**Grille de validation - Dossier Professionnel**

**Grille de validation pour le choix des exemples professionnels par Activité Type**

**Conseils**

Cette grille a été conçue pour vous permettre de choisir les pratiques professionnelles (tâches, opérations) pour chaque Activité Type que vous devez décrire et développer dans votre dossier professionnel.

Avant de commencer la rédaction de votre DP, faites valider par votre formateur, au moyen de cette grille, les tâches choisies pour être bien en phase avec les attentes du référentiel et du jury.

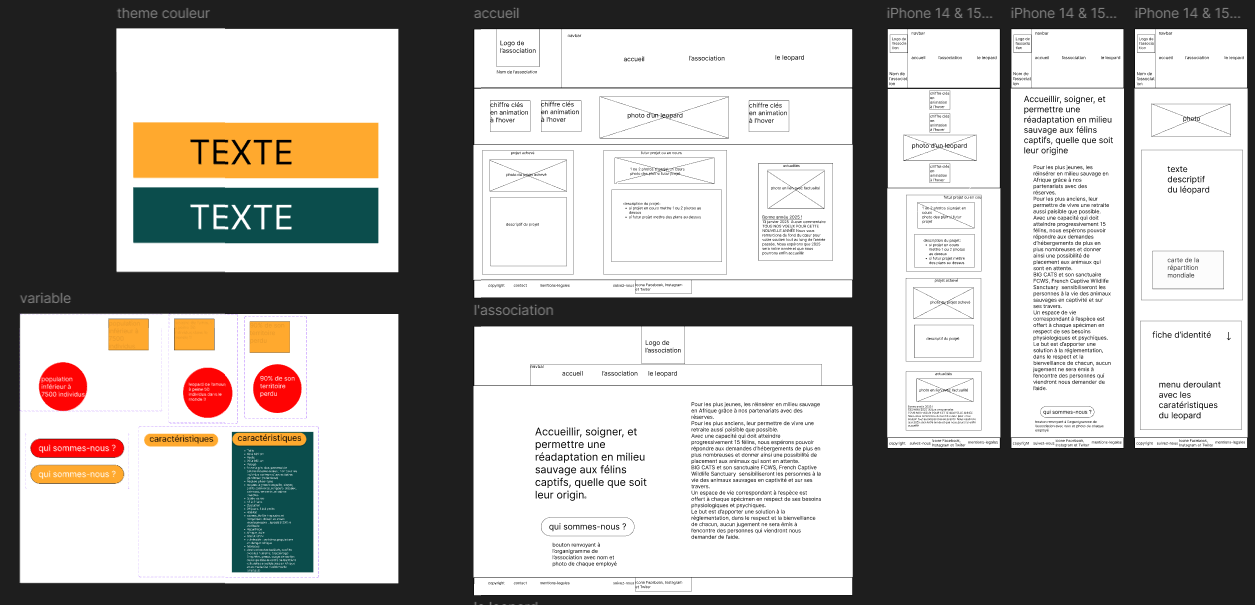
Pour chaque exemple, pensez à justifier votre choix afin que votre formateur puisse avoir une vision précise de ce que vous souhaitez développer. L’étude des supports de formation est obligatoire avant toute rédaction pour vous permettre d’utiliser le vocabulaire professionnel attendu.

Faites le nécessaire pour que vos exemples soient variés et couvrent le plus possible toutes les compétences professionnelles à développer dans l’Activité Type concernée.

