

***TITRE PROFESSIONNEL DWWM***

DEVELOPPEUR WEB ET WEB MOBILE



***VETOTOIL***

regnier sylvain

***Sommaire***

1. Compétence du référentiel couverte
2. Présentation du Projet
3. Spécifications Techniques
4. Cahier des charges
5. Compétences transversales et méthode agile
6. Architecture du projet
7. Réalisation de la partie Front-End

6.1- Wireframes et Maquette

6.2- UML

6.3- Développement des fonctionnalités

6.4- Rendu final

1. Réalisation de la partie Back-End

7.1-Base de données

7.1-Modélisation(Merise)

7.1-Création de la BDD et des requêtes

7.4-Développement

1. Déploiement du site
2. Conclusion

**01| Compétence du référentiel couverte**

*1.1 Développer la partie Front-End d’une application web ou web mobile en intégrant les recommandations de sécurité.*

• LES MAQUETTES

• REALISER UNE INTERFACE UTILISATEUR WEB OU MOBILE STATIQUE ET ADAPTABLE

• DÉVELOPPER UNE INTERFACE UTILISATEUR WEB OU MOBILE DYNAMIQUE

Pour la réalisation du site web « VETOTOIL », j’ai créé des maquettes afin de développer le site. Ensuite, elles ont été rendue dynamique et fonctionnelle en suivant les bonnes pratiques de développement.

*1.2 Développer la partie Back-End d'une application Web ou Web-mobile en intégrant les recommandations de sécurité.*

-En développement la partie back-end du site, j’ai intégré les mesures nécessaires de sécurité pour la protection des données ainsi que des attaques malveillantes comme par exemple en vérifiant chaque donnée insérée par l’utilisateur (injection SQL, injection de script …), mais aussi en me servant de UuidV7 unique, pour rendre chaque ID aléatoire au lieu d’avoir des chiffres qui se suivent, captchaV3 de Google pour éviter les inscriptions des robots.

-J’ai aussi fait le contrôle sur l’accès au donnée et aux fonctionnalité du site via le système des rôles.

-La partie back-end présenté est faite sous Symfony mais une version en MVC a aussi été faite.

-Une partie envoi de mail à l’administrateur est aussi proposé, cela a été fait avec PHPMailer

*1.3 Compétence Transversale*

• MÉTHODE AGILE (AGILE SOLO) : Dans le cadre de la réalisation de VETOTOIL j’ai appliqué à moi-même la méthode AGILE, je m’appliquais des taches journalières, des étapes et la durée de chacune des étapes pour la conception et la réalisation.

**Présentation du projet**

VETOTOIL est une application qui a pour but de pouvoir prendre des rendez-vous chez le professionnel de votre choix, vétérinaire, toiletteur (liste non exhaustive) sans avoir à rechercher le professionnel sur internet puis de le contacter ou par moment il n’y a pas de réponse à l’appel et nous devons réitérer l’appel, puis ensuite une fois les avoir au téléphone de pouvoir trouver une date et une heure qui conviennent à tout le monde.

VETOTOIL a pour but de facilité cela car après l’inscription sur le site, l’utilisateur pourra rechercher les professionnels qui sont inscrit dans son secteur, et en choisissant son domaine d’activité et ainsi de voir les premières disponibilités pour une petite urgence mais aussi la possibilité de voir par professionnel ses disponibilités au dates souhaité, cela permet d’être devant son téléphone ou son ordinateur et de voir directement les rdv disponibles. Le patient pourra aussi voir ses rdv en cours et ses rdv passés ainsi que le suivi de traitement s’il y en a pour son animal. Il pourra aussi ajouter des animaux a sa fiche, pour le moment chien ou chat avec une liste de race pour chacun d’entre eux prédéfini.

Pour des raisons de sécurité a société quand a elle pourra s’inscrire, mais il devra forcément avoir un Siret valide et sera soumis à validation par l’administrateur du site. Ensuite il pourra rattacher des employés déjà crée ou en crée des nouveaux, tous cela sera de la responsabilité de la société.

