



Documentation technique



par Cédric Phung



Spécifications techniques

Serveur :

Heroku : Plateforme de déploiement

PHP 8.2 : Langage programmation côté serveur

PDO (PHP Data Objects) : Extension PHP pour accéder aux bases de données.

Base de Données :

ClearDB : Service de base de données MySQL hébergé.

MongoDB Atlas : Service cloud pour MongoDB

Front-end :

HTML5 : Langage de balisage pour structurer le contenu des pages web

CSS3 : Langage de feuilles de style

Bootstrap 5 : Framework CSS

JavaScript : Langage de programmation côté client

Back-end :

PHP 8.2 : Langage de programmation côté serveur

Symfony 6.4 : Framework PHP

Doctrine : ORM pour manipuler les bases de données

MySQL : Système de gestion de base de données relationnelle

MongoDB : Base de données NoSQL pour le stockage et la gestion des données non relationnelles.

Cloud :

AWS S3 : Service de Stockage



Choix des technologies

Front End : Pour le développement du front-end, j'ai opté pour HTML5, CSS3, Bootstrap et JavaScript.

Bootstrap facilite la gestion du positionnement et des breakpoints grâce à son système de grilles, tandis que **CSS3** permet d'appliquer des styles personnalisés et de définir des media queries spécifiques. JavaScript rend le site interactif en permettant des mises à jour dynamiques des données, comme les filtres ou les tris, sans nécessiter un rechargement complet de la page.

Back End : Pour le back-end, j'utilise **Symfony**, qui dispose de nombreux bundles très utiles, comme EasyAdmin, facilitant la création d'un espace d'administration. Le profiler intégré de Symfony aide à identifier et résoudre les erreurs plus efficacement.

Symfony contribue à la sécurité des applications en offrant des protections robustes contre diverses attaques, telles que les attaques CSRF (*Cross-Site Request Forgery*), grâce à son bundle de sécurité intégré. En utilisant le système de gestion des sessions et les fonctionnalités de validation et de protection des formulaires, Symfony aide à prévenir les attaques courantes et assure une meilleure protection des données utilisateur.

En parallèle, **Doctrine** facilite la prévention des injections SQL en utilisant des requêtes préparées. Cette méthode permet de séparer les instructions SQL des données fournies par les utilisateurs, ce qui protège contre les attaques par injection SQL.

MySQL bénéficie d'une vaste communauté de développeurs et d'utilisateurs, offrant ainsi un soutien précieux et des solutions rapides aux problèmes rencontrés.

Pour la gestion des données NoSQL, j'utilise **MongoDB**. Très populaire et facile à utiliser, MongoDB offre une grande flexibilité grâce à son modèle de données basé sur des documents.

Serveur : **Heroku** possède d'un très bon service client qui offre une assistance rapide et efficace. Son intégration directe avec GitHub simplifie le processus de déploiement et l'utilisation de la CLI facilite la gestion de la mise en production et des configurations. Par ailleurs, Heroku permet une connexion simplifiée avec ClearDB.

Enfin, Heroku dispose d'une grande communauté en ligne qui fournit de nombreuses ressources ou solutions.

Le service de stockage **AWS** (Amazon Web Services) me permet de télécharger mes images directement sur le cloud, ce qui évite d'alourdir mon application et améliore ses performances.

Configuration de l'environnement de travail

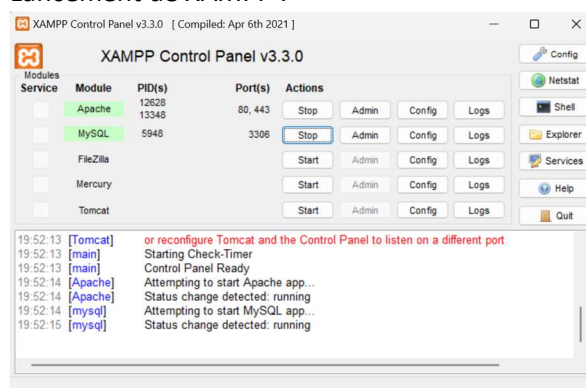
- Installation Visual Studio Code

Visual Studio Code est un éditeur de code puissant, qui offre des fonctionnalités avancées telles que l'autocomplétion, et l'intégration de nombreux plugins.

