

Remerciements

Je tiens à adresser mes remerciements au centre de formation DORANCO, pour la qualité de la formation et le soutien de l'équipe administrative.

Je saisis cette occasion pour adresser mes profonds remerciements à Mr Yassine AABIDOUCE pour tous ses conseils durant toute la formation.

Je tiens également à remercier les formateurs qui ont tous été à la hauteur.

je voudrais remercier également mes camarades de promotion.

Pour finir, un grand MERCI à mon père, ma mère, ma femme, mes enfants, mon frère et ma sœur qui m'ont apporté leur soutien moral et intellectuel tout au long de ma formation.

1 Table des matières

1	Table des matières.....	2
2	À PROPOS.....	6
3	PRESENTATION DU PROJET.....	8
3.1	Présentation de l'entreprise et son projet.....	9
3.2	Les besoins fonctionnels	9
3.2.1	Espace membre et administrateur.....	9
3.2.2	Interface d'administration « Back-office »	10
3.2.3	Formulaire de dépôt d'annonce.....	10
3.2.4	Formulaires de contact	10
3.2.5	Barre de recherche.....	10
3.3	Les cibles de l'application.....	10
3.4	Benchmark concurrentiel	10
3.4.1	objets-trouve.com.....	11
3.4.2	cpasperdu.com.....	12
3.5	Arborescence de l'application	13
3.5.1	Front-end «Plan du site »	13
3.5.2	Back-end « Organisation des fichiers »	16
3.6	Wireframe	17
3.7	Budgétisation du projet	24
3.8	Calendrier prévisionnel	24
4	UML	25
4.1	Diagramme de cas d'utilisation	26
4.2	Diagramme de classes.....	27
5	DÉVELOPPEMENT DE L'APPLICATION.....	28
5.1	Langages et Framework	29
5.2	Installation du framework Symfony	30
5.3	L'arborescence des fichiers	30
5.3.1	Le dossier « bin ».....	31
5.3.2	Le dossier « config »	31
5.3.3	Le dossier « public »	31
5.3.4	Le dossier « src »	31
5.4	La Base de données.....	32
5.4.1	Paramétrage.....	32
5.4.2	Création de la base de données	33
5.4.3	Les tables de la base de données	33
5.4.4	Diagramme EA « entité-association » de la base de données	34
5.4.5	Création des tables.....	34
5.4.6	Insertion des données dans la base	35

5.5	Formulaire d'inscription.....	37
5.5.1	Création du formulaire.....	37
5.5.2	Validation du formulaire et la protection contre les injections SQL	39
5.5.3	Le Controller.....	42
5.6	Formulaire de contact	46
5.7	La Sécurité	49
5.7.1	L'authentification et les autorisations.....	49
5.7.2	La protection contre les injections SQL.....	50
5.7.3	Le Cryptage des mots de passe	50
5.7.4	Anti-Spam.....	50
5.7.5	HTTPS « Protocole de transfert hypertexte sécurisé »	50
5.8	Les Cookies et le respect de la loi	50
5.9	Interface d'administration « Back-Office »	52
5.10	La page d'erreur 404	54
5.11	Intégrer les données dans une vue.....	55
5.11.1	Intégration des données.....	55
5.11.2	Pagination.....	56
5.11.3	Affichage des annonces active	57
6	LE RÉFÉRENCEMENT	61
6.1	Le Référencement Naturel « SEO ».....	62
6.1.1	La balise « title »	62
6.1.2	La balise « meta description ».....	62
6.1.3	Les balises titre Hn	62
6.1.4	L'attribut <alt>	63
6.1.5	Responsive - Mobile friendly.....	63
6.1.6	Le temps de chargement.....	63
6.1.7	La duplication de contenu « duplicate content ».....	64
6.1.8	L'indexation.....	64
6.1.9	Sitemap	64
7	DESIGN	66
7.1	Charte graphique	67
7.1.1	Le logo	67
7.1.2	Les couleurs.....	68
7.1.3	La typographie.....	69
7.1.4	Les icônes	69
7.1.5	Les illustrations	69
7.1.6	Les boutons	69
7.2	Responsive Web Design	70
7.3	Le rendu final de l'application	72
8	LES MENTIONS OBLIGATOIRES.....	80
8.1	Les obligations à respecter.....	81
8.1.1	Conditions générales d'utilisation (CGU) »	81
8.1.2	Politique de Confidentialité	81
8.1.3	Règlement Général sur la Protection des Données (RGPD) »	81
8.1.4	Cookies	81
9	LA MISE EN LIGNE	82

9.1	La mise en ligne de l'application.....	83
9.1.1	Le choix de l'hébergeur	83
9.1.2	Paramétrage « cPanel »	83
9.1.3	Test et Préparation des fichiers en local	83
9.1.4	Le transfert.....	84
10	OUTILS D'ANALYSE	85
10.1	Google Analytics	86
11	CONCLUSION	87

CAHIER DES CHARGES



IFOUNDIT

Projet de création de l'application web

www.ifoundit.fr

2 À PROPOS...



Je suis Noredine BENYAMINA, j'ai débuté ma carrière dans l'informatique depuis le plus jeune âge, par la réparation des ordinateurs et la résolution des problèmes liés au fonctionnement, aussitôt j'ai suivi une formation de technicien de maintenance informatique et réseaux afin de me professionnaliser dans ce domaine.

Après quelques années d'expérience, j'ai créé la société « Nordinateur », localisé à Boulogne-Billancourt, spécialisé dans la maintenance informatique et le câblage des réseaux.

Passionné par le développement web et avec des notions en Html et CSS, je me suis intéressé à la création des sites internet, des sites vitrines et des site e-commerces avec le CMS Prestashop où j'ai réalisé plusieurs projets, entre-autres « lacartouche.fr » un site e-commerce spécialisé dans la vente de cartouches d'encre.

Fort de mon expérience dans le domaine informatique depuis plus de 20 ans, aussi passionné par le développement web et la création des sites internet, autodidacte je me suis orienté dans ce domaine et afin d'approfondir mes compétences en suivant une formation de développeur web et web mobile au sein de l'école Doranco qui a duré 7 mois dont 1 mois de stage en entreprise en tant que développeur.



3 PRESENTATION DU PROJET

3.1 Présentation de l'entreprise et son projet

Il a été décidé de mettre en place une application web pour l'entreprise Normus.

L'entreprise Normus est immatriculée au RCS de Nanterre sous le numéro 812100295, dont le siège social est situé au 3 rue Pasteur 92210 St-Cloud.

Elle a pour objectif d'offrir aux habitants de la ville de St-Cloud et les villes voisines, un service en ligne d'objets trouvés sous forme d'annonces, qui se présentent comme suit :

- Signalement des objets trouvés.
- Signalement des objets perdus.

Actuellement la ville de St-Cloud ne possède aucun moyen afin de signaler les objets trouvés.

L'entreprise Normus a décidé d'offrir ce service gratuitement pour les habitants dans un premier temps afin de se faire connaître, puis mettre en place un service de donation et un service de publicité payant pour les professionnels.

Elle souhaite à long terme collaborer avec :

- La mairie de St-Cloud.
- La gare de St-Cloud.
- Les mairies des villes voisines ainsi que les gares.

Présentation de l'application

« ifoundit » qui signifie en Français « **je l'ai trouvé** », c'est le nom qui a été choisi pour cette application web avec l'extension « .fr ».

L'application web www.ifoundit.fr a pour but la mise en ligne des annonces d'objets trouvés pour la ville de St-Cloud et les villes voisines.

Elle permettra aux utilisateurs de consulter, déposer et contacter l'annonceur.

3.2 Les besoins fonctionnels

3.2.1 Espace membre et administrateur

- **Utilisateur non inscrit « invité »**

Un utilisateur non inscrit sur www.ifoundit.fr pourra consulter les annonces mais ne pourra pas déposer ou voir les coordonnées d'un annonceur, afin de pouvoir déposer une annonce ou contacter l'annonceur, ce dernier doit posséder un compte client en s'inscrivant avec une adresse e-mail via le formulaire d'inscription.

- **Utilisateur inscrit « membre »**

Un utilisateur aura la possibilité de créer son propre espace membre afin de déposer, modifier, supprimer une annonce, il pourra aussi modifier ses informations personnelles et supprimer son compte définitivement.

- **Utilisateur Administrateur « admin »**

Un administrateur aura les mêmes droit qu'un utilisateur inscrit mais en plus un accès back-office dédié afin de gérer les utilisateur, les annonces et les catégories.

3.2.2 Interface d'administration « Back-office »

Un back-office dédié afin de gérer le contenu du site internet et les abus.

Seul la personne avec les droits admin aura accès au back-office, ce dernier aura le droit de désactiver, activer, supprimer une annonce ou un utilisateur et aussi créer, éditer ou supprimer une catégorie.

3.2.3 Formulaire de dépôt d'annonce

Un membre pourra déposer une annonce via un formulaire dédié.

3.2.4 Formulaires de contact

Deux types formulaires de contact, un pour contacter le propriétaire du site et l'autre formulaire pour contacter l'annonceur (membre à annonceur).

3.2.5 Barre de recherche

Un utilisateur a la possibilité de rechercher dans les annonces par des mots clés ou un indice.

3.3 Les cibles de l'application

L'application web ciblera les habitants de Saint-Cloud et les villes voisines, les utilisateurs pourront déposer des annonces d'objets perdus.

3.4 Benchmark concurrentiel

Actuellement aucun service ou plateforme en ligne d'objets trouvés n'est proposé par la ville de Saint-Cloud « Voir Fig. 1 ».

Après une recherche sur internet, le seul service des objets perdus qui est proposé se trouve à Paris mais pas officiellement de la ville, afin de vérifier ces informations, on a pris contact avec la mairie de Saint-Cloud qui nous ont confirmé l'inexistence de ce service.

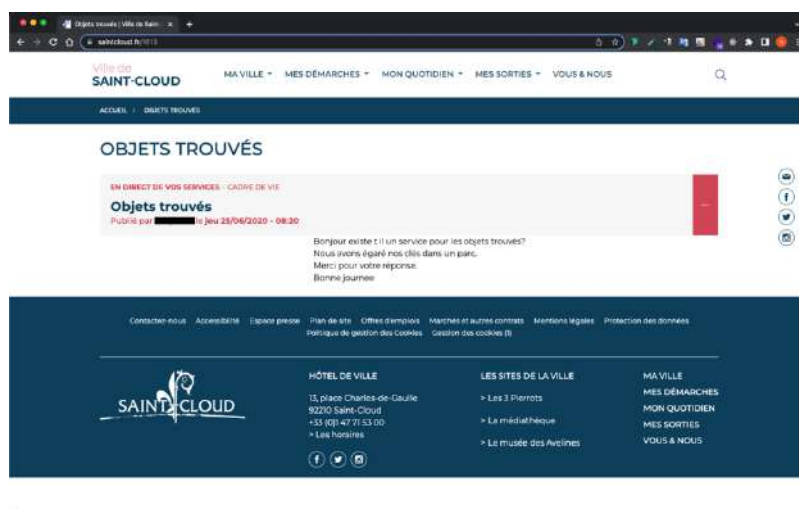
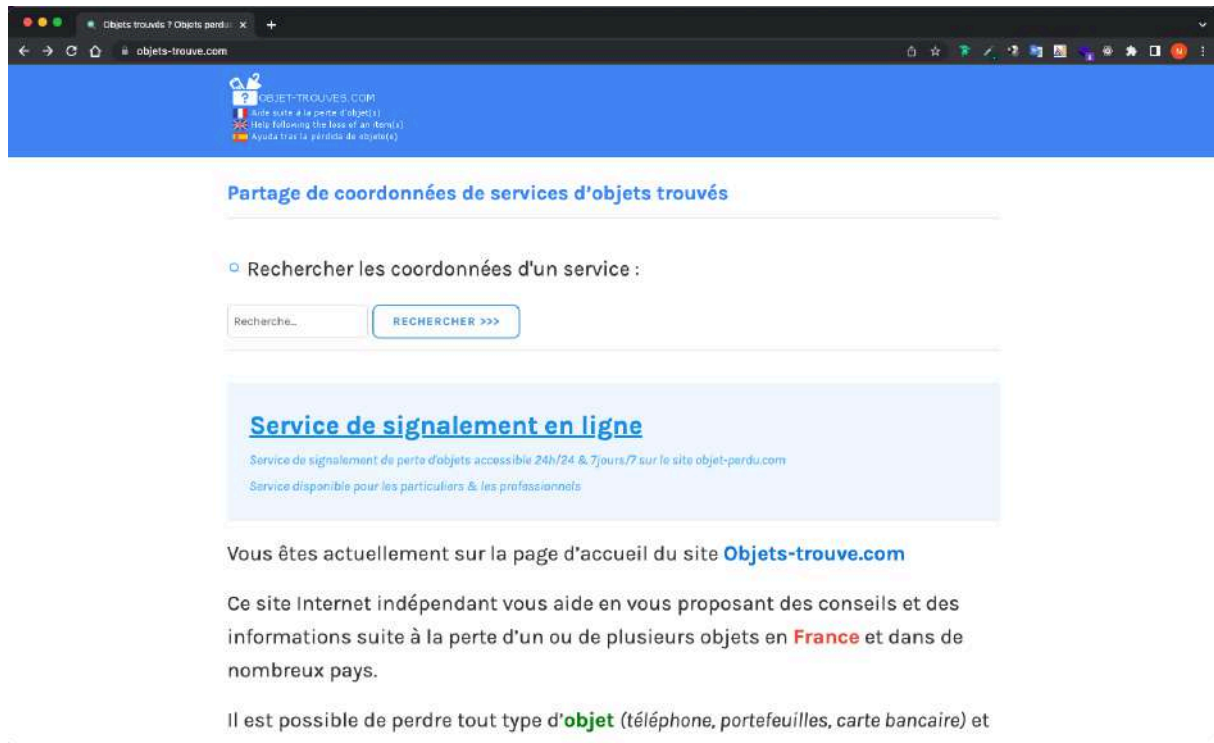


Figure. 1 « Site internet de la mairie de Saint-Cloud »

Nous avons tout de même analysé 2 concurrents :

3.4.1 objets-trouve.com



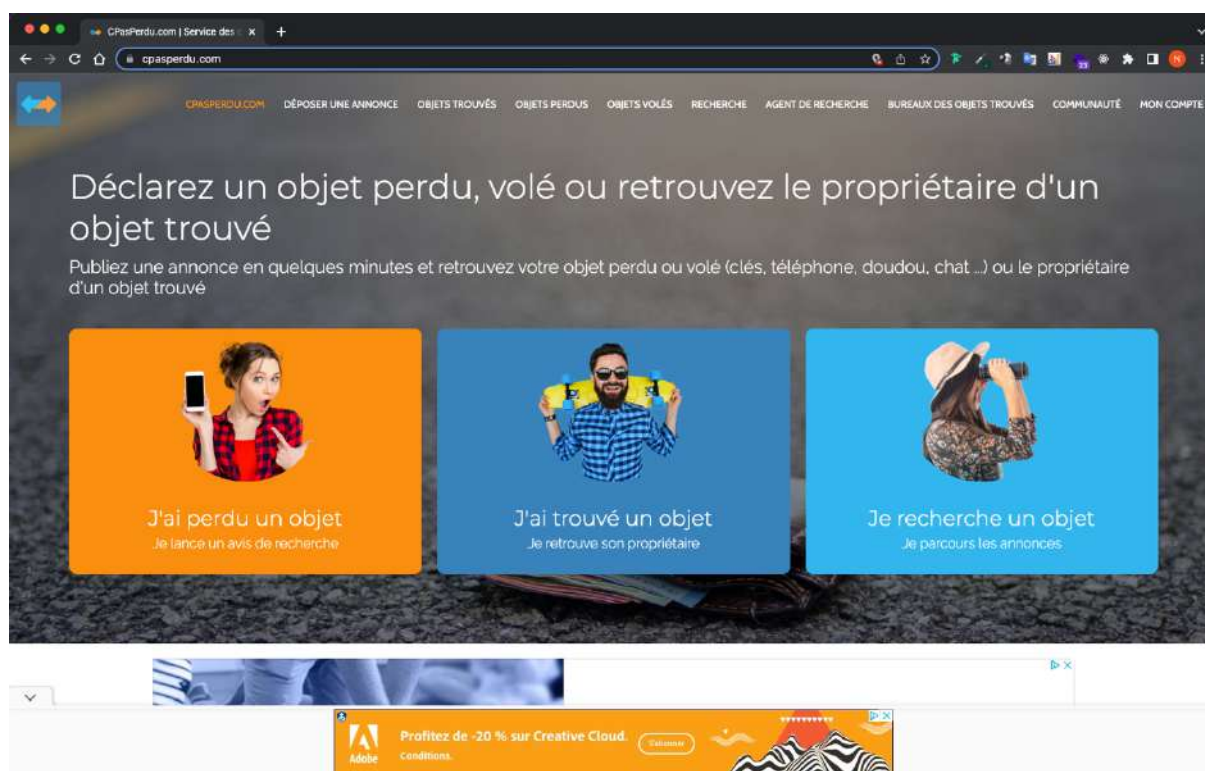
URL : https://objets-trouve.com

Balise Title : Objets trouvés ? Objets perdus ? Obtenez de l'aide avec objets-trouve.com

Meta description : Des millions d'objets sont perdus chaque année en France . Le site objets-trouve.com vous conseille pour contacter le service des objets trouvés adéquat.

Points forts	Points faibles
<ul style="list-style-type: none">• Multilingue• Responsive• Référencement sur les moteurs de recherche	<ul style="list-style-type: none">• Aucune présence réelle dans la ville de Saint-Cloud• Design• Aucun formulaire de contact• Pas de réseaux sociaux

3.4.2 cpasperdu.com



URL : <https://www.cpasperdu.com>

Balise Title : CPasPerdu.com | Service des objets trouvés, perdus et volés sur Internet

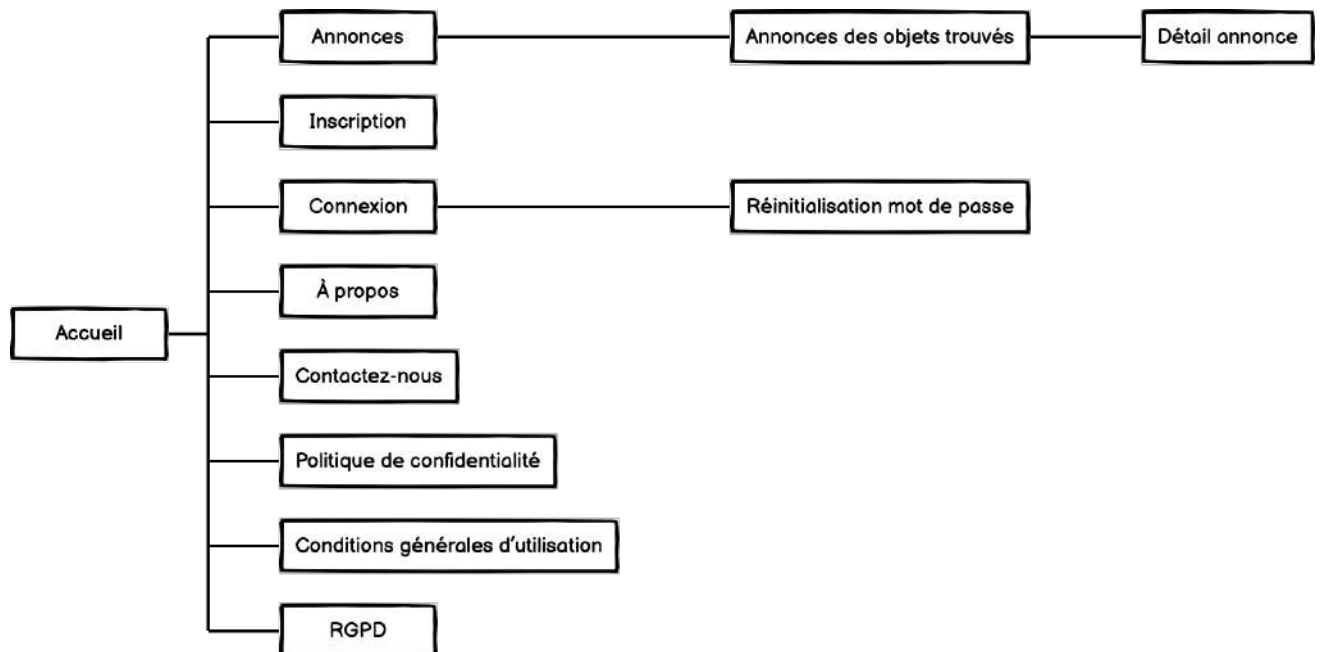
Meta description : CPasPerdu.com, service des objets trouvés sur Internet, vous permet de retrouver vos objets perdus ou volés en quelques minutes

Points forts	Points faibles
<ul style="list-style-type: none">• Responsive• Référencement sur les moteurs de recherche	<ul style="list-style-type: none">• Expérience utilisateur médiocre• Page d'accueil très lourde en chargement• Aucune présence réelle dans la ville de Saint-Cloud• Logo• Barre de navigation• Aucun formulaire de contact• Pas de réseaux sociaux• Footer surchargé de liens• Beaucoup de pop-up et espaces publicitaire présent