*Le but principal de VETOTOIL est de faire un Doctolib mais pour nos animaux*

**Spécifications Technique**

*Général :*

- Editeur de code : Visual Studio Code

-extension utilisées :

\*Composer for Visual Studio Code : permet de de faire des commandes rapides pour composer

\* IntelliPHP for Visual Studio Code : permet de travailler plus rapidement car l’IA essaie de prédire la suite du code

\*Live Server for Visual Studio Code : permet d’ouvrir une page HTML ou JavaScript en local

\*Prettier Formatter for Visual Studio Code : formatage du code

\*Twig Formatter for Visual Studio Code : formatage du code pour twig

- Outil de versionning : GitHub

- Maquettages : Figma

- Modèle conceptuel et logique de donnée : Looping

- Serveur php : Xampp

- Modele physique de données : Interface visuelle de PhpMyAdmin

**Projet VETOTOIL(version 2)**

-Language et balisage : HTML,Bootstrap et CSS

-Language de programmation : PHP, Javascript et twig

-Fait sous Symfony

Librairie :

-PHPMailer pour l’envoie de mail (‘https://github.com/PHPMailer/PHPMailer’)

-VichUploaderBundle pour insérer des images(ou fichier) (‘https://symfony.com/doc/current/controller/upload\_file.html’)

-Karser pour faire le RecaptchaV3 de google pour sécuriser les envoie de mail et d’inscription (‘https://github.com/karser/KarserRecaptcha3Bundle’)

-UuidV7 (fonction unique) pour générer des id aléatoire et « unique » pour certaine base de donnée sensible.(‘ https://symfony.com/doc/current/components/uid.html#working-with-uuids’)

- password-hasher pour crypter un mot de passe pour qu’il ne soit pas récupéré(‘https://symfony.com/doc/current/security/passwords.html’)

**Cahier des charges**

*Objectif :*

-Etablir un programme utile pour simplifier la prise de rdv chez les professionnels pour les animaux.

Pour les Professionnels :

-Gagner du temps, car il y aura moins d'appels téléphoniques pour la prise/modification/annulation de rendez-vous.

- Possibilité de ne plus avoir de créneaux libres dans leurs plannings.

-Pour les Particuliers :

-Ne plus attendre d'avoir des dates fournies par le professionnel et de vérifier ses propres disponibilités, car les disponibilités seront disponibles immédiatement sur le site internet.

-Avoir un suivi des rendez-vous et chez quel professionnel, ainsi que les traitements dans l'historique.

-Il doit être responsive et s’adapter aux différente taille d’écran (smartphone, tablette, pc)

-Le site aura aussi bien du Back-End et du Front-end

*Fonctionnalités principales :*

*Pour tous :*

Partie inscription users ou tout le monde s’inscrira avec les mêmes information demandé (nom, prénom, email(ne peux avoir deux emails identique), password) ensuite à la connexion ils seront redirigés vers une autre page des complément d’information.

*Pour les Sociétés :*

-création d’une fiche société (information demandé Siret, nom de la société, profession de la société, adresse de la société, complément adresse de la société, code postal de la société, ville de la société, téléphone de la société, téléphone du dirigeant, image société, date de création, date de résiliation, date de validation)

-Doit pouvoir crée un employé (fiche user normale)

-Doit pouvoir rechercher un employé par son email s’il est déjà inscrit

-Doit pouvoir ajouter un employé a son équipe et définir les jours travaillé ainsi que la pause repas et les vacances

-Sur le compte de la société doit pouvoir voir les informations de ses employés ainsi que leur planning, doit pouvoir aussi retirer un employé.

*Pour les employés :*

-Doit pouvoir compléter sa fiche (adresse employer, complément adresse employer, code postal employer, ville employer, téléphone employer, profession employer, image employer, date de création employer)

-Ne peux être rechercher que si la partie précédente est faite

-Doit pouvoir voir ses rdv du jour et de la semaine en fonction de chaque heure et de voir la fiche du patient et animal en fonction de ses rdv (consultation uniquement), doit pouvoir prendre un rdv en « live », possibilité d’annuler un rdv, possibilité, valider ou annuler le rdv.

