



# DOSSIER PROFESSIONNEL (DP)

Nom de naissance	▶ Cherbuin
Nom d'usage	▶ Entrez votre nom d'usage ici.
Prénom	▶ Lucas
Adresse	▶ Grand-rue 10 1071 Chexbres Suisse

## Titre professionnel visé

Développeur Web et Web Mobile

### MODALITÉ D'ACCÈS :

- ☐ Parcours de formation
- ☐ Validation des Acquis de l'Expérience (VAE)

## Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.  
**Ce titre est délivré par le Ministère chargé de l'emploi.**

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente  
**Obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

### Pour prendre sa décision, le jury dispose :

1. Des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. Du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. Des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. De l'entretien final (dans le cadre de la session titre).

*[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels  
Du ministère chargé de l'Emploi]*

### Ce dossier comporte :

- Pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- Un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- Une déclaration sur l'honneur à compléter et à signer ;
- Des documents illustrant la pratique professionnelle du candidat (facultatif)
- Des annexes, si nécessaire.

*Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.*

## Sommaire

### Exemples de pratique professionnelle

#### Développer la partie Front-end d'une application web ou web mobile sécurisée p.

5

- ▶ CP 1 Maquetter une application  
p.
- ▶ CP 2 Réaliser une interface statique et adaptable  
p.
- ▶ CP 3 Développer une interface utilisateur dynamique  
p.
- ▶ CP 4 Installer et configurer un environnement de travail

5

11

26

31

#### Développer la partie back-end d'une application web ou web mobile sécurisée p.

- ▶ CP 5 Mettre en place une base de données relationnelles  
p.
- ▶ CP 6 Développer des composants d'accès aux données SQL et NoSQL  
p.
- ▶ CP 7 Développer des composants métier coté serveur  
p.
- ▶ CP 8 Déploiement d'une application

41

46

52

59

**Titres, diplômes, CQP, attestations de formation** (*facultatif*)

p. 63

**Déclaration sur l'honneur**

p. 64

**Documents illustrant la pratique professionnelle** (*facultatif*)

p. 65

**Annexes** (*Si le RC le prévoit*)

p. 66

---

---

---

# **EXEMPLES DE PRATIQUE PROFESSIONNELLE**

## Activité-type 1

### Développer la partie Front-end d'une application web ou web mobile sécurisée

Exemple n°1 ► Maquetter une application

#### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour ce projet, le directeur du zoo Arcadia à contacter Studi afin de réaliser leur site internet , étant néophyte dans l'informatique et le développement, il s'est tourné vers des professionnels et a choisi un développeur junior.

Après un échange entre le directeur, il a souhaité apporter des indications

Le site doit faire office de vitrine pour les visiteurs en présentant les services, habitation avec leurs animaux ainsi qu'une plateforme de travail pour les différents utilisateurs qui ont un lien professionnel avec Arcade puissent saisir et voir des informations, de plus, le thème de l'interface doit être lié à l'écologie représentant les valeurs du zoo.

#### 1. User Stories

Pour valider les actions entreprises par les différents acteurs du site web, il faut bien entendu inclure les visiteurs, mais aussi les utilisateurs tels que l'administrateur, les employés du zoo et les vétérinaires, qui disposeront d'un espace pour travailler

En tant que	Je souhaite	Afin de
Visiteur	Consulter la page d'accueil	Connaitre les informations du zoo et pouvoir laisser un avis
Visiteur	Consulter les habitats	Apprendre sur les animaux disponibles
Visiteur	Consulter les profil des animaux	Consulter les informations et le retour des vétérinaires
Visiteur	Consulter les services	Consulter les services proposés par le zoo

Visiteur	Contacter le zoo	Avoir la possibilité de communiquer avec les employés
Utilisateur	Me connecter	Avoir accès au menu attribué
Utilisateur	Me déconnecter	Laisser la place à un autre utilisateur
Employé	Gérer les avis des visiteurs	Valider l'avis avant la publication sur la plateforme
Employé	Modifier un service	Mettre à jour une activité
Employé	Saisir le repas d'un animal	Indiquer le repas donné par le vétérinaire
Vétérinaire	Consulter les repas des animaux	Connaitre le suivis nutritionnel pour le dossier
Vétérinaire	Remplir un rapport médical	Donner un rapport sur l'état de santé d'un animal
Vétérinaire	Réaliser un compte rendu d'un habitat	Proposer des améliorations pour la qualité de l'enclos
Administrateur	Gérer les animaux, l'habitat, les horaires et les services	Pouvoir créer, consulter, mettre à jour et effacer les données relatif aux zoo
Administrateur	Gérer les utilisateurs	Pouvoir créer, consulter, mettre à jour et effacer les employés et vétérinaires d'Arcadia
Administrateur	Consulter les rapports vétérinaire	Connaitre l'état de santé des animaux ainsi que la qualité des habitats

Administrateur	Consulter un Dashboard de consultation	Connaitre la popularité des animaux selon leur consultation par les visiteurs
----------------	--	---

L'échange aboutit à la réalisation d'user stories, dans le but d'être sur la même longueur d'onde que les attentes du directeur, ce qui me permet de passer à l'étape suivante.

## 2. Wireframe

La réalisation des wireframes va servir à mettre en place la structure de l'application, c'est-à-dire la position des éléments importants (boutons de navigation, images, blocs de texte).

Le wireframe doit rester simple et basique dans sa structure afin de montrer le fond et non la forme.

J'ai donc réalisé les pages aux formats mobile et ordinateur, en optant pour un design orienté mobile : on commence par le format mobile, qui s'adaptera ensuite au format ordinateur.

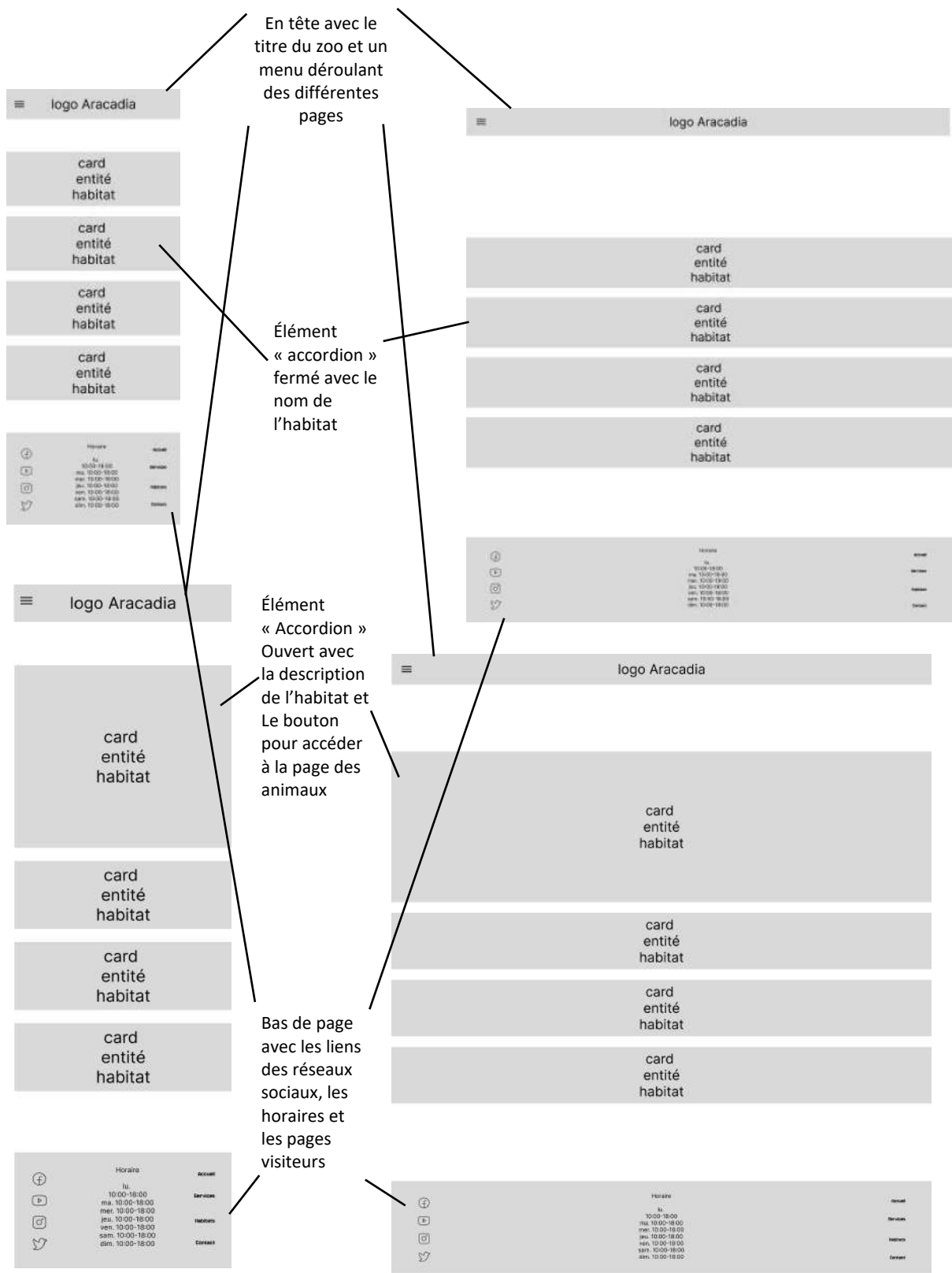
Voici une selection de wireframes qui vont permettre d'expliquer la méthodologie et les outils utilisé avec Le logiciel Figma

### Espaces entités

Avant de parler du contenu de la page, j'ai avant tout utiliser l'outil composant pour créer un modèle pour l'entête et le bas de page, cela permet de réutiliser ce modèle dans tout mes Wireframes, cet outil me servira à dupliquer d'autres éléments dans mon projet comme des cartes et des boutons.

J'ai construis l'espace des habitats de manière que, dès qu'un visiteur clique sur une card d'un habitat, celui ci s'ouvre pour afficher les informations de l'habitat ainsi qu'un bouton pour accéder à l'espace des animaux.

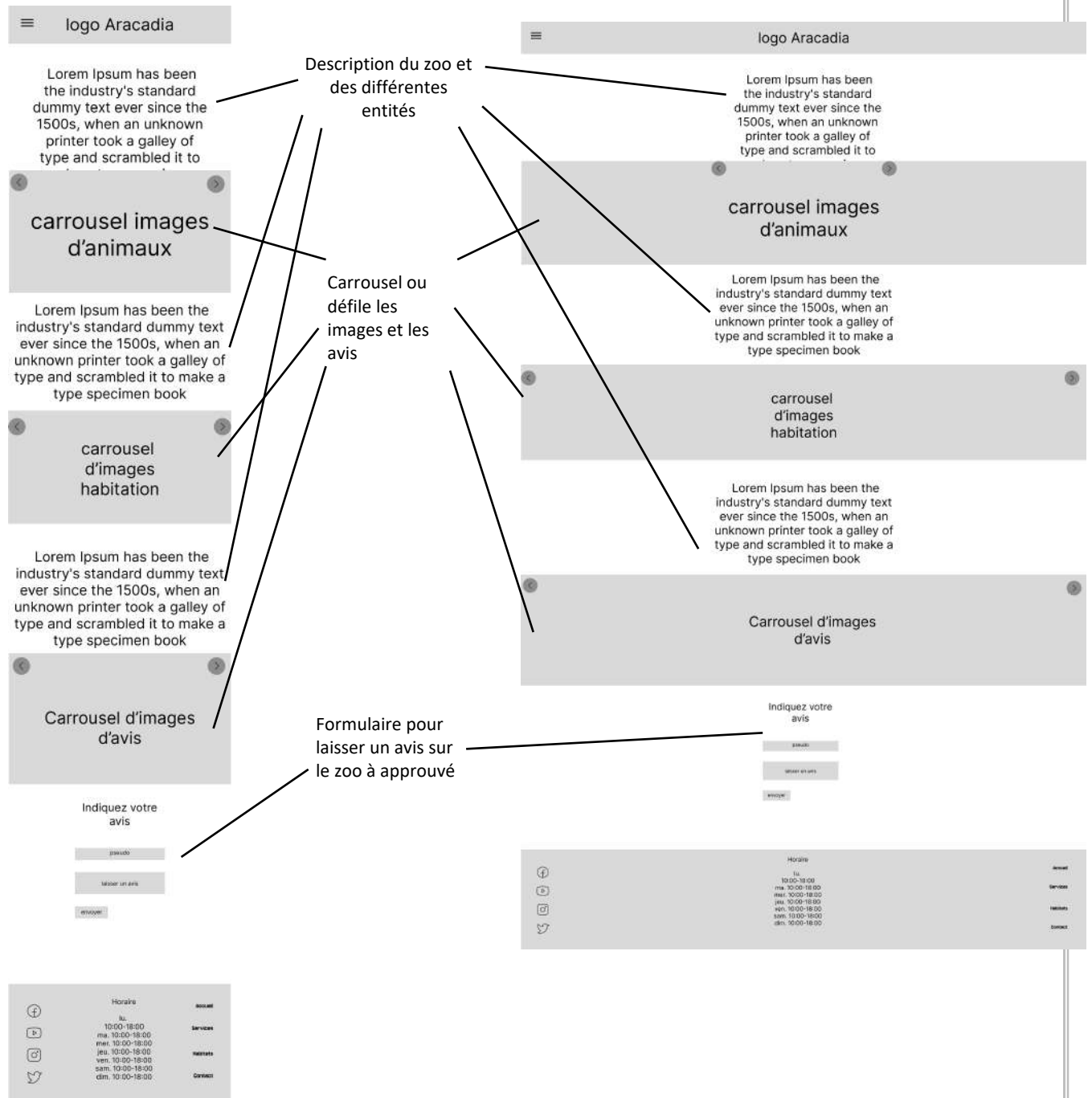
Ce procédé vient du composant UI « Accordion » de Bootstrap afin de ne pas surcharger une page d'informations.





