

# *Titre professionnel* *Développeur Logiciel*



## Alès Communication



# Sommaire

1 - Remerciements-----	p 2
2 - Présentation du projet (en anglais) -----	p 3
3 - Présentation de l'entreprise-----	p 4
4 - Liste des compétences couvertes -----	p 5
5 - Cahier des charges -----	p 6
6 - Spécifications fonctionnelles et techniques -----	p 7
7 - Réalisation -----	p 11
=> 7.0 - Planifications des tâches-----	p 11
=> 7.1 - Maquettage-----	p 12
=> 7.2 - Structure du projet-----	p 14
=> 7.3 - Code HTML et génération des éléments du DOM-----	p 15
=> 7.4 - Exploitation du JSON et implémentation du formulaire-----	p 23
=> 7.5 - Création de la BDD et écriture des données-----	p 30
=> 7.6 - Lecture des données et affichage des covoiturages -----	p 35
=> 7.7 - mise en place de la map et de la géolocalisation-----	p 39
=> 7.8 - fonctionnalités supplémentaires-----	p 45
=> 7.9 - Intégration et Mise en production-----	p 50
8 - Conclusion -----	p 53
9 - Annexe- -----	p 54

# 1 - Remerciements :

Je tiens à remercier notre formateur MORGHADI Patrick qui nous a épaulé tout au long de la formation dans notre apprentissage et nos démarches professionnelles.

Merci à Simplon, à ROBERT Emilie Chef de Projet d'Alès Myriapolis, la région Occitanie, l'AFPA, qui ont contribué au lancement de l'école Régionale du Numérique CODA by Simplon Alès.

Merci à toute la promotion CODA Alès 2017-2018, aux camarades avec qui j'ai pu apprendre et partager mes expériences.

Je remercie à Mathieu GRESSE et Alès Communication pour cette période d'entreprise, et pour le projet que nous avons mené. J'ai profité d'excellentes conditions de travail pendant la période de stage.



## 2 - Présentation du Projet :

I am going to present you my internship project. I built a “car pulling” plugin for Skymove, a progressive web app used by the hang-gliders community.

Skymove is developed by Ales Communication. The project started in september 2017, It was ordered by a professional hang-glider who wanted to create the first application dedicated to this sport.

Mathieu Gresse, the manager and my tutor, has created Skymove with the CMS Goodbarber. It's a French CMS dedicated to mobile application.

The users started to request more features to be included in the App such as car pulling, flying sessions management.

Once the CMS mastered, I realized its limits. I couldn't develop advanced functionalities on it. This technology couldn't allow me to realize the project properly, so I decided to code from scratch.

I developed a single and dynamic webpage, and I integrated it in the existing application using iframes technologies.

To successfully achieve my objectives I, used all skills I learned during my professional training HTML, CSS, Javascript, AJAX and JSON, JQuery, PHP, SQL, XML, and I choose to work without framework.

I met some difficulties but I managed to solve them. I liked working on this application and I maintain a good relationship with my tutor. We intend to work again together in the future.

### 3 - Présentation de l'entreprise

Alès communication est une agence digitale basée sur Alès. Créée en 2012, elle a contribué au développement numérique d'un grand nombre d'entreprises du bassin alésien.

Elle propose notamment des campagnes de communications sur les réseaux sociaux pour promouvoir des événements, de sites internet et tout autre support de communication.

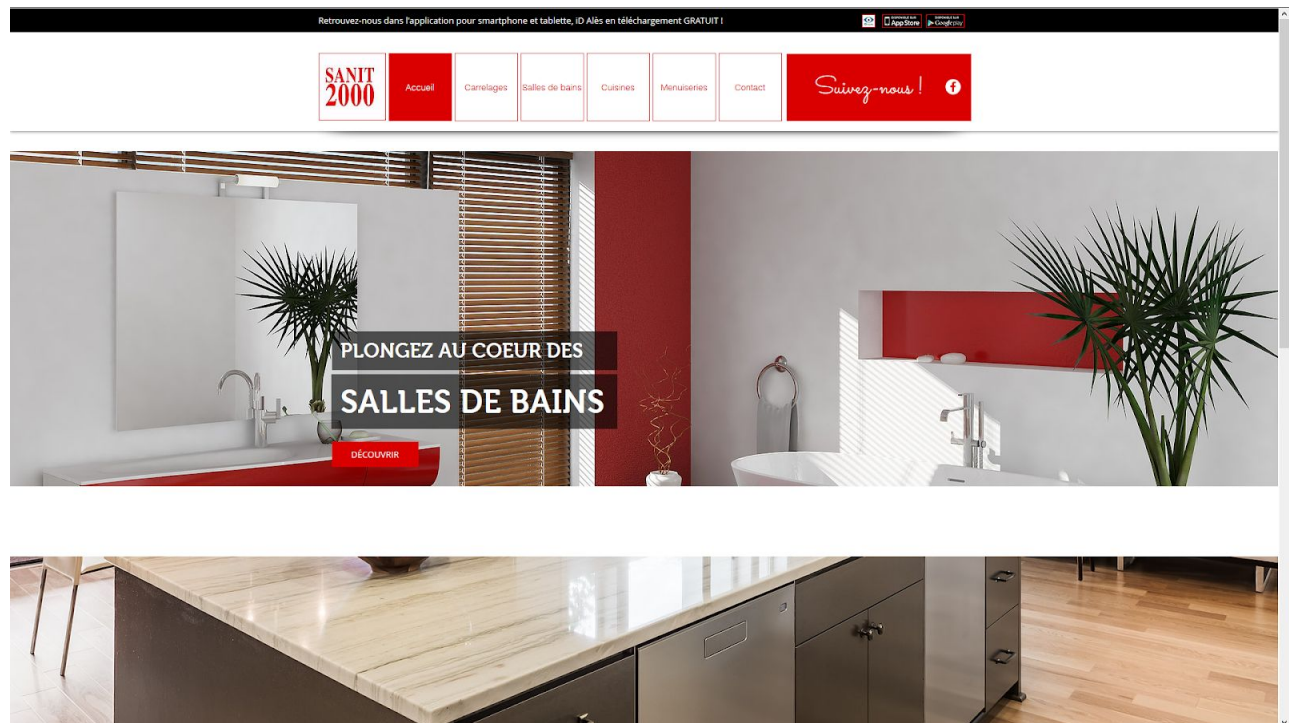
Alès communication tend à s'étendre à plus grande échelle, en élargissant ses domaines d'activités. Le développement d'applications mobiles, notamment de progressive web app permet de cibler un nouveau public.

*Voici quelques exemples de sites développés par l'agence :*

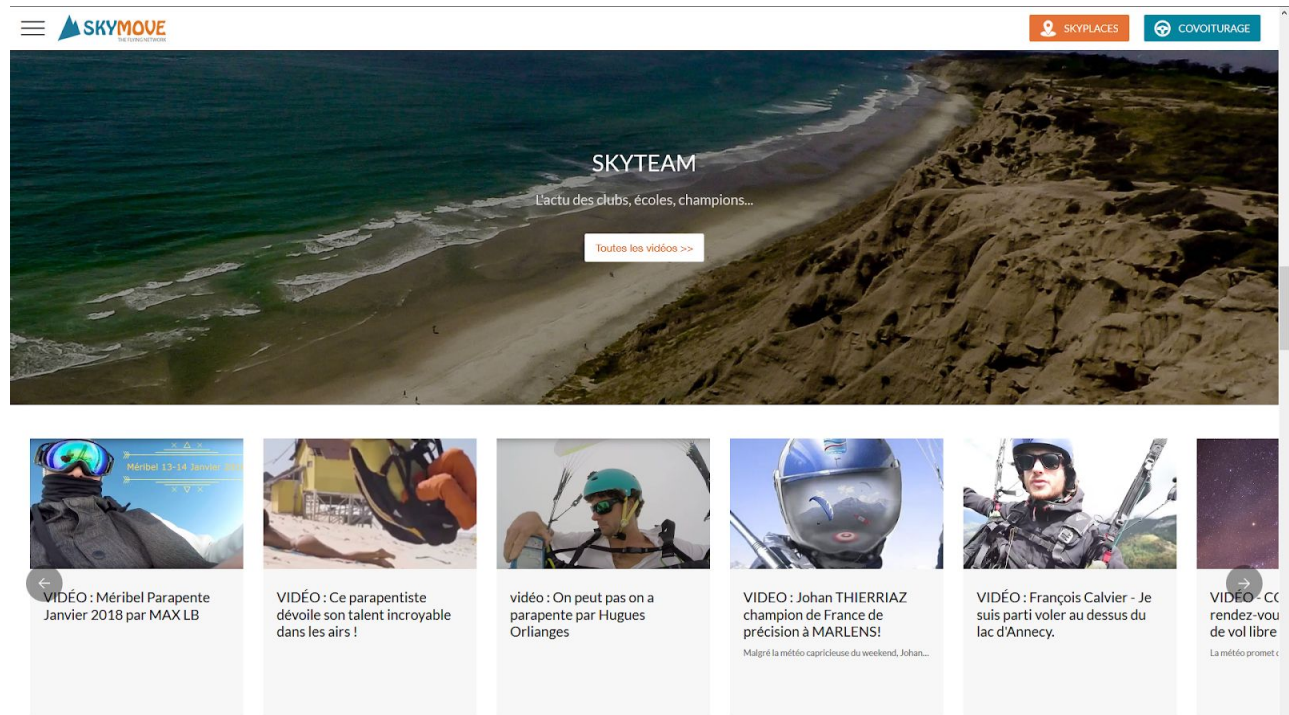
#### Les fous Chantants



## SANIT2000



## Skymove



## 4 - Liste des compétences couvertes

Les compétences du référentiel utilisées dans le projet pour le titre professionnel

Sont les suivantes :

- Maquetter une application
- Mettre en place une base de donnée
- Développer une interface utilisateur
- Développer des composants d'accès aux données
- Développer une application simple de mobilité numérique
- Développer des pages web en lien avec une base de donnée
- Utiliser l'anglais dans son activité professionnelle en informatique



## 5 - Cahier des charges

Pour la création de notre covoiturage, je n'ai pas reçu de cahier des charges spécifique. Un entretien avec le client m'a permis de poser le cadre du projet.

L'application se devait de respecter certains critères :

### **Fonctionnalités :**

- L'utilisateur doit pouvoir proposer un covoiturage de deux manières distinctes :
  - => Site d'atterrissage vers site de décollage
  - => Ville française => site de décollage.
- Le formulaire de soumission doit contenir les infos nécessaires à la prise de contact et au détail du covoiturage
- L'utilisateur de l'application skymove doit pouvoir rapidement et n'importe où trouver un covoiturage vers un site de décollage à proximité.
- La recherche par géolocalisation devrait faire appel à une carte interactive, pour repérer rapidement les covoiturations à proximité.
- L'utilisateur doit pouvoir accéder rapidement aux détails du covoiturage et pouvoir contacter le conducteur.

