Cahier des charges plateforme de petites annonces

Au cours de ces 10 prochains jours, vous allez devoir développer une plateforme de petites annonces avec le framework Laravel 5.

Le site doit permettre une fois inscrit de:

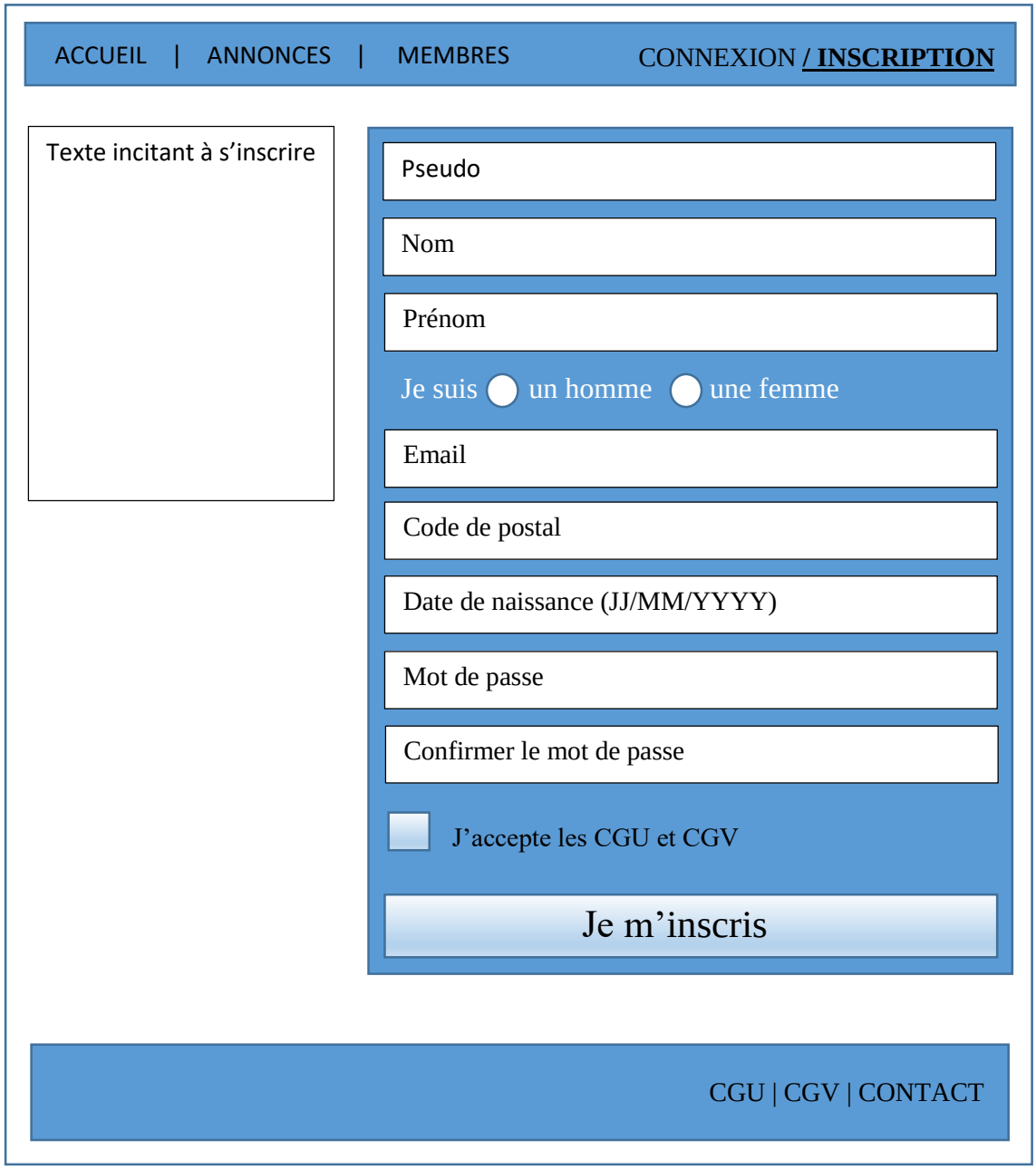
- Publier des annonces de recherche d'emplois.
- Publier des annonces de recherche de compétences.
- Evaluer les autres membres sur 5 tout en leur laissant un commentaire.

