

2021
DOSSIER DE PROJET



Les
Paniers
de
L'Isac



Avant-Propos	4
Remerciement	4
Liste des compétences couvertes par le référentiel	4
Résumé du projet	5
Environnement De Travail	6
Les outils utilisés	6
Cahier Des Charges	8
Analyse de l'existant	8
Les objectifs du site	8
Les objectifs quantitatifs	8
Périmètre du projet	9
Graphisme et ergonomie	9
Spécificités et livrables ☒	9
Veille	11
Charte graphique	12
Wireframe et maquettage	13
Maquetter une application	13
Arborescence du site	15
Arborescence détaillée	16
Réalisation du site	17
Planification du projet	17
Définition des rôles	18
Mcd / Mld et Diagrammes Uml	19
MCD	19
UML	20
MLD	21
Use Case Diagramme	22
Diagrammes d'activités	23
Développement Back-End	25
Création du squelette et de la base du projet	25

Construction de la base de donnée	25
Création des entités	26
Utilisateur et sécurité	28
Réaliser une interface utilisateur avec gestion de contenu	30
Interface d'administration	30
Gestion des rubriques	31
Front-End	35
Mise en place du Framework CSS	35
Mise en place des icons	35
Mise en place des typos	36
Mise en place du CSS	36
Réaliser une interface web statique et dynamique	37
API OpenStreet Map	38
Exemples de codes	40
Recherche FullText	40
SELECT2	45
Filtre de recherche	47
Sécurité	48
Veille sur la sécurité	48
Test unitaire de securité	49
Sécurité Recaptcha	51
Installer reCaptcha Google V3 sur son site internet	51
Mise en Production	52
Google Analytics	53
Intégrer Google Analytics dans son projet web	53
Le Référencement	54
Test de Performance	55
Recherche en Anglais	56
Vie Privée	58
Conclusion	60

Ce dossier Professionnel a été réalisé dans le cadre d'un projet de transition professionnelle où j'ai eu l'opportunité d'obtenir un financement auprès de Transition Pro Pays de la Loire afin de bénéficier d'une formation de Développeur Web et Web mobile au sein de l'organisme de formation ARINFO de Saint Nazaire. Cette formation a duré du 07 juin 2021 au 15 août 2021.

Je suis actuellement infographiste et photographe dans le Print, et j'ai choisi cette formation, car j'ai toujours voulu travailler dans la conception de site internet et le développement web. Je pense que les connaissances apprises lors de cette formation et ma maîtrise de la suite adobe peuvent être un atout pour moi.

Remerciement

Je tiens tout d'abord à fortement remercier Nicolas Gicquel notre formateur, pour la qualité de sa formation, sa disponibilité, sa pédagogie. Elodie Boinet directrice du centre ARINFO de Saint-Nazaire pour son soutien et ses conseils pour réaliser cette formation, ainsi que Marion Lamare qui m'a donné de précieux conseils pour mon CV et ma recherche de stage.

Je remercie également mes Conseillers en évolution professionnelle chez CIFOR ACCOMPAGNEMENT, Karine Billard et Kamel Maiza pour leurs conseils, suivi de dossier pour l'obtention de mon financement et leur bienveillance

Enfin je remercie mes camarades de promo, Charley, Elouan, et Marc pour l'entraide bienveillante dont ils ont tous fait preuve ainsi que nos camarades de Nantes, Anthony, Antoine, Jonathan et Sacha en distancielle, mais toujours là sur discord pour donner un coup de main.

Liste des compétences couvertes par le référentiel

- 1 - Maquetter une application
- 2 - Réaliser une interface utilisateur web statique et adaptable
- 3 - Développer une interface utilisateur web dynamique
- 4 - Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce
- 5 - Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité
- 6 - Créer une base de données
- 7 - Développer les composants d'accès aux données
- 8 - Développer la partie back-end d'une application web ou web mobile
- 9 - Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce

L'idée de ce projet est né pendant le premier confinement, et le constat qu'il était difficile de trouver les producteurs locaux sur internet.

Habitant en campagne et entouré d'agriculteur et producteur, l'idée m'est venu de créer un site internet recensant tout les acteurs locaux liés aux métiers de bouches, du producteur au revendeur.

Le But est de promouvoir les acteurs locaux et les circuits courts et ainsi mettre en relation les producteurs et revendeur des différents métier autour de l'alimentaire, avec les habitants de leur secteur,

J'ai donc pris comme épiscetre à mon projet, ma commune, celle de Guenrouët, et l'étendre aux communes des alentours (dans un rayon de 15km).

Concernant la charte graphique, tout est a définir.

Le nom du site est «les paniers de l'Isac» le panier, pour l'idée de faire ses courses, son marché ou même sa cueillette et l'Isac, c'est le nom de la rivière courant au milieu de ce territoire.

Le choix des couleurs se portent vers des teintes rappelant la campagne, comme le vert, le marron et le bleu et par des nuances pastels pour ça douceur et le bien vivre.

Ce site est un annuaire, qui se divise en 2 grandes rubriques Nos Producteurs et Nos Commerçants. Il y a aussi une rubrique actualité permettant d'informer les visiteurs sur toutes les manifestations locales et nouveautés du secteur lié à notre thème.

Pour la gestion de contenu, chaque adhérent pourra se connecter afin de modifier ses informations personnelles, ou créer leur propre fiche, quand à l'administrateur, il pourra gérer les adhérents, les contacts et créer de nouvelles actualités.

Des wireframes ont été créés afin de valider la charte graphique du site en mode desktop et mobile et des diagrammes de séquences, UML ainsi qu'un MCD/MLD ont été réalisés, amenant un visuel sur les relations et le fonctionnement du site.

Élaboration d'une arborescence détaillée, comprenant un détail global des différentes pages avec pour chacune d'entre elles leurs contenus.

Pour la partie confidentialité, des mentions légales seront mise en place ainsi qu'une politique de confidentialité couplé à une gestion des cookies.

Le tout sera élaboré via les technologies PHP, Symfony et MySQL pour la partie Back-End du site.

Ainsi que les technologies HTML/CSS/ Sass, Bootstrap et Javascript/JQuery pour la partie FrontEnd du site.

Les outils utilisés

Voici les différents outils, logiciel, Framework utilisés pour le développement du site.

BACK-END :



- Pour l'environnement du serveur local j'ai utilisé « WAMP » qui m'a permis de tester et de développer mes scripts PHP sur le PC du centre de formation et « XAMPP » sur mon i mac perso. Lors de la formation j'avais testé « MAMP » mais j'avais quelque problème avec.



- Pour le plus gros morceau j'ai utilisé « Symfony » qui est un Framework Open-Source MVC (Modèle - Vue - Contrôleur) basé sous le langage PHP (Hypertext Preprocessor).



- En ce qui concerne la base de données j'ai utilisé « MySQL » (My Structured Query Language) qui est un gestionnaire Open-Source couplé à « PHP My Admin » une interface simple et facile d'accès.

FRONT-END :



- Tout d'abord j'ai utilisé le préprocesseur CSS (Cascading Style Sheet) « SASS » (Syntactically Awesome Stylesheets). Il permet de mieux gérer son code CSS et facilite les mises à jour en cas de modification du code par exemple avec des Mixins ou des Extends.



- Ensuite j'ai utilisé le Framework « Bootstrap » que j'apprécie tout particulièrement pour son côté responsive ainsi que pour ses menus burger. Bootstrap est une collection d'outils utiles à la création du design 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 et également une bibliothèque d'icône, ainsi que des extensions JavaScript en option. Son Système de Grid est particulièrement efficace.



- Pour les effets dynamiques du site j'ai utilisé le langage « JavaScript » et sa bibliothèque « JQuery ».

LOGICIELS :



Concernant mon environnement de développement je me suis tournée vers l'ide Visual Studio Code pour lequel un très large panel d'extensions existe et facilite grandement la lecture du code

avec une aide non négligeable pour certaines d'entre elles lorsqu'il s'agira d'écrire des requêtes ou créer des fonctions dans les contrôleurs.

- Coté terminal j'ai utilisé powershell qui est intégré à Visual Studio Code, couplé à la gestion de dépendance composer il m'a permis la construction de mon projet, le téléchargement des différents bundles ainsi que d'effectuer les mises à jour nécessaire



- Le Projet est synchronisé sur la plateforme « GitHub » afin de faciliter la programmation avec la permission de revenir en arrière au niveau du code en cas de problème par le biais de « Branch » qui correspondent à un clone du projet « Branch Main », c'est-à-dire le dossier projet validé et sans bugs sur lequel on peut se greffer tout en travaillant dans un environnement sécurisé.

Cela m'a permis durant ma phase de développement d'en créer plusieurs avec pour chacune un thème bien particulier afin de découper au mieux mon projet et ainsi effectuer des tests sans prises de risques.

GitHub offre aussi la possibilité de partager son code sur plusieurs ordinateurs avec un système de 'clone' ce qui m'a énormément servi lors de mes premières phases de développement et ma facilité l'échange en cas de problème auprès de mon formateur qui pouvait y accéder à n'importe quel moment, mais également sur mes deux sites de développement, centre de formation, et mon domicile.



J'ai surtout utilisé GitHub desktop, pour mes commit, mes push et mes pull.

Analyse de l'existant

Nouveau projet, sans existant. Une seule personne engagées.

Les objectifs du site

Les paniers de l'Isac :

C'est un annuaire recensant tout les métiers de bouches* dans un rayon de 15 à 16 km autour de Guenrouët.

Le But est de promouvoir les acteurs locaux et les circuits courts et de mettre en relation les producteurs locaux et revendeur des différents métier autour de l'alimentaire*, avec les habitants de leur secteur.

C'est un site à vocation informatif (palier la difficulté de trouver nos producteur locaux) et peut-être de e-commerce dans un second temps

Dans un premier temps, je souhaite répertorier dans un «annuaire» les producteurs et revendeurs locaux

Dans un second temps, je souhaite y ajouter une rubrique restaurant, et de pouvoir proposer un système de e-commerce avec commande en ligne.

Et dans un troisième temps un moyen de paiement en ligne.

Les visiteurs visés sont des adultes (18 à 75 ans), cherchant des produits locaux et les circuits courts.

Les objectifs quantitatifs

Et un volume de prospect de mille personnes. (un total d'environ 68 000 habitants dans cette zone en comptant des petites villes tel que Pontchâteau, Blain)

Les services proposés :

Un annuaires :

- Producteur agriculteur (en vente direct, sur marché, AMAP...)
 - . Fruits et légumes
 - . Viandes (bovins, volailles, ovins porc...)
 - . Produit transformé, Miel
 - . Produit laitier
- Commerçant :
 - . Boulangerie pâtisserie

- . Boucher charcutier traiteur
- . Poissonnier
- . Supérette
- . Épicerie fine

Un service de mini site pour les adhérents?

Un service de commande en ligne (et paiement dans une troisième phase)

Avec un fichier client (inscription en ligne)

Un CRM pour mettre en ligne les produits à vendre pour chaque «Adhérent»

Un système de géolocalisation du client et des Adhérents.

Périmètre du projet

Mon site ne doit pas être multilingue

Mon site aura une boutique dans un deuxième temps.

Et une solution de paiement dans un troisième temps.

Utilisation de la géolocalisation

Création de compte client au moment de la création de la boutique.

Création de compte Adhérent (renseigner leur produit à vendre y insérer photo du produit, et picto si nécessaires (AB etc...))

Graphisme et ergonomie

Aucune charte graphique, tout était à définir !

Spécificités et livrables ☒

Contenus de votre site que vous allez devoir créer, ou trouver :

Logo et charte graphique

Trouver un certain nombre de producteur locaux, commencer de Guenrouet, et élargir le rayon au fur et à mesure.

(Guenrouët , St Gildas des bois, Quilly, Plessé, St annes sur brivet, Dréfféac, Sévérac, Fégréac, Campbon, Théhillac, Bouvron, Pontchâteau, Rieux, Avessac, St Dolay, Blain, Guéméné Penfao, Le Gavre, Missillac)

Photos

Contraintes techniques :

Il n'y a pas de nom de domaine ni de serveur existant.

Choix du nom de domaine: www.lespanierdelisac.fr

Choix de l'hébergeur : <https://www.o2switch.fr/>

Afin d'établir une charte graphique du site web et de correspondre au mieux à mes attentes, j'ai effectué différentes recherches afin de trouver des renseignements sur divers sites de vente de produit locaux ou d'annuaire. Mes recherches m'ont beaucoup aidé à trouver des solutions design et efficace pour répondre aux besoins de l'association, j'ai découvert des sites très intuitifs et intéressant. Voici les différents sites qui m'ont beaucoup aidé pour le design et les fonctionnalités que les sites ont tendances à utiliser :

Graphiquement je voudrais me rapprocher de ses 2 sites :

<https://www.tauzietnco.fr/>

<https://www.pourdebon.com/>

Autres site intéressant graphiquement :

<https://www.potagercity.fr/>

<https://www.agripousse.com/fr/>

Une philosophie se rapprochant de ce que je souhaite :

<https://www.monpanierderetz.fr/>

<https://www.connexionpaysanne.fr/>

<https://www.terroirs44.org/>

<https://www.cagette.net/>

<https://www.acheteralasource.com/>

<https://www.dulocaldansmonassiette.fr/>

Autres site :

<https://www.panierduclos.fr/index.html>

<https://vitemonmarche.fr/>

<https://lecourtccircuit.fr/index.php>

<https://www.drive-fermier.fr/>

On trouve également tout les drive des grandes surfaces :

<https://www.leclercdrive.fr/>

<https://www.coursesu.com/drive/home>

<https://www.auchan.fr/courses>

<https://www.carrefour.fr/services/drive>

<https://www.casino.fr/>

<https://www.intermarche.com/>

Le logotype :



Typographie choisi pour le logo :

a Antara Distance



Typographie choisi pour le site :

Spartan pour le texte



et Roboto slab pour les titres



Choix des couleurs :



