Dossier de Synthèse

Octobre 2021

Site pour les passionnés du monde équestre

EmprunteMonPoney

Pour le titre professionnel de Développeur Web et Web Mobile

ELAN formation

Colmar

Béatrice Machalica

Remerciements

Je tiens avant tout à remercier les formateurs, Stéphane Smail, Gilles Muess, Cindy Cahen, Mickael Murmann, Virgile Gibello et Paul Van Kalck qui ont su m'aider à atteindre mes objectifs et à me faire parvenir jusqu'au bout de la formation.

Leurs précieux conseils et encouragements m'ont permis d'avancer tout au long de ce projet. En effet, leurs directives et conseils ont pu m'aider dans les moments où je m'éparpillais.

Donc, je tiens réellement à remercier mes formateurs pour leur soutien, aide et patience. Sachez que votre gentillesse et dévouement m'ont véritablement aidé durant cette formation tant exigeante que passionnante.

J'aimerais remercier toute l'équipe de GreenCub où j'ai réalisé mon stage et où j'ai pu apprendre de nouvelles connaissances.

Quant à mon équipe DWWM de Colmar, dont je garderais toujours une pensée chaleureuse, je veux remercier chacune des personnes : Melvin, Alexis, Martin, Jean-Philippe, Florian, Killiann, Thomas, Terence et Victor. Nous avons su nous entraider et nous soutenir durant les périodes difficiles. C'était un véritable plaisir de partager cette aventure auprès de vous tous.

Je souhaite également remercier un ami, un frère et un confident, Mathieu Schweitzer. Une personne qui a su me donner des précieuses pistes de réflexions. Tes conseils et surtout tes encouragements ont été plus que précieux pour mener à bien ce parcours.

Enfin, je tiens à adresser une pensée toute particulière à la personne qui partage ma vie. Anthony, tu ne cesseras jamais d'être le vent dans mes voiles. Je ne te remercierais jamais assez pour ton soutien intarissable.

Résumé

Ce projet a pour but principal la création d'un site destiné aux propriétaires d'équidés et aux personnes passionnés du monde équestre.

Le projet consiste à proposer aux utilisateurs une solution simple et innovante permettant de trouver un cheval ou un « emprunteur » selon les besoins des personnes tout en proposant un design moderne et accessible.

Pour le visiteur il s'agit de pouvoir consulter les annonces de demi-pension et également pouvoir contacter l'auteur de l'annonce via une messagerie.

Pour résumer, le développement comprend les fonctionnalités principales suivantes :

- L'utilisateur doit pouvoir créer un compte et se connecter ;
- L'utilisateur doit pouvoir créer une annonce pour son profil ou pour son cheval selon ses besoins ;
- L'utilisateur doit pouvoir effectuer une recherche d'annonce sur le site ;
- L'utilisateur doit pouvoir ajouter une annonce à ses favoris et commenter les annonces ;
- Et enfin, l'utilisateur doit pouvoir contacter un autre utilisateur via une messagerie.

De plus, ce projet a également pour but la validation du titre professionnel de « Développeur Web et Web Mobile ». En effet, plusieurs compétences ont été travaillées durant la réalisation du projet qui sont exigées par le REAC.[1]

Dans un premier temps, la création de plusieurs maquettes a été nécessaire ainsi que la conceptualisation des données.

Ensuite, les compétences front-end ainsi que back-end ont été travaillées durant le développement du site.

Enfin, l'utilisation constante de l'anglais, le respect des normes de sécurité ainsi que la veille informationnelle ont également occupé une place importante durant la réalisation de ce projet.

Plusieurs technologies, outils et langages ont été utilisés pour ce projet. Cependant, j'ai réalisé ce site en grande partie grâce à *Symfony 5* un puissant framework PHP. J'ai décidé d'utiliser ce framework car, j'ai particulièrement apprécié son utilisation et je souhaite me spécialiser dans cette technologie.

[1] Référentiel Emploi Activités Compétences du Titre Professionnel Développeur web et web mobile niveau III site : Ministère du Travail, de l'Emploi et de l'Insertion (travail-emploi.gouv.fr)

Table des matières

Table des figures

I. Compétences couvertes par le projet

A. Compétences Front-end travaillées

- Maquetter une application : élaborer le modèle conceptuel de données (MCD) ; utilisation de plusieurs outils de maquettage (*JustInMind, Draw.io*, *AdobeXD*).
- Développer une interface utilisateur web statique : réalisation de plusieurs pages à l'aide de HTML, CSS et le framework CSS « Bootstrap ».
- Développer une interface utilisateur web dynamique : intégration de scripts événementiels avec un langage de script client (JavaScript).
- Développer des pages web en lien avec une BDD : implémentation de plusieurs pages web reliées à une BDD et nécessitant des données spécifiques par ex. page du compte utilisateur.

B. Compétences Back-end travaillées

- Concevoir et créer une base de données : stockage organisé de données selon le schéma physique ; gestion des données à l'aide d'un SGBD ; maitrise des instructions de création, de modification et de suppression (CRUD).
- Développer des composants d'accès aux données : manipulation des données à l'aide de *Doctrine* et du langage *SQL*.
- Développer la partie back-end d'une application web ou web mobile : développement de la partie back-end du site à l'aide de *Symfony 5* un framework *PHP* ; utilisation de la programmation orientée objet ainsi que du design pattern MVC.
- Respecter les normes de sécurité : veille technologique sur la sécurité informatique et sur les vulnérabilités connues ; consultation du site *OWASP*.

C. Compétences transversales travaillées

- Utilisation de l'anglais durant l'activité professionnelle en développement web et web mobile : recherche d'information et de solutions en anglais ainsi que lecture de documentations techniques officielle en anglais.
- Mise en place d'un système de veille informationnelle : actualiser et partager ses compétences en développement web et web mobile à l'aide de plusieurs sources.

II. Introduction

A. Reconversion à ELAN

Issue d'une famille polonaise j'ai d'abord voulu étudier les langues. J'ai ainsi poursuivi des études dans la sociolinguistique à l'Université de Strasbourg où j'ai pu acquérir de nombreuses connaissances.

J'ai pu étudier plusieurs langues telles que l'anglais, le polonais, l'allemand ou encore le suédois. Par ailleurs, j'ai pu étudier les dynamiques culturelles et linguistiques de nombreux pays et diasporas ainsi que la politique linguistique européenne.

Néanmoins, durant ce cursus j'ai commencé à m'intéresser au monde informatique notamment grâce à des cours du soir optionnels.

J'ai donc voulu m'orienter vers le développement web après avoir obtenu le master II « Plurilinguisme et Interculturalité européenne ».

Les multiples confinements de l'année 2020 m'ont permis d'utiliser mon temps à bon escient et m'intéresser davantage aux langages informatiques. C'est pourquoi, avec mon intérêt grandissant pour ce domaine, j'ai décidé d'entrer à ELAN-formation de Colmar en janvier 2021.

L'organisme ELAN, fondé en 1993, propose plusieurs formations professionnelles dans les domaines tels que la bureautique, la vente, les techniques de secrétariat, l'informatique etc. Ses locaux sont établis à Strasbourg, Sélestat, Haguenau, Saverne et Colmar.

Cette structure correspondait à mes attentes car les stagiaires sont amenés à développer des projets et à pratiquer au maximum. Par ailleurs, les connaissances visées coïncident avec les exigences du marché de l'emploi.

De plus, j'ai décidé de développer un projet dans le cadre de cette formation qui, non seulement représente un intérêt personnel, mais il implique également l'utilisation de nombreux outils et langages informatiques. La partie suivante est consacrée au contexte de ce projet.

B. Contexte

Ce projet a pour but d'allier deux passions : le développement et le monde équestre. Plus précisément, le projet consiste à créer **un site web de mise en relation des utilisateurs**.

En effet, les cavaliers rêvent pour la plupart d'acheter leur propre cheval. Malheureusement, un animal aussi imposant requière des soins, du temps et un budget conséquent qui n'est pas à la portée de tous.

D'un autre côté, certains propriétaires d'équidés recherchent des personnes souhaitant s'occuper de leur animal. C'est pourquoi la **demi-pension** (ou autrement dit « DP ») s'impose comme un compromis intéressant pour les deux parties.

Une demi-pension est un accord officieux entre un propriétaire (un particulier ou une structure équestre) et une ou plusieurs personnes tierces, que nous appellerons désormais « **emprunteurs** ». Le propriétaire de l'équidé met donc à disposition son animal quelques jours par semaine en échange d'un paiement.

Le but du site sera donc de faciliter la mise en relation des propriétaires et des emprunteurs. En effet, le site proposera aux emprunteurs **et** aux propriétaires d'effectuer des recherches en fonction de leurs besoins.

De ce fait, le site reflète une certaine originalité car, aucun site existant actuellement[2] ne propose, *a priori*, ce type de service à double sens.

En effet, les sites spécialisés existants permettent aux propriétaires de poster des annonces de leurs chevaux mais, les propriétaires sont résignés à devoir attendre un contact d'une personne intéressée.

Mon projet a pour but de permettre aux propriétaires de rechercher des emprunteurs qui correspondent aux critères et besoins spécifiques à chaque demi-pension. Ainsi, ce projet permettrait aux propriétaires des équidés, qui souhaitent proposer une demi-pension, de devenir acteurs de leur initiative.

Afin de répondre à cet objectif, le site doit avoir une structure spécifique et il doit proposer plusieurs fonctionnalités. Ces dernières seront détaillées explicitement dans la prochaine partie.

[2] Cette recherche a été effectuée le 25/02/21

III. Cahier des charges

A. Fonctionnalités du site

Afin que le site remplisse ses objectifs, ce dernier doit proposer plusieurs fonctionnalités qui sont décrites dans les parties ci-dessous. Pour rappel, le but du site est de mettre en relation des propriétaires ainsi que des emprunteurs et leur permettre de trouver une DP qui répondra à leurs besoins respectifs.

1. Inscription et connexion

Dans un premier temps, l'utilisateur pourra donc s'inscrire et par la suite se connecter à l'application. Lors de l'inscription, l'utilisateur pourra choisir s'il souhaite s'inscrire autant qu'emprunteur **ou** propriétaire. L'ergonomie devra ici être particulièrement travaillé pour éviter toute confusion. L'inscription demandera à l'utilisateur les informations suivantes :

- Nom d'utilisateur ;
- Mail ;
- Téléphone (optionnel) ;
- Mot de passe ;
- Code postal ;
- Ville ;
- Département ;
- Statut (emprunteur ou propriétaire d'équidés).

Ces formulaires devront respecter les consignes de sécurités (notamment pour se protéger des failles XSS, injection SQL, etc. cf. Sécurité).

Par ailleurs, le mail de l'utilisateur représente **l'identifiant unique** de chaque utilisateur. Enfin, si l'inscription de l'utilisateur est complète et valide le client sera redirigé vers la page de connexion où il pourra se connecter. Pour ce faire, l'utilisateur devra renseigner son mail ainsi que son mot de passe dans un formulaire de connexion.

2. Compte utilisateur

Une fois inscrit et connecté l'utilisateur doit pouvoir consulter son compte. Afin de respecter le **RGPD** (cf. respect du RGPD), l'utilisateur doit pouvoir modifier ses informations personnelles ou supprimer son compte.

Par ailleurs, si un utilisateur souhaite supprimer son compte et clique sur le bouton prévu à cet effet un message de confirmation devra être validé pour éviter une éventuelle suppression involontaire.

Comme énoncé précédemment, lors de son inscription un utilisateur aura le choix entre le rôle de « l'emprunteur » ou du « propriétaire ». En effet, un « propriétaire » pourra consulter, une fois connecté, ses chevaux inscrits ainsi que les annonces pour ces derniers. Le même fonctionnement concerne l'utilisateur « emprunteur » qui pourra consulter son annonce qui présente son profil. Cette fonctionnalité est présentée plus précisément dans la partie suivante.

