



# DOSSIER DE PROJET

CREATION D'UN MODULE POUR LE RESAU SOCIAL OPEN SOUCE HUMHUB

STAGE EFFECTUE DANS LE FABLAB MADE IN IKI A COMENAILLE  
DU 01/03/2021 AU 26/04/2021



## Sommaire

<b>1) Liste des compétences du référentiel couvertes par le projet</b>	<b>-page 2</b>
<b>2) Résumé du projet</b>	<b>-page 4</b>
<b>3) Cahier des charges, expression des besoins, ou spécifications fonctionnelles du projet</b>	<b>-page 5</b>
<b>4) Spécifications techniques du projet, élaborées par le candidat, y compris pour la sécurité et le web mobile</b>	<b>-page 6</b>
<b>5) Réalisations</b>	<b>-page 9</b>
<b>6) Présentation du jeu d'essai élaboré de la fonctionnalité</b>	<b>-page 33</b>
<b>7) Description de la veille</b>	<b>-page 34</b>
<b>8) Description d'une situation de travail ayant nécessité une recherche</b>	<b>-page 36</b>
<b>9) Extrait du site anglophone, utilisé dans le cadre de la recherche décrite précédemment</b>	<b>-page 38</b>

# 1) Liste des compétences du référentiel couvertes par le projet

## 1. Partie front-end de l'application

Pour la réalisation de l'application, il a été nécessaire de :

- maquetter l'application, j'ai réalisé 4 exemples d'interface dynamique.
- créer une interface utilisateur web statique et adaptable.
- développer une interface web dynamique, qui est nécessaire pour l'intégration d'une carte dynamique

### A. Maquettage de l'application

Le maquettage de plusieurs interfaces a permis de présenter aux adhérents l'interface qui correspondait au mieux à leur demande. Les adhérents du fablab ont choisi la maquette qui affichait la liste des membres avec une carte dynamique avec des markers personnalisés.

### B. Créer une interface utilisateur web statique et adaptable

L'interface utilisateur a été faite suite au maquettage, en prenant en compte le visuel créé précédemment. Cette interface est totalement Responsive. Cette interface a été créée pour un module qui s'implémente dans l'application open source Humhub. Cette application est basée sur le framework Yii2 (Yes, it is). Les contraintes de développements de cette application est qu'elle fonctionne sous la version 3 de Bootstrap. L'utilisation de la Library Leaflet Js m'a demandé l'utilisation du CSS. La carte présente sur la maquette est supprimée pour les petites résolutions.

### C. Créer une interface utilisateur web dynamique

Toutes les pages ont été créées dynamiquement, pour permettre aux utilisateurs de naviguer sur l'application à partir de tout type d'appareil.

Pour permettre l'affichage dynamique de ces pages, j'ai utilisé Bootstrap 3, qui permet d'avoir un visuel adaptatif à la taille d'écran. Mais j'ai aussi utilisé les media queries du CSS pour certains cas particuliers comme la mise en forme des markers de la carte.



## 2. Développer la partie back-end de l'application

Pour la réalisation de l'application, il a été nécessaire de :

- créer une nouvelle table pour les compétences des adhérents
- développer les composants d'accès aux données pour afficher les informations liées aux utilisateurs et solution mais aussi mettre à jour des compétences dans la base de données ;
- développer la partie back-end de l'application en intégrant les recommandations de sécurité.

### A. Créer une nouvelle table

Suite à la demande de modification d'un formulaire texte en case à cocher, pour éviter la saisie par l'utilisateur. J'ai dû créer une nouvelle table dans la base de données de l'application humhub pour y stocker le compétence utilisateur saisie par l'administrateur du site.

### B. Développer les composants d'accès aux données

Les composants d'accès aux données ont été pour la plupart créés grâce aux modèles de Yii, qui permet de gérer le type de données envoyées. Il est composé d'un Model et d'un ModelQuery.

### C. Développer la partie back-end de l'application en intégrant les recommandations de sécurité

La partie back-end a été développée sous Yii. L'application étant à destination des membres de l'association. Les recommandations de sécurité ont bien été pris en compte de base dans l'application humhub, empêchant notamment des attaques de type « injection SQL (*Structured Query Language*) » ou encore le hachage des mots de passe. Plus d'information Page 36.



## 2) Résumé du projet

### Résumé :

Mon stage a été réalisé en télétravail au sein du fablab Made in iki de Commenailles.

Made in iki est une association où sont mis à disposition toutes sortes d'outils. Les membres de cette association mettent à disposition leur savoir-faire pour réaliser toute sorte de projet. Tout en conservant l'esprit open source de leur travail.

Dans un premier temps j'ai transformé leur Nas en serveur web et j'y ai installé l'application web Humhub.

A la demande des membres et du président de l'association, j'ai développé un module s'intégrant à l'application web humhub. Ce module a pour but d'aider les membres à trouver une personne ayant des compétences nécessaires pour un projet.

Pour éviter les saisies par l'utilisateur et son cortège d'erreur, ils m'ont demandé de modifier le formulaire des compétences dans le profil utilisateur en liste de choix. Ces compétences doivent être gérées par l'administrateur.

Humhub est un outil de création de réseau social sous licence AGPL V3 programmé majoritairement en PHP, autour du framework Yii 2. Le principe de fonctionnement ressemble à celui de Facebook.

Yii 2 Framework est un Framework PHP tout en programmation Orienté Objet (POO) sur le principe modèle-vue-contrôleur (MVC).

La partie front-end s'est faite sur la base de l'utilisation de la Library Leaflet Js. L'intégration d'une carte openstreetmap avec des markers personnalisés et dynamique.

La partie back-end, quant à elle, a été gérée à l'aide de phpMyAdmin pour le système de gestion de base de données.

A ce jour, Les membres du fablab souhaitent rajouter un niveau de qualification à ces compétences.



### **3) Cahier des charges, expression des besoins, ou spécifications fonctionnelles du projet**

#### **1-Contexte**

Le projet qui m'a été confié a pour but de répondre au besoin de trouver l'adhérent dans l'association avec les bonnes compétences et de le géolocaliser. Dans un second temps d'adapter l'application web Humhub au besoin du fablab, en changeant l'interface et la gestion du profil.

#### **2-Délais**

Les délais de livraison pour la réalisation de cette application ont été défini sur la période de stage du 01 Mars au 26 Avril 2020.

#### **3-Environnement de travail**

L'absence de personne présente sur les lieux et le peu de disponibilité des membres de l'association. La difficulté fut d'obtenir les informations nécessaires au projet.

Mon stage c'est dérouler en distanciel et par une réunion hebdomadaire.

L'absence de connaissances sur le Framework Yii et Le peu d'aide que j'ai pu obtenir. Ont été une difficulté au début, mais cela m'a appris à me débrouiller seul et à chercher par moi-même et surtout comment faire mes recherches.



## 4) Spécifications techniques du projet

### 1-Présentation Humhub

Humhub est un outil de création de réseau social sous licence AGPL V3 programmé majoritairement en PHP, autour du framework Yii. Humhub permet de créer son propre réseau social non fédéré, ou un réseau social d'entreprise. La version stable actuelle est 1.2.8.

Le principe de fonctionnement ressemble à celui de Facebook :

- Chaque utilisateur a sa page sur laquelle il peut publier du contenu :
  - Texte formaté en Markdown;
  - Photos, vidéos, sons ;
  - Sources externes (YouTube, Soundcloud, Vimeo...) grâce à OEmbed.
- L'utilisateur peut appartenir à des espaces publics/privés (les groupes), et partager du contenu dessus.
- L'utilisateur peut suivre d'autres utilisateurs, ou être ami avec eux.
- Un moteur de recherche permet de trouver du texte dans les différents types de contenus.
- Des modules permettent d'ajouter des fonctionnalités supplémentaires, pour que chaque administrateur puisse configurer Humhub selon ses besoins :
  - Galeries photos
  - Wiki
  - Agenda
  - Sondages
  - Messagerie
  - OnlyOffice...
- L'authentification peut se faire avec LDAP.

Vous pourrez trouver la documentation de Humhub pour le développement sur <https://docs.humhub.org>

Vous trouverez la documentation du framework Yii sur <https://www.yiiframework.com>



## 2-Technologie

Le choix de l'association s'est porté sur l'application humhub open source et gratuite. Cette application est développée principalement en PHP et en utilisant le Framework Yii et Framework Bootstrap 3. Ne connaissant pas le Framework Yii j'ai dû dans un 1<sup>er</sup> temps découvrir et me former sur cette nouvelle technologie. Ayant fait la connaissance de Symfony pendant ma formation, j'ai pu appréhender plus facilement le fonctionnement de ce Framework. De plus tout comme Symfony, Yii est livré avec une sécurité native et avec son découpage MVC, il permet de travailler plus facilement en segmentant les tâches de travail. Yii2 possède sa propre interface de base de données utilisant le modèle ActiveRecord.

## 3-Accessibilité

Pour résoudre les problèmes d'accessibilité en fonction des différents types de supports (mobile, tablette, ordinateur, etc..), j'ai utilisé le Framework Bootstrap3, qui permet de gérer beaucoup plus facilement l'affichage de l'application sur différentes résolutions. L'application humhub est développée avec cette version de Bootstrap, j'ai dû m'adapter pour éviter de casser le rendu de l'application.

## 4-Sécurité

La sécurité est gérée principalement par Yii. En effet, pour chaque table du modèle conceptuel de données, nous créons une migration sur Yii.

L'objet Active Record fournit une interface orientée objet pour accéder aux données stockées dans une base de données. Une classe Active Record est associée à une table de base de données, une instance Active Record correspond à une ligne de cette table et un attribut d'une instance Active Record représente la valeur d'une colonne de cette ligne. Au lieu d'écrire des instructions SQL brutes, vous pouvez utiliser Active Record de manière orientée objet pour manipuler les données dans les tables de base de données. Cette classe permet de lutter contre l'injection SQL.

Le Framework Yii pour la sécurité utilise :

- l'authentification pour la vérification de l'identité d'un utilisateur. Il utilise généralement un identifiant (par exemple un nom d'utilisateur ou une adresse e-mail) et un jeton secret (par exemple un mot de passe ou un jeton d'accès) pour juger si l'utilisateur est celui qu'il prétend. L'authentification est la base de la fonction de connexion.





-L'autorisation est le processus de vérification vérifiant qu'un utilisateur dispose des autorisations suffisantes pour faire quelque chose. Yii propose deux méthodes d'autorisation : le filtre de contrôle d'accès (ACF) et le contrôle d'accès basé sur les rôles (RBAC).