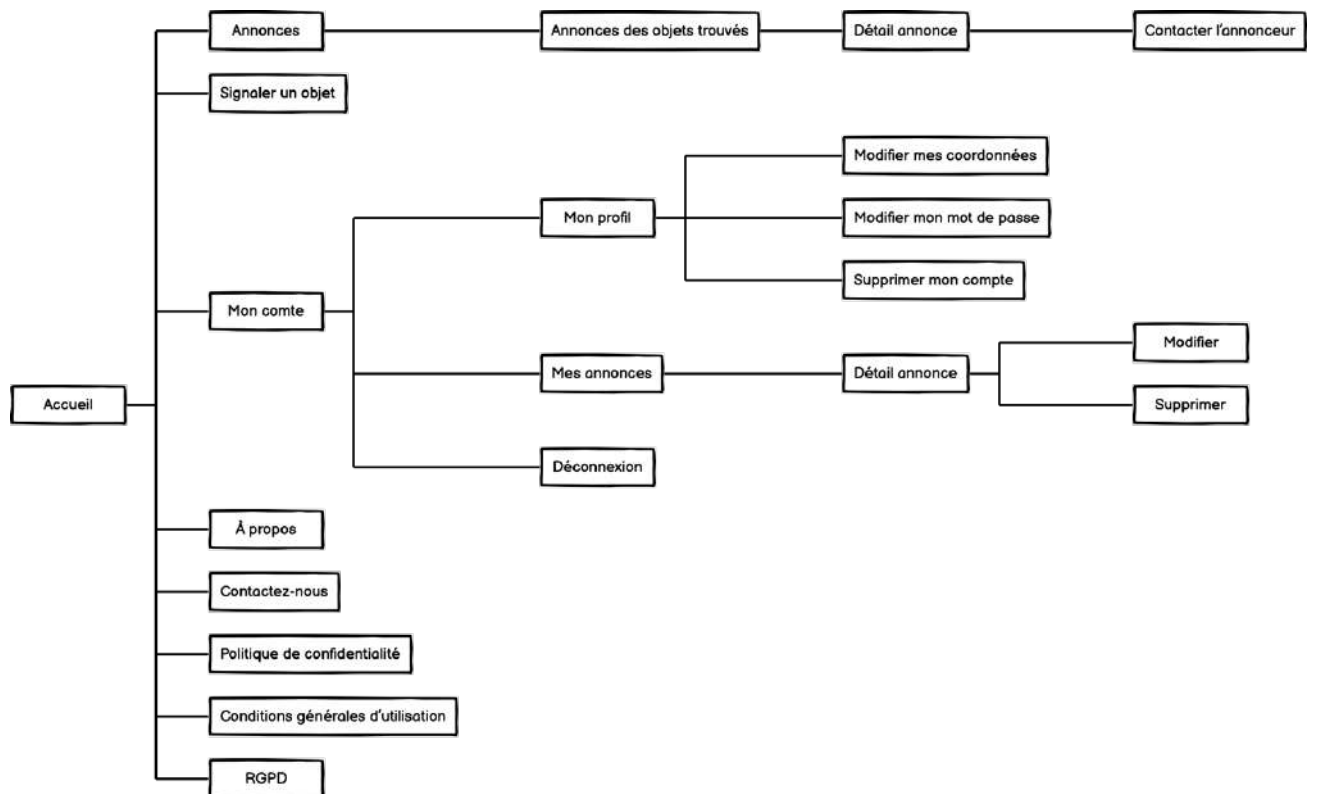
3.5 Arborescence de l'application

3.5.1 Front-end «Plan du site »

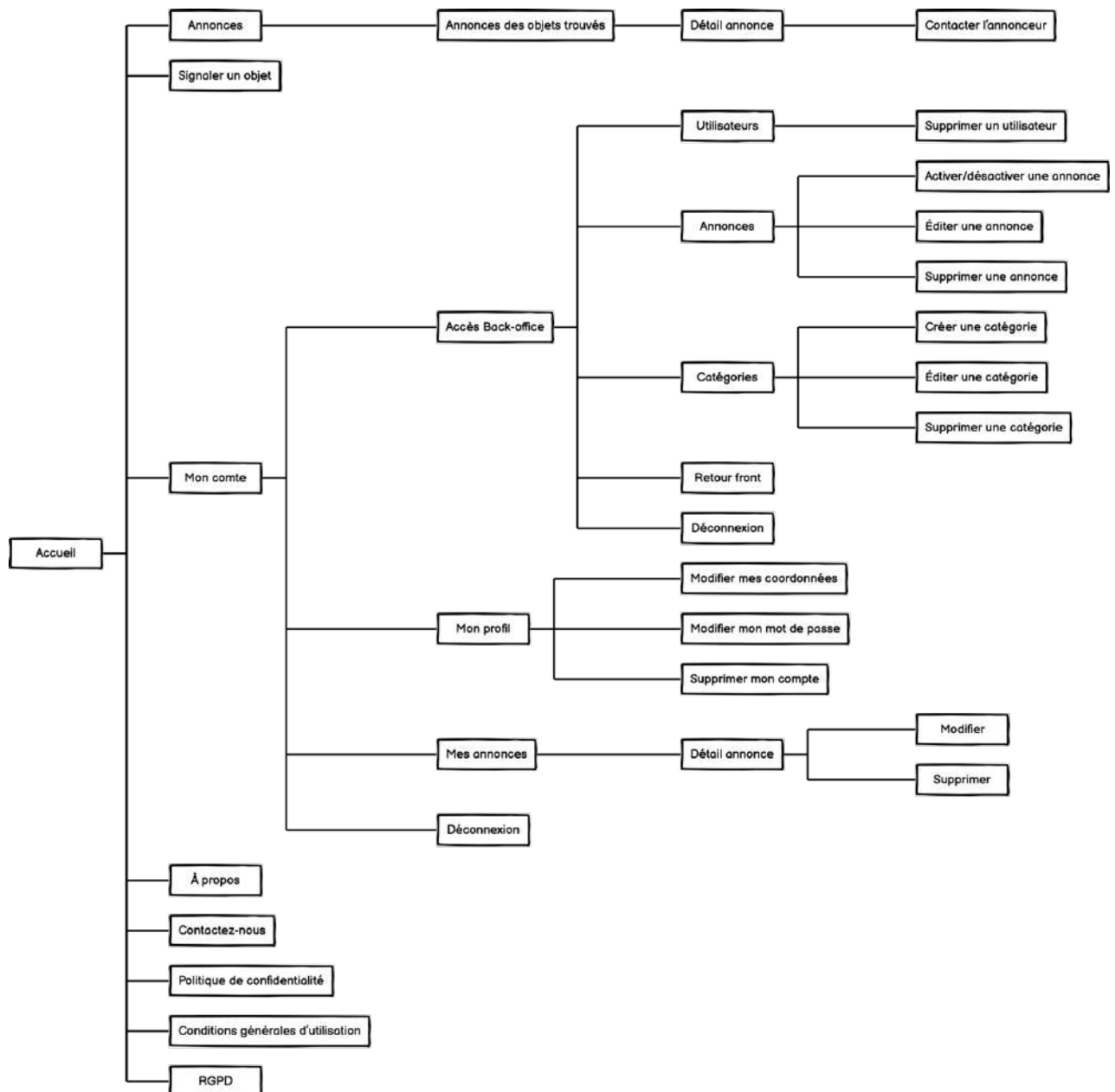
- Utilisateur non inscrit



- Utilisateur inscrit et connecté

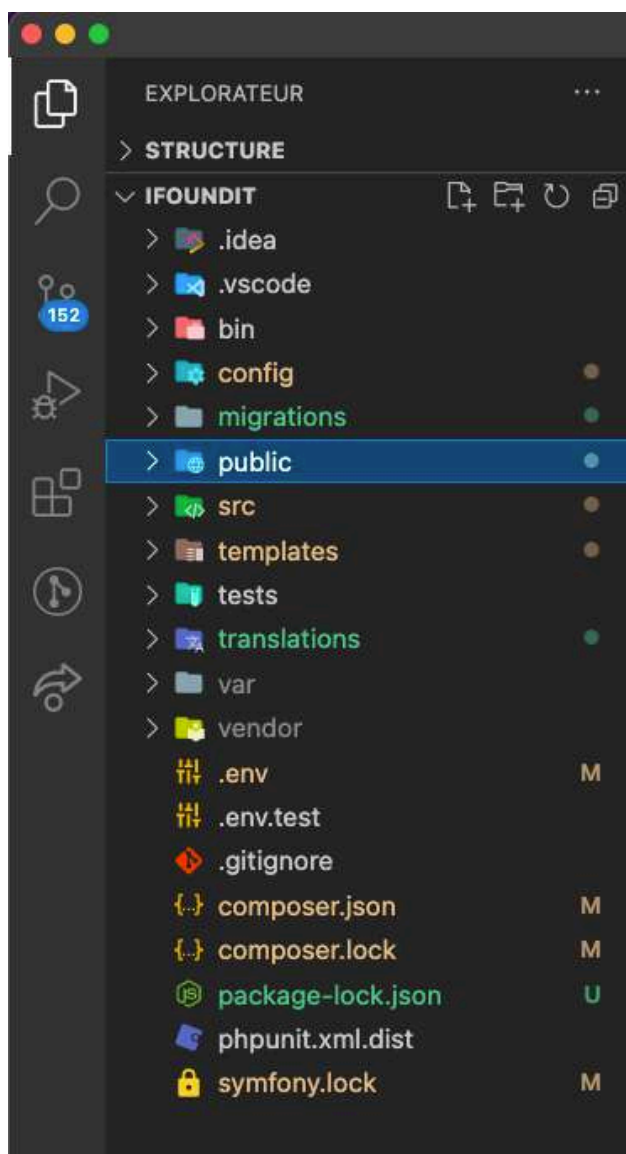


- **Utilisateur administrateur**



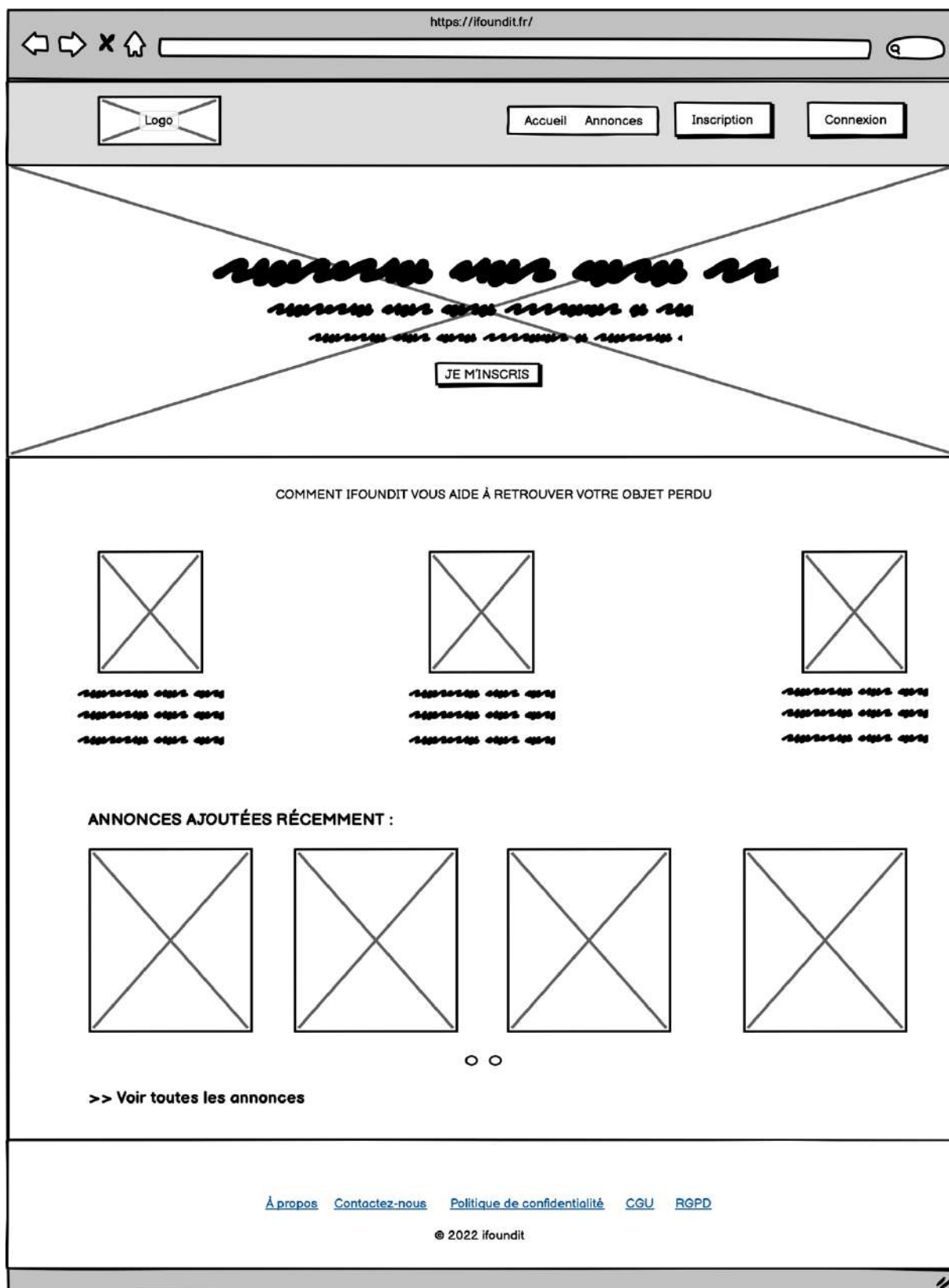
3.5.2 Back-end « Organisation des fichiers »

Une vue générale de l'arborescence des fichiers coté back-end, les dossiers seront exploré dans la partie développement.

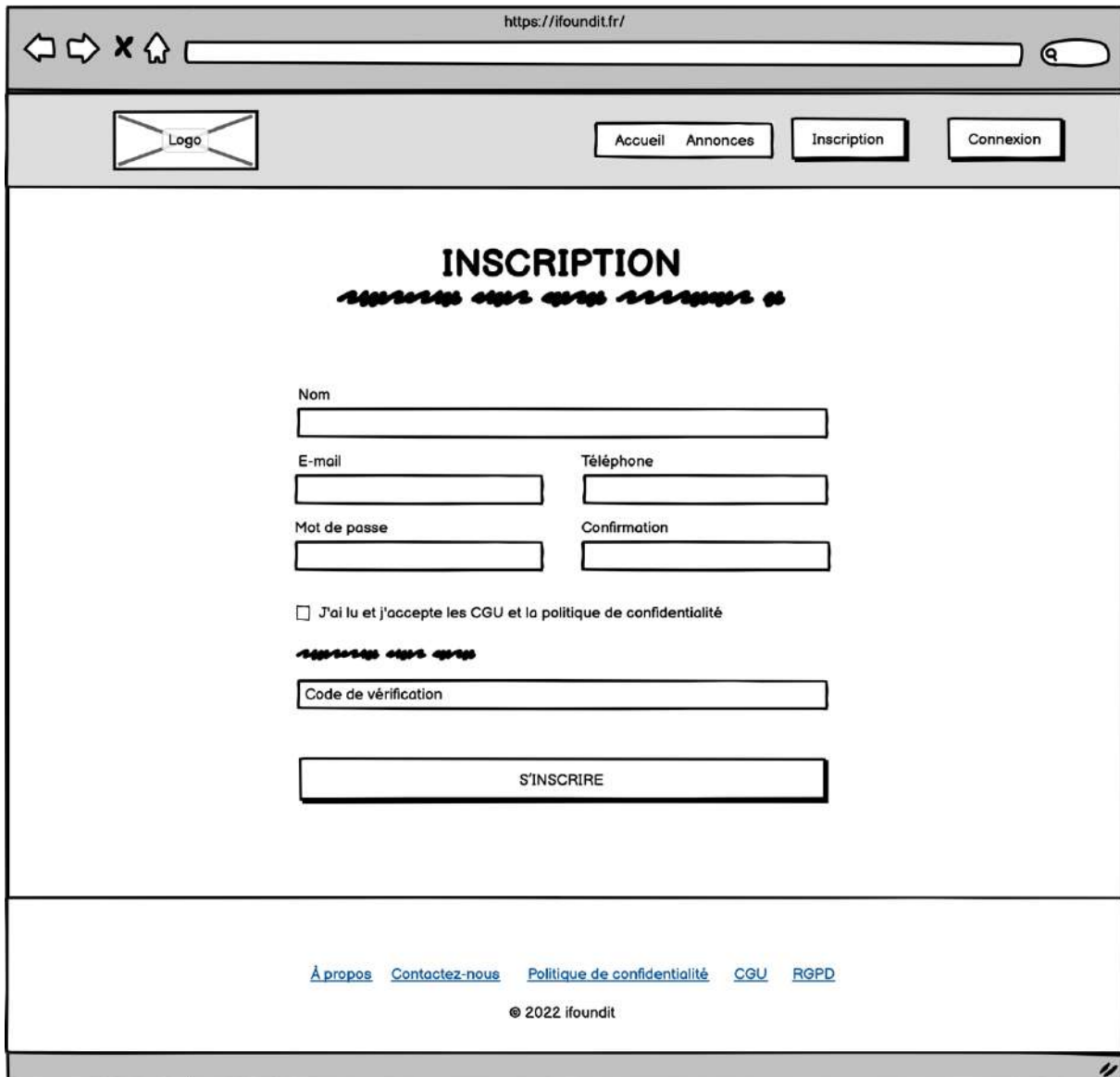


3.6 Wireframe

Page d'accueil



Formulaire d'inscription



The image shows a web browser window with the URL <https://ifoundit.fr/>. The browser's address bar and navigation buttons (back, forward, home, search) are visible. The website's header features a logo placeholder labeled "Logo" and four navigation buttons: "Accueil", "Annonces", "Inscription", and "Connexion". The main content area is titled "INSCRIPTION" in a large, bold, black font, with a decorative horizontal line of small black marks below it. The registration form consists of several input fields: a single field for "Nom", and two columns of fields for "E-mail", "Téléphone", "Mot de passe", and "Confirmation". Below these fields is a checkbox labeled "J'ai lu et j'accepte les CGU et la politique de confidentialité". Another decorative line of small black marks is placed below the checkbox. This is followed by a single input field for "Code de vérification". At the bottom of the form is a large button labeled "S'INSCRIRE". The footer of the page contains five links: "À propos", "Contactez-nous", "Politique de confidentialité", "CGU", and "RGPD", followed by the copyright notice "© 2022 ifoundit".

https://ifoundit.fr/

Logo

Accueil Annonces Inscription Connexion

INSCRIPTION

Nom

E-mail Téléphone

Mot de passe Confirmation

☐ J'ai lu et j'accepte les CGU et la politique de confidentialité

Code de vérification

S'INSCRIRE

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)

© 2022 ifoundit

Formulaire de dépôt d'annonce

https://ifoundit.fr/

Logo

Accueil Annonces Signaler un objet

Mon compte

SIGNALER UN OBJET TROUVÉ

~~~~~

Qu'avez-vous trouvé ?

Image du produit trouvé

Comment j'ai trouvé cet objet ?

Indices et informations supplémentaires (Facultatif)

Détail de l'endroit

Ville  Code postal

J'ai trouvé cet objet le

☐ J'ai lu et j'accepte les CGU et la politique de confidentialité

PUBLIER L'ANNONCE

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)

© 2022 ifoundit

## Annonces des objets trouvés

https://ifoundit.fr/

Logo

Accueil Annonces Signaler un objet

Mon compte

# ANNONCES DES OBJETS TROUVÉS

Recherche...

Il y a 06 annonces

<< Précédent

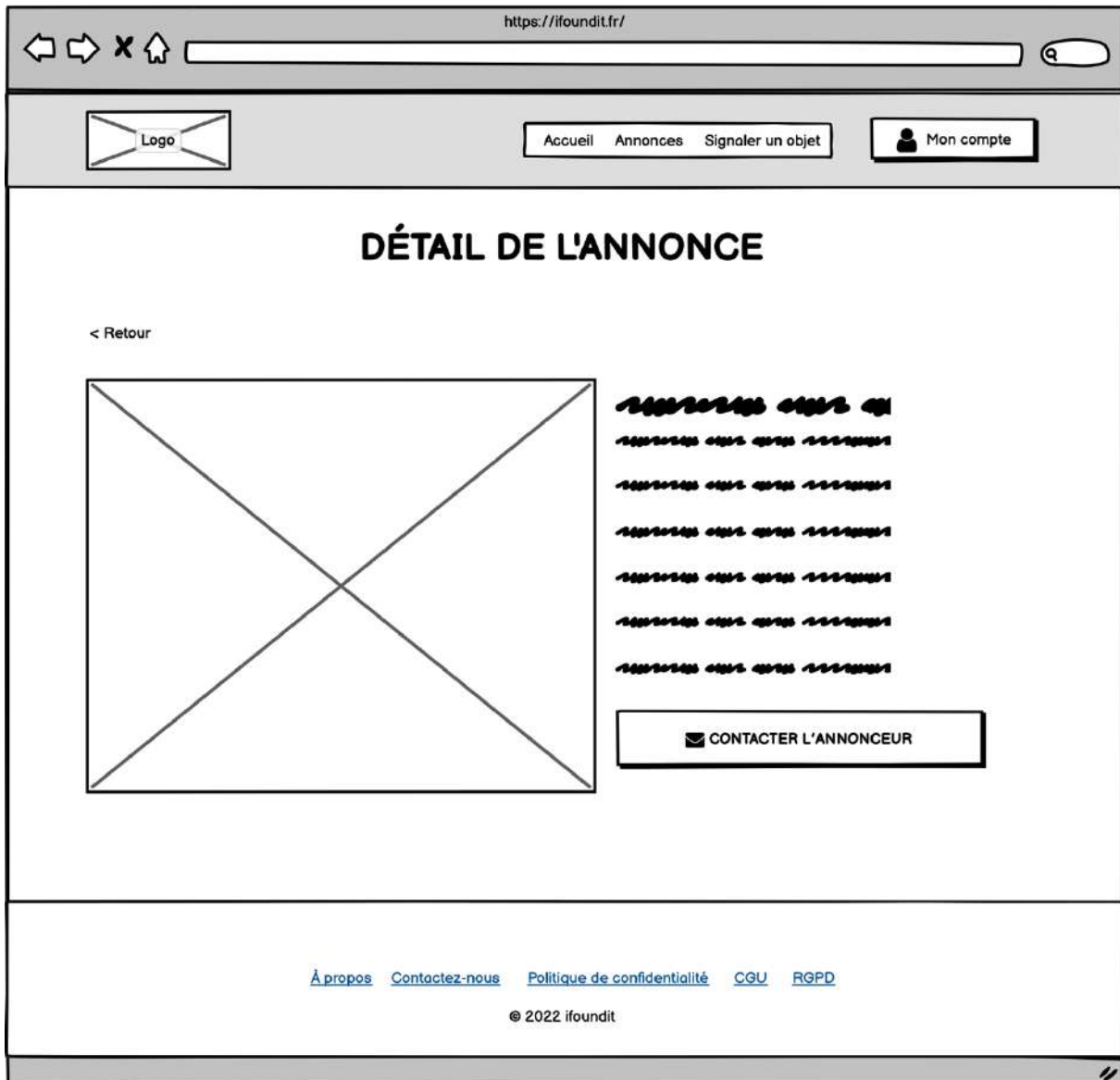
1

2 Suivant >>

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)

© 2022 ifoundit

## Détail de l'annonce



## Profil membre

https://ifoundit.fr/









Logo

AccueilAnnoncesSignaler un objet

Mon compte

# MON PROFIL

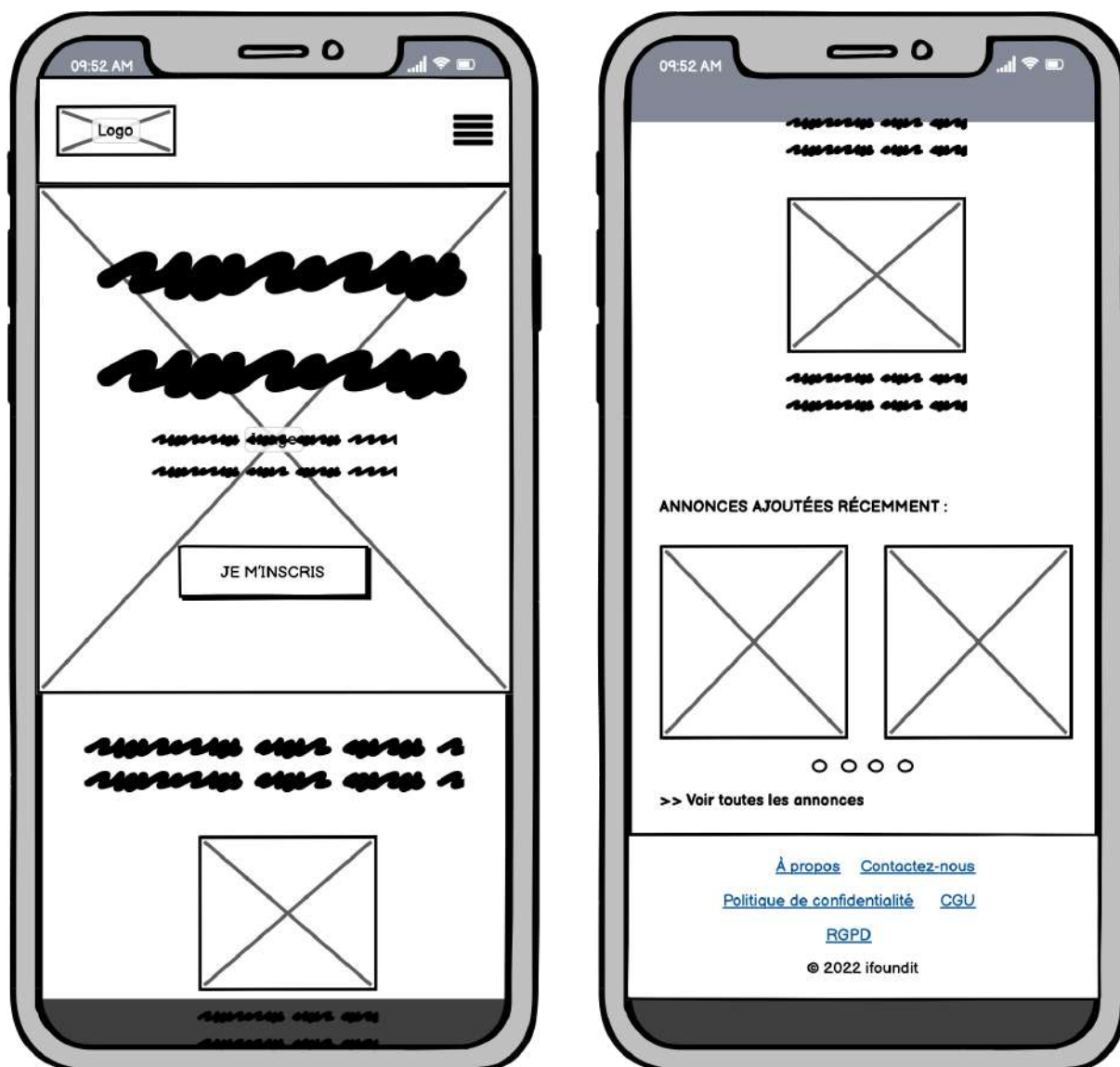
## DÉTAIL DE MON COMPTE

|                                                                                     |                             |                                           |
|-------------------------------------------------------------------------------------|-----------------------------|-------------------------------------------|
|    | Nom :                       | XXXXXXXXXX XXXX XXXX XXXX                 |
|    | E-mail :                    | XXXXXXXXXX XXXX XXXX XXXX                 |
|    | Téléphone :                 | XXXXXXXXXX XXXX XXXX XXXX                 |
|    | Compte crée le :            | XXXXXXXXXX XXXX XXXX XXXX                 |
|    | Mes signalements d'objets : | <a href="#">Voir mes annonces</a>         |
|  | Gérer mon compte :          | <a href="#">Modifier mes coordonnées</a>  |
|  | Mot de passe :              | <a href="#">Modifier mon mot de passe</a> |
|  | Supprimer mon compte :      | <a href="#">Supprimer</a>                 |

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)

© 2022 ifoundit

## Coté Responsive





### 3.7 Budgétisation du projet

Un budget prévisionnel a été mis en place concernant l'application.

Le devis prend en considération les éléments suivants :

- Développement de l'application sur-mesure sous PHP/Symfony.
- Une interface d'administration « Back-office »
- Création de logo.
- Optimisation pour le référencement.
- Paramétrage et préparation de l'espace hébergement.
- Mise en ligne de l'application.
- Mise à jour et SAV pour une durée de 1 an.

Le nom de domaine et l'hébergement sont à la charge du client.

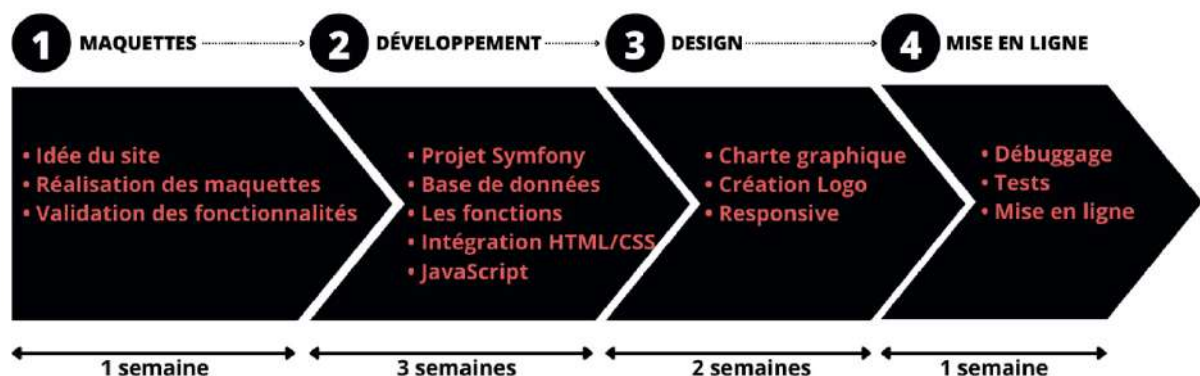
Après étude de ces critères, le coût de l'application reviendrai à 5000€.

### 3.8 Calendrier prévisionnel

La durée de la conception de l'application web « ifoundit.fr » est estimée à 7 semaines.

La livraison est prévue pour début juillet 2022.

De la maquette à la mise en ligne la conception de l'application se déroule comme suit :



## 4 UML

« **Unified Modeling Language** »

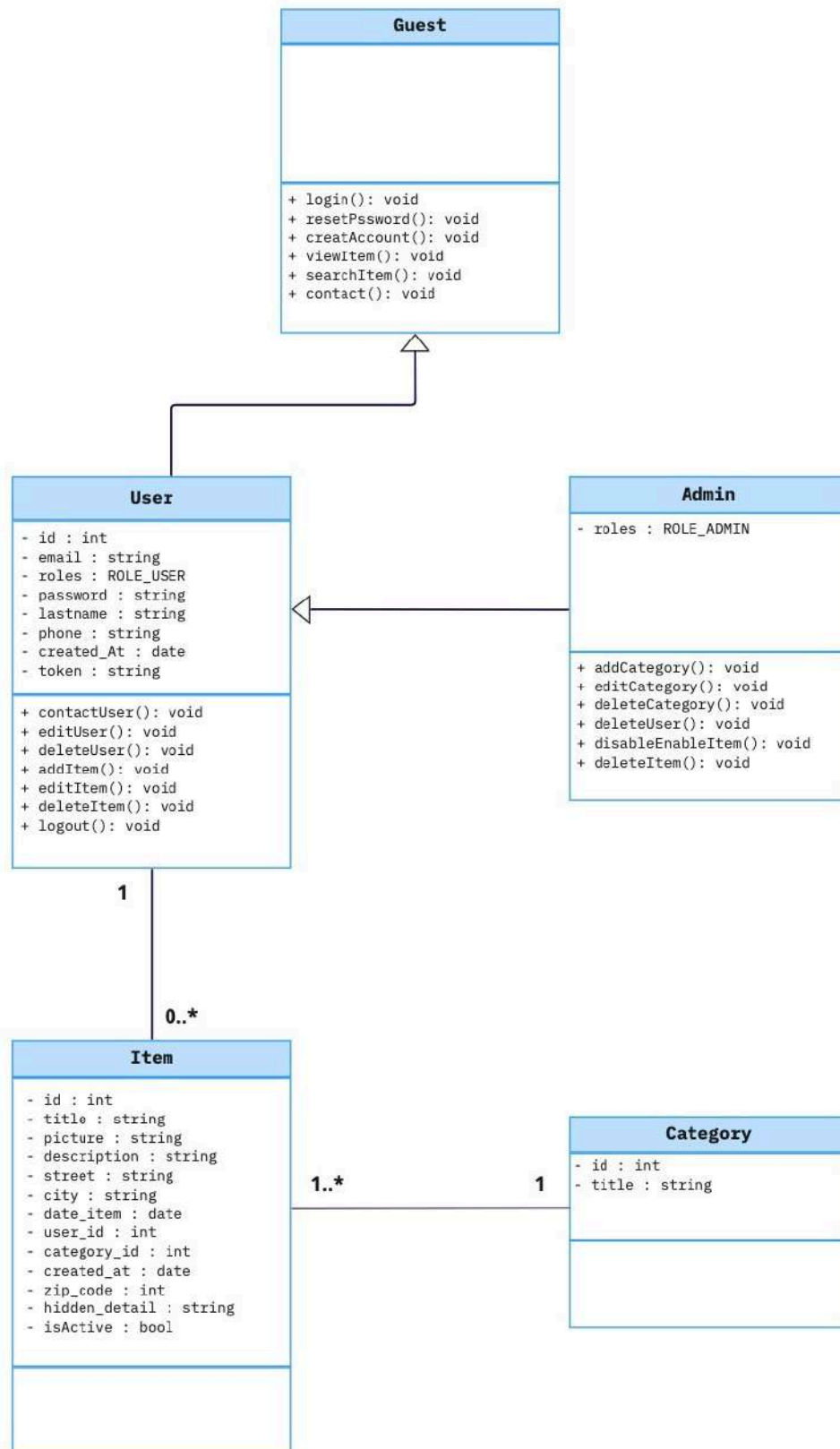
## 4.1 Diagramme de cas d'utilisation

Ci-dessous le diagramme UML de cas d'utilisation afin de donner une vision globale du comportement fonctionnel de l'application web « ifoundit »



## 4.2 Diagramme de classes

Ci-dessous le diagramme de classes UML, afin de donner une vision plus clair de la structure de l'application « ses classes, ses attributs, ses méthodes et ses relations ».







# 5 DÉVELOPPEMENT DE L'APPLICATION



## 5.1 Langages et Framework

Pour la réalisation du projet, nous avons utilisés les langages et les logiciels suivants :



- **HTML 5**

Signifie « HyperText Markup Language » qu'on peut traduire par « langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure. « Source : MDN Web Docs »



- **CSS**

Signifie « Cascading Style Sheets », permet de créer des pages web à l'apparence soignée. « Source : MDN Web Docs »



Bootstrap

- **Bootstrap 5**

est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs. « Source : Wikipedia »



- **JavaScript**

JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. « Source : MDN Web Docs »



- **jQuery**

Une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web. « Source : Wikipedia »



Symfony

- **Framework Symfony**

Est un ensemble de composants PHP ainsi qu'un framework MVC libre écrit en PHP. Il fournit des fonctionnalités modulables et adaptables qui permettent de faciliter et d'accélérer le développement d'un site web. « Source : Wikipedia »



- **PHP**

PHP: Hypertext Preprocessor, est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur http. « Source : Wikipedia »



- **Base de données MySQL**

MySQL Database Service est un service de base de données entièrement géré pour déployer des applications natives du cloud en utilisant la base de données open source la plus populaire au monde. Ce service est développé, géré et supporté à 100% par l'équipe de MySQL. « Source : <https://mysql.com/fr/> »



- **PhpMyAdmin**

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes de gestion de base de données MySQL et MariaDB. « Source : Wikipedia ».



- **Visual Studio Code**

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes de gestion de base de données MySQL et MariaDB. « Source : Wikipedia »



- **Balsamiq**

Est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS.

## 5.2 Installation du framework Symfony

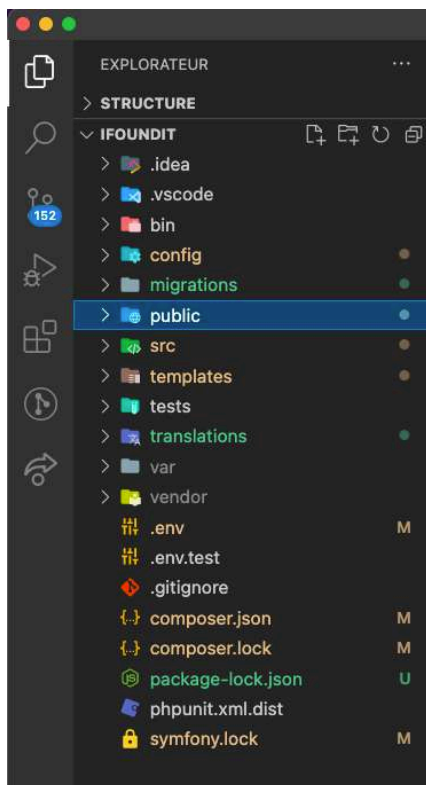
L'application web « ifoundit » sera entièrement développer sous le Framework Symfony version 5.4.

Ce framework utilise l'architecture MVC « Modèle-Vue- Contrôleur »

Afin de commencer le développement de l'application web, nous lançons un nouveau projet Symfony en exécutant la commande suivante dans le terminal à la racine du projet :