3. Les annonces liées aux profils

L'utilisateur, une fois inscrit et connecté, pourra créer une annonce selon son rôle (soit une annonce d'une demi-pension ou une annonce d'un profil emprunteur). Voici un schéma qui illustre ce fonctionnement :

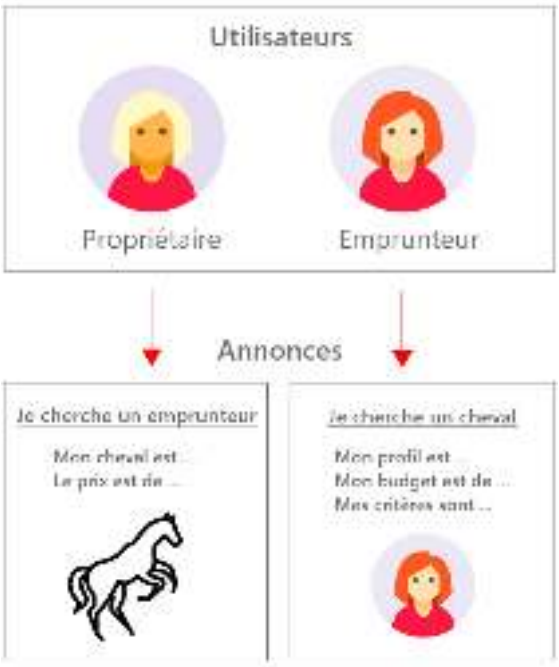

Figure 1 Deux types d'annonce

Ainsi, un « emprunteur » pourra poster et proposer son profil aux propriétaires d'équidés. Plusieurs informations seront présentes sur cette dernière :

- La catégorie de l'annonce ;
- Le nom d'utilisateur de l'emprunteur ;
- Une ou plusieurs photos de l'emprunteur (cf. ajout de photos) ;
- Les activités souhaitées (dressage, complet, balade, western, éthologie) ;
- L'adresse de l'emprunteur ;
- Le budget de l'emprunteur ;
- Un texte de description.

Pareillement, un « propriétaire » pourra proposer son animal aux emprunteurs à l'aide d'une annonce. Cette dernière comprendra :

- La catégorie de l'annonce ;
- Le nom de l'animal ;
- Une ou plusieurs photos du cheval ;
- La race et l'âge du cheval ;
- La taille au garrot et le sexe ;
- L'adresse à laquelle se trouve l'équidé ;
- Les activités pratiquées (dressage, complet, balade, western, éthologie) ;
- Le prix de la demi-pension ;
- Un texte de description.

Les nouvelles annonces, récemment publiées, apparaîtront sur la page d'accueil dans une section dédiée afin d'accorder une visibilité supplémentaire aux nouveaux utilisateurs.

4. L'ajout de photos

Lorsqu'un utilisateur postera une annonce il doit pouvoir également ajouter plusieurs photos. En effet, cela permet de rendre une annonce plus intéressante et attractive.

De ce fait, lorsque l'utilisateur connecté sera sur le point de créer une annonce, il pourra cliquer sur un bouton lui permettant de choisir une ou plusieurs photos. Cette fonctionnalité demande une attention toute particulière car, il sera nécessaire de se prémunir de la faille upload (cf. sécurité).

5. Recherche d'un équidé ou d'un emprunteur

Une fois inscrit, l'utilisateur doit pouvoir effectuer une recherche selon la catégorie de l'annonce :

- Soit une recherche d'offre de demi-pension (profil d'un cheval) ;
- Soit une recherche de profil emprunteur.

Par ailleurs, le visiteur doit pouvoir filtrer ses recherches à l'aide d'un système de tri spécifique. Ainsi, un emprunteur pourra par exemple ajuster des critères tels que :

- Le rayon (distance) autour de la ville souhaitée ;
- Les disciplines praticables (dressage, complet, balade, western, éthologie) ;
- Le prix de la demi-pension (prix minimum et prix maximum).

Quant à l'utilisateur « propriétaire », il pourra trier sa recherche en fonction des critères suivants :

- Le rayon (distance) autour de la ville souhaitée ;
- Le prix souhaité (prix minimum et prix maximum) ;
- Les disciplines pratiquées par le cavalier (dressage, complet, balade, western, éthologie).

Pour résumer, le site proposera plusieurs filtres de recherche dans le but de proposer aux utilisateurs des offres adaptées à leurs profils et besoins.

6. Ajouter une annonce aux favoris

L'utilisateur connecté doit pouvoir ajouter une annonce aux favoris en cliquant sur une icône située au sommet de la page des détails d'une annonce appréciée.

Lorsque l'utilisateur clique sur cette icône en forme de cœur elle devra être remplacée par une icône entièrement remplie. Ainsi, l'utilisateur peut directement constater que l'annonce a bien été ajoutée à ses favoris. Dans le cas où l'utilisateur souhaite enlever une annonce des favoris il lui suffira de cliquer à nouveau sur l'icône qui cèdera alors sa place à une icône de cœur vide.

7. Les commentaires

Un utilisateur connecté doit pouvoir commenter les différentes annonces. De plus, l'utilisateur doit pouvoir modifier ou supprimer son commentaire. Par ailleurs, il sera également possible pour l'utilisateur de commenter plusieurs fois une même annonce.

L'utilisateur pourra également voir les commentaires des autres utilisateurs. En effet, l'ensemble des commentaires des utilisateurs seront affichés sous les détails d'une annonce.

8. Messagerie

Si un utilisateur trouve une annonce intéressante il doit pouvoir contacter l'auteur de cette dernière. C'est pourquoi la messagerie privée doit pouvoir permettre aux utilisateurs inscrits et connectés de communiquer entre eux. La messagerie comportera une boite de réception et une boite d'envoi. Le bouton qui permet d'envoyer des messages sera quant à lui présent directement à côté du nom de l'auteur d'une annonce sous la forme d'une icône d'enveloppe.

9. Espace administrateur

Le back office sera exclusivement accessible aux administrateurs. Un administrateur est un utilisateur particulier qui possède le rôle « ROLE_ADMIN ». Ce dernier peut par exemple vérifier si l'utilisation du site par les utilisateurs est conforme et non malveillante. De ce fait, l'administrateur peut en cas de besoin supprimer un utilisateur ou changer leurs rôles. Il peut également ajouter, modifier ou supprimer une catégorie d'annonce ou une activité équestre.

B. Arborescence

Voici un schéma de l'arborescence du site souhaitée :

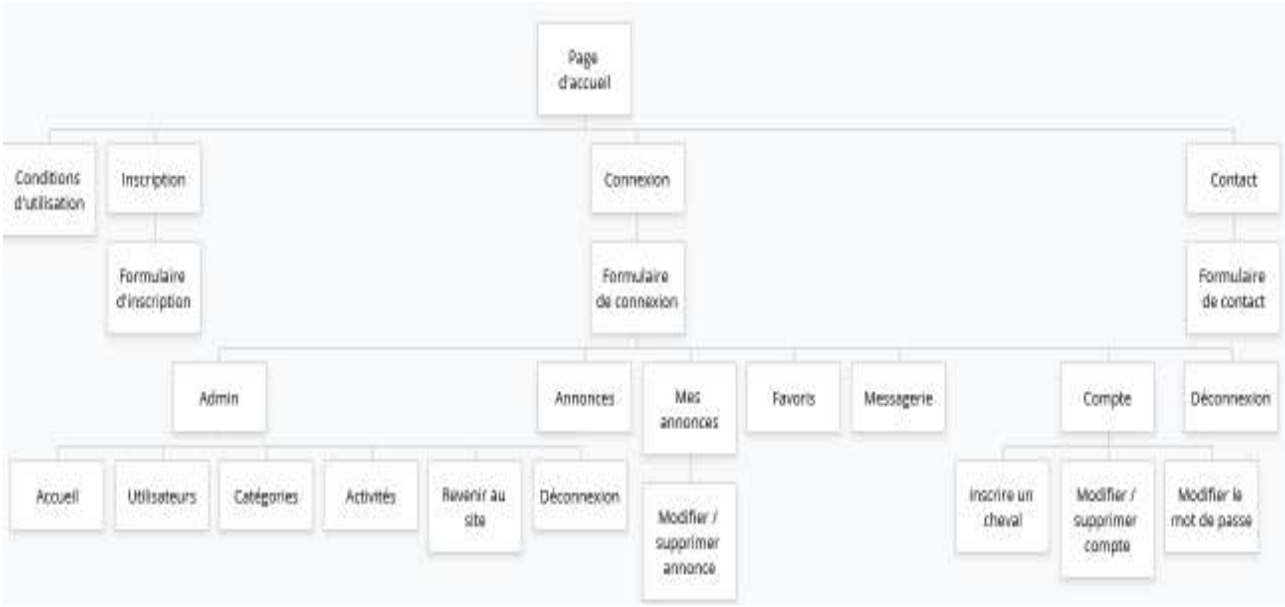

Figure 2 Arborescence du site web

C. Cibles

Le site vise à toucher les publics suivants :

- Particuliers, propriétaires ou non d'équidés ;
- Structures propriétaire d'équidés (centres équestres etc.).

Bien évidemment, le site vise un public qui se passionne pour le monde équestre ainsi, des images et symboles se rapportant à cette thématique seront régulièrement utilisés. Le design du site devra donc être sobre et moderne tout en restant attrayant.

Le site doit inspirer confiance aux utilisateurs. En effet, la confiance est une valeur fondamentale puisque les propriétaires sont amenés à confier leur animal à des personne étrangères.

L'ergonomie doit être travaillée pour permettre une navigation simple et intuitive répondant aux besoins d'un public dont l'âge peut énormément varier. D'autant plus, que le public visé peut ne pas être habitué à l'utilisation des services web.

D. Respect du RGPD

Le règlement général sur la protection des données (RGPD) « *est un texte réglementaire européen qui encadre le traitement des données de manière égalitaire sur tout le territoire de l'Union Européenne. Il est entré en application le 25 mai 2018.* »[3]

Toutes les structures sont tenues de respecter cette loi car, cette dernière s'applique à toutes organisations, publiques ou privées et quelles que soit leurs tailles (entreprises, ministères, administrations, associations, etc.).

Le RGPD stipule qu'il s'agit de protéger les « *données à caractère personnel* »[4] ou autrement dit « *toute information se rapportant à une personne physique identifiée ou identifiable* ».[5] Ainsi, les données concernées sont par exemple : « *un identifiant, tel qu'un nom, un numéro d'identification, des données de localisation, un identifiant en ligne, ou à un ou plusieurs éléments spécifiques propres à son identité physique, physiologique, génétique, psychique, économique, culturelle ou sociale* ».[6]

Afin de respecter ce texte, l'utilisateur doit pouvoir modifier ou supprimer son compte. Mais, il doit également pouvoir modifier ou supprimer les données personnelles qui sont requises lors de l'inscription telles que : le prénom, le nom, le courriel, le numéro de téléphone, le mot de passe, etc. Un travail de réflexion a été effectué au préalable pour ne demander à l'utilisateur que les informations les plus adéquates et pertinentes (minimisation des données).

Dans le cadre du respect du RGPD il sera également nécessaire d'effectuer un hachage des mots de passe des utilisateurs. Le hachage des mots de passe est non seulement indispensable pour renforcer la sécurité (cf. veille sécurité) mais, il constitue également un point important du respect du RGPD.

Sur Symfony, depuis la version 4.3, l'algorithme de hachage est automatiquement sélectionné dans le fichier `security.yaml` (l'algorithme '`sodium`' peut être utilisé ou en second plan '`argon2i`' ou encore '`bcrypt`').[7]

E. Design et ergonomie

Avant tout, le site web doit respecter les contraintes de « responsive design », c'est à dire que l'interface doit s'adapter au terminal qui la reçoit (mobile, tablette, ordinateur de bureau). Le site devra également passer la validation et respecter la norme W3C (convention de nommage etc.).

[3] Source : Site du « Ministère de l'économie des finances et de la relance », consulté le 10/03/2021 lien : https://www.economie.gouv.fr/entreprises/reglement-general-sur-protection-des-donnees-rgpd#
[4] RGPD, 25 mai 2018, CNIL. Chapitre 1 – dispositions générales, Article 4 – Définitions. Consulté le 18/02/2021.
[5] RGPD *Ibid*.
[6] RGPD *Ibid*.
[7] Source : https://symfony.com/blog/new-in-symfony-4-3-native-password-encoder

1. Inspirations et couleurs

D'un point de vue esthétique (design, animations, sobriété) j'ai particulièrement apprécié le blog équestre de Pauline Barbier « d'un cheval l'autre ».[8]

Mon choix des couleurs s'est porté sur un bleu roi foncé, le noir et un brun rappelant la couleur d'une robe d'un cheval bai. En effet, d'après les différents schémas de la psychologie des couleurs[9] le bleu reflète la confiance, le calme et la communication. Le noir apportera une touche d'élégance. Le brun quant à lui apportera du contraste et va attirer l'œil sur les informations importantes. *De prime abord* ces couleurs correspondent bien à nos *personas*.[10]

Les couleurs retenues pour ce projet[11] :

Figure 3 Choix des couleurs

2. Navigation

Le site aura une navigation simple et verticale. La page d'accueil expliquera succinctement le principe et le but du site. L'ergonomie sera donc ici particulièrement travaillée.

Pour rappel, les menus seront fixes et explicites. Un premier menu sera commun à la page d'accueil et à toutes les pages de niveau 1 (cf. arborescence). Un fil d'ariane ne semble, *a priori*, pas nécessaire pour ce projet.