- Les mots de passes, Yii utilise l'algorithme de hachage bcrypt.

- La Cryptage et le décryptage, Yii fournit des fonctions d'assistance pratiques qui vous permettent de crypter / décrypter les données à l'aide d'une clé secrète. Les données sont transmises via la fonction de cryptage afin que seule la personne qui possède la clé secrète puisse la décrypter.

- Par sécurisation de la vue, lors de la création de vues générant des pages HTML, il est important d'encoder et/ou de filtrer les données provenant des utilisateurs finaux avant de les présenter. Sinon, on peut faire l'objet d'attaques de scripts intersites.

Voire page 36 pour plus d'informations sur la sécurité et les bonnes pratiques.

### **5-Documentation du code**

Toutes les pages que j'ai modifiées ont été commentées pour préciser chaque partie et rendre le code accessible pour une personne extérieure souhaitant intervenir sur l'application.

### **6-Validation de la structure des pages**

La validation de la structure des pages via le site W3C m'a permis de me conformer aux standards du web et de repérer les balises mal fermées.



## 5) Réalisations

### 1-Installation et configuration de Humhub :

Après la conversion du NAS en serveur web il m'a suffi de décompresser le code téléchargé sur le site d'Humhub dans le dossier web pour que le site soit en ligne. J'ai réalisé la même chose en local sur ma machine.

Humhub a une interface de configuration à la première installation. Cette interface permet de créer la base de données et un accès administrateur et du contenu générique. Il y a juste à créer une nouvelle base de données sous PhpMyAdmin.

The screenshot shows the 'Configuration de la base de données' (Database Configuration) screen of the HumHub installation wizard. The interface is in French and includes the following fields and instructions:

- Nom d'hôte** (Host): Set to 'localhost'.
- Nom d'utilisateur** (Username): Set to 'root'.
- Mot de passe** (Password): A masked password field.
- Nom de la base de données** (Database Name): Set to 'humhub'.

Instructions in French: 'Vous devez préciser ci-dessous les détails de votre connexion à la base de données. Si vous n'êtes pas certain de ces renseignements de contacter votre administrateur système.' and 'Le script de la base de données sera téléchargé à partir de nos serveurs.' A 'Sauver' (Save) button is at the bottom.

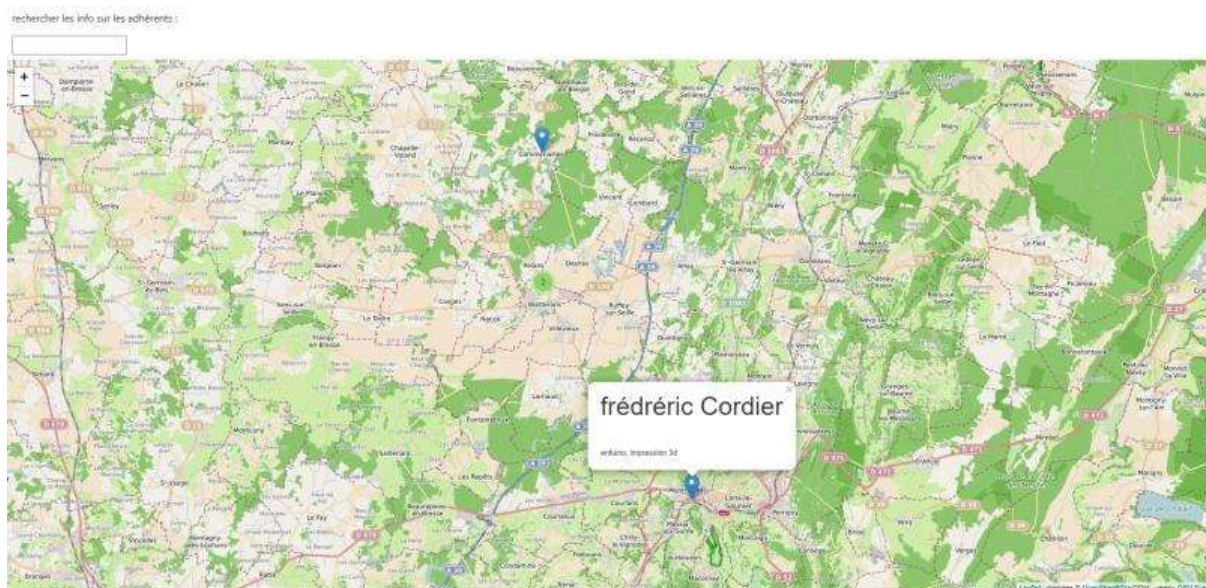


## 2-conception d'un module pour Humhub :

### 1-création de l'interface utilisateur

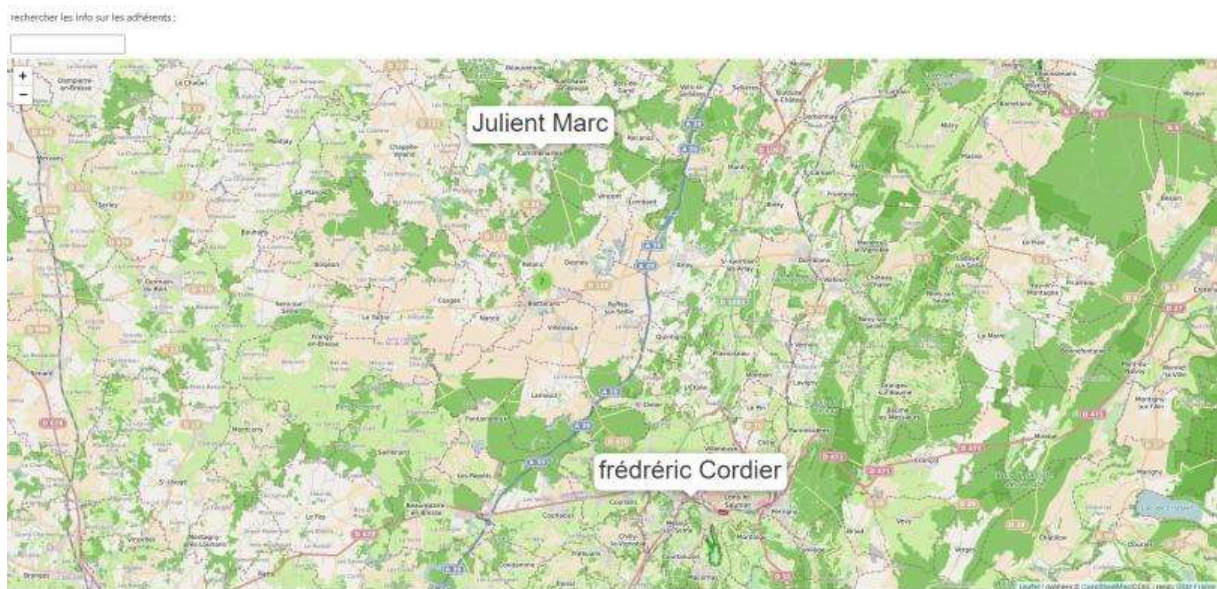
Le cahier des charges est assez bref, j'ai pris l'initiative de développer 4 interfaces et de les présenter à l'association.

- La première montre une barre de recherche et une carte avec des markers simple qui donnent accès à des popups avec toutes les informations des adhérents.





- La deuxième également montre une barre de recherche et une carte avec des markers personnalisés qui donnent accès à des popups avec toutes les informations des adhérents.

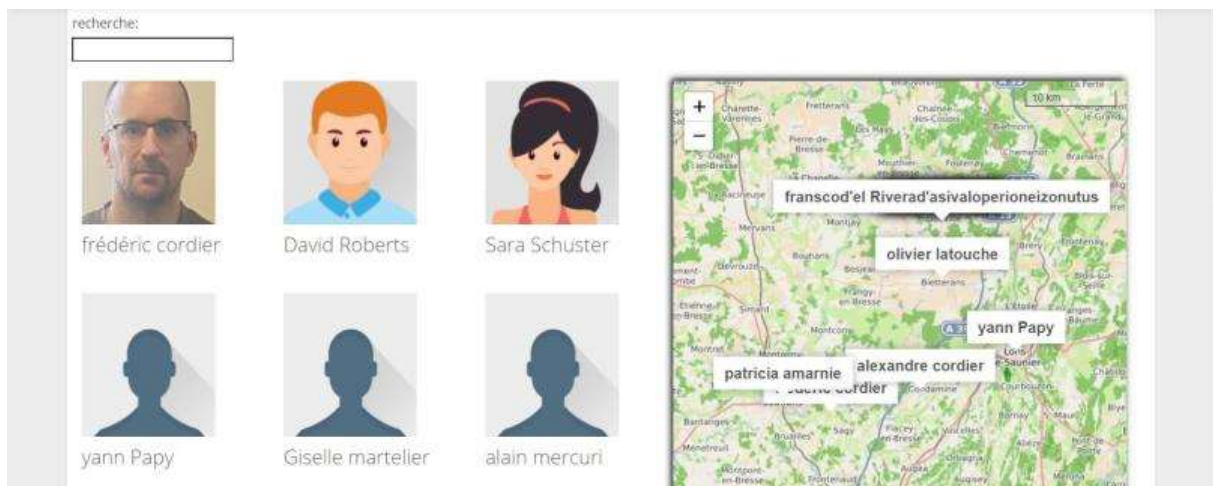


- La troisième également montre une barre de recherche. Une liste d'utilisateurs avec leurs compétences et une carte avec des markers personnalisés qui donnent accès à des popups avec toutes les informations des adhérents. Cet exemple est dynamique au survol de la fiche utilisateur le marker sur la carte passe en bleu et si l'on clique dessus les infos de l'adhérent apparaissent sur la carte à la place du marker.





- La quatrième est identique à la dernière mais ne montre que les informations de l'adhérent que sur la carte.



Les adhérents ont choisi la dernière interface.

## 2-création d'un module

Le Framework Yii fonctionne avec des modules de développement, j'ai utilisé le module GII.