```
composer create-project symfony/website-skeleton:"^5.4" ifoundit
```

## 5.3 L'arborescence des fichiers



### 5.3.1 Le dossier « bin »

Il contient les fichiers « **console** » et « **phpunit** »

Le fichier « **console** » comporte tous les exécutable de Symfony c'est-à-dire toutes les commandes disponible via la CLI.

Le fichier « **phpunit** » permet de faire des tests unitaires.

### 5.3.2 Le dossier « config »

Il contient toute la configuration des packages, services et routes se fera dans ce dossier.

### 5.3.3 Le dossier « public »

C'est le point d'entrée de l'application, il représente la « **racine** » de notre application,

C'est le dossier de base auquel le serveur va accéder depuis le navigateur.

Il contient les éléments suivants :

- **Fichier « index.php »**  
c'est le fichier sur lequel les navigateurs web vont pointer.
- **Dossier « CSS »**  
pour les feuilles de styles.
- **Dossier « Img »**  
pour stocker toutes les images de l'application.
- **Dossier « Js »**  
pour les scripts Javascript
- **Dossier « Upload »**  
pour stocker les images uploadé
- **Fichier « .htaccess »**  
c'est le fichier qui sera créé avant la mise en ligne de l'application, qui servira à la réécriture d'url.  
Ce fichier n'est plus présent par défaut dans les installations de Symfony et il faut donc le créer.

### 5.3.4 Le dossier « src »

C'est le dossier principal de notre application, il contient l'ensemble de nos sources PHP et il contient les dossiers suivants :

- **Le dossier « Controller »**  
C'est le dossier qui contient tous les Controllers de notre application.  
Le rôle principal de la fonction Controller, est de retourner une réponse.
- **Le dossier « Form »**  
Ce dossier contient tous les formulaires générés.

- **Le dossier « Entity »**

Ce dossier contient toutes les entités qui représentent les tables de notre base de données, qui permettra de mettre à jour notre base de données à chaque modification de l'Entity.

- **Le dossier « Migration »**

Contient les tables de notre base de données en locale

- **Le dossier « Repository »**

Rattaché à une « Entity », il permet de créer nos requêtes en utilisant l'ORM Doctrine qui permet de créer nos requêtes SQL à travers les queryBuilder

- **Le dossier « templates »**

C'est la partie « Vue » du modèle MVC.

Il regroupe les fichiers de rendu du moteur de template « twig » avec les extensions « html.twig ».

- **Le dossier « vendor »**

C'est le dossier qui stock toutes les dépendances installées.

Ces dépendances sont installées via un gestionnaire qui est « composer » qui nous permet d'installer une librairie.

- **Le dossier « var »**

Sert a stocké le cache et les fichiers log

- **Le fichier « composer.json »**

Il contient la liste des dépendances installé.

- **Le fichier « .env »**

Ce fichier contient nos variables d'environnement.

C'est dans ce fichier qu'on va paramétrer entre autres l'accès à la base de données, paramétrer les e-mails.

## 5.4 La Base de données

Pour interagir avec la base de données, nous avons utilisé l'ORM « Doctrine » qui a été installé via la console.

L'ORM (Object-Relational Mapping) a pour but de faciliter ses différentes interactions avec la base de données.

### 5.4.1 Paramétrage

Dans un 1<sup>er</sup> temps nous installons le logiciel **MAMP** (Mac Os) qui va nous permettre de lancer un serveur web local.

Ensuite nous devons configurer le fichier « .env » qui permettra de configurer notre application avec la base de donnée comme suit :

```
# DATABASE_URL="mysql://root:@127.0.0.1:3306/skel?serverVersion=MySQL-10.4.21"
DATABASE_URL="mysql://root:root@127.0.0.1:8889/ifoundit?serverVersion=5.7"
#DATABASE_URL="mysql://root:root@127.0.0.1:3306/skel?serverVersion=MySQL-10.4.21"

###> symfony/framework-bundle ###
APP_ENV=dev
```

- **DATABASE\_URL**

C'est l'URL de la base de données qui contient les éléments suivants :

- db\_user = root (Utilisateur)
- db\_password = root (mot de passe)
- db\_name = ifoundit (nom)

- **APP\_ENV**

C'est l'environnement de l'application qui est sur « **dev** » pendant le développement et sera configuré sur production « **prod** » en fin de développement et une fois l'application est mise en ligne.

### 5.4.2 Création de la base de données

Nous procédons à la création de la base de données avec la commande suivante :

```
php bin/console doctrine:database:create
```

La base de données a bien été créée :

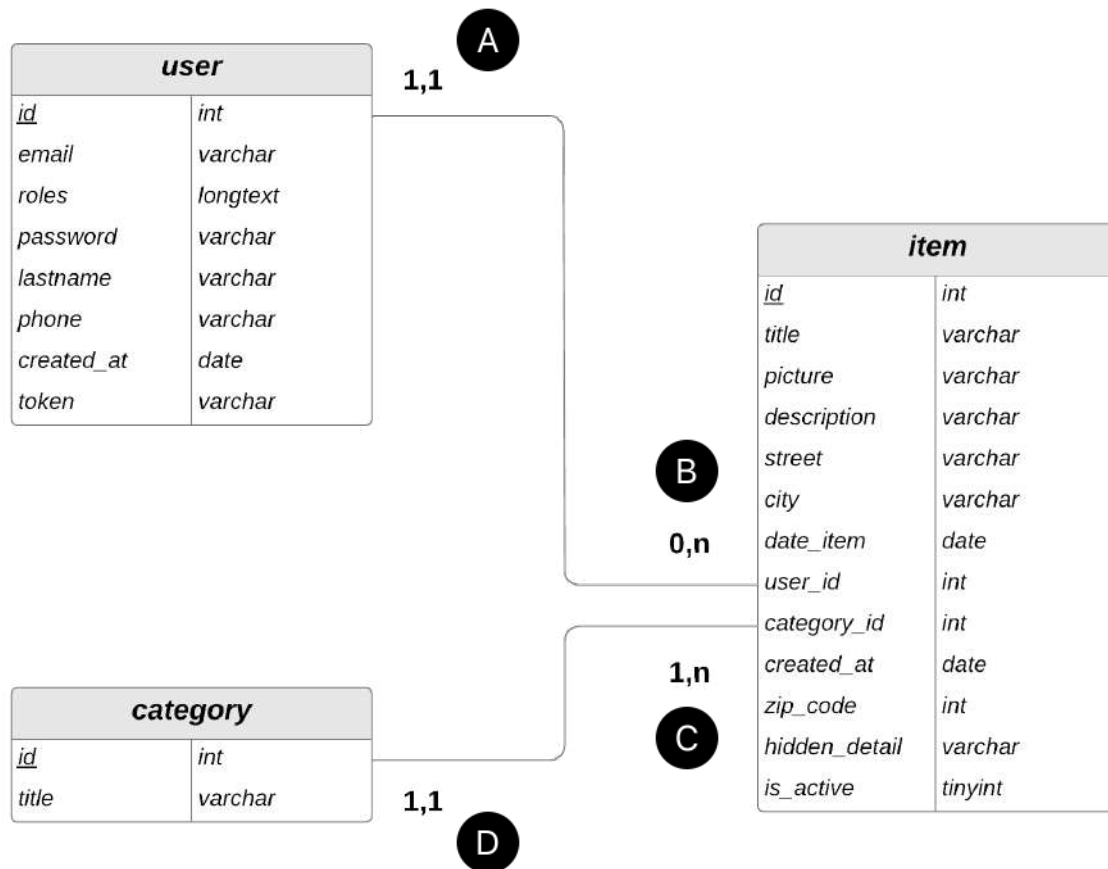


### 5.4.3 Les tables de la base de données

Dans notre projet nous avons besoin de 3 tables :

- Une table « user » pour les membres inscrit et administrateur.
- Une table « item » pour les annonces déposées.
- Une table « category » pour les catégories de dépôt d'annonces.

#### 5.4.4 Diagramme EA « entité-association » de la base de données



- **Les cardinalités et types de relations**

- A « 1,1 » :** Un item peut avoir 1 user et au maximum 1 user.
- B « 0,n » :** 1 user peut avoir 0 ou plusieurs item.
- C « 1,n » :** 1 category peut avoir 1 ou plusieurs item.
- D « 1,1 » :** Un item peut avoir 1 category et au maximum 1 category.

#### 5.4.5 Création des tables

Prenons l'exemple de la table « User » que nous souhaitons créer.

Nous tapons la commande suivante sur la console :

```
php bin/console make:entity
```

Ensuite on nous demande le nom de l'entité que nous souhaitons créer, le nom des propriétés (ex : nom, mail, téléphone...), le type des propriétés «string», «integer»...etc.

Ensuite nous créons les relation entre les entités, dans notre exemple c'est la relation entre la table « user » et « item » :

```
> relation

What class should this entity be related to?:
> User

What type of relationship is this?

-----
Type                Description
-----
ManyToOne            Each Item relates to (has) one User.
                     Each User can relate to (can have) many Item objects
OneToMany            Each Item can relate to (can have) many User objects.
                     Each User relates to (has) one Item
ManyToMany           Each Item can relate to (can have) many User objects.
                     Each User can also relate to (can also have) many Item objects
OneToOne             Each Item relates to (has) exactly one User.
                     Each User also relates to (has) exactly one Item.
-----

Relation type? [ManyToOne, OneToMany, ManyToMany, OneToOne]:
> ManyToOne
```

Nous choisissons la relation « **ManyToOne** » afin d'avoir la relation suivante :

- Chaque « **item** » a un « **user** »
- Et
- Chaque « **user** » peut avoir plusieurs « **item** »

C'est ce qui va correspondre à notre projet pour avoir la relation suivante entre « **user** » utilisateur et « **item** » annonce :

« Chaque annonce a un utilisateur et chaque utilisateur peut avoir plusieurs annonces »

### 5.4.6 Insertion des données dans la base

Après la création de notre entité (table) « user », les propriétés (champs) et les relations, l'insertion définitive des données en base de données se fait après avoir lancée 2 commandes :

- **1<sup>ère</sup> commande :**

```
php bin/console make:migration
```

Cette commande permet de créer un fichier de migration dans « ifoundit/migration »

```
1  <?php
2
3  declare(strict_types=1);
4
5  namespace Doctrine\Migrations;
6
7  use Doctrine\DBAL\Schema\Schema;
8  use Doctrine\Migrations\AbstractMigration;
9
10 You, $y & $m => |author (You)
11
12 // Auto-generated Migration: Please modify to your needs!
13
14 You, $y & $m => |author (You)
15
16 final class Version20220207132750 extends AbstractMigration
17 {
18     public function getDescription(): string
19     {
20         return '';
21     }
22
23     public function up(Schema $schema): void
24     {
25         // this up() migration is auto-generated, please modify it to your needs
26         $this->addSql('CREATE TABLE user (id INT AUTO_INCREMENT NOT NULL, email VARCHAR(180) NOT NULL, roles JSON NOT NULL, password VARCHAR(255) NOT NULL, lastname VARCHAR(255) NOT NULL, phone VARCHAR(100) NOT NULL, UNIQUE INDEX UNIQ_80930649E7927C74 (email), PRIMARY KEY (id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci ENGINE = InnoDB');
27     }
28
29     public function down(Schema $schema): void
30     {
31         // this down() migration is auto-generated, please modify it to your needs
32         $this->addSql('DROP TABLE user');
```



- 2<sup>ème</sup> commande :

**php bin/console doctrine:migrations:migrate**

Cette commande permet à Doctrine l'exécution du SQL et l'insertion définitive de notre table « user » en base de données.

Une fois terminé, une classe PHP est créée dans le dossier ifoundit /src/Entity. C'est notre entité « User » :