Vous demeurez libre du choix de la charte graphique et pouvez si vous le souhaitez travailler avec des frameworks css tel que bootstrap.

Pour une meilleure organisation de travail il est recommandé de suivre les étapes de réalisation suivantes :

1) Création du template parent en 2 colonnes (sidebar + main).
2) Formulaire d'inscription
3) Formulaire d'édition de son profil
4) Annuaire des membres
5) Formulaire de création et édition des annonces
6) Page d'une annonce
7) Annuaire des petites annonces
8) Profil d'un membre
9) Module de notation

Page inscription : ***(localhost/plateform/public/index.php/register)***



Champs de la table users :

> ***id, name, firstname, lastname, description, gender, email, zipcode, city, lat, lng, rate, birthday,password, remember_token, avatar, created_at, updated_at***

Le champ gender :

Le champ gender contient m ou f (m = male , f = female) .

Le champ name :

Doit faire 5 caractères mini, ne doit contenir que des caractères alphanumériques et être unique.

Le champ birthday :
Il faudra vérifier que l'âge de l'utilisateur est supérieur à 18.

Pour se faire vous pouvez intégrer cette règle de validation dans la méthode validator de la classe UserController : 'birthday' => 'required|date|before:-18 years'

Le champ de formulaire birthday peut être générer via un input de type date mais non compatible avec mozilla.

L'autre alternative consiste à générer 3 champs select (un pour les jour, un pour le mois et un 3ème pour l'année. Il faudra ensuite combiner ces 3 valeurs pour reconstituer la date d'anniversaire au format AAAA-MM-JJ

La confirmation de mot de passe

La règle de validation de ce champs est déjà mise en place dans le validateur du champ password:

```
return Validator::make($data, [
    'name' => 'required|min:3|max:25|unique:users',
    'email' => 'required|email|max:255|unique:users',
    'password' => 'required|min:6|confirmed',
    // …
]);
```

La règle de validation « confirmed » va vérifier que la valeur du champ password_confirmation est la même que celle du champ password. Si le nom du champ associé à cette règle était motdepasse alors la valeur avec laquelle il aurait été comparé aurait été celle du champ motdepasse_confirmation.

Les champs lat et lng :

La latitude et longitude seront récupérées en fonction du code postal grâce à l'api de google maps. Il faudra donc extraire au moment du traitement de l'envoi du formulaire d'inscription les informations de la page http://maps.googleapis.com/maps/api/geocode/json?address={{ code postal}}%20france via la fonction http://php.net/manual/fr/function.file-get-contents.php ; et le convertir en tableau associatif avec la fonction http://php.net/manual/fr/function.json-decode.php pour en extraire les données (de la même manière qu'en TP sur l'api Google map en JS).

Exemple de la marche à suivre :

Supposons que le contenu renvoyé par la page http://site.com/api.json soit celui-ci:

```
{ "info1" : "Foo",
  "info2" : "Bar",
  "info3" : ["hello","word"]
 }
```

Si l'on souhaite récupérer le contenu de cette page en php afin d'en extraire les données qui nous intéresse voici la marche à suivre :

```
<?php $json = file_get_contents('http://site.com/api.json');

/*

$json contiendra l'intégralité du contenu renvoyé par la page http://site.com/api.json :
{ "info1" : "Foo",
  "info2" : "Bar",
  "info3" : ["hello","word"]
 }

*/

$data = json_decode($json,true);  // On converti l'objet json en tableau associatif interprétable par php

/*

$data contiendra ce tableau associatif :
[ "info1" => "foo",
  "info2" => "bar",
  "info3" => ["hello","word"]
]
*/
```

Si je veux afficher "hello" (première élément du tableau de la propriété "info3"):

```
$hello = $data["info3"][0] ;  // contient « hello »
```

Le champ city : il sera récupérer en même temps que la latitude et longitude

```
$data['results'][0]['address_components'][1]['long_name']; // permet de cibler la ville
```

Le champ rate : il s'agira de la note moyenne reçu par l'utilisateur

Editer mon profil : **(localhost/platform/public/index.php/user/edit)**

*La photo de profil sera sauvegardée dans le dossier ressources/assets/images/avatars/ en 3 formats :

- Version originale. Nom du fichier : **{{ id user }}_nom_image.jpg**
- Version « découpée » en 150 x 150 px. Nom du fichier : **thumb_{{ id user }}_nom_image.jpg**
- Version moyenne : 800px de large nom du fichier : **medium_{{ id user }}_nom_image.jpg**

Le nom du fichier sera enregistré dans le champ avatar de la table users.

