

Lucas Moniot

IUT INFORMATIQUE

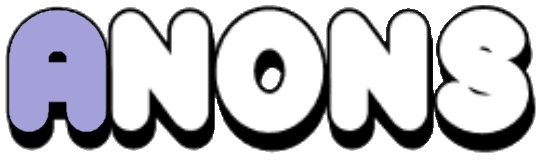


Table des matières

[I. Présentation du projet 3](#_Toc99564331)

[1. Principe du site 3](#_Toc99564332)

[2. Objectif du projet 3](#_Toc99564333)

[a) Pourquoi cette technologie ? 3](#_Toc99564334)

[b) Les objectifs 3](#_Toc99564335)

[3. Déroulement du projet 3](#_Toc99564336)

[a) 1ère étape : déterminer une charte graphique 3](#_Toc99564337)

[b) 2ème étape : définir les fonctionnalités 4](#_Toc99564338)

[c) 3ème étape : organiser les pages 4](#_Toc99564339)

[d) 4ème étape : faire les croquis 4](#_Toc99564340)

[e) 5ème étape : déterminer une base de données 4](#_Toc99564341)

[f) 6ème étape : réaliser le site 4](#_Toc99564342)

[4. Technologies utilisées 4](#_Toc99564343)

[a) Framework : Symfony 4](#_Toc99564344)

[b) Moteur de template : Twig 5](#_Toc99564345)

[c) Gestionnaire de dépendances : Composer 5](#_Toc99564346)

[d) EDI : Visual Studio Code 5](#_Toc99564347)

[e) Doctrine 5](#_Toc99564348)

[II. Aspects graphiques 7](#_Toc99564349)

[1. Chartres graphiques 7](#_Toc99564350)

[a) Palette de couleur 7](#_Toc99564351)

[b) Typographie 7](#_Toc99564352)

[c) L’utilisation de pictogramme 7](#_Toc99564353)

[2. Les boutons 8](#_Toc99564354)

[a) Bouton simple 8](#_Toc99564355)

[b) Le bouton amélioré 8](#_Toc99564356)

[3. La barre de navigation 8](#_Toc99564357)

[a) Utilisateur anonyme 9](#_Toc99564358)

[b) Utilisateur connecté 9](#_Toc99564359)

[4. Les différents pages 10](#_Toc99564360)

[a) Page d’accueil 10](#_Toc99564361)

[5. Ergonomie 10](#_Toc99564362)

[III. Base de données 10](#_Toc99564363)

[1. Modèle conceptuel des données (MCD) 10](#_Toc99564364)

[2. Modèle logique des données (MLD) 11](#_Toc99564365)

[3. Détails des choix 11](#_Toc99564366)

[a) Tables 11](#_Toc99564367)

[b) Relations 12](#_Toc99564368)

[IV. Structure du code 12](#_Toc99564369)

[1. Modèle Vue Controller 12](#_Toc99564370)

[2. Les templates 12](#_Toc99564371)

[V. Fonctionnalités codées 13](#_Toc99564372)

[VI. Conclusion : 14](#_Toc99564373)

[1. Difficultés rencontrées 14](#_Toc99564374)

[2. Les points positifs / ce que j’ai aimé 14](#_Toc99564375)

[3. Une suite pour ce projet ? 14](#_Toc99564376)

[4. A venir 14](#_Toc99564377)

# Présentation du projet

L’intégralité du projet est disponible sur Github : <https://github.com/Arkcoz/Anons>

## Principe du site

Anons est une plateforme de consommation collaborative qui met en relation essentiellement des particuliers en France souhaitant vendre ou acheter des biens d’occasions. Le principe est de permettre aux particuliers de poster leurs annonces en renseignant différentes informations (titre, description, prix, localisation, …), cette dernière sera visible par l’ensemble des visiteurs de la plateforme. Si une annonce intéresse le visiteur, il pourra contacter le propriétaire de l’annonce et s’arranger avec lui.

