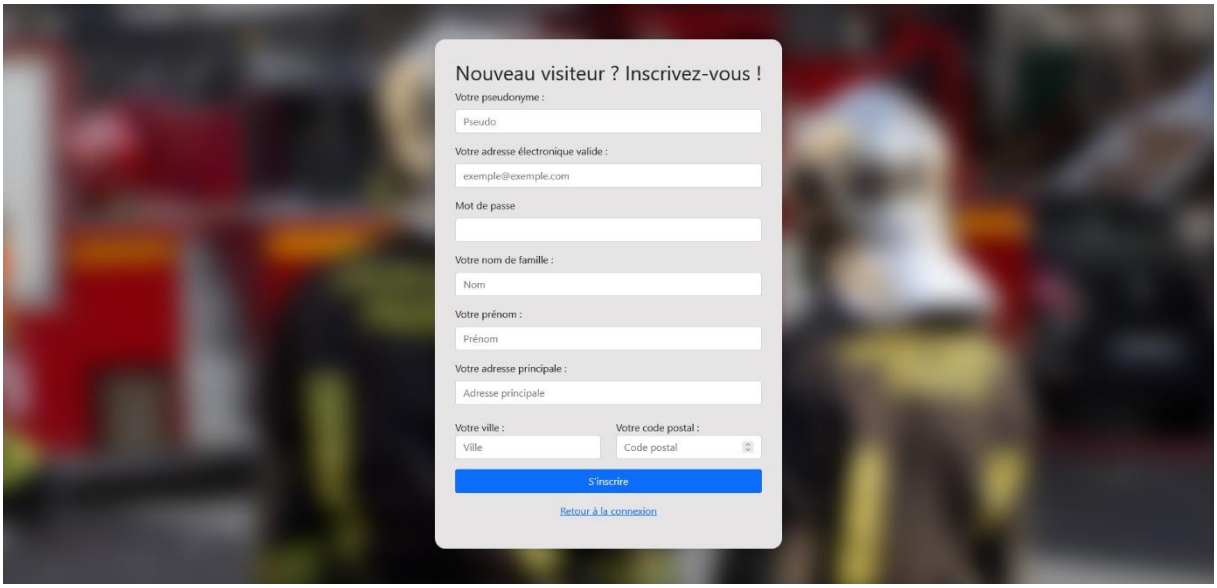
deno_cs	grpt	cie	tel_ville	tel_feu	tel_auto	adresse	pc_cie	em_grpt	geo_point_2d
BLANCHE	1	7	01 40 23 20 28	01 48 74 56 20	35 070	28 rue Blanche 75009 Paris	1	0	48.8785745152848 25, 2.332714097...
CLICHY-SOUS-BOIS	1	14	01 45 09 40 10	01 45 09 40 00	35 140	2 al du Chne Pointu 93390 Clichy...	1	0	48.9048233544908 65, 2.5427779...
LA COURNEUVE	1	26	01 48 36 32 64	01 48 36 32 61	35 262	24 rue de la Convention 93120 La...	0	0	48.9287084496505 1, 2.388892543...
PANTIN	1	10	01 48 45 60 41	01 48 45 06 37	35 102	93-95 rue du Cartier Bresson 93...	0	0	48.8992151623036 4, 2.408006343...
BITCHE	1	10	01 44 65 94 48	01 40 36 86 03	35 101	2 place Bitche 75019 Paris	0	0	48.8892152764077 4, 2.3806023103...
PARMENTIER	1	11	01 43 57 27 43	01 43 57 58 69	35 111	87 ave Parmentier 75011 Paris	0	0	48.8659051793727 95, 2.374537220...
SEVIGNE	2	11	01 44 61 50 28	01 42 72 17 84	35 110	7 rue de Svign 75004 Paris	1	0	48.8559841961222 4, 2.3626145608...
RUEIL-MALMAISON	3	28	01 47 49 47 09	01 47 49 18 18	35 283	112 rte de l'Empereur 92500 Rueil-...	0	0	48.8650064032030 8, 2.1793831112...
BOURG-LA-REINE	3	21	01 46 63 09 48	01 47 02 09 38	35 212	20 rue Ravon 92340 Bourg-la-Rei...	0	0	48.7789770039564 3, 2.3199611800 1...
MASSENA	2	2	01 45 83 82 66	01 45 83 80 80	35 020	37 bld Massna 75013 Paris	1	1	48.8223749239365 8, 2.374391600...
MAISON-ALFORT	2	17	01 43 78 32 04	01 43 78 28 10	35 172	4-6 rue Pasteur 94700 Maisons-...	0	0	48.8002924422322 3, 2.430344019...
CRETEIL	2	17	01 49 80 28 17	01 49 80 28 18	35 170	10-18 rue de l'Orme St-Simon 940...	1	0	48.7864781814556 64, 2.45882466...
CHOISY-LE-ROI	2	22	01 48 52 33 40	01 48 84 09 47	35 221	56-58 rue Jules Vals 94600 Choi...	0	0	48.7543292869202 75, 2.410470166...
MONTREUIL	1	24	01 42 87 20 31	01 42 87 00 02	35 240	11 ave Pasteur 93105 Montreuil/Bo...	1	0	48.8640657598817 2, 2.439918544...
ASNIERES	3	27	01 47 99 71 19	01 47 99 05 01	35 271	4 rue du Cne Bossard 92600 Asni...	0	0	48.9237570735027 84, 2.28513668...
MENILMONTANT	1	12	01 40 31 44 18	01 40 31 72 47	35 120	47 rue St-Fargeau 75020 Paris	1	0	48.8719320261020 66, 2.404627087...
COLOMBIER	3	4	01 45 48 50 70	01 45 48 32 90	35 040	11 rue du Vieux Colombier 75006 ...	1	0	48.8511293584917 75, 2.3321877233...
