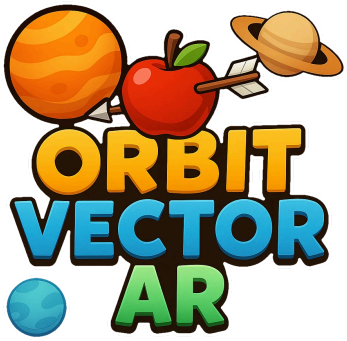
Cahier des Charges – Orbit Vector AR

Auteurs BEAUJARD Traïan, RAMALLO Alex - **Date** 29/04/2025 - **Version** 1.0.1



1. Introduction

Nous proposons de développer un jeu mobile sur Android en réalité augmentée. Le jeu est un jeu de puzzle où le joueur doit tirer des flèches sur des cibles en utilisant la gravité des planètes procéduralement placées dans le décor réel.

2. Objectifs

2.1. Objectifs du projet

Le but de ce projet est d'avoir un jeu créant une expérience unique et originale.

2.2. Objectifs Principaux

- Développer un jeu mobile sur Android utilisant la Réalité Augmentée.
    - L'application doit détecter les surfaces de l'environnement réel et y ancrer des éléments du jeu.
- Implémenter un gameplay de puzzle basé sur la physique.
    - Le joueur doit lancer des flèches dont la trajectoire est influencée par l'attraction gravitationnelle de planètes pour atteindre la cible.
- Intégrer une génération procédurale des niveaux.
    - Les niveaux (position, nombre des planètes, de la cible) doivent être générés procéduralement en fonction de l'environnement détecté.
- Réaliser une interface utilisateur (UI) intuitive et adaptée au contexte d'un jeu mobile et de l'AR.

2.3. Objectifs Secondaires

- Mettre en place un système de score et de classement multijoueur.
    - Le score de chaque joueur est sauvegardé en ligne.
    - Permettre aux joueurs de comparer leurs scores.
- Développer un système monétaire et de personnalisation.
    - Offrir aux joueurs la possibilité d'acheter des skins (flèches, planètes, cibles) avec une monnaie gagnée à chaque partie.

3. Analyse centrée utilisateur

3.1. Personas

Voir projet personnas *(powerpoint)* ici.

3.2. Scénarios d'utilisation

Voir projet scénarios *(powerpoint)* ici.

4. Exigences fonctionnelles

4.1. Cœur du Gameplay "Core"

- Le jeu doit permettre au joueur de lancer des projectiles depuis une position fixe.
    - Les flèches doivent suivre une trajectoire influencée par :
        - Paramètres de lancement de la flèche.
        - L'attraction gravitationnelle simulée des objets placés dans le niveau.
- Le but de chaque niveau est de toucher une cible spécifique avec une flèche.
- Le joueur dispose d'un nombre limité de flèches par niveau.
    - Le jeu affiche un écran de fin quand le joueur n'a plus de flèches. Puis retourne au menu principal créant la boucle de jeu.

4.2. Réalité Augmentée

- L'application doit :
    - Détecter les surfaces planes horizontales dans l'environnement réel de l'utilisateur.
    - Afficher les éléments virtuels du jeu (planètes, flèche de visée, cible, trajectoire prédictive ?) de manière stable par rapport à l'environnement réel.
- L'application doit gérer les conditions de tracking AR (Progression cartographie, ...).

4.3. Génération Procédurale des Niveaux

- L'application doit générer procéduralement la disposition des planètes (nombre, position, masse/taille) et de la cible pour chaque nouveau niveau.
- La difficulté des niveaux doit augmenter progressivement.
- L'algorithme de génération doit s'assurer que chaque niveau généré est solvable.

4.4. Système de Contrôle

- Le joueur doit pouvoir contrôler la direction et la force de tir de la flèche.

4.5. Système de Score

- Le joueur gagne des points à chaque niveau réussi.
- Le score du joueur est sauegardé en ligne à la fin de chaque session de jeu.

4.6. Gestion des Ressources (Flèches)

- Le joueur commence une session avec un nombre défini de flèches.
- Chaque tir consomme une flèche.
- Le joueur gagne des flèches à chaque niveau réussi.

4.7. Conditions de Victoire et de Défaite

- Le jeu se termine lorsque le joueur n'a plus de flèches.

4.8. Économie Interne (Pièces)

- À la fin d'une session de jeu, le score est converti en Pièces (formule à définir).
- Le joueur peut utiliser les Pièces pour acheter des skins cosmétiques (flèches, cibles, planètes).
    - Chaque item doit avoir un coût en Pièces.
    - Le joueur doit pouvoir acheter des items s'il possède assez de pièces.
    - Le joueur doit pouvoir équiper/déséquiper les skins possédés.

4.9. Leaderboard

- L'application doit intégrer un classement des meilleurs scores.
- Le classement doit être accessible depuis le menu principal.
- Il doit afficher le pseudo du joueur et son score.

4.10. Authentification Utilisateur

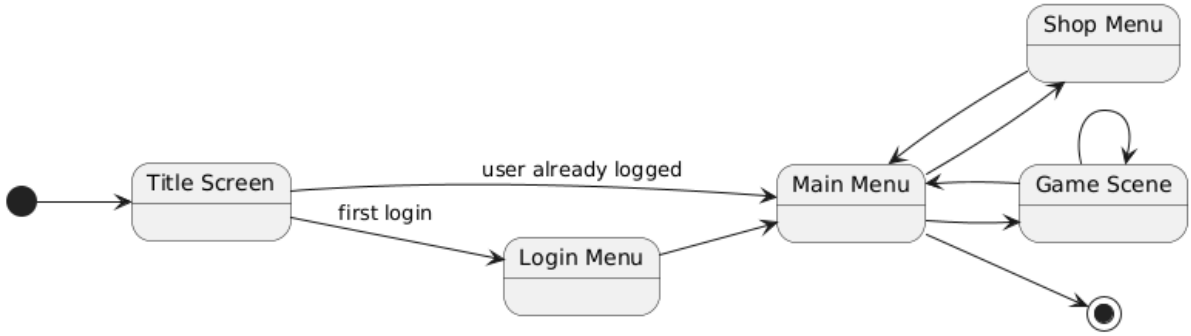
- L'application permet à l'utilisateur de s'authentifier via son uiid et un pseudo.

