Doc technique

Réflexions initiales technologiques

J'ai choisi d'utiliser les technologies suivantes pour le projet : React JS, Tailwind CSS pour le front-end et Express JS (framework pour Node.js) pour le back-end. Ces choix ont été faits en raison de la modernité de cette stack, de sa maintenabilité et du fait qu'ils bénéficient d'une grande communauté.

- React JS est une bibliothèque JavaScript open-source pour la création d'interfaces utilisateur. Le principe de React JS de fonctionner avec des composants réutilisables permet d'accélérer le développement et simplifie la maintenance de l'application.
- Tailwind CSS est un framework CSS permettant d'utiliser des classes utilitaires à l'intérieur du balisage HTML. Le choix d'utiliser Tailwind CSS s'est fait en premier lieu parce que je trouve plus simple de styliser une application de cette manière en évitant d'utiliser de nombreuses feuilles de style. La configuration de Tailwind CSS simplifie également l'utilisation de styles personnalisés. De plus, Tailwind CSS fonctionne parfaitement bien avec React JS.
- Express JS est un framework pour Node.js utilisé pour des applications web et des API robustes et flexibles. Ce framework permet d'utiliser aisément JavaScript côté serveur. Il permet également une communication simplifiée avec la base de données.

Configuration de l'environnement de travail

Système de contrôle des versions

J'ai fait le choix d'utiliser GitHub et GitHub Desktop. Ce dernier a été choisi en raison de son intégration avec VSCode et du fait qu'il facilite la gestion des dépôts Git lorsque l'on est pas à l'aise avec les lignes de commandes.

Environnement de développement local

J'ai fait le choix d'utiliser VSCode car c'est un logiciel léger et performant bénéficiant de nombreuses extensions facilitant le développement.

Gestion de projet

J’ai choisi d’utiliser Notion en raison de sa polyvalence.

Test des API

J’ai choisi d’utiliser Postman pour tester les API et les requêtes HTTP afin de vérifier le bon fonctionnement de celles-ci tout au long du développement de l’application.

Hébergement et déploiement

J’ai choisi d’utiliser deux hébergements distincts pour le front-end et le back-end. Hostinger d’une part pour le front-end en raison de la simplicité de son interface, la facilité pour lier une base de données phpMyAdmin ainsi que son gestionnaire de fichiers facilitant le déploiement. Render d’autre part pour le back-end en raison du déploiement automatisé à partir d’un repository GitHub.

