Documentation technique

Réflexions initiales technologique sur le sujet :

Le besoin principal pour l'utilisation des technologies était la maîtrise et le contrôle, le but n'était pas d'utiliser un outil tellement performant que la réflexion disparait. L'intérêt était quand même de comprendre et d'apprendre, de ramer et de chercher.

Pour cela il aura été préférable d'utiliser des outils parfaitement documentés, et avec une grande communauté, permettant de trouver des réponses facilement aux problèmes qui se poseraient.

Connaissant déjà la taille exponentielle de la documentation de php.net, et ayant pris connaissance de la base communautaire de Symfony, le choix s'est peu à peu dirigé vers ce Framework très intéressant.

Cela n'a pas été le seul facteur bien entendu, il s'agissait également de préférences d'environnement de travail, et le système de classes et d'objets semblait à la fois complexe mais très puissant une fois compris et assimilé.

Configuration de votre environnement de travail :

Il a fallu pour cela créer un nouveau squelette Symfony vierge à l'aide de Composer, auquel nous avons ajouté nombre d'extension au fil du développement pour satisfaire aux besoins.

Après Symfony, l'ajout de Doctrine permettra de configurer l'accès à la base de données MySQL dans le fichier .env.local. Et en ajoutant le maker-bundle de Symfony, la triade de gestion des données est formée, et nous permet de créer des entités objets en PHP puis de les convertir en éléments SQL.

Le gros avantage étant surtout de faciliter l'accès aux entités, en évitant d'effectuer des requêtes SQL manuellement pour chaque fois que l'accès était nécessaire.

En ajoutant à tout cela le bundle Twig pour la génération des pages HTML, ainsi que l'accès facilité aux données de nos contrôleurs PHP.

Il ne manque plus que la gestion du visuel et du CSS, il a donc été installé le gestionnaire d'assets de Symfony, qui s'occupe de rendre disponible les fichiers CSS et JS dans le dossier public du projet au moment de la génération. En ajoutant les fichiers de style de Bootstrap et en incluant les scripts de Bootstrap également.

Modèle conceptuel de données :