RVB : 180 192 146
#B4C092



RVB : 126 202 218
#7ECADA



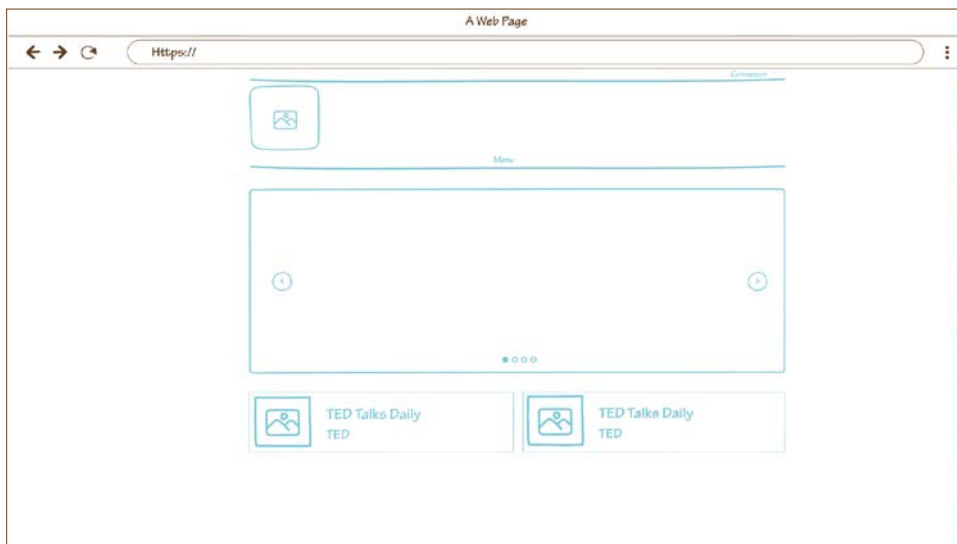
RVB : 104 59 17
#683B11

Maquetter une application

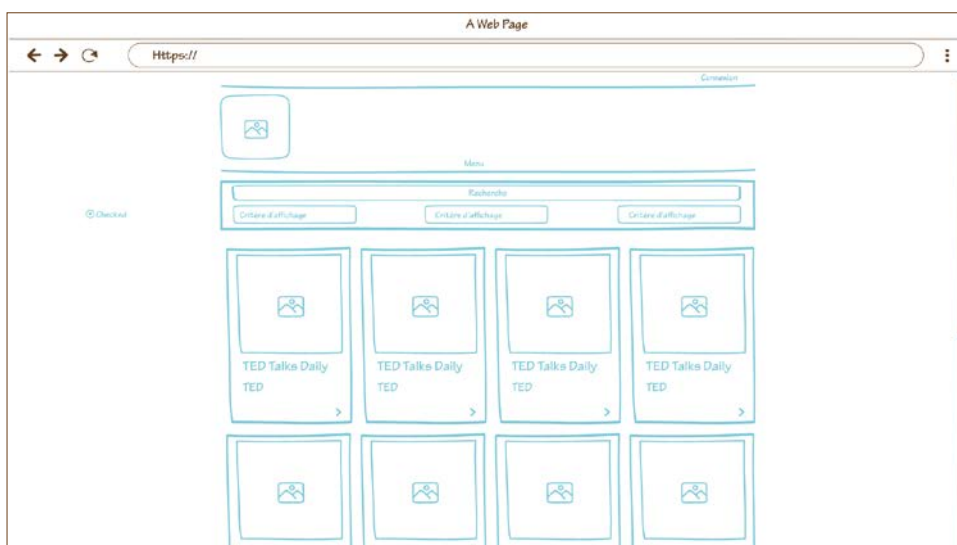


Afin de réaliser une maquette j'ai utilisé l'application XD, car elle est beaucoup utilisée pour le prototypage de site web. J'avais déjà vue l'application gratuite Draw.IO pour créer des wireframes et diagramme pendant ma formation, mais je voulais découvrir une nouvelle application.

Page d'accueil. À droite nous avons le wireframe de la page d'accueil responsive.



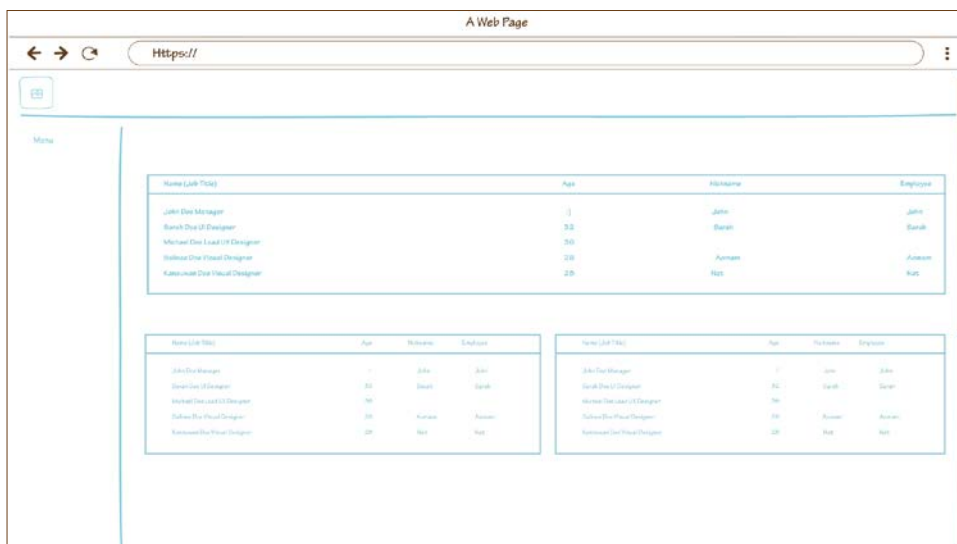
Page rubrique. À droite nous avons le wireframe d'une rubrique responsive.



Page fiche. À droite nous avons le wireframe de la page de la fiche responsive.

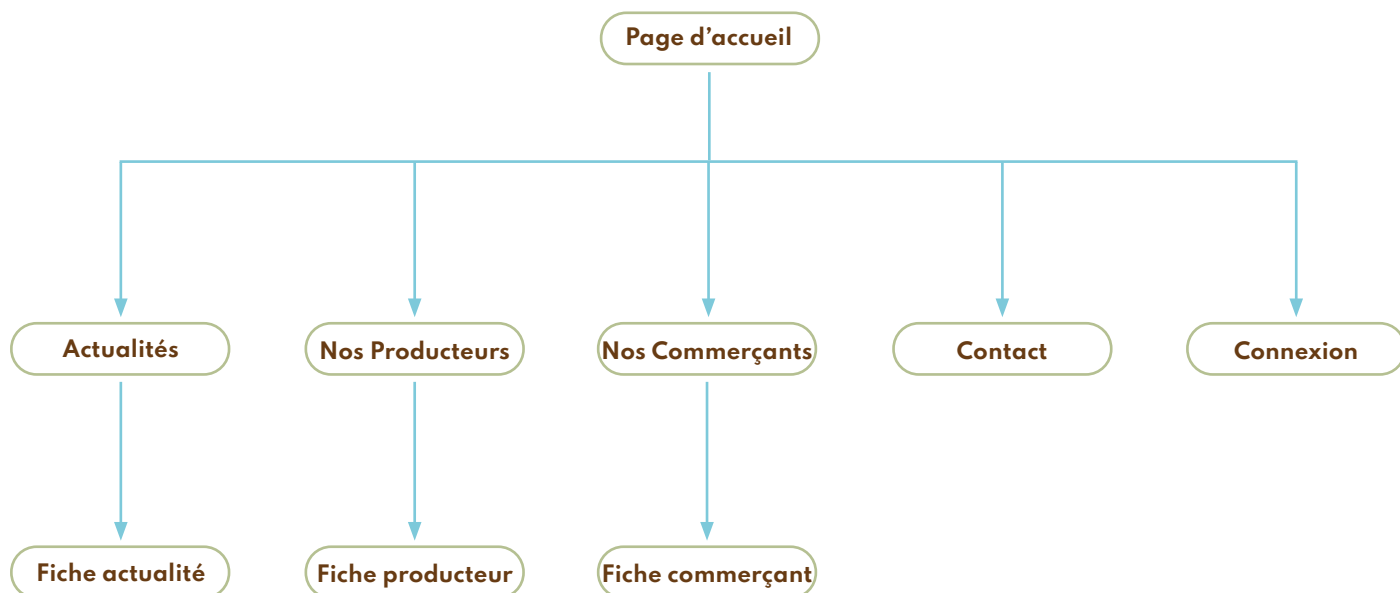


Page d'administration.

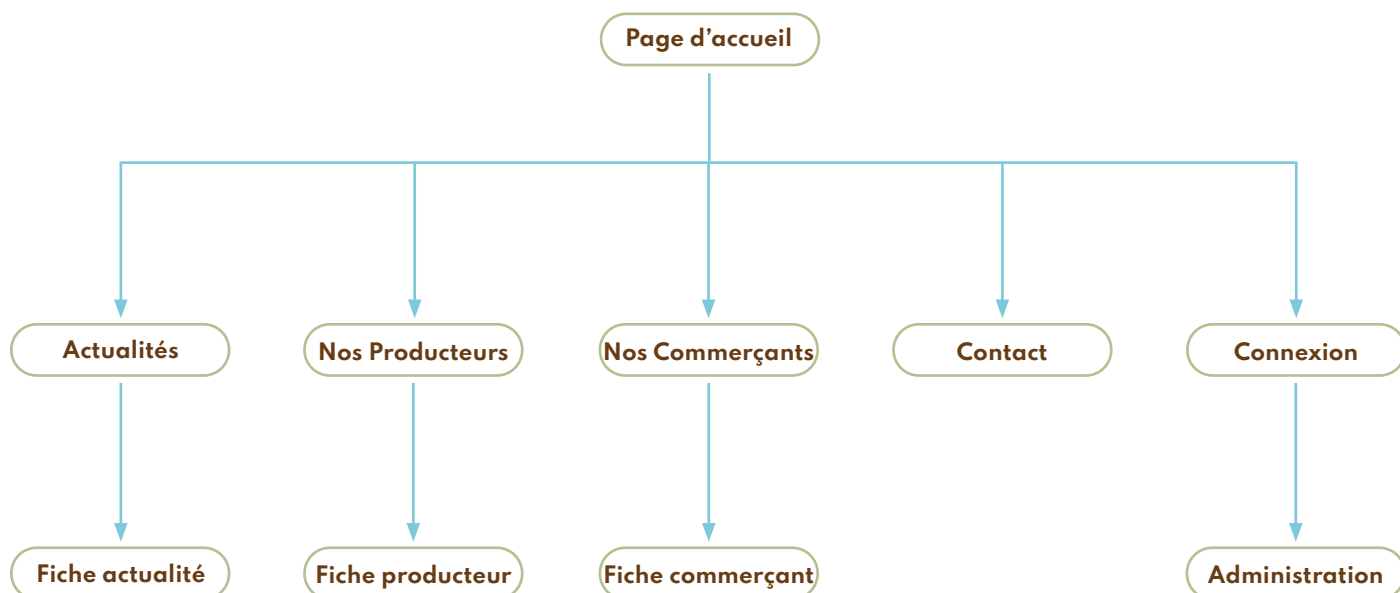


Voici l'arborescence du site en fonction des différents statuts :

Accès Visiteur



Accès Adhérent



Arborescence détaillée

Chaque page du site contient plusieurs éléments dont voici les principaux :

ACCUEIL :

- Petite introduction explicatif.
- Carrousel d'images (mise à avant de certain Adhérent).
- les trois dernières actualités.

ACTUALITÉS :

- Actualité par ordre chronologique.
- Actualité d'un adhérent, ou manifestation local.
- > lien vers la fiche actualité avec tout le détail.

NOS PRODUCTEURS :

- Système d'organisation de classement.
 - Affichage producteurs
- > lien vers la fiche producteur avec tout les détails et l'iframe de localisation.

NOS COMMERÇANTS :

- Système d'organisation de classement.
 - Affichage commerçants
- > lien vers la fiche commerçant avec tout les détails et l'iframe de localisation.

CONTACT :

- Formulaire de contact.

CONNEXION :

- Formulaire d'enregistrement avec texte explicatif.
- Formulaire de connexion.

DÉCONNEXION

ADMINISTRATION adhérent

- Page de gestion de ça fiche (création, modification, suppression).

ADMINISTRATION admin

- Page de gestion des adhérents (création, modification, suppression).
- Page de gestion de fiches (création, modification, suppression).
- Page de gestion des actualités (création, modification, suppression).
- Consulter les e-mails.

Planification du projet

J'ai réalisé une liste de tâches détaillé dans l'ordre chronologique.

BACK-END :

- Définir les rôles utilisateur.
- Création de diagrammes.
- Création du squelette Symfony.
- Création de la base de données.
- Création des entités avec leurs relations.
- Mise en place du service global d'authentification utilisateur.
- Sécurité du site et utilisateur.
- Mise en place de la gestion de cookies, reCAPTCHA
- Test de sécurité

FRONT-END :

- Mise en place du framework CSS Bootstrap
- Travail sur la partie Header
- Travail sur la partie Footer
- Intégration des différents blocs en fonction des pages
- Travail sur le responsive design

FINALISATION :

- Achat du nom de domaine
- Passage en production
- Phase de test
- Correction des bugs
- Test du site en version mobile
- Livraison du site

Définition des rôles

Pour ce qui concerne les rôles, j'ai fait le choix de créer trois rôles, le rôle de USER, celui des adhérents (EDITOR), et celui de l'administrateur (ADMIN).

Le User

Le user est le premier niveau de rôle, il peut juste consulter son compte, et le modifier, et également faire une demande de changement de rôle pour passer «EDITOR». Ces fonctions pourront évoluer dans le cas où ce site évoluera vers un site marchand.

L'Adhérent

L'adhérent est le « EDITOR », il pourra accéder à l'espace Administration de sa ou ses fiches descriptives, les modifier ou les supprimer

L'Administrateur

Dans le cas de ce site, l'Administrateur n'est autre que le Super-Administrateur, c'est à dire le développeur. Il aura accès à la totalité du site et la possibilité de le modifier. Il aura accès au serveur à la base de données ainsi qu'au code source du site. Il pourra donc gérer les adhérents, leurs fiches descriptives (créer, modifier et supprimer).

Créer, modifier ou supprimer les actualités.

Consulter les e-mails et les supprimer.

Lors de ma formation, j'ai appris à réaliser des diagrammes afin de me donner une forme de feuille de route qui m'a beaucoup aidé dans la réalisation du site.

Le visuel que me donne ces éléments me permettent de mettre en avant la partie utilisateur et sécurité du site ainsi que le fait de prévoir les différents éléments que je me devais d'implémenter au niveau de mon code ainsi que des bundles que j'allais utiliser pour arriver au résultat escompté.

J'ai alors réalisé un MCD, un MLD, un Use Case Diagram, ainsi que des diagrammes d'activités.