/!\ N'oubliez pas de définir l'attribut **enctype="multipart/form-data"** dans le formulaire

Lien utile pour l'upload des fichiers en php : http://php.net/manual/fr/features.file-upload.post-method.php

Pour la gestion de l'upload avec Laravel : https://laravel.com/docs/5.2/requests#files

Dans la méthode update de la classe UserController utilisez ce code pour stocker la version originale de l'avatar dans le dossier public/assets/images (à créer):

```
$uploadDir = __DIR__.'/../../../public/assets/images/';

$filename = Auth::Id().'_'.$request->file('avatar')->getClientOriginalName();

$request->file('avatar')->move($uploadDir,$filename);
```

Pour le redimensionnement et cropping des images vous pouvez installer la librairie suivante : http://image.intervention.io/getting_started/installation

Résoudre le problème de la contrainte de champs email et name unique

Lorsque vous allez tester la validation de votre formulaire, vous aurez systématiquement cette erreur :

- Le champ email doit être unique.
- Le champ name doit être unique.

Il faut donc trouver un moyen de dire à Laravel d'ignorer l'entrée de la table users concernant l'utilisateur cherchant à modifier ses informations.

La solution à cette problématique se trouve sur cette page : https://laravel.com/docs/5.2/validation#rule-unique et précisément au chapitre :

« **Forcing A Unique Rule To Ignore A Given ID** »

A vous de la mettre en application

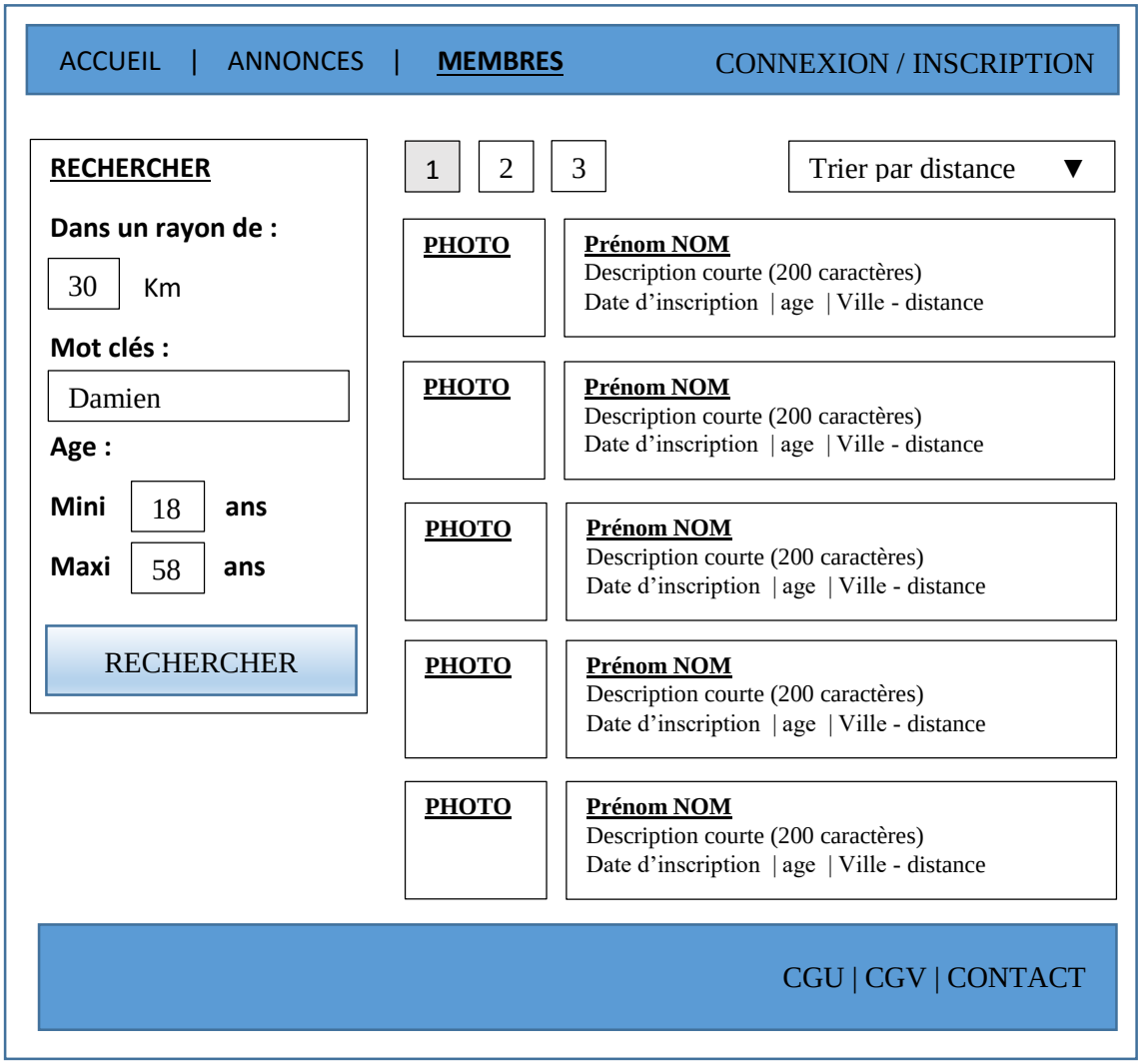
Formulaire éditer mon mot de passe (sidebar) :