## Accueil

La page d'accueil sert à présenter les différentes informations que le zoo propose, pour montrer les différentes données, je vais implémenter le composant « carrousel » qui permet de faire défiler les données comme des images ou les avis laisser par les visiteurs dans le formulaire si dessous ayant des boutons pour passer d'un éléments à l'autre.

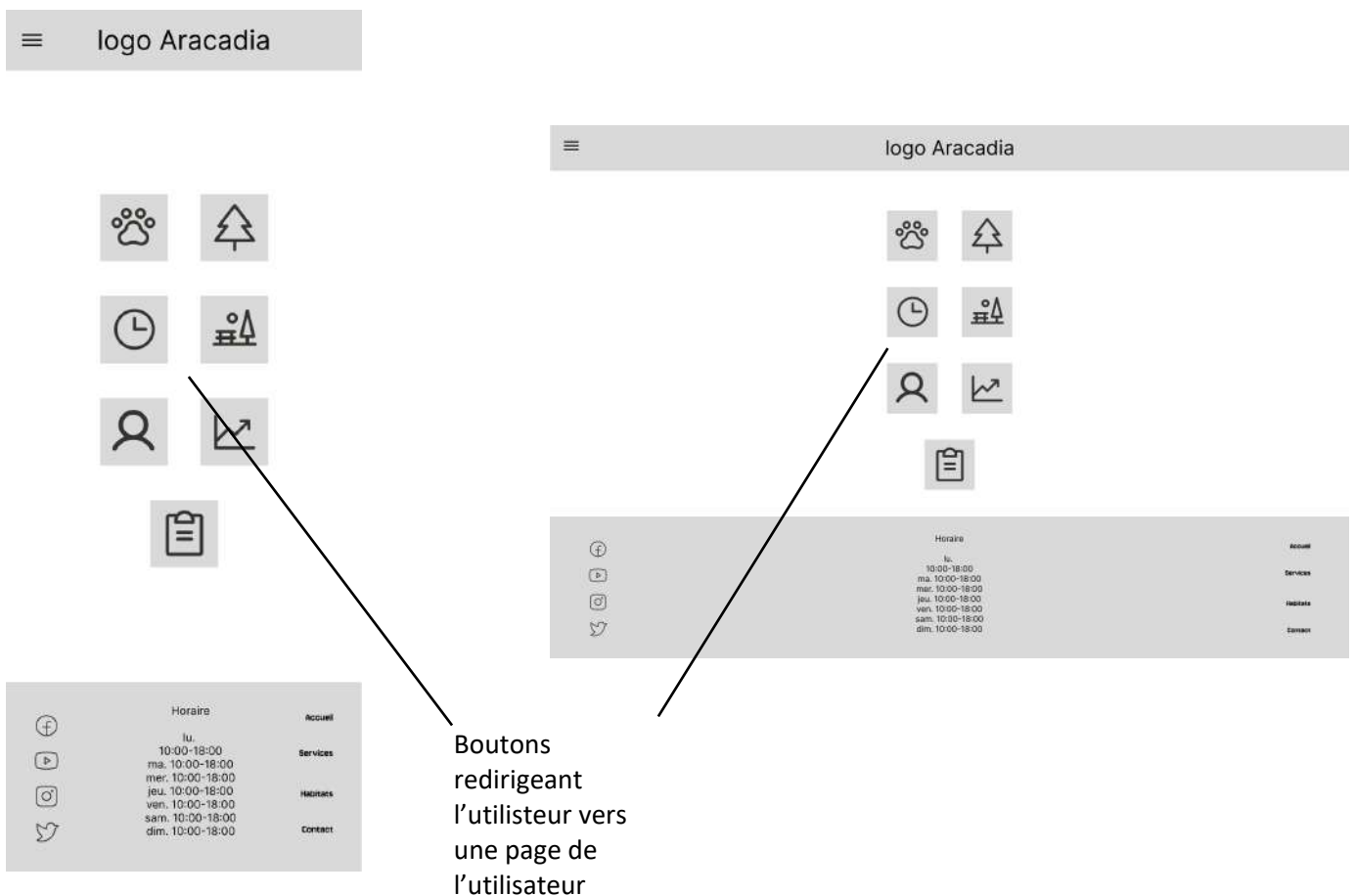


## Menu Administrateur

Le menu d'un utilisateur est construit d'une page avec des boutons redirigeant sur un des espaces de travail.

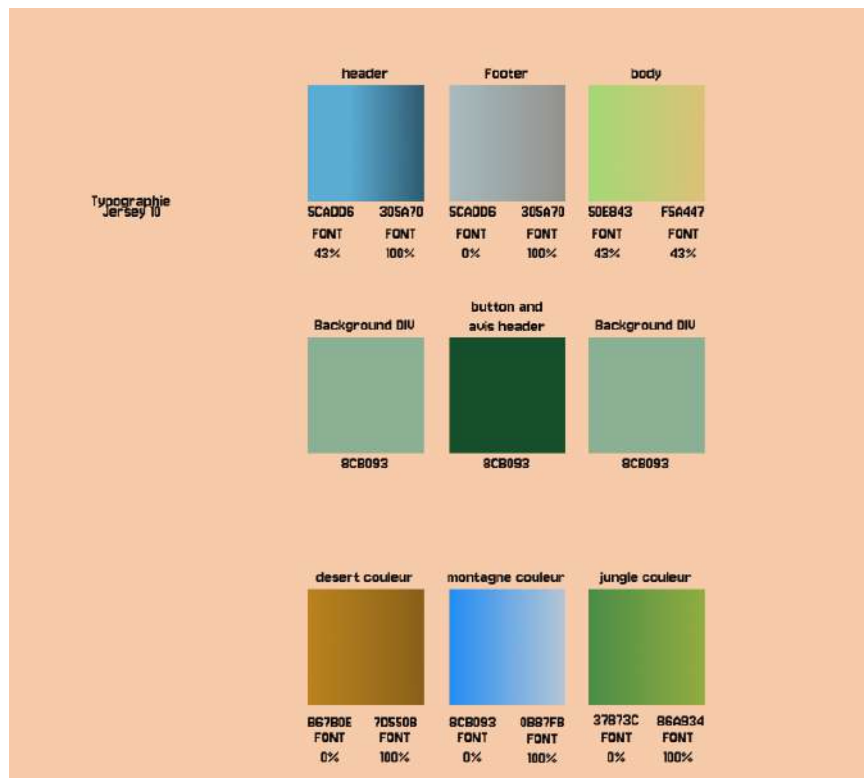
Ici sur le menu administrateur, l'utilisateur pourra aller sur des espaces de gestion des animaux, habitats, services, horaire et des utilisateurs et des pages de consultations comme le rapport des vétérinaire et le dashboard des consultations des animaux.

Pour les icônes, j'ai pu trouver dans la section communauté de Figma, un fichier contenant un catalogue icônes appelé « Phosphore icon » parfait pour représenté où le bouton redirige, de plus, Phosphore icon est une banque d'icône pour les interfaces que je pourrais utiliser lors de la conception des interfaces



### 3. Design

Pour finaliser l'interface du site web, une maquette design servira à se faire une idée de l'apparence visuelle. Je crée avant tout une charte graphique indiquant les couleurs ainsi que la police d'écriture utilisée.

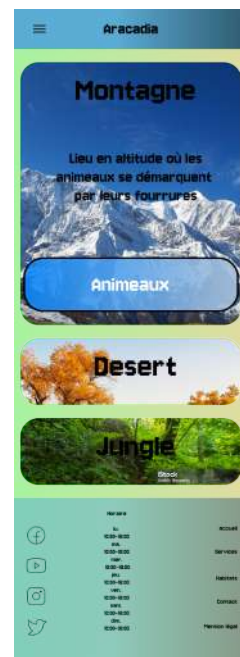
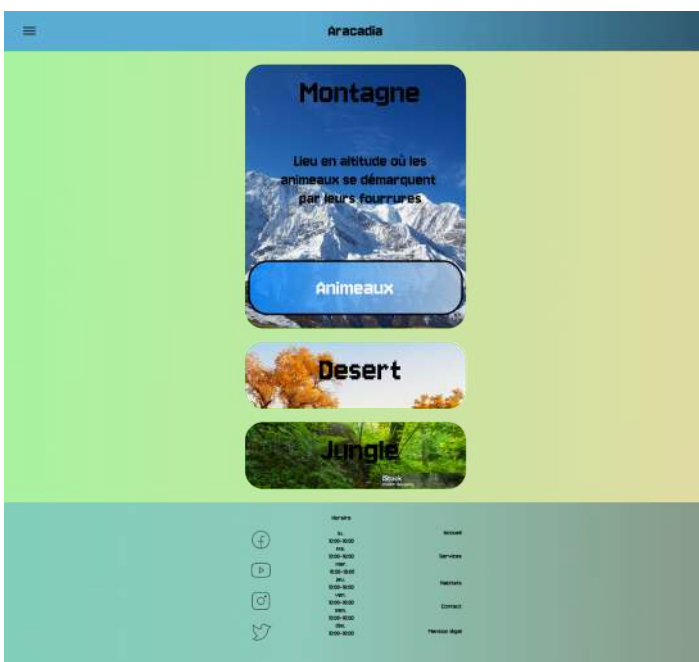


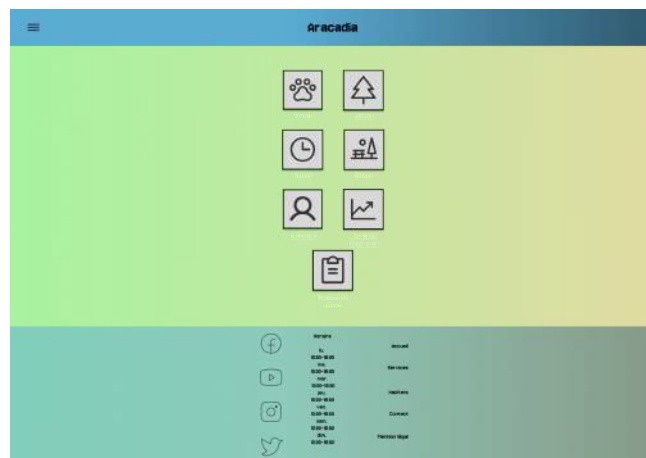
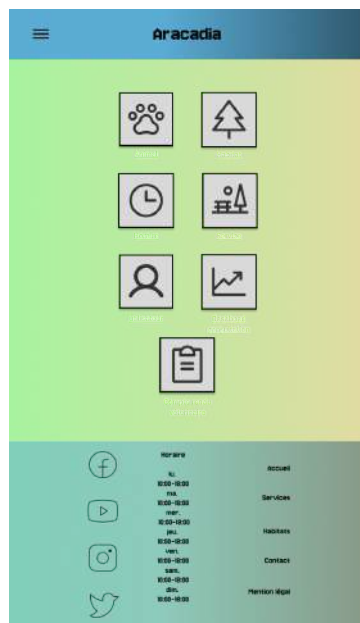
Afin de respecter la consigne, je me suis tourné vers des couleurs naturelles pour représenter l'aspect écologique. J'ai repris les wireframe en les colorisant, rajoutant les la police d'écriture, arrondir les éléments et préciser des informations

## Accueil



## Espace entité





## 2. Précisez les moyens utiliser

Pour le projet, j'ai utilisé Word pour les user stories et Figma pour les wireframes et le design.