-Doit pouvoir inscrire pour les patients les médicaments et la posologie

*Pour les particuliers :*

-Doit pouvoir compléter sa fiche patient (adresse patient, complément adresse patient, code postal patient, ville patient, téléphone patient, image, date de création, date fin patient)

-Doit pouvoir inscrire un ou plusieurs animal (prénom, date de naissance, type, race)

-Rechercher tous les professionnels suivant leurs disponibilités dans une zone géographique de 10km

-Prendre un ou plusieurs rdv

-Pouvoir consulter ses rdv futur comme passé

-Si possible envoyé un mail 48h avant le rdv

*CONCEPTION VISUEL :*

*Pour le Header et le Footer*

Un dégrader de white au purple (#800080) 

*Dans le body*

Les mots importants seront en purple.

*Couleur Générale d’écriture :*

Black

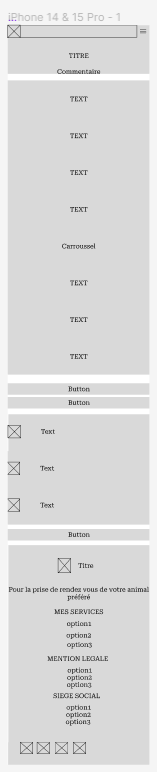
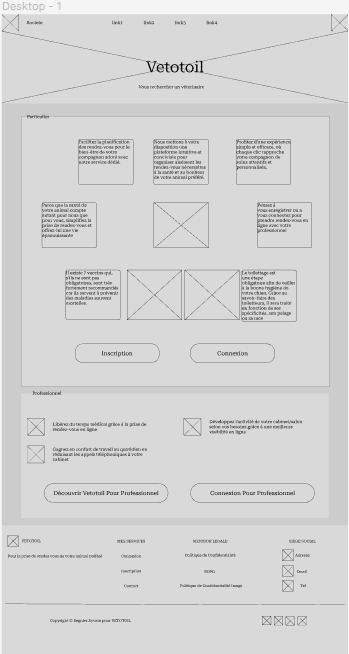
*Icones :*

Purple sauf dans le header et le footer

*Footer icone :*

Facebook , Twitter(X), instagram,linkedin

**Création des WireFrames :**



**Réalisation des maquettes :**

*Version PC*

**

*Version mobile :*

**

**Introduction et structure :**

Pour la partir Front-End, j’ai décidé d’utilisé du HTML,le framework Bootstrap ,Javascript ,twig et un peu de CSS le tout fait avec le Framework Symfony et son architecture.

J’ai aussi utilisé des librairies qui sont à la fois Front-End et Back-End comme :

-Le recaptchaV3 de Google fournit par Karser (<https://github.com/karser/KarserRecaptcha3Bundle>)

-VichUploaderBundle pour insérer des photos et les affichés (<https://symfony.com/doc/current/controller/upload_file.html>)

J’ai aussi utilisé l’API ADRESSE de l’Etat pour permettre de l’utilisateur de trouver celle-ci en fonction de ce qu’il va taper et ainsi renseigner la ville et le code postal automatiquement

Voici la structure de la partie Front-End

|  |  |
| --- | --- |
|  | Dans les assets nous retrouvons un dossier js ou tous les scripts sont inséré dedans.  Nous trouvons aussi le dossier styles ou il y a le fichier css |

|  |  |
| --- | --- |
|  | Dans cette partie dans le dossier public, j’ai créé un dossier images dans lequel il y a 4 autres dossiers.  Le dossier img est pour toutes les images du site  Les 3 autres dossiers sont lorsque les utilisateurs s’inscrivent ils peuvent ajouter une photo à leurs profils via vichupload et sont trié par rapport à leurs rôles.  Ensuite l’on peut voir le dossier templates dans lequel il y a les vues sous l’extension .html.twig qui indique que l’on peut mettre du code html dedans et le twig permet de récupérer des variables pour rendre le site plus dynamique et non statiques  Le fichier base.html.twig est le fichier de référence pour les autres pages c’est dedans que nous allons faire appel au script Bootstrap et importmap App qui fait référence au fichier app.js dans lequel le chemin du style/app.css est déclaré, cela est fait automatiquement par Symfony |

Voici le fichier base.html.twig

|  |
| --- |
|  |

Dans ce fichier nous voyons donc les liens pour accéder à la bibliothèque Bootstrap ainsi qu’à Bootstrap icon

De plus Symfony génère automatiquement le block JavaScript dans lequel il y a l importation de (‘app’) ou l’on peut retrouver le chemin du fichier css pour qu’il soit actif sur toutes les pages.