MCD

Looping

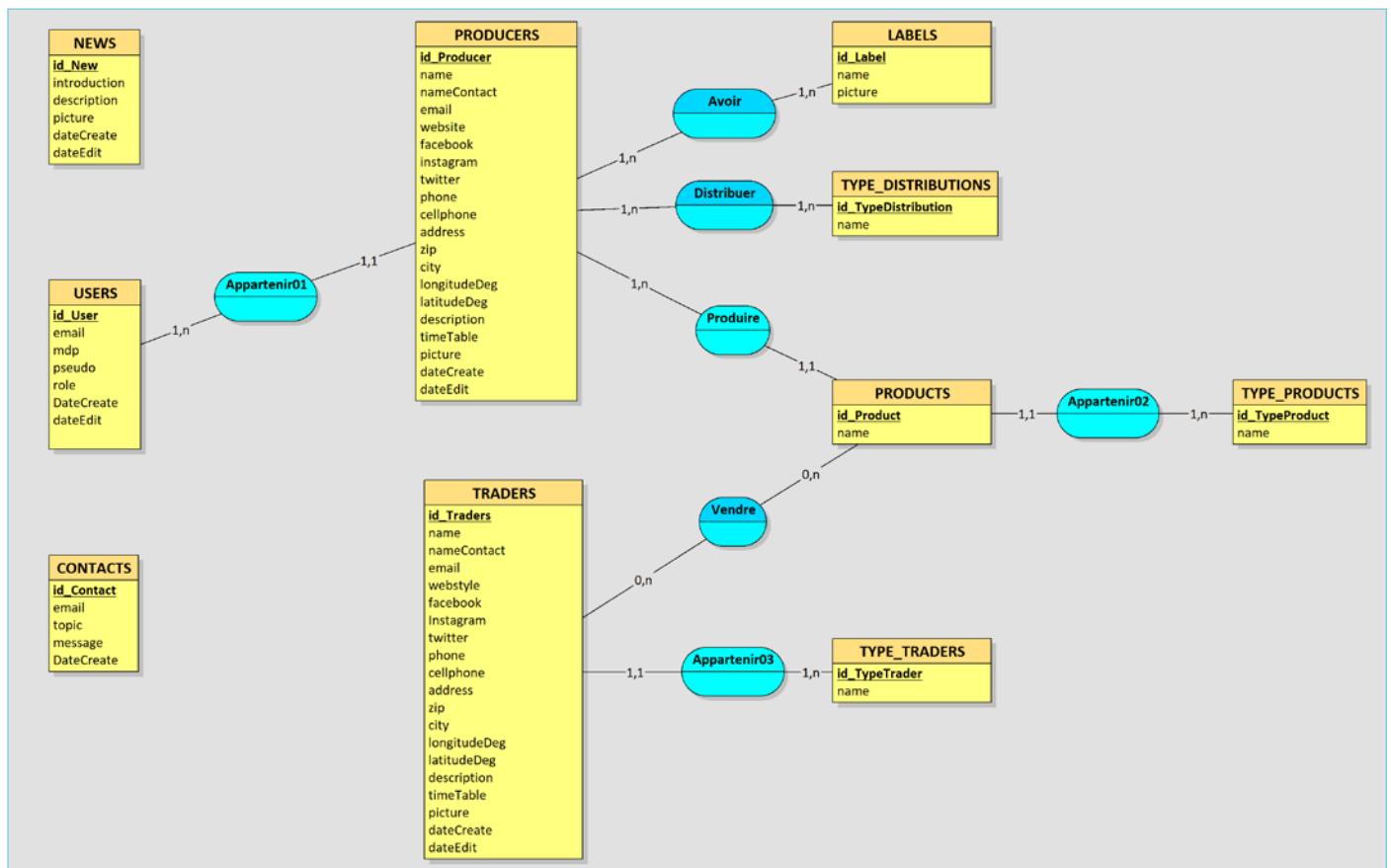
Afin de réaliser mon MCD (Modèle Conceptuel de Données) j'ai utilisé le logiciel Looping, logiciel très efficace et très simple d'utilisation.

Les MCD m'ont permis de construire plus facilement ma base de données en la schématisant.

Les différentes 'entités', en majuscules seront autant de 'classes' dans Symfony, et les propriétés qui les caractérisent autant de 'fonctions' dans ces 'classes'.

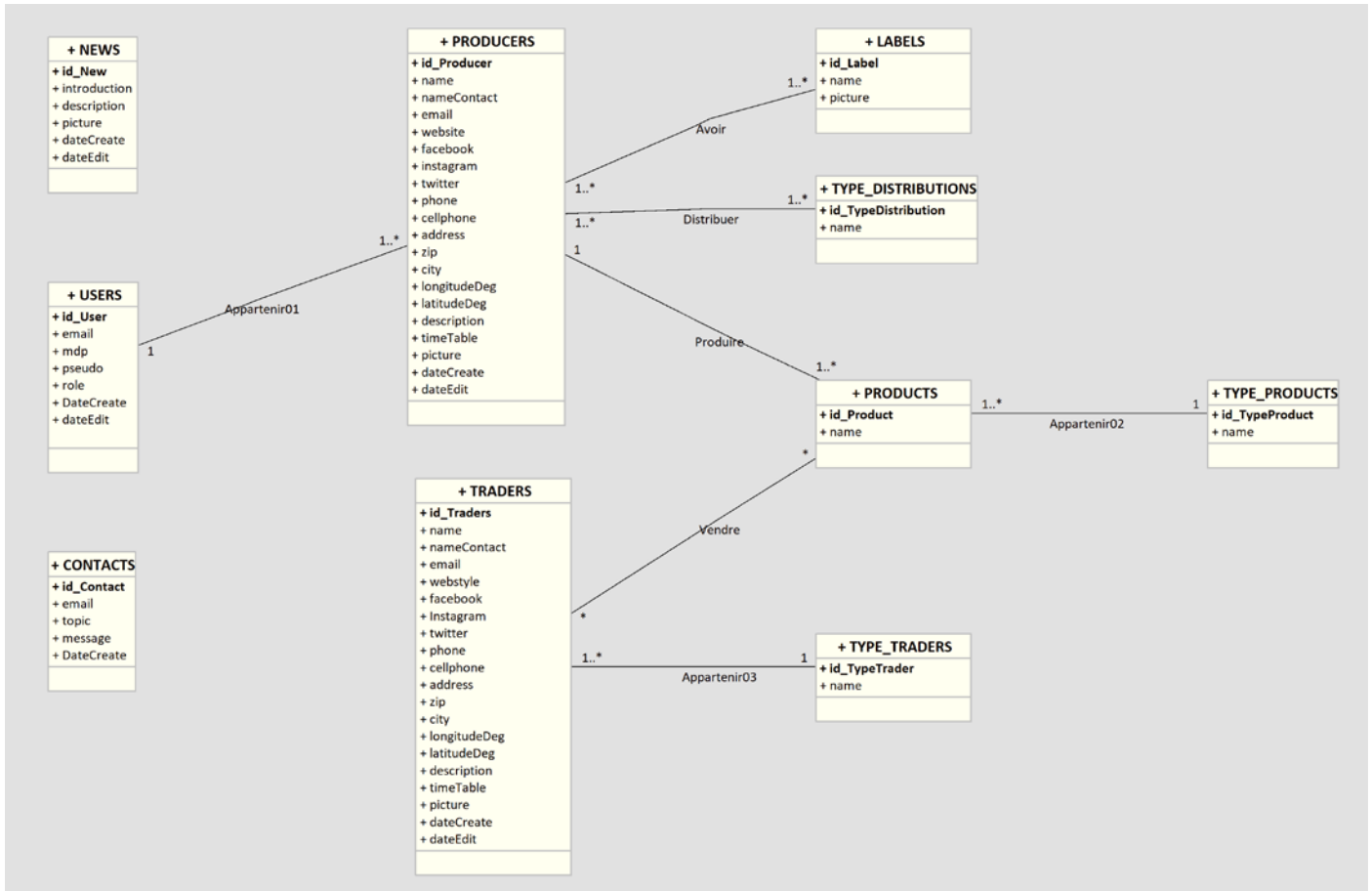
Lors de la conception de ce MCD, j'ai réalisé qu'un système avec héritage aurait été plus judicieux, mais avec mes connaissances actuels et le peu d'information du système avec héritage sur Symfony, la tâche aurait été trop compliquée dans un premier temps.

Voici le MCD que j'ai réalisé pour le site :



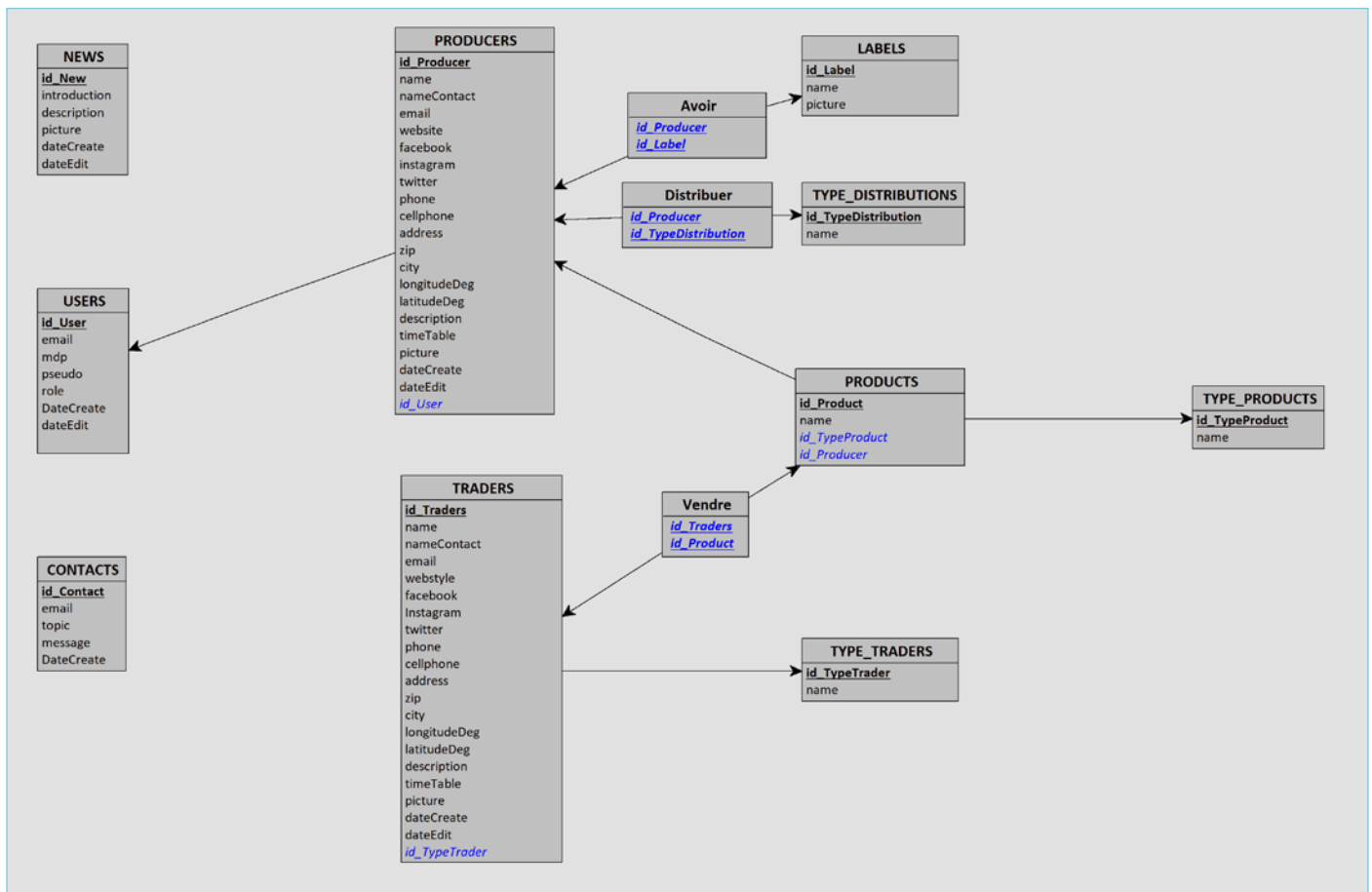
Afin de réaliser mon UML (Langage de Modélisation Unifié) j'ai simplement utilisé la fonction qui me permet de transformer mon MCD en UML via Looping.

Voici le UML :



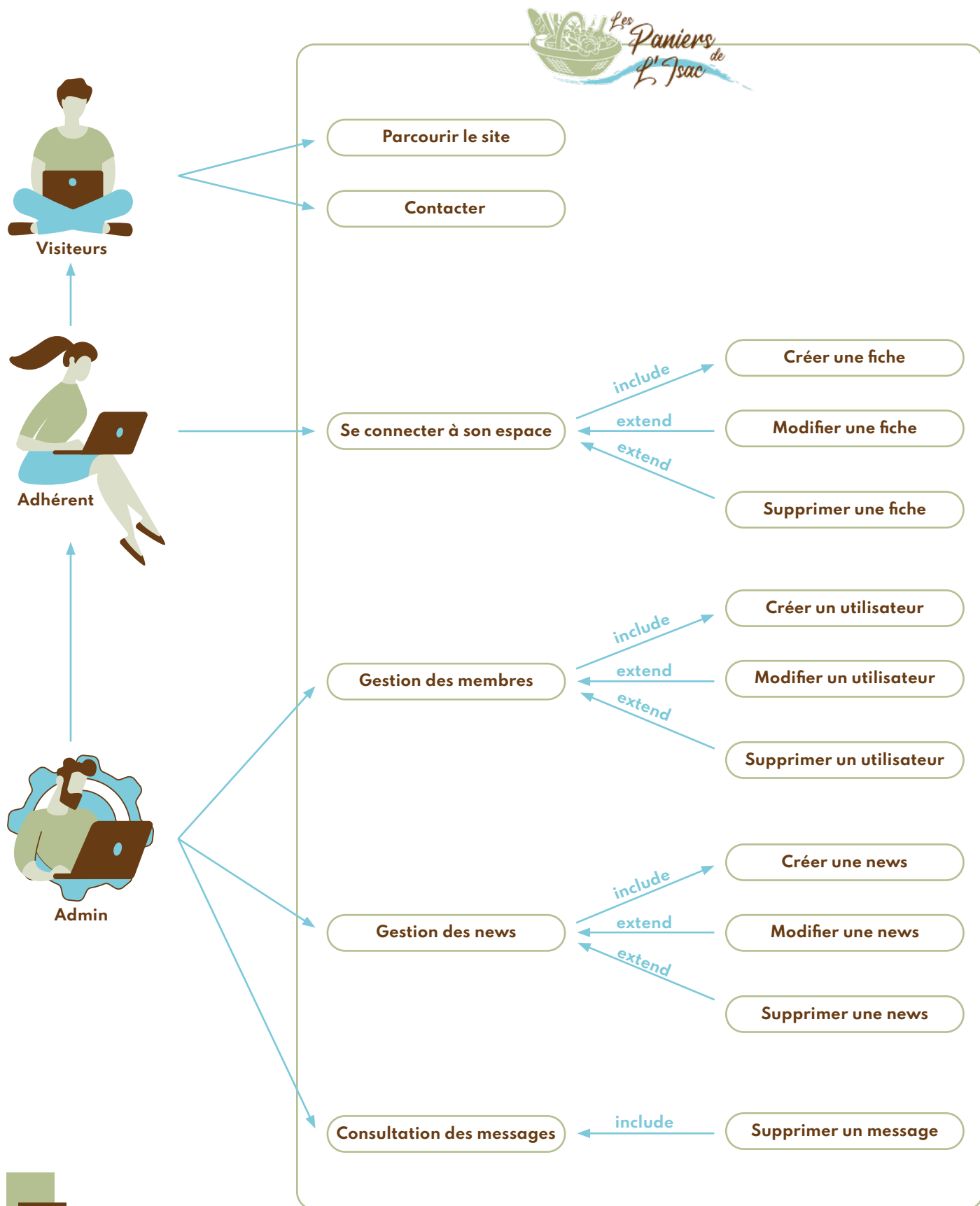
Afin de réaliser mon MLD (Modèle Logique de Données) j'ai simplement utilisé la fonction qui me permet de transformer mon MCD en MLD via Looping.