### **Interface :**

- L'interface doit être simple, ergonomique et respecter la charte graphique de l'application.
- Elle doit être responsive sur tous les devices, PC comme mobiles
- Prévoir Carte interactive pour la recherche géolocalisée
- Privilégier le scroll vertical

### **Recommandations :**

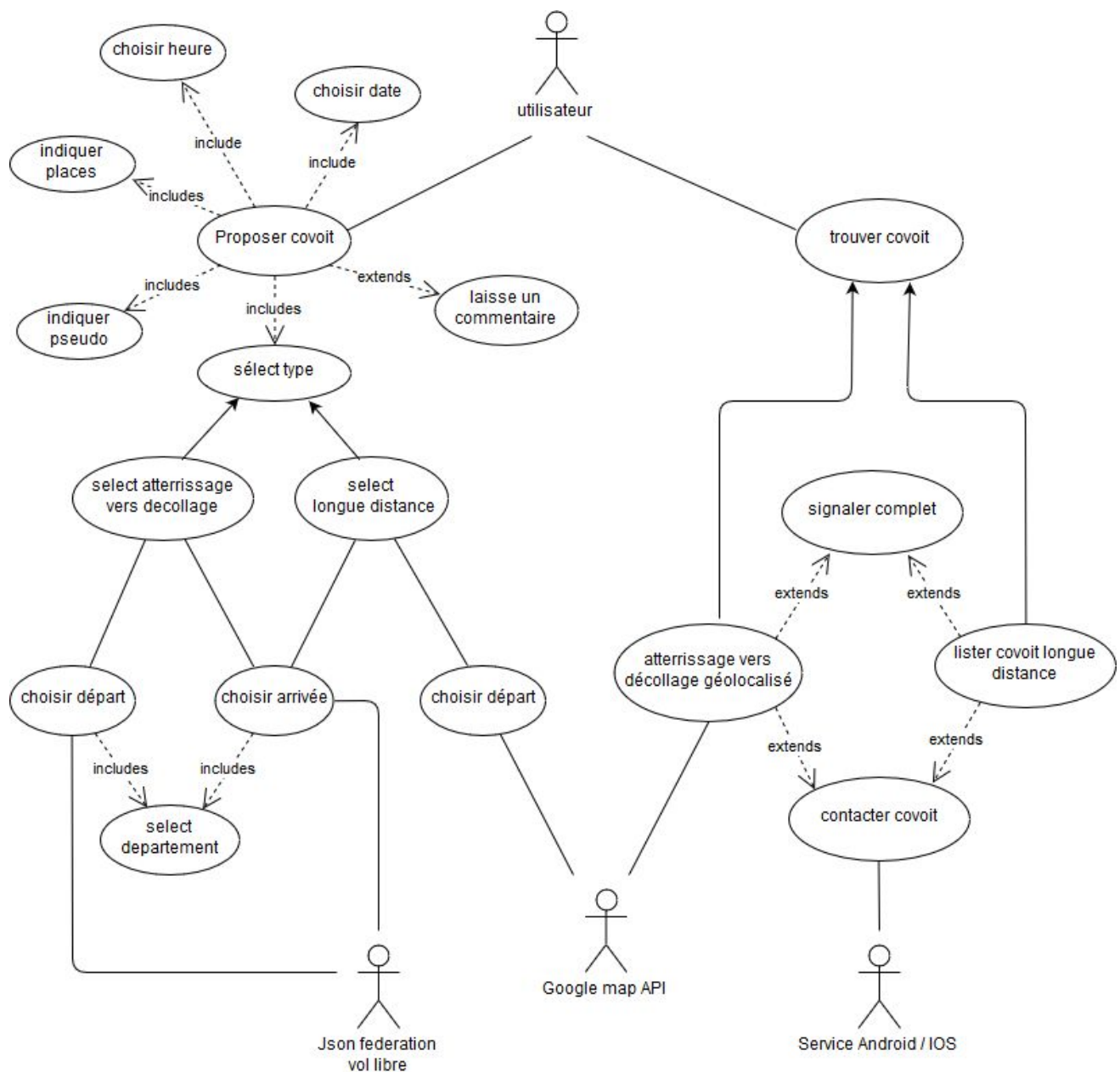
- Maximiser les performances, limiter les chargements.
- Rester simple. Ne pas embrouiller l'utilisateur avec des fonctionnalités compliquées ou une interface trop lourde.
- Mettre en avant l'esprit communautaire de l'application, privilégier le contact direct.



## 6 - Spécifications fonctionnelles et techniques

### Spécifications fonctionnelles:

Ci dessous, le diagramme UML de cas d'utilisation final. Il prend en charge le parcours utilisateur dès son entrée dans le plugin. Nous avons ici quatres acteurs, l'utilisateur ainsi que les différents services nécessaires à la création de l'application.



## Spécifications techniques :

Le projet aura nécessité pratiquement toutes les compétences acquises durant mon parcours de formation dont voici la liste.

### Technologies utilisées :

#### Gestion de projet

- Github
- PhpStorm
- PhpMyAdmin
- OVH

#### Frontend

- CMS Goodbarber
- HTML
- CSS
- Javascript
- JQuery

#### Backend

- PHP
- SQL
- XML
- JSON
- Javascript
- JQuery

#### Pour le maquettage

- GIMP
- Pencil
- Draw.io

#### Veille technologique

- Documentation officielle ( Google-map-api, Mozilla developpeur, Spidifier, Goodbarber...)
- Github ( Coda-Wikicoda repository de la formation et autres )
- et autres Forum de développement ( Stackoverflow, Alsacréation... )

# 7 - Réalisation

## 7.0 Planification des tâches :

### Semaine 1 :

Découverte et prise en main du CMS Goodbarber

### Semaine 2 :

Définition du projet

Maquettage

Parcours utilisateur

### Semaine 3 :

Création vue proposer

Création formulaire atterro vers déco & longue distance

Récupération et traitement des données JSON

### Semaine 4 - 5 :

Création de la BDD

Ecriture et lecture de la table, assurer l'intégrité des données

Créations vue trouver

Découverte et prise en main de l'API Google map Javascript

### Semaine 6 - 9 :

Récupération de la BDD dans l'API, créations de marqueurs et events

Affichage liste longue distance

Développement des fonctionnalités contacter / signaler

Développement de la fonction suppression automatique du contenu de la table selon la date.

Mise en place et validation du Style final de l'application.

### Semaine 10 :

Intégration du plugin dans l'application Skymove en utilisant le système d'iframe.

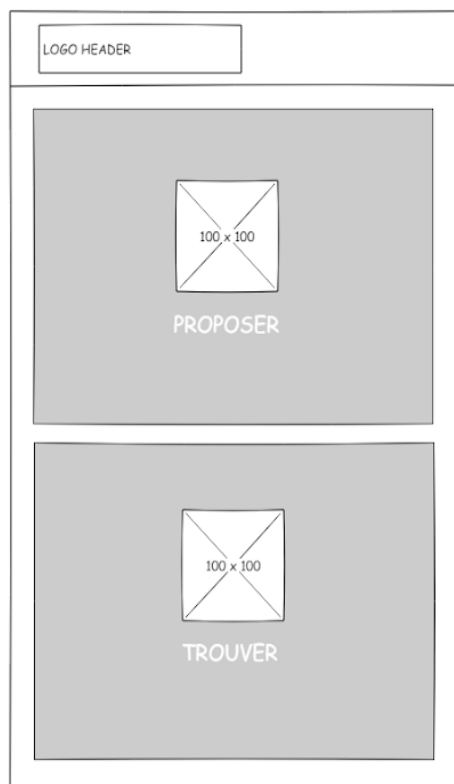
Mise en production du projet

Test et retour utilisateurs

## 7.1 Maquettage

Pour le Développement du plugin de covoiturage, j'ai réalisé les maquettes des différentes vues mobiles et les ai soumises au client. L'ensemble des vues de l'application sont disponibles ci dessous.

*Menu principal et formulaire de soumission :*



Logo Header

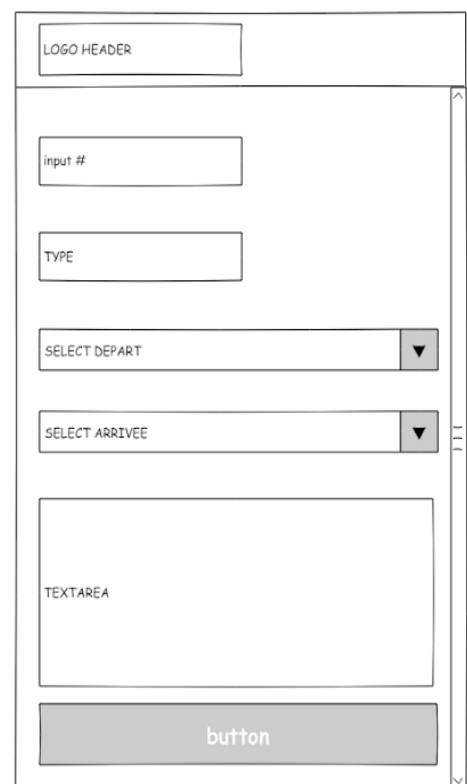
100 x 100

PROPOSER

100 x 100

TROUVER

The mockup shows a mobile app interface with a header containing a 'Logo Header'. Below the header, there are two large rectangular buttons. The top button is labeled 'PROPOSER' and contains a placeholder image labeled '100 x 100'. The bottom button is labeled 'TROUVER' and also contains a placeholder image labeled '100 x 100'.



Logo Header

input #

TYPE

SELECT DEPART ▼

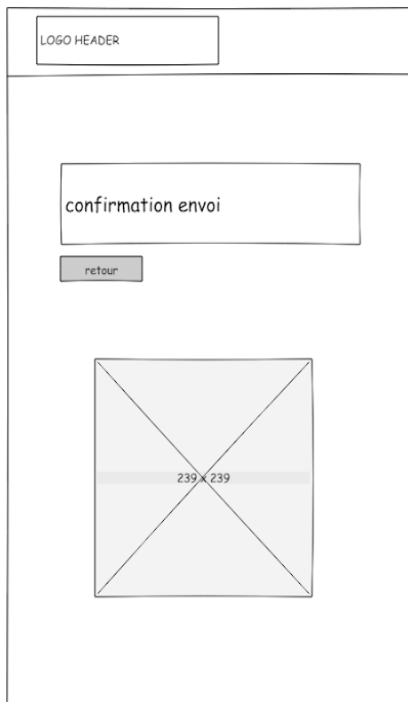
SELECT ARRIVEE ▼

TEXTAREA

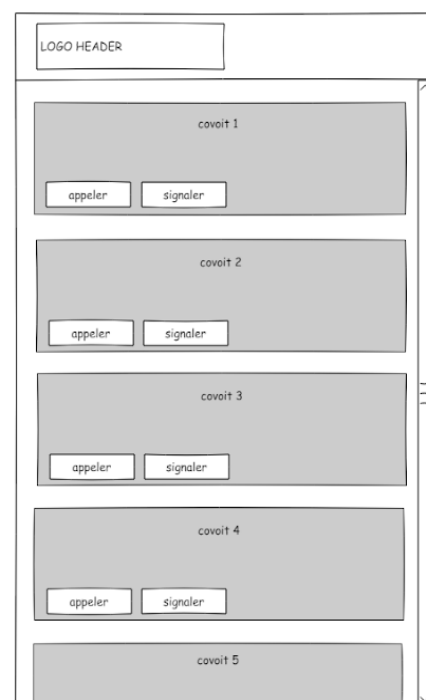
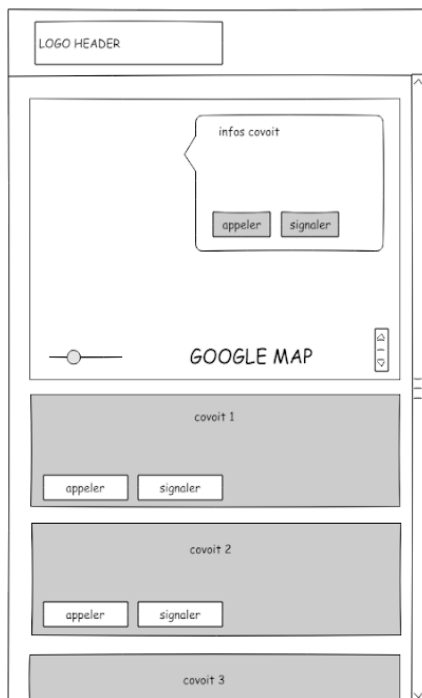
button

The mockup shows a mobile app interface with a header containing a 'Logo Header'. Below the header, there is a form with several input fields: 'input #', 'TYPE', 'SELECT DEPART' (with a dropdown arrow), and 'SELECT ARRIVEE' (with a dropdown arrow). Below these fields is a large 'TEXTAREA' for text input. At the bottom of the form is a wide 'button'.