### 2.1 Qu'est-ce que GII

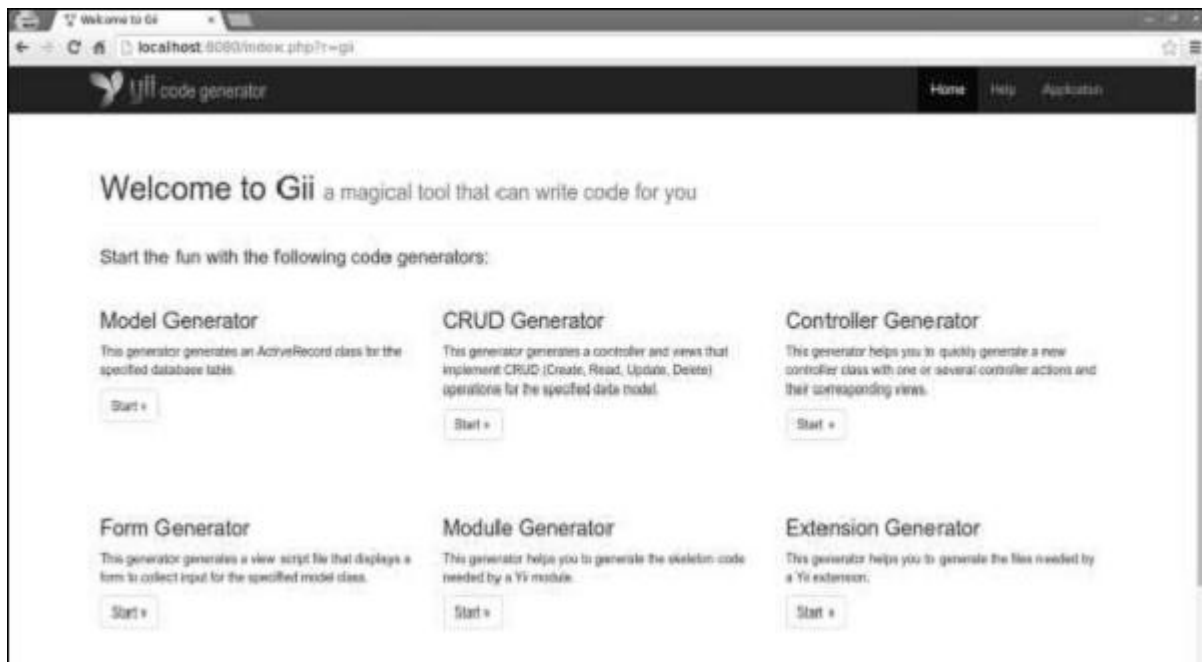
Gii est l'extension, qui fournit un générateur de code basé sur le Web pour générer des modèles, des formulaires, des modules, des CRUD, etc.

Par défaut, les générateurs suivants sont disponibles -

- **Model Generator** - Génère une classe ActiveRecord pour la table de base de données spécifiée.
- **CRUD Generator** - Génère un contrôleur et des vues qui implémentent les opérations CRUD (Create, Read, Update, Delete) pour le modèle spécifié.



- **Controller Generator** - Génère une nouvelle classe de contrôleur avec une ou plusieurs actions de contrôleurs et leurs vues correspondantes.
- **Générateur de formulaires** - Génère un fichier de script de vue qui affiche un formulaire pour collecter l'entrée pour la classe de modèle spécifiée.
- **Module Generator** - Génère le code squelette nécessaire à un module Yii et le système d'activation et de désactivation du module.
- **Générateur d'extension** - Génère les fichiers nécessaires à une extension Yii.



## 2.2 installations de GII sur humhub

Des développeurs ont mis à disposition un module adapté pour le développement de plugin et d'autres éléments pour Humhub. Le plugin s'appelle **Developer Tools**, Vous trouverez les informations à cette adresse <https://github.com/humhub-contrib/devtools>

Télécharger le module en ligne de commande :

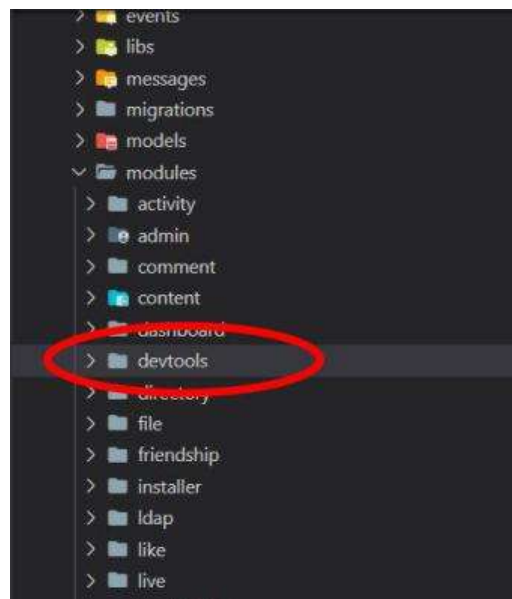
Aller dans le dossier modules: `cd /path/to/your/humhub/protected/humhub/modules`

```
git clone https://github.com/humhub/humhub-modules-devtools.git devtools
```

Et taper la ligne de commande : `git clone https://github.com/humhub/humhub-modules-devtools.git devtools`.



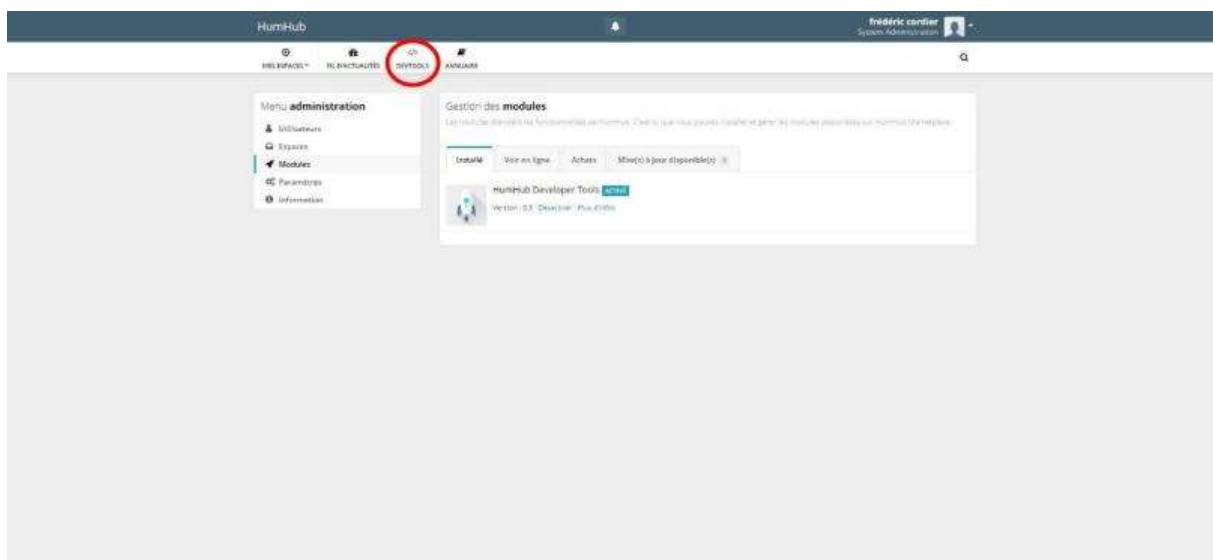
Cela va télécharger dans le dossier  
modules devtool



Un fois le téléchargement fait retournez sur le navigateur et connectez-vous en temps que  
compte administrateur du site humhub.

Allez dans la partie administration et dans le menu Modules activez le module humhub  
developer tools.

Une fois activé le menu devtools apparait dans la barre d'onglet.



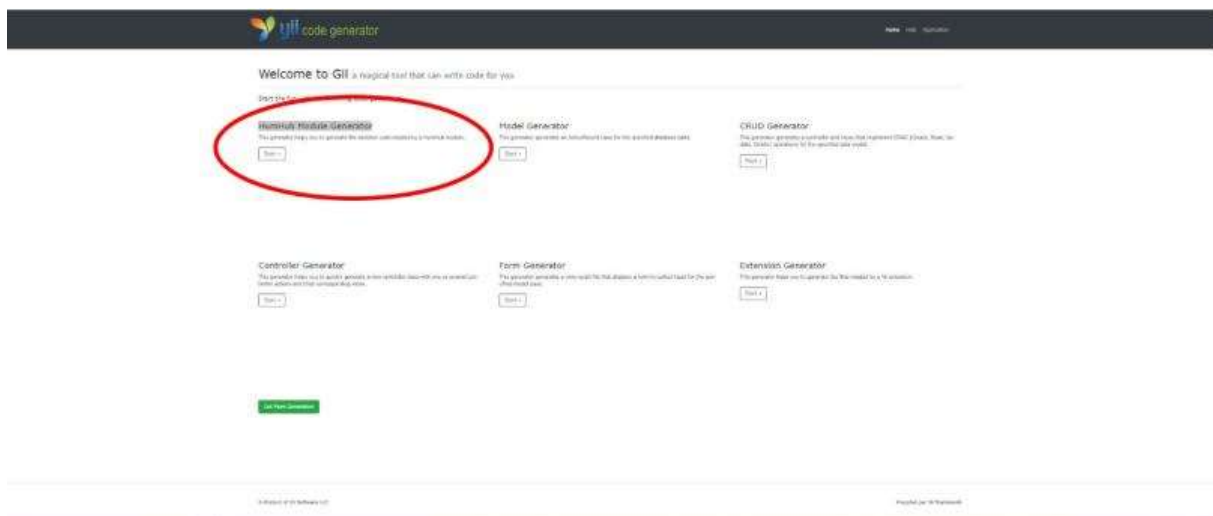






```
1 <?php
2
3 require_once(__DIR__ . '/functions.php');
4
5 /**
6  * This file provides to overwrite the default HumHub / Yii configuration by
7  * your local Web environments
8  * @see http://www.yiiframework.com/doc-2.0/guide-concept-configurations.html
9  * @see http://docs.humhub.org/admin-installation-configuration.html
10  * @see http://docs.humhub.org/dev-environment.html
11  */
12
13 return
14 [
15     'bootstrap' => ['gii'],
16     'modules' =>
17     [
18         'gii' =>
19         [
20             'class' => 'yii\gii\Module',
21             'allowedIPs' => ['127.0.0.1', '::1'],
22             'generators' =>
23             [
24                 'module' =>
25                 [
26                     'class' =>
27                     'humhub/modules/devtools/gii/generators/ModuleGenerator',
28                     'templates' =>
29                     [
30                         'humhub' => '@app/modules/devtools/default',
31                     ]
32                 ]
33             ]
34         ]
35     ]
36 ];
```

Vous arrivez sur la page de Gii comme ci-dessous, cliquez sur le start du menu HumHub Module Generator.