Voici le MLD :



Use Case Diagramme

Afin d'avoir une vision globale du comportement fonctionnel du site en fonction de chaque rôles utilisateur, j'ai réalisé un diagramme de cas d'utilisation.



Diagrammes d'activités

Afin de pouvoir visualiser de manière beaucoup plus précise les interactions entre usager et l'interface du site j'ai développé des diagrammes de séquence. Voici deux de mes diagrammes d'activités :

Diagramme d'activité connexion utilisateur

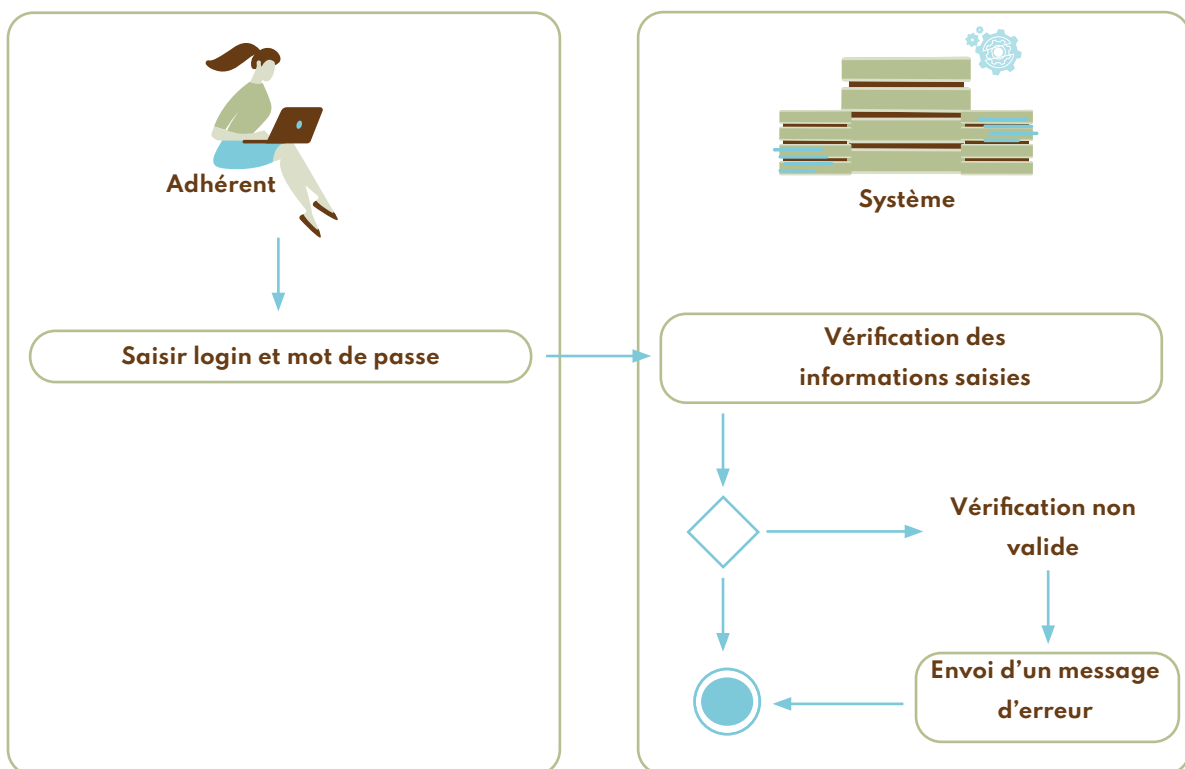
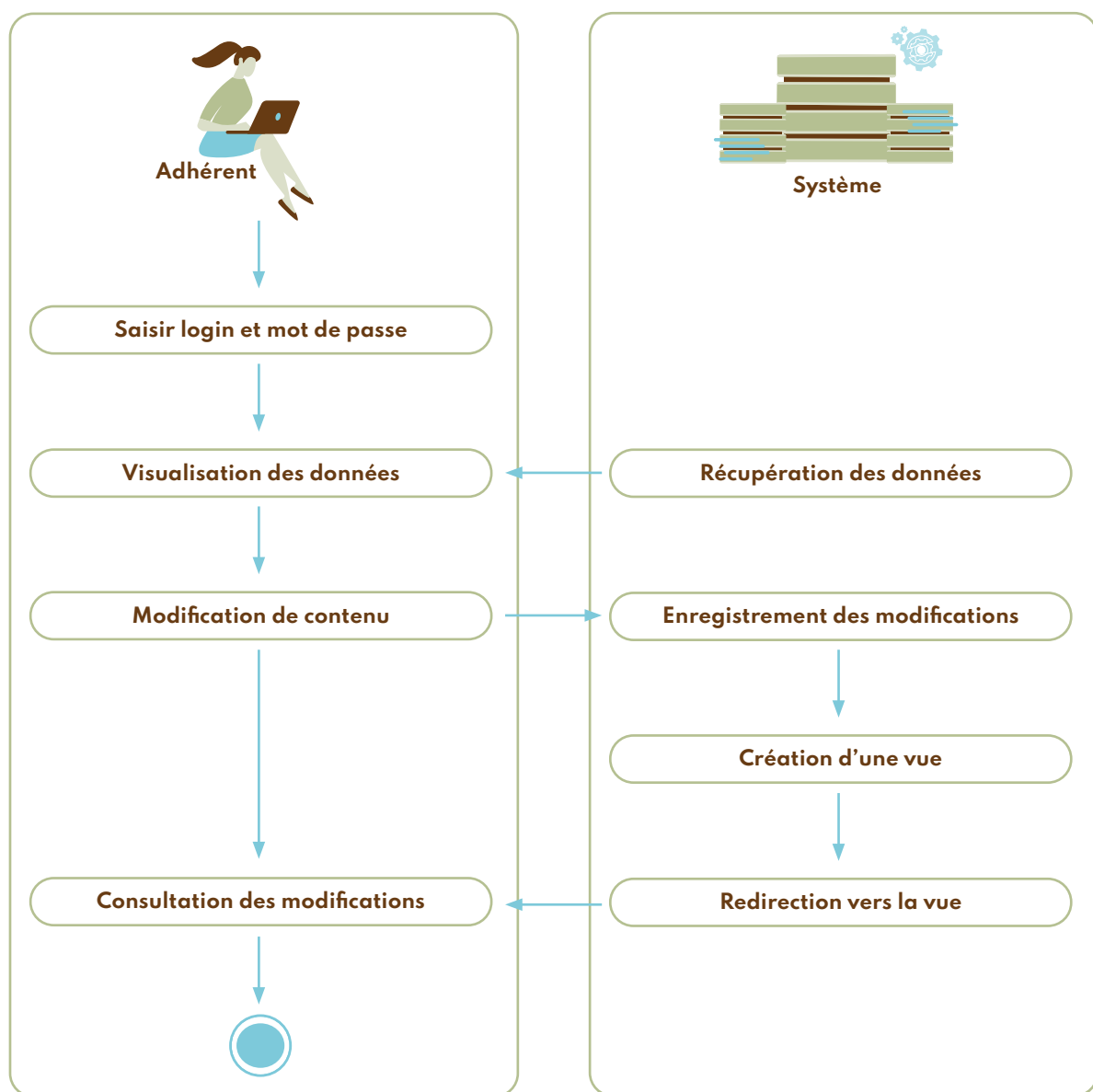


Diagramme d'activité modification de contenu



CRÉATION DU SQUELETTE ET DE LA BASE DU PROJET

Pour commencer, j'ai vérifié que mes deux environnements de travail étaient bien configurés de la même façon, pour cela j'ai tapé les lignes de commandes suivantes sur le terminal de mon IDE :

```
composer -v  
php -v
```

Pour démarrer mon projet j'ai commencé par créer un nouveau dossier de projet dans le dossier « htdocs » de mon serveur XAMP sur mac, et dans le dossier « www » de mon serveur wamp sur PC.

J'ai ouvert ce dernier avec le terminal de mon IDE en utilisant cette ligne de commande bash :

```
➔ ~ /Applications/XAMP/htdocs
```

pour rappel, j'ai utilisé quelques commandes de base pour naviguer sur le terminal :

- La commande **cd** vous permet de changer de répertoire
- Contrairement à la version Windows la commande **cd** utilisée seule ramène au répertoire par défaut de l'utilisateur (ou du root).
- Pour « remonter » d'un répertoire (aller à son parent) on utilise la commande **cd ..**
- La commande **ls** liste les fichiers et les sous-dossiers.
- La commande **pwd** vous permet de savoir dans quel répertoire de l'arborescence vous êtes localisé.

Une fois dans le dossier j'ai exécuté la ligne de commande `composer` afin de créer le squelette symfony pour site web :

```
➔ composer create-project symfony/website-skeleton LesPaniersdelisac
```

Ensuite pour démarrer notre serveur de développement nous allons utiliser la commande, du serveur interne de php.

```
➔ php -S 127.0.0.1:8000 -t public
```

Construction de la base de donnée

Mon projet étant opérationnel, il me fallait maintenant mettre en place ma base de données. Pour ce faire je me devais de modifier le fichier **.env** qui se situe à la racine du projet Symfony.

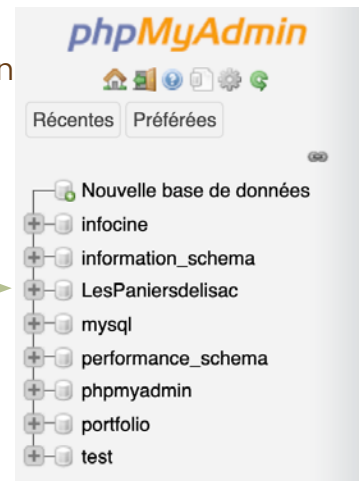
J'ai modifié la variable d'environnement « **DATABASE_URL** » afin d'obtenir une relation entre mon projet et la base de données sous MySQL.

```
DATABASE_URL=mysql://root:''@127.0.0.1:3306/LesPaniersdelisac
```

J'ai ensuite entré cette ligne de commande dans le terminal afin de créer ma base de données :

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

On vérifie ensuite si la base de données a bien été créée via phpMyAdmin. ➔



CRÉATION DES ENTITÉS

CRÉATION D'UNE ENTITÉ : USER

La première entité que j'ai décidé de créer est l'entité pour les utilisateurs du site : User. Elle me permettra par la suite de pouvoir générer un système de rôle, de se connecter et d'ajouter d'autres utilisateurs.

Sur Symfony, la construction des entités est très simple, elle se fait par le biais de ligne de commande dans le terminal à la racine du projet et bénéficie d'une trame chronologique nous demandant certains paramètres utiles à la création de l'entité et à ça futur utilisation, comme les données à intégrer ou encore le type de contenu.

Pour créer l'entité User j'ai utilisé la ligne de commande :

➔ `php bin/console make:user`

J'ai suivi les indications et entré les paramètres nécessaires à la création de l'entité. Une fois l'entité créée, il nous faut l'envoyer dans la base de données, cela se passe aussi en ligne de commande :

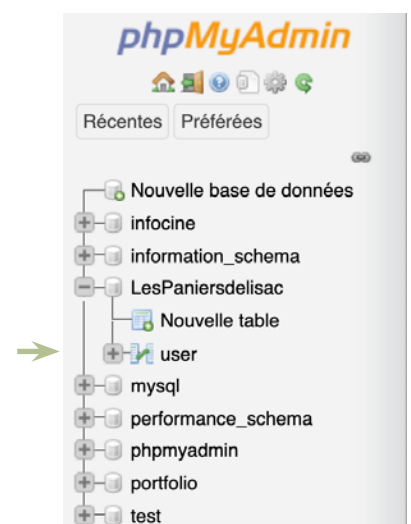
La première permet d'envoyer l'entité dans la base de données :

➔ `php bin/console make:migration`

La seconde permet de confirmer l'opération :

➔ `php bin/console doctrine:migrations:migrate`

Voilà, l'entité a été créée et envoyé dans la base de données



CONSTRUCTION DE L'AUTHENTIFICATION

Symfony nous permet de créer automatiquement un système d'authentification grâce à une ligne de commande et des paramètres à mettre en place suivant cette ligne de commande :

```
→ php bin/console make:auth
```

Cette ligne nous permet entre autre de créer un formulaire de connexion avec un mail ainsi qu'un mot de passe. Le mot de passe est automatiquement haché par symfony .

Une activation du compte par e-mail, est également géré par symfony, et j'ai pu tester cette fonctionnalité grâce à **mailhog** (serveur d'e-mail local).

J'ai par la suite rajouté une double vérification de mot de passe dans le formulaire d'enregistrement sur le site.

Ensuite j'ai ajouté quatre autres paramètres dans l'entité, le pseudo, le role, la date de création et la date de modification avec la commande :

```
→ php bin/console make:entity
```

Pour les deux dates, la date de création est automatiquement créé à la création du compte, et la date de modification est également modifié automatiquement lors de la modification du compte.

Ce qui nous donne dans le code un formulaire comme celui-ci :

```
public function buildForm(FormBuilderInterface $builder, array $options): void
{
    $builder
        ->add('email')
        ->add('pseudo')
        ->add('agreeTerms', CheckboxType::class, [
            'mapped' => false,
            'constraints' => [
                new IsTrue([
                    'message' => 'Vous devez accepter les conditions d\'utilisation.',
                ]),
            ],
        ])
        ->add('plainPassword', RepeatedType::class, array(
            'type' => PasswordType::class,
            'invalid_message' => 'Les mots de passe doivent être identiques',
            'first_options' => ['label' => 'Mot de passe'],
            'second_options' => ['label' => 'Répétez le mot de passe'],

            // instead of being set onto the object directly,
```

```

// this is read and encoded in the controller
'mapped' => false,
'attr' => ['autocomplete' => 'new-password'],
'constraints' => [
    new NotBlank([
        'message' => «Merci d'entrer un mot de passe»,
    ]),
    new Length([
        'min' => 6,
        'minMessage' => «Votre mot de passe n'a pas [{ limit }] caractères
minimum»,
        // max length allowed by Symfony for security reasons
        'max' => 4096,
    ]),
],
));
}

```

UTILISATEUR ET SÉCURITÉ

Afin de sécuriser mon projet j'ai ajouté trois rôles dans la partie src/Entity/User.php :

```

class User implements UserInterface, PasswordAuthenticatedUserInterface
{
    public const ROLE_USER = 'ROLE_USER';
    public const ROLE_EDITOR = 'ROLE_EDITOR';
    public const ROLE_ADMIN = 'ROLE_ADMIN';
}

```

Comme je l'expliquais plus haut mon rôle « USER » est pour le moment un rôle sans beaucoup de fonctionnalité, il permet de s'inscrire, et de faire une demande pour devenir «EDITOR», les adhérents auront un rôle «EDITOR», leur permettant de gérer leurs fiches et le rôle « ADMIN » est en réalité l'équivalent d'un « Super Administrateur » c'est à dire le développeur.