## Objectif du projet

### Pourquoi cette technologie ?

Suite aux différents confinements liés à la pandémie de Covid-19, nous avons pu observer une activité en constante augmentation que ce soit sur les sites de e-commerces (Amazon, Cdiscount, Fnac, …) mais aussi des sites d’annonces tels que **LeBonCoin** en France. Ces deux types de sites sont conçu la plupart du temps grâce aux Framework **Symfony** ou **Laravel**. Or, lors des deux années d’enseignement à l’IUT Informatique, nous avons très peu eut l’occasion de manipuler des **framework**, et encore moins en PHP. C’est pourquoi, j’ai décidé de créer un site d’annonce à l’instar de LeBonCoin grâce au framework **Symfony**, qui propose de mettre en place une structure MVC facilement.

### Les objectifs

Symfony proposant énormément de composant PHP, l’objectif était donc de se familiariser avec sa structure, son fonctionnement et ses différentes fonctionnalités mises à disposition tout en participant à la réalisation d’un site web.

## Déroulement du projet

Évidemment, je n’ai pas commencé à coder directement sans aucune représentation des différents éléments de mon site, j’ai suivi plusieurs étapes lors de la réalisation de mon projet.

### 1ère étape : déterminer une charte graphique

Dans un site web, la chartre graphique est importante puisqu’elle représente l'ensemble des éléments qui définissent l'identité visuelle du site. J’ai donc commencé par chercher la charte graphique adéquat au site que je voulais réaliser.

### 2ème étape : définir les fonctionnalités

Que ce soit du côté administrateur ou utilisateur, énormément de fonctionnalité doivent être proposé pour rendre un site fonctionnel, j’ai regroupé toute mes idées et je les ai organisés.

### 3ème étape : organiser les pages

Les pages se doivent d’être organisées, l’objectif est de mettre les fonctionnalités sur la page la plus pertinente. Par exemple, la fonctionnalité de recherche d’annonce sur la page de « recherche », ou le bouton « modifier profil » sur la page « profil ».

### 4ème étape : faire les croquis

C’est le moment de disposer les fonctionnalités sur les pages tout en respectant la charte graphique décidé en amont. Pour ma part, je les ai réalisés sur la plateforme **Miro**.

### 5ème étape : déterminer une base de données

Puisque mon projet va proposer au visiteur de créer leur compte et créer leur annonce, je vais devoir stocker toute ces informations dans une base de données.

### 6ème étape : réaliser le site

Une fois toutes les étapes préliminaires réalisées, on peut enfin passer à la réalisation du site.

## Technologies utilisées

### Framework : Symfony

Symfony est un framework qui représente un ensemble de composants (aussi appelés librairies) **PHP** autonomes qui peuvent être utilisés dans des projets web privé ou **open source**. Mais c’est également un **puissant Framework PHP** développé par une société française : [***SensioLabs***](https://sensiolabs.com/). Il permet de réaliser des sites internet dynamiques de manière rapide, structurée, et avec un développement clair. Les développeurs peuvent travailler sur ce Framework très facilement, seuls ou en équipe, grâce à la facilité de prise en main.

### Moteur de template : Twig

Twig est le moteur de templates pour le langage de programmation PHP utilisé par défaut par le framework Symfony. Il permet de créer facilement les templates d’un site en proposant un langage moderne et concis.

### Gestionnaire de dépendances : Composer

Pour la gestion des dépendances entre applications et librairies, j’ai choisi d’utiliser Composer. Très simple d’utilisation, il m’a permis notamment d’installer les composants de Symfony.

### EDI : Visual Studio Code

L’EDI Visual Studio Code a été choisi il est très simple d’utilisation, très ergonomique et je suis très familier avec ce dernier.

### Doctrine

Doctrine est l’ORM (Object Relational Mapping) de Symfony. Il est l’intermédiaire entre notre application et la bases de données. Doctrine supporte tous les langages : MySQl, PostGreSQL… C’est une couche intermédiaire qui nous permet de nous affranchir d’utiliser les langages de gestion de bases de données. Doctrine permet d’assurer aussi une sécurité contre les failles qui atteignent le plus souvent l’accès aux bases. Il est donc conseillé de l’utiliser. Sous Symfony, Doctrine est installé par défaut. Il n’est pas besoin de charger des packages supplémentaires.