|  |  |  |
| --- | --- | --- |
| **Nom – Prénom :**  **FAVRE-BONVIN Yoann** | **Pratique professionnelle (tâche, opération)** | **Commentaires**  **Formateur correcteur**  ***A REMPLIR PAR LE CORRECTEUR*** |
| **RNCP37674BC01 - Développer la partie front-end d'une application web ou web mobile sécurisée** | | |
| **Compétences professionnelles possibles :**   1. Installer et configurer son environnement de travail en fonction du projet web ou web mobile 2. Maquetter des interfaces utilisateur web ou web mobile 3. Réaliser des interfaces utilisateur statiques web ou web mobile 4. Développer la partie dynamique des interfaces utilisateur web ou web mobile | Exemple 1 : projet de création d’un site internet pour une association (consigne uniquement HTML,CSS)  Justifiez votre choix :  Dans le cadre de cet exercice, j’ai choisi de créer une association fictive sur la défense d’un animal.   * 1ere étape : démarrer par les mockups et wireframes en version desktop et responsive (annexe 1 et 2). Ils m’ont permis de me familiariser avec le logiciel Figma et mettre en page les idées. * 2e étape : réaliser le projet via VS Code uniquement en HTML et CSS (annexe 3 et 4). Mise en place du Doctype avec le head. Réalisation de la navbar et le footer car ils seront identiques sur toutes les pages. Le corps de la page avec des animations en CSS afin de rendre le site plus dynamique mais en restant dans une interface statique.   Compétence(s) visée(s) : 2 - 3 |  |
| Exemple 2 : projet ECF sur la base d’Ecoride  Justifiez votre choix :  Il s’agit d’un projet complet sur la base d’un cahier des charges fournis.   * 1ere étape : j’ai mis en place mon environnement de travail via Docker. Après avoir créé ma page « index.php », j’ai créé le fichier dockerfile qui construit une image de l’application (annexe 5). J’ai construit l’image Docker via la commande « docker build -t Ecoride-php », ensuite j’ai créé un fichier « docker-compose.yml » afin qu’il gère le PHP + MySQL. Ensuite j’ai utilisé la commande « docker-compose up – build » afin de lancer le projet sur Docker. Il me suffit plus qu’à aller sur l’adresse <http://localhost:8080> afin de voir le site. * 2e etape : dynamiser une partie dynamique. J’ai mis en place un système AJAX qui permet à une page web de communiquer avec un serveur sans avoir à se recharger entièrement, cela est notamment très utile pour les formulaires. Il est lié au système JavaScript (annexe 6). J’ai mis en place le « fetch » dans le code du formulaire de recherche d’un itinéraire sur la page d’accueil.   Compétence(s) visée(s) : 1-4 |  |

|  |  |  |
| --- | --- | --- |
|  | Exemple 3 : Projet site vitrine  Justifiez votre choix :  Il s’agit d’un site vitrine mais pour le rendre plus vivant, dynamique et esthétique j’ai mis en place 2 solutions :   * La 1ere : un système de « parrallax » qui permet un « scroll » dynamique puisque la photo se dévoile au fil du scroll (annexe 7). Ce système se fait uniquement en HTML et CSS. * La 2e: un système d’apparition de texte ou image au scroll via l’inclusion de JavaScript avec une bibliothèque <https://unpkg.com/aos@2.3.1/dist/aos.css> et un script (annexe 8) : qui a l’image d’un Bootstrap s’inclut directement dans le HTML (avec des fonctions « data-… » en fonction de ce que l’on veut) (annexe 9).   Compétence(s) visée(s) : 3 - 4 |  |
| **RNCP37674BC02 - Développer la partie back-end d'une application web ou web mobile sécurisée** | | |
| **Compétences professionnelles possibles :**   1. Mettre en place une base de données relationnelle 2. Développer des composants d'accès aux données SQL et NoSQL 3. Développer des composants métier côté serveur 4. Documenter le déploiement d'une application dynamique web ou web mobile | Exemple 1 : projet ECF Ecoride  Justifiez votre choix :  Ce projet couvre l’ensemble des compétences requises que je vais développer :   * Mise en place de la base donnée relationnelle. J’ai utilisé le fonction « docker-compose.yml » qui définit un service MySQL (annexe 10). Le volume « docker-entrypoint-initdb.d » permet d’exécuter des scripts SQL au démarrage avec un service phpMyAdmin qui interagit ave MySQL via l’interface web (port 8081). * Développer des composants d’accès aux données SQL et NoSQL. En ce qui concerne la partie SQL, le fichier « dockerfile » installe « pdo\_mysql » ce qui permet d’utiliser PDO pour accéder à MySQL. J’ai aussi mis le PDO afin de sécuriser l’application contre les injections de SQL. Pour le NoSQL, j’ai utilisé MongoDB via le fichier « log\_mongo.php » (déclaré dans « composer.json ». Ce qui permet une gestion des données volumineuses et dynamiques plus souple. * Développer des composants métiers côté serveur : j’ai choisi la logique MVC en suivant les dossiers : « entity » qui contient les entités métiers (comme User, Trajet, Reservation). J’utilise aussi « repository » qui définit des classes responsables de l’accès aux données car elle encapsule les requêtes SQL et mongoDB. Le dossier « service » regroupe les services métiers comme la création d’un trajet, la gestion d’une réservation,… Enfin le dossier « controller » réceptionne les requêtes des utilisateurs (souvent appelées via des formulaires). Toute cela permet de gérer côté serveur et non côté navigateur ce qui rend l’expérience plus fluide et sécurisée. * Déploiement de l’application et documentations. J’ai utilisé Docker pour un déploiement en local avec le fichier docker-compose.yml qui définit tous les services nécessaires au développement et test et cela sur n’importe quel support (machine), en utilisant le repos github ( <https://github.com/yoannfb/ecf-ecoride.git>). En ce qui concerne le déploiement cloud, j’ai utilisé la plateforme Heroku car très simple d’utilisation et permet d’être relié directement au repos GitHub. Toute la procédure à suivre pour l’installation et la configuration se retrouve dans le fichier « Readme » joint dans le dépôt GitHUb.   Compétence(s) visée(s) : 5 – 6 – 7 – 8 |  |
| Exemple 2 :  Justifiez votre choix :  Compétence(s) visée(s) : |  |
| Exemple 3 :  Justifiez votre choix :  Compétence(s) visée(s) : |  |