Afin d'affecter des autorisations à chaque utilisateur nous utiliserons l'annotation de `methods@IsGranted` à mettre en place dans les controllers désiré ainsi que dans les templates afin d'afficher ou non certaines informations ou de limiter certains accès en fonction du rôle :

```

/**
 * @Route(«/dashboard», name=»dashboard»)
 * @isGranted(«ROLE_USER»)
 */
public function index(UserRepository $userRepository, NewsRepository
$newsRepository, ContactsRepository $contactsRepository, ProducersRepository
$producersRepository, LabelsRepository $labelsRepository,
TypeDistributionsRepository $typeDistributionsRepository, ProductsRepository
$productsRepository, TypeProductsRepository $typeProductsRepository,
TradersRepository $tradersRepository, TypeTradersRepository
$typeTradersRepository): Response
{

```

Dans cet exemple l'annotation `isGranted` signifie que seul les utilisateurs possédant le rôle `USER` et au dessus peuvent accéder à la page `dashboard`.

Ensuite voici un exemple de l'utilisation de l'annotation `Twig is_granted` dans un fichier `html.twig` qui nous indique que si nous voulons afficher des éléments du menu alors il faut être connecté et avoir le rôle « `ADMIN` » afin de pouvoir les voir et les utiliser :

```

{% if is_granted(«ROLE_ADMIN») %}
    <a class=»nav-link navbarAd» href=»{{ path('contacts_index')
}}»><i class=»bi bi-envelope»></i> Messages</a>
    <a class=»nav-link navbarAd» href=»{{ path('news_index')
}}»><i class=»bi bi-newspaper»></i> Actualités</a>
    <a class=»nav-link navbarAd» href=»{{ path('traders_index')
}}»><i class=»bi bi-shop»></i> Commerces</a>
    <a class=»nav-link navbarAd» href=»{{ path('type_traders_
index') }}»><i class=»bi bi-shop-window»></i> Type de commerces</a>
{% endif %}

```

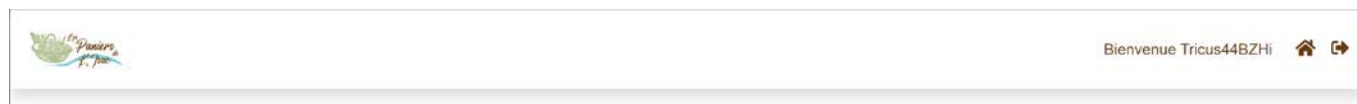
Pour répondre au besoins du site, j'ai mis en place un système de gestion de contenu. L'adhérent ou l'administrateur aura la possibilité d'importer des photos ou logo, de mettre du texte ou de le modifier.

INTERFACE D'ADMINISTRATION

La partie Administration du site est relativement simple.

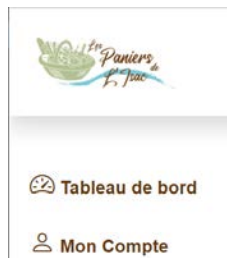
J'ai créé une barre de navigation en deux partie, qui comprend le haut, mais surtout sur la gauche de la page en position fixe.

La partie haute comprend le pseudo du USER, un bouton home qui redirige vers la page d'accueil de la partie « Publique », et un bouton de déconnexion.

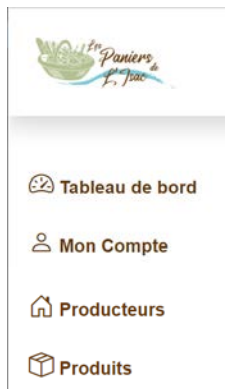


La partie gauche de la barre de navigation comprend tout les liens de navigation dans la partie Administration du site, elle est beaucoup plus complète en version Admin, que pour la version Editor ou même user .

USER



EDITOR





ADMIN



GESTION DES RUBRIQUES

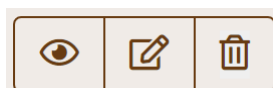
Pour en revenir quelques instants sur la barre de navigation un lien ce présente comme ceci :

Si je clique sur un des liens (Pour une grande partie des liens), j'arrive sur une page d'index ou de modification de la rubrique sélectionnée. Dans mon exemple, je vais prendre les producteurs, et le lien nous amène dans la page d'index des producteurs.

Producteurs					
Photo ou logo	Dénomination	Email	Téléphone mobile	Commune	Actions
	Les Bêles de Ronces	pasdemailencore@test.fr	06 24 20 22 69	Campbon	  
	Earl Vallée De La Lune			Avessac	  
	Gaec Des Châtaigniers		02 99 72 11 76	Avessac	  
	Earl de Chassenon	dominique.briand@lafermedechassenon.fr	06 08 23 24 01	Blain	  
	Esat Les Ateliers Blinois	esatblain@adapei44.asso.fr		Blain	  
	Le jardin du Perche	maryse.oheix@orange.fr	06 33 96 91 45	Blain	  
	Nicolas Moreau	nico44.moreau@orange.fr	06 59 00 95 16	Blain	  
	L'escargote de charlotte		06 18 35 27 09	Blain	  
	Bulle de pain	bulledepain@free.fr		Bouvron	  
	Earl du Friche Blanc		09 66 42 25 21	Bouvron	  
	La ferme des Rochettes	nicole.sellier@free.fr		Bouvron	  
	La Fournée du Pas Gaec Saint Joseph			Bouvron	  
	La ferme des cochon du sillon		06 07 01 10 90	Campbon	  
	Les Paysans Fromagers Nantais-Ferme De Riglanne			Campbon	  
	Brasserie l'hop pression	polofred@icloud.com	06 50 40 13 00	Guenrouet	  
					

Cette dernière est représenté sous la forme d'un tableau avec des boutons pour interagir avec, pour l'administrateur, il peut voir tous les producteurs, tandis que pour l'Éditeur, il peut voir que les producteur qu'il a créé.

Les boutons pour interagir donne la possibilité (dans cette exemple) de Voir la page lié à l'entité (représenté par l'icône « œil »), de l'éditer (icône « crayon »), ou bien de la supprimer (icône « poubelle »).



AJOUT D'UNE RUBRIQUE

Afin d'ajouter une entité dans la rubrique, en l'occurrence un producteur pour l'exemple (globalement le concept est le même pour les autres parties de la gestion de contenu), j'ai mis en place un boutons « plus » en bas à gauche du tableau. En cliquant sur le bouton « + » l'utilisateur est dirigé sur la page d'ajout de contenu, en



l'occurrence « Nouveau Producteur » qui se présente comme suit :

🏠 Nouveau Producteur

Dénomination	Nom du contact
Email	Site internet
Facebook	Instagram
	Twitter
Téléphone fixe	Téléphone mobile
Adresse	
Code postal	Commune
Latitude Deg	Longitude Deg

Description

Labels

☐ Agriculture biologique
☐ BIO européen

Type de distribution

☐ Vente sur place
☐ Vente sur commande
☐ Vente de panier en AMAP
☐ Vente sur marché
☐ pfff

Planning/horaires

Photo ou logo
Choisir un fichier
Aucun fichier choisi

Ici il y a beaucoup de champs à remplir afin d'avoir une fiche détaillée du producteur le premier étant la dénomination du producteur suivi du nom du contact. Après nous avons tout ce qui est lié à internet en commençant par l'e-mail, l'adresse du site internet, l'adresse de facebook, instagram et twitter. Ensuite nous avons les coordonnées téléphoniques, fixe et mobile suivies de l'adresse postale. Il y a aussi deux champs permettant de rentrer les coordonnées GPS du producteur, afin qu'il soit géolocalisé sur une carte.

Il y a le champ description.

On peut aussi sélectionner un label et un type de distribution.

Le champ Planning/horaires.

Il y a aussi la possibilité d'ajouter une photo ou bien le logo du producteur.

Pour les deux champs celui de la description et celui du planning horaires, pour c'est deux champs, j'ai choisi un système de traitement de texte afin d'avoir plus de choix de personnalisation du contenu. J'ai donc utilisé CK Editor, qui est un éditeur de texte open-source, que j'ai installé via composer sous forme de bundle dans symfony :

```
➔ composer require friendsofsymfony/ckeditor-bundle
```

Ensuite j'ai vérifié que le bundle était bien affiché à la fin de la liste des bundles dans votre fichier config/bundles.php

```
FOS\CKEditorBundle\FOSCKEditorBundle::class => ['all' => true],
```

La deuxième étape est d'installer ckeditor en entrant la commande Symfony suivante :

```
➔ php bin/console ckeditor:install
```

Puis on vérifie que les assets soient bien installés avec la ligne de commande suivantes :

```
➔ php bin/console assets:install public
```

La troisième étape va consister à configurer ckeditor afin de paramétrer les options auxquels les utilisateurs auront accès, donc dans le fichier config/packages/fos_ckeditor.yaml j'ai ajouté le code ci-dessous :

```
fos_ck_editor:
  configs:
    my_config:
      toolbar:
- { name: «styles», items: ['Bold', 'Italic', 'Underline', 'Strike', 'Blockquote', '-', 'Link',
  '-', 'RemoveFormat', '-', 'NumberedList', 'BulletedList', '-', 'Outdent', 'Indent', '-',
  '-', 'JustifyLeft', 'JustifyCenter', 'JustifyRight', 'JustifyBlock', '-', 'Image', 'Table', '-',
  'Styles', 'Format', 'Font', 'FontSize', '-', 'TextColor', 'BGColor', 'Source']} }
```

J'ai supprimé l'ajout d'image 'Image', car je ne voulais pas de cette option.

Après il faut intégrer Ckeditor dans mon formulaire de création de producteur. J'ouvre donc mon fichier Form/ProducersType.php et j'ajoute la classe

CKEditorType à ma propriété descriptionProduct.

Attention de ne pas oublier d'importer les classes afin d'ajouter le « use » nécessaire à l'appel de la classe CKEditorType.

```
use FOS\CKEditorBundle\Form\Type\CKEditorType;

class ProducersType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options): void
    {
        $builder
            ->add('name')
            ->add('description', CKEditorType::class)
    }
}
```

Et pour terminer il faut afficher un contenu généré par un ckeditor ce n'est pas compliqué, il suffit d'afficher la donnée comme à notre habitude et d'ajouter la propriété twig « raw » qui permet d'interpréter le langage html.

```
<tr>
    <th>Description</th>
    <td>{{ producer.description|raw }}</td>
</tr>
```

ÉDITION / SUPPRESSION D'UNE RUBRIQUE

Pour éditer ou supprimer une rubrique ou en l'occurrence un producteur, il nous suffit de retourner sur la page d'index des équipes et de cliquer sur le bouton désiré. Il y a deux façon pour supprimer un producteur, soit directement sur la page d'index, soit sur la page d'édition. Je précise que la page d'édition est en tout point similaire à la page d'ajout d'une rubrique, il y a simplement un lien en plus pour supprimer dans la page d'édition.



Mise en place du Framework CSS

Symfony ne possède par nativement Bootstrap mais est conçue pour fonctionner très efficacement avec, la mise en place est plutôt simple.
Pour utiliser Bootstrap avec le CDN il me faut poster dans mon <head> cette balises :

```
<link rel=»stylesheet» href=»https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css» integrity=»sha384-F3w7mX95PdgyTmZZMECAngseQB83DfGTowiOimWaeVhAn4FJkqJByhZMI3AhiU» crossorigin=»anonymous»>
```

Et en bas de page je dois appeler le plugins JavaScript :

```
<script src=»https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js» integrity=»sha384-MrcW6ZMFYlzcLA8NI+NtUVFOsA7MsXsPIUyJoMp4YLEuNSfAP+JcXn/tWtlaxVXM» crossorigin=»anonymous»></script>
```

Nous allons après dans le fichier de configuration de twig qui se trouve dans config/package/twig.yaml afin d'ajouter cette ligne de code :

```
form_themes: ['bootstrap_5_layout.html.twig']
```

Mise en place des icons

j'ai également utilisé les icones proposé par Bootstrap et pour cela j'ai du ajouter le CDN dans mon <head>

```
<link rel=»stylesheet» href=»https://cdn.jsdelivr.net/npm/bootstrap-icons@1.6.1/font/bootstrap-icons.css»>
```

J'ai aussi utilisé les icones de Font Awesome et pour cela j'ai également ajouté le code javascript dans mon <head>

```
<script src=»https://kit.fontawesome.com/cb109f8570.js»  
crossorigin=»anonymous»></script>
```

Mise en place des typos

J'ai utilisé les fonts de Google Fonts et pour cela j'ai du ajouter dans mon <head> les codes:

```
<link href="https://fonts.googleapis.com/css2?family=Spartan:wght@100;200;300;400;500;600;700;800;900&family=Roboto+Slab:wght@100;200;300;400;500;600;700;800;900&display=swap" rel="stylesheet">
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
```

Mise en place du CSS

J'ai décidé pour le CSS t'utiliser SASS, en partie pour la gestion plus simple du code et de ses imbrications, son système @extend, et son système @import sont un plus, mais je ne m'en suis pas servis.

Ma démarche est assez simple je crée un nouveau fichier .scss pour chaque grande partie de mon site et j'ai donc un fichier style.scss pour la partie «Annuaire», et un fichier styleAD.scss pour la partie «Administration» du site.

