Documentation Projet Shifumi :

1) Présentation rapide

- Nom du projet : Shifumi interactif
- Objectif : Développer un jeu de Pierre-Feuille-Ciseaux où l'utilisateur affronte une intelligence artificielle (choix aléatoire) sur une série de 10 manches.
- Contexte : Pratique de la manipulation du DOM et de la gestion des événements JS.

2) Fonctionnalités

L'application gère l'intégralité du cycle de jeu :

- Sélection interactive : Choix du signe via des icônes cliquables avec retour visuel (classe selected).
- Intelligence Artificielle : Génération d'un choix adverse aléatoire à chaque tour.
- Comparaison logique : Algorithme déterminant le vainqueur de la manche (Victoire, Défaite ou Égalité).
- Tableau de bord : Mise à jour en temps réel du score, de la manche actuelle et des choix effectués.
- Fin de partie : Alerte de score final et réinitialisation complète du jeu après la 10ème manche.

3) Partie technique simplifiée

- Langages utilisés : HTML5, CSS3, JavaScript (ES6).
- Gestion des données : Utilisation des attributs data-choice dans le HTML pour lier les images aux valeurs logiques en JavaScript.
- Événements (Event Listeners) : * Utilisation de forEach pour écouter les clics sur les images.
    - Déclenchement de la logique de jeu via un bouton principal "Jouer".
- Logique Aléatoire : Emploi de Math.random() combiné à la longueur d'un tableau de chaînes de caractères pour le choix de l'ordinateur.

4) Difficultés / Ce que j'ai appris

Ce projet m'a permis de consolider des bases essentielles du développement Front-End :

- Récupération de données via le DOM : J'ai appris à utiliser e.target.dataset pour extraire des informations directement depuis les éléments HTML, ce qui rend le code plus flexible.
- Gestion des états visuels : J'ai mis en place une logique permettant de "nettoyer" les sélections précédentes (en supprimant les classes CSS) avant d'appliquer la nouvelle, garantissant une interface cohérente.
- Algorithme de décision : J'ai structuré une suite de conditions if / else if / else pour couvrir tous les cas de figure du jeu, en optimisant la détection de victoire du joueur.
- Contrôle du flux de l'application : J'ai géré le cycle de vie d'une partie (10 manches) et utilisé setTimeout pour synchroniser l'affichage des résultats avec les alertes de fin de jeu, améliorant ainsi l'expérience utilisateur.