## 3. Avec qui avez-vous travaillé ?

J'ai réalisé ce travail seul

---

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01/10/2024** au : 08/02/2025

#### 5. Informations complémentaires *(facultatif)*

---

## Activité-type 1

## Développer la partie Front-end d'une application web ou web mobile sécurisée

Exemple n° 2 ► CP 4 Réaliser une interface statique et adaptable

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

***Je vais utiliser le travail de maquettage effectué dans l'exemple 1 pour réaliser le projet.***

*Ce projet va fonctionner sous Symfony, un framework orienté back-end, mais qui offre également des outils pour travailler sur du front-end basique.*

*Ici, l'utilisation du HTML sera remplacée par Twig, un moteur de templates utilisé par PHP et fourni par Symfony.*

*Le projet fonctionne avec du CSS pour le design ainsi que du JavaScript pour gérer certaines actions du site web.*

*Pour vérifier si mon site est bien **responsive**, le navigateur Firefox propose une fonctionnalité dans ses outils de développement.*

#### **1. Structure et organisation**

*Voici comment j'ai conçu l'arborescence des pages :*

*L'objectif est d'avoir une organisation propre et compréhensible, notamment pour le routing, mais surtout pour faciliter le travail sur les différentes interfaces.*

***HeaderFooter.html.twig***

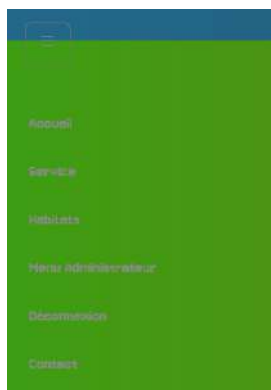


## Header

```
<!-- Header -->
<header>
  <div class="wrapper d-flex">
    <!-- Sidebar -->
    <nav id="sidenav">
      <div class="sidebar-header p-3 d-flex justify-content-between align-items-center">
        <button id="close-button" class="btn btn-outline-light" onclick="toggleNav()" style="display: none;">
          <i class="ph ph-list"></i>
        </button>
      </div>
      <ul class="list-unstyled components mt-3">
        <li><a href="{{ path('app_accueil') }}" class="text-light">Accueil</a></li>
        <li><a href="{{ path('app_service_visiteur') }}" class="text-light">Service</a></li>
        <li><a href="{{ path('app_habitat_visiteur') }}" class="text-light">Habitats</a></li>
        <li>
          {% if app.user %}
            {% if is_granted('ROLE_ADMIN') %}
              <a href="{{ path('admin_menu') }}" class="text-light">Menu Administrateur</a>
            {% elseif is_granted('ROLE_EMPLOYEE') %}
              <a href="{{ path('employee_menu') }}" class="text-light">Menu Employé</a>
            {% elseif is_granted('ROLE_VETERINAIRE') %}
              <a href="{{ path('veterinaire_menu') }}" class="text-light">Menu Vétérinaire</a>
            {% endif %}
            <a href="{{ path('app_logout') }}" class="text-light" style="color: red;">Déconnexion</a>
          {% else %}
            <a href="{{ path('app_login') }}" class="text-light" style="color: blue;">Connexion</a>
          {% endif %}
        </li>
        <li><a href="{{ path('app_contact') }}" class="text-light">Contact</a></li>
      </ul>
    </nav>
    <!-- Page Content -->
    <div id="content" class="w-100">
      <!-- Navbar -->
      <nav class="header-container d-flex align-items-center">
        <div class="container-fluid d-flex justify-content-between align-items-center">
          <button id="open-button" class="btn me-3" onclick="toggleNav()">
            <i class="ph ph-list"></i>
          </button>
          <h1 class="zoo-title">Arcadia</h1>
        </div>
      </nav>
    </div>
  </div>
</header>
```

Contient le menu de navigation des différents pages

Visiteurs et la page de connexion et le menu utilisateur qui s'adapte selon le rôle connecté,

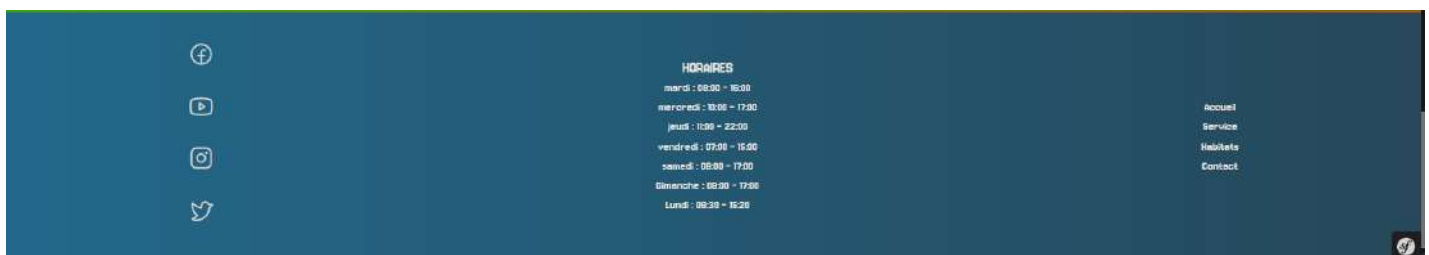


## Footer

```
<main>
  {% block body %}{% endblock %}
</main>

{# Footer #}
<footer>
  <div class="container">
    <div class="footer-section">
      <ul class="social-links vertical">
        <li><a href="#" class="text-body"><i class="ph ph-facebook-logo"></i></a></li>
        <li><a href="#" class="text-body"><i class="ph ph-youtube-logo"></i></a></li>
        <li><a href="#" class="text-body"><i class="ph ph-instagram-logo"></i></a></li>
        <li><a href="#" class="text-body"><i class="ph ph-twitter-logo"></i></a></li>
      </ul>
    </div>
    <div class="footer-section">
      <h5 class="text-uppercase">Horaires</h5>
      <ul>
        {% for horaire in get_horaires() %}
          <li>{{ horaire.jour }} : {{ horaire.ouverture }} - {{ horaire.fermeture }}</li>
        {% endfor %}
      </ul>
    </div>
    <div class="footer-section">
      <ul class="footer-links">
        <li><a href="{{ path('app_accueil') }}">Accueil</a></li>
        <li><a href="{{ path('app_service_visiteur') }}">Service</a></li>
        <li><a href="{{ path('app_habitat_visiteur') }}">Habitats</a></li>
        <li><a href="{{ path('app_contact') }}">Contact</a></li>
      </ul>
    </div>
  </div>
</footer>
```

Contient les liens des différents réseaux sociaux, les Horaires importé stockée de la base de données ainsi que les liens des différentes pages visiteurs



## Main

```
{# Contenu principal #}
<main>
  {% block body %}{% endblock %}
</main>
```

La partie *main* contient un block pour afficher le contenu (body) d'une des pages

Exemple si je prend la page twig du formulaire de contact

Je dois importer la page twig du header et footer dans le fichier twig du formulaire

```
You, 3 months ago | 1 author (You)
{% extends 'headerFooter.html.twig' %}
```

Et construire la balise body avec les accolade

```
{% block body %}
    {{ form_start(form, {'enctype': 'multipart/form-data'}) }}
    <div class="form-floating mb-3">
        {{ form_start(form) }}
    </div>
    <div class="form-floating mb-3">
        {{ form_row(form.email) }}
    </div>
    <div class="form-floating mb-3">
        {{ form_row(form.Titre) }}
    </div>
    <div class="form-floating mb-3">
        {{ form_row(form.votre_demande) }}
    </div>
    {{ form_rest(form) }}
    <button type="submit" class="btn btn-primary w-100">Envoyer</button>
    {{ form_end(form) }}
</div>
{% endblock %}
```

Et cela permet d'afficher le Twig *HeaderFooter.html.twig* dans mes autres fichiers



Pour le contenu *main* les pages principales sont construites avec des balises `<div>` ayant le class encapsuler des block Body

D'autres cellules seront utilisées pour un contexte d'utilisation

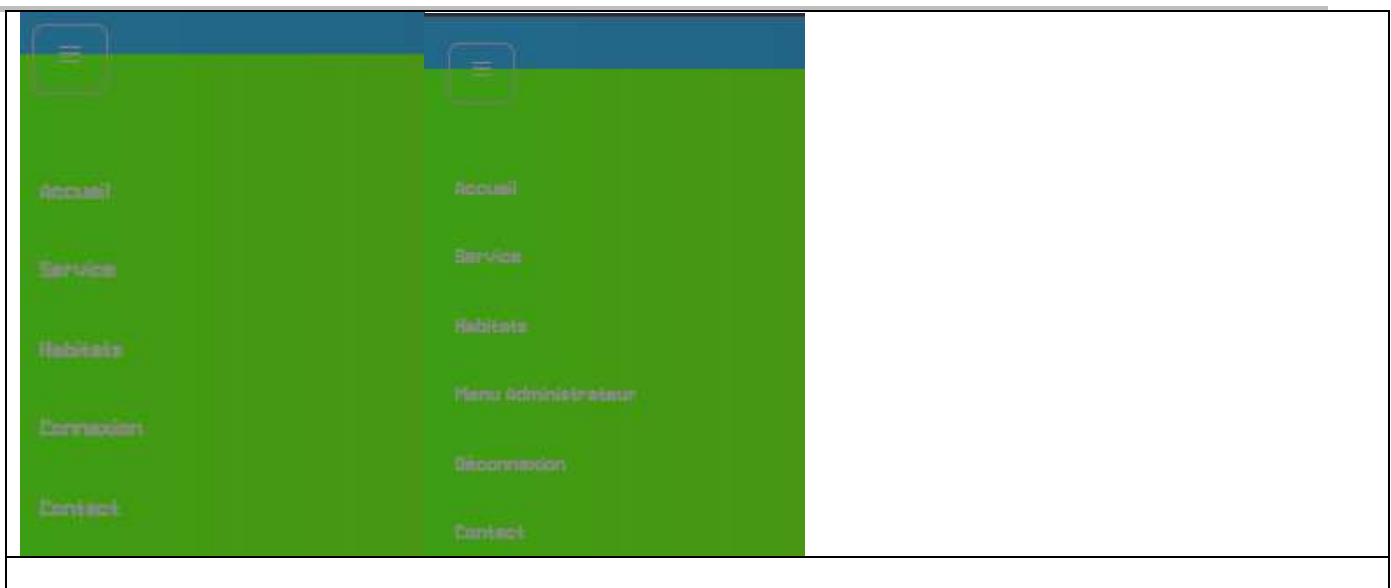
<Nav>

Pour le menu de navigation

```
<!-- Sidebar -->
<nav id="sidenav">
  <div class="sidebar-header p-3 d-flex justify-content-between align-items-center">
    <button id="close-button" class="btn btn-outline-light onclick="toggleNav()" style="display: none;">
      <i class="ph ph-list"></i>
    </button>
  </div>
  <ul class="list-unstyled components mt-3">
    <li><a href="{{ path('app_accueil') }}" class="text-light">Accueil</a></li>
    <li><a href="{{ path('app_service_visiteur') }}" class="text-light">Service</a></li>
    <li><a href="{{ path('app_habitat_visiteur') }}" class="text-light">Habitats</a></li>
    <li>
      {% if app.user %}
      {% if is_granted('ROLE_ADMIN') %}
        <a href="{{ path('admin_menu') }}" class="text-light">Menu Administrateur</a>
      {% elseif is_granted('ROLE_EMPLOYEE') %}
        <a href="{{ path('employee_menu') }}" class="text-light">Menu Employé</a>
      {% elseif is_granted('ROLE_VETERINAIRE') %}
        <a href="{{ path('veterinaire_menu') }}" class="text-light">Menu Vétérinaire</a>
      {% endif %}
      <a href="{{ path('app_logout') }}" class="text-light" style="color: red;">Déconnexion</a>
    {% else %}
      <a href="{{ path('app_login') }}" class="text-light" style="color: blue;">Connexion</a>
    {% endif %}
    </li>
    <li><a href="{{ path('app_contact') }}" class="text-light">Contact</a></li>
  </ul>
</nav>

<!-- Page Content -->
<div id="content" class="w-100">
  <!-- Navbar -->
  <nav class="header-container d-flex align-items-center">
    <div class="container-fluid d-flex justify-content-between align-items-center">
      <button id="open-button" class="btn me-3" onclick="toggleNav()">
        <i class="ph ph-list"></i>
      </button>
      <h1 class="zoo-title">Arcadia</h1>
    </div>
  </nav>
  <div>
    You, 3 months ago • ajout du header/footer du site, le menu de navi...
  </div>
</div>
</div>
```

Dans le menu de la navigation il y a la balise <button> ouvrant le menu de navigation, <ul> permet l'ajout de puc pour scruter les liens, <li> permet d'ajouter les éléments dans la balise de structure et pour finir, les accolades offert par twig permet de configurer un If..else permettant de condition le navigateur selon le rôle connecter



### <Script>

Pour intégrer le java script

```
{# Scripts JS nécessaires #}
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"></script>
<script>
  function toggleNav() {
    const sidenav = document.getElementById("sidenav");
    const openButton = document.getElementById("open-button");
    const closeButton = document.getElementById("close-button");

    if (sidenav.classList.contains("open")) {
      sidenav.classList.remove("open");
      closeButton.style.display = "none";
      openButton.style.display = "block";
    } else {
      sidenav.classList.add("open");
      closeButton.style.display = "block";
      openButton.style.display = "none";
    }
  }
</script>
</body>
</html>
```

Pour que le navigateur ci dessus puisse fonctionner, il faut configurer un script servant a l'ouvrir, Les 2 premiers scripts doivent être chargé au début servant à importer les bibliothèques JQuery (simplifiant le Dom et les événements) et bootstrap JS (ajout des fonctionnalités interactive offert par Bootstrap).

La fonction *ToggleNav()* sert se déclenche dès que l'on interagit avec le bouton de navigateur, les constantes permet de récupérer les classes <id> dans notre navigateur (sidenav pour la barre qui doit être ouvrir, open-button pour ouvrir la barre latéral, close-button pour fermer)

L'instruction if permet de vérifier si la class CSS *open* est déjà ouvert (contain) et si elle est, on l'enlève (remove), Les 2 lignes *style.display* servent à cahcher le bouton fermer (*none*) ou montrer le bouton ouvert (block).

L'instruction else sert quand la barre sidenav n'est pas encore ouverte, la 1<sup>er</sup> ligne va executé la class CSS open en changeant le style de la barre, les 2 autres lignes vont afficher le bouton de fermeture (block) et cacher le bouton d'ouverture.

