Amir
HICHAM
12018131

Escube Game

Mon projet web est un jeu de puzzle dans lequel le joueur évolue autour d'un cube et doit déchiffrer différentes énigmes le plus rapidement possible. Ayant déjà une bonne expérience en HTML, CSS et JavaScript, j'ai souhaité découvrir et apprendre quelque chose de nouveau à travers ce projet : **Three.js**.

Three.js est une bibliothèque JavaScript dédiée à la 3D, que j'ai utilisée pour réaliser la majeure partie de ce projet. La scène occupe 100 % de l'écran et comprend une lumière ambiante.

L'univers du jeu se déroule dans l'obscurité, avec des étoiles artificielles générées aléatoirement au lancement du site, à des distances variables du centre de la scène, créant ainsi une impression d'espace étoilé.

La caméra tourne autour du cube de manière régulière et s'anime à chaque frame. L'utilisateur peut également cliquer sur la scène pour prendre le contrôle de la caméra à la souris.

Le cube, composé de plusieurs morceaux, possède des faces indépendantes, chacune accessible différemment. L'objectif était de concevoir six faces distinctes, correspondant à six sections web différentes.

Grâce à une fonction nommée **html2canvas**, il est possible de prendre une « photo » d'une section HTML. J'ai utilisé cette fonctionnalité pour appliquer l'image des différentes div sur les faces du cube. On a ainsi l'impression de quitter le cube, alors qu'en réalité, une div apparaît et disparaît au bon moment.

Lorsqu'on clique sur une face, la caméra se place en face de celle-ci et effectue un zoom jusqu'à occuper 100 % de la hauteur de l'écran. À ce moment-là, la véritable section HTML s'affiche au-dessus de la scène, renforçant l'illusion d'immersion.

L'utilisateur peut alors interagir avec les différentes parties et trouver les codes.

Les codes sont générés aléatoirement à chaque nouvelle session utilisateur, ce qui évite qu'ils restent identiques et limite la triche. Ils sont créés par une fonction PHP, ce qui empêche également de tricher en inspectant le code source de la page ; la vérification s'effectue côté serveur.

À la fin du jeu, les codes sont vérifiés : s'ils sont corrects, le chronomètre s'arrête et le joueur peut entrer un pseudo pour enregistrer son score. Un leaderboard s'affiche alors, présentant les 10 meilleurs temps. Le classement est stocké côté serveur dans un fichier JSON, avec les attributs « pseudo » et « temps ».

Le code est structuré de la façon suivante :

- **index.html** : contient toute la structure des différentes div correspondant aux faces du cube.
- **style.css** : regroupe tous les styles des différentes faces.
- **app.js** : contient la majeure partie du projet, notamment le code pour la 3D, le jeu du Simon et l'intégration des scripts PHP.
- **generate_code.php** : génère les différents codes nécessaires pour terminer le jeu.
- **verify_code.php** : vérifie que les codes saisis sont corrects.
- **leaderboard_add.php** : ajoute un joueur au leaderboard avec son temps.
- **leaderboard_get.php** : fournit la liste des joueurs et leurs temps respectifs dans le leaderboard.
- **leaderboard.json** : stocke les joueurs et leurs temps (en secondes).

J'ai évidemment utilisé des IA génératives pour m'aider à développer ce projet, mais surtout pour apprendre à utiliser les différentes fonctionnalités de THREE.js.

Solutions du jeu :
Sur la première face de description du jeu, il faut cliquer sur le mot « clé ». Ensuite, il faut sélectionner le charabia avec la souris pour faire apparaître le code, et il faut terminer le Simon (5 de suite) pour obtenir un code.