REPUBLIQUE DU CAMEROUN

****

PAIX- TRAVAIL- PATRIE

****

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

****

UNIVERSITE DES TIC

****

DEPARTEMENT DES TIC

REPUBLIC OF CAMEROON

****

PEACE- WORK- FATHERLAND

****

MINISTER OF HIGHER EDUCATION

****

ICT UNIVERSITY

****

ICT DEPARTMENT

ALGORITHMIQUE ET STRUCTURES DE DONNEES II

PROJET :

SIMULATION D'UN JEU DE CARTE

Supervisé par : INGENIEUR ATANGANA

SEMESTRE :

PRINTEMPS 2025

Noms des participants

- HANGBAN JEAN CLAUDE (chef de groupe)
- BAYIGA BOGMIS IVAN
- GLORIA CHIKOAM TCHAKOUNTE
- BELLE EBULE MADELEINE SHEBANIA
- DEGNON KOMI PRINCE EMILE

CAHIER DE CHARGE DU PROJET

Sommaire

INTRODUCTION

1. But du document

Le cahier de charge est un document élaboré lors du cadrage d'un projet qui a pour objectif de définir de manière complète, claire et non ambiguë les exigences fonctionnelles et non fonctionnelles du jeu de cartes TriPeaks, destiné à être développé comme une application logicielle multiplateforme et ludique.

2. Portée du projet

Le projet consiste à développer un jeu de cartes inspiré du solitaire Tripeaks. Il s'agit d'un jeu de carte ; l'application sera donc conçue pour des ordinateurs et distribuée gratuitement sans publicité intrusive, l'objectif étant de fournir une expérience utilisateur fluide, intuitive et addictive, avec des mécaniques de jeu fidèles aux règles classiques du solitaire TriPeaks, des effets visuels attrayants, un suivi des scores et des sauvegardes.

3. Définition acronymes et abréviations

Tripeaks : variantes du solitaire avec 3 pics de cartes disposés en pyramides.

SRS : Software Requirements Specification.

IEEE 830 : Norme de spécification des exigences logicielles.

ISO/IEC/IEEE 29148 : Norme de processus pour les exigences des systèmes et logiciels.

UI : interface utilisateur.

UX : expérience utilisateur.

API : Application Programming Interface (interface de programmation d'application).

4. Références

Règles officielles du jeu TriPeaks (Microsoft et jeux de société).

Sons libres de droits et musique : www.pixabay.com

Images : www.google.fr

5. Vue d'ensemble du reste du document

Le document sera structuré comme suit : Description générale : (Contexte du Produit, besoins des utilisateurs et objectifs du projet, caractéristiques du produit, contraintes, hypothèses et dépendances) ; Exigences spécifiques ; Préparation à la livraison ; Annexes.

DESCRIPTION GENERALE

1. Contexte du produit

Le jeu Tripeaks est une variante populaire du solitaire qui propose une expérience plus dynamique et visuellement attrayante que le solitaire classique... ce projet vise à proposer une version gratuite, fluide, esthétique et accessible du Tripeaks pour ordinateur. L'accent est mis sur la simplicité, la rapidité de prise en main et la disponibilité hors ligne.

2. Besoins des utilisateurs et objectifs du projet

a. Besoin des utilisateurs

Les utilisateurs ont besoins d'un jeu rapide à prendre en main sans tutoriel complexe, d'une interface claire, intuitive et visuellement attrayante, d'un fonctionnement sans connexions internet et sans option de paiement pour jouer.

b. Objectifs du projet

Notre projet a donc pour objectif de développer une application de jeu destinée aux ordinateurs et d'assurer la stabilité et la jouabilité hors ligne.

3. Caractéristiques du produit

Plateforme : ordinateur.

Mode solo uniquement.

Interface : écran d'accueil, score.

Cartes : 52 cartes standards.

Pioche et carte active visible.