Sur Visual studio code j'ai installé une extension qui me permet de compiler mon fichier .scss en un fichier .css.

```
// NAV
// =====
nav {
  border-bottom: $brown solid 1px;

  li {
    display: inline-block;
    min-width: 50px;

    a {
      text-decoration: none;
      line-height: 30px;
      padding: 0 10px;
      text-align: center;
      font-size: 1rem;
      font-weight: 600;
      color: $brown !important;
    }
  }
}
```

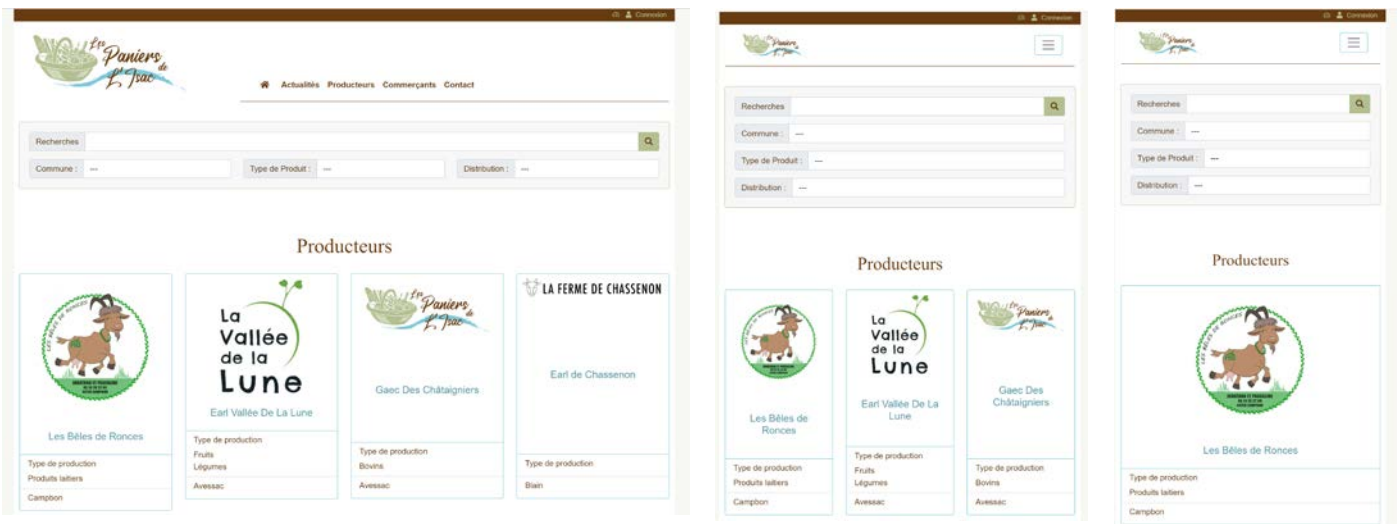
Je trouve pour ma part que le système d'imbrication de Sass particulièrement ingénieux et pratique, à présent, je l'utilise en permanence.

Réaliser une interface web statique et dynamique

Pour que l'expérience utilisateur soit la plus intuitive et interactive possible, il me fallait rendre mon interface responsive adaptable sur plusieurs tailles d'écrans, types d'appareils.

Je me suis servi, tout d'abord, des différentes class Bootstrap «**sm - md - lg - xl**» afin que les éléments s'adapte au mieux à l'écran en fonction du ratio et de l'orientation. Leur utilisation est plutôt simple, on part sur une grille de 12 colonnes qu'il nous faut remplir avec la class «**col**» et des lignes avec la class «**row**», le tout peut être contenu dans une div contenant la class «**container**».

Exemple :



Un autre exemple intéressant de l'utilisation de Bootstrap est la navbar, cette dernière est très facile à mettre en place mais il est plus compliqué de la personnaliser (idem pour le carrousel). Le gros avantage de la navbar Bootstrap est qu'elle est responsive et qu'elle affiche donc naturellement un menu burger sur les petits écrans :



Pour l'affichage mobile grace à Bootstrap je n'est presque pas eut à utiliser les media queries, juste pour l'affichage de l'image en background dans mon menu

```
@media screen and (max-width: 991px) {#slider {
  display: none;}
.backImg{
  background-image: linear-gradient(to left bottom, rgba(255, 255, 255, 0),
  rgba(255,255,255,0.7), rgba(255,255,255,0.9), rgba(255,255,255,1),
  rgba(255,255,255,1)),
  url('../images/PhotoHeader.jpg');
  max-height: 100vh;
  background-position: center;
  background-size: cover;
}
```

Afin de rajouter un peu de dynamisme au site, j'ai utilisé la librairie javascript Leaflet couplé à Open Street Map qui est un projet de carte Open Source et gratuite. J'ai configuré le fichier et lui ai rajouté un marqueur afin de pouvoir identifier la localiser des producteurs ou des commerçants sur leur fiche

Contact

👤 : Jonathan Thibaud

✉ : pasdemailencore@test.fr

🌐 : <https://www.lesbelesderonces.fr/>

📘 : <https://fr-fr.facebook.com/pages/category/Farm/Les-B%C3%AAles-de-Ronces-1524579337764172/>

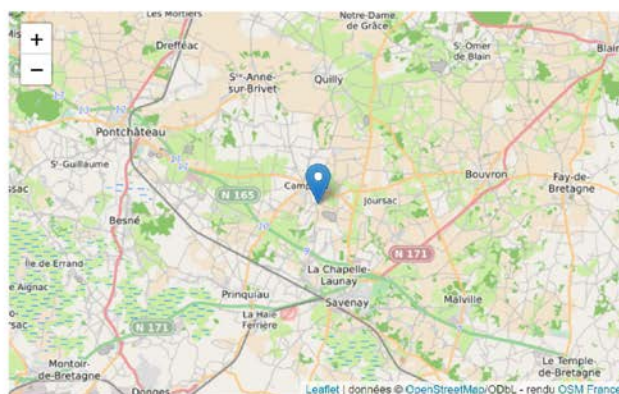
📷 :

📺 :

☎ :

📞 : 06 24 20 22 69

📍 : 6 la lande de la gicquelais
44750
Campbon



Pour situer la carte dans le cadre il me faut utiliser ce script :

```
<!-- Fichiers Javascript Open street Map-->
```

```

    <script src=>https://unpkg.com/leaflet@1.7.1/dist/leaflet.js>
    integrity=>sha512-XQoYMqMTK8LvdxXYG3nZ448hOEQiglfqkJsINOQV44cWnUr
    Bc8PkAOcXy20wOvlaXaVUearIOBhiXZ5V3ynxwA=> crossorigin=>>>
    </script>
    <script type=>text/javascript>

```

```

// On initialise la latitude et la longitude (centre de la carte)
var lat = [{ producer.longitudeDeg }];
var lon = [{ producer.latitudeDeg }];
var macarte = null;

```

```

// Fonction d'initialisation de la carte
function initMap() { // Créer l'objet «macarte» et l'insérer dans l'élément HTML qui a
l'ID «map»
macarte = L.map('map').setView([
lat, lon
], 11);

```

```
// Leaflet ne récupère pas les cartes (tiles) sur un serveur par défaut. Nous devons
// lui préciser où nous souhaitons les récupérer. Ici, openstreetmap.fr
L.tileLayer('https://{s}.tile.openstreetmap.fr/osmfr/{z}/{x}/{y}.png', { // Il est
// toujours bien de laisser le lien vers la source des données
attribution: 'données © <a href=»//osm.org/copyright»>OpenStreetMap</a>/
ODbL - rendu <a href=»//openstreetmap.fr»>OSM France</a>',
minZoom: 1,
maxZoom: 20
}).addTo(macarte);

// Nous ajoutons un marqueur
var marker = L.marker([lat, lon]).addTo(macarte);

}
window.onload = function () { // Fonction d'initialisation qui s'exécute lorsque le
DOM est chargé
initMap();
};
</script>
```

On déclare les variables lat et lon pour l'emplacement de la carte, pour ma part la longitude et la latitude sont automatiquement récupéré dans ma base de donnée.

Recherche FullText

Je voulais pour mon annuaire, faire un champs de recherche multi-mots sur mes rubriques, j'ai cherché sur internet comment le réaliser, et j'ai trouvé un tutoriel de Nouvelle techno et voici comment j'ai procédé.

Étape 1

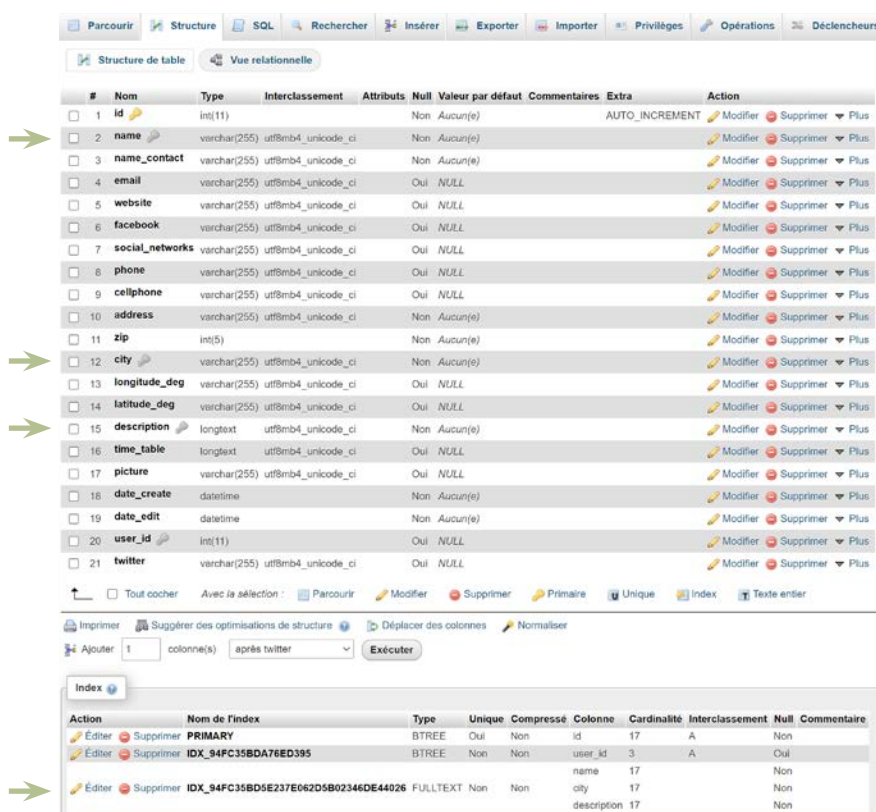
Tout d'abords j'ai été sur mon dossier Entity et sur mon fichier Producers.php, j'ai ajouter ce code à mon annotation permettant de faire un index plein champs de ma table «producers» et je lui est précisé quel sont les colonnes que je veux indexer (name,city et description) et le type d'indexation (flags=fulltext).

```
/**
 * @ORM\Entity(repositoryClass=ProducersRepository::class)
 * @ORM\Table(name='producers', indexes={@ORM\
Index(columns={«name», «city», «description», flags={«fulltext»}}))
 */
```

Ensuite, je vais dans ma console pour créer un index fulltext sur ma base de données

➔ php bin/console make:migration

➔ php bin/console doctrine:migrations:migrate



#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
2	name	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier Supprimer Plus
3	name_contact	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier Supprimer Plus
4	email	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
5	website	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
6	facebook	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
7	social_networks	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
8	phone	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
9	cellphone	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
10	address	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier Supprimer Plus
11	zip	int(5)			Non	Aucun(e)			Modifier Supprimer Plus
12	city	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier Supprimer Plus
13	longitude_deg	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
14	latitude_deg	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
15	description	longtext	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier Supprimer Plus
16	time_table	longtext	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
17	picture	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus
18	date_create	datetime			Non	Aucun(e)			Modifier Supprimer Plus
19	date_edit	datetime			Non	Aucun(e)			Modifier Supprimer Plus
20	user_id	int(11)			Oui	NULL			Modifier Supprimer Plus
21	twitter	varchar(255)	utf8mb4_unicode_ci		Oui	NULL			Modifier Supprimer Plus

Action	Nom de l'index	Type	Unique	Compressé	Colonne	Cardinalité	Interclassement	Null	Commentaire
Modifier Supprimer	PRIMARY	BTREE	Oui	Non	id	17	A	Non	
Modifier Supprimer	IDX_94FC35BDA76ED395	BTREE	Non	Non	user_id	3	A	Oui	
Modifier Supprimer	IDX_94FC35BD5E237E062D5B02346DE44026	FULLTEXT	Non	Non	name city description	17 17 17		Non Non Non	

Étape 2

Il va falloir créer une requête MatchAgainst

Dans le fichier «src», j'ai créé un nouveau dossier «Extensions» puis un dossier «Doctrine» et enfin dans ce dossier j'ai créé un fichier «MatchAgainst.php» dont voici le code

```
<?php
namespace App\Extensions\Doctrine;

use Doctrine\ORM\Query\AST\Functions\FunctionNode;
use Doctrine\ORM\Query\Lexer;
use Doctrine\ORM\Query\Parser;
use Doctrine\ORM\Query\SqlWalker;

class MatchAgainst extends FunctionNode
{
    /** @var array list of \Doctrine\ORM\Query\AST\PathExpression */
    protected $pathExp = null;
    /** @var string */
    protected $against = null;
    /** @var bool */
    protected $booleanMode = false;
    /** @var bool */
    protected $queryExpansion = false;

    public function parse(Parser $parser)
    {
        // match
        $parser->match(Lexer::T_IDENTIFIER);
        $parser->match(Lexer::T_OPEN_PARENTHESIS);
        // first Path Expression is mandatory
        $this->pathExp = [];
        $this->pathExp[] = $parser->StateFieldPathExpression();
        // Subsequent Path Expressions are optional
        $lexer = $parser->getLexer();
        while ($lexer->isNextToken(Lexer::T_COMMA)) {
            $parser->match(Lexer::T_COMMA);
            $this->pathExp[] = $parser->StateFieldPathExpression();
        }
        $parser->match(Lexer::T_CLOSE_PARENTHESIS);
        // against
        if (strtolower($lexer->lookahead['value']) !== 'against') {
            $parser->syntaxError('against');
        }
    }
}
```

```

$parser->match(Lexer::T_IDENTIFIER);
$parser->match(Lexer::T_OPEN_PARENTHESIS);
$this->against = $parser->StringPrimary();
if (strtolower($lexer->lookahead['value']) === 'boolean') {
    $parser->match(Lexer::T_IDENTIFIER);
    $this->booleanMode = true;
}
if (strtolower($lexer->lookahead['value']) === 'expand') {
    $parser->match(Lexer::T_IDENTIFIER);
    $this->queryExpansion = true;
}
$parser->match(Lexer::T_CLOSE_PARENTHESIS);
}

public function getSql(SqlWalker $walker)
{
    $fields = [];
    foreach ($this->pathExp as $pathExp) {
        $fields[] = $pathExp->dispatch($walker);
    }
    $against = $walker->walkStringPrimary($this->against)
        . ($this->booleanMode ? ' IN BOOLEAN MODE' : '')
        . ($this->queryExpansion ? ' WITH QUERY EXPANSION' : '');
    return sprintf('MATCH (%s) AGAINST (%s)', implode(', ',
$fields), $against);
}
}

```

Étape 3

Il va falloir déclarer cette requête dans le fichier doctrine qui se situe dans config/packages/ doctrine.yaml, et dans l'ORM ajouter une partie «dql»

```

orm:
    auto_generate_proxy_classes: true
    naming_strategy: doctrine.orm.naming_strategy.underscore_number_aware
    auto_mapping: true
    mappings:
        App:
            is_bundle: false
            type: annotation
            dir: '%kernel.project_dir%/src/Entity'
            prefix: 'App\Entity'
            alias: App
    dql:
        string_functions:
            MATCH_AGAINST: App\Extensions\Doctrine\MatchAgainst

```

Étape 4

Je vais devoir faire un formulaire de recherche, sur mon terminal je tape :

```
➔ php bin/console make:form
```

Attention a ne surtout pas l'appeler «SearchType», Moi je l'ai appelé «SearchAllType.php» et préciser qu'il n'intervient sur aucune Entité.