Ensuite toutes les pages seront traitées dans la balise <body> importer le header et le footer permet de ne plus avoir besoin de l’importer et sera visible sur toutes les pages du site.

En ce qui concerne le {% block body %} {% endblock %} cela sera à mettre dans toutes les pages pour informer le navigateur ou il doit l affiché sur la page.

Interface web adaptable

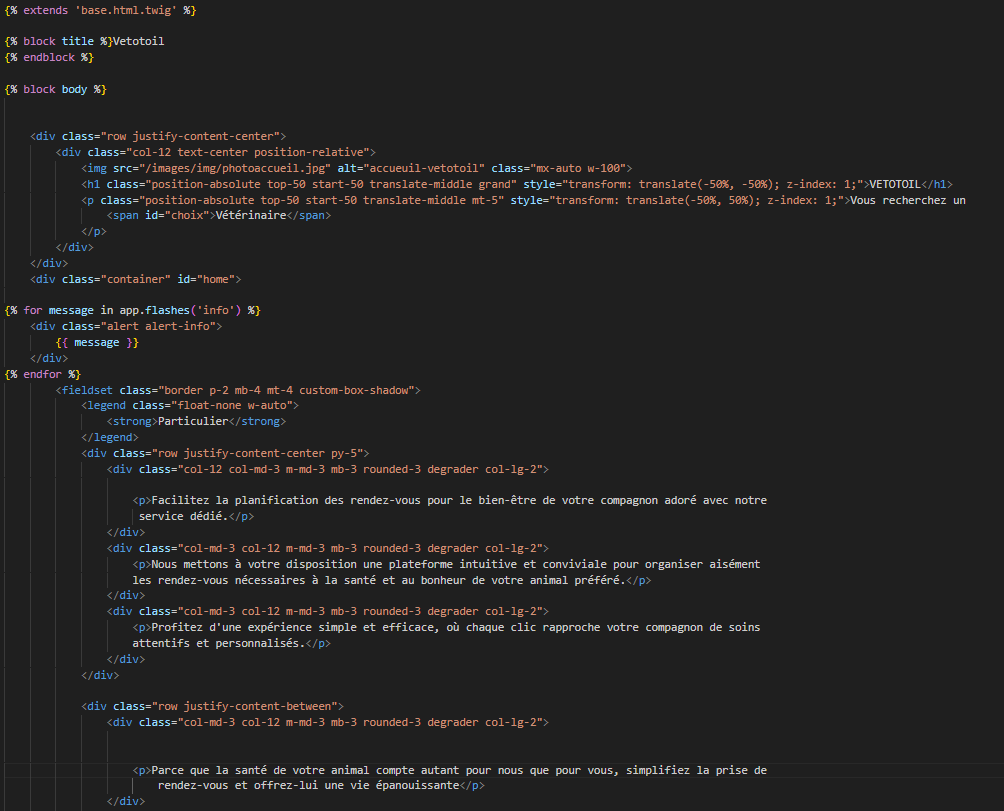
Nous avons vu le fichier principal préalablement et pour faire de notre site un sire responsive nous avons besoin de cette ligne.

<meta name="viewport" content="width=device-width, initial-scale=1">

Qui permet de dire au navigateur qu’il doit s’adapter à la largeur de l’appareil le viewport définissant les propriétés d’affichages.

Grace à la bibliothèque Bootstrap , le responsive est plus facile à gérer et nous avons besoin de moins coder dans le fichier css .

Voici une partie du [home.html.twig](http://home.html.twig)



Dans un premier temps on étend le fichier « base.html.twig » pour permettre d’avoir l’affichage prédéfinit ,puis ensuite nous mettons le code dans la balise {% block body %} et à la fin de la page il faut refermer cette balise avec {% endblock %}

Dans cette partie de code et avec bootstrap tous se fait au niveau des class.

Les row définissent une ligne et les colonnes sont une division de cette ligne, qui ne peux excéder 12 colonnes et chaque colonne peux encore avoir 12 colonnes.

Cette ligne veut dire qu’il y a une ligne et qu’elle est centré horizontalement dans son conteneur parent :

<div class="row justify-content-center">

Ici j’informe le navigateur que je veux que la colonne est une taille de 12 qui correspond à une colonne de 100% de largeur par rapport au paramètre précédent (s’il y a une colonne de 1 alors il prendra 100% de la colonne et non de la page).