<Table>		
Pour définir les tableaux		
<pre> &lt;table class="compte-rendu-table" id="compteRenduTable"&gt;   &lt;thead&gt;     &lt;tr&gt;       &lt;th&gt;Nom&lt;/th&gt;       &lt;th&gt;Date&lt;/th&gt;       &lt;th&gt;Rapport&lt;/th&gt;     &lt;/tr&gt;   &lt;/thead&gt;   &lt;tbody&gt;     {% for veterinaire in veterinaires %}       &lt;tr&gt;         &lt;td&gt;{{ veterinaire.animal.prenom }}&lt;/td&gt;         &lt;td&gt;{{ veterinaire.date }}&lt;/td&gt;         &lt;td&gt;           &lt;button class="rapport-button" data-target="#rapport-veterinaire-{{ veterinaire.id }}" onclick="toggleReport(this)"&gt;             Rapport ▼           &lt;/button&gt;           &lt;div class="rapport-content d-none" id="rapport-veterinaire-{{ veterinaire.id }}"&gt;             &lt;p&gt;{{ veterinaire.detail }}&lt;/p&gt;             &lt;span&gt;{{ veterinaire.utilisateur.email }}&lt;/span&gt;           &lt;/div&gt;         &lt;/td&gt;       &lt;/tr&gt;     {% endfor %}   &lt;/tbody&gt; &lt;/table&gt; </pre>		
<p>On peut trouver des sous balises permettant de structurer celle de &lt;table&gt; , &lt;tr&gt; va définir une ligne dans une cellule &lt;thead&gt; sert d’annoncer que c’est l’entête du tableau avec les balises &lt;th&gt; pour définir les élément d’une cellule propre aux entêtes.</p> <p>Le &lt;tbody&gt; sert pour le contenu principale avec &lt;td&gt; qui sert à définir les lignes des cellules.</p> <p>On peut remarquer j’apporte les données de la base avec des accolades, grace au moteur de template twig offert par symfony, je peux afficher des variables afin d’éviter une injection XSS via une url ou un champ de texte.</p>		
nom	date	rapport
erick	10/08/2024	Rapport ▼
Yuka	10/08/2024	Rapport ▼
Isabelle	10/07/2024	Rapport ▼
Henri	02/05/2025	Rapport ▼
erick	08/02/2025	Rapport ▼

<H1>		
Pour réaliser des titres de Différentes taille (1 à 6)		
<pre> &lt;!-- Liste des avis --&gt; &lt;div class="avis-section"&gt;   &lt;h3 class="mt-5"&gt;Avis des visiteurs&lt;/h3&gt;   &lt;div id="avisCarousel" class="carousel slide" data-bs-ride="carousel"&gt;     &lt;div class="carousel-inner"&gt; </pre>		
Ici la ligne « Avis des visiteurs » va donner le niveau 3 pour un titre de section (une police d’environ 18 px)		

### 2. CSS et responsive

Pour que mon application ressemble à ma maquette, je configure le CSS afin d'appliquer des options en lien avec le positionnement des éléments, les couleurs correspondantes, etc.

(Exemple : CSS pour le header)

Le CSS me permet d'utiliser le format hexadécimal pour intégrer précisément les couleurs utilisées dans ma maquette, ainsi que différentes unités de mesure (par exemple, les pixels pour les dimensions à l'écran, et le *rem* pour des propriétés responsives).

J'utilise également certaines propriétés spécifiques pour le contenu, comme `text-align: center` pour centrer le texte ou `justify-content: space-between` pour ajouter un espace entre les éléments groupés.

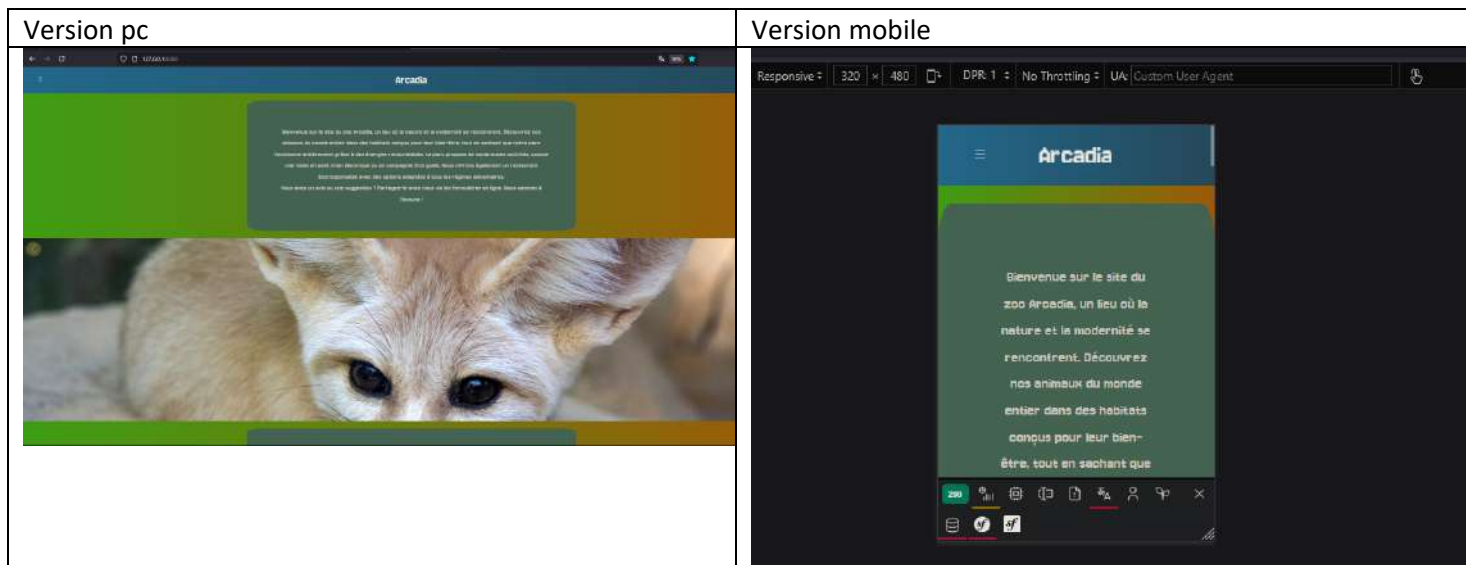
Enfin, pour que mon application puisse être utilisée sur d'autres machines avec des formats d'écran différents, j'utilise les *media queries* afin de rendre la page responsive.

Le texte s'adaptera pour des écrans de smartphones en adaptant la taille de la police et en adaptant la taille du padding

```
@media (max-width: 768px) {  
  .consultation-header, .consultation-item {  
    font-size: 14px;  
    padding: 10px;  
  }  
}
```

You, 3 months ago • ajout des assets (css/im

Grâce aux outils de développement de Firefox, je peux utiliser l'option responsive pour vérifier si mon application s'adapte à un écran de smartphone (ici 320 x 480)



## 2. Précisez les moyens utilisés :

Pour travailler plus facilement avec les technologies front-end, j'ai utilisé Bootstrap, ce qui me permet de personnaliser efficacement les éléments de mon projet.

Cette librairie est importée via CDN afin de ne pas surcharger le projet Symfony. J'importerai également les différentes icônes du projet de la même manière.

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet



---

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01.10.24** Au : **08/02/25**

---

#### 5. Informations complémentaires (*facultatif*)

---

## Activité-type 1

## Développer la partie Front-end d'une application web ou web mobile sécurisée

Exemple n° 3 ► Développer une interface utilisateur dynamique

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le projet **Arcadia**, j'ai réalisé des composants dont le but était d'afficher, au clic, les informations d'un animal. Cependant, cette action servait surtout à incrémenter une unité au nombre de consultations de cet animal.

Le résultat devait s'afficher dans la section **Dashboard** de l'espace administrateur.

#### 1. Mise en place de la base de données et du modèle

```
//entité nosql
You, 2 months ago | 1 author (You)
#[ODM\Document]
You, 2 months ago | 1 author (You) | 4 references | 0 implementations
class Consultation
{
    #[ODM\Id]
    1 reference
    private ?string $id = null;

    #[ODM\Field(type: "string")]
    2 references
    private string $animalId;

    #[ODM\Field(type: "int")]
    2 references
    private int $clicks = 0;

    1 reference | 0 overrides
    public function __construct(string $animalId)
    {
        $this->animalId = $animalId;
    }

    0 references | 0 overrides
    public function getId(): ?string
    {
        return $this->id;
    }

    0 references | 0 overrides
    public function getAnimalId(): string
    {
        return $this->animalId;
    }

    2 references | 0 overrides
    public function getClicks(): int
    {
        return $this->clicks;
    }

    1 reference | 0 overrides
    public function incrementClicks(): void
    {
        $this->clicks++;
    }
}
```

La table que je vais créer n'aura aucune relation. L'utilisation de **MongoDB** est adaptée en raison des mises à jour fréquentes et de sa simplicité d'utilisation, compte tenu de l'objectif.

Ensuite, une entité est créée pour définir les valeurs et les fonctions.

## 2. La page des animaux

```
<script>
function toggleAnimalAccordion(id) {
  const item = document.getElementById(`animal-${id}`);
  const body = item.querySelector('.accordion-body');
  const isOpen = body.style.display === 'block'; // Vérifie si l'accordéon est ouvert
  const bgUrl = item.getAttribute('data-bg'); // Récupère l'URL de fond

  // Ferme tous les autres items
  document.querySelectorAll('.accordion-item').forEach(el => {
    const elBody = el.querySelector('.accordion-body');
    elBody.style.display = 'none'; // Cache le corps de l'élément
    el.style.backgroundColor = ''; // Enlève l'image de fond
    el.classList.remove('open');
    el.classList.add('closed');
  });

  if (!isOpen) {
    body.style.display = 'block'; // Affiche le corps de l'élément
    item.style.backgroundColor = `url(${bgUrl})`; // Applique l'image de fond

    // Envoi de la requête POST pour incrémenter les clics
    fetch(`/animal/click/${id}`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
    })
      .then((response) => response.json())
      .then((data) => {
        console.log('Réponse:', data);
        if (data.clicks !== undefined) {
          document.getElementById(`click-count-${id}`).textContent = data.clicks;
        }
      })
      .catch((error) => console.error('Erreur:', error));
  }
}
</script>
```

Pour faire en sorte que les visiteurs doivent obligatoirement cliquer sur un élément pour accéder aux informations, j'ai effectué des recherches en ligne et sur le site de Bootstrap. Le composant « **Accordion** » m'a semblé le plus adapté, car les informations ne deviennent accessibles qu'en interagissant avec lui.

Pour faire fonctionner l'incrémentation, un script JavaScript exécute une fonction qui ouvre le composant affichant l'image et les informations des animaux.

La suite du code permet de gérer le stockage de l'incrémentation, en ajoutant une valeur dans la base de données NoSQL.

### 3. Récupération des données et affichage dans le Dashboard

Pour que notre script fonctionne, un Controller est créé afin de récupérer le nom de l'animal consulté ainsi que le clic, pour l'incrémenter dans MongoDB.

```
//dashboard pour la consultation des animeaux
#[Route(path: '/admin/consultation', name: 'app_consultation')]
3 references | 0 overrides
public function consultation(): Response
{
    $animals = $this->animalRepository->findAll();
    $preparedConsultation = $this->getConsultationData(animals: $animals);

    return $this->render(view: 'admin/consultation.html.twig', parameters: [
        'preparedConsultation' => $preparedConsultation,
    ]);
}

1 reference
private function getConsultationData($animals): array
{
    $consultations = [];

    foreach ($animals as $animal) {
        $key = "animal:click:{".$animal->getId()."}";
        $clicks = is_numeric(value: $this->mangoService->getClick(animalId: $key)) ? $this->mangoService->getClick(animalId: $key) : 0;

        $consultations[] = [
            'id' => $animal->getId(),
            'prenom' => $animal->getPrenom(),
            'clicque' => $clicks,
        ];
    }

    return $consultations;
}
```

You, 3 months ago • ajout des fonctionnalités du menu admin

Pour finir, les données récupérées s'afficheront sur le Dashboard, avec le nombre de consultations par animal du zoo.

```

{% block body %}
  <div class="consultation-page">
    <div class="container d-flex justify-content-center align-items-center vh-100">
      <div class="w-100" style="max-width: 500px;">
        <div class="consultation-container">
          <div class="consultation-head">
            <p>Animal</p>
            <p>Nb de consultations</p>
          </div>
          <ul class="consultation-list">
            {% for consultation in preparedConsultation %}
              <li class="consultation-item" data-animal-id="{{ consultation.id }}">
                <span class="animal-name">{{ consultation.prenom }}</span>
                <span class="consultation-count">{{ consultation.clique }}</span>
              </li>
            {% endfor %}
          </ul>
        </div>
      </div>
    </div>
  </div>

  <script>
    document.addEventListener('DOMContentLoaded', async function () {
      try {
        const elements = document.querySelectorAll('.consultation-item');

        for (const element of elements) {
          const animalId = element.dataset.animalId;
          const countElement = element.querySelector('.consultation-count');

          if (animalId && countElement) {
            const response = await fetch(`/animal/click/${animalId}`, { method: 'POST' });
            const data = await response.json();

            if (data.clique !== undefined) {
              countElement.textContent = data.clique;
            }
          }
        }
      } catch (error) {
        console.error('Erreur lors de la récupération des clics:', error);
      }
    });
  </script>

{% endblock %}

```

Le script renvoie les consultations effectuées par les visiteurs

---

---

## 2. Précisez les moyens utilisés :

Comme expliqué, j'ai utilisé les technologies proposées par Bootstrap

## 3. Avec qui avez-vous travaillé ?

J'ai réalisé la tâche seul

## 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01.10.24** Au : 08/02/25

## 5. Informations complémentaires *(facultatif)*

## Activité-type 1

### Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°4 ► CP 4 Installer et configurer un environnement de travail

#### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

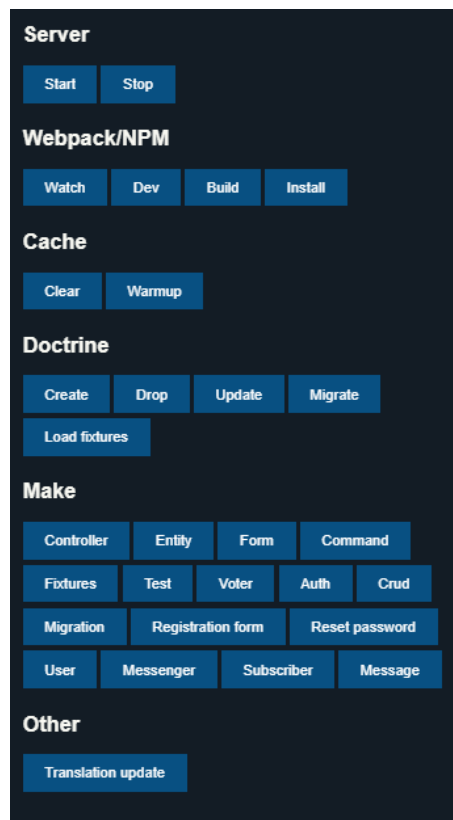
Pour configurer mon environnement j'ai du utiliser plusieurs outils

##### Visual studio code

C'est l'éditeur de texte qui m'a servi pour le travail, je l'ai choisi car il est polyvalent pour le travail front et back end, il peut gérer plusieurs langages de programmation, gérer des frameworks et gérer les outils que je vais citer après.

mais surtout, il peut accueillir des extensions permettant un confort et qualité de travail.