Fonctionnalités de sauvegarde et de chargement des parties.

Effets visuels et sonores.

Interface utilisateur intuitive.

Animation simple pour chaque mouvement.

Score basé sur : rapidité, série d'enchaînements, nombre de cartes restantes.

Langue : Français.

4. Contraintes

Multiplateforme : Windows, Linux, MAC

Interface localisée (monolingue)

Respect de la consommation mémoire limitée (< 150MB RAM)

Lancement rapide ;

Aucune publicité intrusive.

5. Hypothèses et dépendances

- **Hypothèses**

Pour assurer le projet, un certain nombre d'hypothèses sont définies :

- Les utilisateurs doivent posséder un ordinateur et un système d'exploitation compatible ;
- Le jeu doit être utilisé hors ligne ;
- Les ressources graphiques doivent être disponibles ;
- L'utilisateur a un accès local à l'application ;
- Le jeu ne doit contenir aucune publicité, ni offres d'achat.

- **Dépendances**

Le projet peut être utilisé par des joueurs occasionnels (grand public), des joueurs réguliers recherchant des défis ou des classements, des testeurs et développeurs.

Le projet dépends de la disponibilité continue de l'équipe projet et des testeurs pour la validation utilisateur finale.

Le projet est développé en C++ à l'aide de la bibliothèque QT.

EXIGENCES SPECIFIQUES

1. Exigences fonctionnelles

ERFT01 – Distribution initiale des cartes

Identifiant unique : ERFT01

Description détaillée : Le système doit générer une grille de 3 pics avec 28 cartes disposées selon une structure pyramidale. Le reste du paquet (24 cartes) constitue le talon ou banque.

Acteurs impliqués : Joueur

Préconditions : Le joueur a lancé une nouvelle partie ou à recommencer une partie en cours.

Déclencheur : Clic sur le bouton "Nouvelle partie" ou "recommencer".

Séquence normale des événements :

- Le jeu mélange les 52 cartes.
- Il en distribue 28 sur la grille (TriPeaks).
- Les cartes du bas de chaque pic sont visibles.
- Les cartes restantes sont affectées au talon.

Flux alternatifs ou exceptions : Si le jeu ne contient pas 52 cartes, afficher une erreur et empêcher la distribution.

Postconditions : Une nouvelle partie commence avec une grille valide et un talon prêt.

Priorité : Essentielle

Critères d'acceptation : Les 28 cartes sont bien positionnées, les visibles sont sélectionnables, le talon est complet avec les cartes restantes.

ERFT02 – Sélection et retrait de carte

Identifiant unique : ERFT02

Description détaillée : Le joueur peut sélectionner une carte visible de la grille si sa valeur est immédiatement supérieure ou inférieure à la carte actuelle du tas.

Acteurs impliqués : Joueur

Préconditions : Une carte est visible et accessible (non recouverte).

Déclencheur : Clic sur une carte de la grille.

Séquence normale des événements :

- Le joueur clique sur une carte visible.
- Le système vérifie la différence de plus ou moins 1 avec la carte actuelle du tas.
- Si valide, la carte est déplacée vers le tas, la carte en dessous devient visible.

Flux alternatifs ou exceptions :

Si la carte n'est pas sélectionnable, elle ne réagit pas.

Si aucune carte n’est disponible, le joueur pioche une carte du talon.

Postconditions : La carte est retirée de la grille, le score est mis à jour.

Priorité : Essentielle

Critères d'acceptation : La carte est retirée uniquement si la règle de différence plus ou moins 1 est respectée et si elle était visible.

ERFT03 – Pioche d'une carte du talon

Identifiant unique : ERFT03

Description détaillée : Le joueur peut piocher une carte du talon lorsque aucune carte de la grille n’est jouable.

Acteurs impliqués : Joueur

Préconditions : Le talon n’est pas vide.

Déclencheur : Clic sur le talon.

