**Exigences Fonctionnelles Complètes (Vision long terme)**

Les exigences devant être supporter par la plateforme quand elle sera pleinement développée au delas des 4 mois

**Création des jeux :**

L’utilisateur peut créer un jeu de A à Z, incluant :

* Carte du monde (les niveaux)
* Positions des objets, personnages, obstacles
* Choix de mécaniques de jeu
* Choix de l’apparence et des règles du jeu

**Configuration des comportements NFC :**

Chaque carte NFC est liée à un effet spécifique (bonus, malus, environnement, etc)

Après scan de la carte :

* Requête envoyée à l’API
* L’effet est traité côté backend
* Le frontend applique l’effet en temps réel

**Éditeur du monde :**

Interface glisser-déposer pour :

* Ajouter des éléments (objets, montres, herbes …)
* Définir la taille, la position et la couleur
* Sauvegarder plusieurs versions d’un jeu (A voir)

**Gestion de projets :**

* Liste des jeux créés
* Modification ou suppression d’un jeu existant
* Affichage en mode “joueur” (playtest)

**Multijoueur / Websockets (option futur) :**

* Support de jeu multijoueur local ou distant
* Envoi des effets de cartes entre joueurs

**Administration : (à voir)**

Tableau de bord pour visualiser :

* + L’utilisation des cartes
  + Historique des effets scannés
  + Statistiques de jeux créés

**Exigences fonctionnelles à court terme (4 mois)**

Version minimum viable pouvant être livré

**Création simple de jeu :**

Création d’un jeu avec :

* + Nom
  + Description
  + Catégorie
  + Ajout de quelques éléments de jeu de base (portes, monstres, blocs)
  + Ajouter un niveau unique (pas de multi-niveaux à ce stade)

Permettre à l’utilisateur d’ajouter des éléments sur une grille interactive.

Fonctionnalités :

* Grille 2.5D
* Palette d’éléments :
  + Blocs (sol, mur, plateforme)
  + Objets interactifs (porte, téléporteur)
  + Ennemis (avec comportement de base)
  + Départ / arrivée (début et fin de niveau) (définir élément de fin)
* Actions disponibles :
  + Glisser-déposer un élément sur une case
  + Définir la taille et la couleur d’un objet
  + Supprimer un objet
  + Enregistrer les modifications
  + Logique conditionnelle

Éléments de jeu disponibles :

* Objets standards : porte, bloc, obstacle, ennemi
* Dimensions ajustables : largeur/hauteur
* Positionnement libre sur une grille de type 2D

État du jeu ne dépend pas des zones

Restrictions :

* Aucune animation personnalisée
* Système de règles complexes (gravité, score, checkpoint)
* Certains effets sont cumulables jusqu'à 2 ou 3 fois
* Certains effets sont superposables et d’autres ne le sont pas

**Détection NFC avec un effet simple :**

App React Native capable de :

* + Scanner une carte NFC
  + Associer cette carte à un identifiant
  + Appliquer un effet affiché à l’écran

**Application d’effets simples (positifs/négatifs) :**

* 5 ou 10 effets maximum dans le produit minimum viable
* Effets définis dans un fichier statique (JSON)
* Chaque carte NFC est liée à un effet (base clé-valeur)

Exemple d'effets à implémenter :

* + Double saut
  + Gravité augmentée
  + Inversion des commandes
  + Turbo Boost

**Interface minimaliste de test :**

* Menu : “Créer un jeu”, “Jouer à un jeu”
* Liste de jeux créés
* Visualisation simple du monde et des effets
* Icône de lecture rapide (Play) pour tester le jeu

Éditeur de jeu **:**

* Zone de création en grille 2D
* Palette d’objets prédéfinis à glisser
* Bouton “Sauvegarder”
* Bouton “Tester” pour lancer une simulation

Mode test :

* Petit joueur qu’on verra ou contrôlable à l’écran(dépend de ce que l’utilisateur a mis )
* Scan d’une carte déclenche une animation ou un message Effet appliqué : Turbo Boost
* Bouton “Retour”

**Backend de base (Node.js ou Flask) :**

API :

* + Envoi d’effet par ID de carte
  + Récupération des effets
  + Mise à jour d’un jeu

**Base de données simple :**

Stockage de :

* + Utilisateurs (au moins en local)
  + Jeux créés
  + Cartes NFC et effets associés

**Exigences non fonctionnelles de la plateforme**

**Sécurité :**

* Les données échangées entre l’application et le backend doivent être **sécurisées** (ex : HTTPS)
* L’accès aux jeux créés doit être restreint à l’utilisateur connecté sur la plateforme eweb.
* Les requêtes API doivent être protégées contre les usages non autorisés.

**Compatibilité :**

* À court terme : l’application doit fonctionner sur IOS (via React Native).
* À long terme : l’application sera aussi compatible Android.

**Performance :**

* Le scan NFC et l’application des effets doivent être traités en moins d’une seconde.
* Le rendu des effets dans le jeu doit être fluide et sans latence visible.
* Le backend doit répondre aux requêtes API en moins 1s (je suppose mdr)

**UX :**

* L’interface utilisateur doit être intuitive et accessible à un public non technique.
* Le processus de création de jeu doit être simple, guidé et sans complexité excessive.

**Maintenabilité et évolutivité :**

* Le code backend et frontend doit être documenté et organisé pour faciliter les mises à jour.
* La plateforme doit permettre d’ajouter facilement de nouveaux effets ou objets.

**Testabilité**

* Chaque composant doit pouvoir être testé indépendamment (API, affichage, effets…).
* Des tests utilisateurs seront réalisés pour valider l’expérience de création de jeu.

**Accessibilité :**

L’application devra être utilisable par tout public (A voir si cette phrase est correct), avec une attention à :

* + Contrastes visuels suffisants
  + Texte lisible
  + Navigation simple