3. Typographies

En ce qui concerne les typographies, les polices dites « sans empattements » seront privilégiées dans un souci de confort pour l'utilisateur. En effet, les polices d'écriture appelées également « sans sérif » sont réputées plus simples, plus modernes et plus agréable à la lecture et lors de la visualisation à l'écran.

Les polices retenues pour ce projet sont donc :

- « Roboto » pour les textes, car c'est une police simple et sans sérif permettant à l'utilisateur de ne pas se fatiguer à la lecture.

[8] Le site de Pauline Barbier, consulté le 21/02/21, lien : https://dunchevalautre.com/
[9] cf. annexes « Schémas de la psychologie des couleurs ».
[10] Les « *personas* », en marketing, représentent un archétype d'un groupe d'individus qui partagent plusieurs caractéristiques communes.
[11] Outil utilisé : HTML Color Picker du site w3schoolds.com lien: https://www.w3schools.com/colors/colors_picker.asp

- Et « Dancing script » pour les titres de niveau 2 de la page d'accueil. C'est une police type « handwriting » qui apporte une proximité et une touche d'élégance à la page d'accueil.

4. Référencement naturel

Le référencement naturel correspond à une optimisation pour les moteurs de recherche (ou autrement dit « SEO » pour *Search Engine Optimization*) et détermine le positionnement ainsi que la visibilité du site dans les pages de résultats. Il peut être amélioré grâce à un ensemble de pratiques.

Premièrement, la performance de l'application est déterminante (vitesse d'affichage, nombre de requêtes, etc.). Ensuite, les liens internes et externes sont également à prendre en compte. C'est pourquoi, plusieurs liens se trouveront dans le footer de la page d'accueil et menant vers des sites externes (Facebook, Instagram, YouTube) ou internes (conditions d'utilisation, contact).

Par ailleurs, j'ai utilisé des balises comme `<strong>`, `<em>` ou en attribuant une description des images dans les `<alt>` afin d'accroitre le référencement naturel. Les balises `<alt>` jouent également un rôle important dans l'accessibilité. En effet, il est recommandé d'utiliser ces balises d'après les normes WCA[12] de la W3C notamment pour rendre le site le plus accessible possible aux personnes malvoyantes ou aveugles.

5. Ergonomie et responsive design

Au niveau ergonomique, j'ai intégré le site web dans un souci de simplicité d'utilisation pour l'utilisateur. Par exemple, les composants nécessitant une action réagissent au passage de la souris ou changent de couleur.

Par ailleurs, le but du site et les fonctionnalités doivent être explicites et correctement expliqués. En effet, l'UX ne doit pas être négligé car le système de profil « emprunteur » et « propriétaire » par exemple peut potentiellement être déroutant. Donc, il est d'autant plus important de guider l'utilisateur et mettre à disposition des indications claires pour chaque fonctionnalité.

Comme l'exige le REAC, l'expérience de navigation a été repensée et adaptée pour les smartphones : simplification de l'information, dispositions différentes des éléments des pages etc. Il est essentiel d'adapter son site aux mobiles car, de plus en plus d'utilisateurs accèdent au web via leurs smartphones.[13] Ainsi, j'ai utilisé des « media queries » qui sont une spécification de CSS 3 (cf. Langages et technologies) et qui permettent d'appliquer un « responsive design ».

```
4   /* smartphone */
5   @media screen and (max-device-width: 765px) {
6     .mainMenu {
7       display: none;
8     }
```

[12] WCAG pour Web Content Accessibility Guidelines en anglais, sont un ensemble de normes pour rendre un contenu web plus accessible (aux personnes malvoyantes ou aveugles par exemple).
[13] 67% de la population française connecté accède à internet via un smartphone selon l'étude Google/TNS Baromètre Consommateur 2017, consulté le 06/06/2021, source : https://www.thinkwithgoogle.com/intl/fr-fr/strategies-marketing/mobile-et-apps/barom%C3%A8tre-consommateur-2017-le-mobile-pour-le-plus-grand-nombre/

```
142    /* iPad and iPad Pro */
143    @media screen and (min-device-width: 766px) and (max-width: 1025px) {
```

Figure 4 Media queries - responsive design

Ainsi, le contenu des pages va s'adapter en fonction des différents écrans des différents types d'appareils (tablettes, smartphones, ordinateurs etc.).

F. Contraintes non fonctionnelles

- Le site web doit respecter les contraintes graphiques de la charte graphique (cf. Design et ergonomie) ;
- Le site web doit respecter les normes de sécurité (afin de se prémunir de la faille d'injection SQL, la faille XSS et la faille CSRF) exigés par les référentiels suivants :
    - « Référentiel de certification du titre professionnel développeur web et web mobile » ;
    - « Référentiel emploi activités compétences du titre professionnel développeur web et web mobile » ;
- Le site web doit respecter les contraintes de « responsive design », en d'autres termes l'affichage du site doit s'adapter aux écrans des différents matériels numériques utilisés (écrans d'ordinateurs, tablettes, téléphones portables, etc.).

IV. Benchmark

Voici les sites qui ont été consultés durant la réalisation du Benchmark :

- *Securide.fr*
- *Cheval annonce.com*
- *Equirodi.com*
- *eHorses.fr*
- *Facebook.com*
- *Leboncoin.fr*
- *Empruntemontoutou.com*

Ces sites proposent différentes fonctionnalités qui sont exposées progressivement dans les parties suivantes.

A. Les sites spécialisés

Plusieurs sites ont été consultés qui sont spécialisés dans les demi-pensions équines : *securide.fr*, *cheval annonce.com*, *equirodi.com, eHorses.fr*. Chacun de ces sites proposent aux utilisateurs un système de recherche et de tri par différents critères.

Securide.fr propose dans son système de tri le « type de pension » ainsi que le « pays » et la « région ». Le tri ne semble pas assez exploité ici (pour mon projet il serait intéressant d'avoir par exemple la possibilité d'ajouter une fourchette du prix de la DP et le type d'activité pratiqué avec le cheval).

Cependant, une légende se trouve à côté des annonces des demi-pensions ce qui améliore nettement l'ergonomie du site et permet d'afficher explicitement les informations sur la DP (par exemple : une DP sous contrat, pension box, pension au pré, transport en commun, moins de 18 ans, licence FFE, compétition, contact par courriel, contact par téléphone). Le site propose également la recherche de location / cotransport de van[14] et de camion.

Figure 5 Exemple d'une annonce sur securide.fr

Le site *Cheval annonce.com* propose de poster et de rechercher plusieurs types d'annonce (vente d'équidés, de matériel, pensions, camions & vans, saillies, emplois, etc.). Un forum et des articles sont proposés sur le site. Il est nécessaire de noter que le système de tri pour les demi-pensions est beaucoup plus complet et propose :

[14] « Véhicule fermé, aménagé pour le transport des chevaux ». (Larousse)

- Les critères de disciplines praticables (dressage, obstacle, complet, endurance, TREC, horse ball, attelage, voltige, balade, western) ;
- Les installations disponibles (carrière, manège, rond de longe) ;
- L'âge de l'équidé ;
- La taille de l'équidé ;
- Médias (annonce avec photo et / ou avec vidéo) ;
- Et enfin une fourchette de prix.

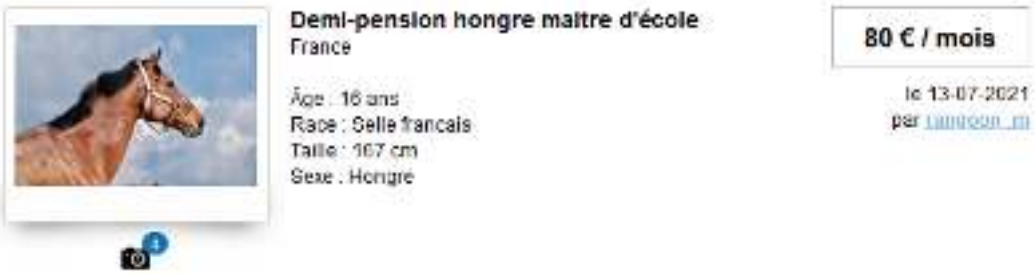

Figure 6 Exemple d'une annonce sur chevalannonce.com

Quant au site *Equirodi.com*, hormis la fonctionnalité « annuaire » qui permet de rechercher un professionnel, il dispose de fonctionnalités similaires à celle de *Cheval annonce.com.* Enfin, *eHorses.fr* est davantage focalisé sur la vente des équidés bien qu'il soit possible également de poster des annonces de demi-pension dans les « petites annonces ».

B. Facebook et Leboncoin

Les sites tels que *Facebook* et le *Boncoin* font figure d'exception car ils n'appartiennent pas au monde équestre. Cependant, le public cible utilise ces sites pour leurs fonctionnalités intéressantes.

En effet, on retrouve sur *Facebook* des groupes spécifiques aux besoins des utilisateurs et la popularité du réseau social représente un avantage certain. Par ailleurs, le *Boncoin* permet d'effectuer des recherches plus ou moins précises et possède également une certaine notoriété.

C. Inspiration principale

Enfin, le site *Empruntemontoutou.com*[15] a été une source d'inspiration pour mon projet. En effet, ce site propose de mettre en relation des **propriétaires de chiens et des emprunteurs**.

La page d'accueil propose une navigation ergonomique et il faut souligner que le fonctionnement du site est plusieurs fois expliqué à l'aide de texte, de témoignages et de schémas.

[15] Le site « Emprunte mon toutou », que j'utilise régulièrement, m'a beaucoup inspiré durant la réalisation de ce projet, lien du site : https://www.empruntemontoutou.com/

V. Réalisation du projet

A. Méthodologies

1. La stratégie « MVP »

La stratégie MVP (*Minimum Viable Product,* en français produit minimum viable) s'impose comme une stratégie évidente pour délivrer un projet fonctionnel en respectant les délais imposés.

La stratégie MVP « *désigne un processus d'innovation et de développement produit qui est notamment en vigueur dans le domaine de l'informatique […] une stratégie MVP consiste à privilégier la vitesse de développement et les délais de mise sur le marché (time to market) en sacrifiant certaines fonctionnalités ou performances du produit ou service considérées au départ comme non indispensables* ».[16]

Autrement dit, le but de cette stratégie est double :

- Premièrement, elle consiste à optimiser le temps de développement afin de respecter les délais imposés ;
- Deuxièmement, elle vise également à obtenir une satisfaction de la part des clients en proposant un produit fonctionnel qui comportera les fonctionnalités jugées comme indispensables.

Voici un schéma explicatif de la stratégie MVP :

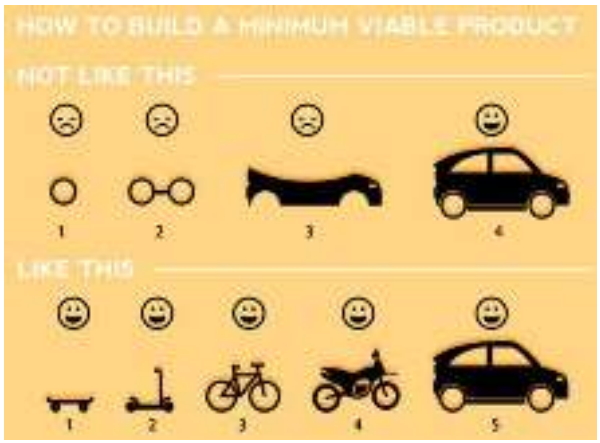

Figure 7 Stratégie MVP

On peut facilement constater grâce à la figure ci-dessus que le but principal d'un MVP consiste à proposer, dès le premier abord, un produit fonctionnel qui va satisfaire l'utilisateur. Dès le départ, le produit proposé au client est plus restreint que l'objectif final mais, il doit proposer une fonctionnalité acceptable. Le produit va par la suite évoluer tout en gardant une fonctionnalité satisfaisante.

Cependant, il faut garder à l'esprit que la stratégie MVP n'est pas infaillible car « *elle peut provoquer éventuellement une déception des utilisateurs qui pourraient désinstaller une application (et donc ne pas voir arriver les fonctionnalités d'enrichissement) ou renoncer à*

[16] Source : « Définitions marketing », MVP écrit par Bertrand Bathelot, lien : https://www.definitions-marketing.com/definition/mvp/ consulté le 02/03/2021.

l'usage d'un produit ou d'un logiciel ».[17] Afin d'éviter ce problème il faut scrupuleusement établir une liste des fonctionnalités prioritaires et essentielles au bon fonctionnement du site.