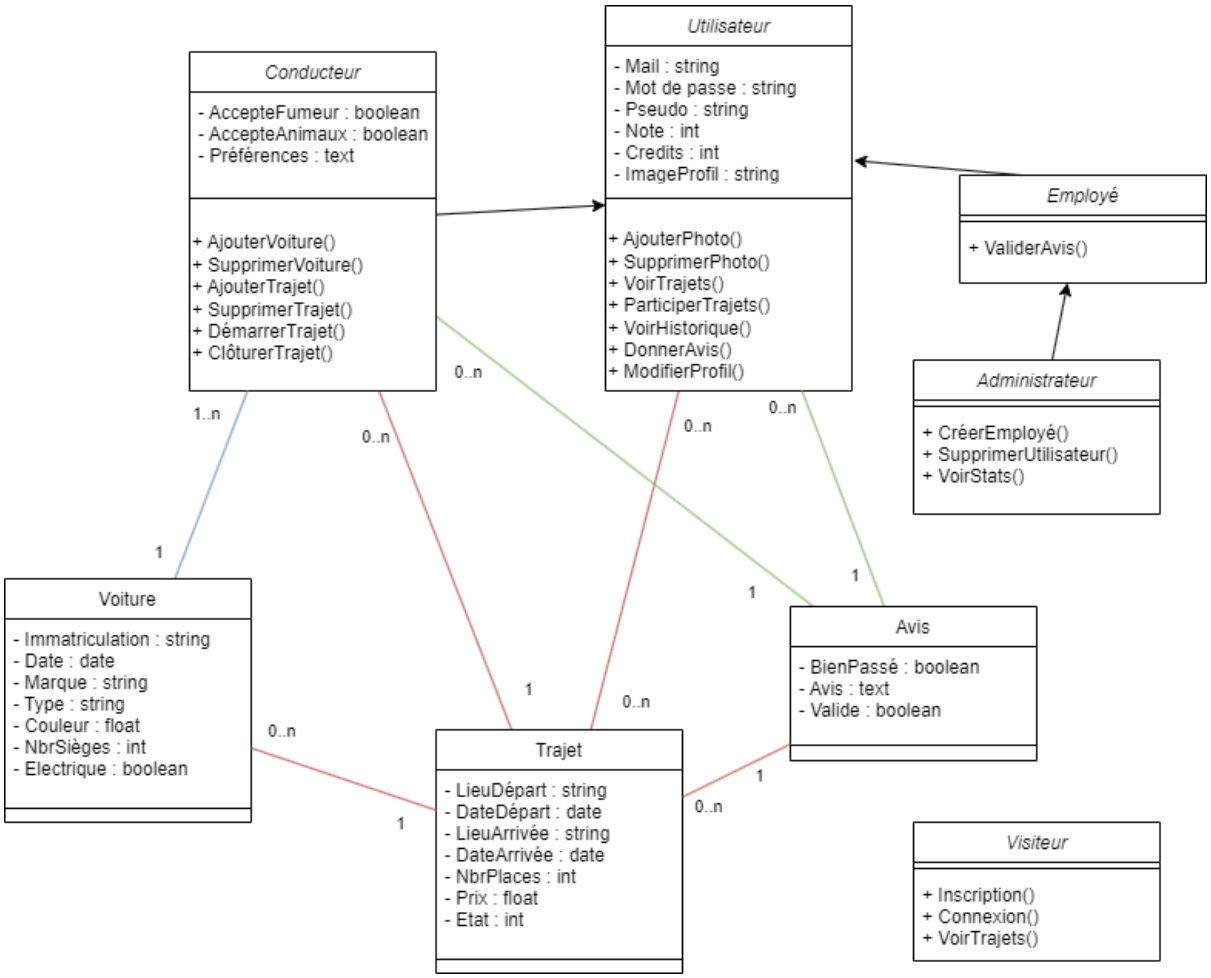


Diagramme d’utilisation :