- Installation XAMPP

XAMPP permet de configurer un environnement web complet en local pour tester l'application en développement. Il inclut Apache (serveur web), MySQL (base de données relationnelle), et PHP.

Lancement de XAMPP :



- Installation MongoDB

MongoDB est une base de données NoSQL permettant de gérer des données en local. Pour intégrer MongoDB avec PHP, il est nécessaire de télécharger et d'installer l'extension PHP PECL pour MongoDB, qui fournit les outils nécessaires pour interagir avec la base de données depuis les scripts PHP.

Extension PHP :

| | | |
|-----------------|------------------|----------------------------|
| php_mongodb.dll | 08/07/2024 11:05 | Extension de l'application |
|-----------------|------------------|----------------------------|

- Installation Composer

Composer est un gestionnaire de dépendances pour PHP. Il permet l'installation, la mise à jour, et la gestion des bibliothèques et des packages PHP.

- Configuration des fichiers

Le fichier .env.local : contient les paramètres spécifiques à l'environnement local, tels que les informations de connexion aux bases de données SQL (MySQL) et NoSQL (MongoDB).

Configuration .env.local :



```
CLEARDB_DATABASE_URL="mysql://root:@localhost:3306/zoo_arcadia"
# CLEARDB_DATABASE_URL="mysql://bbad9cc640de7f:5321c453@eu-cluster-west-01.k8s.cleardb.net/
heroku_160217e5999377a?reconnect=true"
# DATABASE_URL="postgresql://app:!ChangeMe!@127.0.0.1:5432/app?serverVersion=16&charset=utf8"

###< doctrine/doctrine-bundle ###
# MONGODB_URL="mongodb+srv://cedric:KHfCuvPbwkyATVd8@zoo-arcadia.sib6qms.mongodb.net/?
retryWrites=true&w=majority&appName=zoo-arcadia"
MONGODB_URL="mongodb://localhost:27017/zoo-arcadia"
MONGODB_DB="zoo_arcadia"
```

- **Installation GitHub** : Plateforme de gestion de versions qui permet de suivre les modifications du code, de collaborer avec d'autres développeurs, et de revenir en arrière en cas d'erreur.