```

User.php M X
3 namespace App\Entity;
4
5 use Doctrine\ORM\Mapping as ORM;
6 use App\Repository\UserRepository;
7 use Doctrine\Common\Collections\Collection;
8 use Symfony\Component\Validator\Constraints;
9 use Symfony\Component\Validator\Constraints as Assert;
10 use Doctrine\Common\Collections\ArrayCollection;
11 use Symfony\Component\Validator\Constraints\NotBlank;
12 use Symfony\Component\Security\Core\User\UserInterface;
13 use Symfony\Bridge\Doctrine\Validator\Constraints\UniqueEntity;
14 use Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
15
16 You, il y a 37 secondes | 1 author (You)
17 /**
18  * @ORM\Entity(repositoryClass=UserRepository::class)
19  * @UniqueEntity(
20  *   fields={"email"},
21  *   message="Un compte existe déjà avec cette adresse e-mail"
22  * )
23  */
24 You, il y a 37 secondes | 1 author (You)
25 class User implements PasswordAuthenticatedUserInterface, UserInterface
26 {
27     /**
28      * @ORM\Id
29      * @ORM\GeneratedValue
30      * @ORM\Column(type="integer")
31      */
32     private $id;
33
34     /**
35      * @ORM\Column(type="string", length=180, unique=true)
36      * @Assert\NotBlank(message="Veuillez saisir une adresse e-mail")
37      * @Assert\Email(message="E-mail invalide")
38      */
39     private $email;
40
41     /**
42      * @ORM\Column(type="json")
43      */
44     private $roles = ['ROLE_USER'];
45
46     /**
47      * @var string The hashed password
48      */
49     private $password;
50
51     public function getId(): ?int
52     {
53         return $this->id;
54     }
55
56     public function getEmail(): ?string
57     {
58         return $this->email;
59     }
60
61     public function setEmail(string $email): self
62     {
63         $this->email = $email;
64
65         return $this;
66     }
67
68     /**
69      * A visual identifier that represents this user.
70      *
71      * @see UserInterface
72      *
73      * @return string The username
74      */
75     public function getUsername(): ?string
76     {
77         return $this->email;
78     }
79
80     public function getRoles(): array
81     {
82         $roles = $this->roles;
83
84         if (empty($roles)) {
85             // hack for some legacy DBs
86             // to replace the empty array by an empty iterable
87             $roles = new \IteratorAggregate([]);
88         }
89
90         return $roles;
91     }
92
93     public function getPassword(): string
94     {
95         return $this->password;
96     }
97
98     public function setPassword(string $password): self
99     {
100         $this->password = $password;
101
102         return $this;
103     }
104
105     /**
106      * @see UserInterface
107      *
108      * @return bool Whether the user is verified or not
109      */
110     public function isVerified(): bool
111     {
112         return true;
113     }
114
115     /**
116      * @see UserInterface
117      *
118      * @return bool Whether the user is active or not
119      */
120     public function isActive(): bool
121     {
122         return true;
123     }
124
125     /**
126      * @see UserInterface
127      *
128      * @return bool Whether the user is locked or not
129      */
130     public function isLocked(): bool
131     {
132         return true;
133     }
134
135     /**
136      * @see UserInterface
137      *
138      * @return bool Whether the user is expired or not
139      */
140     public function isExpired(): bool
141     {
142         return true;
143     }
144
145     /**
146      * @see UserInterface
147      *
148      * @return bool Whether the user is disabled or not
149      */
150     public function isDisabled(): bool
151     {
152         return true;
153     }
154
155     /**
156      * @see UserInterface
157      *
158      * @return bool Whether the user is confirmed or not
159      */
160     public function isConfirmed(): bool
161     {
162         return true;
163     }
164
165     /**
166      * @see UserInterface
167      *
168      * @return bool Whether the user is trusted or not
169      */
170     public function isTrusted(): bool
171     {
172         return true;
173     }
174
175     /**
176      * @see UserInterface
177      *
178      * @return bool Whether the user is persistent or not
179      */
180     public function isPersistent(): bool
181     {
182         return true;
183     }
184
185     /**
186      * @see UserInterface
187      *
188      * @return bool Whether the user is enabled or not
189      */
190     public function isEnabled(): bool
191     {
192         return true;
193     }
194
195     /**
196      * @see UserInterface
197      *
198      * @return bool Whether the user is active or not
199      */
200     public function isActive(): bool
201     {
202         return true;
203     }
204
205     /**
206      * @see UserInterface
207      *
208      * @return bool Whether the user is locked or not
209      */
210     public function isLocked(): bool
211     {
212         return true;
213     }
214
215     /**
216      * @see UserInterface
217      *
218      * @return bool Whether the user is expired or not
219      */
220     public function isExpired(): bool
221     {
222         return true;
223     }
224
225     /**
226      * @see UserInterface
227      *
228      * @return bool Whether the user is disabled or not
229      */
230     public function isDisabled(): bool
231     {
232         return true;
233     }
234
235     /**
236      * @see UserInterface
237      *
238      * @return bool Whether the user is confirmed or not
239      */
240     public function isConfirmed(): bool
241     {
242         return true;
243     }
244
245     /**
246      * @see UserInterface
247      *
248      * @return bool Whether the user is trusted or not
249      */
250     public function isTrusted(): bool
251     {
252         return true;
253     }
254
255     /**
256      * @see UserInterface
257      *
258      * @return bool Whether the user is persistent or not
259      */
260     public function isPersistent(): bool
261     {
262         return true;
263     }
264
265     /**
266      * @see UserInterface
267      *
268      * @return bool Whether the user is enabled or not
269      */
270     public function isEnabled(): bool
271     {
272         return true;
273     }
274
275     /**
276      * @see UserInterface
277      *
278      * @return bool Whether the user is active or not
279      */
280     public function isActive(): bool
281     {
282         return true;
283     }
284
285     /**
286      * @see UserInterface
287      *
288      * @return bool Whether the user is locked or not
289      */
290     public function isLocked(): bool
291     {
292         return true;
293     }
294
295     /**
296      * @see UserInterface
297      *
298      * @return bool Whether the user is expired or not
299      */
300     public function isExpired(): bool
301     {
302         return true;
303     }
304
305     /**
306      * @see UserInterface
307      *
308      * @return bool Whether the user is disabled or not
309      */
310     public function isDisabled(): bool
311     {
312         return true;
313     }
314
315     /**
316      * @see UserInterface
317      *
318      * @return bool Whether the user is confirmed or not
319      */
320     public function isConfirmed(): bool
321     {
322         return true;
323     }
324
325     /**
326      * @see UserInterface
327      *
328      * @return bool Whether the user is trusted or not
329      */
330     public function isTrusted(): bool
331     {
332         return true;
333     }
334
335     /**
336      * @see UserInterface
337      *
338      * @return bool Whether the user is persistent or not
339      */
340     public function isPersistent(): bool
341     {
342         return true;
343     }
344
345     /**
346      * @see UserInterface
347      *
348      * @return bool Whether the user is enabled or not
349      */
350     public function isEnabled(): bool
351     {
352         return true;
353     }
354
355     /**
356      * @see UserInterface
357      *
358      * @return bool Whether the user is active or not
359      */
360     public function isActive(): bool
361     {
362         return true;
363     }
364
365     /**
366      * @see UserInterface
367      *
368      * @return bool Whether the user is locked or not
369      */
370     public function isLocked(): bool
371     {
372         return true;
373     }
374
375     /**
376      * @see UserInterface
377      *
378      * @return bool Whether the user is expired or not
379      */
380     public function isExpired(): bool
381     {
382         return true;
383     }
384
385     /**
386      * @see UserInterface
387      *
388      * @return bool Whether the user is disabled or not
389      */
390     public function isDisabled(): bool
391     {
392         return true;
393     }
394
395     /**
396      * @see UserInterface
397      *
398      * @return bool Whether the user is confirmed or not
399      */
400     public function isConfirmed(): bool
401     {
402         return true;
403     }
404
405     /**
406      * @see UserInterface
407      *
408      * @return bool Whether the user is trusted or not
409      */
410     public function isTrusted(): bool
411     {
412         return true;
413     }
414
415     /**
416      * @see UserInterface
417      *
418      * @return bool Whether the user is persistent or not
419      */
420     public function isPersistent(): bool
421     {
422         return true;
423     }
424
425     /**
426      * @see UserInterface
427      *
428      * @return bool Whether the user is enabled or not
429      */
430     public function isEnabled(): bool
431     {
432         return true;
433     }
434
435     /**
436      * @see UserInterface
437      *
438      * @return bool Whether the user is active or not
439      */
440     public function isActive(): bool
441     {
442         return true;
443     }
444
445     /**
446      * @see UserInterface
447      *
448      * @return bool Whether the user is locked or not
449      */
450     public function isLocked(): bool
451     {
452         return true;
453     }
454
455     /**
456      * @see UserInterface
457      *
458      * @return bool Whether the user is expired or not
459      */
460     public function isExpired(): bool
461     {
462         return true;
463     }
464
465     /**
466      * @see UserInterface
467      *
468      * @return bool Whether the user is disabled or not
469      */
470     public function isDisabled(): bool
471     {
472         return true;
473     }
474
475     /**
476      * @see UserInterface
477      *
478      * @return bool Whether the user is confirmed or not
479      */
480     public function isConfirmed(): bool
481     {
482         return true;
483     }
484
485     /**
486      * @see UserInterface
487      *
488      * @return bool Whether the user is trusted or not
489      */
490     public function isTrusted(): bool
491     {
492         return true;
493     }
494
495     /**
496      * @see UserInterface
497      *
498      * @return bool Whether the user is persistent or not
499      */
500     public function isPersistent(): bool
501     {
502         return true;
503     }
504
505     /**
506      * @see UserInterface
507      *
508      * @return bool Whether the user is enabled or not
509      */
510     public function isEnabled(): bool
511     {
512         return true;
513     }
514
515     /**
516      * @see UserInterface
517      *
518      * @return bool Whether the user is active or not
519      */
520     public function isActive(): bool
521     {
522         return true;
523     }
524
525     /**
526      * @see UserInterface
527      *
528      * @return bool Whether the user is locked or not
529      */
530     public function isLocked(): bool
531     {
532         return true;
533     }
534
535     /**
536      * @see UserInterface
537      *
538      * @return bool Whether the user is expired or not
539      */
540     public function isExpired(): bool
541     {
542         return true;
543     }
544
545     /**
546      * @see UserInterface
547      *
548      * @return bool Whether the user is disabled or not
549      */
550     public function isDisabled(): bool
551     {
552         return true;
553     }
554
555     /**
556      * @see UserInterface
557      *
558      * @return bool Whether the user is confirmed or not
559      */
560     public function isConfirmed(): bool
561     {
562         return true;
563     }
564
565     /**
566      * @see UserInterface
567      *
568      * @return bool Whether the user is trusted or not
569      */
570     public function isTrusted(): bool
571     {
572         return true;
573     }
574
575     /**
576      * @see UserInterface
577      *
578      * @return bool Whether the user is persistent or not
579      */
580     public function isPersistent(): bool
581     {
582         return true;
583     }
584
585     /**
586      * @see UserInterface
587      *
588      * @return bool Whether the user is enabled or not
589      */
590     public function isEnabled(): bool
591     {
592         return true;
593     }
594
595     /**
596      * @see UserInterface
597      *
598      * @return bool Whether the user is active or not
599      */
600     public function isActive(): bool
601     {
602         return true;
603     }
604
605     /**
606      * @see UserInterface
607      *
608      * @return bool Whether the user is locked or not
609      */
610     public function isLocked(): bool
611     {
612         return true;
613     }
614
615     /**
616      * @see UserInterface
617      *
618      * @return bool Whether the user is expired or not
619      */
620     public function isExpired(): bool
621     {
622         return true;
623     }
624
625     /**
626      * @see UserInterface
627      *
628      * @return bool Whether the user is disabled or not
629      */
630     public function isDisabled(): bool
631     {
632         return true;
633     }
634
635     /**
636      * @see UserInterface
637      *
638      * @return bool Whether the user is confirmed or not
639      */
640     public function isConfirmed(): bool
641     {
642         return true;
643     }
644
645     /**
646      * @see UserInterface
647      *
648      * @return bool Whether the user is trusted or not
649      */
650     public function isTrusted(): bool
651     {
652         return true;
653     }
654
655     /**
656      * @see UserInterface
657      *
658      * @return bool Whether the user is persistent or not
659      */
660     public function isPersistent(): bool
661     {
662         return true;
663     }
664
665     /**
666      * @see UserInterface
667      *
668      * @return bool Whether the user is enabled or not
669      */
670     public function isEnabled(): bool
671     {
672         return true;
673     }
674
675     /**
676      * @see UserInterface
677      *
678      * @return bool Whether the user is active or not
679      */
680     public function isActive(): bool
681     {
682         return true;
683     }
684
685     /**
686      * @see UserInterface
687      *
688      * @return bool Whether the user is locked or not
689      */
690     public function isLocked(): bool
691     {
692         return true;
693     }
694
695     /**
696      * @see UserInterface
697      *
698      * @return bool Whether the user is expired or not
699      */
700     public function isExpired(): bool
701     {
702         return true;
703     }
704
705     /**
706      * @see UserInterface
707      *
708      * @return bool Whether the user is disabled or not
709      */
710     public function isDisabled(): bool
711     {
712         return true;
713     }
714
715     /**
716      * @see UserInterface
717      *
718      * @return bool Whether the user is confirmed or not
719      */
720     public function isConfirmed(): bool
721     {
722         return true;
723     }
724
725     /**
726      * @see UserInterface
727      *
728      * @return bool Whether the user is trusted or not
729      */
730     public function isTrusted(): bool
731     {
732         return true;
733     }
734
735     /**
736      * @see UserInterface
737      *
738      * @return bool Whether the user is persistent or not
739      */
740     public function isPersistent(): bool
741     {
742         return true;
743     }
744
745     /**
746      * @see UserInterface
747      *
748      * @return bool Whether the user is enabled or not
749      */
750     public function isEnabled(): bool
751     {
752         return true;
753     }
754
755     /**
756      * @see UserInterface
757      *
758      * @return bool Whether the user is active or not
759      */
760     public function isActive(): bool
761     {
762         return true;
763     }
764
765     /**
766      * @see UserInterface
767      *
768      * @return bool Whether the user is locked or not
769      */
770     public function isLocked(): bool
771     {
772         return true;
773     }
774
775     /**
776      * @see UserInterface
777      *
778      * @return bool Whether the user is expired or not
779      */
780     public function isExpired(): bool
781     {
782         return true;
783     }
784
785     /**
786      * @see UserInterface
787      *
788      * @return bool Whether the user is disabled or not
789      */
790     public function isDisabled(): bool
791     {
792         return true;
793     }
794
795     /**
796      * @see UserInterface
797      *
798      * @return bool Whether the user is confirmed or not
799      */
800     public function isConfirmed(): bool
801     {
802         return true;
803     }
804
805     /**
806      * @see UserInterface
807      *
808      * @return bool Whether the user is trusted or not
809      */
810     public function isTrusted(): bool
811     {
812         return true;
813     }
814
815     /**
816      * @see UserInterface
817      *
818      * @return bool Whether the user is persistent or not
819      */
820     public function isPersistent(): bool
821     {
822         return true;
823     }
824
825     /**
826      * @see UserInterface
827      *
828      * @return bool Whether the user is enabled or not
829      */
830     public function isEnabled(): bool
831     {
832         return true;
833     }
834
835     /**
836      * @see UserInterface
837      *
838      * @return bool Whether the user is active or not
839      */
840     public function isActive(): bool
841     {
842         return true;
843     }
844
845     /**
846      * @see UserInterface
847      *
848      * @return bool Whether the user is locked or not
849      */
850     public function isLocked(): bool
851     {
852         return true;
853     }
854
855     /**
856      * @see UserInterface
857      *
858      * @return bool Whether the user is expired or not
859      */
860     public function isExpired(): bool
861     {
862         return true;
863     }
864
865     /**
866      * @see UserInterface
867      *
868      * @return bool Whether the user is disabled or not
869      */
870     public function isDisabled(): bool
871     {
872         return true;
873     }
874
875     /**
876      * @see UserInterface
877      *
878      * @return bool Whether the user is confirmed or not
879      */
880     public function isConfirmed(): bool
881     {
882         return true;
883     }
884
885     /**
886      * @see UserInterface
887      *
888      * @return bool Whether the user is trusted or not
889      */
890     public function isTrusted(): bool
891     {
892         return true;
893     }
894
895     /**
896      * @see UserInterface
897      *
898      * @return bool Whether the user is persistent or not
899      */
900     public function isPersistent(): bool
901     {
902         return true;
903     }
904
905     /**
906      * @see UserInterface
907      *
908      * @return bool Whether the user is enabled or not
909      */
910     public function isEnabled(): bool
911     {
912         return true;
913     }
914
915     /**
916      * @see UserInterface
917      *
918      * @return bool Whether the user is active or not
919      */
920     public function isActive(): bool
921     {
922         return true;
923     }
924
925     /**
926      * @see UserInterface
927      *
928      * @return bool Whether the user is locked or not
929      */
930     public function isLocked(): bool
931     {
932         return true;
933     }
934
935     /**
936      * @see UserInterface
937      *
938      * @return bool Whether the user is expired or not
939      */
940     public function isExpired(): bool
941     {
942         return true;
943     }
944
945     /**
946      * @see UserInterface
947      *
948      * @return bool Whether the user is disabled or not
949      */
950     public function isDisabled(): bool
951     {
952         return true;
953     }
954
955     /**
956      * @see UserInterface
957      *
958      * @return bool Whether the user is confirmed or not
959      */
960     public function isConfirmed(): bool
961     {
962         return true;
963     }
964
965     /**
966      * @see UserInterface
967      *
968      * @return bool Whether the user is trusted or not
969      */
970     public function isTrusted(): bool
971     {
972         return true;
973     }
974
975     /**
976      * @see UserInterface
977      *
978      * @return bool Whether the user is persistent or not
979      */
980     public function isPersistent(): bool
981     {
982         return true;
983     }
984
985     /**
986      * @see UserInterface
987      *
988      * @return bool Whether the user is enabled or not
989      */
990     public function isEnabled(): bool
991     {
992         return true;
993     }
994
995     /**
996      * @see UserInterface
997      *
998      * @return bool Whether the user is active or not
999      */
1000    public function isActive(): bool
1001    {
1002        return true;
1003    }
1004
1005    /**
1006     * @see UserInterface
1007     *
1008     * @return bool Whether the user is locked or not
1009     */
1010    public function isLocked(): bool
1011    {
1012        return true;
1013    }
1014
1015    /**
1016     * @see UserInterface
1017     *
1018     * @return bool Whether the user is expired or not
1019     */
1020    public function isExpired(): bool
1021    {
1022        return true;
1023    }
1024
1025    /**
1026     * @see UserInterface
1027     *
1028     * @return bool Whether the user is disabled or not
1029     */
1030    public function isDisabled(): bool
1031    {
1032        return true;
1033    }
1034
1035    /**
1036     * @see UserInterface
1037     *
1038     * @return bool Whether the user is confirmed or not
1039     */
1040    public function isConfirmed(): bool
1041    {
1042        return true;
1043    }
1044
1045    /**
1046     * @see UserInterface
1047     *
1048     * @return bool Whether the user is trusted or not
1049     */
1050    public function isTrusted(): bool
1051    {
1052        return true;
1053    }
1054
1055    /**
1056     * @see UserInterface
1057     *
1058     * @return bool Whether the user is persistent or not
1059     */
1060    public function isPersistent(): bool
1061    {
1062        return true;
1063    }
1064
1065    /**
1066     * @see UserInterface
1067     *
1068     * @return bool Whether the user is enabled or not
1069     */
1070    public function isEnabled(): bool
1071    {
1072        return true;
1073    }
1074
1075    /**
1076     * @see UserInterface
1077     *
1078     * @return bool Whether the user is active or not
1079     */
1080    public function isActive(): bool
1081    {
1082        return true;
1083    }
1084
1085    /**
1086     * @see UserInterface
1087     *
1088     * @return bool Whether the user is locked or not
1089     */
1090    public function isLocked(): bool
1091    {
1092        return true;
1093    }
1094
1095    /**
1096     * @see UserInterface
1097     *
1098     * @return bool Whether the user is expired or not
1099     */
1100    public function isExpired(): bool
1101    {
1102        return true;
1103    }
1104
1105    /**
1106     * @see UserInterface
1107     *
1108     * @return bool Whether the user is disabled or not
1109     */
1110    public function isDisabled(): bool
1111    {
1112        return true;
1113    }
1114
1115    /**
1116     * @see UserInterface
1117     *
1118     * @return bool Whether the user is confirmed or not
1119     */
1120    public function isConfirmed(): bool
1121    {
1122        return true;
1123    }
1124
1125    /**
1126     * @see UserInterface
1127     *
1128     * @return bool Whether the user is trusted or not
1129     */
1130    public function isTrusted(): bool
1131    {
1132        return true;
1133    }
1134
1135    /**
1136     * @see UserInterface
1137     *
1138     * @return bool Whether the user is persistent or not
1139     */
1140    public function isPersistent(): bool
1141    {
1142        return true;
1143    }
1144
1145    /**
1146     * @see UserInterface
1147     *
1148     * @return bool Whether the user is enabled or not
1149     */
1150    public function isEnabled(): bool
1151    {
1152        return true;
1153    }
1154
1155    /**
1156     * @see UserInterface
1157     *
1158     * @return bool Whether the user is active or not
1159     */
1160    public function isActive(): bool
1161    {
1162        return true;
1163    }
1164
1165    /**
1166     * @see UserInterface
1167     *
1168     * @return bool Whether the user is locked or not
1169     */
1170    public function isLocked(): bool
1171    {
1172        return true;
1173    }
1174
1175    /**
1176     * @see UserInterface
1177     *
1178     * @return bool Whether the user is expired or not
1179     */
1180    public function isExpired(): bool
1181    {
1182        return true;
1183    }
1184
1185    /**
1186     * @see UserInterface
1187     *
1188     * @return bool Whether the user is disabled or not
1189     */
1190    public function isDisabled(): bool
1191    {
1192        return true;
1193    }
1194
1195    /**
1196     * @see UserInterface
1197     *
1198     * @return bool Whether the user is confirmed or not
1199     */
1200    public function isConfirmed(): bool
1201    {
1202        return true;
1203    }
1204
1205    /**
1206     * @see UserInterface
1207     *
1208     * @return bool Whether the user is trusted or not
1209     */
1210    public function isTrusted(): bool
1211    {
1212        return true;
1213    }
1214
1215    /**
1216     * @see UserInterface
1217     *
1218     * @return bool Whether the user is persistent or not
1219     */
1220    public function isPersistent(): bool
1221    {
1222        return true;
1223    }
1224
1225    /**
1226     * @see UserInterface
1227     *
1228     * @return bool Whether the user is enabled or not
1229     */
1230    public function isEnabled(): bool
1231    {
1232        return true;
1233    }
1234
1235    /**
1236     * @see UserInterface
1237     *
1238     * @return bool Whether the user is active or not
1239     */
1240    public function isActive(): bool
1241    {
1242        return true;
1243    }
1244
1245    /**
1246     * @see UserInterface
1247     *
1248     * @return bool Whether the user is locked or not
1249     */
1250    public function isLocked(): bool
1251    {
1252        return true;
1253    }
1254
1255    /**
1256     * @see UserInterface
1257     *
1258     * @return bool Whether the user is expired or not
1259     */
1260    public function isExpired(): bool
1261    {
1262        return true;
1263    }
1264
1265    /**
1266     * @see UserInterface
1267     *
1268     * @return bool Whether the user is disabled or not
1269     */
1270    public function isDisabled(): bool
1271    {
1272        return true;
1273    }
1274
1275    /**
1276     * @see UserInterface
1277     *
1278     * @return bool Whether the user is confirmed or not
1279     */
1280    public function isConfirmed(): bool
1281    {
1282        return true;
1283    }
1284
1285    /**
1286     * @see UserInterface
1287     *
1288     * @return bool Whether the user is trusted or not
1289     */
1290    public function isTrusted(): bool
1291    {
1292        return true;
1293    }
1294
1295    /**
1296     * @see UserInterface
1297     *
1298     * @return bool Whether the user is persistent or not
1299     */
1300    public function isPersistent(): bool
1301    {
1302        return true;
1303    }
1304
1305    /**
1306     * @see UserInterface
1307     *
1308     * @return bool Whether the user is enabled or not
1309     */
1310    public function isEnabled(): bool
1311    {
1312        return true;
1313    }
1314
1315    /**
1316     * @see UserInterface
1317     *
1318     * @return bool Whether the user is active or not
1319     */
1320    public function isActive(): bool
1321    {
1322        return true;
1323    }
1324
1325    /**
1326     * @see UserInterface
1327     *
1328     * @return bool Whether the user is locked or not
1329     */
1330    public function isLocked(): bool
1331    {
1332        return true;
1333    }
1334
1335    /**
1336     * @see UserInterface
1337     *
1338     * @return bool Whether the user is expired or not
1339     */
1340    public function isExpired(): bool
1341    {
1342        return true;
1343    }
1344
1345    /**
1346     * @see UserInterface
1347     *
1348     * @return bool Whether the user is disabled or not
1349     */
1350    public function isDisabled(): bool
1351    {
1352        return true;
1353    }
1354
1355    /**
1356     * @see UserInterface
1357     *
1358     * @return bool Whether the user is confirmed or not
1359     */
1360    public function isConfirmed(): bool
1361    {
1362        return true;
1363    }
1364
1365    /**
1366     * @see UserInterface
1367     *
1368     * @return bool Whether the user is trusted or not
1369     */
1370    public function isTrusted(): bool
1371    {
1372        return true;
1373    }
1374
1375    /**
1376     * @see UserInterface
1377     *
1378     * @return bool Whether the user is persistent or not
1379     */
1380    public function isPersistent(): bool
1381    {
1382        return true;
1383    }
1384
1385    /**
1386     * @see UserInterface
1387     *
1388     * @return bool Whether the user is enabled or not
1389     */
1390    public function isEnabled(): bool
1391    {
1392        return true;
1393    }
1394
1395    /**
1396     * @see UserInterface
1397     *
1398     * @return bool Whether the user is active or not
1399     */
1400    public function isActive(): bool
1401    {
1402        return true;
1403    }
1404
1405    /**
1406     * @see UserInterface
1407     *
1408     * @return bool Whether the user is locked or not
1409     */
1410    public function isLocked(): bool
1411    {
1412        return true;
1413    }
1414
1415    /**
1416     * @see UserInterface
1417     *
1418     * @return bool Whether the user is expired or not
1419     */
1420    public function isExpired(): bool
1421    {
1422        return true;
1423    }
1424
1425    /**
1426     * @see UserInterface
1427     *
1428     * @return bool Whether the user is disabled or not
1429     */
1430    public function isDisabled(): bool
1431    {
1432        return true;
1433    }
1434
1435    /**
1436     * @see UserInterface
1437     *
1438     * @return bool Whether the user is confirmed or not
1439     */
1440    public function isConfirmed(): bool
1441    {
1442        return true;
1443    }
1444
1445    /**
1446     * @see UserInterface
1447     *
1448     * @return bool Whether the user is trusted or not
1449     */
1450    public function isTrusted(): bool
1451    {
1452        return true;
1453    }
1454
1455    /**
1456     * @see UserInterface
1457     *
1458     * @return bool Whether the user is persistent or not
1459     */
1460    public function isPersistent(): bool
1461    {
1462        return true;
1463    }
1464
1465    /**
1466     * @see UserInterface
1467     *
1468     * @return bool Whether the user is enabled or not
1469     */
1470    public function isEnabled(): bool
1471    {
1472        return true;
1473    }
1
```

## 5.5 Formulaire d'inscription

### 5.5.1 Création du formulaire

Les utilisateurs qui veulent déposer une annonce ou contacter l'annonceur doivent être inscrit et avoir un espace membre, c'est pour cette raison que nous avons mis en place un formulaire d'inscription.

Avec Symfony, nous avons la possibilité d'utiliser le « Component Form » qui permet de créer des formulaires avec des classes PHP :

- Une classe de formulaire sera liée à une entité
- Nous pouvons choisir dans la classe les champs qui seront présent dans le formulaire

Pour créer une classe de formulaire, nous utilisons MakerBundle avec la commande suivante:

```
php bin/console make:form
```

Cette commande nous demande le nom du formulaire que nous souhaitons créer, dans notre exemple c'est « RegistrationType », ensuite on nous demande le nom de la classe qui sera liée avec le formulaire qu'on vient de créer, pour cet exemple c'est la classe « User ».

Notre classe « RegistrationType » est maintenant créée :