#### Tokens

Doctrine offre beaucoup de possibilité, pour la gestion des droits des visiteurs j’ai utilisé les « tokens » qui permettent de déterminer si l’utilisateur actuel possède les droits pour accéder à une page. Les principaux rôles créer grâces à ses tokens sont l’ « admin » et l’ « utilisateur ».

#### Migration

La gestion de la base de données est faite directement dans le projet grâce aux outils proposés par Doctrine. Après chaque modification de la base de données, vous pouvez faire une migration des changements. Cela va créer une nouvelle version de migration qui va être stocker dans le dossier adéquat.

Cet outil est vraiment très intéressant, car pour une personne ou un poste récupérant le projet et n’ayant pas la base de données à jour, il n’aura plus qu’à exécuter une ligne de commande dans un terminal pour appliquer les différentes versions de la base de données stocker dans le projet.

#### Fixture

Doctrine offre aussi la possibilité d’utiliser des fixtures. Une fixture est un jeu de données généré en fonction des entités que l’on possède dans notre base de données qui permet d’avoir un environnement de développement proche d’un environnement de production avec des fausses données.

# Aspects graphiques

## Chartres graphiques

### Palette de couleur



La palette de couleur a été créer en fonction des couleurs du logo.

### Typographie

La typographie initialement choisie sur les maquettes était « **Spoof** », cependant difficile d’accès, je me suis orienté sur une typographie dans la même ambiance avec un rendu optimal et des lettres espacées et facile de compréhension « **Open Sans** ».

Les éléments les plus importants sont mis en gras pour accentuer leurs présences dans la page, ce qui va attirer l’œil car plus présent (exemple : titre et prix de l’annonce)

Évidemment, concernant la police d’écriture, plus élément est important, plus la police est élevée. (La police d’écriture la plus élevée est cette des titres de pages).

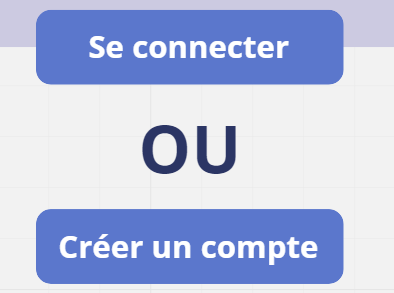
### L’utilisation de pictogramme

Pour améliorer l’accessibilité, j’ai fait le choix d’utiliser des pictogrammes lorsque que j’en avais la possibilité. Ils permettent de diriger les visiteurs vers des parties du site sans avoir à lire, ce qui aère la page et la rend plus agréable à l’œil. De plus, puisque le site est uniquement disponible en français, une personne ne comprenant pas le français pourrait facilement comprendre.

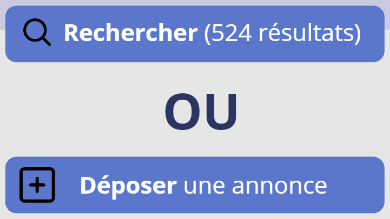
## Les boutons

Pour améliorer l’ergonomie et l’unicité des éléments du site, le design des boutons sont les mêmes pour ceux possédant le même contenu.

### Bouton simple

Comme son nom l’indique, le bouton simple possède simplement une instruction, écrite en gras pour attirer l’attention du visiteur.

### Le bouton amélioré

 Semblable au bouton simple, il possède le contenu avec le mot important en gras, mais aussi son pictogramme. Comme vous pouvez le voir, le bouton recherche est symbolisé par une loupe.

## La barre de navigation

La barre de navigation est présente tout en haut de la page est offre une facilité de navigation pour les utilisateurs du site. On va retrouver des boutons avec leurs labels ainsi qu’un pictogramme adéquat. La transition entre la barre de navigation et le contenu du site est faite pas une vague créer sur le site : <https://www.softr.io/tools/svg-wave-generator>.

Que vous soyez connecté sur un compte ou non, vous retrouverez les deux boutons principaux de cette barre de navigation ; le bouton « Déposer une annonce » et le bouton « Rechercher une annonce ».