Diagrammes

Diagramme de cas d'utilisation

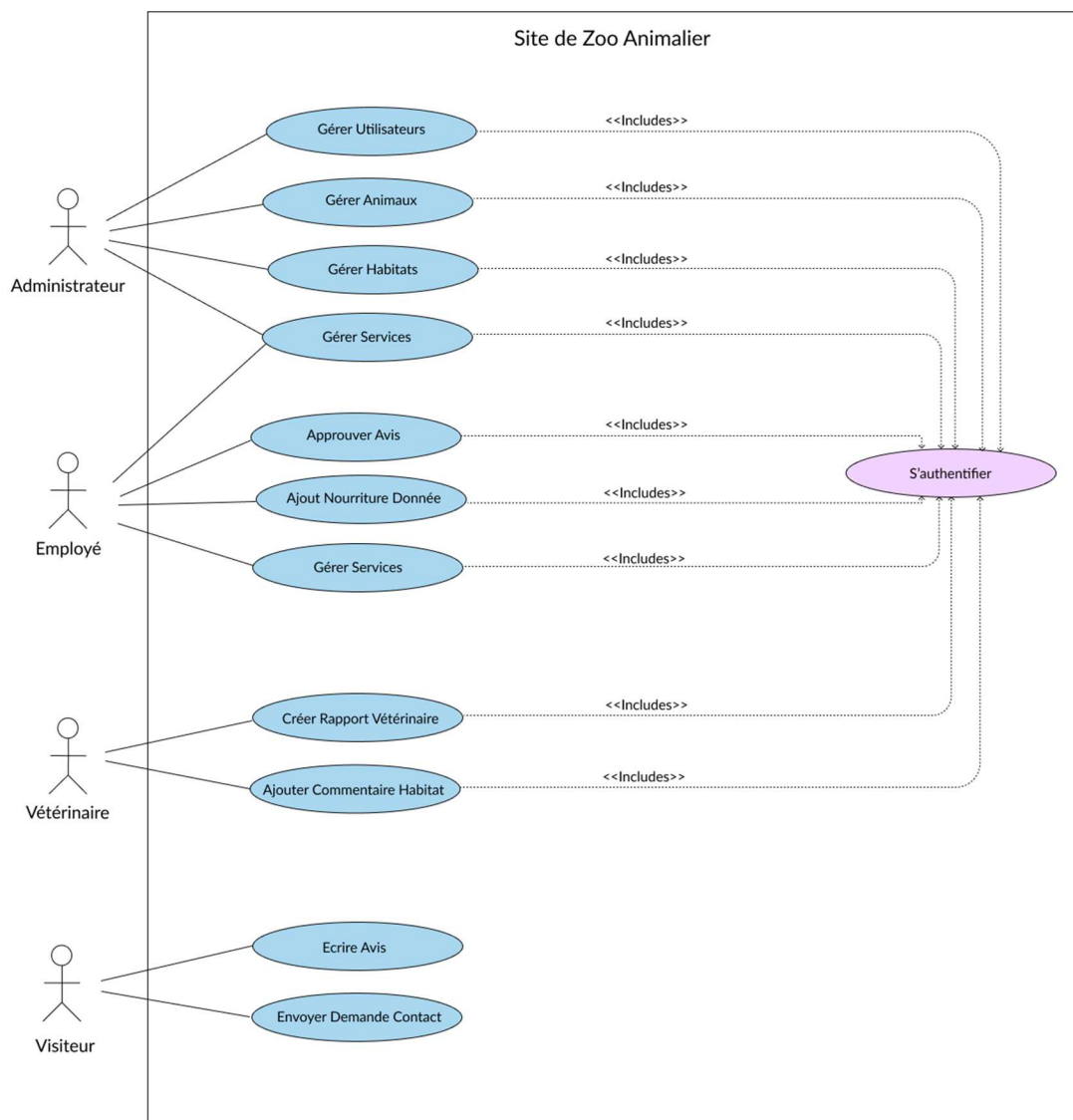


Diagramme de séquence : Création d'un service

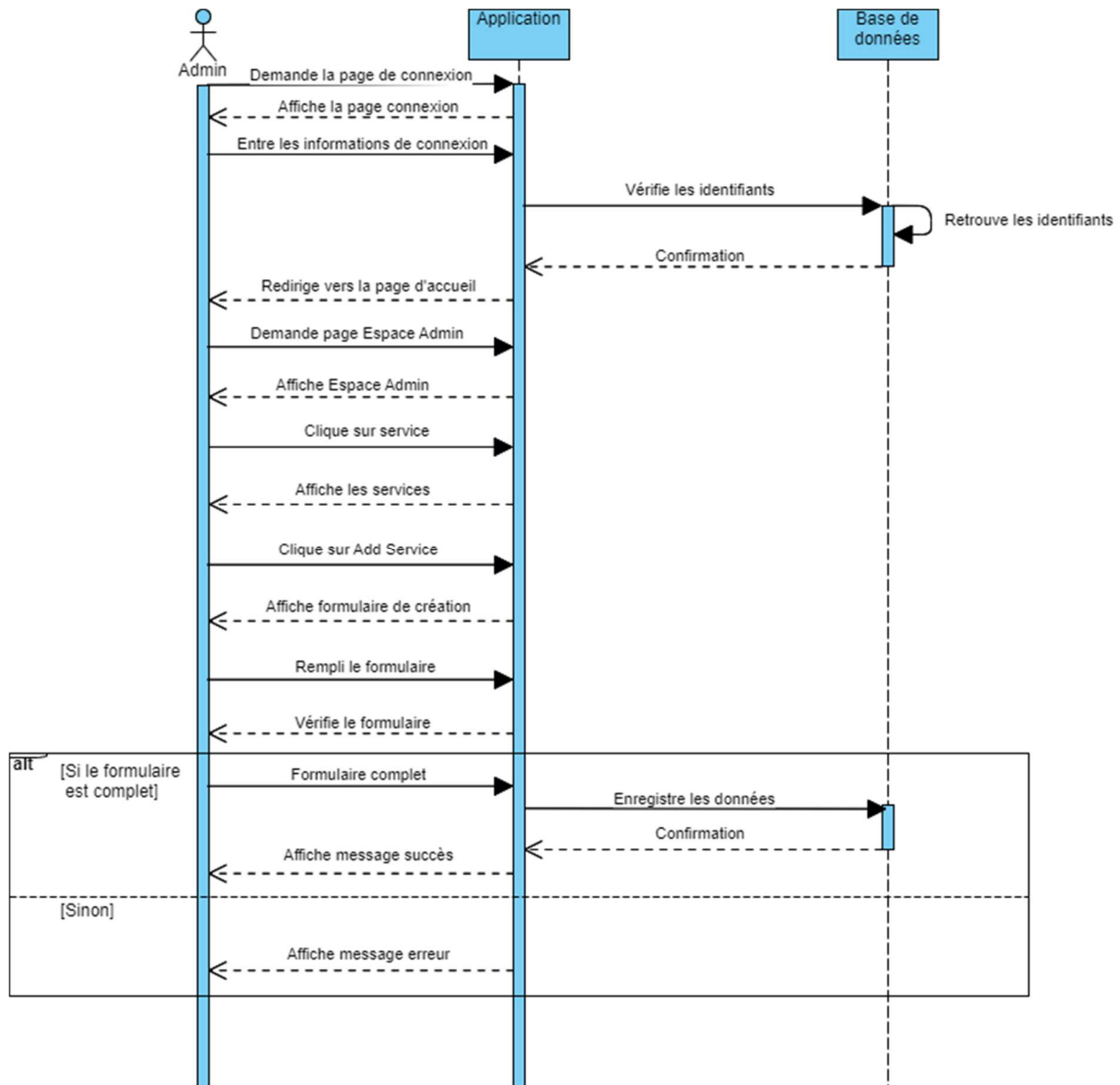
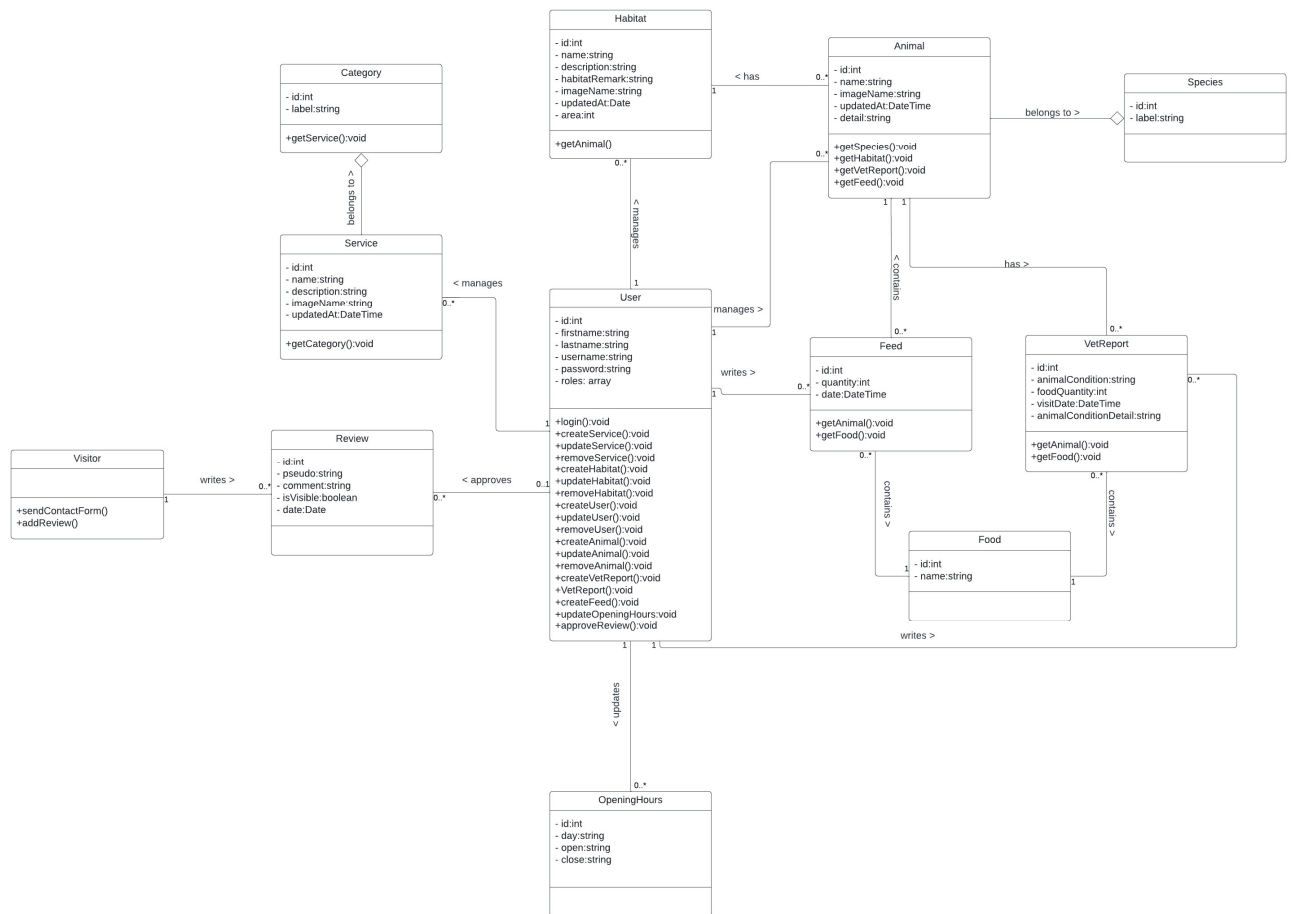