Séquence normale des événements :

- Le joueur clique sur le talon.

- La première carte du talon devient la carte actuelle.

Flux alternatifs ou exceptions :

Si le talon est vide, rien ne se passe ou un message d'erreur s'affiche.

Postconditions : Une nouvelle carte devient active.

Priorité : Importante

Critères d'acceptation : Le talon se vide progressivement, et la carte actuelle est mise à jour.

ERFT04 – Condition de victoire

Identifiant unique : ERFT04

Description détaillée : Le joueur gagne la partie si toutes les cartes de la grille sont retirées.

Acteurs impliqués : Joueur

Préconditions : Au moins une carte est encore présente dans la grille.

Déclencheur : Le joueur retire la dernière carte de la grille.

Séquence normale des événements :

- Le joueur joue une carte.
- Le système détecte que la grille est vide.
- Le système affiche un message de victoire et le score final.

Flux alternatifs ou exceptions : Aucun.

Postconditions : La partie est terminée.

Priorité : Essentielle

Critères d'acceptation : Une fois la grille vide, une victoire est automatiquement déclarée.

ERFT05 – Condition d'échec

Identifiant unique : ERFT05

Description détaillée : Le joueur perd la partie si toutes les cartes du talon sont retirées et qu'il y a encore des cartes sur la grille.

Acteurs impliqués : Joueur

Préconditions : Au moins une carte est encore présente dans le talon.

Déclencheur : Le joueur retourne la dernière carte du talon.

Séquence normale des événements :

- Le joueur joue une carte.
- Le système détecte que le talon est fini.
- Le système détecte des cartes sur la grille.
- Le système affiche un message d'échec.

Flux alternatifs ou exceptions : Aucun.

Postconditions : La partie est terminée.

Priorité : Essentielle

Critères d'acceptation : Une fois le talon fini et des cartes présente sur la grille, un échec est automatiquement déclaré.

2. Exigences non fonctionnelles

Exigences de Performance

Le temps de réponse entre une action du joueur (cliquer une carte, piocher) et la mise à jour de l'interface ne doit pas dépasser 150 ms.

Le jeu doit être fluide sur un système avec CPU ≥ 1.5 GHz et RAM ≥ 2 Go.

La consommation mémoire ne doit pas excéder 150 Mo pendant une session normale.

Exigences de Sécurité

Les fichiers de score ou sauvegarde ne doivent pas être modifiables par un utilisateur non autorisé (en lecture seule ou protégés).

L'application ne doit pas permettre d'exécuter du code externe via les fichiers de configuration.

Aucun accès réseau ne doit être ouvert.

Exigences de Fiabilité

Le système doit fonctionner au moins 99 % du temps sans erreur critique pendant une session de jeu standard.

En cas de fermeture inattendue, la dernière partie doit pouvoir être restaurée automatiquement si une sauvegarde automatique est activée.

Le système doit gérer gracieusement les exceptions (pioche vide, mouvement illégal, etc.).

Exigences d'Ergonomie et d'Interface Utilisateur

L'interface utilisateur doit être claire, intuitive et permettre un apprentissage autonome en moins de 2 minutes.

Le joueur doit pouvoir utiliser le jeu à la souris ou au clavier (accessibilité).

Les éléments interactifs doivent être clairement visibles, avec des animations douces au survol.

Exigences de Maintenabilité

Le code source doit être commenté selon les conventions C++.

L'architecture du code doit permettre d'ajouter des variantes de jeu (ex : TriPeaks inversé) sans refondre tout le projet.

Une documentation développeur doit être fournie pour faciliter les modifications futures.

Exigences de Portabilité

Le jeu doit fonctionner sur les systèmes Windows, Linux et Mac (grâce à Qt ou une bibliothèque multiplateforme).
La compilation doit être possible avec CMake ou qMake sur tous les systèmes supportés.

