Documentation technique

1. Réflexions initiales technologique sur le sujet

Développer une application web pour un zoo implique plusieurs réflexions initiales technologiques et de conception pour assurer une solution robuste, évolutive et conviviale. Considérons les choix technologiques.

a) Front-end
- Framework et langages : étude et choix des différents langages suivant le besoin. Pour ma part j'ai choisi HTML, Javascript et Blade (laravel)
- UI/UX design : étude et choix d'un Framework pour un design cohérant et attrayant. Pour ma part j'ai choisi le Framework Tailwind CSS
- Cartes : choix de l'intégration d'une carte Google Maps

b) Back-end
- Langages et Framework : étude et choix des différents langages suivant le besoin. Pour ma part j'ai choisi PHP8 et le Framework Laravel

c) Base de données
- Base de données SQL : étude et choix de la base données SQL. Pour ma part j'ai choisi PostgreSQL
- Base de données noSQL : étude et choix de la base données noSQL. Pour ma part j'ai choisi MongoDB

d) Infrastructure et Déploiement
- Hébergement : étude et choix de l'hébergement. Pour ma part j'ai choisi Fly.io

Suite à ces études et aux choix de mes langages j'ai pu commencer à développer mon application.

2. Configuration de mon environnement de travail

Une fois l'étape de la réflexion des technologies pour mon application j'ai mis en place mon environnement de travail.

a) Installer Visual Studio Code (VS Code)
- Téléchargement et installation depuis le site officiel de Visual Studio Code.

b) Configuration de GitHub
- Création d'un compte GitHub
- Installation de GitHub
- Configuration de mes informations utilisateurs

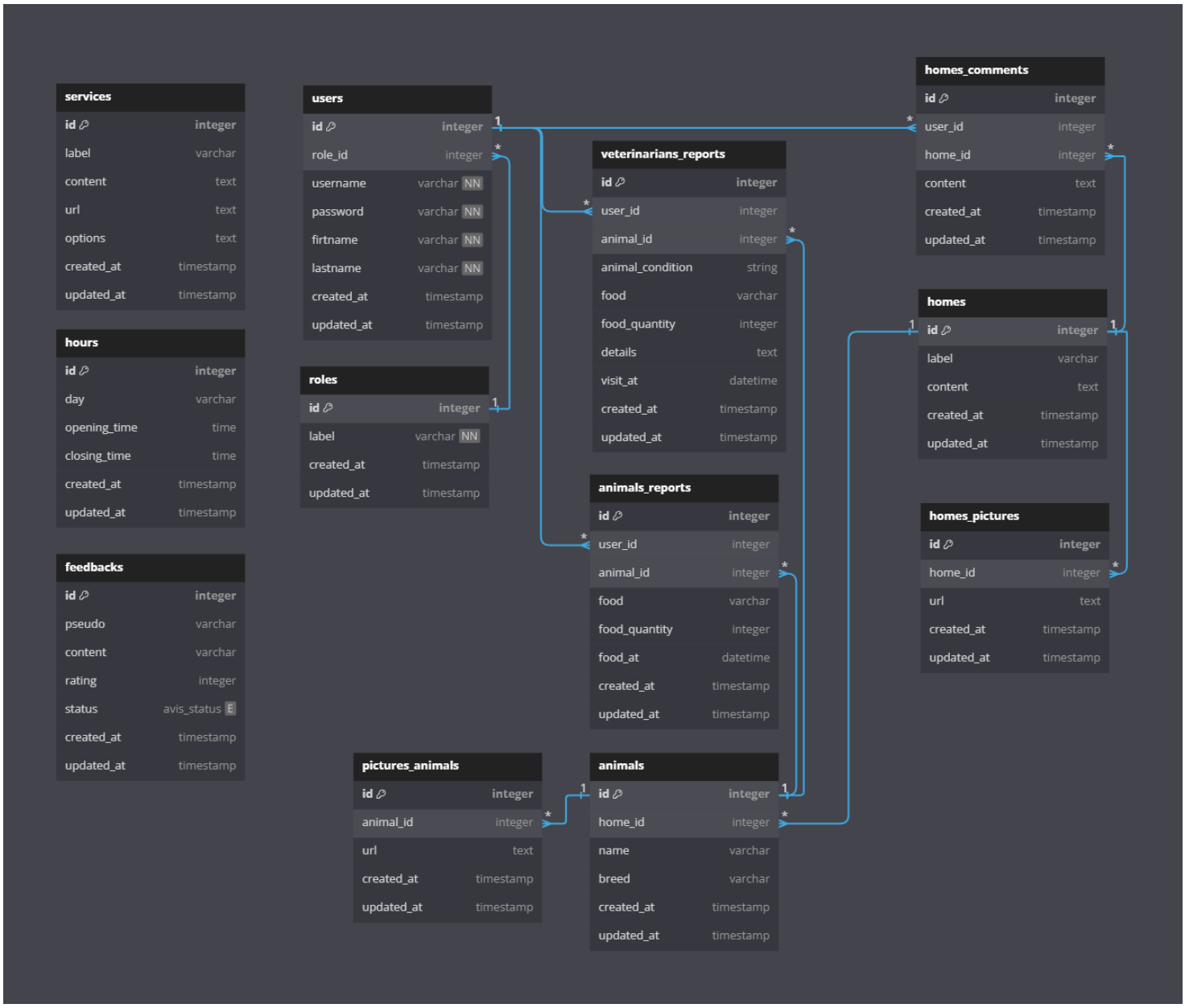
c) Configuration de Trello pour la Gestion de Projet