Il est demandé en premier l'emplacement du module. Cela dépend de vos besoins, si vous désirez qu'il apparaisse automatiquement dans la barre d'onglet comme pour le module devtools. Alors place le module dans myCompany\humhub\modules\ .

Dans le cas contraire placez le module dans protected\modules\ .

Ensuite rentrez le nom de votre module, sans caractères spéciaux ni d'espaces.



Choisissez votre icône de modules et cochez la case Is Space Module si votre module affiche une nouvelle page. Si votre module concerne également les profiles utilisateurs, cochez également la case Is User Profile Module.

Laissez par défaut Output Path et Code Template.

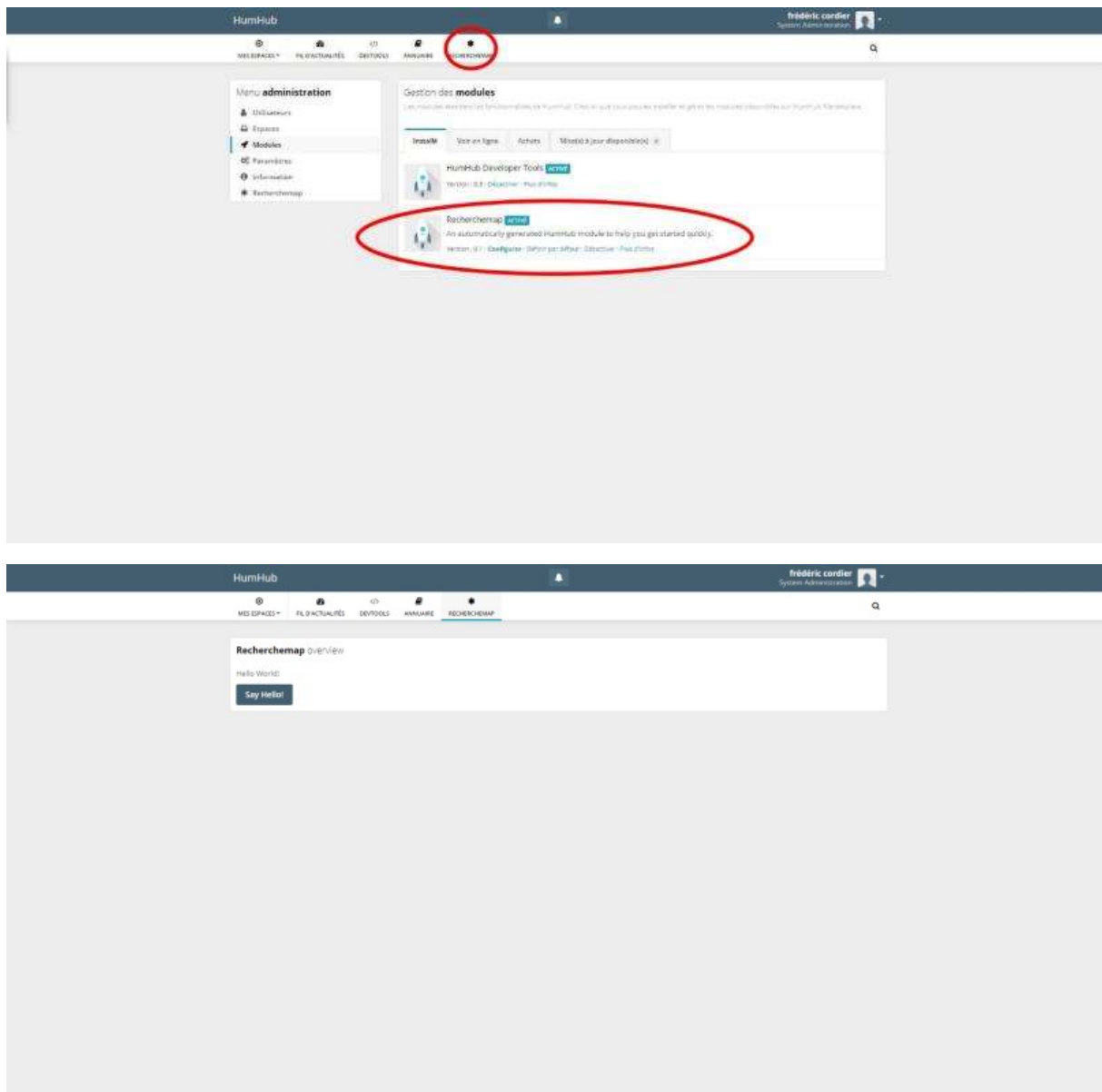
Cliquez sur Previw puis sur generate.

Voilà le module a été généré dans le dossier protected\humhub\modules\devtools/result

Il ne reste plus qu'à le copier-coller le module dans le dossier protected\humhub\modules\

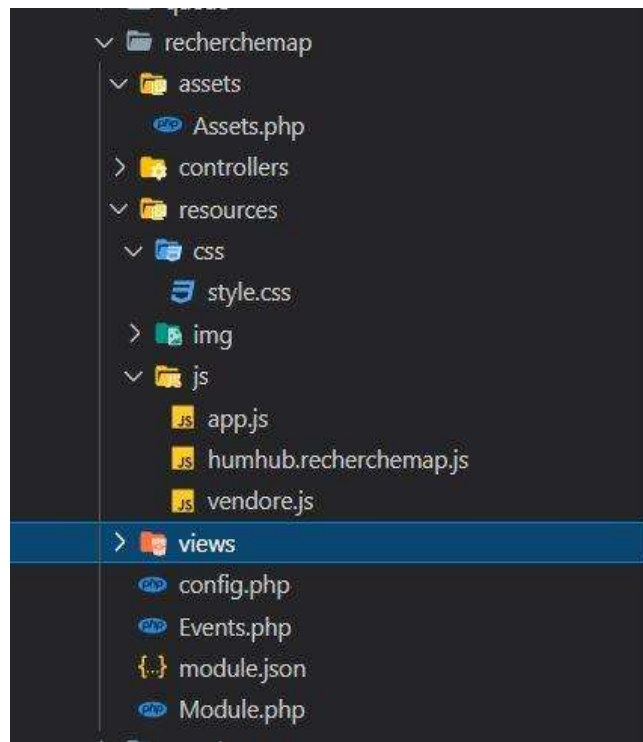
Après cette étape, on retourne sur notre navigateur et l'on se connecte toujours en tant qu'administrateur du site. Dans l'administration puis dans la partie module on voit apparaitre notre module. Il ne reste plus qu'à l'activer.

Un fois activé, on voit l'onglet apparaitre dans la barre et quand on l'ouvre on voit une page avec un titre est un bouton.



### 3-Modifications des assets et lien vers ces fichiers de dépendances js, css, bootstrap

La gestion des fichiers CSS et Javascript du module se gère par le biais du fichier assets/Assats.php et se place dans le dossier resources.



Comme indiqué ci-dessus, il faut créer un dossier CSS et y créer un fichier style.css

Vous pouvez faire la même chose pour vos images et vos fichiers Javascript.

Un fois ces fichiers créés il faut déclarer dans le fichier Assets.php et y renseigner les différents paramètres de lien :

- \$sourcePath -> définit le chemin de vos actifs du module.

- \$jsOptions -> définit où les fichiers js sont inclus dans la page, notez que vos fichiers js personnalisés doivent être inclus après les fichiers de base (qui sont inclus dans head)

Définit où les fichiers js sont inclus dans la page, notez que vos fichiers js personnalisés doivent être inclus après les fichiers de base (qui sont inclus dans head)

-\$publishOptions -> changez forceCopy sur true lors du test de vos js afin de reconstruire ces actifs à chaque demande (sinon ils seront mis en cache)

- \$css -> définit le chemin de vos fichier css

- \$js -> définit le chemin de vos fichier JavaScripte

- \$depends -> définit le chemin des dépendances

Vérifier bien que \$sourcePath retourne bien à votre dossier ressources.



Ainsi que le lien vers vos fichiers css et Javascript, comme indiqué ci-dessous.

Pour les dépendances, pour garder le thème de base de humhub il faut intégrer yii\web\YiiAsset .

Pour Utiliser Bootstrap dans votre module il faut intégrer yii\bootstrap\BootstrapAsset , comme ci-dessous.

! important : humhub a été développé en utilisant bootstrap 3, vous ne pouvez pas passer à la version 4 de bootstrap car elle ne sera pas prise en charge et pourrait entrainer des dysfonctionnements de l'affichage de l'application web humhub.

```
16 class Assets extends AssetBundle
17 {
18     // Base string defines the path of your module assets
19     public $sourcePath = '@webresources';
20     // Not string defines the path of your module assets
21     public $jsOptions = '@webresources';
22     // Base string defines where the JS files are included into the page, note
23     // your custom JS files should be included after the core files (which are
24     // included in head)
25     public $jsOptions = ['position' => \yii\web\View::POS_HEAD];
26     // What script should be loaded in the head, loading your JS in the head
27     // instead of the body is more common (otherwise they will be rendered)
28     public $cssOptions = [
29         'position' => \yii\web\View::POS_HEAD
30     ];
31     public $css = [
32         'css/bootstrap.css'
33     ];
34     public $js = [
35         '@webresources/js',
36         '@webresources/js',
37         '@webresources/js'
38     ];
39     public $depends = [
40         'yii\web\YiiAsset',
41         'yii\bootstrap\BootstrapAsset'
42     ];
43 }
```

lien vers les ressources

lien vers le css

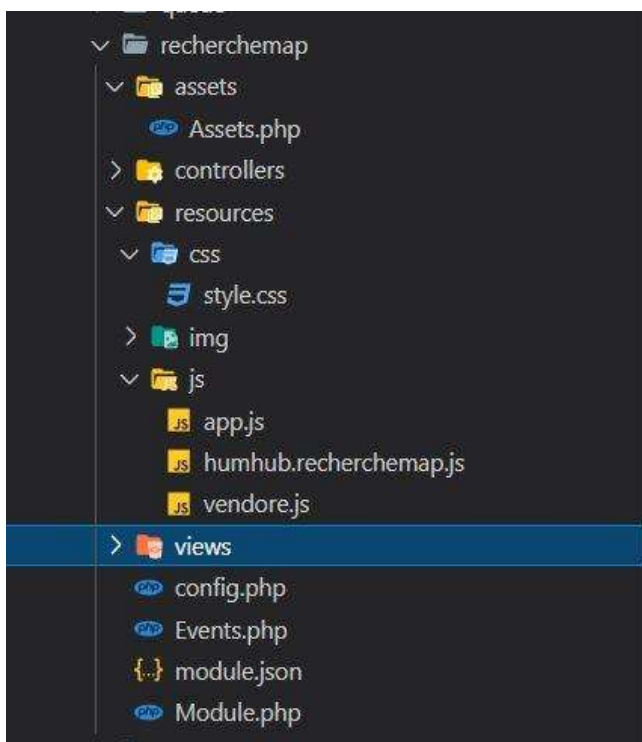
lien vers le JavaScript

lien vers les dépendance



## 4-Personalisation du module

### -structure du module



Le module est structuré en MVC (Modèle-vue-contrôleur).