Exigences de Conformité

Le projet doit respecter les standards ISO C++17 ou supérieur.
Il ne doit pas intégrer de bibliothèques tierces sans licences open source compatibles (MIT, BSD, GPLv3).
Le projet doit respecter les règles de conception Open/Closed, DRY et SOLID autant que possible.

3. Exigences d'interface externe

Interfaces Utilisateur (UI)

Le jeu devra proposer une interface graphique (GUI) basée sur Qt (ou équivalent), composée de :

- Une fenêtre principale contenant la grille de jeu, le talon et le score.
- Un menu principal avec les options suivantes : Nouvelle Partie, Charger Partie, Sauvegarder, Options, Quitter.
- Un panneau comment jouer et à propos expliquant le jeu et ses règles.
- Des scores et des cartes dans le talon.

Interfaces Matérielles

Le jeu devra pouvoir être utilisé avec :

- Une souris (clic gauche pour sélectionner une carte).
- Un clavier (raccourci clavier pour quitter).

Interfaces Logicielles

Le projet doit pouvoir s'intégrer à un système de gestion de scores ou de sauvegardes.
L'accès aux fichiers doit passer par une couche logicielle abstraite pour permettre l'adaptation à une base de données future si nécessaire.

Interfaces de Communication

Aucune communication réseau n'est requise pour la version locale.
Si une version en ligne est envisagée plus tard, les échanges devront se faire via WebSocket ou API REST, avec JSON comme format de données.

4. Autres exigences

- Le jeu inclura une interface en français ;

- Le jeu inclura un système de scores ;
- Le jeu inclura des effets sonores (clics de cartes, victoire, erreur) et une musique de fond et l'utilisateur pourra activer/désactiver séparément les effets sonores et la musique ;
- Le système devra permettre de sauvegarder l'état actuel d'une partie (cartes visibles, score, talon…) et de charger une partie sauvegardée à tout moment depuis le menu.

PREPARATION A LA LIVRAISON

1. Plan de déploiement

Le jeu sera livré sous forme d'un exécutable multiplateforme (Windows/Linux/Mac) accompagné d'un installeur graphique.

Un script CMakeLists.txt (ou .pro.user) permettra la compilation sur différents environnements.

Aucun accès Internet requis à l'installation.

Aucune option d'achat, ni de publicités requises après installation.

2. Exigences de formation

Une option comment jouer sera disponible dans le jeu et fera office de tutoriel statique.

Une documentation utilisateur en PDF ou HTML décrira les règles du jeu, les commandes, les erreurs fréquentes.

3. Exigences de documentation

Guide d'installation pour chaque OS (Linux/Windows/Mac) ;

Documentation technique pour les développeurs ;

Fichier README incluant les objectifs du jeu, architecture du projet, compilation, licences fournit lors de la livraison du jeu.

4. Exigences de maintenance

Le code doit être modulaire (ex. : classes Carte, Grille, Talon…), pour faciliter les mises à jour futures ;

Etablissement de mises à jour correctives en cas de bugs signalés.

Un système de logs internes devra permettre d'identifier les erreurs facilement.

Les sauvegardes de partie devront être compatibles avec les futures versions.

ANNEXE

1. Glossaire détaillé

TriPeaks : Variante du solitaire basée sur trois pyramides de cartes à vider en suivant une séquence ascendante ou descendante.

Talon : Tas de cartes à retourner pour obtenir une nouvelle carte jouable.

Grille de jeu : Ensemble des cartes disposées en forme de trois pics.

Carte visible : Carte qui n'est pas recouverte par d'autres cartes.

Mouvement valide : Déplacement d'une carte sur une autre si leur valeur est directement consécutive.

Partie : Session de jeu avec score et état sauvegardable.

Défausse : c'est l'emplacement ou l'on place les cartes défaussées c'est-à-dire les cartes que le joueur retire volontairement ou obligatoirement du talon.