## Confirmation d'envoi et "menu trouver :"

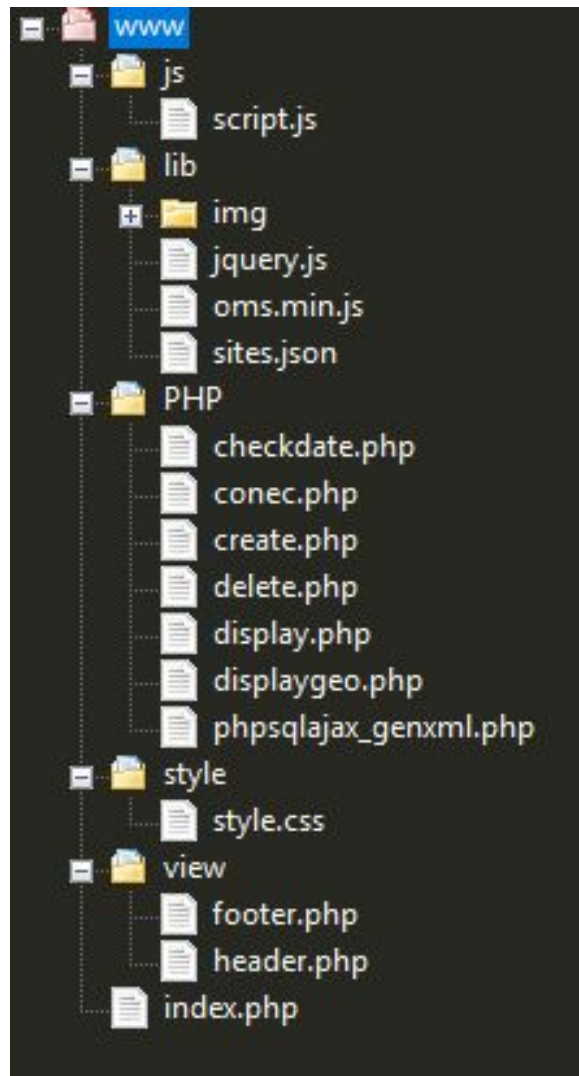


## "Atterro vers déco géolocalisé" et "lister longue distance"



## 7.2 Structure du projet :

J'ai adapté le modèle MVC pour ma structure du projet. Voici l'arborescence :



L'index est à la racine du projet.

Les bibliothèques sont présentes dans le dossier lib, ainsi que le dossier img.

Tous les fichiers de traitement et de connexion à la base de données sont stockés dans le dossier PHP.

## 7.3 Code HTML et génération des éléments du DOM

header.php

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="initial-scale=1.0,
user-scalable=no">
  <link href="https://fonts.googleapis.com/css?family=Lato"
rel="stylesheet">
  <link rel="stylesheet" href="../style/style.css">
  <title>covoit</title>
</head>
<body>
```

footer.php

```
<script src="../lib/jquery.js"></script>
<script type="text/javascript" src="../js/script.js"></script>

<script async defer
src="https://maps.googleapis.com/maps/api/js?v=3&key=*****
*****&libraries=places&sensor=false">
</script>
<script async defer src="../lib/oms.min.js"></script>

</body>
</html>
```

Les fichiers javascript sont appelés dans le footer pour des questions de performances, les scripts s'exécutent une fois le DOM chargé correctement.



Index.php :

```
<?php include(' ./view/header.php') ; ?>

<div class="content">
    <div class="menu-covoit">

<!-- BLOC PROPOSER-->
        <div class="choix proposer">
            <div class="choix-content">
                
                <h3>Proposer</h3>
            </div>
        </div>

<!-- BLOC TROUVER-->
        <div class="choix trouver">
            <div class="choix-content">
                
                <h3>Trouver</h3>
            </div>
        </div>
    </div>
</div>
<?php include(' ./view/footer.php') ; ?>
```

Ayant opté pour un site “one page”, le reste des éléments est généré dans la div “content” en Javascript/JQuery en fonction des événements.

Vous trouverez ci dessous, en détail, les triggers et les fonctions qui génèrent le DOM pour toutes les vues du projet.

### Menu principal :

Selon le choix de l'utilisateur, je génère le formulaire ou le “sous menu Trouver”

```
$( '.proposer' ).click(function() {
    displayMenu();
});
```

```

$('.trouver').click(function(){
    displayMenu2();
});

```

```

function displayMenu(){
    $('.content').html(
        '<div class="proposer-content">'+
        // =====ENSEMBLE FORMULAIRE===== //
        '<div class="create-form">'+
        '<form id="create" method="post"
action=" ../PHP/create.php">'+
        // =====BLOC DETAILS=====
        '<div class="form-details">'+
        '<label for="date">Date :</label><br>'+
        '<input id="date" name="date" type="date"
        required="required"><br>'+
        '<label for="time">Heure :</label><br>'+
        '<input id="time" name="time" type="time"><br>'+
        '<label for="nbdispo">Places disponibles :</label><br>'+
        '<input id="nbdispo" name="nbdispo" type="number" min="1"
        max="99" required="required"><br>'+
        '<label for="type">Type de covoiturage :</label><br>'+
        '<select id="type" name="type">'+
        '<option selected>Type</option>'+
        '<option value="atterro_vers_deco">Atterro vers
        Déco</option>'+
        '<option value="longue_distance">Longue
        distance</option>'+
        '</select><br>'+
        '</div>'+
        // =====BLOC LIEUX=====
        '<div class="form-where"></div>'+
        '<div class="form-pos"></div>'+
        // =====BLOC USERS=====
        '<div class="form-users">'+
        '<label for="nom">Pseudo :</label><br>'+
        '<input pattern="^[ _a-zA-Z0-9.]+$" id="nom" name="nom"
        type="text" required="required"><br>'+
        '<label for="tel">Tel :</label><br>'+
        '<input pattern="[0-9]{10}" id="tel" name="tel"

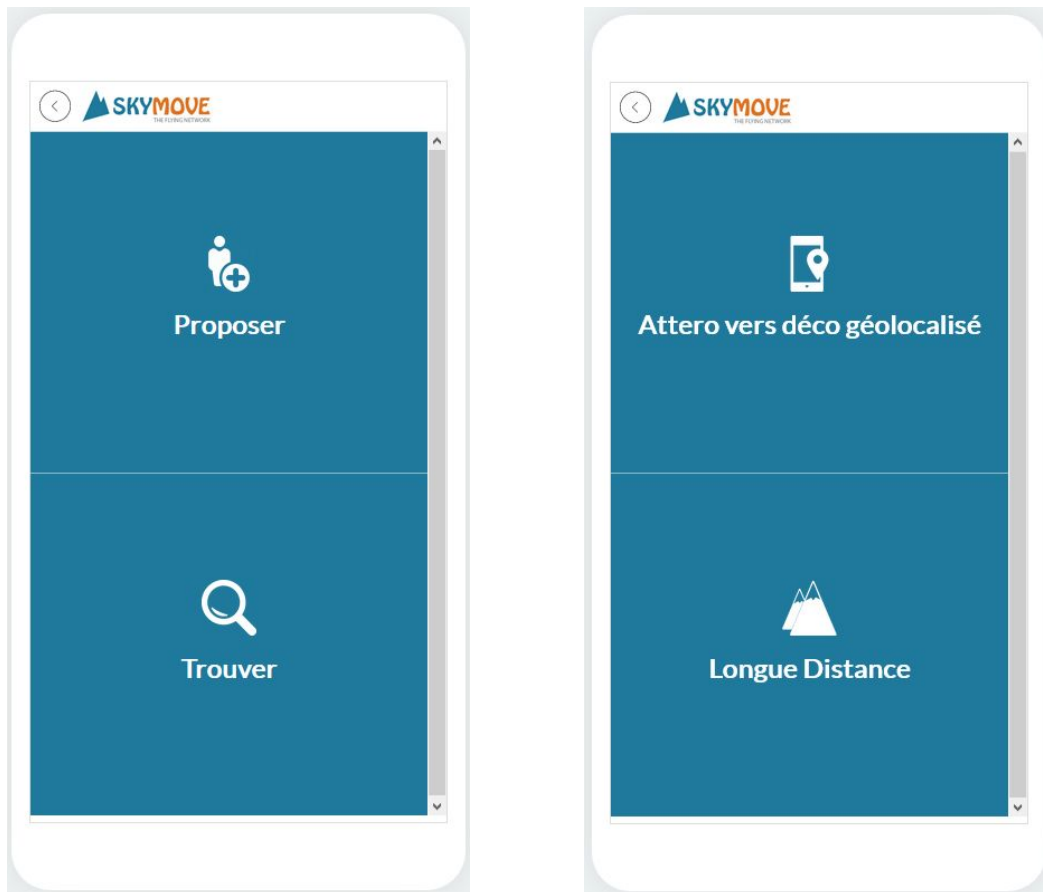
```

```

        type="text"
        placeholder="ex: 0611223344"required="required"><br>' +
'<label for="com">Commentaire :</label><br>' +
'<textarea id="com" name="com" placeholder="ex:
    récupération
    véhicule..."></textarea>' +
'</div>' +
'<input id="validate-form" type="submit"
    value="Proposer">' +
'</form>' +
'</div>' +
'</div>'
    );
}
// =====AFFICHAGE SOUS MENU TROUVER=====
function displayMenu2() {
    $(''.content').html(
        '<div class="trouver-content">' +
        '<div class="menu-covoit">' +
        '<div class="choix geo">' +
        '<div class="choix-content">' +
        '' +
        '<h3>Attero vers déco géolocalisé</h3>' +
        '</div></div>' +
        '<div class="choix list">' +
        '<div class="choix-content">' +
        '' +
        '<h3>Longue Distance</h3>' +
        '</div>'</div></div>')

```

### Menu principal et sous menu trouver :



#### **Formulaire :**

Le formulaire a lui aussi ces propres événements et modification du DOM. Les champs du formulaire varient en fonction du type de covoiturage choisi : atterro vers déco ou longue distance.

- Le Type “Atterro vers déco” :

Il va nécessiter un premier filtre par département, qui filtrera les sites de décollage et d'atterrissage. L'utilisateur pourra donc choisir dans cette liste le lieu de départ et d'arrivée.

- Le type “Longue distance” :

Ici l'utilisateur devra indiquer sa ville de départ, qui sera localisée et géocodée, et devra sélectionner un département pour choisir un site de décollage à rejoindre.

Voici le code pour la génération du formulaire :

script.js