Par exemple, grâce à l'extension *Symfony tools* je peux exécuter les commandes communes de Symfony en appuyant sur un bouton



Je peux ici lancer le serveur et l'éteindre. Gérer la génération d'entité de formulaire, Interagir avec la base de données et vider le cache tout ça en 1 clic, ce qui me permet de ne pas aller sur la documentation toute les 5 minutes.

## Symfony

Ce framework Back end open source orienté PHP va me permettre de construire mon application avec la structure MVC (Modèle, vue, controller) pour un code organisé et propre.

Avant d'expliquer les point sécurité que propose Symfony, je vais expliquer les besoins pour pouvoir l'utiliser.

### XAMPP

Ce logiciel d'environnement de développement local va me permettre de configurer une base de donnée mySql pour la gestion des données ainsi que pouvoir avoir les version de PHP compatible avec Symonfy.

### Composer

Ce gestionnaire de dépendance pourra installer les dépendences de mon projet, je dois installer le gestionnaire avec windows et cocher l'option « ADD to PATH » pour être utiliser dans tout les dossiers

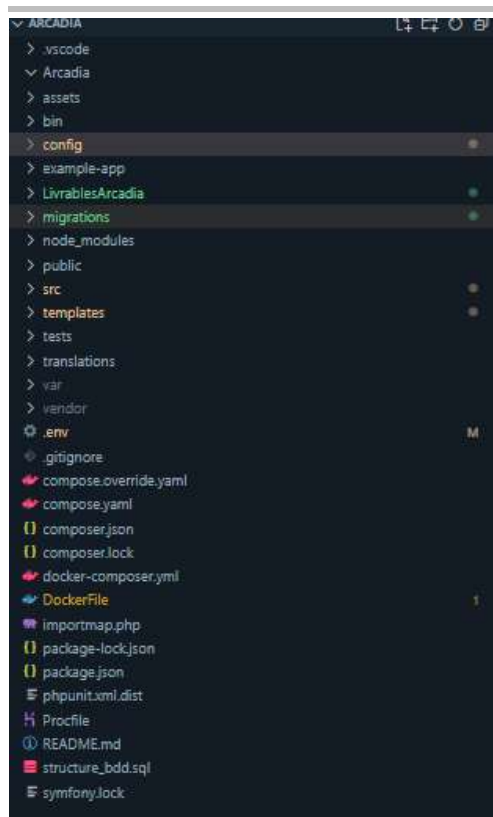
Après avoir configurer, je vérifie si ils sont présent avec `composer -v` pour savoir les version de symfony et php

```
G:\projet code\Arcadia>composer -V
Composer version 2.8.4 2024-12-11 11:57:47
PHP version 8.2.12 (C:\xampp\php\php.exe)
Run the "diagnose" command to get more detailed diagnostics output.
```

Ensuite je peux créer une nouvelle application sous symfony avec la commande `symfony new my_project --version=7.2`

Qui me donnera une structure de travail basique





## Sécurité Symfony

### Formulaire

Afin de contrer les attaques CSFR qui permet qu'un autre attaquant puissent utiliser les droits d'autres utilisateurs de manière détournée à son insu.

Pour contrer ça on va prendre le formulaire pour ajouter un habitat

Avec Symfony, je peux sécuriser les formulaires contre ça avec, premièrement, les classes FormTypes

Grâce à la commande de génération offerte par Symfony, je peux créer un fichier pour conditionner les champs à remplir.

You, 3 months ago | 1 author (You) | 8 references | 0 implementations

```
class HabitatType extends AbstractType
{
    0 references | 0 overrides
    public function buildForm(FormBuilderInterface $builder, array $options): void
    {
        $builder
            ->add(child: 'nom', type: TextType::class, options: [
                'label' => 'Nom de l\'habitat',
                'attr' => [
                    'placeholder' => 'Entrez le nom de l\'habitat',
                    'class' => 'form-control form-floating', // Ajout de la classe form-floating
                ],
                'required' => true, // Champ obligatoire
            ])
            ->add(child: 'description', type: TextareaType::class, options: [
                'label' => 'Description',
                'attr' => [
                    'placeholder' => 'Entrez une description',
                    'class' => 'form-control form-floating', // Ajout de la classe form-floating
                ],
                'required' => true, // Champ obligatoire
            ])
            ->add(child: 'image', type: FileType::class, options: [
                'label' => 'Image de l\'habitat',
                'required' => false, // Le champ image est optionnel
                'mapped' => false, // Nous le gérons séparément dans le contrôleur
                'attr' => [
                    'class' => 'form-control form-floating',
                    'accept' => 'image/png, image/jpeg', // Types d'images acceptés
                ],
            ]);
    }
}
```

Ici, la fonction buildform va configurer le formulaire avec les champs de la table habitat (ici nom, descript et image) configurer les types accepter différents champs du formulaire, placeholder d'afficher un text pour indiquer quoi inscrire. Le fait de déclarer les éléments réduit de pouvoir injecter des éléments

Ensuite, on utilise le FormType dans notre Controller pour

```
// Création d'un nouvel habitat
#[Route(path: '/new', name: 'app_habitat_new', methods: ['GET', 'POST'])]
8 references | 0 overrides
public function new(Request $request, EntityManagerInterface $entityManager, ImageUploadService $imageUploadService): Response
{
    $habitat = new Habitat();
    $form = $this->createForm(type: HabitatType::class, data: $habitat);
    $form->handleRequest(request: $request);

    if ($form->isSubmitted() && $form->isValid()) {
        // Récupérer le fichier d'image
        $imageFile = $form->get(name: 'image')->getData();
        if ($imageFile) {
            // Utiliser le service d'upload avec le type 'habitats'
            $newFilename = $imageUploadService->upload(file: $imageFile, type: 'habitats');

            // Créer ou récupérer l'entité Image
            $image = new Image();
            $image->setPath(path: $newFilename);

            // Associer l'image à l'habitat
            $habitat->setImage(image: $image);
        }

        // Persister l'entité Habitat et l'entité Image
        $entityManager->persist(object: $habitat);
        if (isset($image)) {
            $entityManager->persist(object: $image); // Persister l'image si elle existe
        }
        $entityManager->flush(); // Sauvegarder les données

        $this->addFlash(type: 'success', message: 'Habitat ajouté avec succès.');
```

```
return $this->redirectToRoute(route: 'app_habitat_index');
```

\$*habitat* créer un nouvel objet « habitat »

La 1<sup>er</sup> ligne \$*form* va récupérer le Formtype la seconde traite la soumission et la dernipre va vérifier le formulaire et le soumet pour la validation. \$*entityManager* s'occupe de sauvegarder sur la table. (L'insertion d'image est gérer par un service)

Pour finir on la vue twig qui affiche notre formulaire

```
{{ form_start }}
    <div class="form-floating mb-3">
        {{ form_row(form.nom) }}
    </div>

    <div class="form-floating mb-3">
        {{ form_row(form.description) }}
    </div>

    <div class="form-floating mb-3">
        {{ form_row(form.image) }}
    </div>

    <button type="submit" class="btn btn-primary w-100">Envoyer</button>

    <a href="{{ path('app_utilisateur_index') }}" class="btn btn-secondary">Retour</a>
{{ form_end(form) }}
```

Le fait que L'on utilise des Accolades au lieu de balise permet de sécurisée les données ce qui peut aussi prévenir des attaques XSS .

L'accolade {{form\_start}} permet à de commencer le formulaire html mais surtout d'inclure automatiquement la méthode d'envoi Post, le champ CSRF contre les attaques du même nom.

{{form\_row.(Nom\_du\_champ) }} génère une classe fomulaire sécurisée que j'ai configuré dans le fichier formType.

Enfin {{form\_end}} ferme le formulaire et ajoute un champ CSRF.

Si je peux utiliser la protection CSRF c'est que Symfony à configuer un token unique par formulaire et vérifie que la requetes correspond à la session active.

La configuration est faite dans *framework.yaml*

```
***
framework:
    csrf_protection: true
```

## Docker

L'utilisation de Docker permet de créer un conteneur isolé avec les dépendances d'un projet.

Imaginons que quelqu'un voudrait travailler avec moi sur ce projet mais il n'a pas la même configuration en local.

Il faut configurer le fichier *DockerFile* dans la racine du projet Symfony qui définit l'image de la configuration nécessaire pour les dépendances.

```
you, 2 months ago | 1 author (you)
FROM php:8.2-cli

# Installer les extensions PHP nécessaires
RUN apt-get update && apt-get install -y \
    git \
    unzip \
    libicu-dev \
    libpq-dev \
    libzip-dev \
    && docker-php-ext-install pdo pdo_mysql zip intl

# Installer Composer
COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

# Définir le dossier de travail
WORKDIR /var/www

# Installer Symfony CLI (si nécessaire)
RUN curl -sS https://get.symfony.com/cli/installer | bash \
    && mv /root/.symfony/bin/symfony /usr/local/bin/symfony

# Exposer le port Symfony
EXPOSE 8000

CMD ["php", "-S", "0.0.0.0:8000", "-t", "public"]
```

Le fichier Docker va lui permettre d'avoir toutes les dépendances Symfony PHP ainsi que les bases de données SQL et NoSQL sur son environnement.

```

You, 2 months ago | 1 author (You)
version: '3.8'

▶ Run All Services
services:
  ### --- Serveur Web avec PHP ---
  ▶ Run Service
  app:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: symfony_app
    restart: unless-stopped
    working_dir: /var/www
    volumes:
      - ../var/www
    depends_on:
      - mysql
      - mongodb
    environment:
      APP_ENV: dev
      DATABASE_URL: mysql://root:root@mysql:3306/symfony?serverVersion=8.0
      MONGODB_URL: mongodb://mongodb:27017
    ports:
      - "8000:8000"

  ### --- Base de données MySQL ---
  ▶ Run Service
  mysql:
    image: mysql:8.0
    container_name: symfony_mysql
    restart: unless-stopped
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: symfony
    ports:
      - "3307:3306"
    volumes:
      - mysql_data:/var/lib/mysql

  ### --- Base de données MongoDB ---
  ▶ Run Service
  mongodb:
    image: mongo:6.0
    container_name: symfony_mongodb
    restart: unless-stopped
    ports:
      - "27017:27017"
    volumes:
      - mongodb_data:/data/db

```

## Git et Github

Pour que mon code puisse avoir un suivi des modification et de son évolution je vais devoir utiliser Git qui est l'outil parfait pour ça, je vais le combiner avec Github qui va me permettre d'héberger un dépôt ou le code sera stocké et versionné accompagné d'un commentaire expliquant le travail effectué.

## Configuration

### 1. création d'un repository

Sur Github je crée un nouveau dépôt avec le nom du projet, si il doit être public ou non et ajouter un read me pour les informations du projet.

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

Owner \*

Repository name \*

LucasCherbuin

/

Great repository names are short and memorable. Need inspiration? How about [symmetrical-octo-broccoli](#) ?

Description (optional)

☒

Public

Anyone on the internet can see this repository. You choose who can commit.

☐

Private

You choose who can see and commit to this repository.

Initialize this repository with:

☐

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

You are creating a public repository in your personal account.

Create repository

Après la création, cela va généré un dépôt (Repository) et me créer une url qui me servira pour coonnecter le dépôt local .

## 2. dépôt local

Je dois créer le dépôt local avec une suite de commande :

« *git init* » va créer le dépôt local

« *git add .* » va ajouter les fichier de symfony de base de symfony

« *Git commit -m* initiation du projet » va enregistrer le travaille effectué (-m permet de laisser un commentaire sur le travaille effectué)

### 3. Connecter le dépôt local et celui de github

Maintenant que l'on a configuré les dépôts, on va les connecter.

Dans le terminal je vais marqué

```
git remote add origin git remote add origin https://github.com/ton-utilisateur/mon-projet-symfony.git
```

Qui va associer le dépôt local à celui de github

### 4. Faire le 1<sup>er</sup> push

Maintenant la connexion réalisée, on peut pousser le travail réalisé en local sur Github

```
git push -u origin master
```

Cela va stocker le squelette de mon projet sur la branch Master du dépôt Github

### 5. Configuration des branches

Afin de répartir le travail, je vais configurer des branches qui chacune correspond à une fonctionnalité. Je crée une nouvelle branche avec `git checkout -b nom-de-la-nouvelle-branche` et je serais basculer dans cette branche.