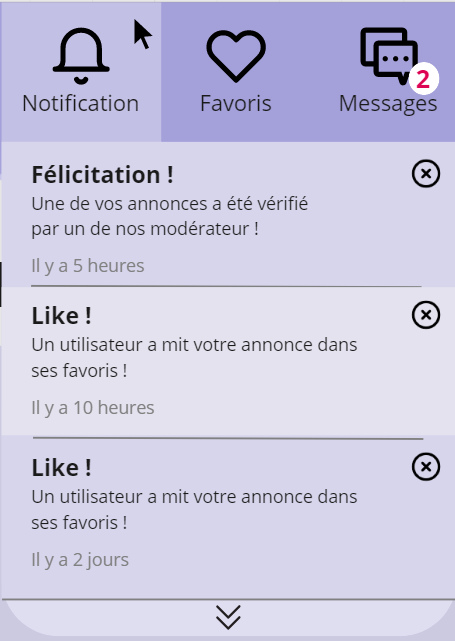
### Utilisateur anonyme

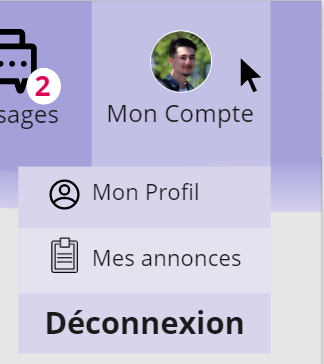
En plus des boutons principaux, lorsqu’un visiteur anonyme, c’est-à-dire non connecté, va naviguer sur le site, il aura le bouton « se connecter » qui le rédigera sur la page de connexion.

### Utilisateur connecté



Une fois connecté, l’utilisateur voir avoir accès à plus d’option lié à son compte, il va donc voir apparaitre plusieurs nouveaux boutons.

 Si le bouton notification est survolé, toute les informations relatives aux interactions du compte sont affichée.

Si le bouton « Mon compte » est survolé, l’utilisateur pourra facilement accéder à son profil, ses annonces ou se déconnecter.

## Les différents pages

### Page d’accueil

### Page de recherche

### Page d’ajout d’annonce

### Page d’une annonce

### Page de connexion

### Page de création de compte





### Page de profil – Mon profil

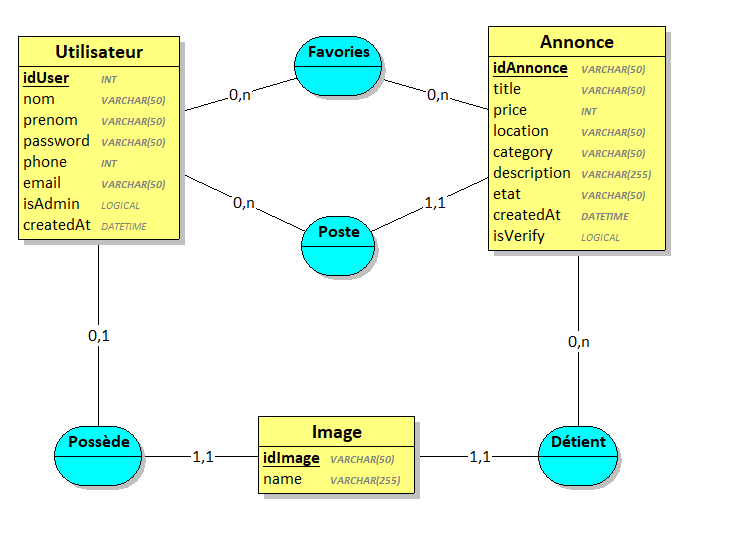
### Page de profil – Autre profil

## Ergonomie

Les deux axes principaux de l’aspect graphique du site étaient d’afficher toutes les informations nécessaires à l’utilisateur pour une navigation agréable et fluide, sans pour autant afficher trop d’informations afin d’éviter la surcharge informationnelle (l’infobésité). Les boutons de navigations sont présents sur tous les pages pour guider l’utilisateur et respecter ses deux axes.

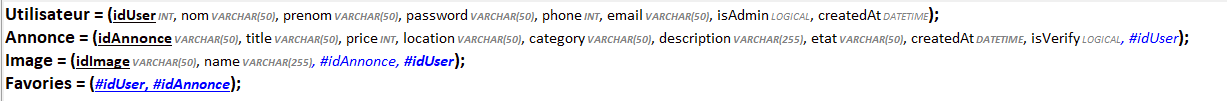
# Base de données

## Modèle conceptuel des données (MCD)



A noté qu’il s’agit de la base de données minimal, elle offre la possibilité par exemple, de rajouter des tables « Discussion » et « Message » pour permettre aux utilisateurs connectés de s’envoyer des messages en interne grâce à un espace de discussion.