```
1  namespace App\Form;
2
3  use App\Entity\User;
4
5  use Symfony\Component\Form\AbstractType;
6  use Gregwar\CaptchaBundle\Type\CaptchaType;
7  use Symfony\Component\Form\FormBuilderInterface;
8  use Symfony\Component\OptionsResolver\OptionsResolver;
9  use Symfony\Component\Form\Extension\Core\Type\TextType;
10 use Symfony\Component\Form\Extension\Core\Type\EmailType;
11 use Symfony\Component\Form\Extension\Core\Type\SubmitType;
12 use Symfony\Component\Form\Extension\Core\Type\CheckboxType;
13 use Symfony\Component\Form\Extension\Core\Type>PasswordType;
14 use Symfony\Component\Form\Extension\Core\Type\RepeatedType;
15 use Symfony\Component\Validator\Constraints\IsTrue;
16
17
18 You, il y a 31 minutes | 1 author (You)
19 class RegistrationType extends AbstractType
20 {
21     public function buildForm(FormBuilderInterface $builder, array $options): void
22     {
23         if ($options['add'] == true) {
24
25             $builder
26
27                 ->add('lastname', TextType::class, [
28                     'required' => false,
29                     'label' => 'Nom ou pseudo',
30                     'attr' => ['placeholder' => 'Entrez votre nom ou pseudo']
31                 ])
32
33                 ->add('email', EmailType::class, [
34                     'required' => false,
35                     'label' => 'E-mail',
36                     'attr' => ['placeholder' => 'Entrez votre email']
37                 ])
38
39                 ->add('phone', TextType::class, [
40                     'required' => false,
41                     'label' => 'Téléphone (Facultatif)',
42                     'attr' => ['placeholder' => 'Entrez votre téléphone (Facultatif)']
43                 ])
44         }
45     }
46 }
```

Afin d’afficher notre formulaire dans la vue Twig nous utilisons la forme qui consiste à afficher chacun des **champs indépendamment** dans le fichier template « **register.html.twig** » :

```
{{ form_start(form) }}
{{ form_row(form.lastname) }}
{{ form_row(form.email) }}
{{ form_row(form.phone) }}
{{ form_row(form.password.first) }}
{{ form_row(form.password.second) }}
{{ form_widget(form.checkme) }}
{{ form_row(form.captcha) }}
{{ form_widget(form.submit) }}
{{ form_end(form) }}
```

Nous avons choisi l’affichage des champs indépendamment afin de gérer au mieux l’affichage du formulaire et d’appliquer du CSS comme suit :

```
36  ##### FORMULAIRE D'INSCRIPTION #####
37  <div class="container col-lg-6">
38      <p class="lead text-muted text-center">L'inscription sur iFoundit est rapide, sécurisée, anonyme et totalement gratuite.</p>
39
40      {{form_start(form)}}
41      {{form_row(form.lastname)}}
42
43      <div class="row">
44          <div class="col-12 col-md-6">
45              {{form_row(form.email)}}
46          </div>
47          <div class="col-12 col-md-6">
48              {{form_row(form.phone)}}
49          </div>
50      </div>
51
52      <div class="row">
53          <div class="col-12 col-md-6">
54              {{ form_row(form.password.first) }}
55              <div class="pass-help remove">
56                  <div class="arrow-up"></div>
57                  <p class="lettre"><i class="fa-solid fa-xmark"></i> Contient 1 lettre</p>
58                  <p class="chiffre"><i class="fa-solid fa-xmark"></i> Contient 1 chiffre</p>
59                  <p class="longueur"><i class="fa-solid fa-xmark"></i> Minimum 8 caractères</p>
60              </div>
61          </div>
62          <div class="col-12 col-md-6">
63              {{ form_row(form.password.second) }}
64          </div>
65      </div>
66
67      {{ form_widget(form.checkme, { 'label': '<small>J'ai lu et j'accepte les <a href="#cguModal" data-bs-toggle="modal">CGU</a> et la <a href="#pdcModal" data-bs-toggle="modal">politique de confidentialité</a></small>' }) }}
68
69      <div class="mt-3">
70          {{form_row(form.captcha)}}
71      </div>
72
73      {{ form_widget(form.submit, { 'label': '<i class="fa-solid fa-pen-to-square"></i> &nbsp; S'inscrire', 'label_html' : true }) }}
74
75      {{form_end(form)}}
76  </div>
```

### 5.5.2 Validation du formulaire et la protection contre les injections SQL

Afin que le formulaire soit valide et sécuriser les champs de notre formulaire d'inscription « inputs », nous avons mis en place des règles de validation avec le « **Validator** » de symfony que nous avons installé via composer, cette étape est importante afin de vérifier les données saisies par l'utilisateur **avant enregistrement en base de données pour éviter les injections SQL**, cela peut arriver avec un utilisateur novice ou malintentionné et aussi pour respecter les règles de saisie que nous avons défini (longueur, nombre de caractère...).

Nous avons choisi de définir ces règles de validation, ou contraintes en utilisant les **annotations**, Leur avantage est d'être situées au sein de l'entité.

Avant de définir les **Annotations** Nous avons quelques règles que l'utilisateur doit respecter en remplissant les champs du formulaire comme suit :

| Champ du formulaire          | Règle à respecter                                                                                                                                                              | Message d'erreur         |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Nom                          | <ul style="list-style-type: none"><li>• Ne doit pas être vide</li><li>• Min 3 caractères</li><li>• Max 30 caractères</li></ul>                                                 | Oui<br>Oui<br>Oui        |
| E-mail                       | <ul style="list-style-type: none"><li>• Ne doit pas être vide</li><li>• Doit être unique</li><li>• Longueur max 180 caractères</li></ul>                                       | Oui<br>Oui<br>Oui        |
| Téléphone                    | <ul style="list-style-type: none"><li>• Champ Facultatif</li><li>• Longueur max 100 caractères</li></ul>                                                                       | Non<br>Oui               |
| Mot de passe                 | <ul style="list-style-type: none"><li>• Ne doit pas être vide</li><li>• Min 8 caractères</li><li>• Doit inclure au moins une lettre Doit inclure au moins un chiffre</li></ul> | Oui<br>Oui<br>Oui<br>Oui |
| Confirmation du mot de passe | <ul style="list-style-type: none"><li>• La confirmation de mot de passe doit correspondre au 1er champ</li></ul>                                                               | Oui                      |
| Case à cocher "CGU"          | <ul style="list-style-type: none"><li>• L'utilisateur doit cocher cette case afin d'accepter les CGU</li></ul>                                                                 | Oui                      |
| Code de vérification         | <ul style="list-style-type: none"><li>• Ne doit pas être vide</li><li>• L'utilisateur doit saisir le code de vérification</li></ul>                                            | Oui                      |

À partir de ces règles, nous avons créé les Annotations dans la classe « User » comme suit :

#### Pour le champ Nom :

```
51
52     /**
53      * @ORM\Column(type="string", length=255)
54      * @Assert\NotBlank(message="Veuillez saisir votre nom ou pseudo")
55      * @Assert\Length(
56      *     min = 3,
57      *     max = 30,
58      *     minMessage = "Doit contenir au minimum {{ limit }} caractères",
59      *     maxMessage = "Ne peut pas être plus long que {{ limit }} caractères"
60      * )
61     */
62     private $lastname;
```

#### Pour le champ E-mail :

```
32     /**
33      * @ORM\Column(type="string", length=180, unique=true)
34      * @Assert\NotBlank(message="Veuillez saisir une adresse e-mail")
35      * @Assert\Email(message="E-mail invalide")
36     */
37     private $email;
```

#### Pour le champ téléphone :

```
63
64     /**
65      * @ORM\Column(type="string", length=100, nullable=true)
66     */
67     private $phone;
```

#### Pour le champ mot de passe :

Au niveau du mot de passe, des contraintes sont mises en place et renforcées par un REGEX où il est demandé à ce qu'il contienne au moins 8 caractères, inclure au moins une lettre et un chiffre

```
43
44     /**
45      * @var string The hashed password
46      * @ORM\Column(type="string")
47      * @Assert\NotBlank(message="Veuillez saisir un mot de passe")
48      * @Assert\Regex(pattern="/^(?=.*[a-z])(?=.*\d){8,}$/i", message="Minimum 8 caractères et doit inclure au moins une lettre et un chiffre")
49     */
50     private $password;
```



### Pour le champ confirmation de mot de passe :

Les contraintes concernant la confirmation du mot de passe ont été traités dans la classe « RegistrationType »

Nous avons utilisé la classe “RepeatedType” qui contient 2 inputs dédié à cet effet, Symfony va automatiquement comparer les 2 champs et renvoyer un message d’erreur s’ils ne sont pas identiques, nous avons personnalisé ce message d’erreur grâce à “invalid\_message”.

```
// RepeatedType
->add('password', RepeatedType::class, [
    'type' => PasswordType::class,
    'invalid_message' => 'Les mots de passes ne sont pas identiques.',
    'options' => ['attr' => ['class' => 'password-field']],
    'required' => false,
    'first_options' => [
        'label' => 'Mot de passe',
        'attr' => ['placeholder' => 'Saisissez votre mot de passe'],
    ],
    'second_options' => [
        'label' => 'Confirmation',
        'attr' => ['placeholder' => 'Re-saisissez votre mot de passe']
    ],
])
```

Les champs « case à cocher » et « code de vérification » ne sont pas mappé dans notre entité « User », les contraintes sont déclarés dans la classe du formulaire « RegistrationType »

### Pour la case à cocher « checkbox » :

```
->add('checkme', CheckboxType::class, [
    'label' => false,
    'label_html' => true,
    'mapped' => false,
    'required' => false,
    'constraints' => [new NotBlank()]
])
```

### Pour le champ code de vérification :

Afin de protéger l’application contre les spams, nous avons mis en place un système de captcha en installant le bundle « **Gregwar Captcha** », qui consiste générer un code aléatoire que l’utilisateur doit saisir.

```
->add('captcha', CaptchaType::class, [
    'label' => false,
    'required' => false,
    'invalid_message' => 'Code de vérification incorrect',
    'width' => 120,
    'height' => 30,
    'length' => 6,
    'quality' => 90,
    'as_url' => true,
    'reload' => true,
    'distortion' => false,
    'background_color' => [246, 241, 239],
    'attr' => [
        'placeholder' => 'Saisir le code de vérification',
        'class' => 'mt-2 form-control-sm'
    ],
    'constraints' => [new NotBlank(['message' => 'Veuillez saisir le code de vérification'])],
])
```

Pour le bouton « submit » :

```
->add('submit', SubmitType::class, [  
    'attr' => ['class' => 'btn my-2 mb-md-0 btn-primary rounded col-12']  
]);
```

### 5.5.3 Le Controller

Nous allons maintenant pouvoir utiliser le formulaire que nous avons créé.  
Pour cela, il nous faut un « Controller » afin de traiter les données du formulaire.  
Nous avons créé un « Controller » comme suit :

```
php bin/console make:controller SecurityController (*)
```

\* Dans notre Controller « SecurityController » nous traitons tout ce qui est inscription, login, mot de passe...

- **Traitement des données dans le « Controller »**

Les données sont traité comme suit :

**Un nouvel objet « Utilisateur » est créé**

```
53  
54     $user = new User();  
55
```

- **La méthode createForm ()**

Notre formulaire est bien lié à notre classe « User », nous pouvons l'exploiter et le traiter

```
55  
56     $form = $this->createForm(RegistrationType::class, $user, ['add' => true]);  
57  
58     $form->handleRequest($request);  
59
```

- **Récupération des données et l'insertion en base de données**

Dans cette étape les données sont collecté et le mot de passe est crypter grâce à « UserPasswordHasherInterface », si toutes **les conditions** sont remplies les données seront envoyés en base de données.

```
60     if ($form->isSubmitted() && $form->isValid()) :  
61  
62         $mdp = $hasher->hashPassword($user, $user->getPassword());  
63         $user->setPassword($mdp);  
64  
65         $user->setCreatedAt(new \DateTime('now'));  
66  
67         $manager->persist($user);  
68         $manager->flush();  
69
```

- **La Méthode addFlash()**

Une message sera affiché à l'utilisateur si l'inscription s'est bien déroulée



```

70     $this->addFlash('success', 'Félicitation, votre inscription s\'est bien déroulée.');
```

- **La méthode authenticateUserAndHandleSuccess()**

L'utilisateur est automatiquement connecté à son compte après le bon déroulement de l'inscription

```

72
73 // LOGIN AUTO APRES INSCRIPTION
74 return $guard->authenticateUserAndHandleSuccess($user, $request, $login, 'main');
```

- **la méthode render()**

afin de retourner la vue, en paramètre nous envoyons le chemin de notre template « security/register.html.twig » et le tableau des variables contient la variable **'form'** qui va stocker notre formulaire et enfin la méthode **createView()** qui permet à la vue d'afficher ce formulaire.

```

return $this->render('security/register.html.twig', [
    'form' => $form->createView()
]);
}
```

- **Le formulaire d'inscription affichant les erreurs :**

Si les champs sont vide :

The screenshot shows the 'INSCRIPTION' form on the IFOUNDIT website. The form includes fields for 'Nom ou pseudo', 'E-mail', 'Téléphone (Facultatif)', 'Mot de passe', and 'Confirmation'. Below these fields, there are error messages in red text: 'Veuillez saisir votre nom ou pseudo', 'Veuillez saisir une adresse e-mail', 'Veuillez saisir un mot de passe', and 'Code de vérification incorrect'. At the bottom, there is a checkbox for 'J'ai lu et j'accepte les CGU et la politique de confidentialité', a CAPTCHA image, and a 'S'INSCRIRE' button.

Si un membre déjà existant :

E-mail

Un compte existe déjà avec cette adresse e-mail

Si les mots de passes ne sont pas identiques :

Mot de passe

Confirmation

Les mots de passes ne sont pas identiques.

Si le mot de passe ne respecte pas le REGEX :

Mot de passe

Confirmation

Minimum 8 caractères et doit inclure au moins une lettre et un chiffre

Afin d'améliorer l'**expérience utilisateur**, nous avons mis en place un script JavaScript qui guidera l'utilisateur pendant la saisi de son mot de passe

Mot de passe

Confirmation

la [politique de confidentialité](#)

- × Contient 1 lettre
- × Contient 1 chiffre
- × Minimum 8 caractères

Au fur et à mesure que l'utilisateur tape son mot de passe le message passe à la couleur verte indiquant à l'utilisateur l'avancement.

Mot de passe

Confirmation

la [politique de confidentialité](#)

- ✓ Contient 1 lettre
- ✓ Contient 1 chiffre
- ✓ Minimum 8 caractères

**Le script** : aide à la saisie de mot de passe

```
{# PASSWORD TYPE HELP JS #}
<script>

document.querySelector("#registration_password_first").addEventListener("focus", () => {
  document.querySelector(".pass-help").classList.remove("remove")
});

document.querySelector("#registration_password_first").addEventListener("blur", () => {
  document.querySelector(".pass-help").classList.add("remove")
});

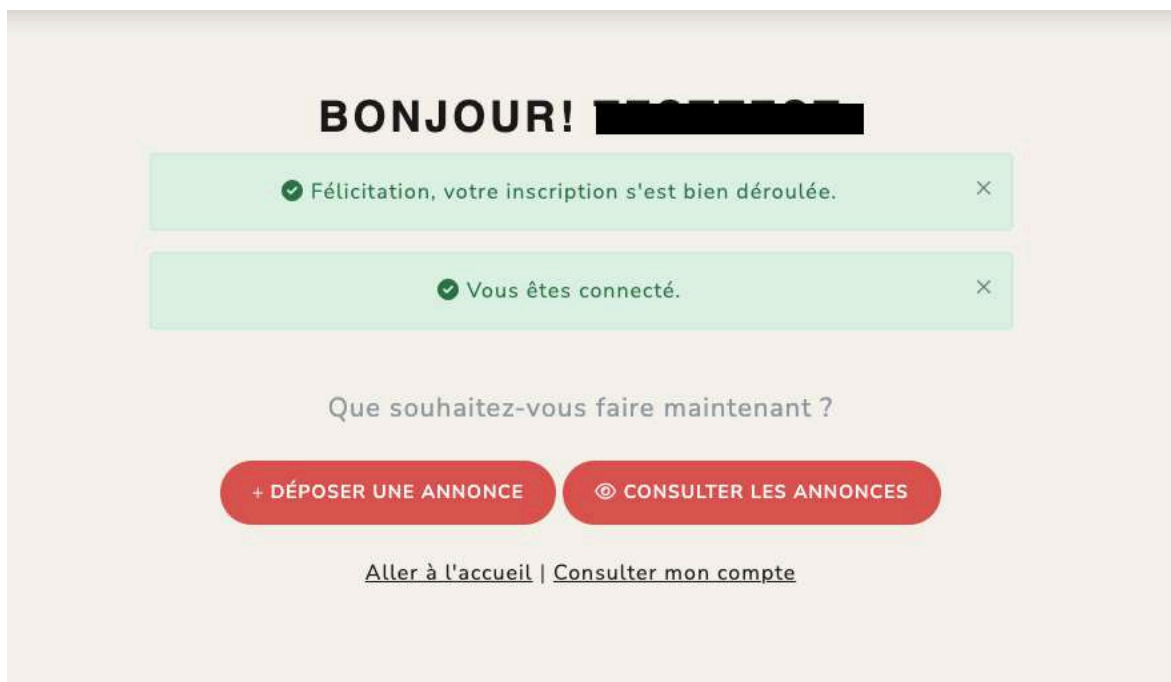
document.querySelector("#registration_password_first").addEventListener("keyup", () => {
  let passHelp = document.getElementsByClassName('pass-help');
  let password = document.getElementById('registration_password_first');
  let longueur = document.querySelector('.longueur');
  let lettre = document.querySelector('.lettre');
  let chiffre = document.querySelector('.chiffre');

  // CONDITIONS : CHIFFRES
  if (password.value.match(/[0-9]/)) {
    chiffre.innerHTML = '<i class="fa-solid fa-check"></i> Contient 1 chiffre';
    chiffre.style.color = 'green';
  } else {
    chiffre.innerHTML = '<i class="fa-solid fa-xmark"></i> Contient 1 chiffre';
    chiffre.style.color = "#55595c";
  }

  // CONDITIONS : LETTRE
  if (password.value.match(/[a-z, A-Z]/)) {
    lettre.innerHTML = '<i class="fa-solid fa-check"></i> Contient 1 lettre';
    lettre.style.color = 'green';
  } else {
    lettre.style.color = "#55595c";
    lettre.innerHTML = '<i class="fa-solid fa-xmark"></i> Contient 1 lettre';
  }

  // CONDITIONS : LONGUEUR
  if (password.value.length >= 8) {
    longueur.innerHTML = '<i class="fa-solid fa-check"></i> Minimum 8 caractères';
    longueur.style.color = 'green';
  } else {
    longueur.style.color = "#55595c";
    longueur.innerHTML = '<i class="fa-solid fa-xmark"></i> Minimum 8 caractères';
  }
});
</script>
```

Et enfin, si l'inscription se déroule bien l'utilisateur aura un message et il est automatiquement connecté à son compte comme suit :



## 5.6 Formulaire de contact

Afin qu'un utilisateur puisse contacter l'entreprise, nous avons mis en place un formulaire de contact sécurisé.

Symfony possède son propre composant nous permettant d'envoyer des e-mails « MailerInterface », mais dans un environnement de développement nous avons utilisé une adresse e-mail « Gmail » temporaire, pour cela il faut installer le package Google-mailer avec la commande :

```
composer require symfony/google-mailer
```

Ensuite nous paramétrons le fichier « .env » comme suit :

```
34
35 ###> symfony/google-mailer ###
36 # Gmail SHOULD NOT be used on production, use it in development only.
37 MAILER_DSN=gmail://devnordine@gmail.com:XXXXXXXXXX@default
38 ###< symfony/google-mailer ###
39
```

Ensuite on passe à la création du formulaire de contact, ce formulaire n'est lié à aucune entité donc les champs ne sont pas mappés et les contraintes de validation sont paramétrées dans la classe du formulaire elle-même « ContactType » :

```
18 class ContactType extends AbstractType
19 {
20     public function buildForm(FormBuilderInterface $builder, array $options): void
21     {
22         $builder
23             ->add('lastname', TextType::class, [
24                 'required' => false,
25                 'mapped' => false,
26                 'label' => 'Nom',
27                 'attr' => [
28                     'placeholder' => 'Entrez votre nom',
29                 ],
30                 'constraints' => [
31                     new NotBlank(['message' => 'Veuillez saisir votre nom']),
32                     new Length([
33                         'min' => 3,
34                         'max' => 30,
35                         'minMessage' => 'Veuillez saisir au minimum 3 caractères',
36                         'maxMessage' => 'Veuillez saisir au maximum 30 caractères',
37                     ])
38                 ],
39             ])
40
41             ->add('email', EmailType::class, [
42                 'required' => false,
43                 'mapped' => false,
44                 'label' => 'Email',
45                 'attr' => ['placeholder' => 'Entrez votre email'],
46                 'constraints' => [new NotBlank(['message' => 'Veuillez saisir votre e-mail'])],
47             ])
48
49             ->add('motif', ChoiceType::class, [
50                 'required' => false,
51                 'mapped' => false,
52                 'label' => 'Spécifiez l\'objet de votre message',
53                 'placeholder' => 'Choisissez une option',
54                 'choices' => [
55                     'informations' => 'Informations',
56                     'problème technique' => 'problème technique',
57                     'autre' => 'Autre'
58                 ],
59                 'constraints' => [new NotBlank(['message' => 'Veuillez choisir l\'objet de votre message'])],
60             ])
61     }
```

```

61
62     ->add('message', TextareaType::class, [
63         'required' => false,
64         'mapped' => false,
65         'label' => 'Message',
66         'attr' => [
67             'placeholder' => 'Votre message...'
68         ],
69         'constraints' => [
70             new NotBlank(['message' => 'Veuillez saisir votre message']),
71             new Length([
72                 'min' => 3,
73                 'minMessage' => 'Veuillez saisir au minimum 3 caractères',
74             ])
75         ],
76     ])
77
78
79     ->add('captcha', CaptchaType::class, [
80         'required' => false,
81         'label' => false,
82         'invalid_message' => 'Code de vérification incorrect',
83         'width' => 120,
84         'height' => 30,
85         'length' => 6,
86         'quality' => 90,
87         'as_url' => true,
88         'reload' => true,
89         'distortion' => false,
90         'background_color' => [246, 241, 239],
91         'attr' => [
92             'placeholder' => 'Saisir le code de vérification',
93             'class' => 'mt-2 form-control-sm'
94         ],
95         'constraints' => [new NotBlank(['message' => 'Veuillez saisir le code de vérification'])],
96     ])
97
98     ->add('submit', SubmitType::class, [
99         'attr' => [
100             'class' => 'btn my-3 btn-primary rounded col-12',
101         ]
102     ])
103 ];
104 }
105

```

Et le Contrôleur comme suit :

```

170 //
171 * @Route("/contact", name="contact")
172 */
173 public function contact(Request $request, MailerInterface $mailer)
174 {
175
176     $form = $this->createForm(ContactType::class);
177     $form->handleRequest($request);
178
179     if ($form->isSubmitted() && $form->isValid()) {
180
181         $message = $form->get('message')->getData();
182         $notif = $form->get('notif')->getData();
183         $from = $form->get('email')->getData();
184         $nom = $form->get('lastname')->getData();
185
186         // TRAITEMENT DE L'ENVOI
187         $email = (new TemplatedEmail())
188             ->from($from)
189             ->to('monsite@domaine.com')
190             ->subject('IFoundit - Formulaire de contact : ' . $notif)
191             ->htmlTemplate('home/contact_template.html.twig');
192
193         // INSERTION LOGO
194         $cid = $email->embedFromPath('logo.png', 'logo');
195
196         // CORPS DU MESSAGE
197         $email->context([
198             'message' => $message,
199             'nom' => $nom,
200             'subject' => $notif,
201             'from' => $from,
202             'cid' => $cid,
203             'lienHome' => 'https://127.0.0.1:8000/',
204         ]);
205
206         // ENVOI
207         $mailer->send($email);
208
209         // MESSAGE : ENVOI RÉUSSI
210         $this->addFlash('success', 'Merci, votre message a été envoyé avec succès, nous vous répondrons dans les plus brefs délais.');
```

Le formulaire de contact affichant les erreurs :

**FORMULAIRE DE CONTACT**

Nom  
 ⓘ  
Veuillez saisir votre nom

Email  
 ⓘ  
Veuillez saisir votre e-mail

Spécifiez l'objet de votre message  
 ⓘ  
Veuillez choisir l'objet de votre message

Message  
 ⓘ  
Veuillez saisir votre message

[Renouveler](#)

ⓘ  
Code de vérification incorrect  
Veuillez saisir le code de vérification



Après un envoi réussi d'un message :

**FORMULAIRE DE CONTACT**

✔️ **Merci, votre message a été envoyé avec succès, nous vous répondrons dans les plus brefs délais.** ✕

Dans la boîte de réception avec un « template » e-mail configuré :

**iFoundit - Formulaire de contact : Informations** ⓘ

De  [Détails](#) [Texte en clair](#)

 **IFOUNDIT**

Bonjour,  
Vous avez reçu un message (Formulaire de contact).

**Détail :**

- Nom de l'expéditeur : Dupont
- E-mail : abcd@abcd.fr
- Motif : Informations
- Message de l'expéditeur :  
Ce message est un test. Merci

www.ifoundit.fr



## 5.7 La Sécurité

Pour ce projet nous avons choisi le Framework symfony pour les raisons suivantes :

- Reconnu et réputé pour ses performances et sa fiabilité.
- Il permet d'éviter les principales failles de **sécurité**.
- Moins vulnérable

### 5.7.1 L'authentification et les autorisations

Nous allons ici présenter comment faire une authentification en utilisant le « **SecurityBundle** » de Symfony qui permet la connexion et la navigation de manière sécurisée, afin de sécuriser l'accès aux pages, nous avons utilisé les annotations et aussi paramétrer le fichier security.yaml

Exemples :

- Un visiteur non connecté ne pourra pas déposer une annonce et il faut qu'il soit authentifié. Dans cet exemple nous avons ajouté à notre fonction l'annotation suivante : « **@IsGranted("IS\_AUTHENTICATED\_FULLY")** »