-Les assets gèrent le chemin d'accès aux différentes dépendances qu'a besoin le Module.

-Le Controller qui récupère les données et les renvoie à la vue.

-Les ressources qui contiennent le fichier CSS et JavaScript et la différente ressource nécessaire au module.

-La vue qui contient l'affichage du module mais aussi de l'affichage de l'interface de configuration et de présentation du module.

-Le fichier config.php contient généralement le nom de la classe de l'objet en cours de création, et une liste de valeurs initiales qui doivent être assignées aux propriétés de l'objet.

-Le fichier Events sont utilisables à la fois pour personnaliser le comportement de base et pour rendre les applications et les modules plus flexibles.

-Le fichier Module contient la classe module qui gère la suppression ou l'ajout du module dans l'interface utilisateur.

-Le fichier module.json contient les données qui seront utilisées pour l'affichage dans l'interface administrateur



## - Récupération des données et les affichées

Les données dont nous avons besoin pour afficher la liste des utilisateurs et leurs compétences ainsi que leurs adresses se trouvent dans deux tables séparées.

J'ai dû modifier la table profil et lui rajouter la latitude et la longitude nécessaire au positionnement des markers sur la carte.

J'ai fait appelle à une api (open.mapquestapi.com) qui converti une adresse postale en latitude longitude.

Pour éviter de surcharger j'ai intégré cette api dans la validation du formulaire de profil. J'ai fait remplir automatiquement à chaque validation du formulaire, les champs latitude et longitude.

En utilisent la méthode CURL.

Field	Type
user_id	int(11)
first_name	varchar(255)
last_name	varchar(255)
title	varchar(255)
gender	varchar(255)
street	varchar(255)
zip	varchar(255)
city	varchar(255)
country	varchar(255)
state	varchar(255)
birthday_hide_year	int(1)
birthday	date
about	text
phone_private	varchar(255)
phone_work	varchar(255)
mobile	varchar(255)
fax	varchar(255)
im_skype	varchar(255)
im_xmpp	varchar(255)
url	varchar(255)
url_facebook	varchar(255)
url_linkedin	varchar(255)
url_xing	varchar(255)
url_youtube	varchar(255)
url_vimeo	varchar(255)
url_flickr	varchar(255)
url_myspace	varchar(255)
url_twitter	varchar(255)
lat	float(11,6)
lng	float(11,6)

```
if ($form->submitted('save') && $form->validate() && $form->save()) {
    if ($user->profile->city != null) {

        $curl = curl_init("http://open.mapquestapi.com/geocoding/v1/
        address?key=p7SEPMy7uimN1D7jnf0U3KtclKmdlco&location=" .
        $user->profile->street . ' ' . $user->profile->zip . ' ' .
        $user->profile->city);
        curl_setopt($curl, CURLOPT_CAINFO, __DIR__ .
        DIRECTORY_SEPARATOR . 'cert.cer'); // je n'ai pas utiliser
        curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
        $data = curl_exec($curl);
        if ($data === false) {
            var_dump(curl_error($curl));
        } else {
            if (curl_getinfo($curl, CURLINFO_HTTP_CODE) == 200) {
                $data = json_decode($data, true);
                if (isset($data["results"][0]["locations"][0]["latlng"]))
                {
                    $user->profile->lat = $data["results"][0]
                    ["locations"][0]["latlng"]["lat"];
                    $user->profile->lng = $data["results"][0]
                    ["locations"][0]["latlng"]["lng"];

                    $user->profile->save();
                    // dd($user->profile);
                } else {
                    echo '<h1> un erreur est survenu veuillez contacter
                    le créateur du site.</h1>';
                }
            }
        }
        curl_close($curl);
    } else {
        $user->profile->lat = null;
        $user->profile->lng = null;
        $user->profile->save();
    }
}
```

Maintenant il me reste à transférer via le contrôleur les informations dont j'ai besoin dans la vue. Pour cette partie cela ressemble beaucoup à Symfony.

Nous avons des models qui contiennent les champ de ma table ( entity pour symfony) et des modelsQuery qui permettent de récupérer les informations dont on a besoin ( les repository pour Symfony).





Tout cela est généré automatiquement par le module Gii vue précédemment.

### Le models User.php

```
class User extends \yii\db\ActiveRecord
{
    /**
     * @inheritdoc
     */
    public static function tableName()
    {
        return '{{user}}';
    }

    /**
     * @inheritdoc
     */
    public function rules()
    {
        return [
            [['status', 'created_by', 'updated_by', 'visibility',
             'contentcontainer_id'], 'integer'],
            [['auth_mode'], 'required'],
            [['tags'], 'string'],
            [['created_at', 'updated_at', 'last_login', 'safe'],
             'integer'],
            [['guid'], 'string', 'max' => 45],
            [['username'], 'string', 'max' => 50],
            [['email'], 'string', 'max' => 150],
            [['auth_mode'], 'string', 'max' => 10],
            [['language'], 'string', 'max' => 5],
            [['time_zone'], 'string', 'max' => 100],
            [['authclient_id'], 'string', 'max' => 60],
            [['email'], 'unique'],
            [['username'], 'unique'],
            [['guid'], 'unique'],
            [['authclient_id'], 'unique'],
        ];
    }

    /**
     * @inheritdoc
     */
    public function attributeLabels()
    {
        return [
            'id' => 'ID',
            'guid' => 'Guid',
            'status' => 'Status',
            'username' => 'Username',
            'email' => 'Email',
            'auth_mode' => 'Auth Mode',
            'tags' => 'Tags',
            'language' => 'Language',
            'created_at' => 'Created At',
            'created_by' => 'Created By',
            'updated_at' => 'Updated At',
            'updated_by' => 'Updated By',
            'last_login' => 'Last Login',
            'visibility' => 'Visibility',
            'time_zone' => 'Time Zone',
            'contentcontainer_id' => 'Contentcontainer ID',
            'authclient_id' => 'Authclient ID',
        ];
    }
}
```

### Le Query UserQuery.php

```
<?php

namespace app\models;
use yii\db\ActiveQuery;

You, seconds ago | 1 author (You)

/**
 * This is the ActiveQuery class for [[User]].
 *
 * @see User
 */
class UserQuery extends ActiveQuery
{
    /**public function active()
    {
        return $this->andWhere('[[status]]=1');
    }*/

    /**
     * @inheritdoc
     * @return User[]|array
     */
    public function all($db = null)
    {
        return parent::all($db);
    }

    /**
     * @inheritdoc
     * @return User|array|null
     */
    public function one($db = null)
    {
        return parent::one($db);
    }
}
```

Dans le contrôleur j'ai appelé les informations de ma table user « User::find() » et je lui ajoute les informations de la table profile « User::find()->with('profile') ».

Et je récupère toutes les informations de la table user « ->all() », et je le retourne à la vue.

```
<?php

namespace myCompany\humhub\modules\recherche\controllers;

use app\models\PostTags;
use humhub\components\Controller;
use app\models\User;
use app\models\UserQuery;

You, seconds ago | 1 author (You)

class IndexController extends Controller
{
    public $subLayout = "@recherche/views/layouts/default";

    /**
     * Renders the index view for the module
     *
     * @return string
     */
    public function actionIndex()
    {
        //renvoyer mes utilisateur
        $users = User::find()->with('profile')->all();
        //renvoyer la compétence
        $queryPostTags = PostTags::find()->all();

        if (sleep(5)) {
            header("Location: /index.php?r=recherche%2Findex");
            exit;
        }
        return $this->render('index', [
            'users' => $users,
            'skill' => $queryPostTags
        ]);
    }
}
```





Dans la vue, je réalise un foreach pour afficher tous les profils utilisateur. Et je lui passe toutes les informations qui vont m'être utiles à la création des markers sur la carte. Comme les données GPS ou encore les informations du profil. La mise en forme utilise Bootstrap 3 et mon fichier CSS, tout cela rend la page responsive.

```
<div class="container-fluid">
  <div class="row">
    <div class="list row col-xs-12 col-sm-5 col-lg-7" id="documents">
      <?php foreach ($users as $user) : ?>

        <div class='document item js-marker col-xs-12 col-lg-4 '
          <?php if ($user->profile->lat != null &&
            $user->profile->lng != null) : ?> data-lat='<?= Html::encode(
              "{$user->profile->lat}" ) ?>' data-lng='<?= Html::encode(
                "{$user->profile->lng}" ) ?>' <?php else : ?> data-lat='46.
              80<?php echo rand(270, 300); ?>' data-lng='5.45<?php echo
              rand(170, 200); ?>' <?php endif; ?> data-nom='<?=
              Html::encode("{$user->profile->firstname}
              {$user->profile->lastname}") ?>'>

          <?php if (file_exists("uploads/profile_image/
            {$user->guid}.jpg")) : ?>
            <?= Html::img("uploads/profile_image/{$user->guid}.
              jpg", ['alt' => 'My logo']) ?>
          <?php else : ?>
            <?= Html::img("static/img/default_user.jpg", ['alt'
              => 'My logo']) ?>
          <?php endif; ?>
          <h4><?= Html::encode("{$user->profile->firstname}
            {$user->profile->lastname}") ?> </h4>

          <div class="display">
            <h6>Mes compétences: </h6>
            <p>
              <?= Html::encode("{$user->tags}") ?>
            </p>
            <h6>Contacter moi: </h6>
            <p>
              <?= Html::encode("{$user->email}") ?>
            </p>
          </div>
        </div>
      <?php endforeach; ?>
    </div>
  </div>
```



Il ne reste plus qu'à créer une div contenant la carte et générer la carte avec la librairie JavaScript Leaflets.JS .

Utilisation de la librairie scriptjs pour le traitement asynchrone, ainsi que le polyfill pour l'utiliser le Array.form(), pour être sûr qu'il soit bien supporté par tous les navigateurs.