Diagramme de classe

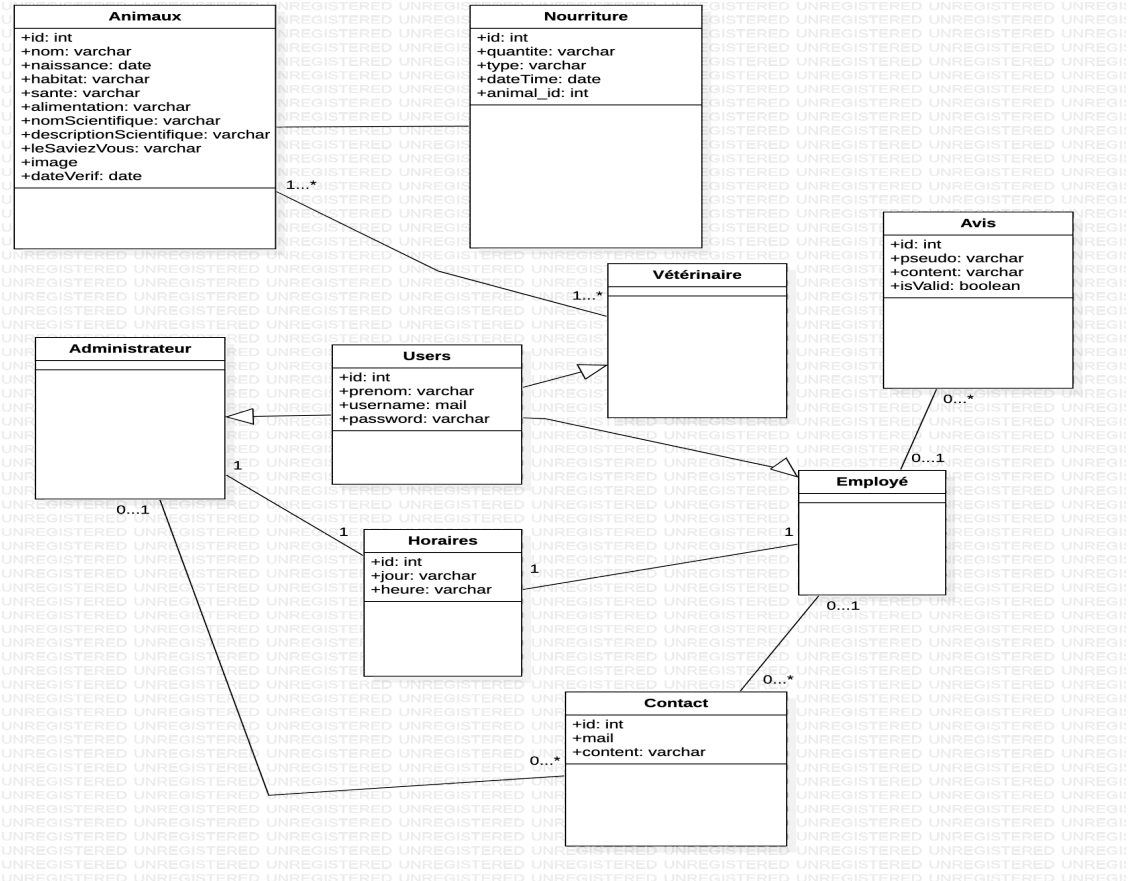


Diagramme de cas d'utilisation

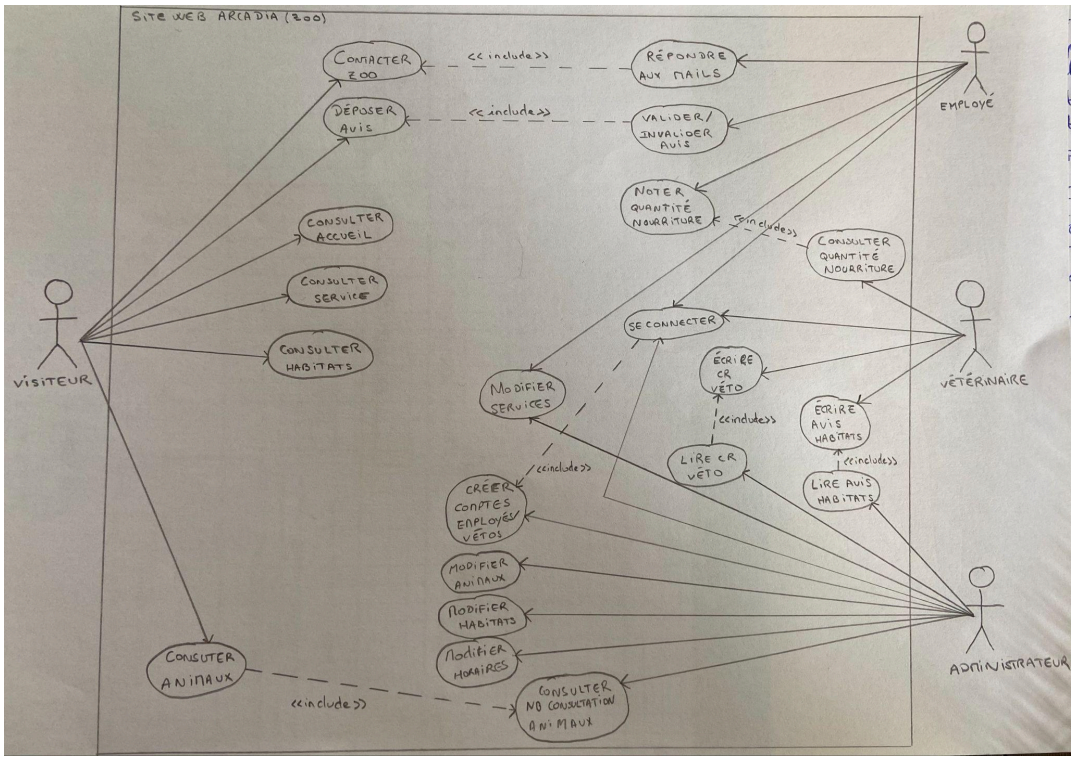
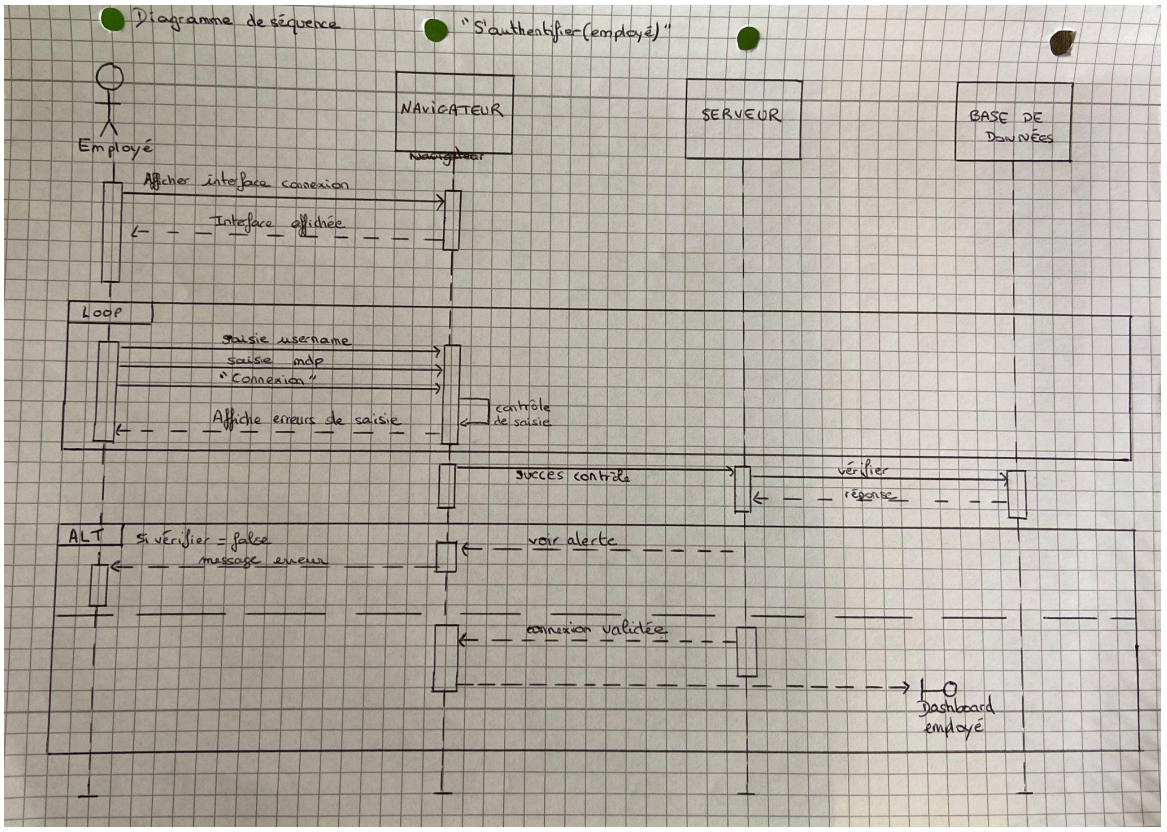
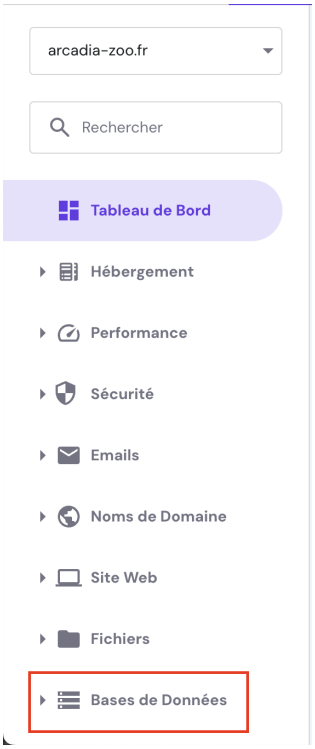