Diagramme de classes



Déploiement de l'application

Sur Heroku :

J'ai créé d'une nouvelle application : *zoo-broceliande-arcadia*

Dans déploiement, j'ai connecté mon compte GitHub et le dépôt de mon application.

J'ai ajouté ClearDB MySQL : dans Ressources puis Add-ons. Les variables d'environnement pour la base de données ClearDB sont automatiquement ajoutées à votre configuration Heroku.

Dans les variables d'environnement, j'ai passé APP_ENV à prod.

Sur MongoDB Atlas : j'ai créé un nouveau projet et j'ai ajouté l'URL et la base de données dans les variables d'environnement d'Heroku.































AWS : J'ai extrait les variables d'environnement depuis AWS et les ai ajoutées à la configuration de mon application sur Heroku.

Config Vars

Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Hide Config Vars

| | | |
|-------------------------------|---|---|
| APP_ENV | prod |   |
| APP_SECRET | b1c03f9beea091bd86b1d9a9adec574e |   |
| APP_UPLOAD_SERVICE_URI_PREFIX | https://zoo-arcadia.s3.eu-north-1.amazonaws.com |   |
| AWS_ACCESS_KEY_ID | AKIATCKASDHT5CBJ77H4 |   |
| AWS_BUCKET_NAME | zoo-arcadia |   |
| AWS_ENDPOINT | s3.amazonaws.com |   |
| AWS_REGION | eu-north-1 |   |
| AWS_SECRET_ACCESS_KEY | 1tz8zv5geZ1cketz5grSUT2Q3lnpF12RfBQ7Q66I |   |
| CLEARDB_DATABASE_URL | mysql://bbad9cc640de7f:5321c453@eu-cluste |   |
| MAILER_DSN | smtp://4ecb9a6b8511e8:bdf67534083d4e@sand |   |
| MESSENGER_TRANSPORT_DSN | doctrine://default?auto_setup=0 |   |
| MONGODB_DB | zoo-arcadia |   |
| MONGODB_URL | mongodb+srv://cedric:khFCuvPbukyATVd8@zoo |   |
| KEY | VALUE |   |

Add

Variables d'environnement sur Heroku

Sur Symfony :

J'ai utilisé Composer pour mettre à jour les dépendances en excluant les packages de développement avec la commande `composer install --no-dev --optimize-autoloader`. Ensuite, j'ai ajouté le fichier Procfile et le fichier .htaccess.

- Procfile : Ce fichier permet de définir à Heroku les processus à exécuter lors du démarrage de l'application. Par exemple, il peut spécifier la commande pour démarrer le serveur web.
- .htaccess : Ce fichier est un fichier de configuration pour le serveur web Apache. Il permet de configurer des fonctionnalités telles que la réécriture des URL, la protection par mot de passe, et d'autres paramètres de configuration du serveur.