La documentation : <https://leafletjs.com>

<https://www.openstreetmap.fr>

En premier je sélectionne ma div où va s'afficher la carte.

```
let $map = document.querySelector('#map');
```

Ensuite je crée un class LeafletMap qui s'occupe de l'affichage de la carte.

Dans cette class je prépare mon constructeur avec l'initialisation de mes variables.

Après vient le traitement de la carte. J'ai choisi d'utiliser une fonction asynchrone pour éviter d'attendre le chargement de la carte pour exécuter le reste de mon code.

Pour éviter l'appelle des CDN de la Librairie leaflet sur tous les pages du site. Je l'appelle dans cette class.

Ensuite nous utilisons les fonctionnalités de leaflet pour afficher la carte.

Je crée une fonction pour ajouter des markers et une autre fonction pour le centrage de la carte par rapport aux markers.

```
class LeafletMap {
  constructor() {
    this.map = null;
    this.bounds = []; //tableau des différent point des marker pour centrer
    la carte
  }

  async load(element) {
    return new Promise((resolve, reject) => { // on retourne une promesse
      // pour charger la carte en asynchrone on utilisant la librairie scriptjs
      $script('https://unpkg.com/leaflet@1.3.1/dist/leaflet.js', () => { //
        //chargement du cdn de leaflet
        this.map = L.map(element);
        L.tileLayer('https://tile.openstreetmap.fr/osmfr/{z}/{x}/{y}.png', {
          // le lien vers la source des données(mention légale)
          attribution: 'données © <a href="//osm.org/">OpenStreetMap</a> - rendu <a href="//openstreetmap.fr">OSM France</a>',
          minZoom: 1, //définir le zoom mini et maxi
          maxZoom: 20
        }).addTo(this.map);
        //gestion de l'échelle de la carte
        L.control.scale({
          metric: true,
          imperial: false,
          position: 'topright'
        }).addTo(this.map);
        resolve();
      })
    })
  }

  //création des marker
  addMarker(lat, lng, text) { //text est contient tout les info de la fiche : You, etc
    let point = [lat, lng];
    this.bounds.push(point);
    return new LeafletMarker(point, text, this.map);
  }

  center() { //centrer la carte par rapport a tout les point
    this.map.fitBounds(this.bounds, {
      padding: [50, 50]
    });
  }
}
```



Je crée un class LeafletMarker qui traitera la création des markers personnalisés sur la carte.

Dans la classe on crée un constructeur qui se charge d'ajouter les markers sur la carte avec les informations qu'on lui a données (coordonnées GPS, le contenu et la carte). Comme indiqué sur la documentation de leafletjs.

J'y ai placé différentes fonctions pour rendre les markers dynamiques.

```
class LeafletMarker { //class qui contient la gestion des marker
  constructor(point, text, map) {
    this.text = text;
    this.popup = L.popup({ //création de la popup
      autoClose: false,
      closeOnEscapeKey: false,
      closeOnClick: false,
      closeButton: false,
      maxWidth: 400
    })
    .setLatLng(point) //position
    .setContent(text) //texte dans la marker(nom)
    .openOn(map);
  }

  setActive() {
    this.popup.getElement().classList.add('is-active'); //rajoute une la
    //class is-active pour lors du survole le marker devin bleu
  }

  unsetActive() {
    this.popup.getElement().classList.remove('is-active'); //retirer la
    //classe is-active
  }

  addEventListener(event, cb) {
    this.popup.addEventListener('add', () => {
      this.popup.getElement().addEventListener(event, cb);
    })
  }

  setContent(text) {
    this.popup.setContent(text);
    this.popup.getElement().classList.add('is-expanded'); //rajoute une la
    //class is-expanded pour agrandir la popup et afficher les infos de
    //l'utilisateur
    this.popup.update();
  }

  resetContent() {
    this.popup.setContent(this.text);
    this.popup.getElement().classList.remove('is-expanded'); //retirer la
    //classe is-expanded
    this.popup.update();
  }
}
```

Pour finir, Une fonction est créée et réutilise la class leafletMap pour initialiser la carte et ses markers, le tout dans un fonction asynchrone.

Dans cette fonction je crée la Map et j'attends son chargement. Une fois ma Carte chargée, je lui ajoute des markers avec des écouteurs d'évènement qui me permettent d'afficher le marker en bleu ou au survole de la photo lui correspondant, ou encore d'afficher le contenu en cliquant sur la photo.

Puis je centre la carte par rapport au marker.

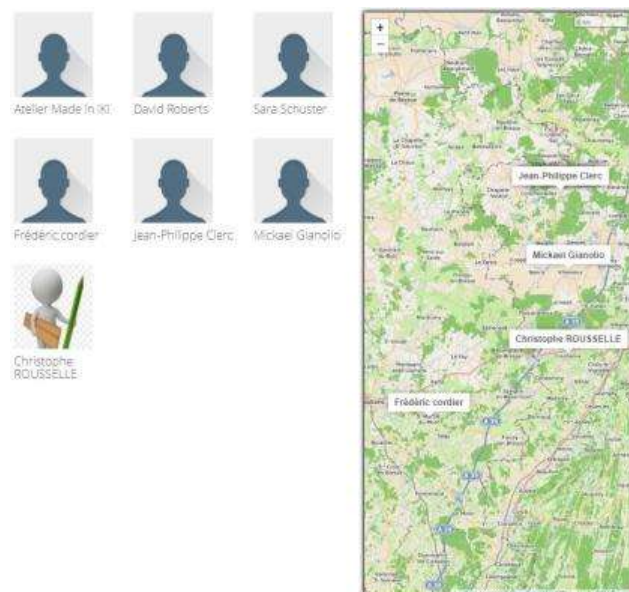


Après cette fonction je m'assure que l'emplacement de ma div est bien sélectionné et je lance la fonction `initMap()`.

```
//initialisation de la carte
const initMap = async function() {
  let map = new LeafletMap();
  let hoverMarker = null; //variable pour sauvegarder le marker sur laquelle
  on est
  let activeMarker = null;
  await map.load($map); //on attend le chargement de la carte
  //par surt  on t l ch ge le polifyle de array.from
  Array.from(document.querySelectorAll('.js-marker')).forEach((item) => {
    let marker = map.addMarker(item.dataset.lat, item.dataset.lng, item.
    dataset.nom);
    item.addEventListener('mouseover', () => { //ajouter au survole de la
    souris sur la liste   gauche
      if (hoverMarker !== null) { //supprimer le dernier marker actif
        hoverMarker.unsetActive();
      }
      marker.setActive();
      hoverMarker = marker;
    })
    item.addEventListener('mouseleave', () => { //retirer le marker actif
      if (hoverMarker !== null) {
        hoverMarker.unsetActive();
      }
    })
    item.addEventListener('click', () => { //au click sur le item afficher
    du contenu
      if (activeMarker !== null) {
        activeMarker.resetContent();
      }
      marker.setContent(item.innerHTML);
      activeMarker = marker;
    })
    item.addEventListener('mouseleave', () => { // enlever l'affichage du
    contenu
      if (activeMarker !== null) {
        activeMarker.resetContent();
      }
    })
  })
  map.center(); //centre la carte sur le marker
}

if (map !== null) {
  initMap();
}
```

Une fois cela fait j'obtiens ce r sultat.







J'ai rajouté une recherche sur le contenu en JavaScript, cela fait disparaître les profils non recherchés.

```
// traitement de la barre de recherche
document.getElementById('search').addEventListener('keyup', function(e) {
    var recherche = this.value.toLowerCase();
    var documents = document.querySelectorAll('.document');

    Array.prototype.forEach.call(documents, function(document) {

        // On a bien trouvé les termes de recherche.
        if (document.innerHTML.toLowerCase().indexOf(recherche) > 1) {
            document.style.display = 'block';
        } else {
            document.style.display = 'none';
        }
    });
});
```

## 5-Personalisation du formulaire de compétence

A la base le formulaire était composé de champs textes comme ci-dessous et je l'ai transformé en cases à cocher.

Paramètres de l'utilisateur

Général Étiquettes

Mots-clé  
Administration, Support, HumHub

Langue  
Français

Fuseau horaire  
UTC-11:00 - Pacific/Pago\_Pago

☐ Masquer le panneau "premiers pas" sur le fil d'actualités

Enregistrer

Pour ce faire j'ai créé une nouvelle table (PostTag).

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
2	skill	varchar(255)	utf8_general_ci		Non	Aucun(e)			Modifier Supprimer Plus



J'ai modifié le fichier formulaires en enlevant les validation dans la fonction rules().

Cette fonction indique le type de données attendues. Comme par exemple une date ou encore un email.

```
namespace humhub\modules\user\models\forms;

use app\models\PostTags;
use app\models\User;
use Yii;
use yii\base\Model;

You, seconds ago | 1 author (fou)
/**
 * Form Model for changing basic account settings
 *
 * @since 0.9
 */
class AccountSettings extends Model
{
    public $tags;
    public $language;
    public $show_introduction_tour;
    public $visibility;
    public $timeZone;

    /**
     * @inheritdoc
     */
    public function rules()
    {
        return [
            [['show_introduction_tour'], 'boolean'],
            [['timeZone'], 'in', 'range' => \DateTimeZone::listIdentifiers()],
            [['language'], 'in', 'range' => array_keys(
                (Yii::$app->i18n->getAllowedLanguages())
            )],
            [['visibility'], 'in', 'range' => [1, 2]],
        ];
    }

    /**
     * @inheritdoc
     */
    public function attributeLabels()
    {
        return [
            'tags' => 'You, a month ago • first commit',
            'tags' => yii::t('UserModule.account', 'compétence'),
            'language' => yii::t('UserModule.account', 'Language'),
            'show_introduction_tour' => yii::t('UserModule.account', 'Hide introduction tour panel on dashboard'),
            'timeZone' => yii::t('UserModule.account', 'TimeZone'),
            'visibility' => yii::t('UserModule.account', 'Profile visibility'),
        ];
    }
}
```

Dans la vue j'ai utilisé la class ArrayHelper que propose Yii. La classe est une assistante au traitement des tableaux dans Yii fournit des méthodes statiques supplémentaires qui vous permettent de traiter les tableaux avec plus d'efficacité. Et la fonction Map() de la class permet de construire une carte à partir d'un tableau multidimensionnel ou d'un tableau d'objets. A cette fonction je lui passe les informations de ma table postTags.