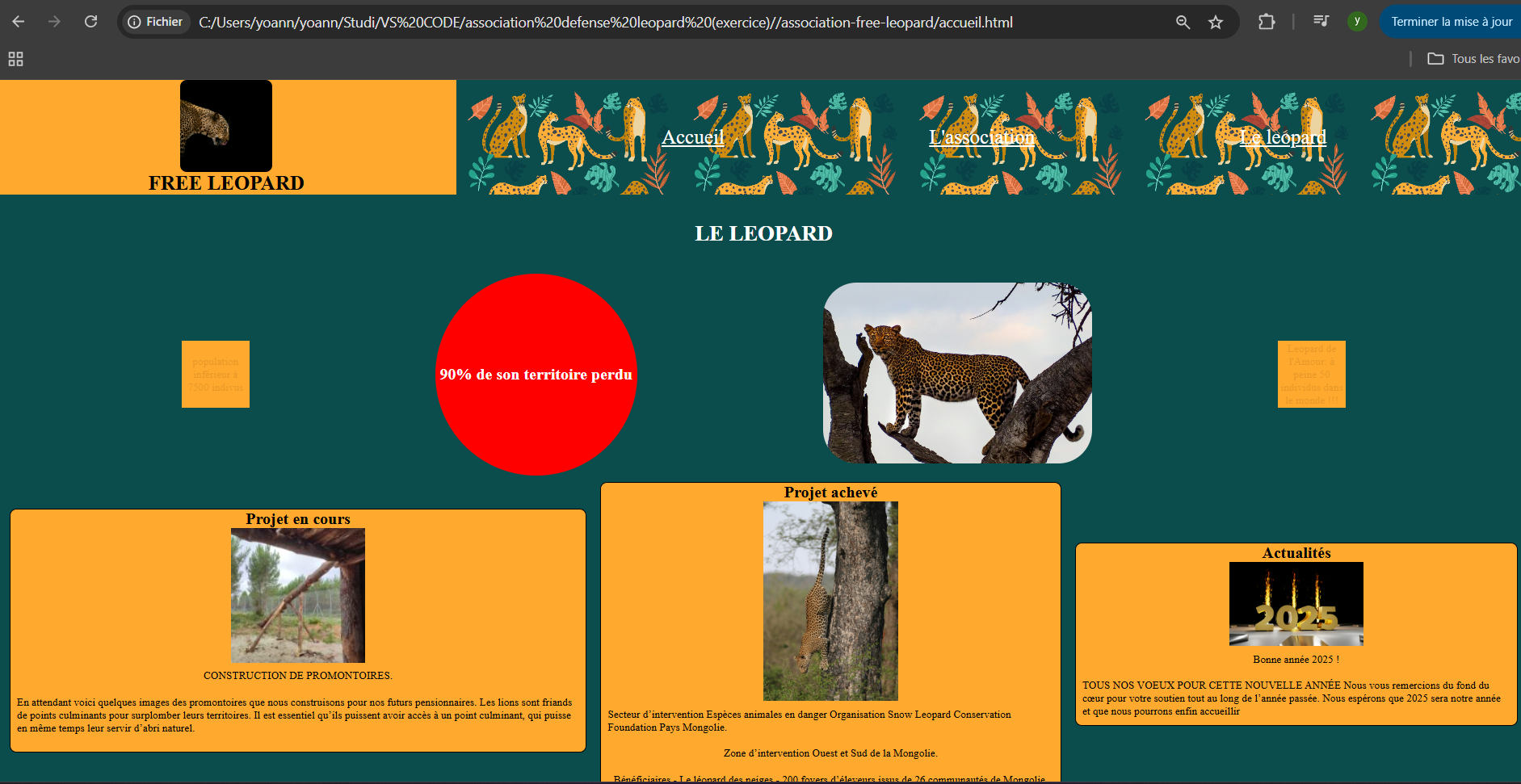
**ANNEXE**



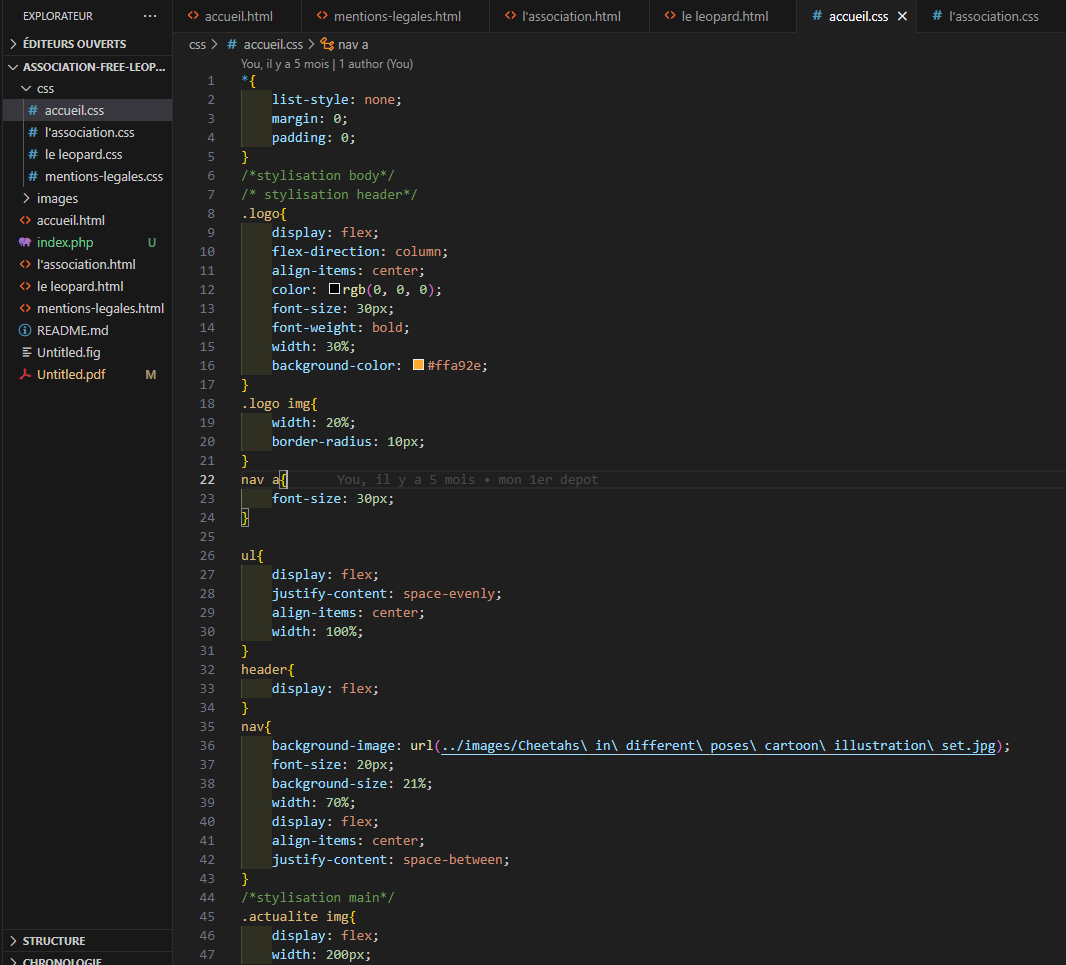