De ce fait, le tableau ci-dessous présente une liste des fonctionnalités ainsi que le degré de priorité de ces dernières :

Fonctionnalités	Priorités
L'inscription de l'utilisateur	Primaire
Connexion sécurisée	Primaire
Accès au compte utilisateur	Primaire
Modification des informations du compte	Primaire
La suppression du compte	Primaire
L'ajout d'une annonce (emprunteur ou pour un équidé)	Primaire
Modification d'une annonce	Primaire
La suppression d'une annonce	Primaire
L'ajout de commentaires sous les annonces	Secondaire
Modification des commentaires	Secondaire
La suppression des commentaires	Secondaire
Recherche d'un emprunteur ou d'une demi-pension	Primaire
Système de messagerie	Secondaire
La possibilité d'ajouter une annonce aux favoris	Secondaire
L'ajout de plusieurs filtres de recherche	Secondaire
L'ajout de plusieurs photos à une annonce	Secondaire
Tri de l'ordre d'affichage des résultats par critères (prix, etc.)	Secondaire
Back office accessible aux administrateurs	Primaire
Formulaire de contact	Secondaire
Anti-spam grâce au Captcha	Secondaire

2. Méthode Agile

Pour réaliser ce projet je me suis intéressée à la méthode Agile. En effet, cette méthode a été créée pour assurer une meilleure gestion des projets de développement web et informatique. Le manifeste prône ainsi « 4 valeurs » fondamentales : l'esprit d'équipe, l'application, la collaboration avec le client, et l'acceptation du changement (l'importance d'une plus grande souplesse au cours du développement d'un projet).

Il faut préciser que je n'ai pas travaillé au sein d'une équipe dans le cadre de ce projet, néanmoins j'ai voulu m'intéresser tout de même à cette méthode. En effet, elle est régulièrement utilisée dans le cadre professionnel et son cycle de développement peut apporter un cadre méthodologique non négligeable.

Ainsi, le cycle de développement se découpe en plusieurs parties interconnectées. Dans un premier temps le cycle commence par une rencontre ou une réunion de l'équipe responsable d'un projet. Dans mon cas j'ai remplacé cette étape par une séance de mise au point des

[17] Bertrand Bathelot, *Ibid*. Consulté le 02/03/2021.

objectifs ou par une consultation des formateurs en cas de besoin. Les étapes suivantes sont : la planification, le travail sur le design, le développement, les tests et enfin l'évaluation du résultat. L'ensemble de ces étapes sont représentées dans le schéma ci-dessous :

Figure 8 Cycle Méthode Agile

De surcroît, afin d'organiser au mieux la gestion des tâches j'ai utilisé l'outil « Trello »[18]. Cet outil permet de répartir les tâches sur un « tableau » consacré à un projet. Plusieurs « listes » vont contenir des « cartes » qui, à leur tour, contiennent les tâches à accomplir. Ainsi, j'ai pu trier les cartes à l'aide de 5 listes :

- Sujets à aborder avec les formateurs ;
- À faire ;
- En cours de réalisation ;
- Bloqué ou bug ;
- Terminé.

J'ai ainsi pu trier les différentes tâches notamment grâce aux étiquettes et établir un ordre de priorité (cf. Stratégie MVP). Trello est particulièrement efficace car, il m'était possible de stocker des ressources et afficher des images (MCD, MLD, maquettes) qui pouvaient être consultées à tout moment en cas de besoin.

B. Langages et technologies

1. Langages utilisés

Pour réaliser la partie front-end du site j'ai utilisé les langages suivants : HTM 5, CSS 3 et JavaScript. L'HTML ou « *l'HyperText Markup Language* » est un langage dit de « balisage », il s'utilise afin d'intégrer le contenu des pages web.

Quant au CSS, il s'utilise pour appliquer une mise en forme du contenu. Le CSS travaille de pair avec le langage HTML et il peut directement s'ajouter à ce dernier. Cependant, il est

[18] La capture d'écran de mon tableau Trello pour ce projet est consultable dans les annexes.

préférable de consacrer un fichier uniquement au style visuel fait en CSS. Autrement dit, il est préférable de créer une « feuille de style » CSS pour chaque projet.

Enfin, le JavaScript, qui est un langage de script, s'utilise principalement pour réaliser des pages web interactives, animées et dynamiques. Il permet de réaliser par exemple des animations et peut très bien travailler de pair avec le langage CSS. Il est nécessaire de préciser que l'utilité de JavaScript ne se limite pas au visuel. En effet, il est possible de programmer en POO, de faire appel à des API, etc. et il est utilisé dans plusieurs environnements tels que Node.js.

PHP 7.4 (ou Hypertext Preprocessor) est un langage de programmation côté serveur, libre et orienté objet. Il est particulièrement apprécié dans le développement web. En effet, il permet de créer des pages web dynamiques.

Enfin, le langage SQL est un langage de requête qui permet de communiquer avec une base de données.

2. Technologies utilisées

Tout d'abord, pour développer mon projet j'ai utilisé « Visual Studio Code » qui est un éditeur de code développé par Microsoft.

« AdobeXD » est un logiciel qui offre la possibilité de créer des maquettes d'une application. Mais, il permet également de créer des prototypes interactifs de sites web. Je me suis fixé comme objectif d'utiliser ce logiciel durant ce projet car, je voulais découvrir et prendre en main cet outil. Pour créer les zonings j'ai également utilisé « Draw.io ».

Ensuite j'ai pu réaliser les modélisations de données MCD et MLD à l'aide de « Looping » qui est un logiciel gratuit et libre d'utilisation développé par l'Université de Toulouse.

Par la suite j'ai utilisé **Symfony 5**, qui est un puissant framework libre en PHP (cf. PHP). Le framework Symfony est un ensemble de composants PHP qui utilise le design pattern MVC. Il a été créé par *SensioLabs* et est particulièrement utilisé dans les projets de développement web. Le design pattern MVC est particulièrement efficace car, il sépare les différentes couches d'une application en 3 groupes distincts qui fonctionneront à l'unisson. Ainsi, la couche modèle est responsable des données, la vue est responsable de l'affichage et enfin le contrôleur jouent le rôle de chef d'orchestre et peut faire appel à la couche modèle et peut renvoyer une vue (cf. Développement du site).

« Composer » est un logiciel gestionnaire de dépendances libre écrit en PHP. Il permet d'installer des bibliothèques au besoin et selon les projets sur Symfony.

 « Twig » est un moteur de templates pour le langage de programmation PHP, il est utilisé par le framework Symfony. C'est grâce à Twig que je pouvais en partie gérer l'affichage des vues (cf. Développement du site).

 Doctrine est une ORM (*object relational mapping*) qui va me permettre de récupérer des informations de ma BDD sous forme d'objet. Plus précisément, Doctrine forme une couche d'abstraction entre Symfony et la base de données relationnelle. Doctrine est un logiciel libre et sous licence GNU LGPL. C'est également l'ORM par défaut de Symfony mais, son utilisation n'est pas obligatoire. Doctrine utilise le langage DQL (ou « *Doctrine Query Langage* ») qui est un langage de requête (cf. langage SQL) orienté objet et spécifique à Doctrine.

 Git est un logiciel de « gestion de versions décentralisé ». L'utilisation de ce logiciel est libre et il a été créé par Linus Torvaldsà.[19] J'ai utilisé Git ainsi que GitHub afin de mettre mes avancés en ligne. Ce logiciel requiert une certaine prise en main mais, il est particulièrement efficace lors des projets en équipe. De plus, grâce à quelques lignes de commande il est possible de rapidement mettre à jour le projet ou de créer de nouvelles « branches » de travail.

  J'ai utilisé « Font awesome » qui est une police d'écriture et un outil qui met à disposition plusieurs icônes gratuitement. Enfin, « google-font » est quant à lui un service d'hébergement gratuit de polices d'écritures.

 Enfin, pour m'aider durant le développent de la partie front-end j'ai également décidé d'utiliser Bootstrap 4.6. Bootstrap, un framework CSS, est une collection d'outils disponibles et utilisables dans un projet qui permet de travailler le design d'une application web. Néanmoins, afin de valider les compétences front-end requises j'ai développé la page d'accueil principal sans l'aide de cet outil.

C. Maquettes

1. Zoning

Dans un premier temps, c'est le travail de zoning qui a été réalisé et qui représente la première étape de réflexion. [20] Le zoning est le travail de maquettage le plus basique. En effet, son but principal est de déterminer les zones principales des différentes pages du site. En d'autres termes, le zoning se présente sous forme de schémas simples qui indiquent la place des différents éléments d'une page (comme celle du header, du menu, du footer, des différentes sections, etc.).

2. Wireframe

Le wireframe représente un travail de maquettage plus détaillé que le zoning. En effet, à ce stade peuvent s'ajouter les détails graphiques des pages, les couleurs, des textes, etc. Le

[19] Linus Benedict Torvalds, né le 28 décembre 1969 à Helsinki en Finlande, est un informaticien américano-finlandais (source Wikipédia).
[20] Le zoning de la page d'accueil se trouve dans les annexes.

wireframe a pour but de détailler la composition des pages tout en apportant des informations sur l'ergonomie et l'expérience utilisateur (ou autrement dit « l'UX » pour User eXperience).

3. Mock-up

Le mock-up représente le travail de maquettage le plus poussé.[21] Ce dernier se présente parfois sous forme d'un prototype constitué de plusieurs maquettes interactives. Ainsi, il est possible d'expérimenter l'enchainement entre les différentes pages, de vérifier la fluidité de la navigation et d'analyser précisément l'UX du projet.

D. Conceptualisation des données

1. MCD

Après avoir constitué le cahier des charges je me suis penchée sur la modélisation des données. La modélisation des données est l'analyse d'un système de données qui est représentée par une structure contenant un ensemble d'entités logiques et leurs relations.

Dans un premier temps, j'ai créé le MCD ou « modèle conceptuel de données ». Ce dernier représente l'un des outils majeurs de la méthodologie *Merise*.[22] Plus précisément, le MCD est une représentation graphique qui permet de comprendre l'ensemble d'un système de données.

J'ai ainsi dû créer plusieurs entités et les relier entre elles : Utilisateur, Equidé, Message, Annonce, Catégorie, Photo, Activité. Le MCD ci-dessous illustre la dynamique entre les différentes entités ainsi que leurs relations et cardinalités.

[21] Le mock-up de la page d'accueil que j'ai réalisé est consultable à l'adresse suivante : https://xd.adobe.com/view/fb266102-668d-4e30-809b-5675bb4bae93-ae75/

[22] « Merise est une méthode de conception, de développement et de réalisation de projets informatiques. La méthode MERISE date de 1978-1979, et fait suite à une consultation nationale lancée en 1977 par le ministère de l'Industrie dans le but de choisir des sociétés de conseil en informatique afin de définir une méthode de conception de systèmes d'information. » source : MERISE - Initiation à la conception de systèmes d'information, consulté le 22/06/2021, lien : https://web.maths.unsw.edu.au/~lafaye/CCM/merise/concintro.htm

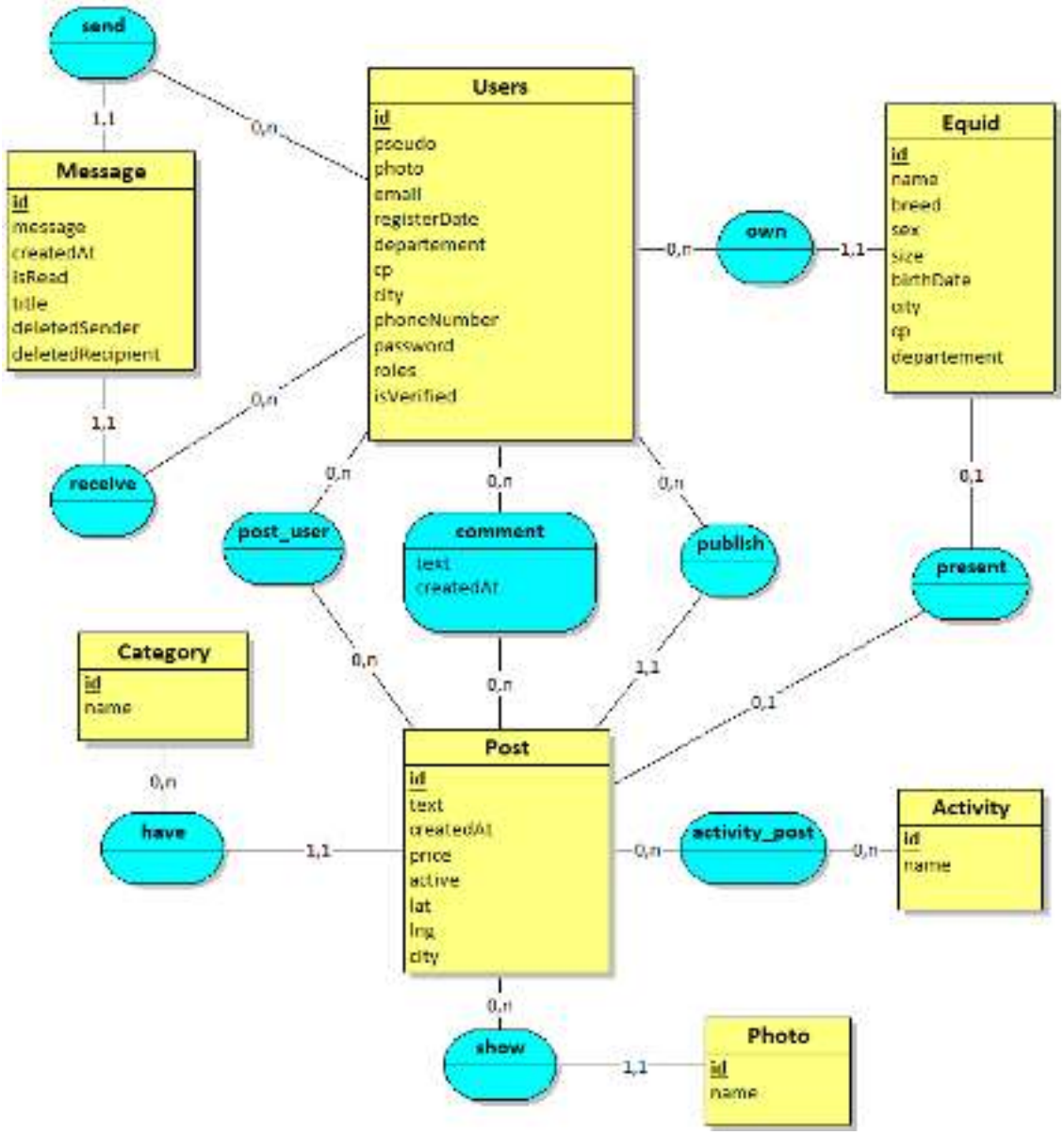

Figure 9 MCD du projet

2. MLD

Le MLD, ou « Modèle logique de données », découle d'un MCD. Dans les faits, on peut observer une transformation des entités en tables et l'apparition des clés étrangères grâce aux précédentes cardinalités du MCD.