SAINT HONORE	1	7	01 42 61 03 26	01 42 61 03 93	35 071	10 rue St-Anne 75001 Paris	0	0	48.8658264192511 85, 2.336208772...
SAINT CLOUD	3	16	01 55 39 39 28	01 47 71 05 20	35 163	40 ave du Marchal Foch 92210 St...	0	0	48.8477731023841 2, 2.2109435835...
ANTONY	3	21	01 46 66 09 90	01 46 66 07 51	35 211	2 ave Armand Guillebeaud 92160...	0	0	48.7491139141844 9, 2.3051193512 07...
CHAMPIGNY	2	15	01 48 81 55 45	01 48 81 03 97	35 150	16 rue de Dunkerque 94500 Cha...	1	0	48.8084898612267 7, 2.531643033...
RUNGIS	2	22	01 46 86 90 10	01 47 26 90 20	35 220	382 ave de Stalingrad 94669 Ch...	1	0	48.7600415899873 1, 2.3675779830...
AULNAY-SOUS-BOIS	1	13	01 43 83 53 33	01 48 83 76 28	35 130	156 rte de Mitry 93600 Aulnay	1	0	48.9481447734854 24, 2.5146597171...
DRANCY	1	13	01 48 32 02 64	01 48 32 02 63	35 132	19-21 rue Roger Salengro 93700 D...	0	0	48.9204570780292 4, 2.440845316...
SAINT DENIS	1	26	01 48 13 85 28	01 48 13 85 18	35 260	Fort de la Briche Che du Fort de l...	1	0	48.9452078406480 2, 2.3413577107...
SAINT OUEN	1	9	01 41 66 49 28	01 40 11 12 56	35 093	89 rue du Dc Bauer 93400 Saint-...	0	0	48.908221630426 8, 2.34152916875...
GENNEVILLIERS	3	27	01 47 94 18 14	01 47 94 38 94	35 270	136-140 rue Henri Barbusse 9223...	1	0	48.9298610016063 8, 2.302037083...
COLOMBES	3	27	01 42 42 00 63	01 42 42 14 56	35 272	20 rue Hoche 92700 Colombes	0	0	48.9227582102336 6, 2.262550581...
MALAR	3	4	01 45 55 08 81	01 47 05 41 99	35 042	7 rue Malar 75007 Paris	0	0	48.8616025805900 6, 2.3066019107...
LANDON	1	10	01 40 35 58 31	01 40 35 74 58	35 100	QUAI DE VALMY	1	0	48.8816663035710 6, 2.3693146591...
BOURSAULT	1	9	01 45 22 43 13	01 45 22 36 77	35 091	27 rue Boursault 75017 Paris	0	0	48.8840156247903 6, 2.3202946791...
DAUPHINE	3	5	01 45 53 84 68	01 45 53 80 11	35 051	8 rue Mnil 75016 Paris	0	0	48.8685407579953 7, 2.28483989...
AUTEUIL	3	6	01 42 88 76 20	01 42 88 51 81	35 061	2-4 rue Franois Millet 75016 Paris	0	0	48.8507837234889 95, 2.2738130 8...
CHARONNE	1	12	01 43 71 51 22	01 43 71 53 66	35 121	93 rue des Pymes 75020 Paris	0	0	48.8548089179347 1, 2.4059587 87...
CHATEAU D'EAU	1	8	01 40 03 98 28	01 40 40 03 33	35 081	50 rue de Chteau-d'Eau 75010 Pa...	0	0	48.8739130707958, 2.3592529188