```
$('.content').delegate('#type', 'change', function () {
    selectType();
});

// -FUNCTION SELECTION TYPE FORM--//

function selectType() {
    var type = $("#type").val();
    if (type === "atterro_vers_deco") {
        displayform1();
    }
    else if (type === "longue_distance") {
        displayform2();
    }
}

// -FUNCTION CREATION FORMULAIRE ATTERRO -> DECO --//

function displayform1() {
    $(".form-where").html(
        '<p>Départ :</p>\n' +
        '<select class="departement" id="departement-depart" ' +
        'name="departement">\n' +
        '<option selected>Département</option>\n' +
        '</select><br>\n' +
        '<select id="atterro" name="atterro">\n' +
        '<option selected>Attero</option>\n' +
        '</select><br>\n' +
        '\n' +
        '<p>Arrivée :</p>\n' +
        '<select id="deco" name="deco">\n' +
        '<option selected>Deco</option>\n' +
        '</select><br><br>'
    );
    checkdep();
}
```

```
//---FUNCTION CREATION FORMULAIRE LONGUE DISTANCE-----

function displayform2() {
    $('form-where').html(
        '<label for="autocomplete-google-map">Départ :
        </label><br>\n' +
        '<input type="text" id="autocomplete-google-map"
        class="controls" name="autocomplete-google-map"
        placeholder="search"><br>\n' +
        '<p>Arrivée :</p>\n' +
        '<select class="departement" id="departement-arrive"
        name="departement">\n' +
        '<option selected>Département</option>\n' +
        '</select><br>\n' +
        '<select id="deco" name="deco">\n' +
        '<option selected>Déco</option>\n' +
        '</select><br><br>'
    );
    checkdep();

    $('#autocomplete-google-map').focus(function () {
        autocomplete();
    })
}
}
```

Voici les différentes vues du formulaire :

SKYMOVE  
THE FLYING NETWORK

Date :  
jj / mm / aaaa

Heure :  
-- : --

Places disponibles :  
[dropdown]

Type de covoiturage :  
Type [dropdown]

Pseudo :  
[input]

Tel :  
ex: 0611223344 [input]

SKYMOVE  
THE FLYING NETWORK

Heure :  
-- : --

Places disponibles :  
[dropdown]

Type de covoiturage :  
Attero vers Déco [dropdown]

Départ :  
Departement [dropdown]  
Attero [dropdown]

Arrivée :  
Déco [dropdown]

Pseudo :  
[input]

SKYMOVE  
THE FLYING NETWORK

Type de covoiturage :  
Longue distance [dropdown]

Départ :  
nimes [input]  
Nîmes France [location pin]  
powered by Google

Departement [dropdown]  
Déco [dropdown]

Pseudo :  
[input]

Tel :  
ex: 0611223344 [input]

Commentaire :  
[input]





## 7.4 - Exploitation du JSON et implémentation du formulaire

J'ai récupéré le JSON officiel de la fédération française de vol libre, qui contient toutes les informations nécessaires à la création du formulaire.

Voici une partie du fichier, que j'ai revu et corrigé pour améliorer la lisibilité des données.

sites.json

```
[
  {
    "suid": "13180",
    "id": "73050",
    "numero": "73A050B",
    "nom": "SAINT FRANCOIS LONGCHAMP",
    "sous_nom": "EPIERRE - LES REMBLAIS",
    "cp": "73220",
    "ville": "EPIERRE",
    "site_type": "vol",
    "site_sous_type": "Atterrissage",
    "pratiques": "parapente;delta;0;0;0",
    "lat": "45.4462",
    "lon": "6.2905",
    "alt": "375",
    "acces": "En venant de Chambéry, sortie Epierre, traversée du village direction st jean de Maurienne , se garer de préférence au parking du gymnase sur la droite à la sortie de Epierre .\r\nl'atterrissage est situé sur la zone des Remblais 200m plus loin .",
    "trajet_parcking": "5",
    "trajet_attero_deco": "0",
    "handi": "1",
    "orientation": "",
    "vent_favo": "",
    "vent_defavo": "",
```

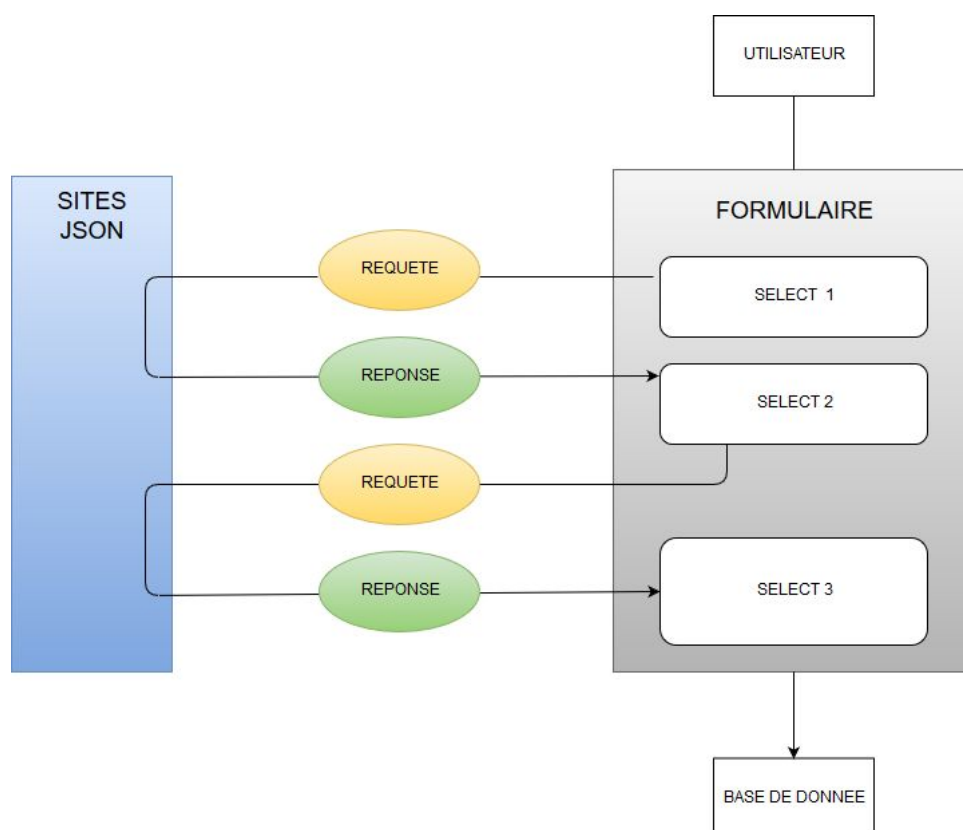
```

"conditions_ideales": "atterrissage du matin afin d'éviter les
brises qui peuvent être fortes au delà de 11h00 selon la saison
.",
"balise": "",
"webcam": "",
"signalétique": "0",
"description": "Atterrissage servant aux vols Randos depuis les
massifs de La Lauzière et envers de Belledonne",
"restrictions": "Respecter les clôtures et interdiction de pose
dans les parcelles si présence de Moutons\r\n",
"reg_aerienne": "",
"dangers": "Présence d'une ligne Haute tension 20 000 Volts au
Sud du terrain et d'une ligne longeant la route\r\nvoie ferrée e
autoroute à l'ouest du terrain . ",
"numero_cross": null,
"date_modification": "2015-03-12 14:57:52"
},

```

### Utilisation:

Ce fichier sera consulté à de nombreuses reprises, pour implémenter les champs de notre formulaire. Le Diagramme suivant en résume le fonctionnement :



### En pratique :

La sélection des données nécessaires à la soumission du formulaire passe par trois fonctions qui vont chacune parcourir le fichier JSON pour récupérer les informations demandés:

- checkdep() va récupérer les départements
- trisites() va récupérer et trier les sites de vol
- checkpos() va récupérer les coordonnées GPS du site sélectionné.

### script.js

```
//-----FUNCTION TRI DEPARTEMENT -----//

function checkdep(){
    //on récupère le JSON
    $.getJSON('../lib/sites.json', function (data) {
        //on initialise nos tableaux
        var departementTab = [];
        var uniqdepartementTab = [];

        //on parcourt le JSON
        $.each(data, function(index, x){
            //on push dans le tableau les 2 premiers
            //caractères de chaque code postal
            departementTab.push(x.cp.substr(0,2));
        });

        // On boucle sur le tableau departement
        $.each(departementTab, function(i, el){
            //pour chaque département
            //on push une valeur unique dans uniqdepartementTab
            if($.inArray(el, uniqdepartementTab) === -1) {
                uniqdepartementTab.push(el);
            }
        });

        var departement = "";
        //on appelle la fonction de tri alphabétique
        uniqdepartementTab.localeSort();

        //on boucle sur notre tableau  uniqdepartementTab
```

```

for (var i = 0; i < uniqdepartementTab.length; i++) {
    departement += '<option value="' +
        uniqdepartementTab[i] + '>' +
        uniqdepartementTab[i] + '</option>'
    }
    //on incrémente le select du formulaire
    $(".departement").html(
        '<optionselected>Departement</option>' + departement
    );
});

```

Le contenu du menu déroulant est donc généré au chargement du formulaire par notre script.

Quand l'utilisateur sélectionne un département dans cette liste, il déclenche la fonction trisites(), qui récupère tous les sites de vol du département et les sépare dans deux tableaux distincts, un pour les sites d'atterrissage et l'autre pour les sites de décollage. Ces tableaux alimentent les menus déroulants du formulaire.

```

$(".departement").on('change', function(){
    // var attero = "";
    // var deco = "";
    trisites();
});

```

Cette fonction suit la même procédure que la précédente.

```

//-----TRI SITES PAR DEPARTEMENT-----//

function trisites(){

    //on récupère la valeur du select département
    var currentdep=$('.departement').val();

    //on récupère le JSON
    $.getJSON('../lib/sites.json', function (data) {
        //on initialise nos tableaux
        var atterrotab = [];
        var decotab = [];
    });
}

```

```

//on parcourt le JSON
$.each(data, function (index, x) {

//si le departement correspond et le type est
//atterrissage
if (x.site_sous_type === "Atterrissage" && currentdep ==
x.cp.substr(0,2)) {
//on push les résultats dans le tableau
    atterrotab.push(x.nom + ' - ' + x.sous_nom);
}
//si le departement correspond et le type est décollage
else if (x.site_sous_type === "Décollage" && currentdep
== x.cp.substr(0,2)) {
//on push les résultats dans le tableau
    decotab.push(x.nom + ' - ' + x.sous_nom);
}
});
var atterro = "";
var deco = "";
//on trie nos tableau par ordre alphabétique
atterrotab.localeSort();
decotab.localeSort();

//on boucle sur le tableau atterrotab
for (var j = 0; j < atterrotab.length; j++) {
    atterro += '<option value="' +
    atterrotab[j]+'">' +
    atterrotab[j] + '</option>'
}
//on incrémente le select atterro
$("#atterro").html('<option selected>Attero</option>'+
atterro);
//on boucle sur le tableau decotab
for (var k = 0; k < decotab.length; k++) {
    deco += '<option value="' + decotab[k] + '">' +
    decotab[k] + '</option>'
}
//on incrémente le select deco
$("#deco").html('<option selected>Déco</option>'+ deco);
});
}

```

Le choix d'un site de départ déclenche la fonction checkpos() qui va aller chercher dans le fichier JSON les coordonnées GPS correspondantes à la sélection actuelle. Ces coordonnées sont ensuite écrites dans un <input> généré et caché dans le formulaire.