- Création d'un compte Trello
- Création d'un nouveau tableau de projet
- Ajouts de cartes et tâches

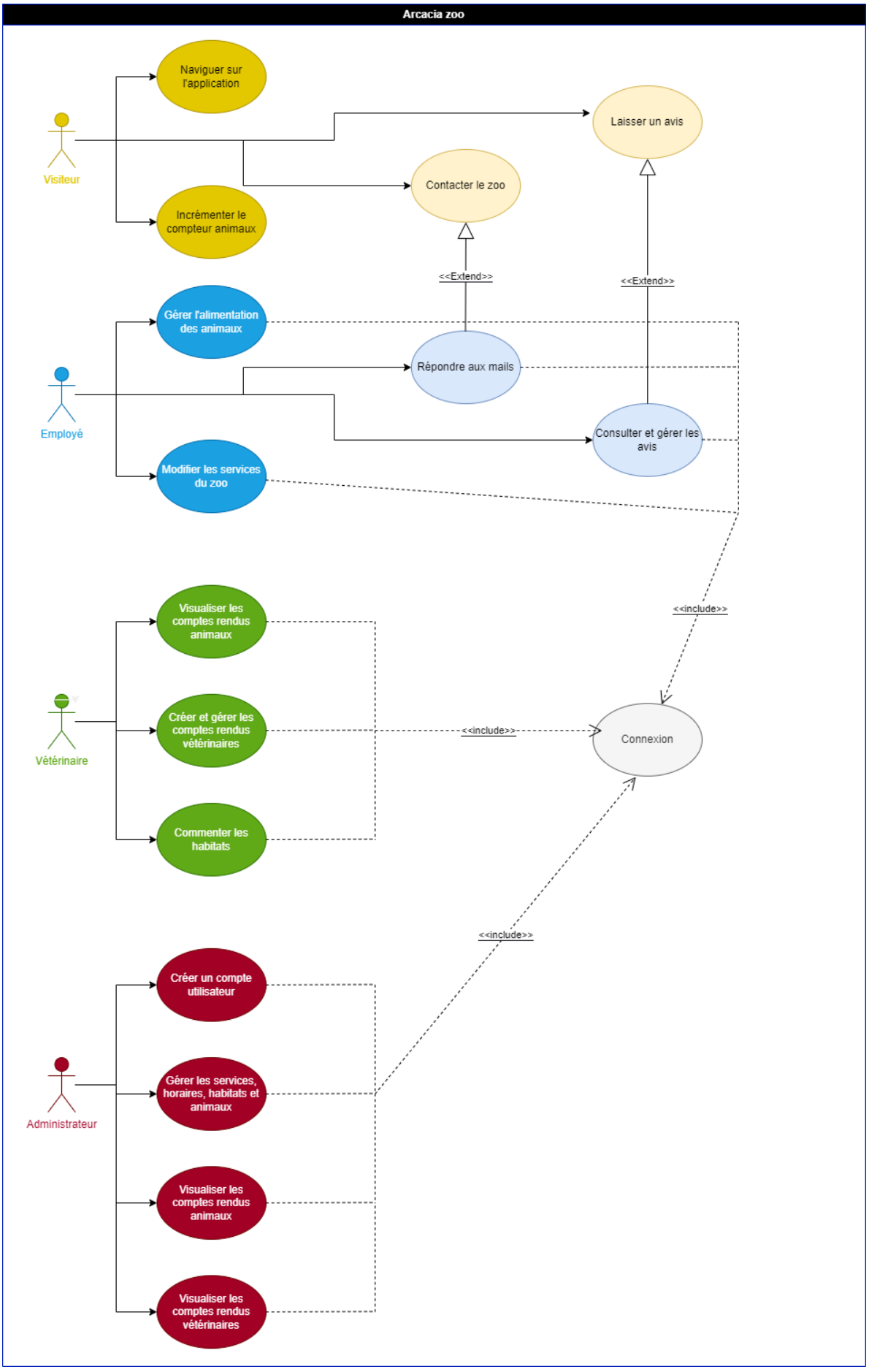
d) Configuration de Figma

- Création d'un compte Figma
- Création d'un nouveau projet Figma
- Création des Wireframes et Mockups