Fonctionnalités détaillées

Comme dit précédemment : L'objectif de notre site est de faciliter avant tout le listage des casernes de pompiers de Paris et sa grande ceinture. Néanmoins, les fonctionnalités présentes sont bien plus nombreuses et renforce le côté pratique, l'ergonomie générale et l'expérience de l'utilisateur.

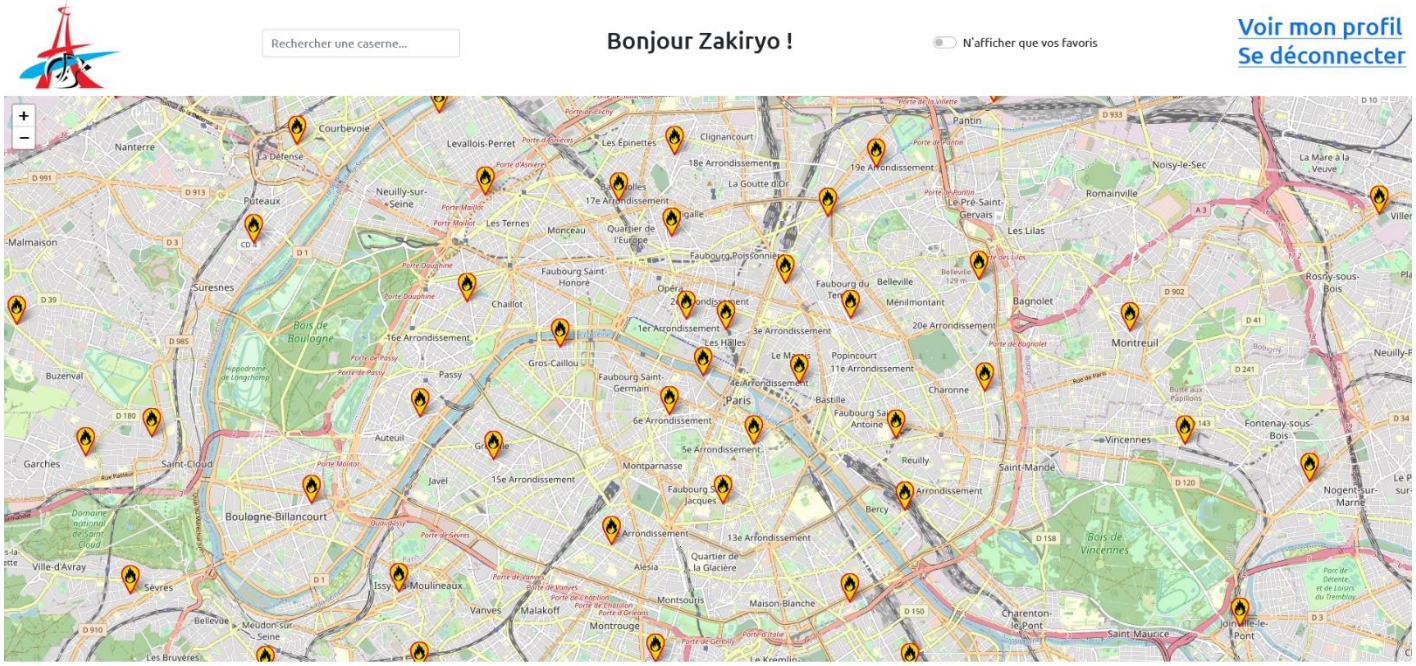
Tout d'abord : Notre site fonctionne **uniquement** si l'utilisateur est **enregistré dans notre base de données** et **connecté** à son compte. Un écran de connexion et de création de compte **sécurisé** et ergonomique est alors mis en place permettant à l'utilisateur de renseigner ses informations qui ne seront utilisé que pour son utilisation personnelle du service.





Le fonctionnement technique des deux formulaires est expliqué dans la partie « Sécurité de l'application » de ce rapport. L'adresse est utilisé pour la fonctionnalité d'itinéraires qu'offre le service et donc nécessaire à la création du compte.

Une fois connecté à l'aide de son adresse électronique et de son mot de passe : L'utilisateur se retrouve directement sur la carte présentant toutes les casernes de pompier de Paris marquées par des marqueurs personnalisé en forme de flamme ainsi que son pseudonyme en haut de l'écran.