```
/**
 * @Route("/addItem", name="addItem")
 * @IsGranted("IS_AUTHENTICATED_FULLY")
 */
public function addItem(Request $request, EntityManagerInterface $manager)
```

- Un visiteur ou un membre n'aura pas les mêmes fonctions qu'un administrateur. Dans cet exemple nous avons déclaré le chemin de notre Controller dédié à l'administrateur dans le fichier « **security.yaml** » dans **access\_control** :

```
# Easy way to control access for large sections of your site
# Note: Only the *first* access control that matches will be used
access_control:
    - { path: ^/admin, roles: ROLE_ADMIN }
    #- { path: ^/profile, roles: ROLE_USER }
```

- Un visiteur ou un membre n'aura pas le même menu de navigation, l'administrateur aura un lien afin d'accéder au back-office



Grace à la condition « **{% if app.user and is\_granted('ROLE\_ADMIN') %}** » nous pouvons gérer cet affichage

```
<div class="dropdown-menu rounded border-0 shadow-lg px-2">
  {% if app.user and is_granted('ROLE_ADMIN') %}
  <a class="dropdown-item" href="{{ path('admin') }}"><i class="fa-solid fa-screwdriver-wrench"></i> &nbsp;accès Back-office</a>
  {% endif %}
```



### 5.7.2 La protection contre les injections SQL

Afin de protéger notre application contre les injections SQL, la règle numéro 1 est la validation des « inputs ».

Nous avons mis en place des contraintes de validation afin de filtrer les champs des formulaires.

### 5.7.3 Le Cryptage des mots de passe

Les mots de passe inscrits dans la base de données sont automatiquement « **crypté** » grâce au bundle « **PasswordHasher** » de Symfony, nous avons aussi mis en place une politique de mot de passe solide afin d'éviter que des hackers trouvent les mots de passe les plus évidents.

### 5.7.4 Anti-Spam

Afin de protéger l'application contre les **spams** « les robots spameur » et les **e-mails abusifs**, nous avons mis en place un système de « captcha » en installant le bundle « Gregwar Captcha », qui consiste à générer un code aléatoire que l'utilisateur doit saisir, ce code est sous forme d'image que seul l'être humain peut lire mais pas un robot.

Nous avons rendu ce champ obligatoire en renforçant les contraintes de validation avec « NotBlank ».

### 5.7.5 HTTPS « Protocole de transfert hypertexte sécurisé »

Pendant la mise en ligne de l'application on installera le plug-in « let's Encrypt » qui fournira un certificat SSL, un protocole de cryptage utilisé pour sécuriser la communication de la source (serveur) au destinataire (le navigateur du client).

## 5.8 Les Cookies et le respect de la loi

Le Règlement Général sur la Protection des Données « **RGPD** », nous exige de mettre en évidence et de permettre aux utilisateurs de contrôler l'activation des cookies et des traceurs qui collectent leurs données personnelles.

Afin de respecter le RGPD et la vie privée des utilisateurs nous avons mis en place le script « **Tarte au Citron** » qui va permettre de :

- Recueil des consentements avec désactivation sélectionnées par défaut,
- Bandeau offrant les possibilités de tout accepter, tout refuser ou de personnaliser,
- Paramétrage de ces cookies par le visiteur,
- Durée de conservation limitée du consentement à 13 mois,
- Interdiction du dépôt de cookie avant tout consentement (blocage des scripts par défaut)

Dans notre application y'aura 2 catégories de cookies :

- Cookies obligatoires : strictement nécessaires et indispensables qui servent au bon fonctionnement de l'application (cookie d'authentification par exemple).
- Cookies de mesure et d'analyse : Google Analytics

La liste de ces cookies sera clairement affichée à l'internaute et aura la possibilité de les accepter, les refuser, les personnaliser et de consulter la « politique de confidentialité ».

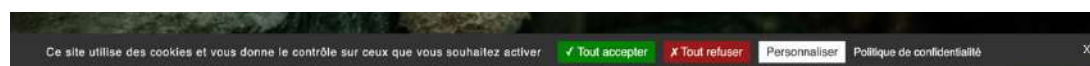
Le scripte « tarte au citron » dans la section « head »

```

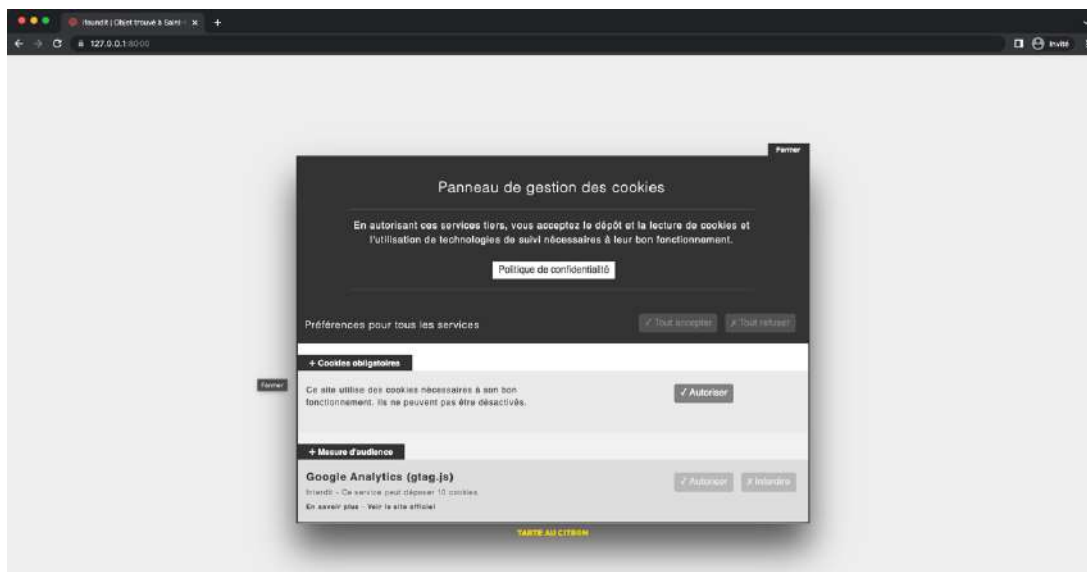
5
6 {# ##### SCRIPT COOKIES TARTEAUCITRON ##### #}
7 <script src="{{ asset('tarteau citron/tarteau citron.js') }}" ></script>
8 <script type="text/javascript">
9     tarteau citron.init({
10         "privacyUrl": "{{ path('politique-de-confidentialite') }}" , /* Privacy policy url */
11
12         "hashtag": "#tarteau citron", /* Open the panel with this hashtag */
13         "cookieName": "tarteau citron", /* Cookie name */
14
15         "orientation": "bottom", /* Banner position (top - middle - bottom) */
16
17         "groupServices": false, /* Group services by category */
18
19         "showAlertSmall": false, /* Show the small banner on bottom right */
20         "cookiesList": true, /* Show the cookie list */
21
22         "closePopup": true, /* Show a close X on the banner */
23
24         "showIcon": false, /* Show cookie icon to manage cookies */
25         // "iconSrc": "", /* Optional: URL or base64 encoded image */
26         "iconPosition": "BottomLeft", /* BottomRight, BottomLeft, TopRight and TopLeft */
27
28         "adblocker": false, /* Show a Warning if an adblocker is detected */
29
30         "DenyAllCta": true, /* Show the deny all button */
31         "AcceptAllCta": true, /* Show the accept all button when highPrivacy on */
32         "highPrivacy": true, /* HIGHLY RECOMMENDED Disable auto consent */
33
34         "handleBrowserDNTRequest": false, /* If Do Not Track == 1, disallow all */
35
36         "removeCredit": false, /* Remove credit link */
37         "moreInfoLink": true, /* Show more info link */
38
39         "useExternalCss": false, /* If false, the tarteau citron.css file will be loaded */
40         "useExternalJs": false, /* If false, the tarteau citron.js file will be loaded */
41
42         // "cookieDomain": ".my-multisite-domaine.fr", /* Shared cookie for multisite */
43
44         "readmoreLink": "", /* Change the default readmore link */
45
46         "mandatory": true, /* Show a message about mandatory cookies */
47     });
48     /* google analytics */
49     tarteau citron.user.gtagId = 'G-XSXG1B41CD';
50     // tarteau citron.user.gtagCrossdomain = ['example.com', 'example2.com'];
51     tarteau citron.user.gtagMore = function () { /* add here your optional gtag() */ };
52     (tarteau citron.job = tarteau citron.job || []).push('gtag');
53 </script>
54 {# ##### FIN SCRIPT COOKIE ##### #}

```

Le bandeau des cookies de « tarte au citron » :



La fenêtre de personnalisation des cookies:



## 5.9 Interface d'administration « Back-Office »

Afin de permettre à l'administrateur du site de gérer le contenu et les utilisateurs nous avons installé le bundle « **EasyAdmin** » qui est proposé par Symfony et qui propose une interface d'administration bien complète et sécurisée, l'accès sera réservé exclusivement à l'administrateur du site.

Nous installons « EasyAdmin » avec la commande suivante :

```
composer require easycorp/easyadmin-bundle
```

Ensuite on est demandé de choisir le nom du Controller, nous gardons le nom donné par défaut « **DashboardController** » qui sera généré dans le dossier Admin « **src/Controller/Admin** », Ce dossier est sécurisé et seul **un utilisateur avec le rôle Admin aura accès**, ce réglage est fait dans le fichier **security.yaml** comme précédemment vu dans la section sécurité.

Maintenant nous créons nos différents CRUD qui vont nous permettre comme le nom l'indique créer, lire, éditer, effacer.

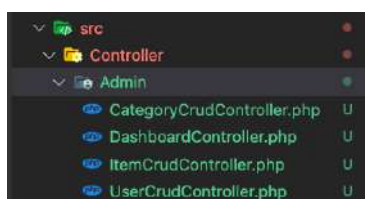
Un CRUD est créer avec la commande suivante :

```
php bin/console make:admin:crud
```

Ensuite on nous demande à quelle entité ce CRUD sera lié, une fois c'est fait notre CRUD est créé.

Pour notre projet nous avons besoin des CRUD suivants : User, Item, Category qui correspondent à nos entités.

Notre « Dashboard » et nos « CRUD » sont maintenant créé :



Ensuite nous configurons les liens du menu et ses icones dans le « DashboardController » :

```
public function configureDashboard(): Dashboard
{
    return Dashboard::new()
        ->setTitle('Administration ifoundit')
        ->setFaviconPath('../img/favicon/favicon.ico');
}

public function configureMenuItems(): iterable
{
    yield MenuItem::linkToDashboard('Home', 'fa fa-home');

    yield MenuItem::section('Utilisateurs');
    yield MenuItem::linkToCrud('Utilisateurs (Users)', 'fas fa-users', User::class);

    yield MenuItem::section('Produits');
    yield MenuItem::linkToCrud('Annonces (Item)', 'fas fa-bullhorn', Item::class);
    yield MenuItem::linkToCrud('Categories', 'fas fa-list', Category::class);

    yield MenuItem::section('Liens');
    yield MenuItem::linkToRoute('Retour Front', 'fa fa-globe', 'home');
    yield MenuItem::linkToLogout('Déconnexion', 'fa fa-sign-out-alt');
}
```

Maintenant notre interface d'administration est prête.

Seul un administrateur aura accès au back-office à partir du menu de la page d'accueil :



## Interface d'administration « Utilisateurs » :

Dans cette rubrique un administrateur pourra gérer les utilisateurs.

Administration ifoundit

Rechercher

Utilisateurs

Utilisateurs (Users)

Produits

Annonces (Item)

Categories

Liens

Retour Front

Déconnexion

|                          | ID | Nom ou Pseudo | Email             | Téléphone  | Créé le       | Roles       | Nbre d'annonces |     |
|--------------------------|----|---------------|-------------------|------------|---------------|-------------|-----------------|-----|
| <input type="checkbox"/> | 1  | admin         | admin@foundit.com | 0000000000 | 10 févr. 2022 | Admin, User | 8               | ... |
| <input type="checkbox"/> | 22 | admin         | admin@foundit.com | 0000000000 | 2 mars 2022   | User        | 1               | ... |
| <input type="checkbox"/> | 27 | admin         | admin@foundit.com | Aucun(e)   | 5 mars 2022   | User        | 0               | ... |
| <input type="checkbox"/> | 28 | admin         | admin@foundit.com | Aucun(e)   | 30 mars 2022  | User        | 0               | ... |
| <input type="checkbox"/> | 29 | admin         | admin@foundit.com | Aucun(e)   | 3 juin 2022   | User        | 0               | ... |

5 résultats

Précédent 1 Suivant

## Interface d'administration « Les annonces » :

Dans cette rubrique un administrateur pourra gérer les Annonces.

| ID | Utilisateur | Catégorie | Active                              | Titre                      | Créé le       | Trouvé le     | Photo | Description                                                                                                                                                | Rue ou endroit      | Ville                | Code postal | Champ indice |
|----|-------------|-----------|-------------------------------------|----------------------------|---------------|---------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|----------------------|-------------|--------------|
| 5  | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | Porte feuille              | 10 févr. 2022 | 1 janv. 2022  |       | J'ai trouvé ce porte feuille dans le bus 51 à Saint Cloud,                                                                                                 | Parc de saint cloud | Saint Cloud          | 92210       | Aucun(e)     |
| 17 | User #22    | Aucun(e)  | <input checked="" type="checkbox"/> | JGUGU                      | 2 mars 2022   | 15 févr. 2022 |       | BUS 467                                                                                                                                                    | RUE LAVAL           | SAINT CLOUD          | 92210       | Aucun(e)     |
| 18 | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | Téléphone                  | 3 mars 2022   | 3 mars 2022   |       | dans la rue                                                                                                                                                | rue de Rivoli       | Paris                | 75001       | bus ligne 71 |
| 19 | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | Téléphone portable samsung | 3 mars 2022   | 16 févr. 2022 |       | dans la rue                                                                                                                                                | rue de Paris        | saint cloud          | 92210       | rien         |
| 23 | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | rttrrt                     | 3 mars 2022   | 3 mars 2022   |       | rttrrt                                                                                                                                                     | rttrrt              | rttrrt               | 12122       | rttrrt       |
| 26 | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | sac à dos en cuir noir     | 10 mars 2022  | 12 janv. 2022 |       | dans le bus ligne 10 à boulogne en direction de paris, dans le bus ligne 10 à boulogne en direction de paris, dans le bus ligne 10 à boulogne en direction | Parc de saint cloud | Boulogne billancourt | 92210       | benyamina    |

## 5.10 La page d'erreur 404

Pour préparer la mise en production de notre application web, nous souhaitons personnaliser la page erreur afficher par défaut sous Symfony, afin d'améliorer l'expérience utilisateur si une page n'existe plus par exemple, le but est d'orienter l'utilisateur et afficher un message personnalisé avec un lien vers l'accueil.

Nous devons surcharger le template existant.

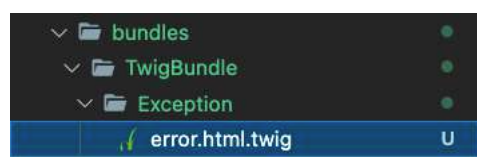
Nous tapons la commande suivante :

```
composer require symfony/twig-pack
```

Ensuite on crée l'arborescence suivante dans le dossier « **templates** » :

« Bundles/TwigBundle/Exception/ »

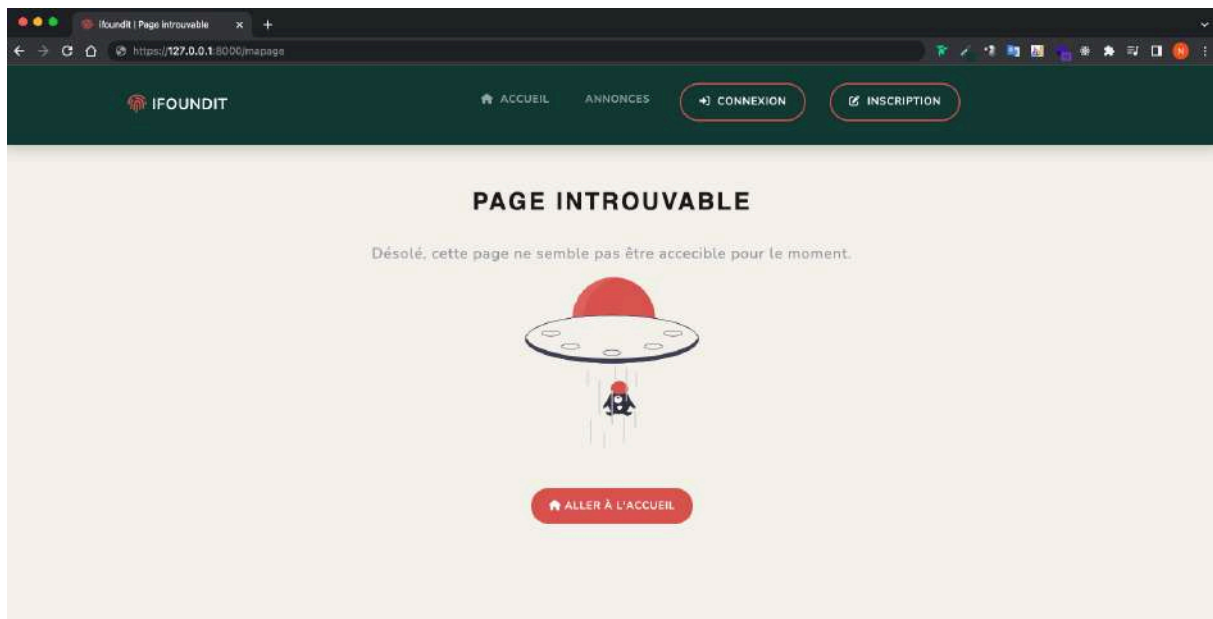
Et dans le dossier « **Exception** » on vient créer la vue qui porte le nom « error.html.twig »



Maintenant nous pouvons personnaliser la page d'erreur.

Afin de tester notre page d'erreur nous mettons l'environnement en PROD dans le fichier .env

Exemple d'une page d'erreur générer volontairement en tapant un nom de page inexistant  
« <https://ifoundit.fr/mapage> »



## 5.11 Intégrer les données dans une vue

### 5.11.1 Intégration des données

Afin d'intégrer les données dans une vue, nous allons voir l'exemple suivant :

Nous souhaitons afficher **les annonces postés par les membres** dans une page dédiée, nous allons intégrer les données dans la vue Twig « listItem.html.twig » ce fichier Twig va contenir toutes les annonces postées.

Nous procédons dans notre Controller comme suit :

```
////////////////////////////////////  
//////////////////////////////////// AFFICHAGE LISTE ITEM ///////////////////////////////////  
  
/**  
 * @Route("/listItem", name="listItem")  
 */  
public function listItem(ItemRepository $repository):Response  
{  
  
    // RÉCUPÉRATION DES ANNONCES  
    $items = $repository->findAll();  
  
    // ENVOI VERS LA VUE  
    return $this->render('home/listItem.html.twig', [  
  
        'items' => $items,  
  
    ]);  
}
```

« La méthode listItem() avant ajout de pagination »



Dans notre méthode **listItem()** on injecte la dépendance « **ItemRepository** » ensuite notre variable **\$items** va stocker toutes les annonces grâce à la méthode **findAll()**, cette variable est envoyée vers la vue Twig grâce à la méthode **render()**.

Dans le Controller on a réalisé quelques modifications :

- Ajout de pagination
- Afficher que les annonces « Active »

### 5.11.2 Pagination

Le problème rencontré avant l'ajout de la pagination c'est que toutes les annonces vont s'afficher dans une seule page.

C'est pour cette raison qu'on a mis en place un système de pagination afin de scinder l'affichage des annonces.

Grace au bundle « KNP Paginator » nous allons résoudre ce problème.

On l'installe avec la commande suivante :

```
composer require knplabs/knp-paginator-bundle
```

La partie « KNP Paginator » dans le Controller :

```
// PAGINATION KNP/PAGINATOR
$items = $paginator->paginate(
    $data,
    $request->query->getInt('page', 1),
    6
);
```

La méthode **paginate()** prend 3 paramètres, le premier c'est la variable **\$data** qui va contenir toutes les annonces, le 2<sup>ème</sup> paramètre c'est le numéro de la page en cours et le 3<sup>ème</sup> paramètre c'est le nombre d'annonces qu'on veut afficher, le tout est stocké dans la variable **\$items**.

#### Côté vue dans Twig

En suivant la documentation KNP Paginator, on nous demande d'insérer ce code afin d'avoir les boutons de navigations dans les pages

```
<div class="d-flex justify-content-center mt-5">
    {{ knp_pagination_render(items, 'base/pagination.html.twig') }}
</div>
```

### 5.11.3 Affichage des annonces active

Nous souhaitons faire apparaître que les annonces qui ont le statut « active » et cacher les autres annonces qui ne l'ont pas.

Nous avons dans la classe « **Item** » (qui correspond aux annonces) une propriété « **isActive** » de type booléen.

```
/**
 * @ORM\Column(type="boolean")
 */
private $isActive;
```

À l'insertion d'une annonce on a défini cette propriété par défaut à « **1** » :

```
$item->setIsActive(1);
```

La méthode **findByIsActive()** qui a été créée dans la classe **ItemRepository**.

```
* @method Item[] findByIsActive(array $criteria, array $orderBy = null, $limit = null, $offset = null)
```

Dans le Controller on récupère les annonces active avec la méthode **findByIsActive()** et en paramètre « **1** » qui va récupérer que les annonces qui ont ce statut.