C'est un formulaire indépendant du formulaire d'édition des informations générales.

Avant de valider le nouveau mot de passe il faut vérifier que le champ ancien mot de passe est bon.

Pour se faire il faut crypter le mot de passe en claire avec la fonction bcrypt (helper fourni par laravel pour crypter les mots de passe), puis la comparer avec celle de l'utilisateur.

Annuaire des membres : (localhost/platform/public/index.php/members)



La photo affichée sera celle au format 150 x 150 px

Il faut pouvoir trier les utilisateurs par âge croissant, date d'inscription croissante, note moyenne décroissante et distance décroissante.

Exemple : localhost/platform/public/index.php/members?orderBy=age triera les membres par age croissant.

Au sujet de la pagination :

On affichera 5 utilisateurs par page. La pagination pourra s'effectuer aisément grâce aux méthodes **paginate** et **links** de la laravel : https://laravel.com/docs/5.2/pagination

Formulaire de publication / modification d'une annonce :

Url création d'annonce : localhost/platform/public/index.php/advert/create
Url édition d'annonce : localhost/platform/public/index.php/advert/{id}/edit

Champs de la table adverts :

id, title, content, search, type, category_id, user_id, hourly_wage, created_at, updated_at

Nb : Le champ search est un booléen qui sera égal à 0 si l'utilisateur indique Je propose et à 1 s'il indique je recherche.

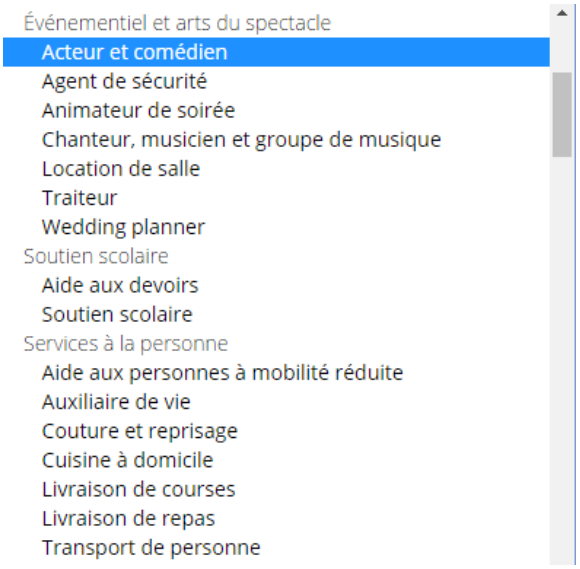
Le champ hourly_wage correspond au taux horaire (tarif).