5. Interfaces utilisateur

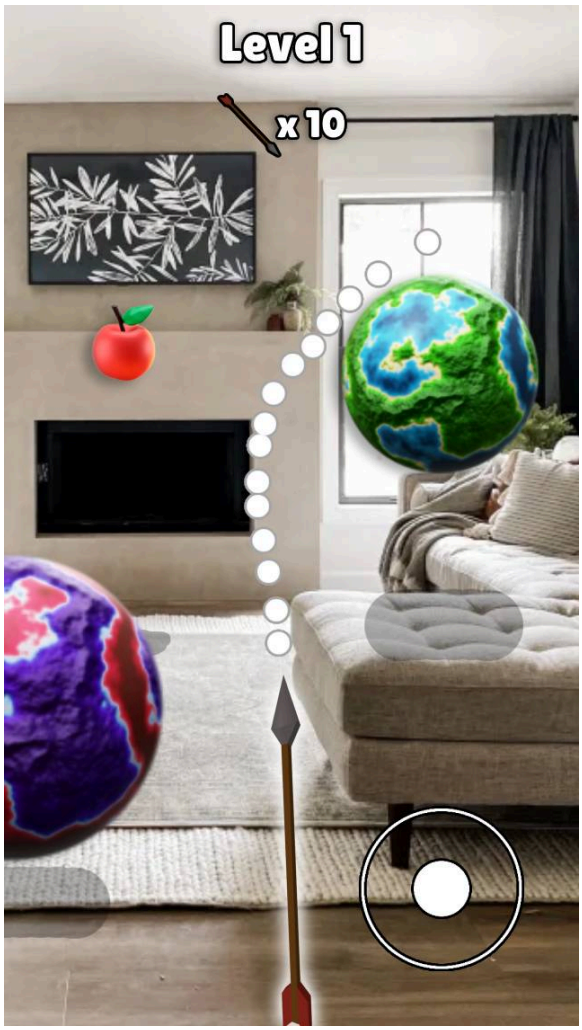
5.1. Objectifs

- Cohérence : Maintenir un style visuel à travers tous les menus.
- Simplicité : Éviter la surcharge d'informations, privilégier des indications visuelles, claires et intuitives.
- Feedback : Fournir des retours visuels et sonores lors des interactions.
- Adaptation AR : Concevoir une interface ergonomique malgré les contraintes de la réalité augmentée.

5.2. Les menus et la navigation



5.3. Prototype de la scène de jeu



6. Spécifications techniques

6.1. Outils de développement

6.1.1. Outils de programmation

- Plateforme : Android

- Langage : Kotlin
- Framework : ARCore
- Base de données : MySQL
- API : PHP
- Versioning : GitHub
- IDE : Android Studio/Visual Studio Code
- Assistant de développement : Github Copilot

6.1.2. Outils de conception

- Logiciel modélisation 3D : Blender
- Logiciel graphique : Photopea

6.2. Matériel

- Le serveur sera hébergé sur une raspiberry pi 4b sous Ubuntu Server.

7. Plan de travail

Le projet sera réalisé pour 2 matières (SY43 x HM40), ainsi les tâches seront réparties entre les deux matières avec un appui sur l'interface homme-machine pour l'UV H40 et la logique et technologie de jeu pour l'UV SY43.

7.1. Répartition des tâches

Projet à 4 élèves, *2 des personnes du groupe ne sont pas dans l'UV HM40*.

- **Alex Ramallo** & **Elève 3** : interfaces utilisateur (menus, navigation, boutique) via Jetpack Compose.
- **Beaujard Traïan** : logique du jeu, intégrationd de la réalité augmentée, interface utilisateur en réalité augmentée, intégration des systèmes leaderboard, paramétrage du serveur, ...
- **Elève 4** : base de données, API PHP, intégration SQL (authentification, leaderboard).

7.2 Étapes du projet

Étape 1. : Préparation (jusqu'au 30 avril)

- Rédaction cahier des charges et game design document.
- Préparation des environnements de développement (repos git, serveur, librairies, ...).
- Tests sur le développement de l'AR.
- Début du développement :
    - Prototypes d’écrans.
    - Début de la logique du jeu.

Étape 2. : Développement prototype (1-19 mai)

- Core gameplay AR (physique, tir de flèches, gravité des planètes).
- Boucle de jeu complète (menu principal, jeu, fin de niveau).
- Maquettes finalisées pour tous les écrans + Intégration dans le jeu.
- Intéraction API PHP et base de données (authentification).

→ L'objectif est d'avoir un Minimum Viable Prototype (MVP) jouable à la fin de cette phase.

Étape 3 : Intégration et ajustements (20-28 mai)

- Tests d’usage et ajustements IHM.
- Intégration du système audio.
- Rédaction du rapport sur les choix IHM.

Étape 4 : Améliorations & Finalisation (29 mai - 14 juin)

- Itération auprès d'utilisateurs variés pour récupérer feedback utilisateur.
- Corrections de bugs, optimisation des performances et polissage du jeu.
- Préparation de la soutenance (slides, démos, livrables).