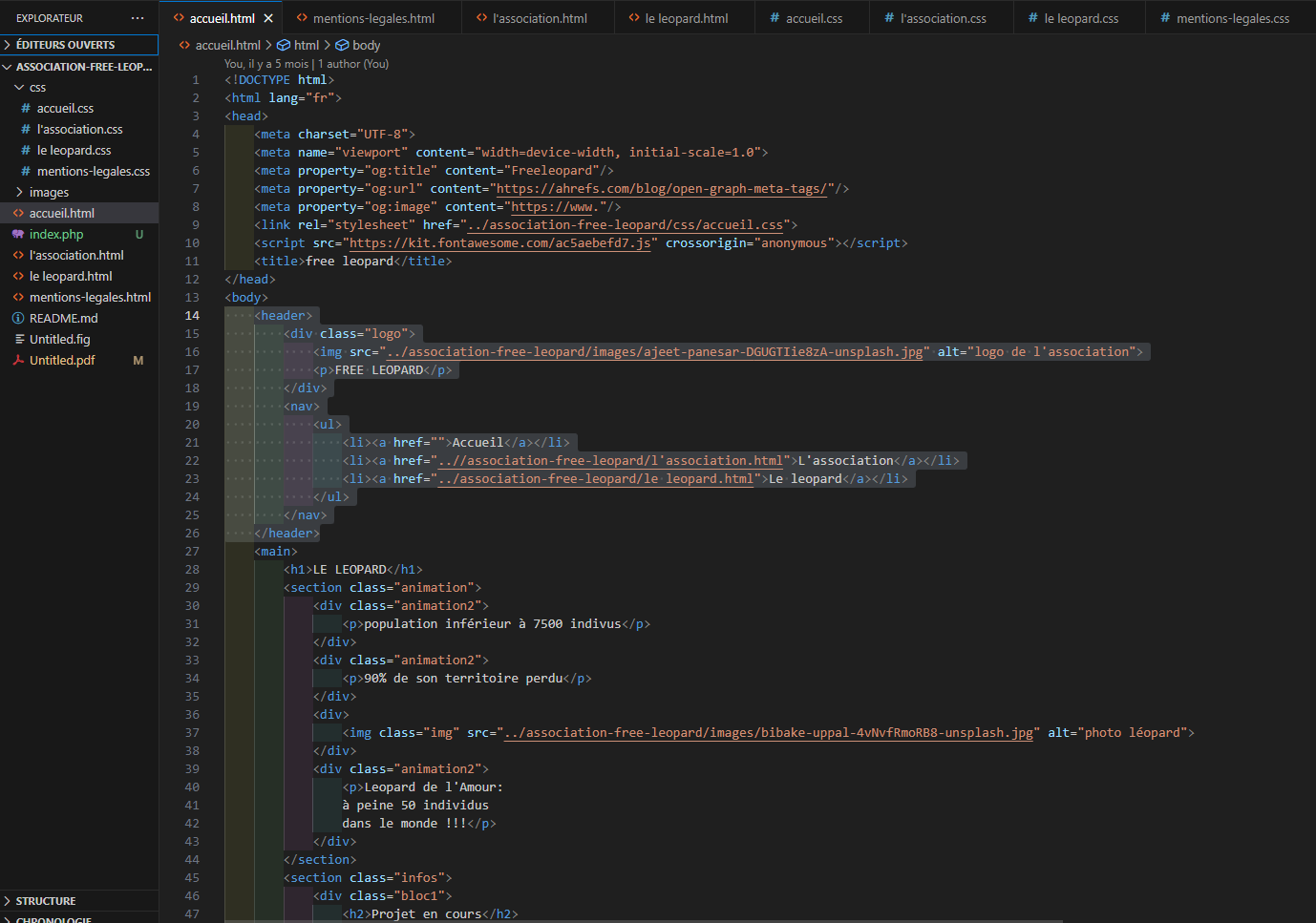
Annexe : Wireframe