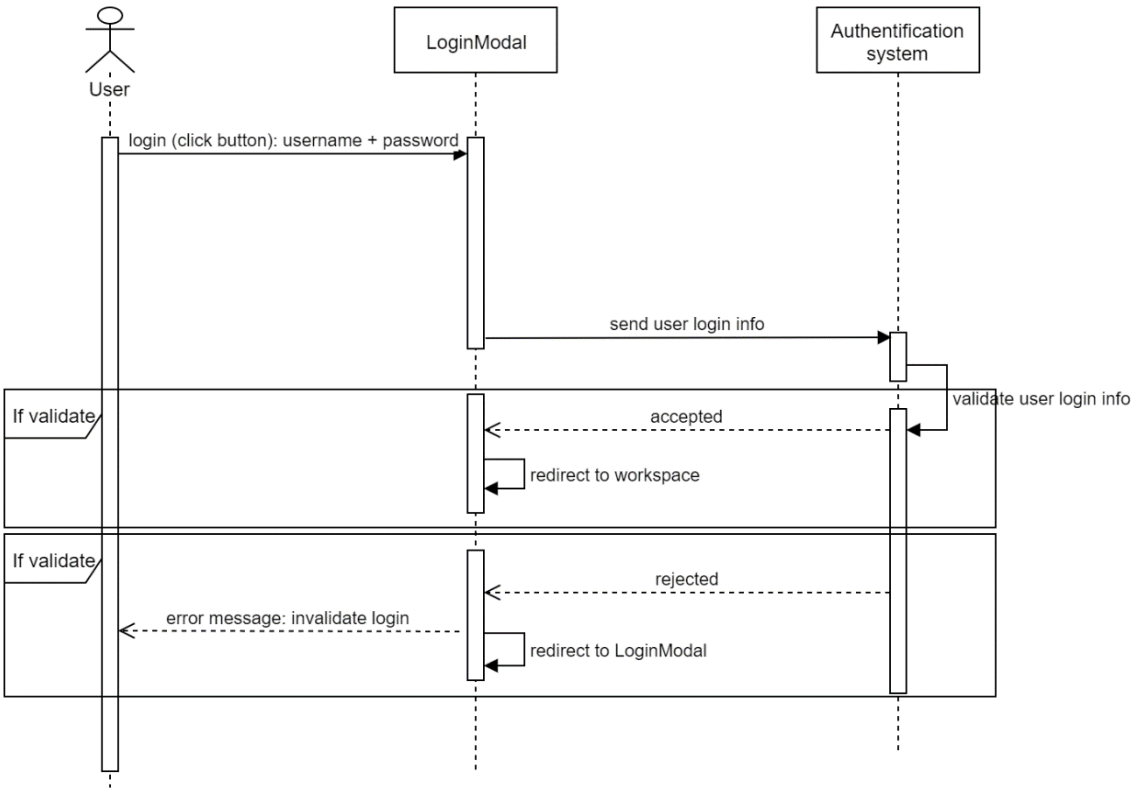
3. Diagramme de classe

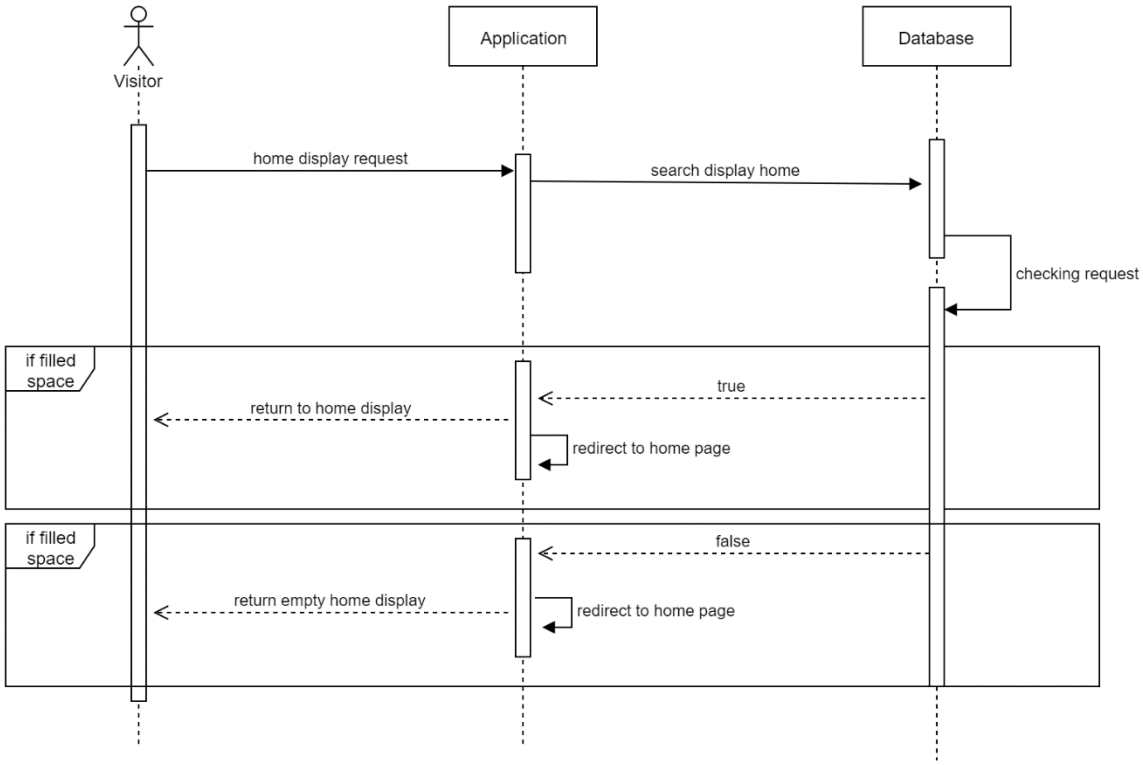


4. Diagramme de cas d'utilisation



5. Diagramme de séquence





6. Déploiement de l'application

J'ai choisi d'utiliser Fly.io pour déployer mon application Laravel car il offre des avantages significatifs ;

- ❖ Infrastructure Globale :
  - Fly.io propose une infrastructure distribuée qui permet de déployer des applications proches de des utilisateurs finaux. Cela réduit la latence et améliore les performances globales de l'application.