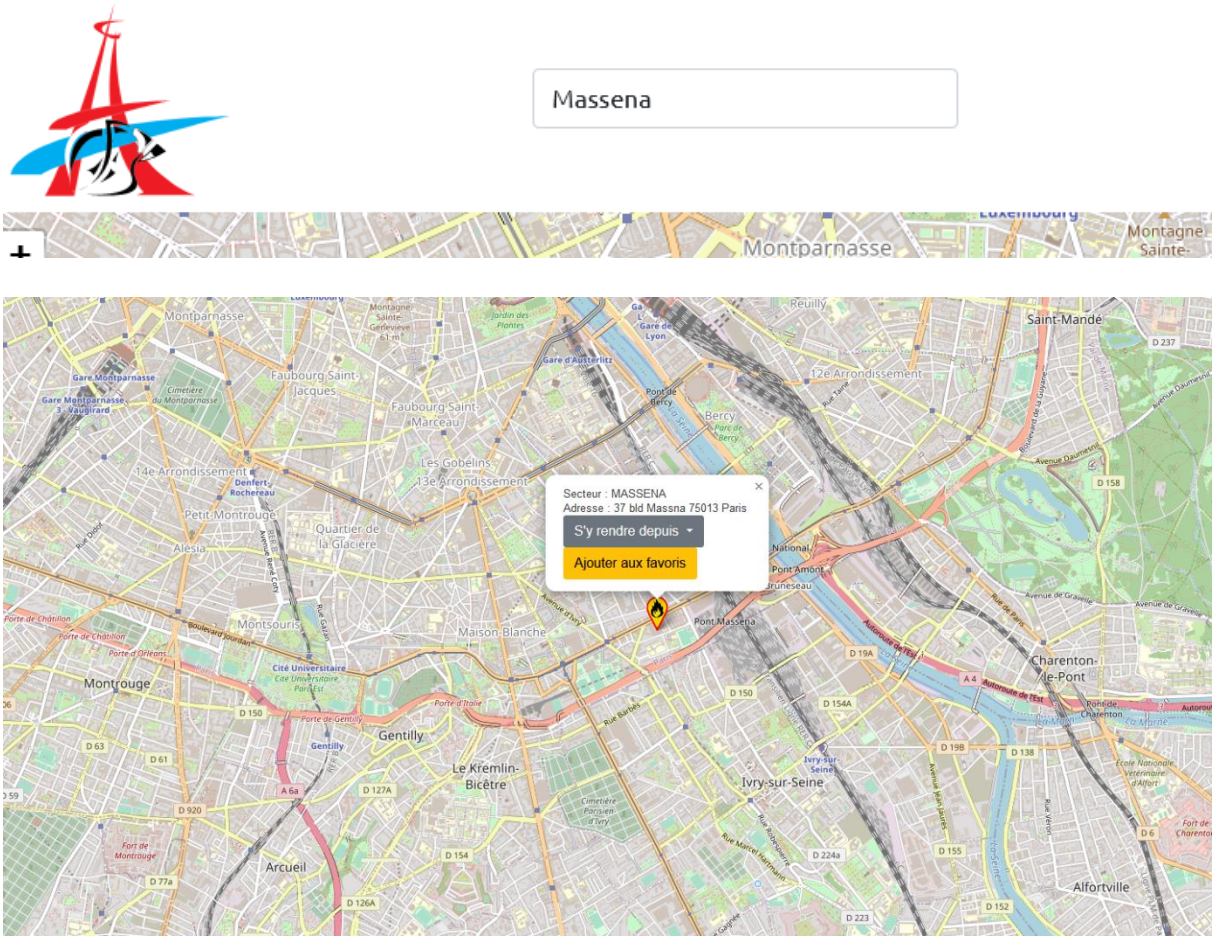


Cet écran représente l'interface utilisateur en elle-même et comporte plusieurs fonctionnalités liées aux différentes casernes présentes sur la carte.

Les 77 casernes de pompiers récupérées de l'API sont ici recensées sur une carte générée à l'aide de l'API **Leaflet** et **Open Street Map** dans un **LayerGroup** nommé « stations » et initialisé au chargement du document.

Lors de la sélection d'un marqueur, un menu contextuel s'ouvre permettant d'accéder aux différentes fonctionnalités offertes à l'utilisateur qui sont expliquées plus loin dans ce rapport.

Il est tout d'abord possible de **rechercher une caserne en particulier à l'aide de la barre de recherche** située en haut à gauche de l'écran. En saisissant le secteur de la caserne recherchée : Tous les autres marqueurs dont le secteur ne contient pas la chaîne de caractères recherchée disparaissent pour n'afficher que le(s) résultat(s) trouvé(s).



D'un point de vue technique : La barre de recherche est un sélecteur jQuery **keyup** invoquant une fonction « **searchStation** ». Celle-ci a pour rôle de filtrer, dans tous les marqueurs du LayerGroup « **stations** », ceux dont la Pop-up contient la chaîne de caractères saisie et donc le secteur recherché pour ainsi masquer de la carte toutes les casernes ne comportant pas cette même chaîne de caractères.

À noter qu'étant donné que Leaflet ne délivre pas de fonctionnalité pour masquer un marqueur sans le supprimer, il est nécessaire de stocker les marqueurs supprimés dans un tableau « **save** » qui permettra de replacer dynamiquement ceux-ci sur la carte une fois la chaîne de caractère supprimée de la barre de recherche.

L'application embarque également **un système de favoris** pour chaque utilisateur enregistré. Celui-ci permet à chaque utilisateur d'ajouter une ou plusieurs casernes de pompier à leurs favoris afin de n'afficher que les casernes qu'ils fréquentes le plus et faciliter leur lisibilité.

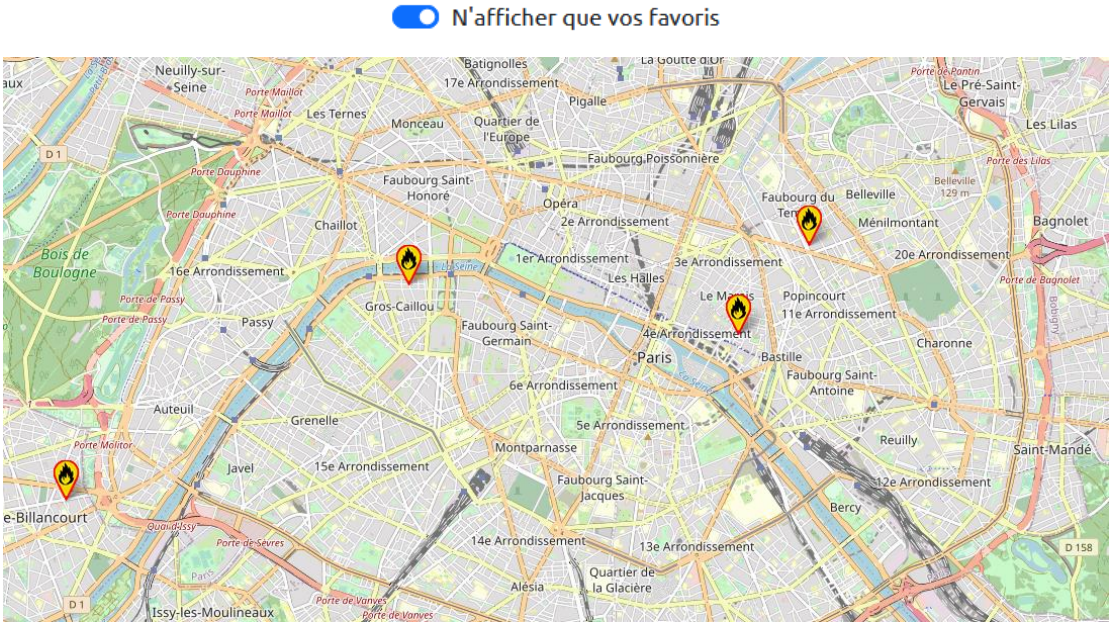
Cela se fait directement sur la Pop-up d'une caserne sélectionnée. Un bouton « **Ajouter au favoris** » sera cliquable, ce qui ajoutera côté serveur la caserne à la liste des favoris de l'utilisateur connecté. Si la caserne se trouve déjà dans les favoris de l'utilisateur, un bouton « Supprimer des favoris » sera affiché en conséquence faisant l'action inverse.