```
$('#atterro').on('change', function () {
    checkpos();
});

// =====FUNCTION RECUPERER POS=====

function checkpos(){
    //on récupère la valeur du select
    var depart= $('#atterro').val();
    var lat="";
    var lng="";
    //on récupère le JSON
    $.getJSON('../lib/sites.json', function (data) {
        //on boucle dessus
        $.each(data, function (index, x) {
            //si le nom complet correspond à notre sélection
            if (x.nom + ' - ' + x.sous_nom === depart) {
                //on récupère les coordonnées GPS
                lat=(x.lat);
                lng=(x.lon);
            }
        });

        //et on les injecte dans un input type hidden du form
        $('#form-pos').html(
            '<input id="lat" name="lat" type="hidden" '
            'value="'+lat+' ">\n' +
            '<input id="lng" name="lng" type="hidden" '
            'value="'+lng+' ">'
        )
    });
}
```

### Place Autocomplete:

Pour les longues distances, le focus d'un utilisateur sur le champ de saisie de la ville de départ va déclencher la fonction autocomplete().

Cette fonction active le service Place Autocomplete de l'API Google map.

script.js

```
// =====AUTOCOMPLETION=====

function autocomplete() {
    //on cible le champ de saisie
    var input =
document.getElementById('autocomplete-google-map');
    //on définit les options : ville françaises
var options = {
    types: ['(cities)'],
    componentRestrictions: {country: 'fr'}
};
    //on instancie le service Place Autocomplete
var autocomplete = new google.maps.places.Autocomplete(input,
options);
    //on ajoute un listener, quand on change de ville
autocomplete.addListener('place_changed', function () {
    //getplace retourne les coordonnées GPS de la ville
var place = autocomplete.getPlace();
var lat = place.geometry.location.lat();
var lng = place.geometry.location.lng();
    //pour finir on injecte lat lng dans des input hidden
$('.form-pos').html(
    '<input id="lat" name="lat" type="hidden"
    value="' + lat + '">\n' +
    '<input id="lng" name="lng" type="hidden"
    value="' + lng + '">'
    )
});
}
```

Nous avons maintenant toutes les informations disponibles pour soumettre notre formulaire. Nous allons maintenant créer la BDD, afin d'enregistrer les données.



## 7.5 - Création de la BDD et écriture des données

Le projet étant hébergé par OVH, j'ai opté pour l'utilisation de PhpMyAdmin pour administrer ma base de données.

Le covoiturage ne nécessitant qu'une seule table, j'ai décidé de la concevoir directement à la main au lieu de passer par un modèle logique.

Requête de création de la table 'covoit':

```
CREATE TABLE `covoit` (  
  `id` int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
  `date` date NOT NULL,  
  `time` time NOT NULL,  
  `nbdispo` int(11) NOT NULL,  
  `type` varchar(255) NOT NULL,  
  `nom` varchar(255) NOT NULL,  
  `tel` varchar(255) NOT NULL,  
  `com` text,  
  `lat` float NOT NULL,  
  `lng` float NOT NULL,  
  `depart` varchar(255) NOT NULL,  
  `arrivee` varchar(255) NOT NULL  
);
```

## Structure dans PhpMyAdmin:

		STRUCTURE DE TABLE		VUE RELATIONNELLE					
	#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
<input type="checkbox"/>	1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT
<input type="checkbox"/>	2	date	date			Non	Aucun(e)		
<input type="checkbox"/>	3	time	time			Non	Aucun(e)		
<input type="checkbox"/>	4	nbdispo	int(11)			Non	Aucun(e)		
<input type="checkbox"/>	5	type	varchar(255)	latin1_swedish_ci		Non	Aucun(e)		
<input type="checkbox"/>	6	nom	varchar(255)	latin1_swedish_ci		Non	Aucun(e)		
<input type="checkbox"/>	7	tel	varchar(255)	latin1_swedish_ci		Non	Aucun(e)		
<input type="checkbox"/>	8	com	text	latin1_swedish_ci		Oui	NULL		
<input type="checkbox"/>	9	lat	float			Non	Aucun(e)		
<input type="checkbox"/>	10	lng	float			Non	Aucun(e)		
<input type="checkbox"/>	11	depart	varchar(255)	latin1_swedish_ci		Non	Aucun(e)		
<input type="checkbox"/>	12	arrivee	varchar(255)	latin1_swedish_ci		Non	Aucun(e)		
	<input type="checkbox"/>	Tout cocher	Avec la sélection :		Parcourir	Modifier	Supprimer	Primaire	Unique
	Imprimer			Suggérer des optimisations de structure			Déplacer des colonnes		Améliorer la structure de la table
	Ajouter	<input type="text" value="1"/>	colonne(s)		<input type="text" value="après arrivee"/>		Exécuter		

Nous allons maintenant soumettre les données du formulaire à notre base de données. Pour se faire, j'ai opté pour l'utilisation des requêtes AJAX en JQuery, pour appeler les fichiers de traitement de la Base de données en PHP.

Il y a plusieurs avantages à utiliser cette technologie :

- pouvoir utiliser les callback,
- respecter la structure single page
- gagner en performance.

### En pratique :

Coté javascript, voici le script de soumission du formulaire.

On appelle la fonction send() en faisant passer le formulaire en paramètre

script.js

```
$( '.content' ).delegate( '#create', 'submit', function (e) {
    // On empêche le navigateur de soumettre le formulaire
    e.preventDefault();
    var $form = $(this);
    send($form);
});
// =====FUNCTION SOUMISSION FORMULAIRE=====
function send($form) {
    $.ajax({
        //paramètres de la requête AJAX, action et method form
        url: $form.attr('action'),
        type: $form.attr('method'),
        data: $form.serialize(),
        //callback affichage message success
        success: function () {
            $(".content").html(
                '<div class="callbacklogo"></div>'+
                '<h3 class="return-msg return-msg-success">
                Proposition de covoiturage envoyée</h3>'
            );
            $(".content").css({backgroundColor: "white"});
        },
        //callback affichage message error
        error: function () {
            $(".content").css({backgroundColor: "white"});
            $(".content").html(
                '<div class="callbacklogo"></div>'+
                '<h3 class="return-msg return-msg-ko">Erreur
                lors de l\'appel de la requête.
                Veuillez réessayer plus tard
                ou contactez votre administrateur.</h3>'
            );
        }
    });
}
```

Voici la connexion à la base de donnée, qui sera en entête de tous les fichiers de traitement de la base de données.

### Conec.php

```
<?php
//----- Connexion à la base de donnée-----//
try{
    $pdo = new
PDO('mysql:host=skymovecukskyco.mysql.db;dbname=***','***','***')
    $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_WARNING);

    $pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,PDO::FETCH_OBJ);
}
catch(PDOException $e){
    echo 'Connexion impossible';
}
```

La requête SQL d'insertion dans la base de données :

### create.php

```
<?php
include('conec.php');
//on récupère les valeurs des champs du form
$date=$_POST['date'];
$time=$_POST['time'];
$nbdispo=$_POST['nbdispo'];
$type=$_POST['type'];
$nom=$_POST['nom'];
$tel=$_POST['tel'];
$com=$_POST['com'];
$lat=$_POST['lat'];
$lng=$_POST['lng'];
//si l'atterro est précisé le prendre pour départ
if(!empty($_POST['atterro'])){
    $depart=$_POST['atterro'];
}
Else{
    //sinon prendre autocomplete pour départ
    $depart=$_POST['autocomplete-google-map'];
}
$arrivee=$_POST['deco'];
```

```
// -----REQUETE INSERT DANS LA BDD-----//

$req = $pdo->prepare("INSERT INTO covoit (date, time, nbdispo,
type, nom, tel, com, lat, lng, depart, arrivee)
VALUES (:date, :time, :nbdispo, :type, :nom, :tel, :com,
:lat, :lng, :depart, :arrivee)");

$req->execute(array(
    'date' => $date,
    'time' => $time,
    'nbdispo' => $nbdispo,
    'type' => $type,
    'nom' => $nom,
    'tel' => $tel,
    'com' => $com,
    'lat' => $lat,
    'lng' => $lng,
    'depart' => $depart,
    'arrivee' => $arrivee
));
```

Voici un aperçu de la table ‘covoit’ et de son contenu dans l’interface de PhpMyadmin après quelques insert :

id	date	time	nbdispo	type	nom	tel	com	lat	lng	depart	arrivee
79	2018-03-17	12:00:00	2	longue_distance	Lucien	0647663721		44.8378	-0.57918	Bordeaux, France	PYLA et LES GAILLOUNEYS - LES GAILLOUNEYS-PYLA
80	2018-05-09	10:00:00	1	atterro_vers_deco	Nico	0647663721		45.3003	5.91002	ST HILAIRE DU TOUVET -	ST HILAIRE DU TOUVET - Sud
81	2018-06-16	11:00:00	2	atterro_vers_deco	KIKI	0647663721	Vol Ã Bisanne toute la journÃe	45.7214	6.5536	BISANNE - BEAUFORT	BISANNE NORD OUEST - BISANNE NW
82	2018-04-04	13:00:00	3	atterro_vers_deco	Pascal	0647663721	Cherche des pots pour aller voler ;-)	45.7	6.31722	SAMBUY -	SAMBUY - COL DE LA SAMBUY

## 7.6 - Lecture des données et affichage des covoiturages

Pour récupérer et afficher les covoiturages de la base de données, j'utilise encore une requête AJAX. Les deux types de covoiturages doivent être listés séparément.

Script.js

```
$('.content').delegate('.list', 'click', function(){
    $('.content').html('<ul class="trouver-display"></ul>');
    $('.trouver-display').ready(function () {
        list();
    });

$('.trouver-displaygeo').ready(function () {
    listgeo();
});

//-----FUNCTION LIST LOGUE DISTANCE-----
function list() {
    //on appelle le fichier display.php
    $.ajax({
        url: '../PHP/display.php',
        type: 'get',
        //callback
        success: function (data) {
            //on injecte les données reçus dans le DOM
            $(".trouver-display").html(data);
        },
        error: function () {
            $(".trouver-display").html("Erreur de la
            récupération
            des données");
        }
    })
}
```

```
//-----FUNCTION LIST ATTERO DECO-----
function listgeo() {
    $.ajax({
        url: '../PHP/displaygeo.php',
        type: 'get',
        success: function (data) {
            $(".trouver-displaygeo").html(data);
        },
        error: function () {
            $(".trouver-displaygeo").html("Erreur de la
            récupération des données");
        }
    })
}
}
```

#### display.php

```
<?php include('../PHP/conec.php');
//-----RECUPERATION LONGUE DISTANCE-----//
//on en profite pour convertir date et time au format Fr
$req = $pdo->query('SELECT *,
                    DATE_FORMAT(date, \'%d/%m/%Y\')
                    AS datefr,
                    TIME_FORMAT(time, \'%Hh%i\')
                    AS timefr FROM covoit
                    WHERE type = "longue_distance"
                    ORDER BY date DESC');
```

#### Displaygeo.php