Dans la figure du MLD ci-dessous il est possible d'observer l'apparition de nouvelles tables associatives comme « comment », dont la clé primaire est obtenue par association des clés étrangères.

La table associative « comment » a requis une attention toute particulière durant le développement sur *Symfony 5* lors de la création des entités. En effet, Symfony peut facilement gérer les tables associatives (relation ManyToMany) sans champs supplémentaires (comme pour les tables « post_user » et « activity_post »). Mais, dans le cas de la table « comment » j'ai d'abord dû créer l'entité ainsi que ses champs puis j'ai rajouté des champs de type « relation » « ManyToOne » vers l'entité utilisateur et annonce.

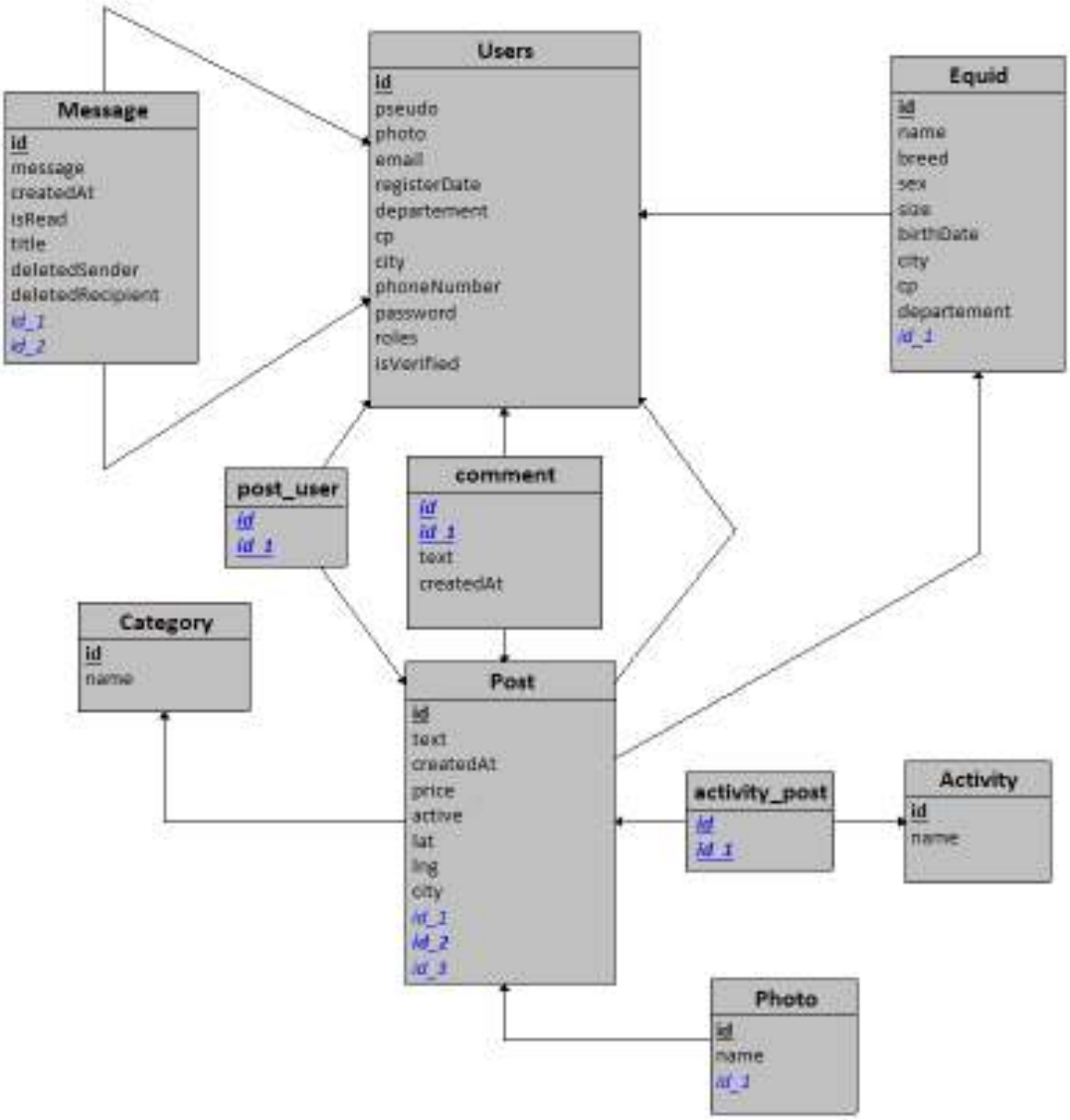

Figure 10 MLD du projet

E. Développement du site

Comme nous l'avons vu précédemment, *Symfony* utilise l'architecture logicielle MVC (pour « Modèle, Vue, Contrôleur »). En d'autres termes, la structure de l'application est divisée en plusieurs parties distinctes. Néanmoins, ces parties travaillent ensemble et communiquent entre elles durant le fonctionnement du site web. Ci-dessous un schéma explicatif du design pattern MVC :

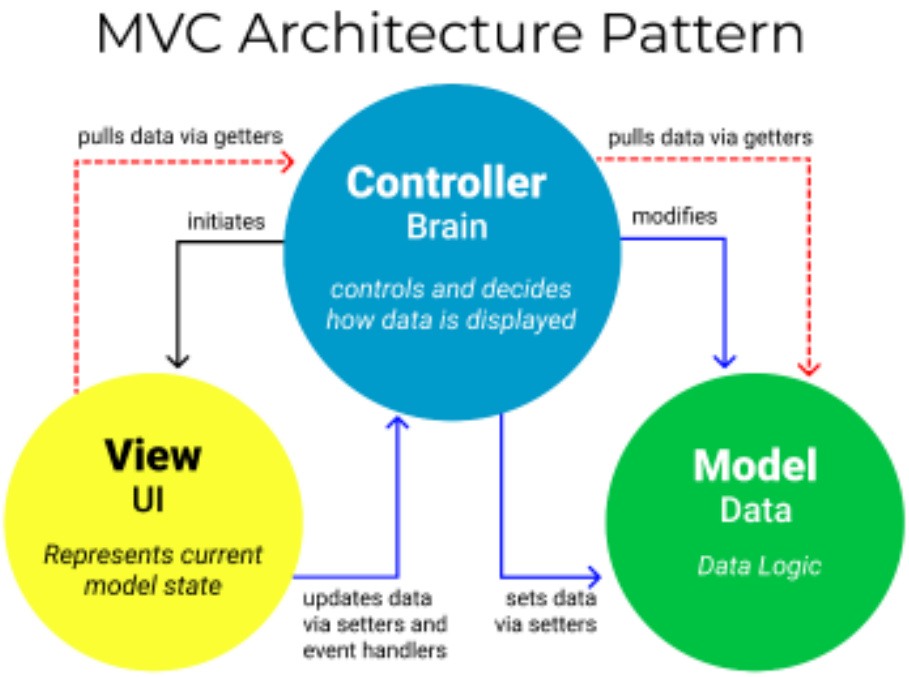

Figure 11 MVC Architecture Pattern[23]

1. Contrôleurs

Le contrôleur représente le chef d'orchestre de l'application et il est responsable de l'exécution des actions effectuées par l'utilisateur. Par ailleurs, les contrôleurs font le lien entre la couche « modèle » et les « vues » après exécution des fonctions requises par une action donnée de l'utilisateur.

Par exemple, un utilisateur souhaite consulter la page consacrée à ses annonces. Il clique sur le lien « mes annonces » dans la barre de navigation.

```
47    <a href="{{path('my_posts')}}">Mes annonces</a>
```

Figure 12 Code du lien "Mes annonces"

Cette action du client va engendrer une URL qui va appeler le système de routing. Ce dernier va ensuite appeler le bon contrôleur et par extension la bonne méthode selon l'URL demandée. Dans cet exemple, la route « my_posts » fait appel à la méthode « myPosts », dont le code est illustré ci-dessous :

[23] Schéma « MVC Architecture Pattern », consulté le 05/07/2021, source : "The Model View Controller Pattern – MVC Architecture and Frameworks Explained" *freeCodeCamp*, lien : https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-architecture-and-frameworks-explained/

```
39      /**
40       * @Route("/myPosts", name="my_posts")
41       */
42      public function myPost(): Response
43      {
44          $this->denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY');
45          // deny the access if the user is not completely authenticated
46
47          // get current user id
48          $userId = $this->getUser()->getId();
49
50          // get all user's posts
51          $posts = $this->getDoctrine()
52              ->getRepository(Post::class)
53              ->findBy(array('user' => $userId), null);
54
55          // get all user's horses
56          $horses = $this->getDoctrine()
57              ->getRepository(Equid::class)
58              ->findBy(array('user' => $userId), null);
59
60          return $this->render('post/myPost.html.twig', [
61              'posts' => $posts,
62              'horses' => $horses
63          ]);
64      }
```

Figure 13 Méthode myPost d'un contrôleur, extrait de code

Un contrôleur peut également interagir avec la couche modèle et peut renvoyer une vue comme c'est le cas dans l'exemple ci-dessus à la ligne 60. C'est ainsi que la requête est traitée par le contrôleur.

2. Modèle

La gestion des données est principalement réalisée grâce à *Doctrine*. *Doctrine* est l'ORM[24] utilisé par défaut par *Symfony 5*. L'ORM est un programme qui permet de faire interface entre *Symfony* et une base de données relationnelle pour simuler une base de données orientée objet.

Ainsi, nous retrouvons les « entités » (entities en anglais) qui représentent nos tables. Une entité possède un ou plusieurs « attributs » qui représentent un ensemble de caractéristiques. C'est le cas de l'entité « Equid » (équidé en anglais) qui nécessitait plusieurs attributs : le nom (name), la race (breed), la taille (size), etc.

[24] ORM : acronyme en anglais pour « object-relational mapping » ou « mapping objet relationnel » en français.

Figure 14 Attributs de l'entité "Equid", extrait de code

L'entity manager, un service Doctrine, permet de manipuler les entités. En somme, il permet l'insertion, la mise à jour ou la suppression de données. Enfin, le repository permet de faire des sélections grâce à des fonctions natives Symfony (comme `find()`, `findOneBy()`, `findAll()`, `findBy()`) ou il est également possible d'implémenter des fonctions personnalisées selon les besoins de l'application (cf. Fonctionnalité représentative).

Dans l'exemple précédent, il était nécessaire de récupérer l'ensemble des annonces de l'utilisateur grâce à la fonction `findBy()`, à la ligne de code 53 et 58 de la figure 13, pour pouvoir ensuite les afficher.

3. Vues

Le dossier « templates » détient toutes les vues qui se chargent du rendu visuel du contenu format HTML et des données côté client. Ces fichiers détiennent une double extension « .html » et « .twig ».