L'action d'ajouter un favori aura pour conséquence d'incrémenter la table « favorites » de la base de données du site avec comme paramètres :

- Un identifiant de favori (clé primaire de table) qui s'incrémente automatiquement.
- L'identifiant de l'utilisateur connecté (clé étrangère vers la table « users »).
- Le secteur de la caserne ajoutée aux favoris.

Les détails techniques sont expliqué dans la troisième partie du rapport.

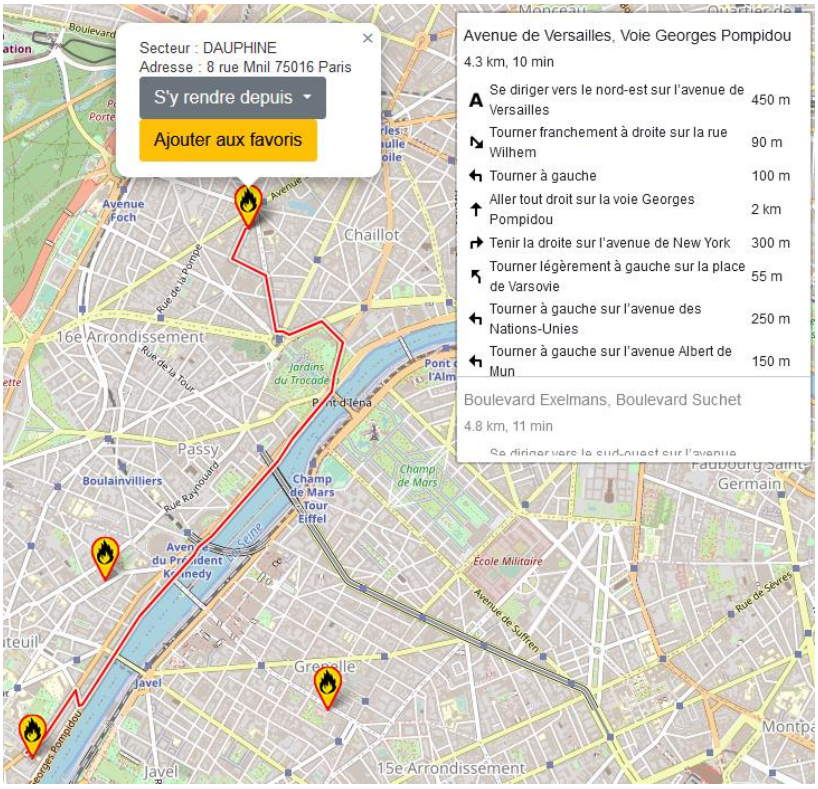
Un « Switch » est mis à disposition en haut à droite de l'écran afin de n'afficher que les favoris de l'utilisateur connecté.



La seconde fonctionnalité que l'on retrouve sur chaque marqueur de la carte est la **création d'un itinéraire vers la caserne**. Celle-ci représente la fonctionnalité principale de l'application. L'utilisateur connecté possède une adresse principale, saisie lors de la création de son compte et a aussi la possibilité de saisir plusieurs adresses secondaires dans le menu de configuration de son compte (fonctionnalité détaillée plus tard).

Toutes les adresses (principale et secondaires) de l'utilisateur connecté sont chargée dans une liste déroulant que l'on retrouve en cliquant sur le bouton « **S'y rendre depuis** » d'une Pop-up.

Au choix de l'une des adresses par l'utilisateur, **<u>un itinéraire en voiture sera immédiatement tracé de l'adresse choisie à la caserne sélectionnée</u>** avec tous les détails du trajet sur la droite de l'écran.



L'itinéraire disparaît automatiquement lors de la fermeture de la Pop-up. D'un point de vue technique, 3 nouvelles API sont utilisées :

Leaflet-Control-Geocoder, **Leaflet-Routing-Machine** et **Nominatim**.

Les deux premières API permettent de créer un itinéraire à l'écran à l'aide de coordonnées (longitude et latitude) puis d'afficher tous les détails du trajet à l'écran. **Nominatim** permet de retourner des coordonnées GPS à l'aide d'une adresse donnée.

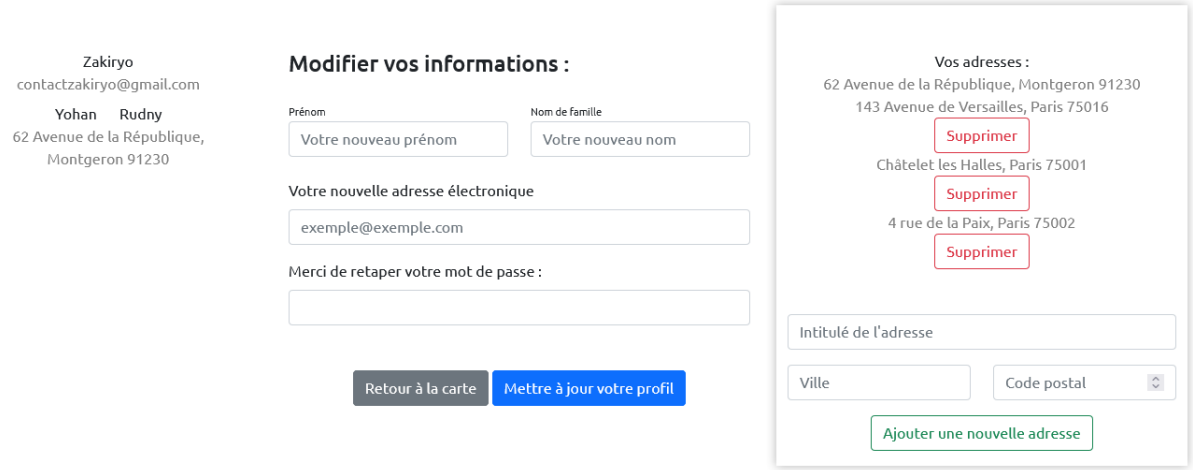
La combinaison de ces trois API permettent de créer une fonction prenant en paramètres **des coordonnées cible ainsi qu'une adresse structurée afin de retourner un itinéraire entre les deux points**. L'adresse sélectionnée est ici passée en paramètre avec les coordonnées de la caserne de pompier cible afin de créer un itinéraire précis entre les deux points.