Master	C'est la branche principal, c'est elle qui contient le travail fini et tester.  Cette branche servira lors du déploiement en ligne de l'application
Sous branche de ^	
Develop	C'est ici où les fonctionnalités merge seront tester avec les autres branches
Sous branche de ^	
Features/visiteur	Fonctionnalité liée aux pages des visiteurs
Features/administrateur	Fonctionnalité liée à l'administrateur
Features/employe	Fonctionnalité liée aux employés
Features/vetinaire	Fonctionnalité liée aux vétérinaires
Features/Email test	Test relatif au mailer
Features/Config	Travail lié à la configuration des logins
Features/Datafixtures	Création de données factices pour tester les tables de la base de données
Features/AdminCréation	Permet de créer un compte administrateur avec un mot de passe sécurisé

Le fait de répartir les branches permet de réaliser des tâches spécifiques sans affecter le travail principal.

Si J'ai préciser le principe de sous branche c'est que, dès le travail terminée, je vais *merge*, c'est a dire fusionner une branche avec une autre afin d'y apporter les modifications.

Je refais le processus de commande vue dans le point « 2 » et « 3 » pour enregistrer et pousser le travail sur le github principal qui aura stocker le code sous la branche correspondante.

Maintenant j'utilise la commande `git switch develop` pour accéder à la branch de développement pour fusionner les branches, avant tout, j'utilise la commande *git branch* pour vérifier que je me trouve bien sur la bonne branche.

Pour finir j'utilise *git merge nom\_de\_la\_fonctionnalité* pour importer le travail de la branche sur celle de developpment.

## 2. Précisez les moyens utilisés :

Pour configurer mon environnement, j'ai utiliser Symfony et composer pour le framework, docker pour contenir les mêmes dépendances ainsi que Github pour stocker et versionner mon projet et Git pour le suivi et sauvegare de mon code.

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet

## 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **30/10/24** au : 08/02/25

## 5. Informations complémentaires (facultatif)



## Activité-type 2

## Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n°5 ► Mettre en place une base de données relationnelles

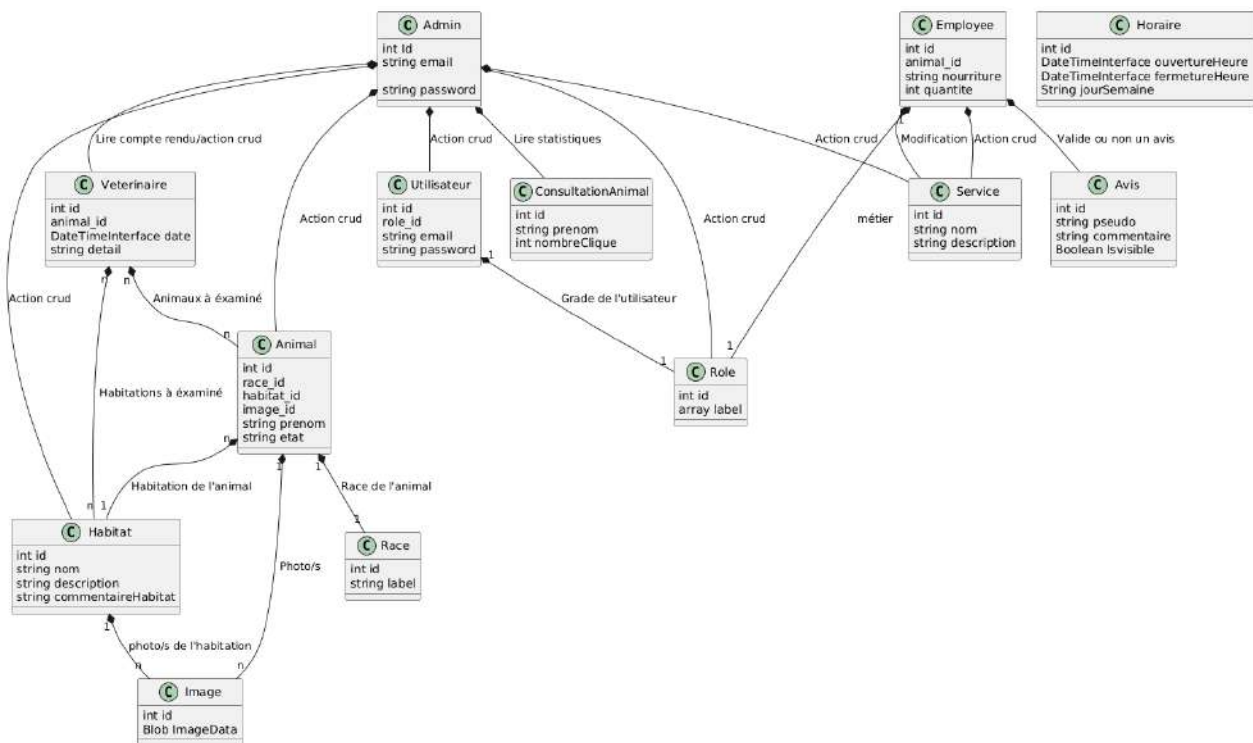
### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour que le site web d'Arcadia puisse échanger avec les données de la base de données, j'ai dû réaliser des tâches de maquettage.

Les données concernent les services proposés par Arcadia aux visiteurs, ainsi que les utilisateurs et leurs activités au sein du zoo.

Voici le procédé :

#### 1. Diagramme de classes



Le schéma ci-dessus illustre l'interaction entre les différentes tables ainsi que le type de relation entre elles.

- Les relations **1–1** (*one to one*) indiquent qu'une table dépend d'une seule autre (ex. : 1 rôle pour 1 utilisateur, 1 race pour 1 animal).
- Les relations **1–M** ou **M–1** (*one to many* et *many to one*) indiquent qu'une table peut dépendre de plusieurs entrées dans une autre table, et inversement (ex. : 1 habitat peut contenir plusieurs animaux, plusieurs images peuvent être attribuées à 1 animal).
- Les relations **M–M** (*many to many*) signifient que plusieurs enregistrements d'une table peuvent être liés à plusieurs enregistrements d'une autre (ex. : plusieurs utilisateurs ayant le rôle de vétérinaire peuvent laisser une opinion sur plusieurs habitats ; plusieurs animaux peuvent être examinés par plusieurs vétérinaires).

## 2. Dictionnaire des données

Afin d'avoir un aperçu des champs mes tables, j'ai réalisé un tableaux indiquant son nom, le type définit, les détails ainsi qu'une description de son utilisation

Champ	Type	Détail	Description
Utilisateur			
Id	INT	Primary key, Not null	Identifiant d'un utilisateur
Email	Varchar	Not null	Email de l'utilisateur
Password	Varchar	Not null	Mot de passe haché
Role_id	Int	Foreing key, Not null	Relation avec la table rôle
Animal			
Id	INT	Primary key, Not null	Identifiant d'un animal
Race_id	INT	Foreing key, Not null	Relation avec la table animal
Habitat_id	INT	Foreing key, Not null	Relation avec la table habitat
Image_id	INT	Foreing key, Not null	Relation avec la table image
Prenom	VARCHAR	Not null	Prénom de l'animal
État	VARCHAR	Null	État de santé de l'animal
Vétérinaires	INT	Foreing key, Not null	Relation avec la table vétérinaire
Avis			
Id	INT	Primary key, Not null	Identifiant d'un avis

Pseudo	VARCHAR	Not null	Pseudo que le visiteur à laisser dans le formulaire
Commentaire	VARCHAR	Not null	Commentaire que le visiteur à laisser dans le formulaire
Is visible	Boolean	Not null	Si le commentaire est validé par un employé, il sera visible sur la page d'accueil
Commentaire			
Id	INT	Primary key, Not null	Identifiant d'un commentaire
Commentaire	VARCHAR	Primary key, Not null	Texte laisser par un vétérinaire sur l'état d'un habitat
Habitat_id	INT	Foreing key, Not null	Relation avec la table habitat
Utilisateur_id	INT	Foreing key, Not null	Relation avec la table utilisateur
Employée			
Id	INT	Primary key, Not null	Identifiant d'un Employée
Nourriture	VARCHAR	Not null	Aliment donnée à un animal
Quantité	INT	Not null	Quantité en piece
Utilisateur_id	INT	Foreing key, Not null	Relation avec la table utilisateur
Habitat			
Id	INT	Primary key, Not null	Identifiant d'un habitat
Image_id	INT	Foreing key, Not null	Relation avec la table image
Nom	VARCHAR	Not null	Nom de l'habitat
Description	VARCHAR	Not null	Description de l'habitat
Horaire			
Id	INT	Primary key, Not null	Identifiant d'un horaire

Jour	VARCHAR	Not null	Jour de la semaine
Ouverture	ITEM	Not null	Heure d'ouverture
Fermeture	TIME	Not null	Heure de fermeture
Image			
Id	INT	Primary key, Not null	Identifiant d'une image
Path	VARCHAR	Not null	Chemin de l'image enregistré sur le projet
Race			
Id	INT	Primary key, Not null	Identifiant d'une race
Label	VARCHAR	Not null	Race attribué
Role			
Id	INT	Primary key, Not null	Identifiant d'une role
Label	VARCHAR	Not null	Role attribué
Service			
Id	INT	Primary key, Not null	Identifiant d'un service
Nom	VARCHAR	Not null	Nom du service
Description	VARCHAR	Not null	Description du service
Image_id	INT	Foreing key, Not null	Relation avec la table image
Vétérinaire			
Id	INT	Primary key, Not null	Identifiant d'un vétérinaire
Animal_id	INT	Foreing key, Not null	Relation avec la table animal
Date	DATE	Not null	Date du rapport
Détail	VARCHAR	Not null	Détail du rapport
Utilisateur_id	INT	Foreing key, Not null	Relation avec la table utilisateur

---

---

---

## 2. Précisez les moyens utilisés :

Pour la réalisation de la base de données, j'ai utilisé le site PlantUML, qui me permet de créer le schéma à l'aide de lignes de code

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet

## 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du :01/10/2024 Au : 08/02/2025

## 5. Informations complémentaires (facultatif)

## Activité-type 2

## Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n° 2 ► CP 6 Développer des composants d'accès aux données SQL et NoSQL

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

#### 3. Configuration dans le projet

Maintenant que tout le schéma et le dictionnaire sont réalisés, on peut configurer la base de données dans le projet.

##### 1. Configuration des bases de données

Dans le fichier .env, j'initie les deux adresses des bases de données.

```
# Remplacer les valeurs  
MAILER_DSN=mailjet://c  
###< symfony/framework  
###> doctrine/doctrine  
# Connexion à la base  
DATABASE_URL=mysql://c
```

On teste si l'adresse MySQL fonctionne en créant une base de données à cette adresse avec la commande :

*Bash CopierModifier php bin/console doctrine:database:create*

Ensuite, j'utilise la commande suivante pour intégrer **MongoDB** (NoSQL) dans le projet :

*Bash CopierModifier composer require doctrine/mongodb-odm-bundle*

Puis, je vais dans le fichier config/packages/doctrine\_mongodb.yaml pour configurer la base de données NoSQL.

```
You, 4 hours ago | 1 author (You)  
doctrine_mongodb:  
  connections:  
    default:  
      server: "%env(resolve:MONGODB_URL)%"  
      options: {}  
  default_database: "%env(resolve:MONGODB_DB)%"  
  document_managers:  
    default:  
      auto_mapping: false  
      mappings:  
        Consultation:  
          type: attribute  
          dir: '%kernel.project_dir%/src/Document'  
          prefix: 'App\Document'  
          alias: App
```

Après avoir configuré la base de données dans le fichier .env, j'utilise la commande *php bin/console make:entity* pour générer les entités et pouvoir configurer selon les valeurs définies.

(la table **Utilisateur** va être générée avec des options spécifiques notamment de sécurité des mots de passes .)

## 2. Entités

```
You, 3 months ago | 1 author (You)
#[ORM\Entity(repositoryClass: UtilisateurRepository::class)]
You, 3 months ago | 1 author (You) | 67 references | 1 implementation
class Utilisateur implements UserInterface, PasswordAuthenticatedUserInterface
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    1 reference
    private ?int $id = null;

    #[ORM\Column(length: 180, unique: true)]
    3 references
    private ?string $email = null;

    #[ORM\ManyToOne(targetEntity: Role::class, inversedBy: 'utilisateur')]
    #[ORM\JoinColumn(nullable: false)]
    4 references
    private ?Role $role = null;

    #[ORM\Column]
    2 references
    private ?string $password = null;

    4 references | 0 overrides
    public function getId(): ?int
    {
        return $this->id;
    }

    1 reference | 0 overrides
    public function getEmail(): ?string
    {
        return $this->email;
    }

    2 references | 0 overrides
    public function setEmail(string $email): self
    {
        $this->email = $email;

        return $this;
    }
}
```

```
2 references | 0 overrides
public function getRole(): ?Role
{
    return $this->role;
}

4 references | 0 overrides
public function setRole(?Role $role): self
{
    $this->role = $role;

    return $this;
}

2 references | 0 overrides
public function getPassword(): ?string
{
    return $this->password;
}

6 references | 0 overrides
public function setPassword(string $password): self
{
    $this->password = $password;

    return $this;
}
```

Il faut configurer les entités, notamment en y incluant les champs nécessaires, les relations entre les tables, ainsi que des options de sécurité si besoin (ex. : encodage des mots de passe, gestion des rôles, etc.).