Pour rappel, Twig est un puissant moteur de templates utilisé par défaut par *Symfony 5*. Il est particulièrement efficace puisqu'il permet de gérer l'affichage des données à l'aide d'une syntaxe spécifique. En effet, il est possible d'utiliser l'héritage des templates (ligne 1 figure 15 ci-dessous) ou encore l'utilisation de « block » pour pouvoir découper efficacement le code (ligne 3, 6 et 9 figure 15) :

```
1   {% extends 'base.html.twig' %}
2
3   {% block title %}EmprunteMonPoney - Compte
4   {% endblock %}
5
6   {% block body %}
7     <div class="backgroundBlue">
8
9       {% block header %}
10        {% include "smallHeader.html.twig" %}
11      {% endblock %}
12
13      {% block container %}
14
15        <main
16          class="container-lg">
17
18          {# flash message #}
19          {% for message in app.flashes('message') %}
20            <div class="alert alert-success" role="alert">
21              {{ message }}
22            </div>
23          {% endfor %}
24
25          <h2 class="mt-2">Bienvenue
26            {{app.user.pseudo |capitalize }}
27          </h2>
```

Figure 15 Exemple d'une vue : le compte utilisateur

Il est possible de faire appel à des variables ou encore d'effectuer des commandes comme :

- Des conditions, comme dans l'exemple ci-dessous où le lien menant à l'espace administrateur s'affiche uniquement lorsque l'utilisateur connecté possède le rôle d'administrateur (« ROLE_ADMIN ») :

```
12          {% if is_granted("ROLE_ADMIN") %}
13
14            <a href="{{path('admin_home')}}">
15              Admin
16            </a>
```

Figure 16 Condition Twig

- Ou encore des boucles, comme dans l'exemple ci-dessous où la boucle permet d'afficher les annonces de l'utilisateurs. Ainsi, il est possible d'afficher pour chaque annonce (post) du tableau des annonces obtenu (posts) une disposition sous forme de carte :

```
41  {% for post in posts %}
```

Figure 17 Boucle « for » Twig

4. Fonctionnalité représentative

Le filtre est particulièrement important dans ce projet il est donc nécessaire d'expliquer son fonctionnement. Lorsque l'utilisateur clique sur le bouton « filtrer », ou lorsqu'il accède à la page des annonces, une requête est faite et envoyée au contrôleur frontal de Symfony. Le système de routing (ligne 32 de la figure ci-dessous) permet d'accéder au contrôleur approprié (ici PostController) et à la méthode adéquate :

```
29
30      /**
31       * @IsGranted("ROLE_USER")
32       * @Route("/posts", name="posts")
33       */
34      public function index(PostRepository $postRepository, Request $request): Response
35      {
36          $this->denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY');
37          // deny the access if the user is not completely authenticated
38
39          // data initialization
40          $data = new SearchData();
41
42          // paginator
43          $data->page = $request->get('page', 1);
44
45          // form creation
46          $form = $this->createForm(SearchType::class, $data);
47
48          $form->handleRequest($request);
49          // handleRequest() - read data off of the correct PHP superglobals (i.e. $_POST or $_GET)
50          // based on the HTTP method configured on the form (POST is default).
51
52          $posts = $postRepository->findSearch($data);
53
54          return $this->render('post/index.html.twig', [
55              'posts' => $posts,
56              'form' => $form->createView()
57          ]);
58      }
```

Figure 18 PostController

Les utilisateurs non-inscrits et non-connectés ne doivent pas accéder aux annonces et donc au filtre. Ainsi, une sécurisation a été ajoutée à la ligne 31 ainsi qu'à la ligne 36. `IsGranted` permet de vérifier l'accès à une méthode en fonction du rôle de l'utilisateur (ici `ROLE_USER`) tandis que, `denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY')` permet de vérifier si l'utilisateur est correctement authentifié.

Si l'utilisateur peut accéder à cette méthode alors, cette dernière va traiter la demande du client. Cette méthode permet de créer le filtre présent sur la page des annonces. Pour ce faire, je stocke dans la variable `$data` une instance de la classe `SearchData()` à la ligne 40. Cette classe va posséder plusieurs attributs comme `$q`, `$categories`, `$max`, `$min`, *etc.* :

```php
<?php

namespace App\Data;

use App\Entity\Activity;
use App\Entity\Category;


You, a day ago | 1 author (You)
class SearchData
{

    /**
     * @var int
     */
    public $page = 1;

    /**
     * @var string
     */
    public $q = '';

    /**
     * @var Category[]
     */
    public $categories = [];

    /**
     * @var Activity[]
     */
    public $activities = [];

    /**
     * @var null|float
     */
    public $max;
```

Figure 19 SearchData

Les attributs de cet objet seront nécessaires à la création du filtre à la ligne 46 (figure 18), grâce à `SearchType`. `SearchType` est la classe fille d'`AbstractType` qui permet de créer des formulaires dans Symfony. Le filtre est créé grâce aux fonctions `createForm()` à la ligne 46 et `createView()` dans le `return` à la ligne 56 (figure 18).

Ici la méthode d'envoi des données du formulaire est `GET` (par opposition à `POST` qui est normalement la méthode d'envoi par défaut). Ainsi, on peut constater que les données du formulaire (donc du filtre) sont empaquetées dans une chaîne dans l'URL de la page lorsque la requête est soumise comme ci-dessous :

```
/posts?q=&min=&max=750&distance=100&categories[]=2&activities[]=9&lat=48.573405383&lng=7.752111299999999
```

Ensuite, le contrôleur doit faire appel à la couche modèle pour récupérer les données appropriées. Grâce à l'ORM Doctrine il est possible d'interagir sur Symfony avec la base de données. Doctrine utilise le langage DQL et récupère des objets grâce à des requêtes :

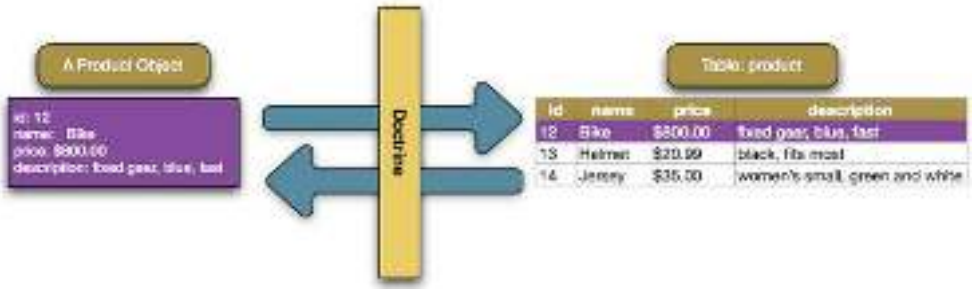

Figure 20 Doctrine et la BDD[25]

[25] Consulté le 17/09/2021, Source : documentation officielle de Symfony : symfony.com, lien :

Afin de récupérer les objets `Post` (annonce) correspondants aux critères du filtre j'ai créé une requête personnalisée DQL `findSearch()` dans le `PostRepository`, puisque les méthodes natives `find()`, `findOneBy()`, `findAll()`, `findBy()` étaient inadéquates dans ce cas précis.

Ainsi, le contrôleur fait appel à cette méthode `findSearch()` à la ligne 52 de la figure 18 et il est nécessaire d'injecter en paramètre la variable `$data`. En effet, les différentes données des inputs du filtre sont récupérées grâce à `handleRequest` (méthode `GET`). Quant à la méthode `findSearch()` elle se présente comme suit :

```
28      * Returns all posts per page
29      * @return PaginationInterface
30      */
31     public function findSearch(SearchData $search): PaginationInterface
32     {
33         $query = $this
34             ->createQueryBuilder('p')
35             ->select('c', 'p', 'a')
36             ->join('p.category', 'c')
37             ->leftJoin('p.activities', 'a')
38             ->where('p.active = 1');
39
40         if (!empty($search->q)) {
41             $query = $query
42                 ->andWhere('p.text LIKE :q')
43                 ->setParameter('q', "%{$search->q}%");
44         }
45
46         if (!empty($search->min)) {
47             $query = $query
48                 ->andWhere('p.price >= :minPrice')
49                 ->setParameter(':minPrice', $search->min);
50         }
51
52         if (!empty($search->max)) {
53             $query = $query
54                 ->andWhere('p.price <= :maxPrice')
55                 ->setParameter(':maxPrice', $search->max);
56         }
```

Figure 21 Méthode findSearch partie 1

Cette méthode va créer une requête (query) afin de récupérer les objets `Post` adéquates. Je récupère les attributs grâce à la variable `$search` passée en paramètre. Plusieurs conditions peuvent s'additionner dans cette requête. Par exemple, lorsque l'attribut `q` n'est pas vide (ligne 40 figure 21) j'ajoute une condition supplémentaire `andWhere()` (ligne 42). Il est nécessaire d'utiliser des *placeholders* comme `:q` puis, de définir sa valeur à l'aide de `setParameter`() afin de se prémunir contre l'injection SQL. Il est nécessaire de procéder ainsi pour l'ensemble des valeurs passées et utilisées dans la requête afin de se protéger correctement (cf. L'injection SQL).

https://symfony.com/doc/current/doctrine.html#querying-for-objects-the-repository

```
58      if (!empty($search->categories)) {
59          $query = $query
60              ->andWhere('c.id IN(:categories)')
61              ->setParameter(':categories', $search->categories);
62          }
63
64      if (!empty($search->activities)) {
65          $query = $query
66              ->andWhere('a.id IN(:activities)')
67              ->setParameter(':activities', $search->activities);
68          }
69
70      if (!empty($search->lat) && !empty($search->lng) && !empty($search->distance)) {
71          $query = $query
72              ->select('p')
73              ->andWhere('(6353 * 2 * ASIN(SQRT( POWER(SIN((p.lat - :lat) *
74              pi()/180 / 2), 2) + COS(p.lat
75              * pi()/180) * COS(:lat * pi()/180) *
76              POWER(SIN((p.lng - :lng) * pi()/180 / 2), 2) ))) <= :distance')
77              ->setParameter('lng', $search->lng)
78              ->setParameter('lat', $search->lat)
79              ->setParameter('distance', $search->distance);
80          }
81
82      $query = $query
83          ->addOrderBy('p.createdAt', 'DESC')
84          ->getQuery();
85
86      return $this->paginator->paginate(
87          $query,
88          $search->page,
89          9
90      );
91  }
```

Figure 22 Méthode findSearch partie 2

Afin d'effectuer la recherche selon une distance j'ai dû récupérer la latitude et la longitude d'une ville soumise par le client (ligne 70 à 79 figure 22). Pour ce faire, j'ai ajouté des champs supplémentaires dans le formulaire : un champ où l'utilisateur ajoute la ville de son choix (`input type text`) ainsi qu'un champ pour la distance souhaitée (`form.distance`) comme ci-dessous :

```
18  <div class="ml-5">
19    <h2 class="filterTitle" style="font-size: 1rem;">Votre ville</h2>
20    <input type="text" value="" placeholder="Ville" id="search_city"
21    class="form-control" style="margin-bottom:1rem;">
22    {{ form_row(form.distance) }}
23  </div>
```

Ensuite, j'ai utilisé l'API Geocoding[26] de Google pour récupérer la latitude et la longitude de la ville donnée. Une API, acronyme de *Application Programming Interface* en anglais, est un programme qui sert d'interface et permet ainsi à deux applications distinctes de communiquer et d'échanger des données ou des services. Voici ci-dessous l'extrait de code correspondant :

[26] https://developers.google.com/maps/documentation/geocoding/overview

```
1   // Google geocode
2   window.onload = () => {
3     let cityInput = document.getElementById("search_city");
4     // get input city
5     cityInput.addEventListener("change", geocode);
6   };
7
8   function geocode() {
9     let city = document.querySelector("#search_city").value;
10    // get the city name (value of input #search_city)
11    // console.log(city);
12
13    axios
14      .get("https://maps.googleapis.com/maps/api/geocode/json?", {
15        params: {
16          address: city,
17          key: "AIzaSyBtck3NP2KMzE3bOv6Z2rhl-HLtozb-UZ0",
18        },
19      })
20      .then(function (response) {
21        // full data response
22        // console.log(response.data);
23
24        // get latitude and longitude
25        let lat = response.data.results[0].geometry.location.lat;
26        let lng = response.data.results[0].geometry.location.lng;
27        // console.log(lng);
28
29        // set the right latitude and longitude in inputs type hidden
30        document.querySelector("#lat").value = lat;
31        document.querySelector("#lng").value = lng;
32      })
33      .catch(function (error) {
34        console.log(error);
35      });
36  }
```

Figure 23 API geocoding Google

Dans un premier temps, il est nécessaire « d'écouter » un changement effectué dans l'input (ligne 3 et 5). Lorsque le client entre une ville, cela déclenche la fonction `geocode`. Je stocke à ce moment la valeur de l'input dans la variable `city` à la ligne 9. Ensuite, j'ai décidé d'utiliser Axios[27] pour faire appel à l'API (équivalent de `fetch`). J'injecte la route correspondante de l'API à la ligne 14 et j'injecte également les paramètres nécessaires comme la variable `city` ainsi que la clé (lignes 16 et 17). Cela retourne une promesse que je traite à l'aide de `.then` à la ligne 20.