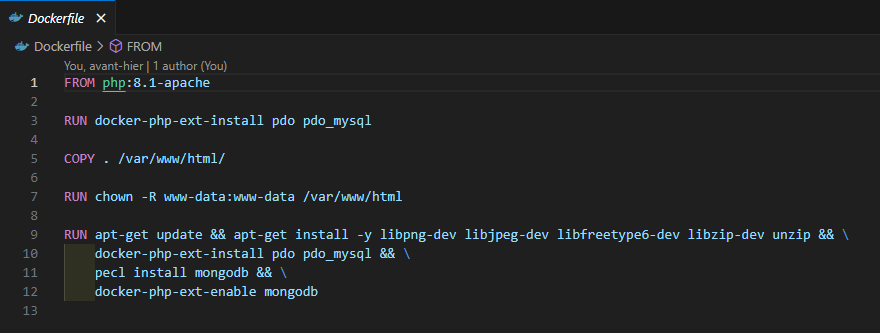
Annexe 2: Mockup



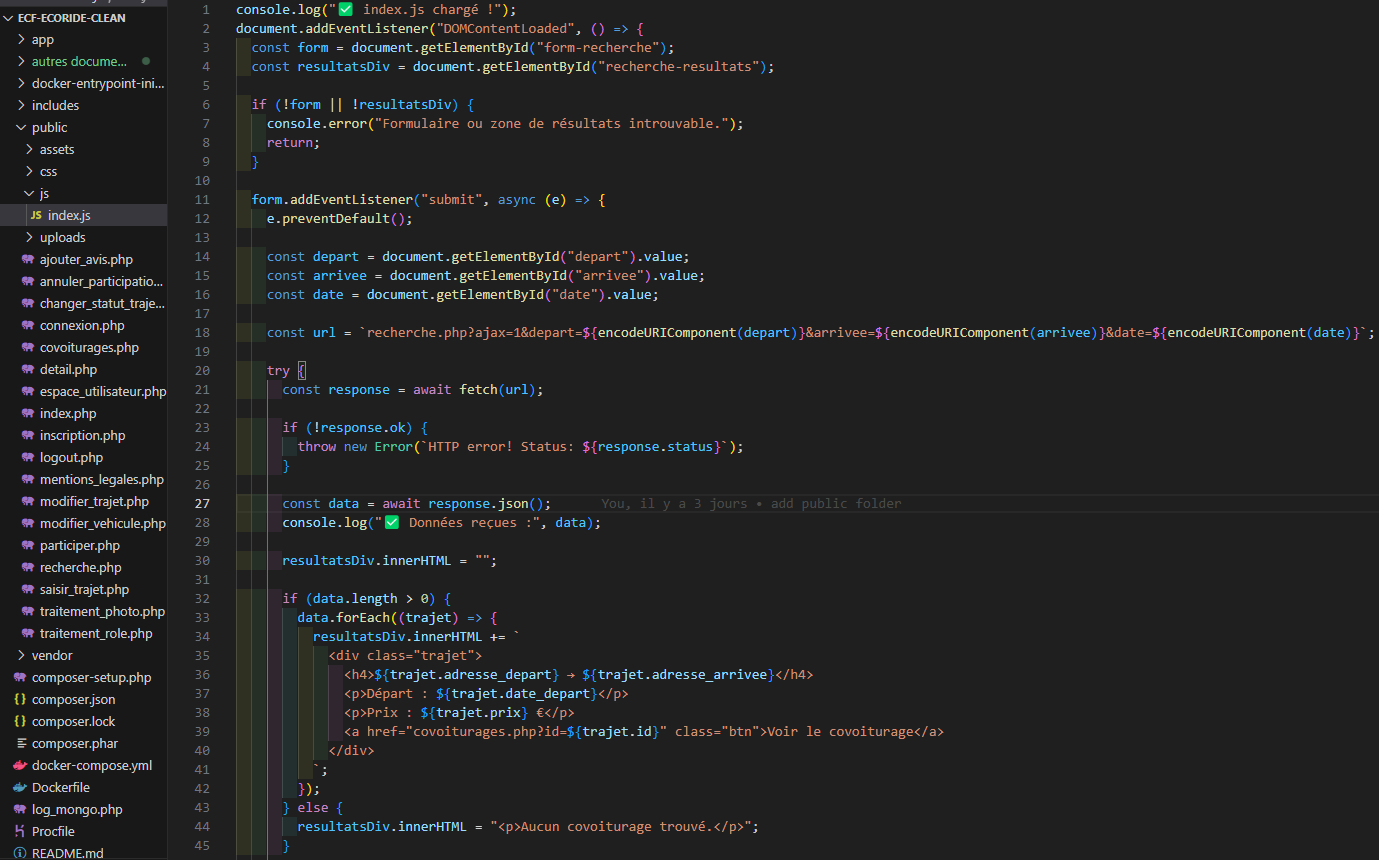