```
<?php include('../PHP/conec.php');
//-----RECUPERATION ATTERRO DECO-----//
//on en profite pour convertir date et time au format Fr
$req = $pdo->query('SELECT *,
                    DATE_FORMAT(date, \'%d/%m/%Y\')
                    AS datefr,
                    TIME_FORMAT(time, \'%Hh%i\')
                    AS timefr FROM covoit
                    WHERE type = "atterro_vers_deco"
                    ORDER BY date DESC');
```



La suite du code est commune aux deux fichiers. On Parcours les résultats et on injecte les données dans le DOM

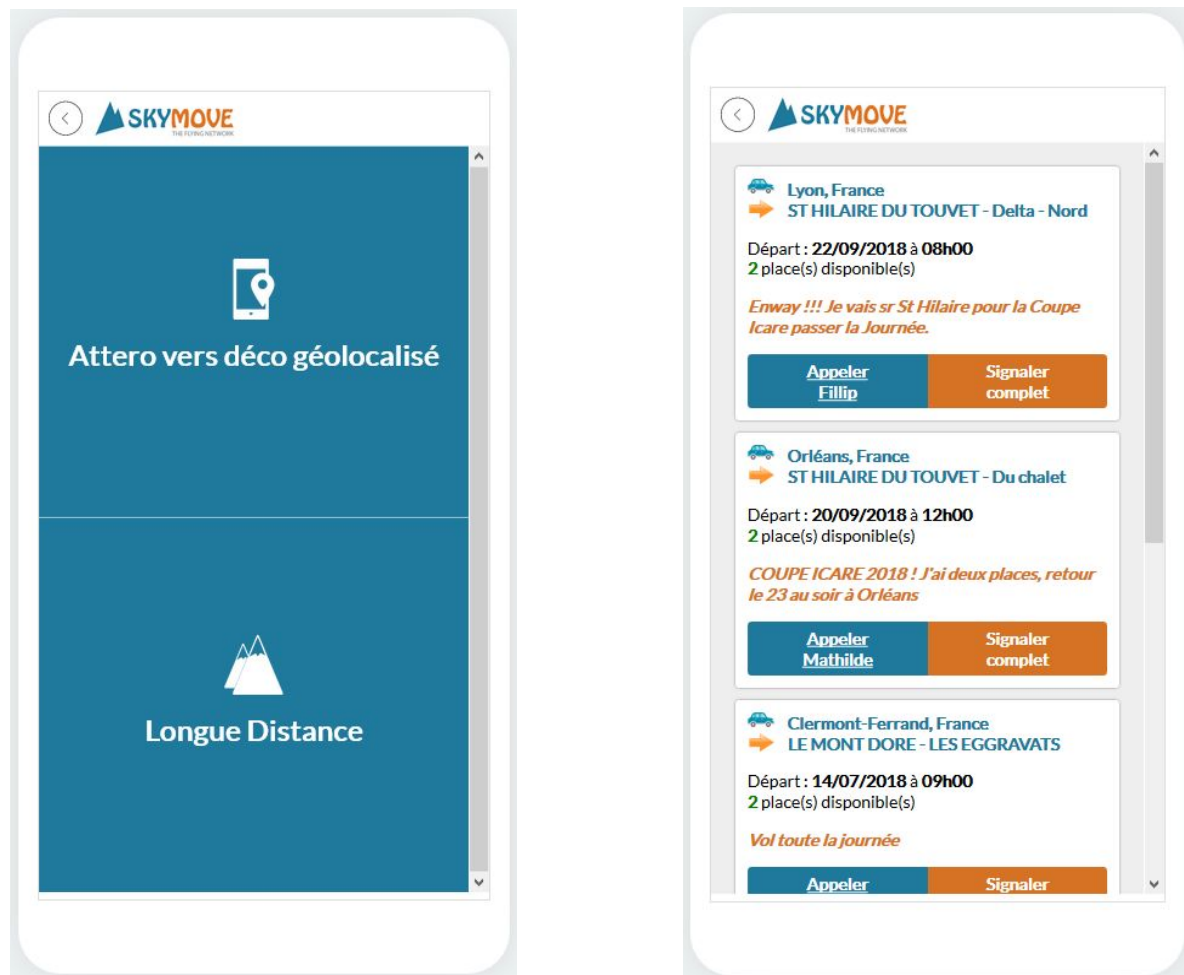
```
while ($data = $req->fetch()) {
    ?>
    <li class="uniqucovoit">
        <div class="details-bloc">
            
            <p class="details details-title">
                <?php echo htmlspecialchars($data->depart); ?></p>
            </div>

            <div class="details-bloc">
                
                <p class="details details-title">
                    <?php echo htmlspecialchars($data->arrivee); ?></p>
                </div>

                <div class="details-content">
                    <p class="details details-infos">Départ : <strong>
                        <?php echo ($data->datefr); ?></strong> à <strong>
                            <?php echo ($data->timefr); ?></strong></p>
                    <p class="details details-nbdispo">
                        <strong class="green"><?php echo ($data->nbdispo); ?>
                        </strong> place(s) disponible(s)</p>
                    <p class="details details-com">
                        <?php echo ($data->com); ?></p>

                    <div class="btn-group">
                        <a class="btn details-tel" href="tel:+33
                            <?php echo substr(($data->tel), -9); ?>">
                            Appeler<br><?php echo ($data->nom); ?></a>
                        <button class="btn details-supr"
                            id="<?php echo ($data->id); ?>">
                            Signaler<br>complet</button>
                    </div>
                </div>
            </li>
        <?php
    }
    ?>
```

## Aperçu de la liste des covoiturages longue distance

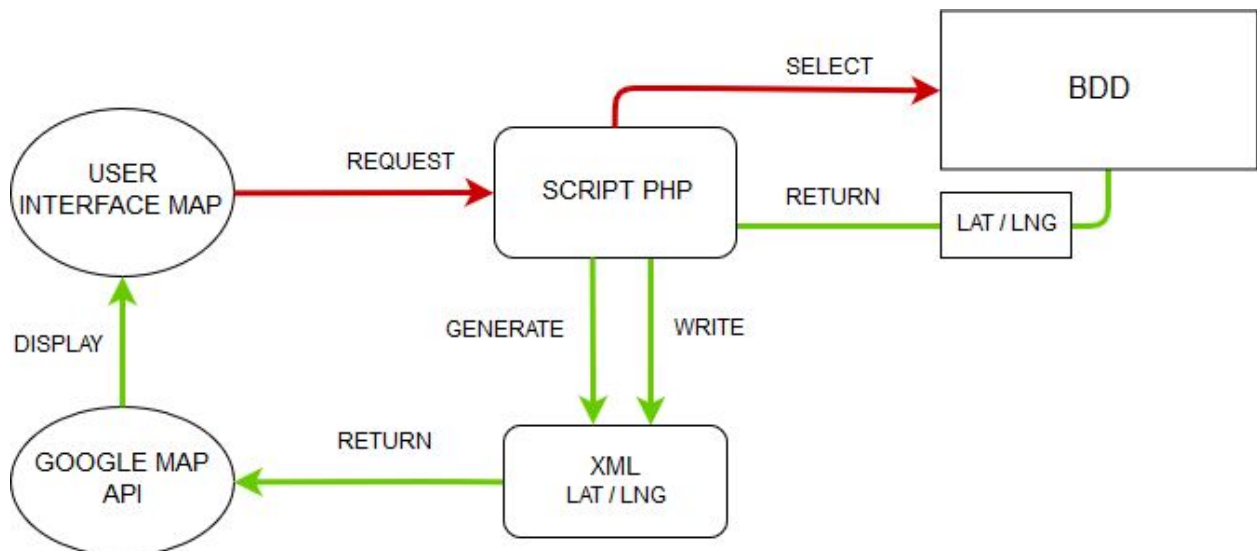


Maintenant que les listes de covoiturages sont fonctionnelles, nous allons nous intéresser au fonctionnement de l'API Google map. Elle permet de créer et d'administrer des cartes interactives, de géolocaliser et géocoder une position, et dans notre cas de figure, de lier une base de donnée à la map.

## 7.7 - mise en place de la map et de la géolocalisation

Nous attaquons la partie la plus complexe du projet. Pour afficher les covoiturages sur la map, il faut créer un fichier de traitement qui va récupérer les données dans la base de données ( lat, lng ) et les écrire dans un fichier XML créé à la volée pour que l'API de Google Map les interprète.

Voici un schéma détaillé du processus :



### En pratique :

Vous trouverez ci dessous le fichier qui récupère les convoiturages de la BDD, créer le XML et retranscrit les données dessus.

#### Phpsqlajax\_genxml.php

```
<?php

require('conec.php');

//on créer le DOM
$dom = new DOMDocument("1.0");
$node = $dom->createElement(utf8_encode("markers"));
$parnode = $dom->appendChild($node);

//REQUETE SELECT ATTERRO VERS DECO
$req = $pdo->query('SELECT *, DATE_FORMAT(date, \'%d/%m/%Y\') AS
datefr, TIME_FORMAT(time, \'%Hh%i\') AS timefr FROM covoit WHERE
type = "atterro_vers_deco" ');

header("Content-type: text/xml");

//on parcours les résultats
while ($row = $req->fetch()){
//pour chaque entrées on créer sur le XML l'attribut
correspondant
$node = $dom->createElement("marker");
$newnode = $parnode->appendChild($node);
$newnode->setAttribute("id", $row->id);
$newnode->setAttribute("nom", $row->nom);
$newnode->setAttribute("tel", $row->tel);
$newnode->setAttribute("com", $row->com);
$newnode->setAttribute("date", $row->datefr);
$newnode->setAttribute("time", $row->timefr);
$newnode->setAttribute("nbdispo", $row->nbdispo);
$newnode->setAttribute("depart", $row->depart);
$newnode->setAttribute("arrivee", $row->arrivee);
$newnode->setAttribute("lat", $row->lat);
$newnode->setAttribute("lng", $row->lng);
}
echo $dom->saveXML();
```

La partie javascript, plus fournie, qui récupère et interprète le fichier XML.  
On appelle la fonction initmap(), qui initialise la map, la géolocalisation, et l’affichage des données du XML.

#### script.js

```
$('#map').ready(function () {  
    initMap();  
});  
  
// ===== FUNCTION MARKERS XML JS MAP =====  
  
function initMap() {  
    //on cible l'élément map  
    var map = new  
google.maps.Map(document.getElementById("map"), {  
        //coordonnées par défaut  
        center: new google.maps.LatLng(47.6145,  
-122.3418),  
        zoom: 9,  
        //type de map  
        mapTypeId: 'roadmap'  
    });  
  
    //instantiation de Spiderfier  
    var oms = new OverlappingMarkerSpiderfier(map);  
    //instantiation de InfoWindow;  
    var infoWindow = new google.maps.InfoWindow;  
  
-----GEOLOC-----  
//si la géolocalisation est activée  
if (navigator.geolocation) {  
    //on récupère la position de l'user  
  
    navigator.geolocation.getCurrentPosition(function(position) {  
        var pos = {  
            lat: position.coords.latitude,  
            lng: position.coords.longitude  
        };  
        //on transmet la position à Infowindow  
        infoWindow.setPosition(pos);  
        infoWindow.setContent('Location found.');
```