Le champs type contiendra les valeurs JOB / STAGE / ITERIM / CDD / CDI ou PRESTA

Le champ select pour les catégories sera généré dynamiquement via la base de données (table categories). Vous pouvez mettre en place cette table en important simplement ce fichier SQL : http://conception.website/3wa/PHP/Laravel/categories.sql

Les catégories devront être regroupées par selon leur catégorie parent et par ordre alphabétique. Une catégorie parent ne pourra pas être sélectionnée :

Champ select du formulaire permettant de sélectionner une catégorie




```
<select name="category" placeholder="Choisissez un service">

<optgroup label="Culture et loisirs">

<option value="55">Accordage d'instruments</option>
<option value="164">Association</option>
<option value="52">Cameraman et photographe</option>
<option value="127">Wedding planner</option>

</optgroup>
<optgroup label="Soutien scolaire">

<option value="109">Aide aux devoirs</option>
<option value="110">Soutien scolaire</option>

</optgroup>

...

</select>
```

NB : La valeur de l'attribut value est l'id de la catégorie.

Dans le model Category créer une méthode children pour récupérer les enfants d'une catégorie à partir de la clé étrangère parent_id, et définissez-y une relation **one to many** (une catégorie parent peut avoir plusieurs catégories enfants, mais une catégorie enfant ne peut avoir qu'un parent) :

http://laravel.com/docs/5.2/eloquent-relationships#one-to-many

Dans la classe AdvertController créer une méthode getCategoryFields qui se chargera de renvoyer le code html ci-dessus à partir de la classe Category. Vous pourrez ainsi la réutiliser pour la partie édition de l'annonce.

Astuce pour générer le code html du champ select des catégories :

Faire une première boucle pour lister les catégories parents (c'est-à-dire celles ayant un champ parent_id nul) pour générer les labels :

```
<optgroup label="Culture et loisirs">
```

A l'intérieur de cette boucle faite une autre boucle pour lister les options (catégories enfants) :

```
<option value="109">Aide aux devoirs</option>
```

/!\ Une fois tous les enfants d'une catégorie listés pensez à bien refermer la balise optgroup : </optgroup>

Page d'une annonce : ***(localhost/plateform/public/index.php/advert/{id})***

ACCUEIL | ANNONCES | MEMBRES CONNEXION / INSCRIPTION

Auteur de l'annonce :

PHOTO

Prénom NOM
Age
Sexe
Ville + distance en km

Note : ★ ★ ★ ★ ★ (36)