## Modèle logique des données (MLD)



## Détails des choix

### Tables

#### Utilisateur

Cette table permet de stocker toutes les informations personnelles et de contact de du membre (nom, prénom, mot de passe, numéro de téléphone et adresse mail). Elle possède aussi un booléen « **isAdmin** » permettant de déterminer si l’utilisateur est un administrateur et donc s’il a les droits adéquats.

L’attribut « **createdAt** » est de type **DATETIME** et permet de stocker la date et l’heure de création du compte. L’année de création est visible pendant l’affichage d’un profil, elle permet de potentiellement rassurer l’utilisateur qui va entrer en contact avec un autre utilisateur (Exemple : BlaBlaCar) : c’est un parfait moyen de contrer les arnaques.

#### Annonce

Comme pour la table « Utilisateur », la table « Annonce » va stocker les informations importantes de l’annonce que l’utilisateur aura renseigné (titre, prix, localisation, catégorie, description, état).

Pour contrer les annonces ne respectant pas les règles du site et les arnaques, « **isVerify** » est un booléen permettant de savoir si l’annonce a été vérifié manuellement par un administrateur.

#### Image

Les images ajoutées par les utilisateurs sur le site (photo d’annonce et photo de profil) ne sont pas stockées dans la base de données, mais dans un dossier « **upload** » interne au projet.

On va alors stocker uniquement un identifiant et le nom de l’image, pour pouvoir récupérer l’image à partir de son nom.

### Relations

#### Possède (Utilisateur – Image)

L’utilisateur peut avoir une image de profil et l’image est utilisée pour un utilisateur.

#### Détient (Image – Annonce)

Une annonce peut avoir plusieurs images du produit et l’image est utilisée pour une annonce.

#### Favories (Utilisateur – Annonce)

Les utilisateurs peuvent mettre en favori les annonces qui leurs intéressent, cette relation va prendre les identifiants des deux entités concernées.

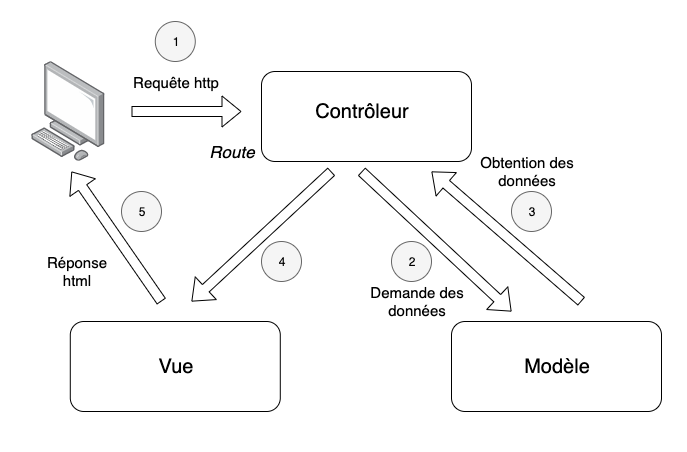
#### Poste (Utilisateur – Annonce)

Une annonce est obligatoirement créée et posté par un utilisateur. Un utilisateur peut poster aucune ou plusieurs annonces.