Annexe 3: page accueil finale



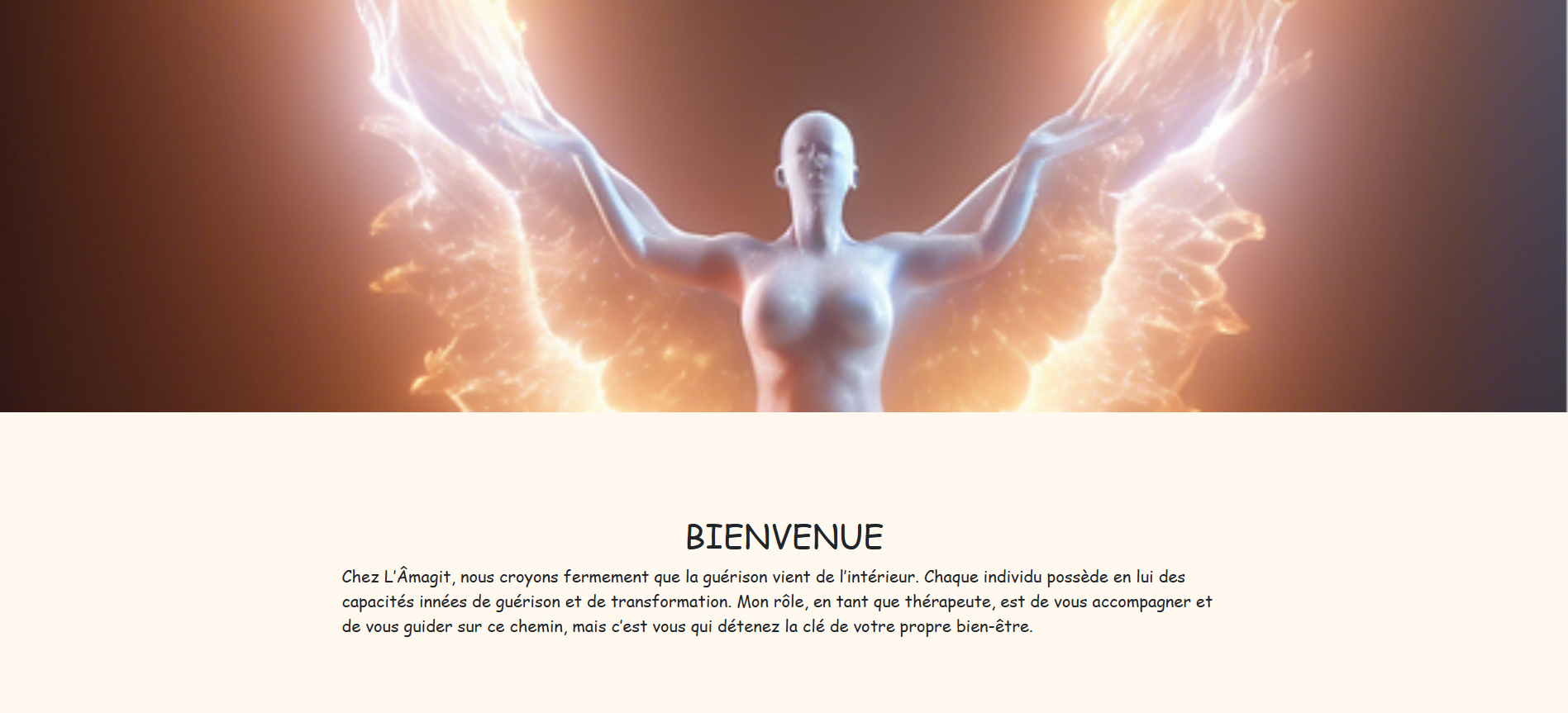