- ❖ Simplicité de Déploiement :
  - Fly.io simplifie le processus de déploiement grâce à une interface utilisateur intuitive et des commandes CLI faciles à utiliser. En tant que développeurs nous pouvons rapidement mettre en ligne les applications Laravel.

- ❖ Scalabilité Automatique :
  - Les applications déployées sur Fly.io peuvent automatiquement s'adapter à la charge de trafic. Fly.io gère le scaling horizontal et vertical sans nécessiter une intervention manuelle.

- ❖ Infrastructure as Code :
  - Fly.io supporte l'infrastructure as code, ce qui permet à nous développeurs de définir et de gérer nos infrastructures à l'aide de fichiers de configuration, facilitant ainsi le versioning et la répétabilité des déploiements.

- ❖ Gestion des Bases de Données :
  - Fly.io offre des services de base de données intégrés, simplifiant ainsi la gestion des bases de données pour les applications Laravel. Les bases de données peuvent également être configurées pour être répliquées globalement, améliorant les performances et la résilience.

- ❖ Réseaux Privés et Sécurité :
  - Fly.io permet de créer des réseaux privés entre les différentes parties de l'application, augmentant ainsi la sécurité et l'isolation des composants.

- ❖ Flexibilité du Code et des Langages :
  - Laravel est un Framework PHP. Fly.io supporte plusieurs langages et Framework, permettant une grande flexibilité si l'application utilise des micro services écrits dans différents langages.

- ❖ Observabilité et Monitoring :
  - Fly.io propose des outils intégrés pour le monitoring et l'observabilité, ce qui me permettra de surveiller les performances de mon application et d'identifier rapidement les problèmes.

Voici les différentes étapes que j'ai réalisé pour déployer mon application via Fly.io :

1. J’ai installé Fly CLI
   curl -L https://fly.io/install.sh | sh

2. J’ai configuré Fly.io
   fly auth signup
   fly auth login
   flyctl launch
   Pendant l'exécution de *« flyctl launch »,* j’ai été invité à choisir un nom pour l’application et à sélectionner la région de déploiement.

3. J’ai configuré PostgreSQL sur Fly.io
   fly postgres create

   J’ai suivi les instructions pour configurer la création de la base de données. Une fois la création terminée, j’ai obtenu des détails de connexion.

   fly postgres attach --app my-laravel-app
   Cela m’a permis de configurer mes variables d’environnement pour la connexion Laravel à PostgreSQL

4. J’ai configuré Laravel pour Utiliser PostgreSQL
5. J’ai mis à jour le fichier « .env » de mon application avec les informations de base :
   DB_CONNECTION=pgsql
   DB_HOST=<your-db-host>
   DB_PORT=5432
   DB_DATABASE=<your-db-name>
   DB_USERNAME=<your-db-username>
   DB_PASSWORD=<your-db-password>

   Sachant que les valeurs <your-db-host>, <your-db-name>, <your-db-username>, et <your-db-password> m’ont été fourni par Fly.io après la création et l'attachement de la base de données.

6. J’ai déployé l'Application sur Fly.io
   flyctl deploy

7. J’ai effectué les migrations de la Base de Données après mon déploiement pour configurer mes tables
   fly ssh console -C "php artisan migrate"

8. J’ai testé mon application afin de m’assurer que celle-ci fonctionnait bien.