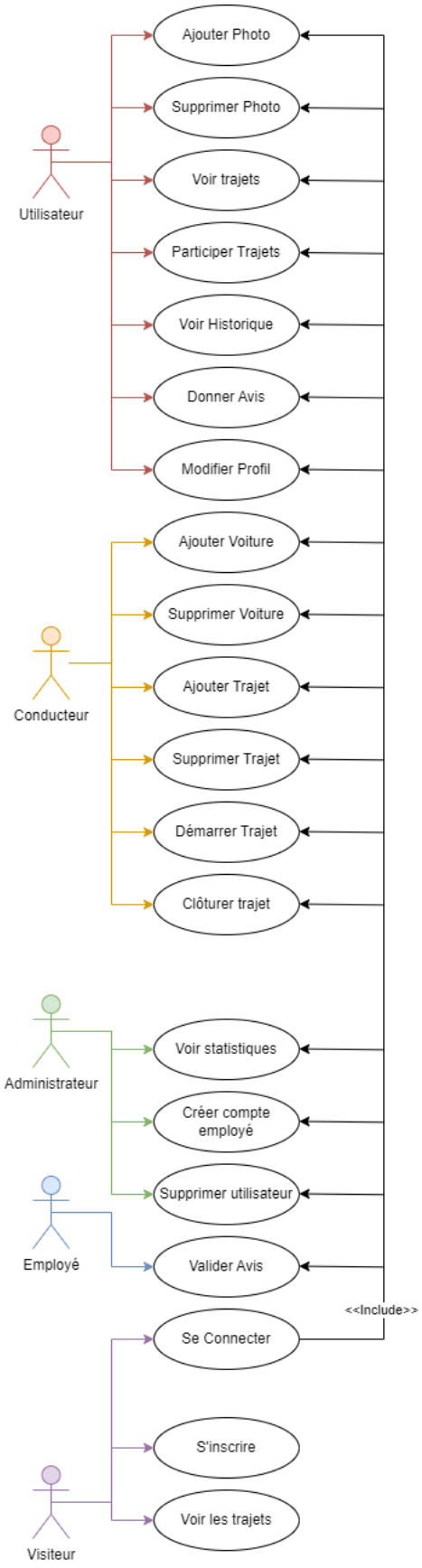
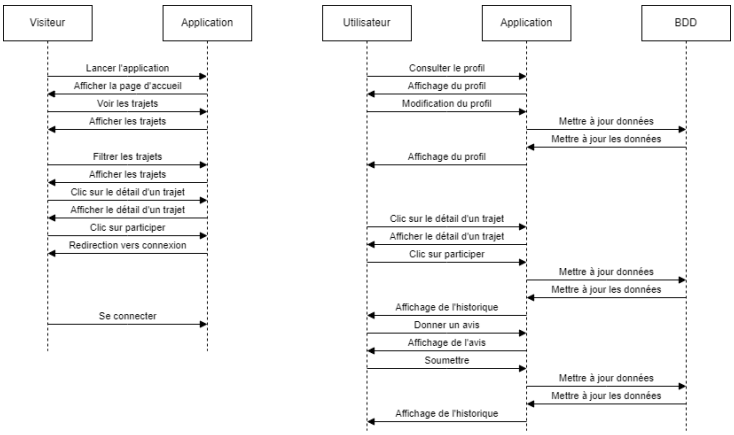
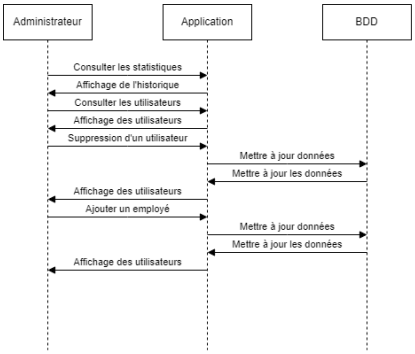
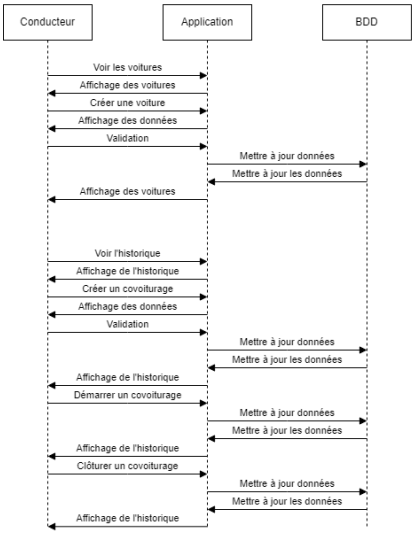
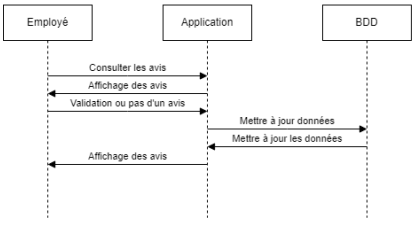


Diagramme de séquence :









Déploiement de l'application sur Heroku :

Le déploiement était prévu sur Heroku, très réputé, et bien compatible avec un projet Symfony, en plus de disposer d'une documentation expliquant en détail l'installation d'un projet Symfony.

Il aura tout d'abord fallu installer un package pour configurer le projet Symfony en mode de fonctionnement avec un serveur web Apache, le pack Apache. Puis ne pas oublier de compiler nos assets pour bien obtenir toutes les images dans le dossier public.

Ensuite le client Heroku a été téléchargé et installé sur la machine, pour disposer des lignes de commandes telles que "Heroku create", lancée dans le projet Symfony pour créer le projet Heroku.

Il aura ensuite fallu se connecter au compte Heroku via le terminal, puis créer ce qui s'appelle un ProcFile chez Heroku pour pouvoir finalement synchroniser le repository Git et ensuite pousser le repository Git vers Heroku.

Sur Heroku il aura fallu ajouter un module pour la gestion de base de données en MySQL, puis effectuer la migration avec Doctrine. Et modifier les variables d'environnement APP_ENV pour le mode de production.

Après quelques ajustements et l'ajout de packages manquants comme Chart.js pour les graphiques, le déploiement était terminé.