```

        //on centre sur la position
        map.setCenter(pos);
        //on définit icones perso
        var geoicon = {
            url: "../lib/img/placeholder.png"
        };
        //on instantie le service Marker
        var markerpos = new google.maps.Marker({
            position: pos,
            map: map,
            title: 'Vous êtes ici',
            icon: geoicon
        });

//animation du marker
markerpos.setAnimation(google.maps.Animation.BOUNCE);},
function() {
    handleLocationError(true, infoWindow,
        map.getCenter());
    });

    //si erreur
    } else {
        // Browser doesn't support Geolocation
        handleLocationError(false, infoWindow,
            map.getCenter());
    }
function handleLocationError(browserHasGeolocation, infoWindow,
pos) {
    infoWindow.setPosition(pos);
    infoWindow.setContent(browserHasGeolocation ?
        'Error: The Geolocation service failed.' :
        'Error: Your browser doesn\'t support
            geolocation.');
```

-----

```
//=====TRAITEMENT XML=====

downloadUrl("../PHP/phpsqlajax_genxml.php", function (data) {
    var xml = data.responseXML;
    var markers =
xml.documentElement.getElementsByTagName("marker");

    //on boucle sur markers
    for (var i = 0; i < markers.length; i++) {

        //pour chaque markers, on récupère ses attributs
        var id = markers[i].getAttribute("id");
        var nom = markers[i].getAttribute("nom");
        var tel = markers[i].getAttribute("tel");
        var com = markers[i].getAttribute("com");
        var date = markers[i].getAttribute("date");
        var time = markers[i].getAttribute("time");
        var nbdispo = markers[i].getAttribute("nbdispo");
        var depart = markers[i].getAttribute("depart");
        var arrivee = markers[i].getAttribute("arrivee");
        //coordonnées GPS
        var point = new google.maps.LatLng(
            parseFloat(markers[i].getAttribute("lat")),
            parseFloat(markers[i].getAttribute("lng")));

// =====CREATE INFOWINDOW=====
        var html = '<div class="details-bloc">'+
            ''+
            '<p class="details details-title">'+ depart +'</p>'+
            '</div>'+
            '<div class="details-bloc">'+
            ''+
            '<p class="details details-title">'+ arrivee +'</p>'+
            '</div>'+
            '<p class="details details-infos">Départ :<strong>'+
            date + '</strong> à <strong>'+ time +
            '</strong></p>' +
            '<p class="details details-nbdispo">
<strong class="green">'+ nbdispo + '</strong> places
disponible(s)</p>'+
            '<p class="details details-com">'+ com + '</p>'+
```

```

        '<div class="btn-group">' +
        '<a class="btn details-tel" href="tel:+33' +
        tel.substr(1, 9) + '">Appeler<br>' + nom + '</a>' +
        '<button class="btn details-supr" id="' + id +
        '">Signaler<br>complet</button>' +
        '</div>';
//on créer des marqueurs pour chaque entrée
var marker = new google.maps.Marker({
    map: map,
    position: point
});
//on lie les marqueurs au service Spidifier
oms.addMarker(marker);
bindInfoWindow(marker, map, infoWindow, html)
    }
});
}
//event Marqueur
function bindInfoWindow(marker, map, infoWindow, html) {
    google.maps.event.addListener(marker, 'click', function()
{
        //on implémente la fenêtre
        infoWindow.setContent(html);
        infoWindow.open(map, marker);
    });
}
//XMLHttpRequest
function downloadUrl(url, callback) {
    var request = window.ActiveXObject ?
        new ActiveXObject('Microsoft.XMLHTTP') :
        new XMLHttpRequest;
    request.onreadystatechange = function() {
        if (request.readyState == 4) {
            request.onreadystatechange = doNothing;
            callback(request, request.status);
        }
    };
    request.open('GET', url, true);
    request.send(null);
}
function doNothing() {}

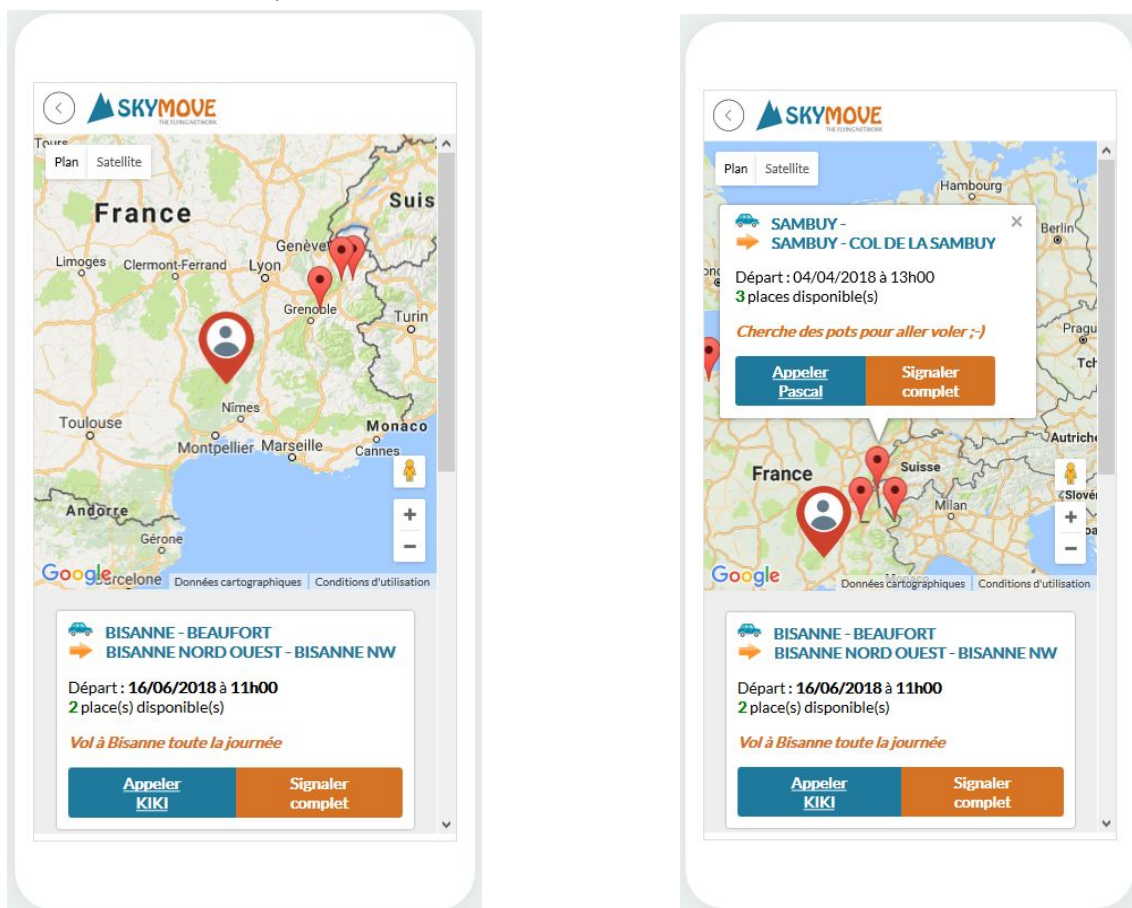
```



## Aperçu du fichier XML généré :

```
<markers>
<marker id="80" nom="Nico" tel="0647663721" com="" date="09/05/2018" time="10h00" nbdispo="1"
depart="ST HILAIRE DU TOUVET - " arrivee="ST HILAIRE DU TOUVET - Sud" lat="45.3003"
lng="5.91002"/>
<marker id="81" nom="KIKI" tel="0647663721" com="Vol à Bisanne toute la journée" date="16/06/2018"
time="11h00" nbdispo="2" depart="BISANNE - BEAUFORT" arrivee="BISANNE NORD OUEST - BISANNE
NW" lat="45.7214" lng="6.5536"/>
<marker id="82" nom="Pascal" tel="0647663721" com="Cherche des pots pour aller voler ;-)" date="04/04/2018"
time="13h00" nbdispo="3" depart="SAMBUY - " arrivee="SAMBUY - COL DE LA SAMBUY" lat="45.7"
lng="6.31722"/>
<marker id="92" nom="luk" tel="0602010405" com="yolo on teste" date="23/03/2018" time="15h00"
nbdispo="1" depart="OUFFIERES - " arrivee="LE GRAND BEC - LE GRAND BEC" lat="49.0167" lng="-
0.469444"/>
<marker id="93" nom="bebert" tel="0645218596" com="" date="07/03/2018" time="10h00" nbdispo="1"
depart="SION - VAUDEMONT - " arrivee="SION - VAUDEMONT - BARRES" lat="48.4072" lng="6.0666"/>
```

## Aperçu de la liste des covoiturages géolocalisées



L'affichage des données est désormais achevé. L'utilisateur peut accéder rapidement à la liste des covoiturages, ou les repérer sur la carte en fonction de sa position. Il reste désormais à Développer quelques fonctionnalités supplémentaires, que nous allons aborder dans le prochain chapitre.

## 7.8 - fonctionnalités supplémentaires

### Appeler un covoitureur :

Sur le détail d'un covoiturage, l'utilisateur doit pouvoir appeler contacter l'offrant rapidement et simplement.

### Signaler un covoiturage complet :

Un membre doit pouvoir signaler un covoiturage complet pour qu'il puisse être retiré de la liste, afin d'éviter des prises d'appel inutiles.

J'ai donc créé deux bouton, à l'intérieur de la fenêtre d'informations du covoiturage:



Pour pouvoir contacter un covoitureur, j'ai utilisé l'HTML. Le lien suivant lance la fonctionnalité d'appel du téléphone.

Il faut préciser dans l'attribut "href" le code région ( pour nous, +33) ainsi que le numéro de téléphone lu de la base de données.

script.js

```
//href="tel:+33" code région FR
var html = <a class="btn details-tel" href="tel:+33'
//on récupère le numéro de téléphone sans le 0 pour avoir
+ tel.substr(1, 9) + '>Appeler<br>'
+ nom + '</a>'
```

Pour signaler un covoiturage complet, j'utilise une requête AJAX qui appelle un fichier de suppression PHP et lui envoie en paramètre l'ID du covoiturage, et le supprime de la base de données.

script.js

```
$('.content').delegate('.details-supr', 'click', function(){
    var id = $(this).attr("id");
    supr(id);
});

// =====FUNCTION SUPPRESSION MANUELLE BDD=====

function supr(id){
    if (confirm('Voulez vous signaler ce covoiturage comme
complet? cette action le supprimera de la liste')) {
        $.post('../PHP/delete.php', {id:id}, function(){
            //on relance les fonctions d'affichage pour rafraichi:
            //le contenu
            list();
            initMap();
            listgeo();
        });
    }
}
```

Voici le fichier de traitement appelé :

Delete.php

```
<?php

include('conec.php');

$req = $pdo->prepare('DELETE FROM covoit WHERE id= ?');
$req->execute(array($_POST['id']));

?>
```

## Notification par mail à l'administrateur :

L'administrateur désigné doit recevoir à chaque soumission de formulaire un email de notification avec le pseudo et la date du covoiturage.

J'ai rajouté dans le fichier create.php après l'insertion en BDD le code relatif au mailing.

### create.php