```
// RÉCUPÉRATION DES ANNONCES ACTIVES
$data = $repository->findByIsActive(
    1,
    ['createdAt' => 'DESC'],);
```

L'administrateur a la possibilité de désactiver une annonce via le back-office

| <input type="checkbox"/> | ID | Utilisateur | Categorie | Active                              | Titre         |
|--------------------------|----|-------------|-----------|-------------------------------------|---------------|
| <input type="checkbox"/> | 5  | User #1     | Aucun(e)  | <input checked="" type="checkbox"/> | Porte feuille |

Coté Controller « **itemCrudController** » du back-office :

```
BooleanField::new('isActive', 'Active'),
```

Voici la méthode **litItem()** avec pagination et l'affichage des annonces active :

```
////////////////////////////////////
//////////////////////////////////// AFFICHAGE LISTE ITEM //////////////////////////////////////
```

```

/**
 * @Route("/listItem", name="listItem")
 */
public function listItem(ItemRepository $repository, PaginatorInterface
$paginator, Request $request): Response
{
    // RÉCUPÉRATION DES ANNONCES ACTIVES
    $data = $repository->findByIsActive(
        1,
        ['createdAt' => 'DESC'],);

    // PAGINATION KNP/PAGINATOR
    $items = $paginator->paginate(
        $data,
        $request->query->getInt('page', 1),
        6
    );
    // RENDER
    return $this->render('home/listItem.html.twig', [
        'items' => $items,
    ]);
}

```

« La version finale de la méthode listItem() »

## Le fichier « listItem.html.twig » :

```
% extends 'base.html.twig' %}

% block title %}{ parent() }} Annonces des objets trouvés{% endblock %}


% block titre %}

Annonces des objets trouvés

% endblock %}

% block body %}



 IFOUNDIT



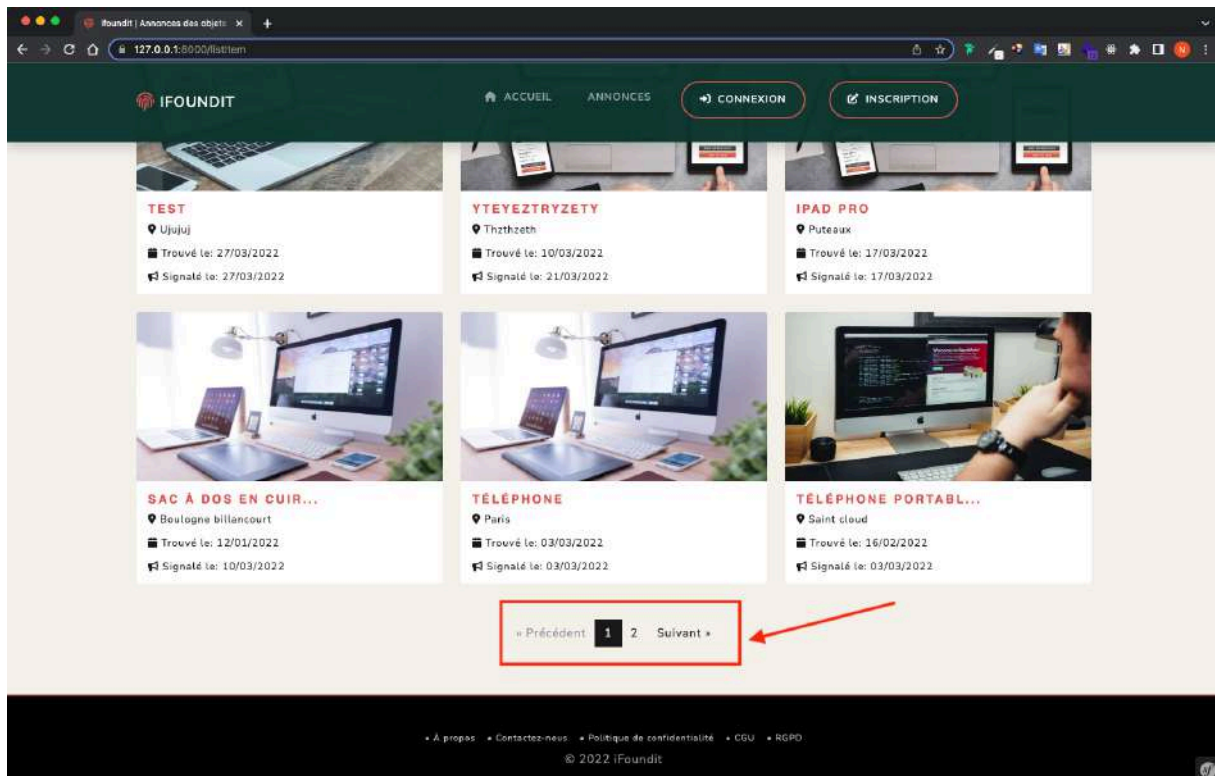
Cahier des charges IFOUNDIT - Noredine BENYAMINA - Mai 2022



59


```

Le rendu de la pagination :



« Intégration de la Pagination »



## 6 LE RÉFÉRENCEMENT



## 6.1 Le Référencement Naturel « SEO »

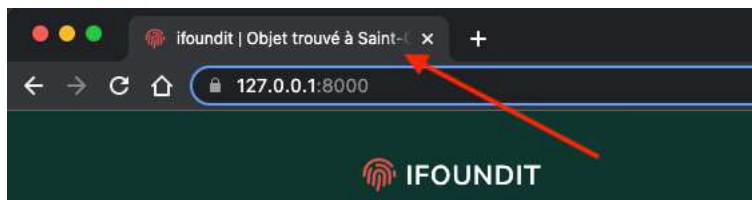
Afin de faire apparaître le site internet « ifoundit » et son contenu dans les résultats des moteurs de recherche, nous devons bien travailler le SEO « Search Engine Optimisation » qui veut dire optimisation pour les moteurs de recherche en Français .

Afin de respecter les exigences du référencement naturel (SEO) des moteurs de recherche (Google pour notre exemple) nous avons procédé comme suit :

### 6.1.1 La balise « title »

La balise <title> dans la section <head>, elle permet de donner un titre à nos pages, ce titre apparaîtra aussi dans les résultats des moteurs de recherche.

```
{% block title %} {{parent()}} Objet trouvé à Saint-Cloud 92210{% endblock %}
```



### 6.1.2 La balise « meta description »

La « meta description » dans la section <head>, c'est le résumé du contenu et c'est le texte qui apparaîtra sous le titre dans les résultats de recherche.

```
<meta name="description" content="ifoundit, annonces des objets perdus à Saint-Cloud 92210, annonces des objets trouvés, retrouvez votre objet perdu">
```

### 6.1.3 Les balises titre Hn

Le « Heading » ces les balises titre de h1 à h6, qui permettent de structurer le contenu. La balise <h1> est la plus importante et ne doit pas être en double dans une page. Dans notre projet chaque page contient une balise h1.

La balise h1 dans le « base.html.twig » notre fichier de base

```
<div class="row mt-3 text-center">
  <h1 class="mt-5"> {% block titre %}{% endblock %}</h1>
</div>
```

Et les « extends », ici la page « home.html.twig » intégrant un bloc titre « h1 » dans la page d'accueil.

```
{% block titre %}
Comment ifoundit vous aide à retrouver votre objet perdu
{% endblock %}
```



#### 6.1.4 L'attribut <alt>

Chaque image à un attribut « alt » bien renseigné de façon dynamique.

```
src="{{ asset('upload/') }}{{ item.picture }}" alt="{{ item.title }}">
```

#### 6.1.5 Responsive - Mobile friendly

Un site internet responsive optimise le référencement naturel (SEO) et les moteurs de recherche recommande de ne pas négliger cette partie et ça permet d'augmenter le trafic de plus de 51%.

Le site internet « ifoundit » est totalement responsive et s'adapte aux différents support digitales.

#### 6.1.6 Le temps de chargement

La rapidité d'affichage des pages d'un site internet fait partie des critères d'optimisation car les moteurs de recherche tiennent également en compte le temps de chargement dans le référencement naturel (SEO).

Pour notre application nous avons obtenu le résultat suivant dans « GTmetrix » :

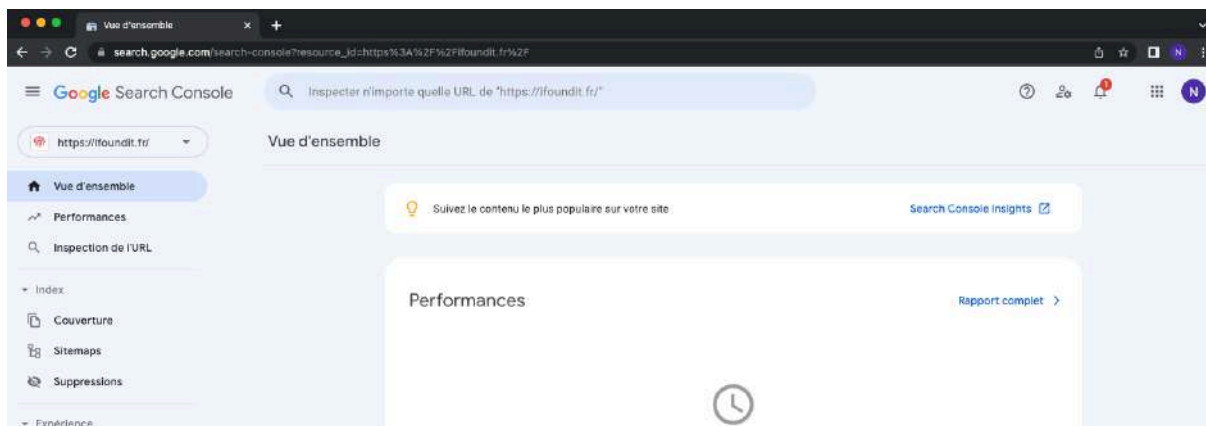


### 6.1.7 La duplication de contenu « duplicate content »

Le contenu de l'application « ifoundit » est un contenu original et non dupliqué, car le moteur de recherche Google considère un contenu dupliqué est une tricherie et Google peut bannir un site internet pour cette pratique.

### 6.1.8 L'indexation

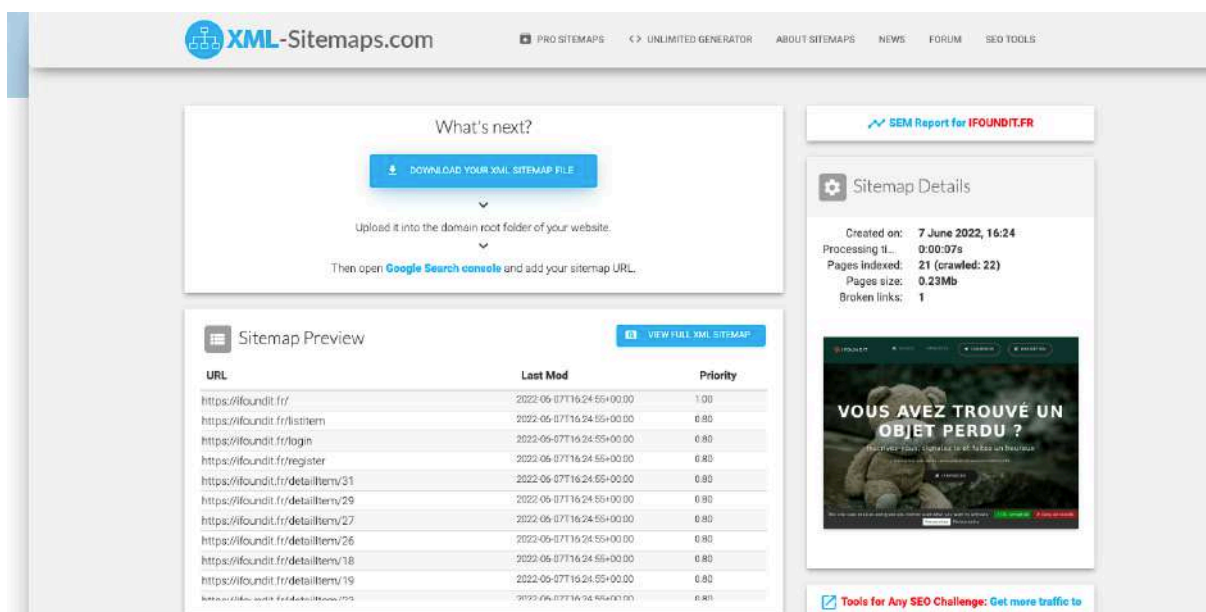
Après la mise en ligne de l'application, nous l'avons indexé sur le moteur de recherche Google et à partir de cette étape l'application est enregistré dans la base de données de Google. Nous avons indexé notre application grâce à l'outil « Search Console » de Google.



### 6.1.9 Sitemap

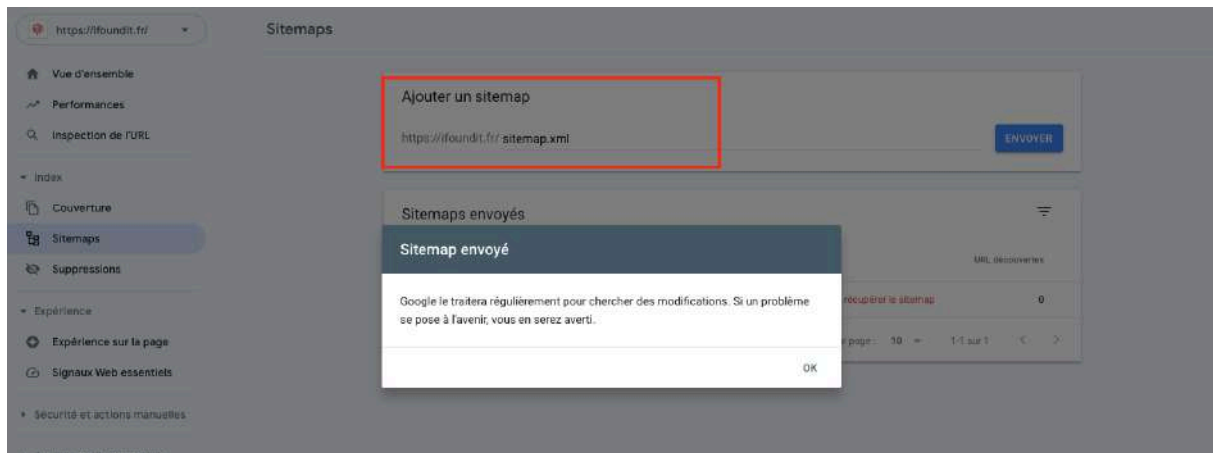
C'est un plan du site appelé « sitemap » en anglais, c'est un fichier qui fournit aux moteurs de recherche des informations détaillées sur toutes les pages de notre application, leurs relations et leurs contenus.

Afin de générer le fichier sitemap, nous avons utilisé le site : <https://www.xml-sitemaps.com>.



Ensuite nous mettons le fichier « sitemap.xml » à la racine du répertoire FTP.

Et enfin nous envoyons le fichier à « Google search Console » en lui indiquant le chemin du fichier « sitemap.xml » :



## 7 DESIGN

## 7.1 Charte graphique

### 7.1.1 Le logo

- Les Propositions de logos :

IFUNDIT

FOUNDIT

 IFOUNDIT

 IFOUNDIT

 IFOUNDIT

 IFOUNDIT

- Logo retenu :



↑  
 **Icône : «Fingerprint»**  
**Couleur : #BB4946**

↑  
**Police : Nunito Sans Bold**  
**Couleur : #000000**

**Favicon :**





### 7.1.2 Les couleurs

4 couleurs sont utilisées dans l'application :



**#BB4946**  
**Framboise**

**Framboise :**

Cette couleur est utilisée dans une partie du logo et dans les titres des annonces, le contour des boutons de la barre de navigation et les boutons de suppression.



**#143C34**  
**Vert impérial**

**Vert impérial :**

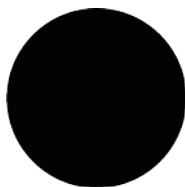
Cette couleur est utilisée pour la barre de navigation.



**#F5F1EC**  
**Lin clair**

**Lin clair :**

Cette couleur est utilisée pour le fond des pages.



**#000000**  
**Noir**

**Noir :**

Cette couleur est utilisée pour les boutons d'appel à l'action (CTA) et le fond du footer.

### 7.1.3 La typographie

Pour ce projet, 2 polices d'écriture de « Google Fonts » ont été choisies :

- **Nunito Sans**
- **Sans-serif**

**Les titres « h1 à h6 »** auront la police d'écriture « Sans-serif »,

**Les paragraphes** auront la police d'écriture « Nunito Sans ».

### 7.1.4 Les icônes

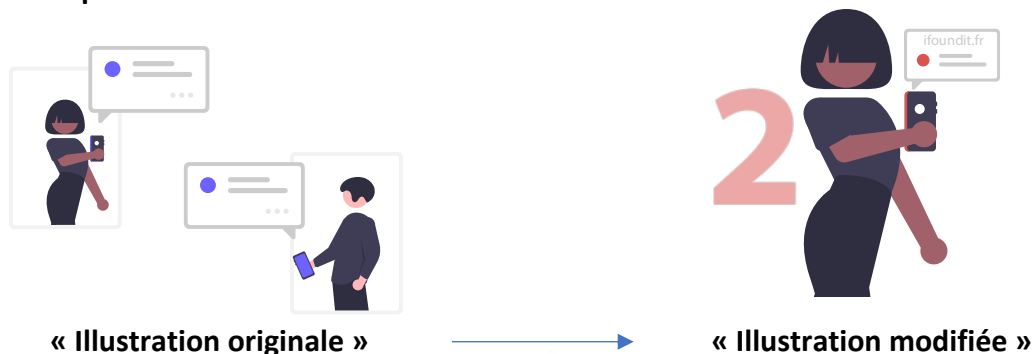
Nous avons choisis pour notre projet les icônes provenant de « <https://fontawesome.com/> ». La présence des icônes dans l'application a un rôle visuel important qui améliore l'expérience utilisateur.

### 7.1.5 Les illustrations

Nous avons utilisé des illustrations vectorielles, ces illustrations ont été personnalisées afin de s'adapter et suivre la charte graphique de « ifoundit ».

Sur « unDraw » nous pouvons récupérer des illustrations libres de droit.

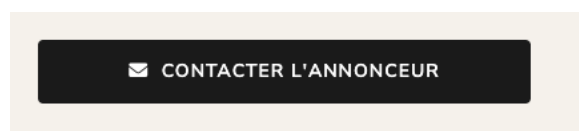
**Exemple de personnalisation d'une illustration :**



### 7.1.6 Les boutons

Les boutons d'appel à l'action sont sur un fond noir, le texte en blanc et une icône exprimant visuellement l'action.

**Exemple d'un bouton :**



## 7.2 Responsive Web Design

Le Responsive Web Design appelé aussi « Design adaptatif » est la possibilité d'adapter la taille d'un site web au support sur lequel l'internaute navigue : ordinateur de bureau, tablette, smartphone, il permet d'optimiser l'ergonomie en ajustant les contenus et les images automatiquement en tenant compte de la résolution de l'écran.

Afin d'avoir une navigation correcte et améliorer l'expérience utilisateur, nous avons appliqué « Le Responsive Web Design » pendant le développement de notre application grâce au « **Framework CSS Bootstrap version 5** » avec son célèbre système de grille qui utilise les propriétés du CSS flexbox.

Nous avons intégré le Thème « Lux » de « Bootswatch » qui fonctionne sous le Framework Bootstrap Version 5 avec un CDN :

Dans le <head> du fichier « base.html.twig » :

```
68
69     {# BootstrapWatch #}
70     <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootswatch/5.1.1/lux/bootstrap.min.css" integrity="sha512-m0ju8pKJzu/
71     lzzsm5WS8MrvFTXE9JMd0mhsz40zN4NNfkERAU4H7qjVQTrWpx5SAJCv6Z2mrGY20ta6W2n+Q==" crossorigin="anonymous" referrerpolicy="no-referrer" />
```

Le JS Bootstrap avant la fermeture de la balise <body>

```
276
277     {##### SCRIPT JS Bootstrap #####}
278     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikTlWxGys0g+0MhuP
+ILRH9sEJ808LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>
```

Et pour la mise en forme des formulaires :

dans le fichier « twig.yaml » dans : app/config/packages comme suit :