À noter que **si l'adresse saisie par l'utilisateur n'est pas valide** : **Aucun itinéraire ne sera créé** mais l'application restera fonctionnelle.

```
/**
 * Créer un itinéraire entre le marqueur sélectionné et l'adresse sélectionnée dans la liste déroulante
 * @param {*} coords Les coordonnées de la caserne cible
 * @param {string} address Libellé de l'adresse sélectionnée
 */
function createRoute(coords, address) {
    const pointedAddress = function () {
        var tmp = null;
        $.ajax({
            async: false,
            type: "POST",
            url: `https://nominatim.openstreetmap.org/search?q=${encodeURI(address)}&format=json`,
            datatype: "json",
            success:
                function (data) {
                    tmp = data;
                }
        });
        return tmp;
    }();
    routing = L.Routing.control({
        waypoints: [
            L.latLng(pointedAddress[0].lat, pointedAddress[0].lon),
            L.latLng(coords.split(','))
        ],
        router: new L.Routing.osrmv1({
            language: 'fr',
            profile: 'car'
        })
    }).addTo(map);
}
```

Cette fonctionnalité d'itinéraires couplée à celle des favoris **facilite grandement la collaboration entre secouristes** et l'ergonomie a été pensée de sorte que **la création d'un itinéraire se fasse bien plus rapidement que depuis un autre service**.

Depuis le site, l'utilisateur a la possibilité de **modifier les informations de son profil et d'ajouter des adresses secondaires** depuis le menu « **Voir mon profil** » en haut à droit de l'écran.



Depuis cet écran, l'utilisateur connecté peut :

- Consulter ses informations de connexions
- Modifier son nom, prénom et adresse électronique
- Ajouter une nouvelle adresse secondaire
- Supprimer l'une de ses adresses secondaires

Deux formulaires sécurisés sont mis en place : **L'édition des informations et l'ajout d'une nouvelle adresse**.

Ceux-ci communiquent directement les champs saisis à la **base de données du site**. Toutes les adresses sont stockées dans la table « **addresses** » et les informations de l'utilisateurs sont stockées dans la table « **users** ».

La soumission des saisies des formulaires invoque des scripts **PHP** qui permettent de vérifier les champs, se connecter et mettre à jour la base de données côté serveur.

Tous ces scripts se trouvent dans le répertoire « **phpFunctions** » du projet.

Les saisis sont contrôlés automatiquement d'une part par **Bootstrap** qui vérifie le bon format d'une adresse mail, la longueur maximum de chaque entrée ainsi que le format numérique et la longueur du code postal.

Enfin, le bouton « **Se déconnecter** » de l'écran principal détruis la session en cours de l'utilisateur et le redirige vers la page de connexion.

Points clés de développement

Langages utilisés

Côté client : Nous avons utilisé pour ce projet tous les langages vus en cours. Pour le frontend de l'application (Interface utilisateur, carte interactive et design général), l'entièreté du développement s'est fait en **JavaScript** avec la librairie jQuery et le balisage en **HTML/CSS**.

Nous n'avions pas d'autres choix qui s'offraient à nous lors du développement car nous ne connaissons aucune base de **React** ou encore de **TypeScript**. Nous n'y sommes néanmoins pas fermé dans le cas d'un projet futur pour lequel ces langages nous aurons été enseignés.

L'utilisation de la librairie jQuery a été nécessaire pour les sélecteurs de l'application afin d'optimiser au mieux la lisibilité du code.

Côté serveur : Nous avons eu recours au langage **PHP** pour établir la communication entre la base de données et l'utilisateur connecté au site. La base de données a été gérée avec **MySQL** et nous avons donc eu recours au langage **SQL** pour effectuer nos requêtes côté serveur.

Déroulé du développement

Après avoir choisi le thème et les services de notre application : Nous avons structurer notre travail au travers plusieurs étapes de développement.

1) Après avoir élaboré des croquis pour identifier l'identité graphique de notre application : Nous avons structuré notre projet en commençant par mettre en place les éléments graphiques de nos pages, sans fonctionnement côté serveur.

2) Une fois le site visuellement prêt, nous avons commencé le développement de la carte **Leaflet** en exploitant les données des casernes récupérées en format **json**. Nous avons placé les marqueurs correspondant aux casernes et préparé les Pop-up (fonction **initMap** du fichier **mapLoad.js** du projet).