La réponse contient un tableau complexe au format JSON. Le JSON, ou *JavaScript Object Notation*, est un format de données textuelles qui permet de représenter un ensemble d'information structuré. Comme son nom l'indique, le JSON provient de l'écriture objet en Javascript et se présente sous forme de paires « nom/valeur » (ou clé/valeur).

Ainsi, je récupère les données qui me sont utiles (à la ligne 25 et 26) puis, je les injecte dans les inputs type hidden du formulaire (ligne 30 et 31). Voici les deux champs cachés que le client n'aperçoit donc pas à travers son navigateur :

[27] https://github.com/axios/axios

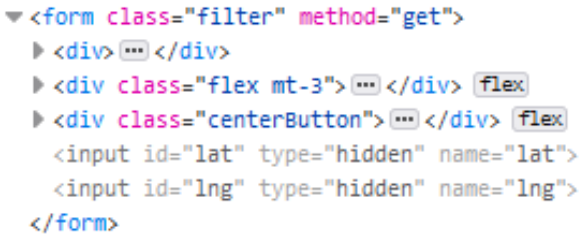

Figure 24 Champs hidden, aperçu grâce à l'outil inspecteur du navigateur

C'est ainsi que, comme toutes les valeurs des inputs visibles du formulaire, la latitude et la longitude seront utilisées par la méthode `findSearch()`. Une fois cette méthode exécutée le contrôleur renvoie une vue au format HTML ainsi qu'un tableau de données contenant les annonces appropriées ligne 61 ci-dessous :

Enfin, grâce à Twig il sera possible de faire une boucle pour afficher les annonces (posts) sur la page (ligne 51) et de renvoyer la vue à l'utilisateur :

Figure 25 Page des annonces

Dans ce chapitre, la dynamique entre les différentes couches du design pattern MVC de Symfony a été particulièrement soulignée. Le contrôleur, la couche modèle et les vues œuvrent ensemble dans le bon fonctionnement de l'application web.

F. Sécurité

L'utilisation de *Symfony* comporte de nombreux avantages : l'architecture MVC, le moteur de templates *Twig*, *Doctrine*, etc. Mais, les nombreux dispositifs de sécurisation natifs sont certainement l'un des atouts les plus intéressants de ce framework. En effet, *Symfony 5* offre une protection native contre les attaques les plus fréquentes du web.

1. La faille XSS

Une faille XSS correspond à l'injection de codes indésirables dans un ou plusieurs champs externes d'un site web. Elle peut entrainer la modification du code HTML, le vol de cookies, ou l'exécution du code malveillant interprétable côté navigateur.

Il est nécessaire de se prémunir de cette faille grâce à la sécurisation de tous les champs externes (formulaires de connexion ou d'inscription, formulaire de contact, paramètres d'URLs, etc.). Ainsi, il est préférable que l'ensemble des inputs de l'application soient filtrés.

Il est également possible sur *Symfony 5* de mettre en place des « contraintes ». Par exemple, si un nombre est attendu dans un formulaire lors de l'inscription d'un cheval, il est possible de vérifier que la donnée fournie par l'utilisateur correspond bien à un nombre (type « integer ») :

```
->add('size', IntegerType::class, [
    'required' => true,
    'label' => 'Taille',
    'attr' => [
        'class' => 'form-control'
    ]
])
```

Figure 26 Type de donnée « IntegerType » dans EquidType.php

Par ailleurs, comme indiqué dans la documentation de Symfony,[28] le moteur de templates *Twig* permet d'échapper automatiquement les données en sortie. L'échappement des données prodigue une défense supplémentaire contre la faille XSS. Ainsi, les caractères possédant une signification spéciale comme les balises <> peuvent par exemple être remplacées par `&lt,` ou leur interprétation peut être désactivée, ce qui va empêcher l'exécution de scripts malicieux.

2. La faille CSRF

La faille CSRF (pour Cross-Site Request Forgery en anglais) a pour but d'exploiter un utilisateur authentifié afin d'exécuter des actions malveillantes ou indésirables. En effet, le but de cette faille est de transmettre à un utilisateur authentifié une requête HTTP falsifiée qui sera exécutée à l'insu de ce dernier.

Cette faille exploite ainsi un utilisateur "complice" qui exécute l'action malveillante. Pour se prémunir de cette faille, il est recommandé d'utiliser un jeton (ou en anglais "token") qui sera généré, stocké en session et vérifié lorsque l'utilisateur souhaite réaliser une action sur le site.

En effet, la session est un moyen de conserver des données sur l'ensemble des pages d'un site dès l'instant ou un identifiant de session unique est renseigné. La session est particulièrement utile pour personnaliser et dynamiser les pages en fonction de l'utilisateur

[28] Twig output escaping, consulté le 13/07/2021, source : https://symfony.com/doc/current/templates.html#output-escaping

mais, dans ce cas elle peut stocker le token CSRF et renforcer la sécurité du site. Symfony fournit un objet session et plusieurs fonctionnalités qu'il est possible d'utiliser pour stocker des informations.

Symfony prend en charge nativement la faille CSRF comme, par exemple, dans le cas de l'authentification d'un utilisateur (UserAuthenticator.php et login.html.twig) :

Figure 27 Le token CSRF du formulaire de connexion

3. L'injection SQL

L'injection SQL correspond elle aussi à l'injection de codes indésirables dans une entrée utilisateur. Contrairement à la faille XSS, l'injection SQL consiste à injecter des requêtes SQL qui peuvent provoquer le vol ou la perte de données. Heureusement, Doctrine dispose d'une protection native contre cette faille. En effet, les requêtes DQL sont nativement "paramétrées" et sont donc protégées des injections SQL.

Cependant, lors d'une utilisation plus personnalisée de DQL il est nécessaire de toujours s'assurer que les valeurs provenant des inputs externes soient bien considérées comme des « placeholders » (utilisation de « setParameter »).[29] C'est le cas dans la figure ci-dessous à la ligne 41 où j'utilise le placeholder ':q'. Puis, à la ligne 42 j'assigne à ce dernier la valeur de `q` de `$search` qui a été passé en paramètre à la ligne 31 :

Figure 28 Exemple setParameter contre l'injection SQL, extrait de code

[29] "You should always use prepared statements to execute your queries." Consulté le 15/07/2021, lien : https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/security.html

4. Veille sur la sécurité

Durant ma veille sur la sécurité j'ai principalement consulté le site OWASP.[30] J'ai ainsi pu me renseigner sur plusieurs types de failles informatiques. J'ai notamment pris connaissance des failles suivantes :

- L'attaque par « flooding » (raz-de-marée en français), vise à inonder un service de requêtes au point de le rendre indisponible. Il est possible de se prémunir de cette attaque grâce au « captcha ». Ce dernier est également utile pour se prémunir du « spam ».

  Le captcha[31] est un outil qui permet de bloquer l'activité de robots (« bots » en anglais) qui peuvent par exemple envoyer plusieurs requêtes en peu de temps. Dans le cadre de ce projet j'ai implémenté le *reCAPTCHA v3* de *Google* au niveau du formulaire de contact. En temps normal, le captcha prend la forme de tâches à réaliser par l'utilisateur (trouver des images spécifiques, écrire des lettres et des chiffres inscrits sur une image, une checkbox à cocher, etc.).

  La version de ce captcha est invisible aux yeux des utilisateurs qui ne seront donc pas interrompus. Ce captcha renvoie un score selon l'activité réalisée sur le site et identifie donc l'utilisateur comme étant un bot ou non (1.0 est un bon score, 0.0 pour un bot).[32] Si l'utilisateur n'est pas un bot un token est attribué à l'input caché (type hidden) du formulaire de contact :

```
56          <input type="hidden" id="recaptchaResponse" name="recaptcha-response">
57
58          {{ form_end(form) }}
```

```
72  {# Google reCaptcha #}
73
74  <script src="https://www.google.com/recaptcha/api.js?render=6LcBGIIcAAAAAAyFLAdwNcsUTcQ4bFw-_Ra2aRK4"></script>
75  <script>
76      grecaptcha.ready(function() {
77        grecaptcha.execute('6LcBGIIcAAAAAAyFLAdwNcsUTcQ4bFw-_Ra2aRK4', {action: 'submit'}).then(function(token) {
78            document.getElementById("recaptchaResponse").value = token
79        });
80      });
81  </script>
```

Figure 29 reCAPTCHA v3 de Google, extrait de code

Le token est ensuite récupéré et utilisé dans le traitement du formulaire dans le contrôleur afin de permettre, ou non, l'envoi du mail :

```
46          // checking the captcha
47          if(empty($_POST['recaptcha-response'])){
48
49              // if recaptcha token is empty redirect to the home page
50              // first protection against bots
51              return $this->redirectToRoute('home');
52          } else {
```

Figure 30 reCAPTCHA v3 traitement, extrait de code

[30] Consulté le 14/03/2021, Lien : https://owasp.org/
[31] Captcha l'acronyme de : 'completely automated public Turing test to tell computers and humans apart'
[32] Consulté le 08/09/2021, https://developers.google.com/recaptcha/docs/v3

- La faille upload est malheureusement possible lorsque les utilisateurs ont la possibilité de télécharger un fichier sur le site (photo de profil, document PDF, images, etc.). Voici les mesures que j'ai utilisé :
  - J'ai généré un nom aléatoire pour le fichier uploadé et enregistré le nom dans la base de données ;
  - J'ai ajouté dans « constraints » du formulaire `PostType.php` une contrainte « `new Image()` » pour m'assurer que les fichiers téléchargés par l'utilisateur sont bel et bien des images. La taille de l'image est également fixée à 8 Mo. Enfin, j'ai rajouté un `mimeTypes` pour m'assurer que le format des images soit de type png, jpeg, jpg, ou webp.

```
47          // upload pictures, but will not be linked with DB (mapped=false)
48          ->add('photo', FileType::class, [
49              'label' => false,
50              'multiple' => true,
51              'mapped' => false,
52              'required' => false,
53              'constraints' => [
54                  new All([
55                      new Image([
56                          'maxSize' => '8M',
57                          'maxSizeMessage' => 'La taille de l\'image est trop grande.
58                          La taille maximale autorisée est de {{ limit }} {{ suffix }}.',
59                      ]),
60                      new File([
61                          'mimeTypes' => [
62                              'image/png',
63                              'image/jpeg',
64                              'image/jpg',
65                              'image/webp',
66                          ],
67                          'mimeTypesMessage' => 'Format invalide ({{ type }}).
68                          Les formats autorisés sont : {{ types }}.',
69                      ])
```

Figure 31 Contraintes pour l'upload d'image dans PostType.php, extrait de code

- L'Attaque par force brute, qui est assez répandue et redoutable, consiste à trouver le mot de passe d'un utilisateur grâce à des outils et techniques spécifiques. Plusieurs solutions sont possibles :

  - Appliquer une « **politique de mots de passe forts** » en utilisant par exemple les expressions régulières (regex). Ainsi, l'utilisateur devra choisir un mot de passe contenant par exemple une majuscule, un certain nombre de caractère, etc. Cependant, il est nécessaire de trouver un équilibre entre une sécurisation forte et la commodité des utilisateurs.

    Dans le cadre de ce projet j'ai décidé d'utiliser une contrainte regex imposant à l'utilisateur un mot de passe d'au moins huit caractères dont des lettres majuscules et minuscules, un chiffre et un symbole (@#$%-) comme l'illustre la figure ci-dessous :

```
116         new Regex([
117             'pattern' => "/^(?=.*\d)(?=.*[A-Z])(?=.*[@#$%-])(?!.*(.)\1{2}).*[a-z]/m",
118             'match' => true,
119             'message' => "Attention : Votre mot de passe doit comporter au moins huit
120             caractères, dont des lettres majuscules et minuscules, un chiffre et un symbole."
121         ])
122     ],
```

Figure 32 Une Regex (constraints) pour renforcer le mot de passe

- Par ailleurs, le *reCAPTCHA v3* peut également contrecarrer l'éventuel bot s'il est implémenté au niveau du formulaire de connexion.

- Il est également important de sécuriser l'accès aux méthodes sensibles. Par exemple, un utilisateur banal ne doit pas avoir accès à l'espace administrateur auquel cas les conséquences peuvent être dévastatrices.