Ensuite j'ai modifié le builder, j'ai ajouté le champ mots, et j'utilise le champs de formulaire «SearchType»

```
$builder
    ->add('mots', SearchType::class, [
        'label' => false,
    ])
;
```

Sans oublier de déclarer la class

```
use Symfony\Component\Form\Extension\Core\Type\SearchType;
```

Étape 5

Pour afficher ce formulaire, il va falloir modifier le controller, ici celui de ProducersController.php

```
/**
 * @Route(«/producersIndex», name=>producersAFF)
 */
public function producersIndex(
    Request $request,
    TypeDistributionsRepository $typeDistributionsRepository,
    ProducersRepository $producersRepository,
    TypeProductsRepository $typeProductsRepository,
    ProductsRepository $productsRepository,
    TradersRepository $tradersRepository): Response
{
    $producers = $producersRepository->findAll();

    $form = $this->createForm(SearchAllType::class);

    $search = $form->handleRequest($request);

    if($form->isSubmitted() && $form->isValid()){

        // on recherche les producteur correspondant au mots
        clés
        $producers = $producersRepository->search($search-
        >get('mots')->getData());
    }
}
```

Attention, penser à supprimer la methods{«GET»} dans la Route

Penser à ajouter les dépendances

variable producers à mettre avant

Affichage du formulaire

Gestion du formulaire et utilisation de la méthode search

```

return $this->render('producers/producersIndex.html.twig', [
    'producers' => $producers,
    'products' => $productsRepository->findAll(),
    'typeProducts' => $typeProductsRepository->findAll(),
    'typeDistributions' => $typeDistributionsRepository->findAll(),
    'trader' => $tradersRepository->findAll(),
    'form' => $form->createView()
]);
}

```

Envoyé à la vue

Ne pas oublier de déclarer les class

```

use App\Form\SearchAllType;
use Symfony\Component\HttpFoundation\Request;

```

Dans mon formulaire, templates/recherch/_form.html.twig

```

{{ form_start(form) }}
<div class=»row»>
    <div class=»col-12»>
        <div class=»input-group»>
            <span class=»input-group-text» id=»basic-addon1»>Recherches</span>
            {{ form_widget(form.mots) }}

            <button type=»submit» class=»btn mb-0 mt-0»>
                <i class=»fas fa-search»></i>
            </button>
        </div>
    </div>
</div>
{{ form_end(form) }}

```

Dans ma vue, templates/producers/producersIndex.html.twig
j'ai fais un «include» de mon formulaire

```

{{ include('recherch/_form.html.twig') }}

```

Étape 5

Dans le repository je vais créer une Méthode permettant de faire une recherche sur producers (src/Repository/ProducersRepository.php).

```
/**
 * Recherche les producteurs en fonction du formulaire
 * @return void
 */
public function search($mots){
    $query = $this->createQueryBuilder('p');
    if($mots != null){
        $query->where('MATCH_AGAINST(p.name, p.city, p.description) AGAINST
(:mots boolean)>0')
        ->setParameter('mots', $mots);
    }
    return $query->getQuery()->getResult();
}
```

SELECT2

Pour un formulaire, je voulais utiliser un Select qui me permettait d'écrire les premières lettres d'un producteur et qu'il me l'affiche, et pour un autre Select, je voulais pouvoir choisir plusieurs commerçants, et j'ai trouvé le SELECT2 en javascript qui propose Single select boxes et le Multi-select boxes, voici comment j'ai procédé.

J'ai copié ce code dans mon head:

```
<link href=>https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/css/select2.min.css< rel=>stylesheet />
```

et celui ci en bas de page :

```
<script src=>https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/js/select2.min.js<></script>
```

Ensuite j'ai copié ce code Javascript dans mon fichier app.js :

```
$(document).ready(function () {
    // select2 pour les commerçants new product
    $('.select-traders').select2();
    // select2 pour les producteurs new product
    $('.select-producers').select2();
});
```

Ensuite j'ai modifié le builder de mon FormType, sans oublié la Class :

```
->add('producers', EntityType::class, [
    'class' => Producers::class,
    'choice_label' => 'name',
    'query_builder' => function(EntityRepository $er){
        return $er->createQueryBuilder('c')
            ->orderBy('c.name', 'ASC');
    },
    'attr' => [
        'class' => 'select-producers',
    ],
    'mapped' => false,
])

->add('trader', EntityType::class, [
    'class' => Traders::class,
    'multiple' => true,
    'expanded' => false,
    'required' => false,
    'choice_label' => 'name',
    'query_builder' => function(EntityRepository $er){
        return $er->createQueryBuilder('c')
            ->orderBy('c.name', 'ASC');
    },
    'attr' => [
        'class' => 'select-traders',
    ]
])
```

J'ai utilisé QueryBuilder, qui est générateur de requêtes fournit avec Doctrine c'est une manière orientée objet d'écrire des requêtes. Il est recommandé de l'utiliser lorsque les requêtes sont construites dynamiquement (c'est-à-dire basées sur des conditions PHP), ici c'est pour classer la «name» par ordre alphabétique

Filtre de recherche

Filtre de recherche en javascript pour afficher ici les producteurs par commune

Ici, le javascript me permet de masquer mes cards, en remplaçant la class :

```
$(«#selectCity»).click(function () {  
    if ($(«#selectCity»).val() != null) {  
        $(«.cardlien»).removeClass(«display»);  
        var idSelect = $(«#selectCity»).val();  
  
        var id = «.» + idSelect;  
        $(id).toggleClass(«display»);  
    }  
});
```

et dans le fichier twig :

```
<div class=»col-lg-4»>  
    <div class=»input-group mt-3»>  
        <span class=»input-group-text» id=»basic-addon1»>Commune :</span>  
        <select class=»form-control» id=»selectCity»>  
            <option value=»0» disabled selected hidden>---</option>  
            {% set tableau = [] %}  
            {% for producer in producers %}  
                {% if producer.city not in tableau %}  
                    <option value=»{{producer.city}}»>  
                        {{ producer.city }}<br>  
                        {% set tableau = tableau|merge([producer.city])%}  
                    {% endif %}  
                {% endfor %}  
            </option>  
        </select>  
    </div>  
</div>
```

VEILLE SUR LA SÉCURITÉ

Je souhaitais avoir une sécurité optimal pour mon site web, ce qui m'a poussé à faire de la veille en sécurité, on m'a conseillé de prendre en compte les actualités sur le site web OWSAP qui répertorie toutes les failles de sécurité web ainsi que les recommandations pour s'en prémunir.

= => Liste des 10 types de risque liés à la sécurité d'application.

www.owasp.org/www-project-top-ten/

J'ai notamment prit le temps de lire la documentation Symfony, très bien expliqué et très complète = => www.symfony.com/doc/current/security.html

Il y a aussi un site très intéressant pour les francophones c'est le site asfy.fr (Association Francophone Des Utilisateurs de Symfony) qui nous donne des petites astuces afin de mieux sécuriser nos sites web et notamment d'autre conseil pour d'autres domaines.

Un autre site qui mérite une petite mention c'est wanadev pour son article très intéressant, sur 5 sources d'amélioration pour la sécurité de votre application Symfony = => <https://www.wanadev.fr/78-5-sources-d-amelioration-pour-la-securite-de-votre-application-symfony/>

TEST UNITAIRE DE SECURITÉ

Il me fallait à présent réaliser des tests un peu plus ciblé sur mon projet, j'ai donc effectué des tests unitaires à l'aide du Framework Open Source PHPUnit.

PHPUnit est un framework open source de tests unitaires dédié au langage de programmation PHP. Il va nous permettre de tester certaines fonctionnalités de notre site internet.

Pour l'installer il m'a fallut utiliser le gestionnaire de package Composer, via mon terminal à la racine de mon projet avec ces deux commandes :

```
➔ composer require —dev symfony/phpunit-bridge
```

```
➔ composer require —dev symfony/browser-kit symfony/css-selector
```

Je vais tester les formulaires, les formulaires sont souvent primordiaux dans le bon fonctionnement des sites. C'est pour cela qu'il est très important de veiller à ce que ceux-ci fonctionnent correctement et qu'ils ne présentent pas de failles de sécurité. Ils faut donc les tester.

Pour cela, dans le dossier « Test », je crée un dossier « Forms » dans lequel j'ai crée un fichier « nomEntitéFormTest ». en réalité le nom importe peu il faut qu'il soit logique en fonction du formulaire que nous voulons tester :

```
<?php
// tests/Form/ProductsFormTest.php
namespace App\Tests\Form;
use App\Entity\Producers;
use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;
class ProducersFormTest extends KernelTestCase{

    public function testNewCategory(){
        $producers=(new Producers())
        ->setName('77')
        ->setNameContact('77')
        ->setEmail('77')
        ->setWebsite('77')
        ->setFacebook('77')
        ->setSocialNetworks('77')
        ->setPhone('77')
        ->setCellphone('77')
        ->setAddress('77')
        ->setZip('77')
        ->setCity('77')
        ->setLongitudeDeg('77')
        ->setLatitudeDeg('77')
        ->setDescription('77')
        ->setTimeTable('77')
        ->setPicture('77');
        ->setTwitter('77');
```

```
self::bootKernel();
$error = self::$container->get('validator')->validate($producers);
$this->assertCount(0,$error);
}
```

Ensuite je lance ma commande :

➔ `php bin/phpunit`

et j'obtiens :

```
PS C:\wamp64\www\LesPaniersdelisac> php bin/phpunit
Could not open input file: bin/phpunit
PHPUnit 9.5.10 by Sebastian Bergmann and contributors.
```

```
Testing
.                                     1 / 1 (100%)
```

```
Time: 00:04.318, Memory: 54.00 MB
```

```
OK (1 test, 1 assertion)
```

Other deprecation notices (1)

1x: Since symfony/framework-bundle 5.2: Accessing the «validator» service directly from the container is deprecated, use dependency injection instead.
1x in ProducersFormTest::testNewCategory from App\Tests\Form

Tout s'est bien passée en l'occurrence j'ai fait exprès de mettre de bonne information mais j'ai testé auparavant avec des informations erroné et j'ai ensuite ajouté des contraintes de validation.

Les contraintes de validation

Pour pouvoir utiliser des contraintes de validation dans mon formulaires, il faut tout d'abord installer un nouveau package à votre projet Symfony grâce à Composer.

➔ `composer require symfony/validator doctrine/annotations`

Ensuite j'ai été regarder la documentation de Symfony afin de trouver les asserts correspondant à mes formulaires :

<https://symfony.com/doc/current/reference/constraints.html>

Ne pas oublier de rajouter la class à l'Entity

```
use Symfony\Component\Validator\Constraints as Assert;
```

Il me fallait ensuite une solution pour éviter un maximum de robots spam sur mon site, j'ai donc installé un outil nommé ReCAPTCHA. Il s'agit d'un script que l'on appelle sur toutes les pages du site afin de les protéger d'éventuelles attaques.



reCAPTCHA

Installer reCaptcha Google V3 sur son site internet

Qu'est-ce que reCaptcha ?

reCAPTCHA protège votre site Web contre la fraude et les abus. Ce service de sécurité utilise un moteur d'analyse des risques avancé et des défis adaptatifs pour empêcher les logiciels malveillants de se livrer à des activités abusives sur votre site Web.

Les utilisateurs légitimes pourront ainsi se connecter, effectuer des achats, afficher des pages ou créer des comptes tandis que les faux utilisateurs seront bloqués.

Etape 1 : Enregistrer votre site

Allez sur le site <https://www.google.com/recaptcha/about/> en étant connecté à son compte Gmail sur chrome puis allez dans la console d'administration (lien « Admin Console »).

Remplir le formulaire et cliquez sur « Envoyer ».

Etape 2 : Intégration du reCaptcha sur votre site

Copiez la clé du site que j'ai généré en remplissant le formulaire d'inscription.

Je copie le script dans le head de mon site.

```
<script src="https://www.google.com/recaptcha/api.js?render=CLE DU SITE">
</script>
```

Puis je copie ce bout de code dans mon fichier

```
<script>
  function onClick(e) {
    e.preventDefault();
    grecaptcha.ready(function() {
      grecaptcha.execute('CLE DU SITE', {action: 'submit'});
      then(function(token) {
        console.log('captacha');
      });
    });
  }
</script>
```

Pour ce qui est de la mise en production j'ai longuement hésité entre OVH et o2switch, mon choix s'est finalement porté sur O2switch, son offre me paraissait plus intéressante et plus complète qu'OVH.

O2switch est une entreprise 100% Française autant pour le serveur que pour le SAV.



Une phase importante du développement est le recettage, c'est ce qui permet d'engager une multitude de test que ce soit au niveau du sport (Mobile, Tablette) mais aussi faire des tests de sécurité, tout ceci permettant de vérifier si son site ne comporte pas de failles (j'en parle un peu plus dans la partie sécurité).

Pour cette mise en production il me fallait un logiciel qui me permette de me connecter au serveur FTP de l'hébergement o2switch, pour ce faire j'ai utilisé le logiciel WinSCP qui est un logiciel client FTP et SFTP pour la plateforme Windows.



Sur le mac j'ai utilisé FileZilla. Le client FileZilla prend non seulement en charge FTP, mais également FTP sur TLS (FTPS) et SFTP. Il s'agit d'un logiciel open source distribué gratuitement.

Je m'en suis servi pour envoyer l'intégralité de mon projet Symfony, dans le dossier de mon sous-domaine lespaniersdelisac.erwangentric.fr, mis à part le dossier var et vendor que je trouvais trop volumineux et que je pouvais rétablir via la ligne de commande « composer update » via le terminal (disponible sur l'interface Cpanel) mis à disposition par o2switch une fois mes fichiers dans le serveur.

Il me fallait ensuite créer un fichier .htaccess et le placer dans le dossier « public » depuis la racine de mon dossier Symfony. Le fichier .htaccess est un fichier de configuration (Apache) permettant de définir des règles bien spécifiques dans un répertoire. Ce type de fichier peut être utilisé pour réaliser des redirections ou protéger un répertoire par un mot de passe.

Il me fallait également un fichier .env à la racine de mon projet afin de configurer l'accès à la base de donnée.

Afin d'analyser le trafic sur le site, j'ai installé le plugin Google Analytics pour avoir un récapitulatif des taux d'affluence.

Intégrer Google Analytics dans son projet web

QU'EST-CE QUE GOOGLE ANALYTICS

Google Analytics est un service gratuit d'analyse d'audience d'un site web ou d'applications utilisé par plus de 10 millions de sites, soit plus de 80% du marché mondial. (wikipédia).

LES PRÉREQUIS

Comme pour tous les services Google, il est absolument nécessaire de détenir un compte Gmail pour pouvoir installer puis utiliser Google Analytics.

Deuxième chose à savoir avant de commencer à vouloir intégrer Google Analytics à votre projet, il faut que celui-ci soit mis en ligne. Cela paraît logique mais il est quand même bon de le rappeler.