3) Nous avons ensuite mis en place tous les services backend de l'application (création de la base de données, des tables, des fonctions PHP et des formulaires). Cette étape était nécessaire afin de faire fonctionner les fonctionnalités supplémentaires de la carte.

4) Pour finir, nous avons développé les fonctions principales de la carte (Création d'itinéraires, ajouts de favoris, choix des adresses…).

Tout le projet a été testé et est fonctionnel sur MAMP avec les ports 80 pour Apache et Nginx et 8080 pour MySQL.

Difficultés rencontrées

Les principales difficultés à la réalisation de ce projet résidaient en l'utilisation de **requêtes AJAX** pour faire communiquer notre code **JavaScript** avec notre **base de données MySQL**. En effet, **le fonctionnement asynchrone des requêtes AJAX** a causé beaucoup de problèmes d'algorithmie lors du développement de l'application car nous n'étions pas suffisamment expérimenté avec ce type de contraintes. Il nous a alors fallut trouver de multiples solutions algorithmiques afin de palier à ces différents problèmes que nous avons finalement réussi à résoudre.

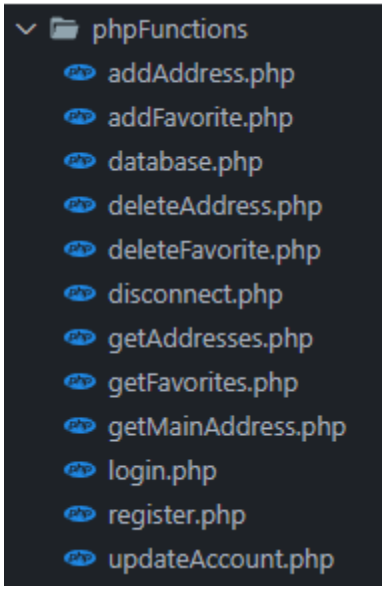
Il nous a également fallut du temps à nous habituer au fonctionnement des requêtes SQL en PHP car nous avions conscience des multiples problèmes de sécurité tel que **les injections SQL ou les failles XSS**.

Services Backend

Sécurité de l'application

Étant donné que notre site embarque un système de création de comptes et de connexion : La sécurité a été notre priorité lors du développement backend.

Notre projet comporte un répertoire « phpFunctions » contenant toutes les interactions que l'application aura avec la base de données du site.



Le fichier « **database.php** » est inclus dans chacun des autres fichiers de l'application afin de se connecter à la base de données.

```
phpFunctions > database.php > ...
1   <?php
2   /**
3    * Connection à la base de données de l'application.
4    */
5   try {
6       $db = new PDO('mysql:host=localhost:8080;dbname=firestationsmapdatabase', 'root', 'root');
7   } catch (Exception $e) {
8       die('Erreur de connexion à la base de données : ' . $e->getMessage());
9   }
```

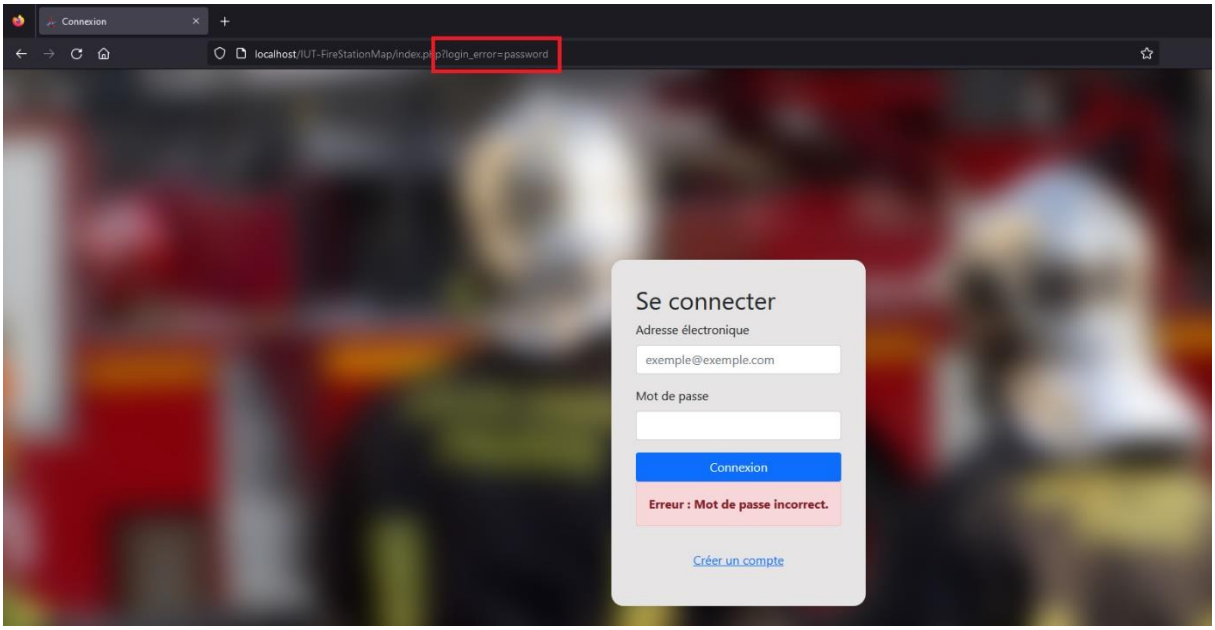
Toutes les requêtes SQL envoyées au serveur par l'une de ces fonctions est protégée contre les **injections SQL** par la fonction « **prepare** » de PHP permettant de préparer les requêtes avant de les exécuter. De la même manière, toutes les saisies seront filtrées par la fonction « **htmlspecialchars** » afin d'éviter l'utilisation de la **faille XSS**.