### 3. Migration

Maintenant, on peut migrer les tables vers notre base de données en exécutant les commandes

suivantes :PHP bin/console doctrine:migrations

Malgrès que symfony ma générer une migration sous forme de SQL Je décide de modifier le code en php afin de m'exposer à des injection Sql involontaires

```
final class Version20250403 extends AbstractMigration
{
    0 references
    public function getDescription(): string
    {
        return 'Ajout des tables Habitat, Employee, Race, Role, Service, Utilisateur et Veterinaire avec leurs relations.';
    }

    0 references
    public function up(Schema $schema): void
    {
        $this->addSql(sql: 'CREATE TABLE habitat (
            id INT AUTO_INCREMENT NOT NULL,
            nom VARCHAR(100) NOT NULL,
            description VARCHAR(255) NOT NULL,
            image_id INT DEFAULT NULL,
            PRIMARY KEY(id)
        ) DEFAULT CHARACTER SET utf8mb4 COLLATE `utf8mb4_unicode_ci` ENGINE = InnoDB');
        $this->addSql(sql: 'ALTER TABLE habitat ADD CONSTRAINT FK_HABITAT_IMAGE FOREIGN KEY (image_id) REFERENCES image (id)');

        // Création de la table "employee"
        $this->addSql(sql: 'CREATE TABLE employee (
            id INT AUTO_INCREMENT NOT NULL,
            nourriture VARCHAR(50) NOT NULL,
            quantite INT NOT NULL,
            date DATETIME NOT NULL, a
            utilisateur_id INT DEFAULT NULL,
            PRIMARY KEY(id)
        ) DEFAULT CHARACTER SET utf8mb4 COLLATE `utf8mb4_unicode_ci` ENGINE = InnoDB');
        $this->addSql(sql: 'ALTER TABLE employee ADD CONSTRAINT FK_EMPLOYEE_UTILISATEUR FOREIGN KEY (utilisateur_id) REFERENCES utilisateur (id)');

        $this->addSql(sql: 'CREATE TABLE race (
            id INT AUTO_INCREMENT NOT NULL,
            nom VARCHAR(100) NOT NULL,
            description VARCHAR(255) NOT NULL,
            PRIMARY KEY(id)
        ) DEFAULT CHARACTER SET utf8mb4 COLLATE `utf8mb4_unicode_ci` ENGINE = InnoDB');
        $this->addSql(sql: 'ALTER TABLE race ADD CONSTRAINT FK_RACE_UTILISATEUR FOREIGN KEY (utilisateur_id) REFERENCES utilisateur (id)');
    }
}
```

Utiliser des shemas php permet de garder une architecture cohérent, sécurisé et conforme à Symfony

```
<?php
declare(strict_types=1);

namespace DoctrineMigrations;

use Doctrine\DBAL\Schema\Schema;
use Doctrine\Migrations\AbstractMigration;

0 references
final class Version20250403 extends AbstractMigration
{
    0 references
    public function getDescription(): string
    {
        return 'Ajout des tables Habitat, Employee, Race, Role, Service, Utilisateur et Veterinaire avec leurs relations (version sécurisée Doctrine schema builder).';
    }

    0 references
    public function up(Schema $schema): void
    {
        // Table: habitat
        $habitat = $schema->createTable(name: 'habitat');
        $habitat->addColumn(name: 'id', typeName: 'integer', options: ['autoincrement' => true]);
        $habitat->addColumn(name: 'nom', typeName: 'string', options: ['length' => 100]);
        $habitat->addColumn(name: 'description', typeName: 'string', options: ['length' => 255]);
        $habitat->addColumn(name: 'image_id', typeName: 'integer', options: ['nullable' => false]);
        $habitat->setPrimaryKey(columnNames: ['id']);
        $habitat->addForeignKeyConstraint(foreignTable: 'image', localColumnNames: ['image_id'], foreignColumnNames: ['id']);

        // Table: employee
        $employee = $schema->createTable(name: 'employee');
        $employee->addColumn(name: 'id', typeName: 'integer', options: ['autoincrement' => true]);
        $employee->addColumn(name: 'nourriture', typeName: 'string', options: ['length' => 50]);
        $employee->addColumn(name: 'quantite', typeName: 'integer');
        $employee->addColumn(name: 'date', typeName: 'datetime');
        $employee->addColumn(name: 'utilisateur_id', typeName: 'integer', options: ['nullable' => false]);
        $employee->setPrimaryKey(columnNames: ['id']);
        $employee->addForeignKeyConstraint(foreignTable: 'utilisateur', localColumnNames: ['utilisateur_id'], foreignColumnNames: ['id']);

        // Table: race
        $race = $schema->createTable(name: 'race');
        $race->addColumn(name: 'id', typeName: 'integer', options: ['autoincrement' => true]);
        $race->addColumn(name: 'nom', typeName: 'string', options: ['length' => 100]);
        $race->addColumn(name: 'description', typeName: 'string', options: ['length' => 255]);
        $race->addColumn(name: 'utilisateur_id', typeName: 'integer', options: ['nullable' => false]);
        $race->setPrimaryKey(columnNames: ['id']);
        $race->addForeignKeyConstraint(foreignTable: 'utilisateur', localColumnNames: ['utilisateur_id'], foreignColumnNames: ['id']);
    }
}
```



En consultant PHPmyAdmin on peut voir le résultat de la migration


De plus, grâce à l'option *exporté*, je peux créer un fichier SQL indiquant les commandes réaliser sur la bdd. Ce qui permet de voir que mes commandes php ont permis de réaliser des commande SQL sur ma base de données

```
-- Active Connection
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Hôte : 127.0.0.1:3306
-- Généré le : mar. 22 avr. 2025 à 17:35
-- Version du serveur : 10.4.32-MariaDB
-- Version de PHP : 8.2.12

▷ Run
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
▷ Run
START TRANSACTION;
▷ Run
SET time_zone = "+00:00";

▷ Run
/*140101 SET @OLD_CHARACTER_SET_CLIENT=@CHARACTER_SET_CLIENT */;
▷ Run
/*140101 SET @OLD_CHARACTER_SET_RESULTS=@CHARACTER_SET_RESULTS */;
▷ Run
/*140101 SET @OLD_COLLATION_CONNECTION=@COLLATION_CONNECTION */;
▷ Run
/*140101 SET NAMES utf8mb4 */;

--
-- Base de données : "arcadia"
--
.....

--5
-- Structure de la table "animal"
--
|
▷ Run |  Select
CREATE TABLE `animal` (
  `id` int(11) NOT NULL,
  `race_id` int(11) NOT NULL,
  `habitat_id` int(11) NOT NULL,
  `image_id` int(11) DEFAULT NULL,
  `prenom` varchar(50) NOT NULL,
  `etat` varchar(50) NOT NULL,
  `veterinaire_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

---

## Utilisation de SQL

Si pour X ou Y raison, je dois passer par MySQL pour gérer des données de l'application, il faudra utiliser des commandes SQL

### *Insérer dans une table*

```
1 INSERT INTO `veterinaire` (`id`, `animal_id`, `date`, `detail`, `utilisateur_id`)
2 VALUES ('10', '1', '2025-04-08', 'Besoin de repos', '1');
```

**Insert Into** me permet de sélectionner la table (vétérinaire) ainsi que les champs à remplir ('id', 'animal\_id', etc.)

**Values** me permet d'insérer des données selon les types acceptés, les tables relationnelles sont sélectionnées selon leurs numéros d'id.

### *Créer une table*

```
Run | Select
CREATE TABLE dessin (
  id INT PRIMARY KEY AUTO_INCREMENT,
  prenom VARCHAR(10) NOT NULL,
  dessin BLOB NOT NULL,
  animal_id INT,
  FOREIGN KEY (animal_id) REFERENCES animal(id)
);
```

Si le zoo organise un concours de dessin et que je crée une table, il faudra que j'utilise **Create Table** nommé dessin et que j'indique des informations

**Primary Key** permet de rendre la colonne ID comme identifiant **AUTO\_INCREMENT** pour incrémenter automatiquement à chaque nouvel enregistrement.

Pour la colonne *prenom* et *dessin* on indique le type comme **varchar(10)** indiquant une chaîne de caractères limitée à 10 caractères et **blob** pour stocker des fichiers binaires (exemple, les png des dessins scannés).

**Foreign Key** permet d'indiquer que *animal\_id* est une clé étrangère et avec **REFERENCES** d'indiquer que c'est la table *animal* et le champ *id*

---

---

### 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet

### 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01/10/24** Au : **08/02/25**

### 5. Informations complémentaires *(facultatif)*

## Activité-type 2

## Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n° 3 ► CP 7 Développer des composants métier coté serveur

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le projet **Arcadia**, j'utilise **Symfony**, un framework spécialisé dans le back-end, facilitant le travail côté architecture logicielle.

#### Organisation de l'architecture

(Exemple avec une entité Utilisateur)

##### 1. Controller

Le **Controller** sert à gérer les requêtes HTTP et les réponses entre la **vue** (interface utilisateur) et les **modèles** (entités et données de la base de données).

Étant donné que j'ai généré une entité liée aux utilisateurs, un **Controller sécurisé** a également été généré.

```
#[Route(path: '/new', name: 'app_utilisateur_new', methods: ['GET', 'POST'])]
8 references | 0 overrides
public function new(Request $request, UserPasswordHasherInterface $passwordHasher, MailerService $mailerService): Response
{
    $utilisateur = new Utilisateur();
    $form = $this->createForm(type: RegistrationFormType::class, data: $utilisateur);
    $form->handleRequest(request: $request);

    if ($form->isSubmitted() && $form->isValid()) {
        $role = $form->get(name: 'role')->getData();
        $utilisateur->setRole(role: $role);

        $hashedPassword = $passwordHasher->hashPassword(user: $utilisateur, plainPassword: $form->get(name: 'plainPassword')->getData());
        $utilisateur->setPassword(password: $hashedPassword);

        $this->entityManager->persist(object: $utilisateur);
        $this->entityManager->flush();

        $this->addFlash(type: 'success', message: 'Utilisateur créé avec succès.');
```

```
        $emailPersonnel = $form->get(name: 'emailPersonnel')->getData();
        try {
            $mailerService->sendEmail(
                from: 'admin@arcadia.fr',
                to: $emailPersonnel,
                subject: 'Votre compte Arcadia Zoo est prêt !',
                htmlBody: '<p>Bonjour,</p><p>Votre compte sur Arcadia Zoo est maintenant prêt et sera nommé <strong>{'$form->getEmail()}</strong>.</p>';
            );
        } catch (\Exception $e) {
            $this->addFlash(type: 'error', message: 'Une erreur est survenue lors de l\'envoi de l\'email de confirmation.');
```

```
        }

        return $this->redirectToRoute(route: 'app_utilisateur_index');
    }

    return $this->render(view: 'admin/utilisateur/new.html.twig', parameters: [
        'form' => $form->createView(),
    ]);
}
```

You, 3 months ago • ajout des fonctionnalités du menu admin

Le fichier générer propose les 4 opérations du CRUD basique avec des spécification relatif à l'utilisateur

Le controller pour les utilisateur est géré par l'administrateur pour concevoir les utilisateurs vétérinaire et employé du zoo, j'ai intégrer l'instance *\$emailpersonnel* qui envoie un mail (le mail personnel du nouvel utilisateur) automatiquement au nouvel utilisateur que son compte pour l'application est prêt et qu'il va devoir contacter son employeur pour son mot de passe.

## 2. FormType

```
class RegistrationFormType extends AbstractType
{
    0 references | 0 overrides
    public function buildForm(FormBuilderInterface $builder, array $options): void
    {
        $builder
            ->add(child: 'email', type: EmailType::class, options: [
                'constraints' => [
                    new NotBlank(options: ['message' => 'Please enter an email address']),
                    new Email(options: ['message' => 'Please enter a valid email address']),
                ],
                'attr' => ['class' => 'form-control form-floating']
            ])
            ->add(child: 'role', type: EntityType::class, options: [
                'class' => Role::class,
                'choice_label' => 'role',
                'query_builder' => function (RoleRepository $roleRepository): QueryBuilder {
                    return $roleRepository->createQueryBuilder(alias: 'r')
                        ->where(predicates: 'r.role IN (:roles)')
                        ->setParameter(key: 'roles', value: ['ROLE_EMPLOYEE', 'ROLE_VETERINAIRE']);
                },
                'multiple' => false,
                'expanded' => false,
                'attr' => ['class' => 'form-control'],
            ])
            ->add(child: 'plainPassword', type: PasswordType::class, options: [
                'mapped' => false,
                'attr' => ['autocomplete' => 'new-password', 'class' => 'form-control form-floating'],
                'constraints' => [
                    new NotBlank(options: ['message' => 'Please enter a password']),
                    new Length(exactly: [
                        'min' => 6,
                        'minMessage' => 'Your password should be at least {{ limit }} characters',
                        'max' => 4096,
                    ])
                ]
            ])
            ->add(child: 'emailPersonnel', type: EmailType::class, options: [
                'label' => 'Email personnel',
                'mapped' => false, // Ne pas mapper à une propriété de l'entité
                'attr' => [
                    'placeholder' => 'Indiquer l\'email personnel du nouvel utilisateur'
                ]
            ])
        ];
    }
}
```

Le fichier RegisteredFormType permet de structurer les formulaires afin de préciser les type des champs et imposer des valeurs par défaut.

Cela garantit que les tables soit sécurisé et correctement remplis par l'utilisateur.

### 3. Repository