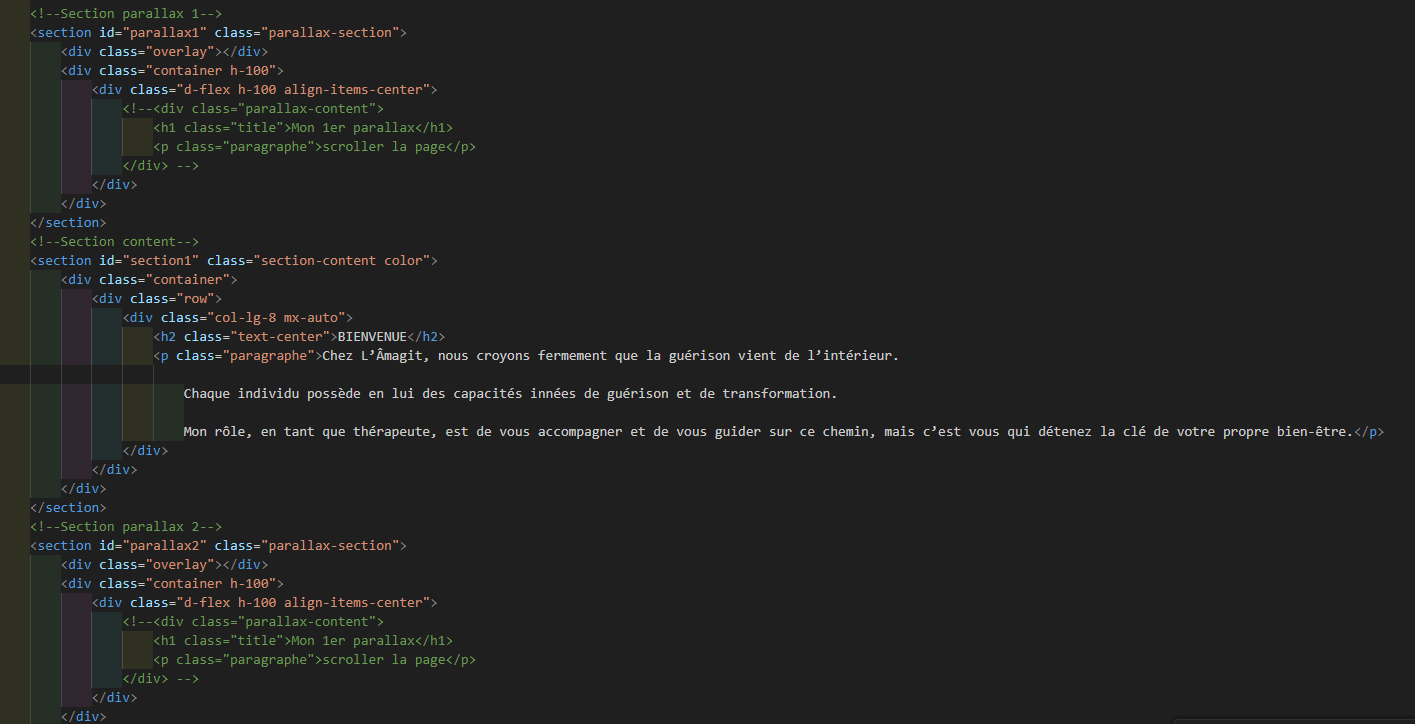
Annexe 4: exemple code sur VS Code