# Structure du code

## Modèle Vue Controller

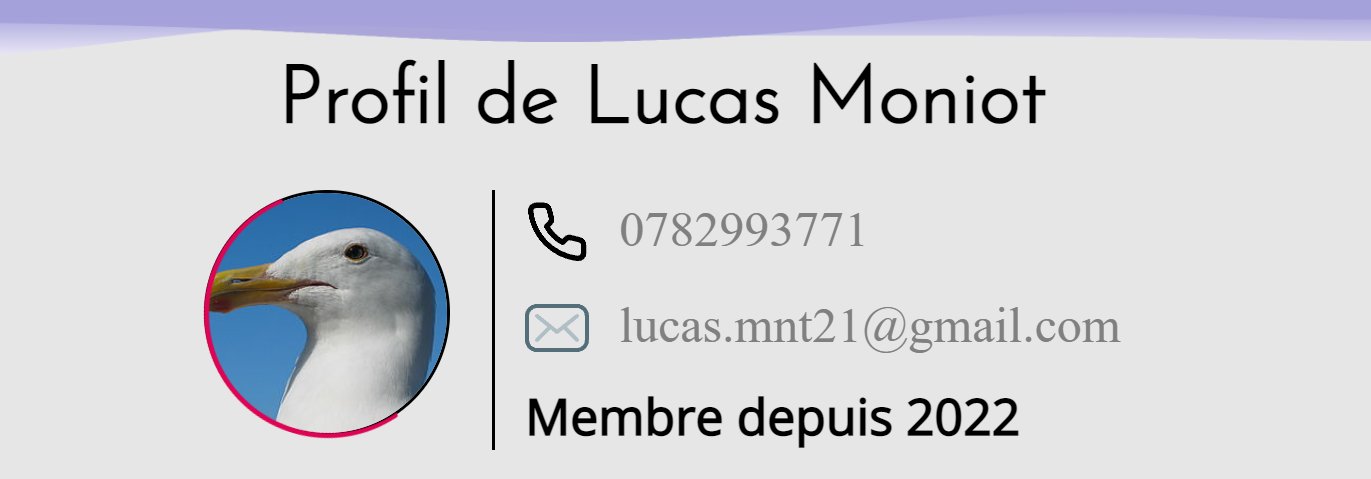
Symfony est un framework MVC (Modèle-Vue-Contrôleur). Cette architecture logicielle est l’une des plus utilisées pour les applications Web, elle consiste à découper l’application en modules qui ont trois responsabilités différentes :

* Les modèles gèrent l’accès aux données (le plus souvent dans une base de données).
* Les vues sont les interfaces graphiques présentées à l’utilisateur.
* Les contrôleurs contiennent la logique des actions à effectuer suivant la requête de l’utilisateur.

L’objectif de ce pattern est de bien organiser son code source en séparant la logique du code.

## Les templates

Comme dit plus haut, dans mon projet j’ai utilisé Twig, le moteur de templates pour le langage de programmation PHP utilisé par défaut par le framework Symfony. Il me permet de facilement créer chaque page du site en utilisant des variables à afficher.

Pour la page de profil d’un utilisateur, on va par exemple facilement afficher les informations de l’utilisateur en question.

Pour la barre de navigation, je n’ai pas eu besoin de l’implémenter dans chacune de mes pages. J’ai simplement créé un template principal avec la barre de navigation puis j’ai créé mes templates à partir du template principal.

# Fonctionnalités codées

Création de compte

Modification de compte

Création d’annonce

Recherche d’annonce

Visite des autres comptes

Supprimer annonce

# Conclusion :

## Difficultés rencontrées

Ce projet a été vraiment utile et intéressant pour moi, de par la découverte de Symfony et de Doctrine mais aussi par l’utilisation de plusieurs technologie récente et nouvelle pour moi. Bien qu’il existe plusieurs tutoriels disponibles sur internet, la principale difficulté pour moi a été de trouver les syntaxes dans les documentations car la plupart des tutoriels utilisaient une version antérieure de Doctrine, et j’ai pu découvrir que d’une version à une autre, la syntaxe à drastiquement changée.

Un autre point vraiment difficile pour moi a été le manque de temps pour finaliser le projet, notamment le retard que j’ai pu prendre à cause de la mise en forme (CSS) des différentes pages du site, ainsi que l’ajout des fonctionnalités (affichage du mot de passe avec un bouton, carrousel pour les photos des annonces, …). Je pense à l’avenir découvrir un framework CSS ou utilisé le Webpack Encore qui propose l’utilisation de CSS simplifié (SCSS) pour parer à ce problème.

## Les points positifs / ce que j’ai aimé

## Une suite pour ce projet ?

Finir les fonctionnalités prévues initialement =>

Message entre les utilisateurs

Volonté d’ajouter des API pour la fluidité

## A venir

Créer d’autre projet