```
1 twig:
2     default_path: '%kernel.project_dir%/templates'
3     form_themes: ['bootstrap_5_layout.html.twig']
4
```

## Version Desktop



## Version Tablette



Format paysage



Format portrait

## Version Mobile



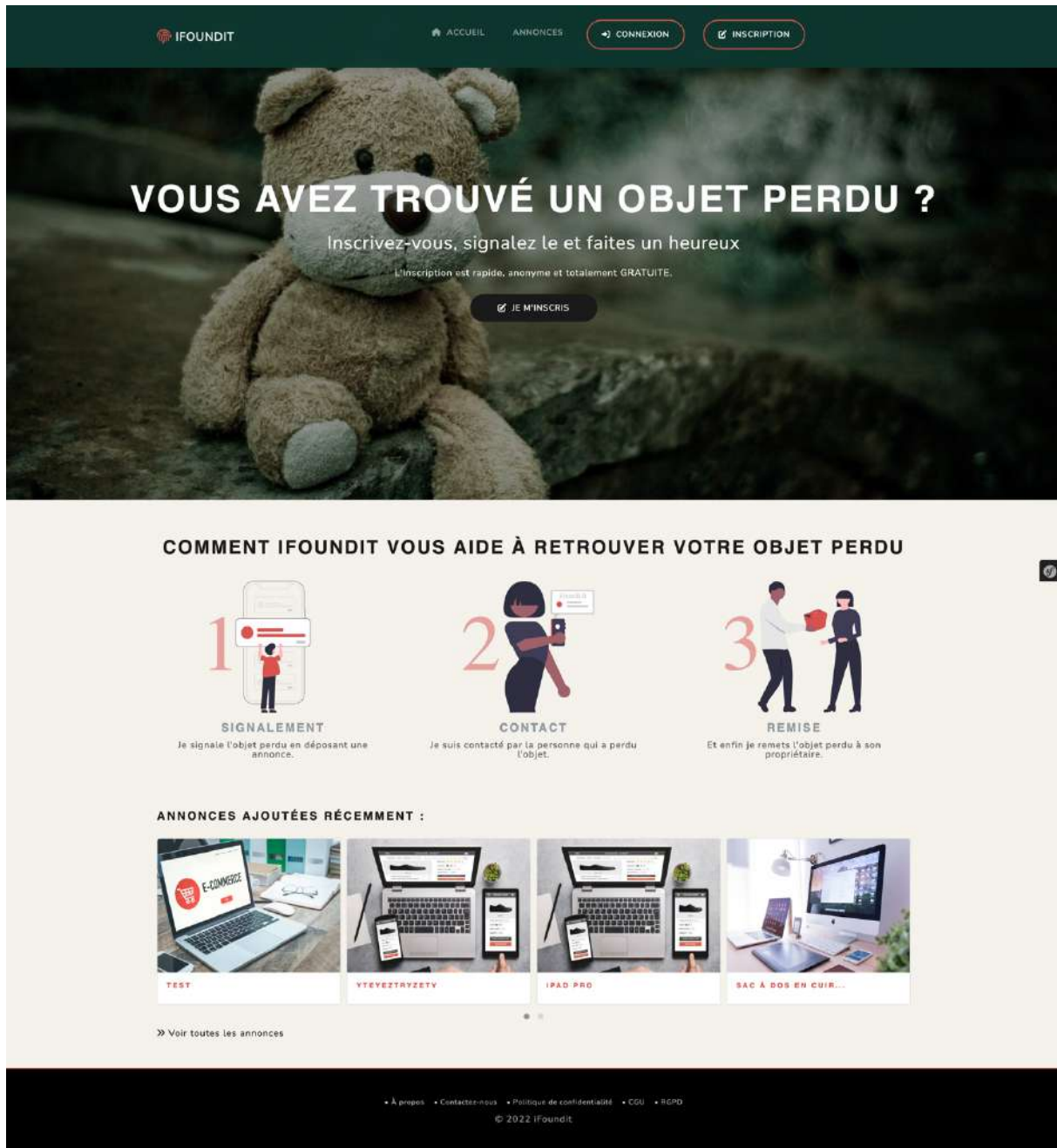
Format paysage




Format portrait

## 7.3 Le rendu final de l'application

### Page d'accueil





## Page des annonces


 [ACCUEIL](#) [ANNONCES](#) [CONNEXION](#) [INSCRIPTION](#)

### ANNONCES DES OBJETS TROUVÉS


Vous pouvez rechercher votre objet perdu par son nom, la ville ou code postale où il a été perdu.

Recherche...  


Il y a **6** annonces




**TEST**  
📍 Ujujuj  
📅 Trouvé le: 27/03/2022  
📅 Signaté le: 27/03/2022




**YTEYEZTRYZETY**  
📍 Thzhzheth  
📅 Trouvé le: 10/03/2022  
📅 Signaté le: 21/03/2022




**IPAD PRO**  
📍 Puteaux  
📅 Trouvé le: 17/03/2022  
📅 Signaté le: 17/03/2022



**SAC À DOS EN CUIR...**  
📍 Boulogne billancourt  
📅 Trouvé le: 12/01/2022  
📅 Signaté le: 10/03/2022



**TÉLÉPHONE**  
📍 Paris  
📅 Trouvé le: 03/03/2022  
📅 Signaté le: 03/03/2022




**TÉLÉPHONE PORTABL...**  
📍 Saint cloud  
📅 Trouvé le: 16/02/2022  
📅 Signaté le: 03/03/2022

« Précédent **1** 2 » Suivant »

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)  
© 2022 IFoundit



## Formulaire de contact

 ACCUEIL ANNONCES **CONNEXION** INSCRIPTION

### FORMULAIRE DE CONTACT


Nom


Email


Spécifiez l'objet de votre message

Choisissez une option

Message


 [Renouveler](#)

 ENVOYER LE MESSAGE



[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)  
© 2022 iFoundit

## Formulaire d'inscription

 ACCUEIL ANNONCES **CONNEXION** **INSCRIPTION**

### INSCRIPTION

L'inscription sur iFoundit est rapide, sécurisée, anonyme et totalement gratuite.

Nom ou pseudo


E-mail


Téléphone (Facultatif)

Mot de passe

Confirmation

J'ai lu et j'accepte les [CGU](#) et la [politique de confidentialité](#)

 [Renouveler](#)

 S'INSCRIRE

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)  
© 2022 iFoundit

## Connexion

The screenshot shows the login page of the IFOUNDIT website. The header is dark green with the IFOUNDIT logo on the left and navigation links: ACCUEIL, ANNONCES, **CONNEXION** (highlighted with a red border), and INSCRIPTION. The main content area has a light beige background. On the left, there is a login form with two input fields: 'Saisissez votre e-mail' (with an @ icon) and 'Saisissez votre mot de passe' (with a lock icon). Below these is a black button with the text '→ CONNEXION'. Under the button, there are two links: 'Mot de passe oublié ?' and 'Vous n'avez pas de compte? Inscrivez-vous'. To the right of the form is an illustration of a woman in a red top and black pants standing next to a large laptop. The laptop screen shows a login form, and a red lock icon is overlaid on the top right of the screen. The footer is dark green and contains links: 'À propos', 'Contactez-nous', 'Politique de confidentialité', 'CGU', and 'RGPD', followed by the copyright notice '© 2022 IFoundit'.

IFOUNDIT

ACCUEIL ANNONCES **CONNEXION** INSCRIPTION

### CONNEXION

Saisissez votre e-mail

Saisissez votre mot de passe

→ CONNEXION

Mot de passe oublié ?  
Vous n'avez pas de compte? Inscrivez-vous

À propos • Contactez-nous • Politique de confidentialité • CGU • RGPD  
© 2022 IFoundit

## Page login

The screenshot shows the user dashboard of the IFOUNDIT website. The header is dark green with the IFOUNDIT logo on the left and navigation links: ACCUEIL, ANNONCES, SIGNALER UN OBJET, and **MON COMPTE** (highlighted with a red border). The main content area has a light beige background. At the top, it says 'BONJOUR! NORDINE'. Below this is the question 'Que souhaitez-vous faire maintenant ?'. There are two red buttons: '→ DÉPOSER UNE ANNONCE' and '→ CONSULTER LES ANNONCES'. Below the buttons is a link: 'Aller à l'accueil | Consulter mon compte'. The footer is dark green and contains links: 'À propos', 'Contactez-nous', 'Politique de confidentialité', 'CGU', and 'RGPD', followed by the copyright notice '© 2022 IFoundit'.

IFOUNDIT

ACCUEIL ANNONCES SIGNALER UN OBJET **MON COMPTE**

### BONJOUR! NORDINE


Que souhaitez-vous faire maintenant ?

→ DÉPOSER UNE ANNONCE → CONSULTER LES ANNONCES

Aller à l'accueil | Consulter mon compte

À propos • Contactez-nous • Politique de confidentialité • CGU • RGPD  
© 2022 IFoundit

## Formulaire de dépôt d'annonce

 [ACCUEIL](#) [ANNONCES](#) [SIGNALER UN OBJET](#) [MON COMPTE](#)

### SIGNALER UN OBJET TROUVÉ !

Vous avez trouvé un objet ? Vous pouvez le signaler dans ce formulaire; le heureux propriétaire de l'objet vous contactera.

**ATTENTION :** Les passeports et cartes d'identité ne doivent pas être signalés chez iFoundit.

Qu'avez-vous trouvé ?

Image du produit trouvé

L'ajout de photo augmente de 80% la chances de retrouver l'objet perdu.

Comment j'ai trouvé cet objet ?

Exemple : Dans le bus ligne 72, en direction de Rivoli vers 07h30, j'ai trouvé un sac noir...

Indices et informations supplémentaires (Facultatif)

Exemple : Un nom, un prénom trouvé dans l'objet; cet indice ne sera pas visible dans l'annonce mais il permet de faire apparaître l'annonce dans les résultats de recherche.

Détail de l'endroit

Un nom de rue, gare, aéroport...

Ville

Le nom de la ville ou l'objet à été trouvé.

Code postal

Le code postal ou l'objet à été trouvé.

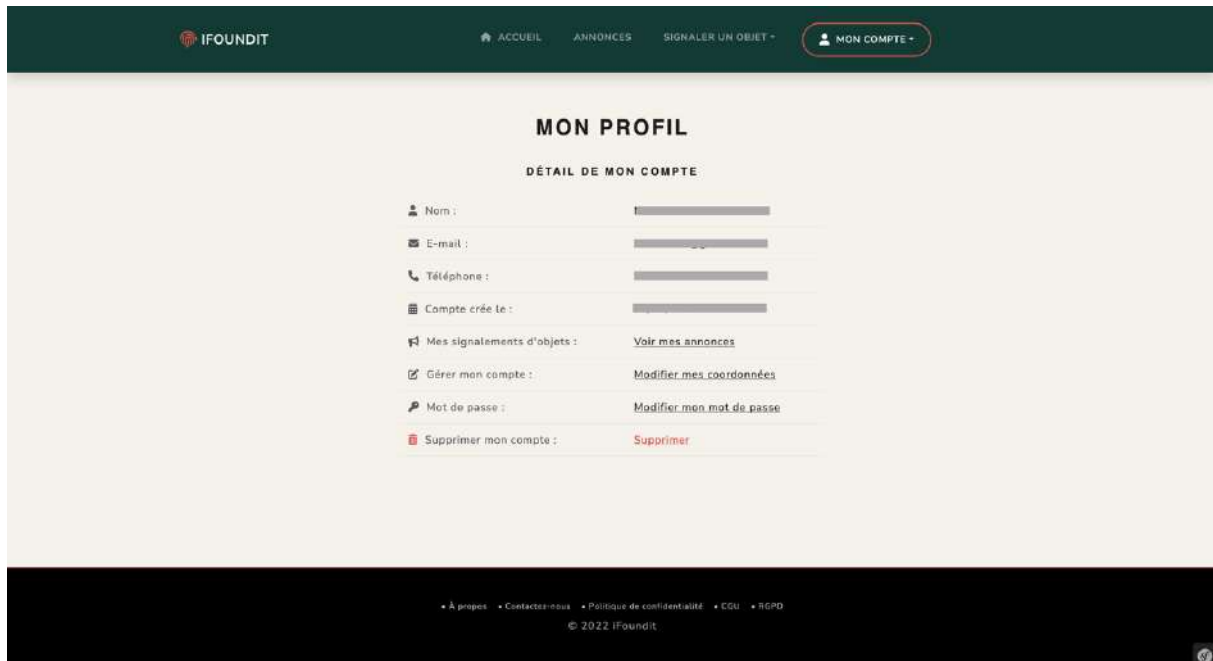
J'ai trouvé cet objet le

J'ai lu et j'accepte les CGU et la politique de confidentialité

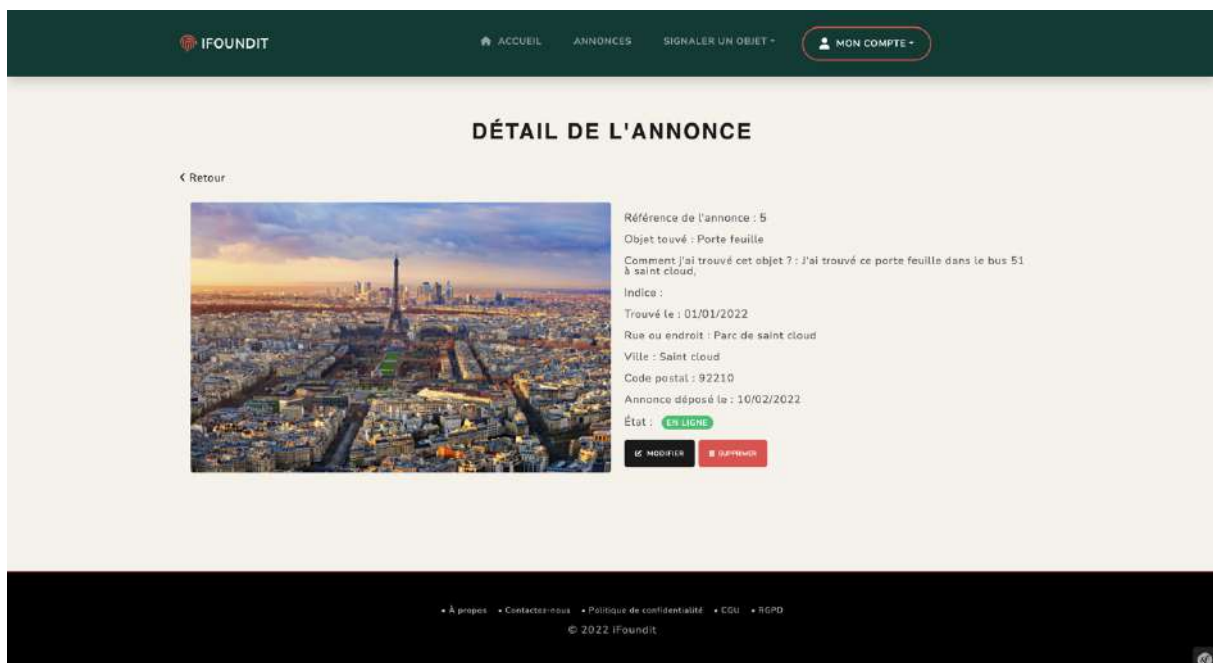
☒ PUBLIER L'ANNONCE

[À propos](#) [Contactez-nous](#) [Politique de confidentialité](#) [CGU](#) [RGPD](#)  
© 2022 iFoundit

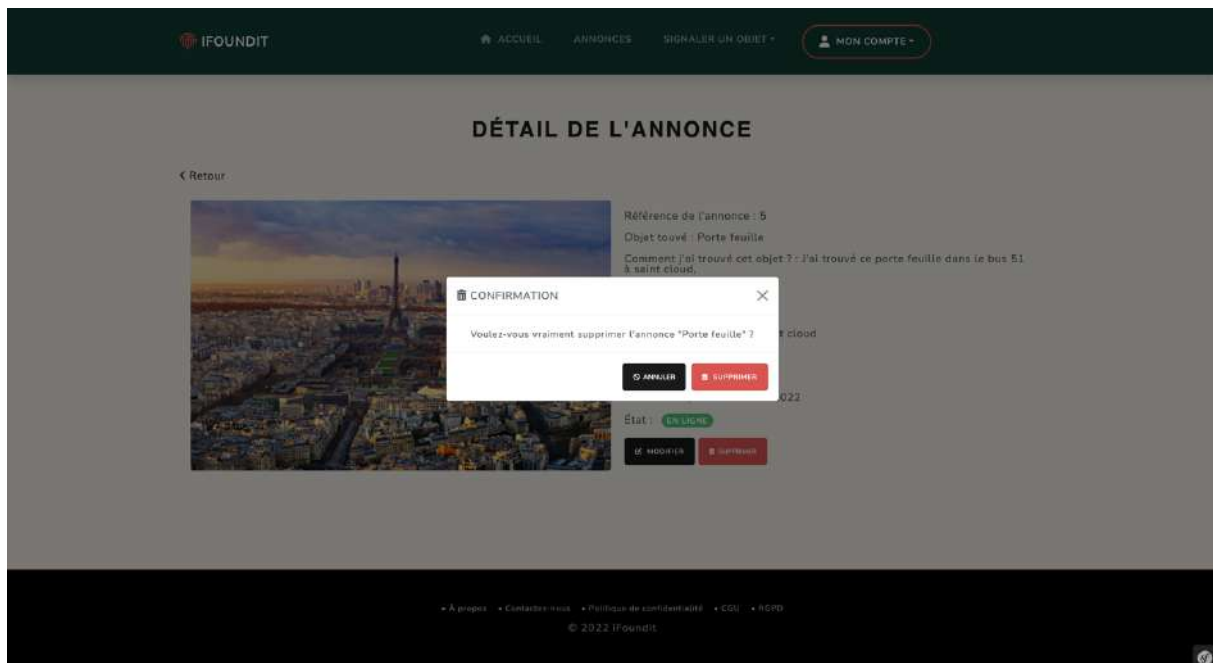
## Page profil



## Détail annonce



## Alert de suppression avec confirmation



## Formulaire de modification d'annonce

The screenshot shows the 'MODIFICATION DE L'ANNONCE' form on the IFOUNDIT website. The form contains the following fields and elements:

- Qu'avez-vous trouvé ?**: A text input field containing 'Porte feuille'.
- Image du produit trouvé**: A section with a 'Parcourir...' button, the text 'Aucun fichier sélectionné.', and a preview of the 'Image actuelle' showing a cityscape with the Eiffel Tower.
- Comment j'ai trouvé cet objet ?**: A text input field containing 'J'ai trouvé ce porte feuille dans le bus 51 à Saint Cloud,'.
- Indices et informations supplémentaires (Facultatif)**: A text input field with the placeholder 'Exemple : Un nom, un prénom trouvé dans l'objet'.
- Détail de l'endroit : nom de rue, gare, aéroport...**: A text input field containing 'Parc de saint cloud'.
- Ville**: A text input field containing 'Saint Cloud'.
- Code postal**: A text input field containing '92210'.
- Date**: A date picker showing '01/01/2022'.
- VALIDER LES MODIFICATIONS**: A black button with a checkmark icon.

The footer of the page includes the same navigation links and copyright notice as the first screenshot.

## Interface d'administration « Back-office »

Administration ifoundit

Home

UTILISATEURS

Utilisateurs (Users)

PRODUITS

Announces (Item)

Categories

LIENS

Retour Front

Déconnexion

Q Rechercher

Item

| ID | Utilisateur | Categorie | Active                              | Titre                      | Crée le       | Trouvé le     | Photo | Description                                                                                                                                                | Rue ou endroit      | Ville                | Code postal | Champ indice |     |
|----|-------------|-----------|-------------------------------------|----------------------------|---------------|---------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|----------------------|-------------|--------------|-----|
| 5  | User #1     | Aucune    | <input checked="" type="checkbox"/> | Porte fouille              | 10 févr. 2022 | 1 janv. 2022  |       | J'ai trouvé ce porte fouille dans le bus 51 à Saint Cloud.                                                                                                 | Parc de saint cloud | Saint Cloud          | 92210       | Aucune       | ... |
| 17 | User #22    | Aucune    | <input checked="" type="checkbox"/> | JGJGJ                      | 2 mars 2022   | 15 févr. 2022 |       | BUS 467                                                                                                                                                    | RUE LAVAL           | SAINT CLOUD          | 92210       | Aucune       | ... |
| 18 | User #1     | Aucune    | <input checked="" type="checkbox"/> | Téléphone                  | 3 mars 2022   | 3 mars 2022   |       | dans la rue                                                                                                                                                | rue de Rivoli       | Paris                | 75001       | bus ligne 71 | ... |
| 19 | User #1     | Aucune    | <input checked="" type="checkbox"/> | Téléphone portable samsung | 3 mars 2022   | 16 févr. 2022 |       | dans la rue                                                                                                                                                | rue de Paris        | saint cloud          | 92210       | rien         | ... |
| 23 | User #1     | Aucune    | <input checked="" type="checkbox"/> | rttrt                      | 3 mars 2022   | 3 mars 2022   |       | rttrt                                                                                                                                                      | rttrt               | rttrt                | 12122       | rttrt        | ... |
| 26 | User #1     | Aucune    | <input checked="" type="checkbox"/> | sac à dos en cuir noir     | 10 mars 2022  | 12 janv. 2022 |       | dans le bus ligne 10 à boulogne en direction de paris, dans le bus ligne 10 à boulogne en direction de paris, dans le bus ligne 10 à boulogne en direction | Parc de saint cloud | Boulogne billancourt | 92210       | benyamina    | ... |
| 27 | User #1     | Aucune    | <input checked="" type="checkbox"/> | ipad pro                   | 17 mars 2022  | 17 mars 2022  |       | dans le tramway ligne 2                                                                                                                                    | gare La Défense     | Puteaux              | 92200       | nordine      | ... |
| 28 | User #1     | Aucune    | <input checked="" type="checkbox"/> | ytayezbzyzety              | 21 mars 2022  | 10 mars 2022  |       | ghshghz                                                                                                                                                    | zethzethzeth        | ththzeth             | 12          | hzhzethaeh   | ... |
| 29 | User #1     | Aucune    | <input checked="" type="checkbox"/> | test                       | 27 mars 2022  | 27 mars 2022  |       | jnyijji                                                                                                                                                    | ijijji              | ijijji               | 32322       | jijji        | ... |

9 résultats

Précédent

1

Suivant





## 8 LES MENTIONS OBLIGATOIRES

## **8.1 Les obligations à respecter**

Vis-à-vis de la loi « pour la confiance dans l'économie numérique », nous devons obligatoirement faire figurer des mentions obligatoires suivantes :

Afin de respecter ces obligations nous avons pensé aux éléments suivants :

### **8.1.1 Conditions générales d'utilisation (CGU) »**

Une page : qui doit afficher clairement les informations suivantes :

- Identité de l'entreprise ou de la personne physique
- Les coordonnées
- Identité de l'hébergeur du site

### **8.1.2 Politique de Confidentialité**

une page afin de décrire clairement la façon dont les informations personnelles sont recueillies, utilisées et partagées

### **8.1.3 Règlement Général sur la Protection des Données (RGPD) »**

Une page afin d'éclaircir les droits de l'utilisateur sur ses données personnelles.

### **8.1.4 Cookies**

Informers les internautes de la finalité des cookies et obtenir leur consentement.

## 9 LA MISE EN LIGNE

## 9.1 La mise en ligne de l'application

### 9.1.1 Le choix de l'hébergeur

Après comparaison des hébergeurs nous avons choisis « o2switch » pour les raisons suivantes :

- Datacenter en France.
- Nom de domaine offert.
- Espace disque illimité.
- Prix compétitif.
- Service Technique 24/7.

o2switch propose une seule offre c'est « L'offre unique » à 5,00 € HT/Mois.

### 9.1.2 Paramétrage « cPanel »

Après activation du compte chez l'hébergeur, nous préparons notre hébergement à recevoir l'application mais avant cela nous devons faire quelques réglages dans l'interface «cPanel » :

- Configurer le nom de domaine.
- Installer un certificat SSL
- Créer un compte FTP
- Créer une base de données MySQL et importer la BDD créée en local.
- Créer un compte e-mail.
- Sélectionner la version PHP qui correspond à la version PHP de notre projet (PHP 8 pour notre projet).

### 9.1.3 Test et Préparation des fichiers en local

Nous effectuons les derniers tests de toutes les fonctionnalités en local avant la mise en ligne et nous vidons le cache.

Nous préparons le fichier « .env » pour passer à l'environnement production, paramétrer la connexion de la base de données et l'envoi des e-mails.

```
16 ##> symfony/framework-bundle ##
17 APP_ENV=prod
18
19 ##< symfony/framework-bundle ##
20
21 ##> symfony/mailer ##
22 MAILER_DSN=smtplib://contact@ifoundit.fr:mail.ifoundit.fr:465
23 ##< symfony/mailer ##
24
25 ##> doctrine/doctrine-bundle ##
26 # Format described at https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/configuration.html#connecting-using-a-url
27 # IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.yaml
28 #
29 # DATABASE_URL="sqlite://%kernel.project_dir%/var/data.db"
30 # DATABASE_URL="mysql://root:@127.0.0.1:3306/skel?serverVersion=MySQL-10.4.21"
31 DATABASE_URL="mysql://scilspj8082@ifoundit:localhost:3306/scilspj8082_ifoundit"
32 #DATABASE_URL="postgresql://symfony:ChangeMe@127.0.0.1:5432/app?serverVersion=13&charset=utf8"
33 ##< doctrine/doctrine-bundle ##
34
```

Ensuite nous avons besoins de générer le fichier « .htaccess » non présent dans les versions récente de Symfony, grace à l'installation d'un package comme suit :

```
composer require symfony/apache-pack
```

Ce package va nous générer le fichier « .htaccess » avec les règles de réécritures des url.

#### 9.1.4 Le transfert

À l'aide d'un logiciel de transfert FTP, nous transférons la totalité de notre projet avec les dépendances (le dossier « vendor ») vers notre répertoire FTP distant « /public\_html » et nous le pointons vers le dossier « **public** » de notre projet Symfony :



**Après avoir effectué tous les tests nécessaires,  
Notre application est maintenant en ligne et fonctionnelle.**

**www.ifoundit.fr**





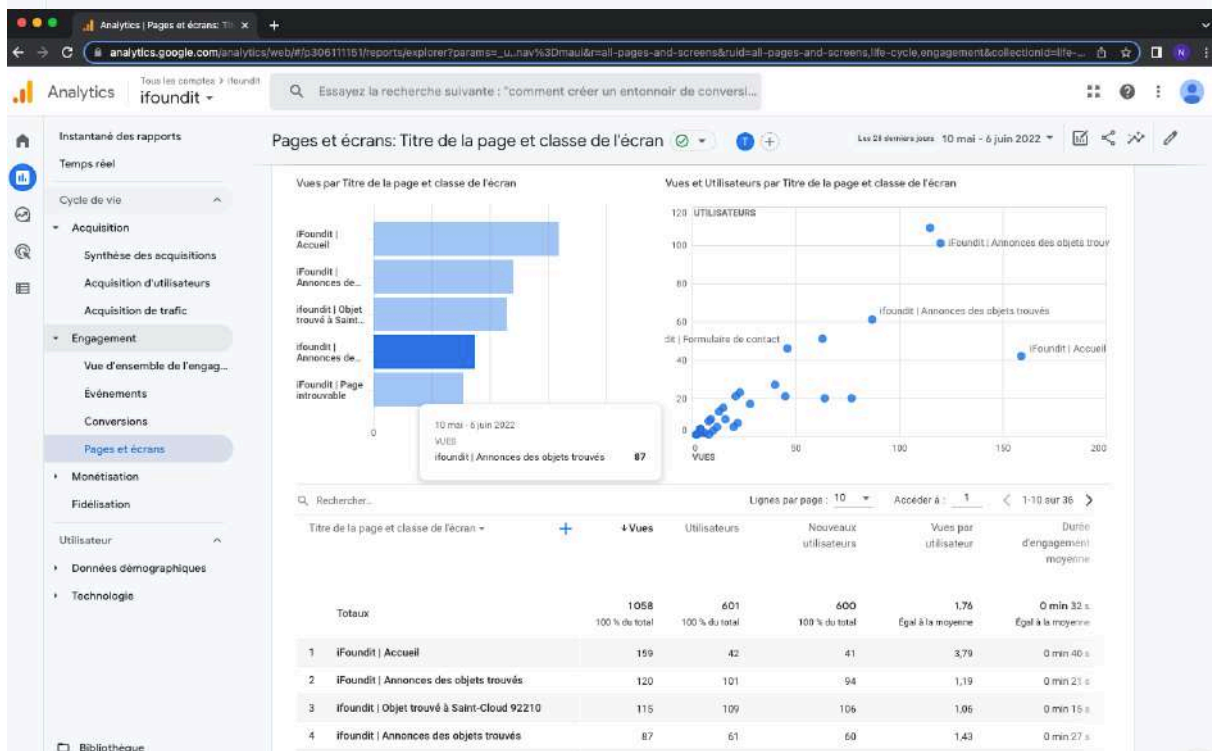
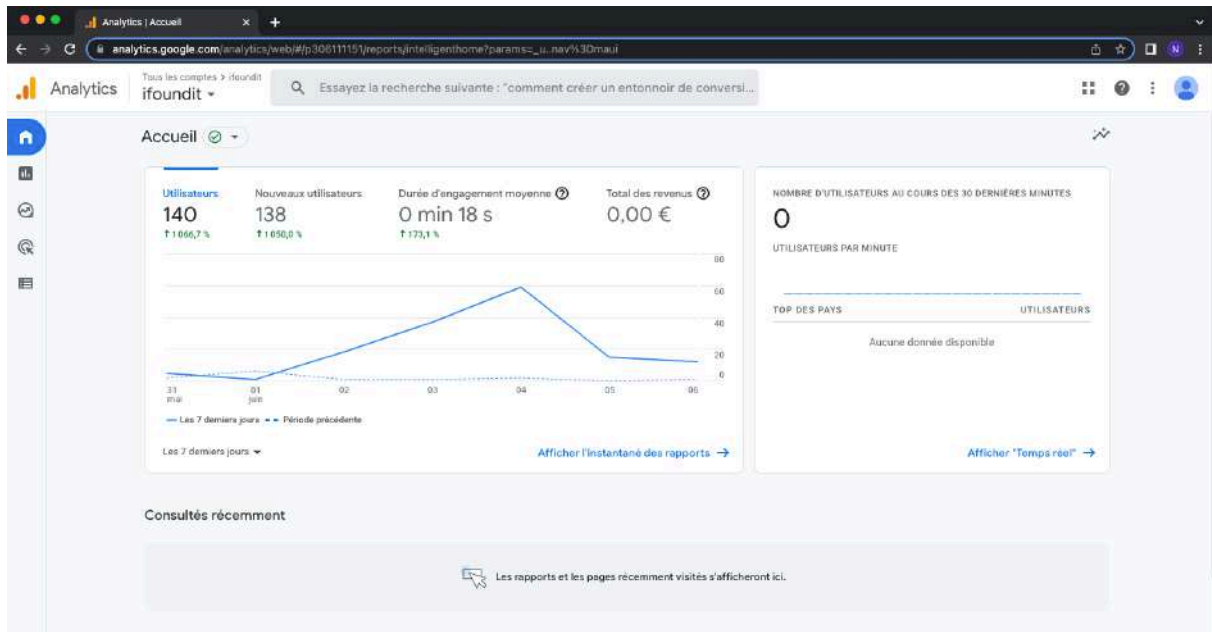
# 10 OUTILS D'ANALYSE



## 10.1 Google Analytics

Nous avons mis en place un outil d'analyse « Google Analytics », cet outil va permettre l'analyse des points suivants :

- Nombre de visite
- Données en temps réel (lieu de connexion, les contenus les plus consultés...).
- Données démographiques afin de mieux cibler la clientèle
- La liste des mots clés
- Technologie et mobile pour adapter notre site web



# 11 CONCLUSION

Nous avons conçu une application web qui répond aux attentes du cahier des charges et fonctionnelle.

Nous projetons de faire évoluer l'application « ifoundit.fr » au fur et à mesure.

À court terme nous ajouterons la catégorie « objet perdu », un système de vérification par e-mail à l'inscription et une mise en place de l'auto-complétion d'adresse.

À long terme, nous envisageons la mise en place d'un système de messagerie entre membres, la possibilité d'insérer plusieurs photos, un système de donation, et un espaces publicitaires payant.