```
//-----MAILING ADMIN-----

$to = "mat.skymove@gmail.com";
$subject = "Ajout d'un covoiturage sur votre Application Skymove";
$message = "<html><head></head><body><strong>".$nom."</strong>vient d'ajouter un covoiturage le <strong>".$date." </strong> à <strong>".$time."</strong></body></html>";
$headers = 'MIME-Version: 1.0' . "\r\n";
$headers .= 'Content-type: text/html; charset=UTF-8' . "\r\n";
$headers .= "Date: ".date("r (T)")." \r\n";
mail($to, $subject, $message, $headers);
```

## Auto nettoyage de la BDD :

Une fois la date d'un covoiturage passée, celui ci ne doit plus être visible.

J'ai choisi d'utiliser un script PHP qui parcourt la BDD et supprime les entrées obsolètes. Le script est appelé quand l'utilisateur charge la liste des covoiturages.

### script.js

```
// =====FUNCTION SUPPRESSION AUTO BDD=====

function checkdate() {
    $.ajax({
        url: '../PHP/checkdate.php',
        type: 'get'
    })
}
}
```

## checkdate.php

```
// =====FUNCTION SUPPRESSION AUTO BDD=====

<?php
include('conec.php');

$req = $pdo->prepare("DELETE FROM covoit WHERE date <
CURDATE()");
$req->execute();
?>
```

## Gérer la superposition des marqueurs :

Dans notre carte interactive, Il est possible que plusieurs covoiturages aient lieu au même endroit, surtout en période estivale. Il est important que tous les covoits proposés soient visibles.

J’ai découvert une petite librairie JQuery, “Spidifier” qui a été développée précisément dans ce but. Pour l’utiliser, il suffit de cibler l’élément map, et lier les marqueurs à la librairie.

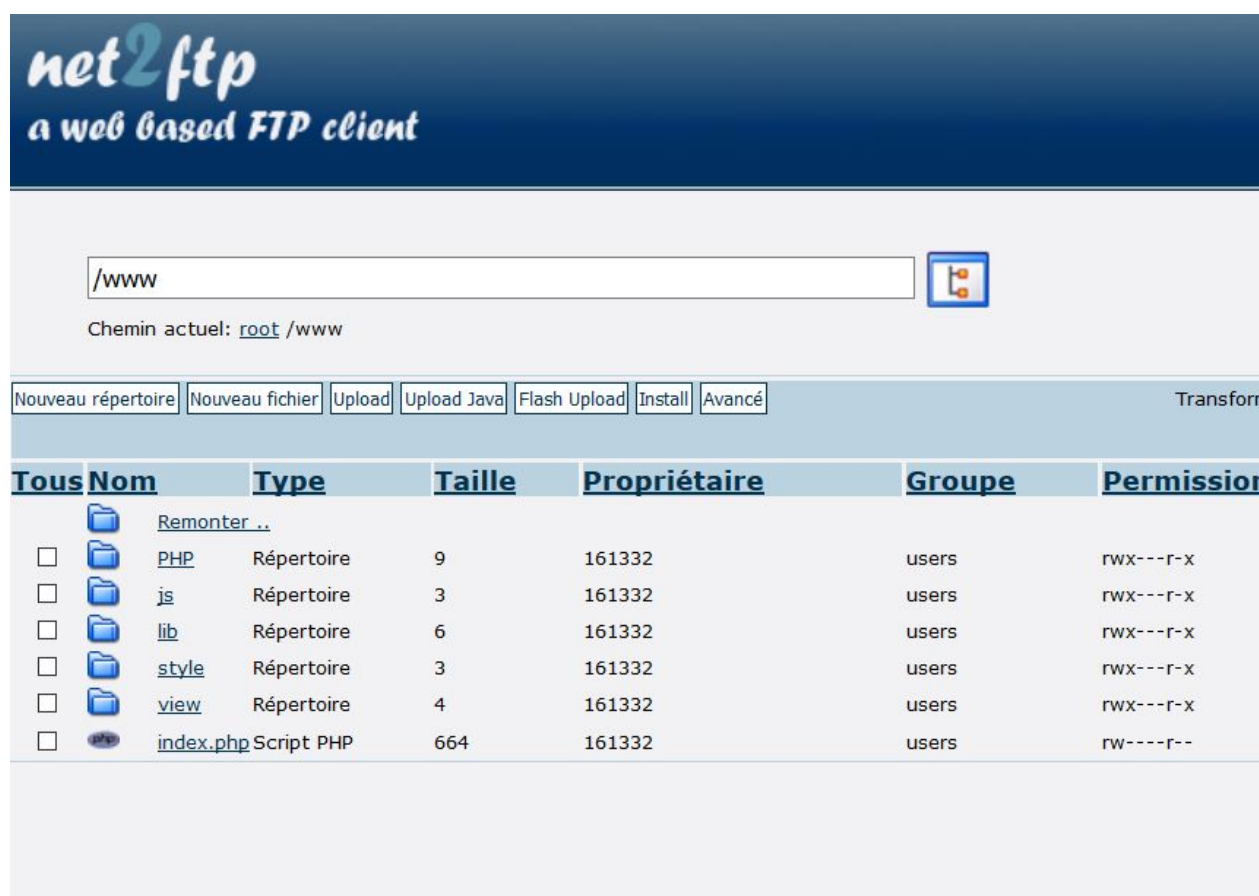
```
//après initiliasation de la map :
var oms = new OverlappingMarkerSpiderfier(map);
//après initialisations des marqueurs :
oms.addMarker(marker);
```

Désormais Les marqueurs qui se chevauchent se positionnent au clic en “toile d’araignée” tout autour de leur point géographique.

## 7.9 - Intégration et Mise en production







Le covoiturage est maintenant achevé, nous devons à présent le déployer sur l'application Skymove.

Tout d'abord, les dernières modifications doivent être push sur le FTP :



Ensuite, dans le CMS, je rajoute notre covoiturage dans le menu latéral et je le met en avant sur la page d'accueil.

Je créer la route vers notre plugin, sa vue associée et je l'intègre en utilisant le système "d'iframe" dans le code de la page.



## Code HTML

```
<!DOCTYPE html>
<html>
<head>
<title></title>
<style type="text/css">







html, body{
width: 100% !important;
height: 100% !important;
margin: 0 !important;
padding: 0 !important;
}


iframe{
width: 100% !important;
height: 99% !important;
border: none;
margin: 0 !important;
padding: 0 !important;
}

</style>
</head>
<body>

<iframe id="iframeskymove" src="https://skymove.co/index.php"></iframe>

</body>
</html>
```

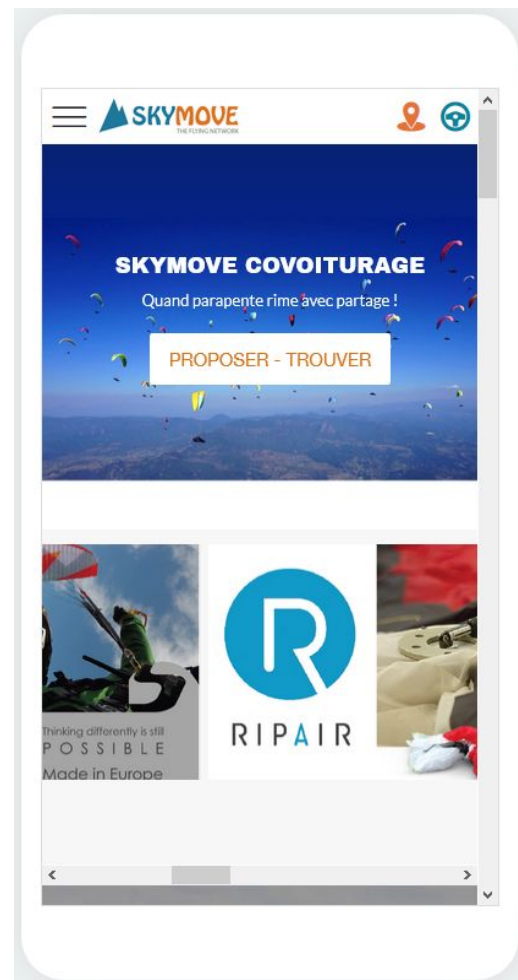
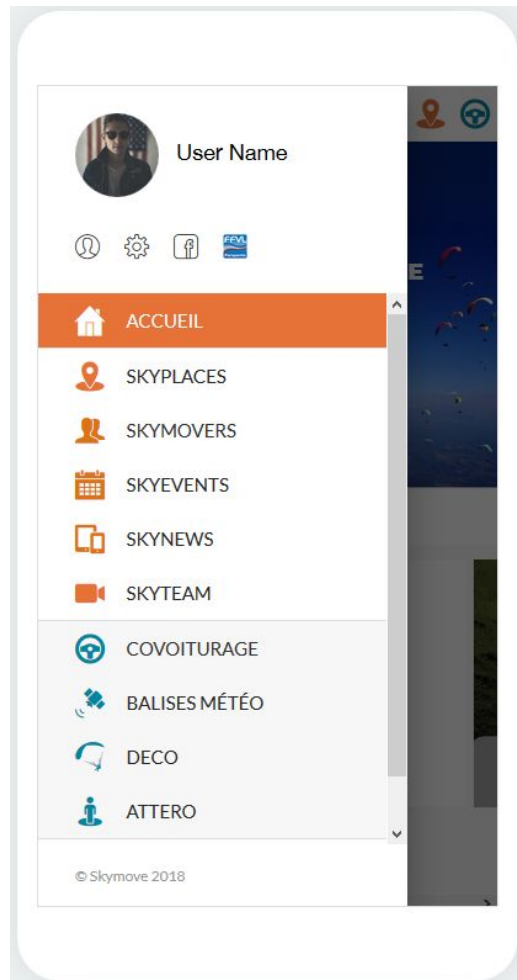


 Télécharger le fichier

Sauvegarder



Notre plugin est désormais 100% opérationnel et déployé sur notre application. Il est accessible depuis le menu latéral ou la page d'accueil d'un simple clic.





## 8 - Conclusion

Je conclus ce projet avec sérénité. Certes tout ne s'est pas déroulé comme prévu : J'ai été beaucoup limité par le CMS, et la politique de Goodbarber qui n'a pas voulu me donner l'accès à la base de données de Skymove. Mais j'ai accompli les objectifs fixés, et produit une application fonctionnelle dans le temps imparti.

Cette expérience m'a enrichi, sur le plan humain technique et professionnel. J'ai pu prendre conscience de mes connaissances et de mes lacunes, me confronter aux contraintes d'un développeur, et participer à un projet d'envergure. J'ai couvert globalement tous les langages de programmation que nous avons étudiés pendant notre parcours de formation et j'ai pu bénéficier de l'expertise de mon Formateur et de l'expérience de mes camarades.

Le nombre de téléchargement de l'application a explosé depuis l'ajout du covoiturage, et le retour des utilisateurs est plutôt positif. Mon tuteur et le Client sont contents de mon travail, et envisagent de faire appel à moi pour améliorer et pérenniser Skymove.

Il manque quelques détails à mettre en place, comme la possibilité de trier les covoiturages par date de départ, ou créer une Gestion utilisateur. Il y aura toujours quelque chose à améliorer et j'espère avoir l'occasion de retravailler sur ce projet très bientôt.

## 9 - Annexe

### Liens utiles :

- Github personnel : <https://github.com/hebi26>
- Github WikiCoda : <https://github.com/Coda-Wikicoda/>
- Goodbarber : <https://fr.goodbarber.com/>
- Skymove.fr : <https://skymove.fr>
- Skymove.co : <https://skymove.co>
- API Google Map : <https://developers.google.com/maps/documentation/?hl=fr>