CONFIGURER SON COMPTE GOOGLE ANALYTICS

Après avoir vérifié de bien respecter les prérequis, je me connecte à l'adresse suivante :

<https://analytics.google.com/analytics/web/>

- 1- Je choisis les paramètres de partage de données de mon compte Analytics
- 2- Je Configure mes propriétés
- 3- Je renseigne les informations de ma petite entreprise
- 4- Je sélectionne le type de projet
- 5- Je configure mon flux de données
- 6- Et enfin, j'intègre le bout de code fourni par Google Analytics dans le Head de mon projet, juste avant la balise body.

```
<!-- Global site tag (gtag.js) Google Analytics -->
```

```
<script async src="https://www.googletagmanager.com/gtag/js?id=Clé google analytic"></script>
```

Google récupère les informations du site grâce à ce « tag ». Et voici mon tableau de bord pour le site Les Paniers de l'Isac:

Il faut savoir avant de parler de la mise en production, que j'ai utilisé mon nom de domaine avec un sous-domaine pour mettre mon site en ligne, et cela simplement pour réaliser des tests.

Afin d'avoir un bon référencement sur le site, j'ai respecté les standards du W3C et de Google. Tout d'abord, j'ai effectué des tests sur le site <https://validator.w3.org> afin d'examiner mon code html et de corriger les éventuels erreurs.

J'ai aussi réalisé un test via Lighthouse qui permet notamment de mesurer les performances du site, l'accessibilité, les bonnes pratiques et le SEO, c'est ce dernier qui va nous intéresser ici.

(Nous reviendrons sur le reste dans la partie « Performance »)



Le test SEO n'était que à 92% au début, mais grâce aux recommandations est au système d'audit de Lighthouse j'ai pu augmenter mon score de SEO.

J'ai commencé par mettre en place une méta description dans le Header.

J'ai fait attention à l'utilisation des balises « h » pour les titres, un seul h1 et un ordre logique pour les suivantes.

J'ai ajouté aussi des descriptions dans chaque image pour qu'elles soient indexées par le moteur de recherche par le biais d'une balise « alt ».

J'ai mis en place la meta viewport pour l'accessibilité au mobile.

J'ai installé une balise <title> dans chaque page du site.

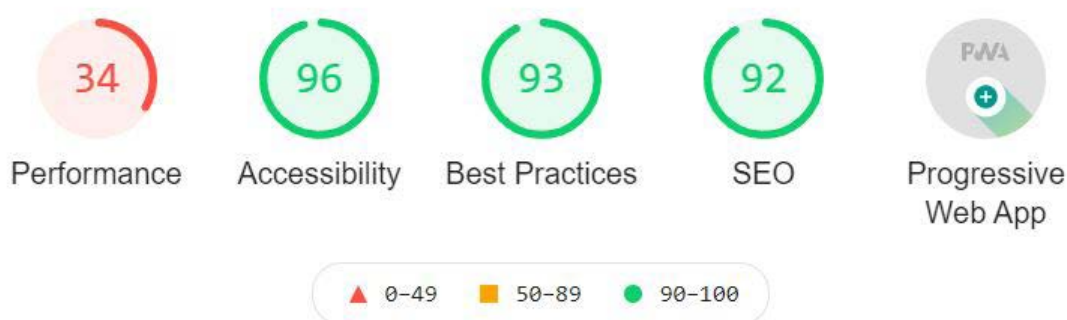
Tout les liens ont des textes descriptifs et sont « crawlable ».

J'ai précisé la langue du site via la balise <html lang="fr">.

J'ai ajouté un URL canonique, plutôt important pour le référencement.

Et j'ai ajouté une favicon pour la plupart des navigateurs et une favicon pour apple.





Pour vérifier la rapidité d'affichage, l'accessibilité, les bonnes pratiques au niveau du code ainsi que le référencement, j'ai utilisé le module Lighthouse cité un peu plus haut, pour noter une évolution vis à vis de la performance, j'ai réalisé un premier test de performance :

Les performances n'étant clairement pas convaincante, en tout cas pour la version mobile, j'ai décidé de suivre les recommandations de Google afin d'augmenter mon score de performance ainsi que de l'accessibilité.

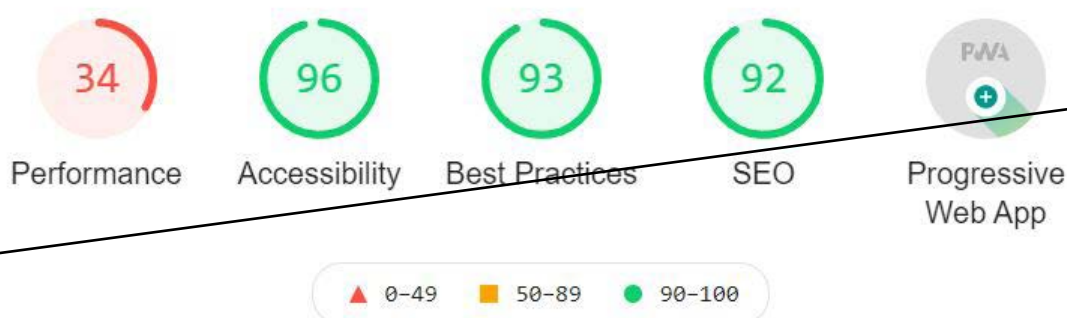
Pour commencer j'ai redimensionner les images puis je les ai compresser avec photoshop.

Il me fallait ensuite régler le problème des images « hors écran » ainsi que des iframes qui prenaient beaucoup de ressources lors du chargement de la page. J'ai découvert qu'il était possible de différer leurs chargements mais j'ai du faire quelques recherches pour ça que je décrirais dans la partie de recherche en Anglais.

J'ai découvert qu'il était possible d'ajouter un simple attribut dans les balise `` et `<iframe>` il s'agit du l'attribut « loading » avec la valeur « Lazy ». Cet attribut est depuis peu nativement pris en charge par les navigateurs Chromium ainsi que Firefox, Safari est en cours de déploiement.

L'attribut loading avec la valeur lazy pour différé le chargement de l'iframe et de l'image.

Suite à cela, voici les valeurs que j'ai réussi à obtenir :



Dans un désir de vouloir optimiser ma page web afin d'améliorer mon score de Performance sur Lighthouse, il me fallait réduire mon temps de chargement, c'est ainsi qu'après quelques recherches j'ai découvert l'attribut mentionné plus haut « loading ».

La plupart des ressources trouvées n'était pas très précise et il me fallait des informations de meilleures qualités ainsi que la manière de le mettre en place. C'est ainsi que je suis tombé sur la page « Lazy loading » de Mozilla, cette page est très intéressante et contenait les informations que je désirais avec une traduction de ma part au préalable, car malgré que ce soit une documentation Mozilla, celle-ci n'est pas disponible en Français pour le moment.

La traduction m'a permis de mettre en place le chargement différé sur les images hors écran ainsi que des iframes :

Je suis tombé sur ce paragraphe ☒

Images and iframes

Very often, webpages contain many images that contribute to data-usage and how fast a page can load. Most of those images are off-screen (non-critical), requiring user interaction (an example being scroll) in order to view them.

Loading attribute

The loading attribute on an element (or the loading attribute on an <iframe>) can be used to instruct the browser to defer loading of images/iframes that are off-screen until the user scrolls near them.

```
<img src=>image.jpg> alt=>...> loading=>lazy>>  
<iframe src=>video-player.html> title=>...> loading=« lazy>></iframe>
```

The load event fires when the eagerly-loaded content has all been loaded; at that time, it's entirely possible (or even likely) that there may be lazily-loaded images that are within the visual viewport that haven't yet loaded.

Que j'ai traduit par ☒

Images et Iframes

Très souvent, les pages Web contiennent de nombreuses images qui contribuent à l'utilisation des données et à la vitesse de chargement d'une page. La plupart de ces images sont hors écran (non-critique), nécessitant une interaction de l'utilisateur (un exemple étant le scroll ou défilement) afin de les voir ou de les visualisés.

Attribut de chargement

L'attribut loading sur l'élément `` (ou l'attribut loading sur un `<iframe>`) peut être utilisé pour demander au navigateur de différer le chargement des images / iframes qui sont hors écran jusqu'à ce que l'utilisateur les fasse défiler.

```
<img src=»image.jpg» alt=»...» loading=»lazy»>  
<iframe src=»video-player.html» title=»...» loading=« lazy»></iframe>
```

L'événement de chargement se déclenche lorsque le contenu chargé avec « empressement » a été chargé; à ce moment-là, il est tout à fait possible (ou même probable) qu'il y ait des images chargées « paresseusement » qui se trouvent dans l'écran visible qui ne sont pas encore chargées.

Afin de mettre en place un système de gestion de cookie, j'ai décidé d'utiliser « tarteaucitron » qui est un script javascript Open-Source qui gère les cookies de façon automatique et conforme aux recommandations du RGPD.

INSTALLER TARTEAUCITRON.JS

- Je vais sur le site <https://tarteaucitron.io/> et je clique sur le menu «Guide d'installation»
- Je clique sur « Installation manuelle gratuite »
- Je clique sur le lien GitHub afin de télécharger la dernière version de Tarteaucitron.js
- Je décompresse celui-ci et je l'ajoute dans mon projet de la manière suivante: Le fichier décompresser est installé dans mon projet dans le fichier public
- j'ajoute le code suivant dans le Head de mon code

```
<script type="text/javascript" src="{{asset('tarteaucitron/tarteaucitron.js')}}">
</script>
<script>
    tarteaucitron.init({
        «privacyUrl»: «>, /* URL de la page de la politique de vie privée */
        «hashtag»: «#tarteaucitron», /* Ouvrir le panneau contenant ce hashtag */
        «cookieName»: «tarteaucitron», /* Nom du Cookie */
        «orientation»: «middle», /* Position de la bannière (top - bottom) */
        «showAlertSmall»: true, /* Voir la bannière réduite en bas à droite */
        «cookieslist»: true, /* Voir la liste des cookies */
        «adblocker»: false, /* Voir une alerte si un bloqueur de publicités est détecté */
        «AcceptAllCta»: true, /* Voir le bouton accepter tout (quand highPrivacy est à true) */
        «highPrivacy»: true, /* Désactiver le consentement automatique */
        «handleBrowserDNTRequest»: false, /* Si la protection du suivi du navigateur est activée, tout interdire */
        «removeCredit»: false, /* Retirer le lien vers tarteaucitron.js */
        «moreInfoLink»: true, /* Afficher le lien 'voir plus d'infos' */
        «useExternalCss»: false, /* Si false, tarteaucitron.css sera chargé */
        «cookieDomain»: '.my-multisite-domaine.fr', /* Cookie multisite */
        «readmoreLink»: '/cookiespolicy' /* Lien vers la page 'Lire plus' */
    });
</script>
```

- Je clique ensuite sur « Next step » en bas de la page
- J'ai ensuite des bouts de code à modifier sur mon projet afin que Tarteaucitron puisse détecter tous les cookies que j'utilise (exemple pour Google Analytics)

Voici comment se présente la gestion des cookies sur le site, il permet à l'utilisateur de sélectionner ou non ce qu'il souhaite stocker.

POLITIQUE DE CONFIDENTIALITÉ

J'ai aussi mis en place une politique de confidentialité accessible dans le footer sur toutes les pages du site :

Copyright© 2021 Les paniers de l'Isac Tous droits réservés - Politique de confidentialité - Mentions légales - 

Le footer du site avec le liens vers les mentions légales et la politique de confidentialité.

Voici un extrait de la politique de confidentialité :

6. Protection des biens et des personnes - gestion des données personnelles : En France, les données personnelles sont notamment protégées par la loi n° 78-87 du 6 janvier 1978, la loi n° 2004-801 du 6 août 2004, l'article L. 226-13 du Code pénal et la Directive Européenne du 24 octobre 1995.

Sur le site www.egentric.fr/lespaniersdelisac, Erwan Gentric ne collecte des informations personnelles (suivant l'article 4 loi n° 78-17 du 06 janvier 1978) relatives à l'utilisateur que pour le besoin de certains services proposés par le site www.egentric.fr/lespaniersdelisac.

L'utilisateur fournit ces informations en toute connaissance de cause, notamment lorsqu'il procède par lui-même à leur saisie.

Il est alors précisé à l'utilisateur du site www.egentric.fr/lespaniersdelisac l'obligation ou non de fournir ces informations.

Conformément aux dispositions des articles 38 et suivants de la loi 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés, tout utilisateur dispose d'un droit d'accès, de rectification, de suppression et d'opposition aux données personnelles le concernant. Pour l'exercer, adressez votre demande à www.egentric.fr/lespaniersdelisac par email : infogentric@gmail.com ou par écrit dûment signée, accompagnée d'une copie du titre d'identité avec signature du titulaire de la pièce, en précisant l'adresse à laquelle la réponse doit être envoyée.

Aucune information personnelle de l'utilisateur du site www.egentric.fr/lespaniersdelisac n'est publiée à l'insu de l'utilisateur, échangée, transférée, cédée ou vendue sur un support quelconque à des tiers. Seule l'hypothèse du rachat du site www.egentric.fr/lespaniersdelisac et de ses droits autorise Erwan Gentric à transmettre les dites informations à l'éventuel acquéreur qui serait à son tour tenu à la même obligation de conservation et de modification des données vis à vis de l'utilisateur du site www.egentric.fr/lespaniersdelisac.

Le site www.egentric.fr/lespaniersdelisac est en conformité avec le RGPD voir notre politique RGPD www.egentric.fr/lespaniersdelisac/RGPD.

Les bases de données sont protégées par les dispositions de la loi du 1er juillet 1998 transposant la directive 96/9 du 11 mars 1996 relative à la protection juridique des bases de données.

Ce rapport m'a permis d'exposer tout les étapes de créations de mon projet pour «les paniers de l'Isac. Ce projet m'a permis de mettre en avant les producteurs locaux ainsi que les commerçants.

Les trois grandes étapes de mon projet consistaient en premier lieu d'établir le cahier des charges.

Dans un second temps, connaître et déterminer les besoins fonctionnels du site. Pour ce faire j'ai réalisé des diagrammes me permettant d'élaborer l'interface utilisateur et d'établir les différents rôles.

Pour finir, la dernière étape était la conception du site, en prenant en compte les aspects de sécurité, d'ergonomie, de responsive et de test.

Mon projet a été une expérience très enrichissante malgré, les difficultés et les contraintes de temps, je garde une impression très positive de ce projet.

Travailler sur un projet de bout en bout et de maîtriser chaque étape de production m'a permis de rapidement progresser et également de voir beaucoup plus clair vis à vis d'une conception similaire dans le monde professionnel,

Le principal défi était à la fois d'apprendre à coder sous Symfony et de respecter les délais du projet.

Pour la suite, je ne compte absolument pas m'arrêter là, je suis déterminé à améliorer mes compétences en Php et Javascript !

Ce sont des langages qui m'intéresse, et lors de mon stage je me suis aperçu que j'avais encore beaucoup de chose à apprendre, pour le javascript c'est un langage que je n'ai pas pu assez mettre en pratique durant ma formation, et que j'aimerais maîtriser.