En ce qui concerne le système de connexion : Le mot de passe saisi par l'utilisateur est **chiffré** à l'aide de la fonction « **password_hash** » et le chiffrement « **PASSWORD_DEFAULT** » correspondant à l'algorithme de chiffrement « **bcrypt** » dont la stratégie repose sur le fait que le hachage change à chaque connexion par l'utilisateur.

Il est ensuite déchiffré à chaque connexion avec la fonction « **password_verify** » qui aura pour rôle de comparer le mot de passe saisi lors de la connexion avec le mot de passe chiffré contenu dans la base de données.

➔ Ce processus permet à ce que les mots de passes des utilisateurs ne soient **ni visibles ni décryptables** dans la base de données par l'administrateur.

En cas d'erreur de saisi, de mot de passe incorrect, d'utilisateur inexistant ou de champs vides : Les codes PHP de soumission de chacun des formulaires du site renvoi à la page initial du formulaire **avec l'erreur en question en paramètre**, ce qui permettra d'afficher un message d'erreur personnalisé à l'utilisateur.



Le même système d'affichage des erreurs est mis en place **pour chacun des formulaires du site** afin d'informer au mieux l'utilisateur de son erreur.

Lors de l'inscription d'un nouvel utilisateur : Celui-ci n**e pourra pas utiliser la même adresse électronique ou le même pseudonyme qu'un utilisateur déjà existant**.

Base de données expliquée de l'application

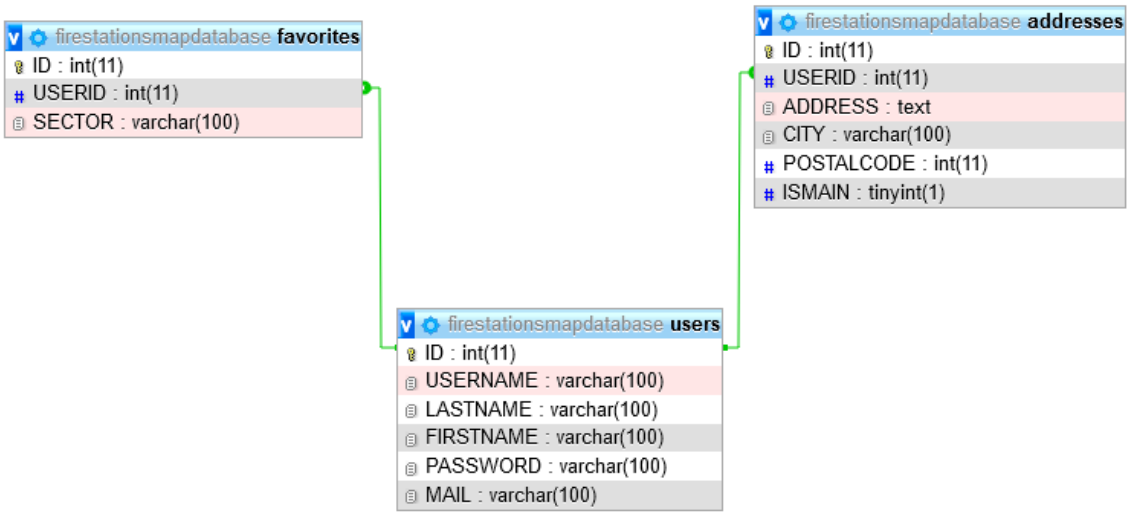
La base de données utilisée par le site se nomme « **firestationsmapdatabase** » et il s'agit d'une base **MySQL** hébergée sur le port **8080**.

Celle-ci comporte trois tables :

- « users » : Contient tous les utilisateurs inscris sur le site et s'incrémente à chaque fois qu'un nouveau visiteur créer un compte. Sa clé primaire est un identifiant unique incrémenté automatiquement à chaque nouvel élément.

- « addresses » : Contient les adresses principales et secondaires de tous les utilisateurs inscris. Elle est liée à la table « users » par une clé étrangère pointant vers l'identifiant unique d'un utilisateur. Chaque utilisateur possède une seule adresse principale et peut avoir plusieurs adresses secondaires.

- « favorites » : Contient toutes les casernes favorites des utilisateurs identifiés ici également par une clé étrangère. Celle-ci s'incrémente à chaque fois qu'une caserne est ajoutée en tant que favorite par un utilisateur identifié et permet le filtrage sur la carte des casernes classiques et favorites de l'utilisateur connecté.



Il est à noter que la taille maximale en nombre de caractères de certains champs est **contrôlée pour chacune des entrées de texte** afin d'éviter les divers problèmes d'insertion.

Évolutions à prévoir

D'autres fonctionnalités intéressantes étaient à la base prévue mais n'ont pas pu être développées par manque de temps et de connaissances.

La principale était de mettre en place un système de mémos sur chacune des casernes présentes sur la carte consultables et incrémentales uniquement par les utilisateurs marqués comme secouristes ou pompiers. L'idée était ici de faciliter la communication entre les différentes personnes du métier et prévenir d'éventuels dangers ou simplement laisser quelques notes pour une station en particulier.

Nous avions alors imaginé un champ supplémentaire dans la table « users » permettant de déterminer le statut de secouriste d'un utilisateur afin qu'il ait accès aux mémos et puisse en ajouter, en lire et en supprimer.