EDITER

SUPPRIMER

TITRE DE L'ANNONCE

Description longue

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

Catégorie / TYPE (CDD etc.) / RECHERCHE ou PROPOSE

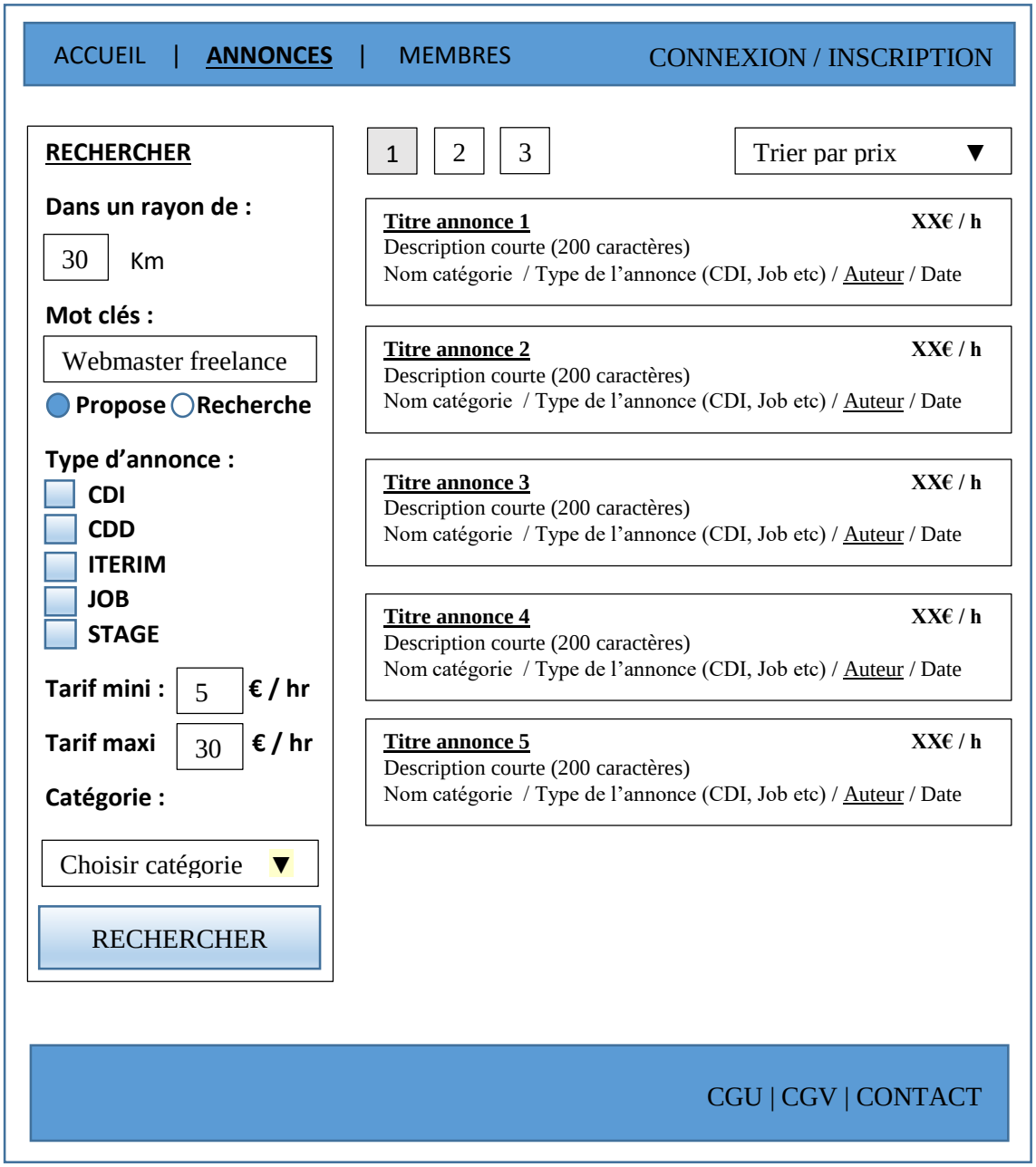
Date de publication

Tarif : XX€ / hr

CGU | CGV | CONTACT

NB : Seul l'auteur d'une annonce doit pouvoir l'éditer ou la supprimer.

Page des annonces : (localhost/plateform/public/index.php/adverts)



Il doit y avoir 5 annonces par page.

Au sujet du moteur de recherche des annonces :

La recherche dans un rayon de X km s'effectuera grâce aux coordonnées GPS de l'auteur de l'annonce (lat et lng) et celle de l'utilisateur qui effectue la recherche.

Pour développer cette fonctionnalité nous utiliseront cette extension Laravel : https://packagist.org/packages/jackpopp/geodistance

Il faut pouvoir trier les annonces par prix décroissant, distance croissante ou date décroissante.

Le champ select pour les catégories sera généré dynamiquement via la base de données (table categories).

Profil d'un membre : (localhost/platform/public/index.php/u/{i})

ACCUEIL | ANNONCES | MEMBRES CONNEXION / INSCRIPTION

PHOTO

Prénom NOM
Age
Sexe
Ville + distance en km

Note : ★ ★ ★ ★ ★ (36)

☆☆☆☆☆

Ecrire commentaire…

Noter

Notes reçus :

★ ★ ★ ★ ★
"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod »
Auteur / Date

_____

★ ★ ★ ★
"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod »
Auteur / Date

PROFIL DE {{ firstname }} {{ lastname }}

Description longue

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

ANNONCES PUBLIEES (4) :

Titre annonce 1 **XX€ / h**
Description courte (200 caractères)
Nom catégorie / Type de l'annonce (CDI, Job etc) / Auteur / Date

Titre annonce 2 **XX€ / h**
Description courte (200 caractères)
Nom catégorie / Type de l'annonce (CDI, Job etc) / Auteur / Date

Titre annonce 3 **XX€ / h**
Description courte (200 caractères)
Nom catégorie / Type de l'annonce (CDI, Job etc) / Auteur / Date

Titre annonce 4 **XX€ / h**
Description courte (200 caractères)
Nom catégorie / Type de l'annonce (CDI, Job etc) / Auteur / Date

CGU | CGV | CONTACT