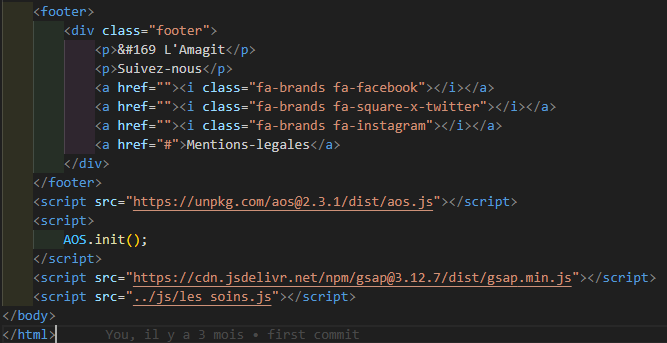
Annexe 5: fichier dockerfile



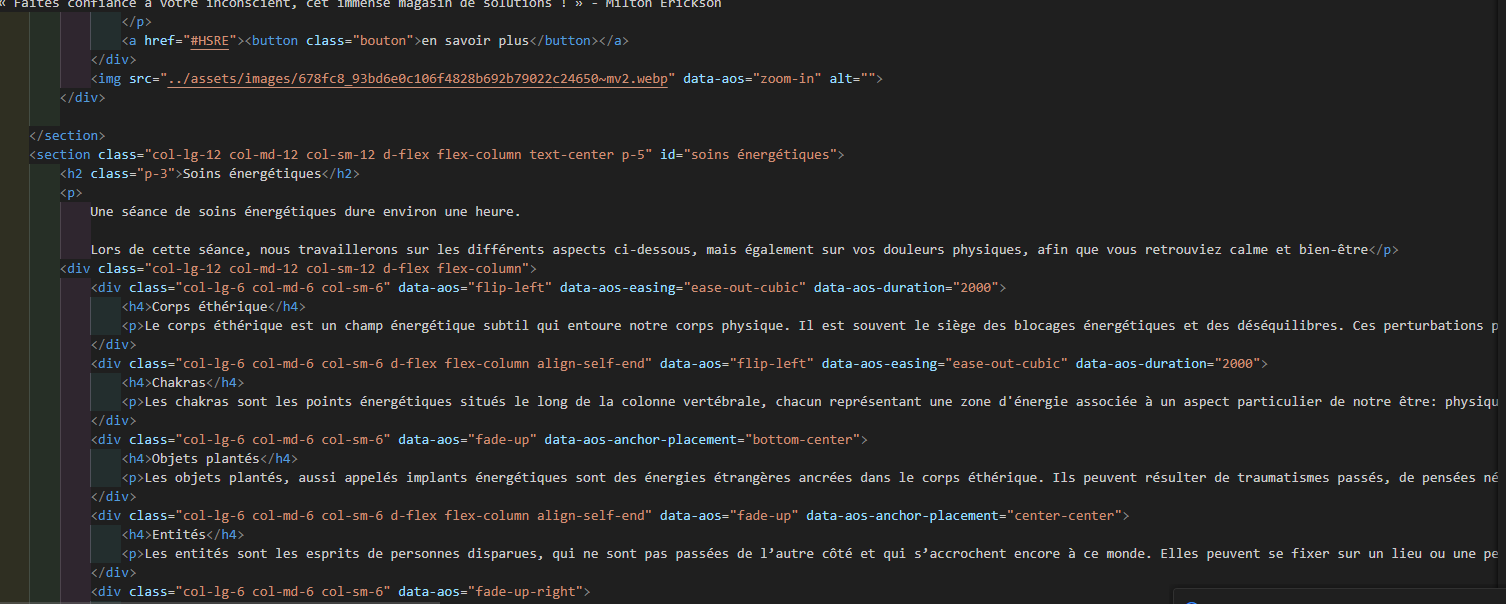
Annexe 6: exemple du sytème AJAX avec "fetch"



Annexe 7: exemple de Parrallax



Annexe 8 : script JS



Annexe 9: exemple de la bibliothèque AOS



Annexe 10 : fichier docker-compose.yml