Le text-center position-relative permet au contenu de cette colonne d’être centre horizontalement et il est positionné de façon relative par rapport à sa position normale dans le flux du document, je peux donc ajuster son positionnement à ma guise.

<div class="col-12 text-center position-relative">

<img src="/images/img/photoaccueil.jpg" alt="accueil-vetotoil" class="mx-auto w-100">

Ici j’insère l’image avec la balise img, le paramètre « alt » a plusieurs utilités qui sont, en cas d’erreur de chargement de l’image c’est l’annotation qui est indiqué qui sera vu à l’écran, il permet aussi aux personnes ayant des dispositifs de vue comme des lecteur d’écran de leur signifié ce que c’est et le dernier point est qu’il sert aussi pour les moteurs de recherche pour indexer les images (SEO).

Le src permet d’indiqué ou est situé le chemin de l’image et la class bootstrap mx auto défini qu’il faut centrer horizontalement l’image et le w-100 d’utiliser 100% de la largeur du conteneur.

La balise <h1> est la balise utilisé généralement pour les titres c’est l’une des balises les plus importantes pour le SEO.

*Le responsive sous bootstrap*

Bootstrap permet de définir dans ses classes les tailles d’ecran comme ceci

<div class="col-12 col-md-3 m-md-3 mb-3 rounded-3 degrader col-lg-2">

Ici on définit que la taille initiale est de de 12 colonnes donc 100% de l’écran, mais dès que l’on passe sur des écrans de taille moyenne la taille est réduite à 3 colonnes donc ¼ de celui-ci le m-md-3 permet de faire un margin de 3 sur tous les coté et le mb-3 rajoute une marge en bas ensuite le col-lg-2 indique que pour les écrans larges cela occupera 2 colonnes sur 12.

Comme on peut le voir bootstrap permet de gérer beaucoup de chose mais nous devons quand même passer par du css lorsque ce que l’on demande sort du type bootstrap par exemple pour faire un dégradé avec mes propres couleurs j’ai dû faire du css pour le header et le footer

.navbar,.footer {

    background: linear-gradient(white, purple);

}

Tous comme pour faire un hover sur un bouton sur les taille d’ecran supérieur a 1200px j ai du le faire en css en utilisant les média querie

<button type="button" class="btn btn-custom rounded-pill">Connexion</button>

@media only screen and (min-width: 1200px) {

    .btn-custom:hover {

        background-color:pink !important;

        color: purple !important;

    }

    .grand{

        font-size: 500%;

    }

}

Donc pour établir ce site j’ai donc commencé par le mobile first puis adapter mes class bootstrap en fonction des tailles d’ecran.

*Partie dynamique :*

Ayant la partie statique j’ai donc ajouter du JavaScript pour le rendre plus dynamique sur certaine fonctionnalité, dans un premier temps sur la saisie du mot de passe j’impose un nombre minimum de chiffre ET lettre, une majuscule et un caractère spécial dès que l’utilisateur commence à taper une « ul » s’affiche avec la liste qui se valide au fur et a mesure,si un champs est manquant la validation ne se fait pas

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  | Dans un premier temps je mets un écouteur pour confirmer le chargement du DOM  Ensuite je recherche l’id du submit  Met un écouteur dessus avec la fonction « click »  Vérifie, même si symfony le gère la concordance entre les 2 mots de passe  Vérifie la longueur du mot de passe avec .length  Vérifie s’il y a une majuscule  S’il y a un chiffre minimum  Si un caractère spécial est bien présent  Et pour finir l’alert dialog si un champs n’a pas été respecté en utilisant la variable déclaré errors en tableau et qui est push (insérer) au fur et à mesure des erreur détecté |

|  |  |
| --- | --- |
|  | Dans cette suite de code je refais les vérifications en direct pour afficher ou masquer les champs qui ne respecte pas la chartre des mots de passe en rendant visible l image ou en la rendant invisible mais aussi en rendant invisible l’ »ul » si le mot de passe est vide |

Ayant la partie statique j’ai donc ajouter du javascript pour le rendre plus dynamique sur certaine fonctionnalité, sur la page ou il y a l adresse a inséré, j’ai utilisé L’API Adresse du site de l’état