Dans le \$from-Field, je lui indique le champ créer dans mon fichier formulaire et je lui indique de l'afficher sous forme de cases à cocher.

Enfin j'ajoute un bouton qui redirige vers le formulaire de contact en lui passant le lien qui permet d'envoyer un message à l'administrateur du site.

Ci-dessous le code et le résultat final.



```
<?php $form = ActiveForm::begin(['id' => 'basic-settings-form']); ?>

<?php // You, a week ago • gestion des skill, il rest la suppression
$PostTags = ArrayHelper::map($PostTags, 'skill', 'skill');
?>
<?= ($form->field($model, 'tags')->checkboxList($PostTags, ['multiple' =>
'multiple'])); ?>

<p>Si vos compétence ne figure pas dans la liste. Veuillez contacter
l'administrateur du site <button type="button" class="btn btn-primary"
data-action-click="ui.modal.load" data-action-click-url="/index.php?
r=mail%2Fmail%2Fcreate&userGuid=3ef2d8cf-3184-440d-9d90-c23fc76e6cbf">Ici</
button></p>
```

Paramètres de l'utilisateur

Général Étiquettes

skill

☐ Impression 3D ☐ Arduino ☐ CNC

Si vos compétence ne figure pas dans la liste. Veuillez contacter l'administrateur du site [Ici](#)

Langue

Français

Fuseau horaire

UTC+02:00 - Europe/Paris

☐ Masquer le panneau "premiers pas" sur le fil d'actualités

Enregistrer

Il ne reste plus qu'à modifier le traitement du formulaire dans le contrôleur.

Dans mon contrôleur, j'importe toutes les données de ma table PostTags.

```
$PostTags = PostTags::find()->all(); //Retrieve information from the
PostTags table
```

Je modifie ma variable \$PostTags en supprimant les cases non cochées.

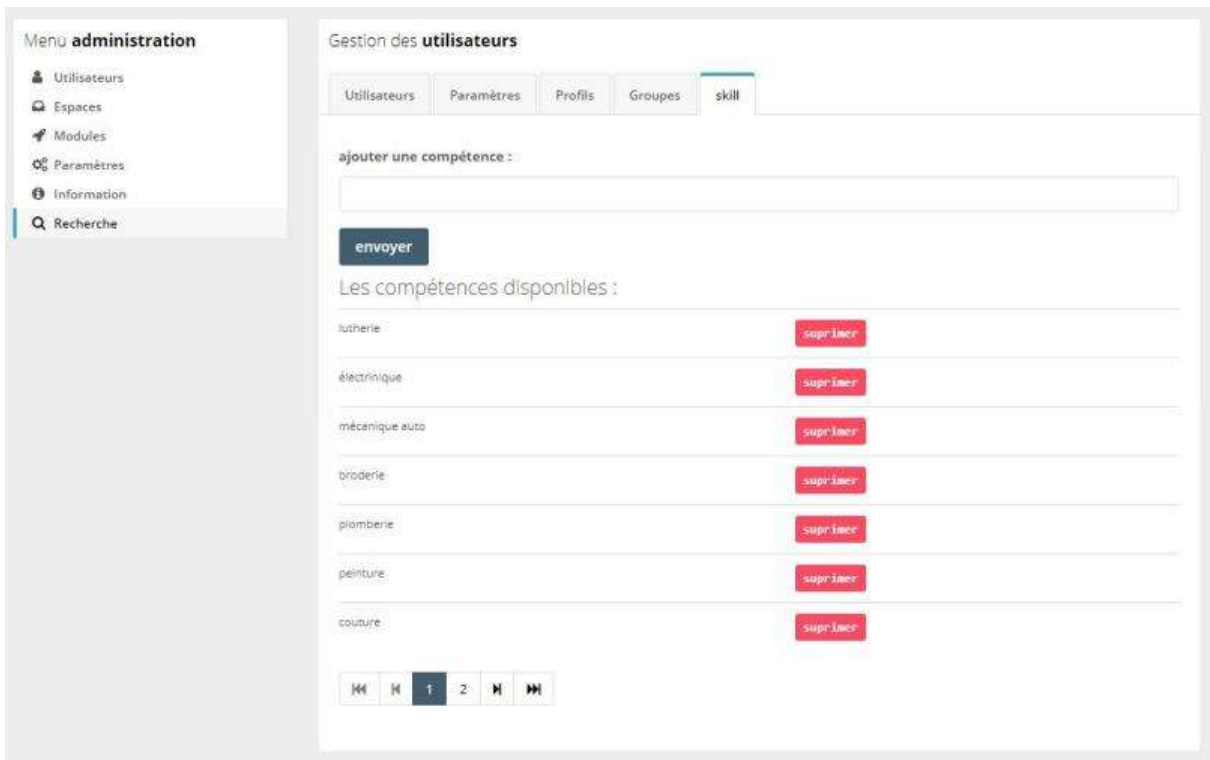
Puis je transforme \$PostTags (un tableau de chaîne de caractère) en chaîne de caractères dont chaque élément est séparé par une virgule. Puis je l'envoie à ma base de données dans le champ tags.

J'ai choisi cette solution, car les données entrées dans le champ tags sont utilisées à un autre endroit de l'application. Ce qui m'a permis de ne pas tout modifier.

```
$tagsliste = $_POST['AccountSettings']['tags'];
$newtags = implode(", ", $tagsliste);
$user->tags = $newtags;
```

Après vérification que les données soient bien transmises à la base de données et qu'elles apparaissent bien à l'endroit où elles sont utilisées dans l'application.

Je crée l'interface d'ajoute de compétences dans la partie administration.



Dans le dossier widget et le fichier UserMenu.php qui permet d'afficher les onglets dans le menu gestion des utilisateurs. J'ai rajouté le code ci-dessous pour afficher un nouvel onglet dans la vue.

```
$this->addEntry(new MenuLink([
    'label' => Yii::t('AdminModule.user', 'skill'),
    'url' => ['/admin/skill'],
    'sortOrder' => 500,
    'isActive' => MenuLink::isActiveState('admin', 'skill'),
]));
```





Et j'ai créé mon fichier formulaire, avec un fonction de traitement signup().

```
class PostTagsFrom extends Model
{
    public $skill;

    /**
     * @inheritdoc
     */
    public function rules()
    {
        return [
            ['skill', 'string'],
            ['skill', 'exist', 'targetClass' => PostTags::class,
            'targetAttribute' => 'skill', ],
        ];
    }

    /**
     * @inheritdoc
     */
    public function attributeLabels()
    {
        return [
            'skill' => 'ajouter une compétence : '
        ];
    }

    /**
     * signs postTags up
     */
    public function signup()
    {
        $skill= new PostTags();
        $skill->skill=$this->skill;
        if($skill->validate())
        {
            $skill->save();
            return $skill;
        }
    }
}
```

Après je traite mes données et je les retourne dans la vue pour l'afficher sous forme de CRUD.

```
/**
 * Shows overview of all
 *
 * @return void
 */
public function actionIndex()
{
    //ajout d'une compétence
    $model = new PostTagsFrom;
    if ($model->load(Yii::$app->request->post())) {
        $model->signup();
        header("Location: /index.php?r=admin2Fskill");
        exit();
    }

    //affichage des donnés avec pagination
    $query = PostTags::find();
    $pagination = new Pagination(['totalCount' => $query->count(),
    'defaultPageSize' => 7]);

    $articles = $query->offset($pagination->offset)
    ->limit($pagination->limit)
    ->orderBy(['id'=>SORT_DESC])
    ->all();

    return $this->render('index', [
        'skills' => $articles,
        'models' => $model,
        'pagination' => $pagination
    ]);
}

public function actionDelete($id)
{
    $skill = PostTags::findOne($id);
    if (empty($skill)) {
        return;
    }

    $skill->delete();
    if ($skill) {
        Yii::$app->getSession()->setFlash('message', 'Skill supprimé');
        return $this->redirect(['/admin/skill/index']);
    }
}
```

```
<?php $this->beginContent('modules/views/layouts/main.php') ?>
<div class="panel panel-default">
    <div class="panel-heading">
        <?php Yii::t('AdminModule.user', 'strong:User/strong: administration');
    </div>
    <div class="panel-body">
        <div id="edit-profile-field-root" class="panel-body">
            <?php
            if (Yii::$app->session->hasFlash('success')) {
                echo Yii::$app->session->getFlash('success');
            }

            <?php $form = ActiveForm::begin();
            <?php $form->field($model, 'skill');

            <?php Html::submitButton('envoyer', ['class' => 'btn btn-primary']);
        </div>

        <?php ActiveForm::end();
        <div class="table table-bordered">
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>Id</th>
                        <th>Skill</th>
                        <th>Action</th>
                    </tr>
                </thead>
                <tbody>
                    <?php if (count($skills) > 0) : ?>
                        <?php foreach ($skills as $skill) : ?>
                            <tr>
                                <td><?php echo $skill->id;</td>
                                <td><?php echo $skill->skill;</td>
                                <td>
                                    <button type="button" class="btn btn-danger">Supprimer</button>
                                    <button type="button" class="btn btn-primary">Modifier</button>
                                </td>
                            </tr>
                        </foreach>
                    </if>
                    <tr>
                        <td colspan="3">Aucune compétence d'enregistrer</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
</div>
```

Après vérification, mes modifications sont validées par les adhérents du fablab.

## 6) Présentation du jeu d'essai élaboré de la fonctionnalité

### 1-modification du formulaire vue précédemment

Après la mise en place du traitement du formulaire j'ai vérifié :

- Que dans la base de données, le champ tags a bien été rempli avec la bonne valeur et que chaque mot soit séparé pas une virgule.