Diagramme de séquence



Documentation déploiement en ligne

Front-end (React JS) sur Hostinger



1) Depuis Hostinger, acquérir un nom de domaine.
2) Depuis le volet latéral du tableau de bord d'Hostinger, cliquer sur "Base de Données" puis sur "Gestion" afin de configurer une base de données relationnelle via phpMyAdmin.
3) Rentrer les identifiants de base de données dans le dossier back-end App/config/db.config.js afin d'y avoir accès depuis l'application web.
4) Depuis le terminal de commande du dossier front-end, exécuter la commande *npm run build*. Stocker le dossier de build dans le dossier front-end.
5) Depuis Hostinger, accéder à l'onglet "Fichiers" dans le volet latéral puis cliquer sur "Gestionnaire de fichiers" puis sur "Accéder aux fichiers de arcadia-zoo.fr.
6) Ouvrir le dossier "public_html", cliquer sur "upload" puis ajouter tous les fichiers contenus dans le dossier build, ajouter ensuite tous les dossiers contenus dans le dossier build.
7) Une fois tous ces fichiers et dossiers uploader, le front-end de notre application web est déployé.

Back-end (Express JS) sur Render via GitHub

1) Créer un dépôt GitHub et un compte sur Render.
2) Depuis Render, cliquer sur "New+", sélectionner "Web Service"
3) Sélectionner "Build and deploy from a Git repository
4) Autoriser Render à accéder au repository
5) Le déploiement se fait automatiquement à chaque Push sur le dépôt GitHub
6) Remplacer toutes les occurrences des adresses en "http://localhost" utilisées pendant le développement par l'URL fournie par Render pour le back-end.
7) Le site est déployé et fonctionnel.