Nom : HABBI
Prénom : Massyl

M1 MIAGE Aix

Rapport de Projet – Dreams Catcher

1. Scénario du jeu

Titre : Dreams Catcher

Concept : Dans un univers onirique, le joueur incarne un avatar humain chargé de collecter des rêves lumineux (sphères colorées) dans un temps limité.

Objectif : Obtenir le meilleur score possible en 30 secondes.

Environnement : Un monde 3D légèrement surréaliste avec un terrain, des obstacles naturels et une ambiance immersive.

Déroulement des parties

Départ : Le joueur commence au centre du terrain.

Interaction : Il utilise les touches ZQSD pour déplacer l'avatar.

Objectifs :

Collecter des sphères rouges (1 pt), bleues (2 pts), jaunes (3 pts)

Les sphères vertes ajoutent 3 secondes au temps de jeu

Victoire : Atteindre un score élevé avant la fin du chronomètre

Défaite : Le temps s'écoule avant d'avoir collecté suffisamment de rêves

2. Description fonctionnelle

Interface utilisateur :

Écran de démarrage avec bouton "Jouer"

Affichage temps/score en haut de l’écran

Écran final avec le score obtenu

Fonctionnalités principales :

Déplacement de l’avatar en temps réel

Collision avec sphères (collecte + mise à jour du score/temps)

Réapparition aléatoire des rêves après collecte

3. Description technique

Technologies utilisées :

Babylon.js : moteur 3D principal (affichage, scène, collisions, logique)

JavaScript : gestion des événements, calculs de score, timer

HTML/CSS : structure de l’interface, affichage des éléments graphiques

Blender : modélisation de l’avatar, terrain et sphères

Structure technique :

index.html : structure de l’interface et import du Canvas Babylon.js

app.js :

Création de la scène

Chargement des modèles 3D

Détection des collisions

Gestion du chronomètre

Mise à jour du score et réapparition des objets

Optimisations :

Objets désactivés plutôt que détruits pour performance

Tests sur Brave Browser et Microsoft Edge

4. Organisation de projet

Développeur unique

Durée : 5 mois

Méthode : Adaptation personnelle de Scrum

Planning réel :

Mois	Activités principales
M1	Lecture doc Babylon.js, rédaction cahier des charges
M2	Modélisation Blender, création scène 3D
M3	Implémentation mouvements, collisions, score
M4	Interface, timer, tests fonctionnels
M5	Débogage, optimisation, rédaction du rapport

Outils :

Visual Studio Code

Blender

GitHub (local, pas de dépôt distant)

ChatGPT, Grok pour détectage de bugs possibles et correction des fautes d’orthographe dans les rapports

5. Ce qui a été fait :

- Modélisation du terrain, avatar et sphères
- Mise en place des mouvements de l’avatar
- Gestion des animations affiché lors du déplacement du joueur
  Apparition aléatoire des sphères sur le terrain
- Mise en place du ciel correspondant au thème du jeu
- Mise en place et masquage de cubes dans les objets du terrain
- Mise en place du chronomètre ainsi que le score
- Mise en place d’une page expliquant à l’utilisateur les règles du jeu
- Mise en place de l’écran d’accueil avec le bouton start au milieu
- Mise en place d’une caméra attaché à l’avatar et qui le suit
- Mise en place de frontière pour que le joueur ne sort pas du terrain
- Mise en place d’un système de point différent pour chaque sphère
- Mise en place des collisions entre l’avatar et les sphères

6. Ce que je n’ai pas eu le temps de finir

- Mise en place des effets sonores
- Mise en place de collisions entre le joueur et les cubes caché dans les objets du terrain pour que le joueur ne les traverse pas
- Travailler plus sur les animations de l’avatar

7. Bilan personnel

Ce projet m’a permis de découvrir en profondeur le moteur Babylon.js, en particulier sa puissance pour créer des mondes 3D interactifs sur le web. Je suis parti d’un niveau débutant dans le domaine du développement 3D et, progressivement, j’ai appris à charger des modèles, gérer des collisions, utiliser des textures et surtout structurer un vrai projet de jeu.

L’une des difficultés majeures a été la gestion des collisions et des performances. Initialement, les objets étaient détruits à chaque collecte, ce qui ralentissait le jeu. J’ai alors appris à désactiver temporairement les objets, les déplacer, puis les réactiver, ce qui a considérablement amélioré la fluidité.

Sur le plan graphique, Blender m’a aussi posé des défis, notamment pour l’import d’animations Mixamo. Mais avec de la persévérance, j’ai réussi à les intégrer dans Babylon.js, en particulier grâce au NLA Editor.

Je suis très satisfait du résultat final. Même si certaines fonctionnalités n’ont pas été intégrées à temps, le cœur du jeu fonctionne : un avatar qui se déplace, collecte, avec un score qui monte et un temps qui descend.

Ce projet m’a aussi appris la patience : tout prend plus de temps que prévu. Mais aussi l’importance de commencer petit (fonctionnalités de base) puis d’ajouter au fur et à mesure, avec des tests fréquents. C’est une leçon que je retiens pour mes futurs projets.

En conclusion, ce projet m’a fait grandir à la fois en tant que développeur, mais aussi en tant que gestionnaire de projet. Je suis désormais plus confiant pour créer des applications complexes, et plus motivé que jamais pour progresser dans le développement de jeux vidéo.

8. Annexes

Vous trouverez dans le fichier zip soumis :

Les codes des classes app.js et index.html

Les fichier ‘.glb’ importé avec Babylon JS qui ont permis d’afficher l’avatar, le terrain et les sphères

Les fichiers blenders des modélisations utilisées

Les sons que j’ai voulu ajouter au jeu
Un ensemble d’image que j’ai utilisé pour les textures du terrain et du ciel

La vidéo de 30 seconde montrant comment jouer au jeu