Options	id	uuid	status	username	email	tags	auth_mode	langue
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	1	ef74d7cc-b662-4f7b-8264-8ac1169b186	1	admin	f.cordier@codeur.online	arduino, raspberry, impression 3d, modélisation 3D...	local	fr
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	4	49ba666c-5382-4f81-ac15-660242ffdb81	1	papy	papy@contact.fr	arduino, raspberry	local	fr
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	5	d46afa58-b658-4a3b-acac-fab575d537a	1	mamy	mamy@contact.fr	Arduino, Raspberry pi, impression 3d, électronique...	local	fr

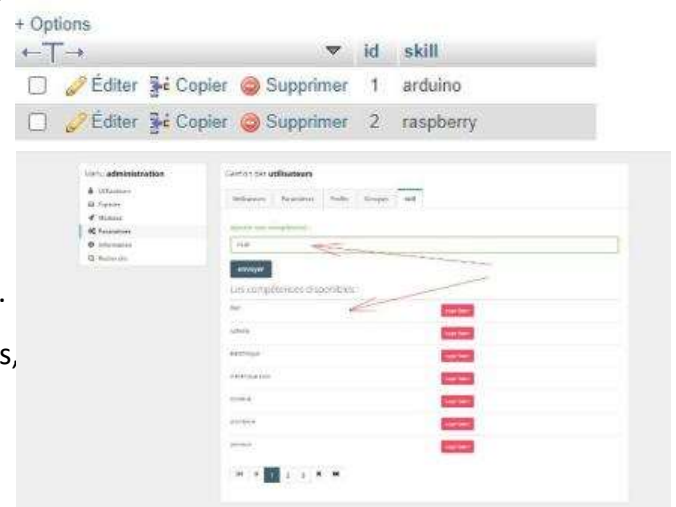
- Est dans l'interface qui affiche les infos de tags.



### 2-ajout dans la partie admin de la gestion des compétences

Après la mise en place du traitement du formulaire j'ai vérifié :

- Que dans la base de données le champ skill a bien été rempli avec la bonne valeur.
- Est qu'il apparaisse bien dans mon CRUD. J'ai fait en sorte que les données ajoutées, en dernier soit en haut du tableau.





## 7) Description de la veille

### Veille sur les bonnes pratiques de sécurité sous le framework Yii2

Source : [www.yiiframework.com](http://www.yiiframework.com)

#### 1-Filtrer les entrées et échapper les sorties

-Filtrer les entrées : Cela signifie de que l'on ne doit pas considérer les entrées comme sûres et que l'on doit toujours vérifier les informations. Par exemple on doit toujours vérifier que ce qui a été rentré dans le formulaire corresponde à ce qui est attendu. On ne doit pas pouvoir rentrer une date dans un champ où l'on attend un email par exemple.

-Echapper les sorties : Echapper les sorties signifie que selon si l'on travaille en HTML ou JavaScript etc... L'on doit utiliser les caractères d'échappement, comme par exemple la barre oblique que l'on place avant le guillemet simple. Plus d'info sur [https://fr.wikipedia.org/wiki/Caract%C3%A8re\\_d%27%C3%A9chappement](https://fr.wikipedia.org/wiki/Caract%C3%A8re_d%27%C3%A9chappement)

#### 2-Eviter les injections SQL

Yii utilise la classe Active Record qui permet d'utiliser les instructions PDO de préparation en interne.

Il est possible également de préparer des requêtes brutes. Il est préférable de créer les instructions de préparation.

#### 3-éviter le XSS

Le XSS ou scriptage inter site se produit lorsque la sortie n'est pas échappée correctement lors de l'envoi de code HTML au navigateur.

Pour éviter cela, l'on peut utiliser `\yii\helpers\Html::encode($variable)` pour des transmettre des données texte.

Ou `\yii\helpers\HtmlPurifier::process($variable)` pour transmettre des données sous forme de code HTML. (HtmlPurifier est très lourd, il faut donc éviter la mise en cache).



### 4-Eviter le CSRF

La CSRF est une abréviation de cross-site request forgery (falsification de requête inter sites). L'idée est que beaucoup d'applications partent du principe que les requêtes provenant d'un navigateur sont fabriquées par l'utilisateur lui-même. Cela peut être faux.

Par exemple l'injection dans une image d'un lien vers un page. Surtout si cette page utilise de simple requête GET. Mais même en passant par des requête POST l'attaque peut être fait en placent un peu le JavaScript à la place de la balise `<img>`, ce qui permet d'envoyer des requêtes POST sur cette URL.

Pour éviter cela GET ne doit pas charger l'état de l'application.

Yii a mis en place une protection active contre le CSRF. Avec la fonction `noCsrfValidationRoutes`.

### 5- Éviter l'exposition de fichiers

Par défaut, la racine du serveur web est censé pointer sur le dossier web, là où se trouve le fichier `index.php`. Dans le cas d'un hébergement partagé, il peut être impossible de réaliser cela et vous pouvez vous retrouver avec tout le code, configurations et journaux sous la racine du serveur web.

Si c'est le cas, n'oubliez pas de refuser l'accès à tout sauf au dossier web. Si cela n'est pas possible, envisagez d'héberger votre application ailleurs.

### 6- Éviter les informations et des outils de débogage en mode production

Bien enlever tous les outils de débogage que propose Yii avant de lancer le site en production.

### 7- Utilisation de connexions sécurisées

Yii fournit des fonctionnalités qui comptent sur les témoins de connexion et/ou sur les sessions PHP. Cela peut créer des vulnérabilités dans le cas où votre connexion est compromise.

Le risque est réduit si l'application utilise une connexion sécurisée via TLS.



## 8) Description d'une situation de travail ayant nécessité une recherche

Pendant ce stage, j'ai dû m'adapter à un nouveau Framework et à l'application qui l'utilise. J'ai passé mon temps à rechercher comment utiliser Yii2 et comment les concepteurs de l'application web Humhub l'avaient pensée et structurée.

Voici la recherche ou j'ai eu le plus de mal à trouver des informations et de les comprendre.

### Problématique

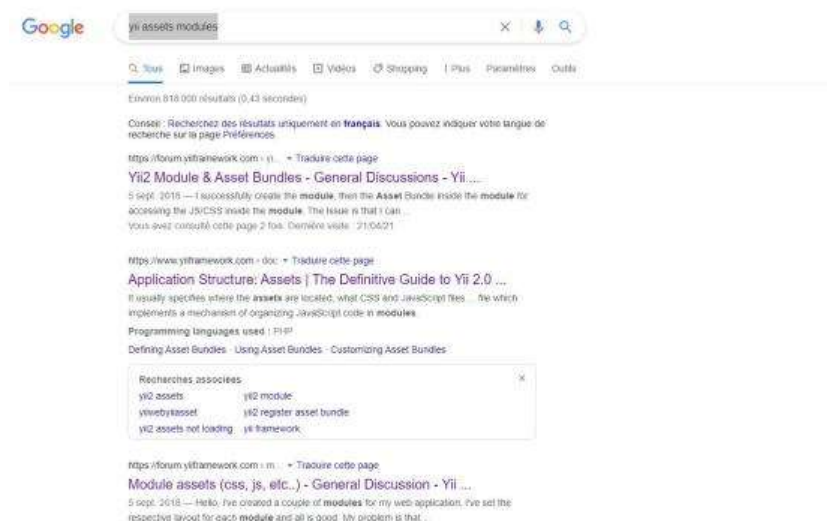
Dans un premier temps j'ai développé mon application sans l'intégrer au Framework.

Cette application a besoin de fichiers séparés pour les dépendances CSS et Javascript et Bootstrap pour éviter un dossier unique lourd à lire et à entretenir.

Le Framework Yii a une gestion des dépendances particulière que j'ai décrit précédemment dans la partie réalisation et je vais vous le redécrire ci-dessous.

### Recherche sur le web

Pour trouver des informations sur la façon d'utiliser mes ressources. J'ai utilisé le moteur de recherche Google en tapant les mots suivants : yii assets modules. Ces sont avec ces mots que j'ai trouvés les informations dont j'avais besoin.







## 9) Extrait du site anglophone, utilisé dans le cadre de la recherche décrite précédemment

### Extrait d'un site anglophone

Source : <https://docs.humhub.org/docs/develop/testing>

## Testing

### Getting Started#

Testing in Humhub/Yii extends the paradigm of unit testing by functional and acceptance tests.

**Acceptance tests** simulate actual user actions on a browser. By means of acceptance tests you can test your backend code as well as your javascript frontend, potentially on different browsers. The downside of acceptance tests is the longer execution time, and a more complex implementation. Implement acceptance tests for general UI tests, ideally access all views and test Javascript based views.

**Functional tests** are similar to acceptance tests, with the difference that functional tests do not run on an actual browser and do not execute any javascript. Functional tests allow easy testing of customized HTTP requests, forms and also allow direct access of application logic as database or the Yii application context. This can be handy if you require a specific application state as specific settings for a test. Write functional tests in order to test your controllers, forms and controller access for different configurations settings.

**Unit tests** are ideal for [white box testing](#) and is the fastest way of writing low level tests for specific classes or components. Implement unit tests for testing single components and classes.

HumHub uses [Codeception](#) as testing framework.

*ATTENTION: Some of the test libraries are developed for use with PHP 7 only*

Information about how to write tests with codeception are available here:

- [Codeception Introduction](#)
- [Yii Testing Guide](#)



## Traduction

# Essai

Commençons.

Les essais dans Humhub/Yii étendant le paradigme des tests unitaire par de test de fonctionnement et d'acceptation.

**Teste d'acceptation**, Simulent les actions que font les utilisateurs sur le navigateur. Grâce à des tests d'acceptation, on peut tester son code backend ainsi que le code frontend javascript, sur différents navigateurs. Le mauvais côté du test d'acceptation, est le temps d'exécution plus long et une implémentation plus complexe. Implémentez des tests d'acceptation pour les tests généraux UI, accédez idéalement à toutes les vues et testez les vues basées sur Javascript.

**Teste de fonctionnalité**, les tests de fonctionnalité son similaire au teste d'acceptation, la différence est que le teste de fonctionnement ne s'exécute pas sur un navigateur réel et n'exécute pas le javascript. Les tests fonctionnels permettent de tester facilement les demandes HTTP personnalisées, les formulaires et permettent également un accès direct à la logique de l'application comme la base de données ou le contexte de l'application Yii. Cela peut être pratique si vous avez besoin d'un état spécifique de l'application comme paramètres spécifiques pour un test. Écrivez des tests fonctionnels afin de tester vos contrôleurs, formulaires et accès aux contrôleurs pour différents paramètres de configuration.

**Les tests unitaires**, les tests unitaires sont idéaux pour les tests en boîte blanche et constituent le moyen le plus rapide d'écrire des tests de bas niveau pour des classes ou des composants spécifiques. Mettez en œuvre des tests unitaires pour tester des composants et des classes uniques.

HumHub utilise Codeception comme test du framwork.

ATTENTION : Certaines des bibliothèques de test sont développées pour être utilisées seulement avec PHP 7.

Des informations sur la façon d'écrire des tests avec Codeception sont disponibles ici :

- Introduction à Codeception
- Guide de test Yii