```
You, 3 months ago | 1 author (You)
<?php

namespace App\Repository;

use App\Entity\Utilisateur;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\Persistence\ManagerRegistry;
use Symfony\Component\Security\Core\Exception\UnsupportedUserException;
use Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
use Symfony\Component\Security\Core\User\PasswordUpgraderInterface;

You, 3 months ago | 1 author (You)
/**
 * @extends ServiceEntityRepository<Utilisateur>
 */
You, 3 months ago | 1 author (You) | 5 references | 0 implementations
class UtilisateurRepository extends ServiceEntityRepository implements PasswordUpgraderInterface
{
    16 references | 0 overrides
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, Utilisateur::class);
    }

    /**
     * Used to upgrade (rehash) the user's password automatically over time.
     */
    0 references | 0 overrides
    public function upgradePassword(PasswordAuthenticatedUserInterface $user, string $newHashedPassword): void
    {
        if (!$user instanceof Utilisateur) {
            throw new UnsupportedUserException(sprintf('Instances of "%s" are not supported.', $user::class));
        }

        $user->setPassword($newHashedPassword);
        $this->getEntityManager()->persist($user);
        $this->getEntityManager()->flush();
    }

    // App\Repository\UtilisateurRepository.php
    You, 3 months ago - correction du cas pour la validation des avis, ...
    // App\Repository\UtilisateurRepository.php

    1 reference | 0 overrides
    public function findByRole(string $role): array
    {
        return $this->createQueryBuilder(alias: 'u')
            ->innerJoin(join: 'u.role', alias: 'r') // Rejoins la relation 'roles' de l'entité 'Utilisateur'
            ->andWhere(where: 'r.role = :role') // Assure-toi que 'r.role' est bien la propriété qui contient le rôle dans l'entité 'Role'
            ->setParameter(key: 'role', value: $role) // Associe le paramètre à la variable $role
            ->getQuery()
            ->getResult();
    }
}
```

C'est le fichier qui va interagir avec la base de données avec Doctrine, l'utilisation de *QueryBuilder* va réaliser des requêtes Sql sous forme de php afin de prévenir des injections.

---

## 2. Services

Les services permettent de séparer une classe réutilisable et d'isoler une fonctionnalité afin d'améliorer la lisibilité du code et de faciliter les tests unitaires.

J'ai dû réaliser un service pour le **mailer**.

```
//service pour la gestion des informations relatif aux envoies d'email
You, 3 months ago | 1 author (You) | 9 references | 0 implementations
class MailerService
{
    2 references
    private MailerInterface $mailer;

    3 references | 0 overrides
    public function __construct(MailerInterface $mailer)
    {
        $this->mailer = $mailer;
    }

    2 references | 0 overrides
    public function sendEmail(
        string $from,
        string $to,
        string $subject,
        string $htmlBody = '',
        string $textBody = ''
    ): void {
        $email = (new Email())
            ->from(addresses: new Address(address: $from))
            ->to(addresses: new Address(address: $to))
            ->subject(subject: $subject);

        if ($htmlBody) {
            $email->html(body: $htmlBody);
        }

        if ($textBody) {
            $email->text(body: $textBody);
        }

        $this->mailer->send(message: $email);
    }
}
```

Le **mailer** servira à confirmer l'envoi d'une demande de contact, ainsi qu'à confirmer la création d'un compte utilisateur comme expliqué dans le controller de création d'un utilisateur.



## Data Fixtures

Les **DataFixtures** permettent de remplir une base de données de test afin de vérifier si notre base fonctionne correctement avec des données réalistes.

Symfony propose un bundle pour leur gestion :

Composer require --dev doctrine/doctrine-fixtures-bundle

```
You, 3 months ago | 1 author (You) | 3 references | 0 implementations
class UtilisateurFixtures extends Fixture
{
    2 references | 0 overrides
    public function __construct(private UserPasswordHasherInterface $passwordHasher)
    {
    }

    /** @throws Exception */
    0 references | 0 overrides
    public function load(ObjectManager $manager): void
    {
        $faker = Factory::create();

        for ($i = 1; $i <= 20; $i++) {
            $utilisateur = (new Utilisateur())
                ->setEmail(email: $faker->email());

            $utilisateur->setPassword(password: $this->passwordHasher->hashPassword(user: $utilisateur, plainPassword: 'password' . $i));

            $manager->persist(object: $utilisateur);
        }
        $manager->flush();
    }

    0 references | 0 overrides
    public function getDependencies(): array
    {
        return [UtilisateurFixtures::class,
    ];
    }
}
```

L'utilisation de faker permet de création des valeurs réalistes selon le champs et le types

La boucle for me permet de configurer le nombre de colonnes à remplir dans la table de la base de données.

Après avoir fini de configure, j'exécute *php bin/console doctrine:fixtures:load* qui va remplir la base de donnée.

	email	password
Éditer Copier Supprimer	armstrong.alva@example.com	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	collins.reta@example.net	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	dbogisich@example.net	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	virginie.sporer@example.com	\$2y\$12\$V7UijEWnRB.uRGVdqd7COJ0jShrzTQsqoxbhJiGobH...
Éditer Copier Supprimer	providenci.balistreri@example.net	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	reggie.windler@example.org	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	dickinson.mireille@example.com	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	denesik.afton@example.org	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	greyson.effertz@example.org	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	meta44@example.org	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...
Éditer Copier Supprimer	katrine.lehner@example.com	\$2y\$12\$Um9DUbj6JlBabTuTvorkfOrmBWhJdZxmYnMSq0bgKa7...



---

---

---

(Précision, j'ai configuré une autre base de données pour les dans le fichier des variables d'environnement .env pour ne pas affecter la base principal)

## 2. Précisez les moyens utilisés :

J'utilise les différents outils proposés par Symfony, ainsi que Composer, pour exécuter les commandes.

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet

---

---

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01/10/24** Au : 08/02/24

#### 5. Informations complémentaires *(facultatif)*

---

## Activité-type 2 Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°4 ► CP 8 Déploiement d'une application

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Maintenant que mon projet est terminé et tester, je peux le déployer pour qu'il soit fonctionnel.

#### *Besoin et choix de l'hébergeur*

Pour que mon site web soit utilisable par tous, je me suis tourné vers Heroku ayant une structure et plusieurs option.

Mon choix sur la proposition d'une double authentification, garantissant une sécurité pour y accéder, le fait que des services comme le mailer et le stockage y sont proposés dans les add-on et surtout de pouvoir publier le projet depuis git hub.

#### *Besoin*

Ayant travaillé en local pour développer, je dois maintenant utiliser des bases de données en cloud et y intégrer un mailer.

Voici ce que j'ai pu trouver

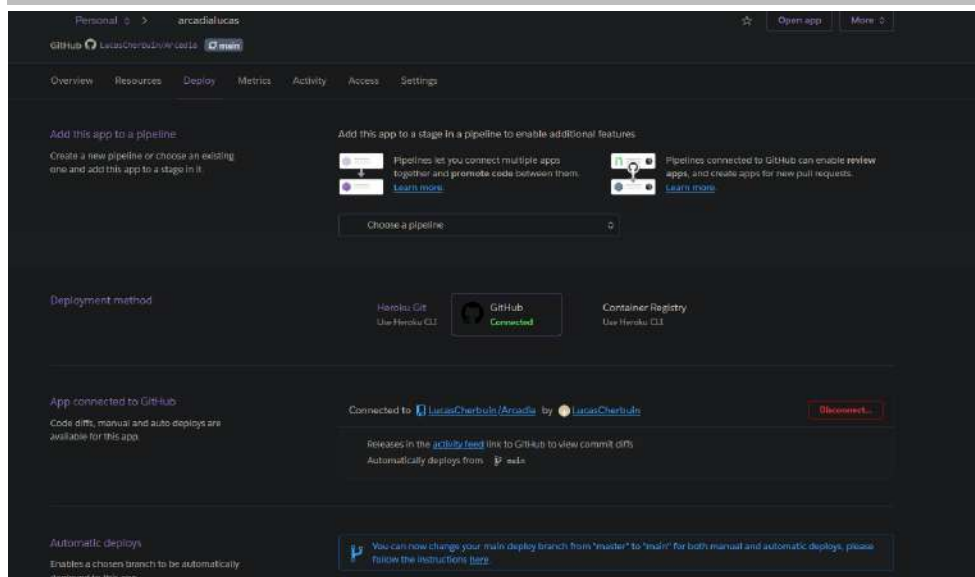
besoin	Nom	Prix (par mois)
Nosql	MongoDb	gratuit
Mysql	JawDb MySql plan Leopard Shared	gratuit
MailerStrap	MailTrap	gratuit

Les options gratuites sont parfaites car mon projet n'atteint pas les capacités de stockage demandées. (Cependant, si mon application a des visites, je vais devoir investir pour avoir plus de places).

Les options seront indiquées dans les ressources et les informations sensibles dans Config Vars.

#### *Déploiement*

Pour publier mon projet, Heroku propose de connecter le dépôt utilisé par mon projet pour le publier



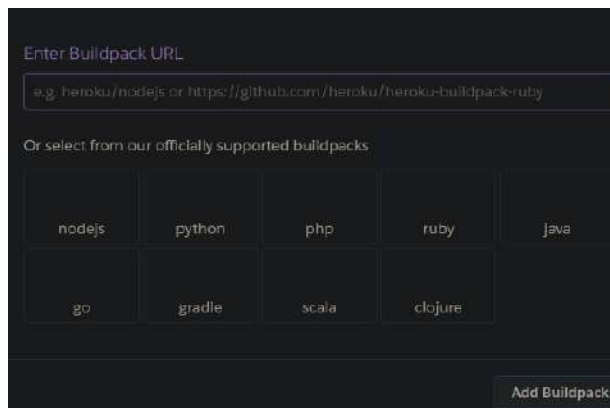
Cela va publier le projet en ligne

### Configuration

Le projet, ne sera pas fonctionnel car il faudra configurer les dépendances

### BuildPack

Les buildsPacks sont des scripts servant à configurer les dépendance du projet



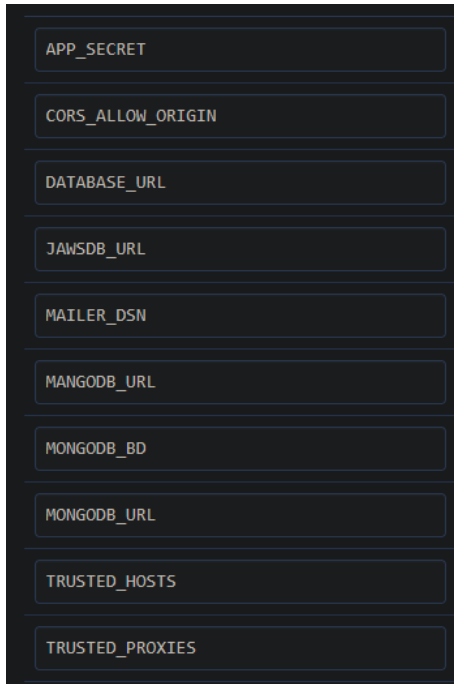
Utilisant Symfony, j'ai configurer le packet PHP

### Config Vars

C'est ici ou on configure les variables de configuration, en effet les informations sensibles ne doivent pas être publiées sur des dépôts Git (le fichier .env)

Pour que mon projet puisse aller récupérer les données de mon projet j'ai configuré un fichier .env.prod qui ira

chercher les informations dans Heroku.



On notera que les add-on configurer avant sont aussi stocker ici

Enfin, le fichier .env.prod

```
# .env.prod

# Environnement de l'application
APP_ENV=prod
APP_SECRET=${APP_SECRET}
APP_DEBUG=0

# Configuration des proxys et hôtes de confiance
# Dans un environnement de production, vous devrez configurer les hôtes et proxies de manière stricte
TRUSTED_PROXIES=${TRUSTED_PROXIES}
TRUSTED_HOSTS=${TRUSTED_HOSTS}

###> symfony/framework-bundle ###
# Configuration du serveur SMTP pour envoyer des emails
# Remplacez les valeurs par vos informations SMTP spécifiques
MAILER_DSN=${MAILER_DSN}
###< symfony/framework-bundle ###

###> doctrine/doctrine-bundle ###
# Connexion à la base de données
DATABASE_URL=${DATABASE_URL}
```

Les syntaxes \${nom\_de\_l'url} ira récupérer les données sensible dans le config vars au nom indiqué

Après avoir configuré le projet, je peux ouvrir l'application avec le bouton en haut à droite



Le site est publié avec les dépendances et configuration nécessaires.

## 2. Précisez les moyens utilisés :

Pour publier Mon projet j'ai utilisé Heroku

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul

---

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► Studi

Chantier, atelier, service ► ECF

Période d'exercice ► Du : **01/10/24** au : **08/02/24**

#### 5. Informations complémentaires *(facultatif)*

## Titres, diplômes, CQP, attestations de formation

*(Facultatif)*

[illegible]



---

---

## Déclaration sur l'honneur

---

Je soussigné, Lucas Cherbuin, déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l'auteur des réalisations jointes.

Fait à 1071 Chexbres (Suisse)

le 23.4.2025

pour faire valoir ce que de droit.

Signature :

A handwritten signature in black ink, appearing to be 'L. Cherbuin', written on a white background.

## Documents illustrant la pratique professionnelle

(Facultatif)

---

## DOSSIER PROFESSIONNEL <sup>(DP)</sup>

---

---

### ANNEXES

---

*(Si le RC le prévoit)*