  En effet, l'accès aux différentes pages et méthodes se déroule en deux temps : **l'authentification et l'autorisation**. L'authentification vérifie « l'identité » de l'utilisateur (à l'aide de l'identifiant unique et du mot de passe à travers le login). L'autorisation quant à elle consiste à accorder la permission à un utilisateur d'accéder ou non à une fonctionnalité, une méthode, etc. Par exemple, j'ai sécurisé l'accès à l'ensemble des méthodes réservées aux administrateurs (possédant ainsi le « ROLE_ADMIN ») directement dans le contrôleur (ligne 20) :

```
19  /**
20   * @IsGranted("ROLE_ADMIN")
21   * Require ROLE_ADMIN for every controller methods in this class
22   */
    You, 2 weeks ago | 1 author (You)
23  class AdminController extends AbstractController
24  {
25      /**
26       * @Route("/admin", name="admin_home")
27       */
28      public function adminPanel(): Response
```

Figure 33 Accès sécurisé aux méthodes de l'AdminController

Il est également possible de configurer les routes réservées à certains rôles dans le fichier `security.yaml` comme ci-dessous :

```
36      access_control:
37          - { path: ^/admin, roles: ROLE_ADMIN }
```

Figure 34 Configuration des accès dans security.yaml

VI. Compétences transversales

A. Techniques de veille informationnelle

Une veille informationnelle est une bonne pratique pour les professionnels qui souhaitent renouveler régulièrement leurs connaissances face à l'évolution constante des technologies.

Premièrement, les *documentations officielles* représentent bien évidemment des sources fiables qu'il est préférable de consulter dès que nécessaire : https://symfony.com ou https://www.doctrine-project.org et sans oublier également le site SymfonyCasts : https://symfonycasts.com

Par ailleurs, il est également possible de rejoindre des communautés tels que *StackOverflow* (forum), *Symfony-friendly* (serveur *Discord* d'utilisateurs du framework *Symfony*), *Slack* (messagerie instantanée, dans mon cas j'ai rejoint *Slack Symfony*) etc. L'intérêt de ces communautés réside dans l'entraide et dans les échanges entre les professionnels et passionnés du développement.

Enfin, il est possible de se former à l'aide de plusieurs sites qui proposent des nouveaux cours régulièrement (*Coursera*, *Udemy*, *Openclassrooms*, etc.). Par ailleurs, *YouTube* représente une source d'information intéressante puisqu'il est possible de se former à l'aide d'un support vidéo (j'ai consulté *freeCodeCamp*, *Traversy Media*, *Lior Chamla*, *Grafikart, etc.*).

B. Utilisation de l'anglais

Dans le cadre de ce projet j'ai dû chercher des informations pour pouvoir envoyer des mails en utilisant Symfony. J'ai donc décidé de consacrer la traduction sur cette thématique. J'ai écrit dans ma barre de recherche « send email symfony ». J'ai obtenu comme premier résultat le site de la documentation officielle de Symfony qui représente une source fiable :

Figure 35 Résultat de la recherche "send email symfony"

Extrait du résultat de la recherche[33] :

> "Symfony's Mailer & Mime components form a *powerful* system for creating and sending emails - complete with support for multipart messages, Twig integration, CSS inlining, file attachments and a lot more. Get them installed with:
>
> ```
> composer require symfony/mailer.
> ```
>
> Emails are delivered via a "transport". Out of the box, you can deliver emails over SMTP by configuring the DSN in your `.env` file (the `user`, `pass` and `port` parameters are optional):
>
> ```
> # .env
> MAILER_DSN=smtp://user:pass@smtp.example.com:port
> ```

[33] Source : https://symfony.com/doc/current/mailer.html consulté le 30/05/2021.

> Instead of using your own SMTP server or sendmail binary, you can send emails via a 3rd party provider. Mailer supports several - install whichever you want: […] Each library includes a Symfony Flex recipe that will add a configuration example to your `.env` file. For example, suppose you want to use SendGrid. First, install it:
>
> ```
> composer require symfony/sendgrid-mailer
> ```
>
> You'll now have a new line in your `.env` file that you can uncomment."

Traduction française :

Les composants Symfony « Mailer & Mime » forment un puissant système pour la création et l'envoi de mails – ainsi que pour la prise en charge des messages multiparts, de l'intégration Twig, de l'inclusion CSS, des pièces jointes et bien plus encore. Installez-les avec :

```
composer require symfony/mailer.
```

Les mails sont envoyés via un « transport ». Dès que son installation est achevée, vous pouvez envoyer des mails grâce au SMTP en configurant le DSN dans votre fichier `.env` (les paramètres `user`, `pass` et `port` sont facultatifs) :

```
# .env
```

```
MAILER_DSN=smtp://user:pass@smtp.example.com:port
```

Plutôt qu'utiliser votre propre serveur SMTP ou sendmail binary, vous pouvez envoyer des mails via un fournisseur tiers. « Mailer » peut en supporter plusieurs - installez celui que vous voulez : […] Chaque bibliothèque inclut une formule Symfony Flex qui ajoutera une configuration à votre fichier `.env`. Par exemple, supposons que vous souhaitiez utiliser SendGrid. Premièrement, installez-le : `composer require symfony/sendgrid-mailer`

Vous aurez désormais une nouvelle ligne dans votre fichier `.env` que vous pouvez décommenter.

VII. Conclusion et perspectives d'évolution

Pour rappel, le but du projet était de développer un site permettant de mettre en relation des propriétaires de chevaux et des emprunteurs cavaliers. Les fonctionnalités requises par le cahier des charges sont disponibles et fonctionnelles mais, le site peut encore être amélioré.

Par exemple, j'aimerais ajouter la fonctionnalité « photo de profil » ou encore améliorer le design sur l'ensemble des pages du site.

Il serait également intéressant d'ajouter des catégories supplémentaires permettant aux utilisateurs d'ajouter d'autres annonces (location de prés, service de cours d'équitation individuel, location de matériel équestre, etc.).

Par ailleurs, ce projet m'a permis de renforcer mes connaissances en back-end, front-end et surtout en Symfony. J'ai également pu découvrir tout le processus de la gestion d'un projet dont j'ai particulièrement apprécié l'ensemble des aspects.

Les délais que je me suis initialement fixés n'ont pas tous été respectés car, ils étaient trop ambitieux et les fonctionnalités ont dû être adaptées à la suite des premiers tests d'utilisation.

J'ai également dû faire face à plusieurs difficultés mais, le soutien de mes formateurs a été particulièrement précieux. Je profite de cet interlude pour les remercier une nouvelle fois.

Je réalise désormais toute l'ampleur que représente un projet web et je suis heureuse d'avoir acquis cette expérience. Je souhaite désormais poursuivre mon parcours dans le développement web et j'aimerais me tourner progressivement vers le domaine de la gestion de projet.

VIII. Annexes

A. Schémas

Figure 36 Psychologie des couleurs n.1

Source : https://www.pcgprint.com/fr/psychologie_couleurs_marketing/

Figure 37 Psychologie des couleurs n.2

Source : https://c-marketing.eu/couleur-differenciation-marketing/

B. Maquettage

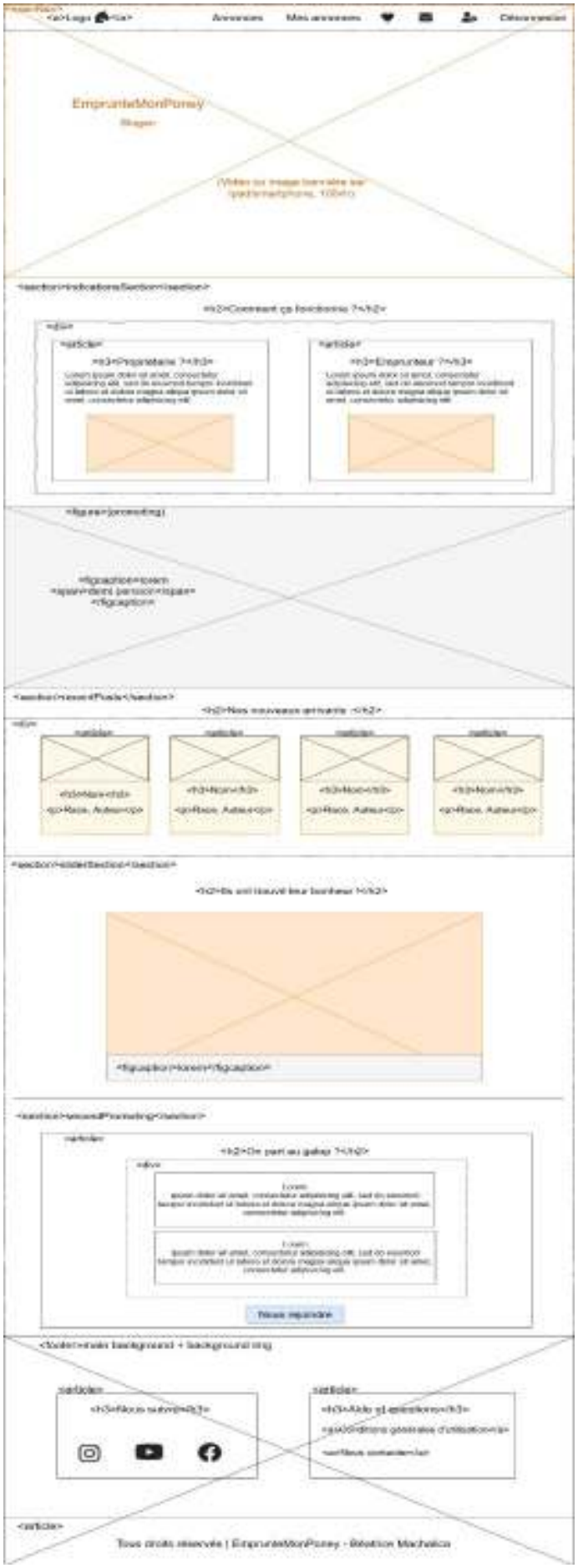
EmprunteMonPoney
Nos nouveaux arrivants
Tous droits réservés | EmprunteMonPoney - Béatrice Machalica

Figure 38 Maquette de la page d'accueil

C. Capture d'écran

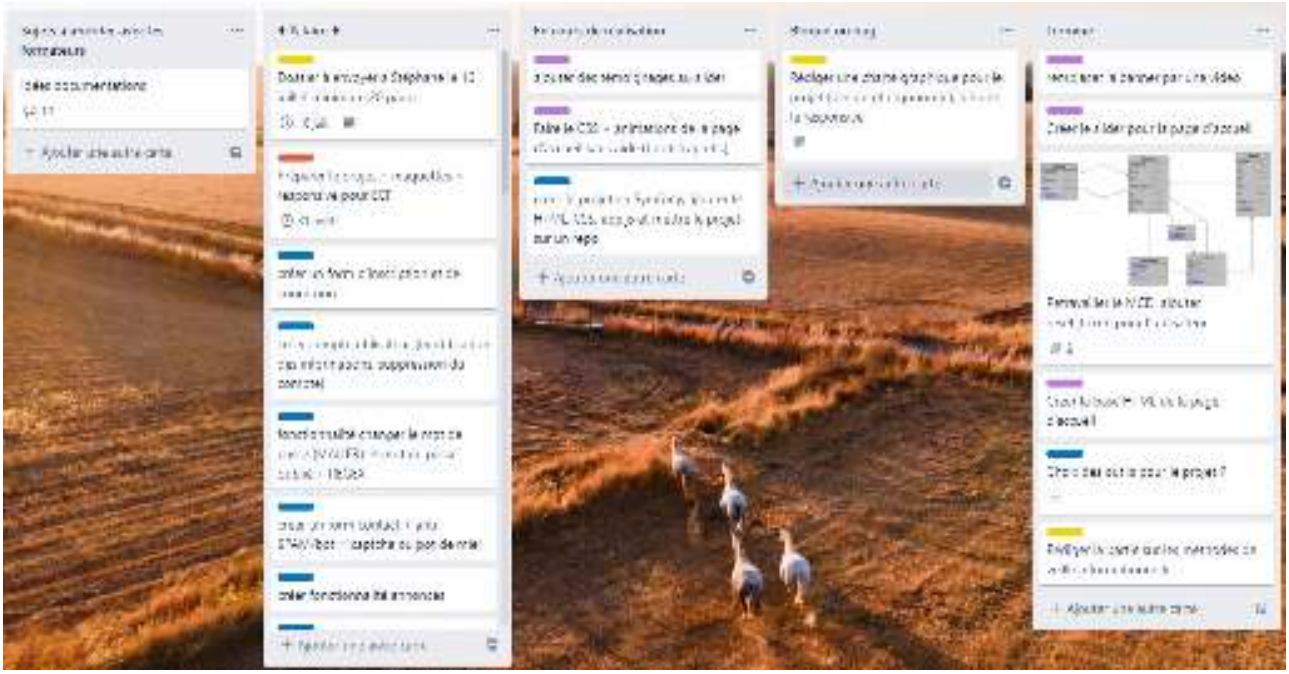

Figure 39 Tableau Trello