Interface utilisateur (UI) : Partie visible avec laquelle l'utilisateur interagit (boutons, cartes...).

Websocket : c'est un protocole de communication qui permet d'établir une connexion bidirectionnelle persistante entre un client (généralement un navigateur web) et un serveur.

API REST : c'est une interface de programmation qui suit les principes de l'architecture REST (Representational State Transfer), un style d'architecture pour les services web, permettant à différentes applications de communiquer entre elles via le protocole http, en échangeant des données souvent au format JSON ou XML.

JSON (JavaScript Object Notation) : c'est un format léger de représentation de données structurées, facile à analyser et générer pour les machines. Il est très utilisé pour échanger des données entre un client (navigateur, application) et un serveur, notamment dans les API REST.

Règles de conceptions (Open/Closed, DRY et SOLID) : se sont les bonnes pratiques de développement visant à rendre le code plus maintenable, modulaire, évolutif et facile à comprendre.

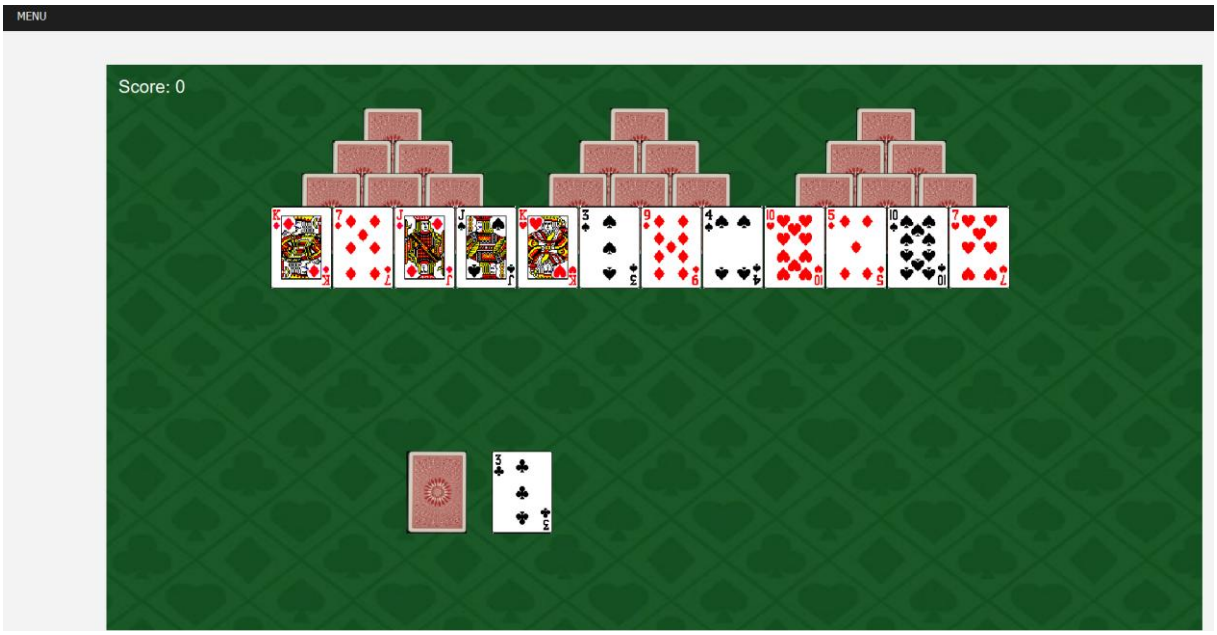
Les standards ISO C++17 : C'est l'évolution importante du langage après C++14, apportant à la fois des améliorations de performance, de syntaxe, et de nouveaux outils pour les développeurs.

Licences open source compatibles (MIT, BSD, GPLv3) : se sont des licences qui permettent au code d'être intégré, combiné ou redistribué légalement avec du code sous d'autres licences, sans conflit juridique.

2. Prototypes ou maquettes d'interface

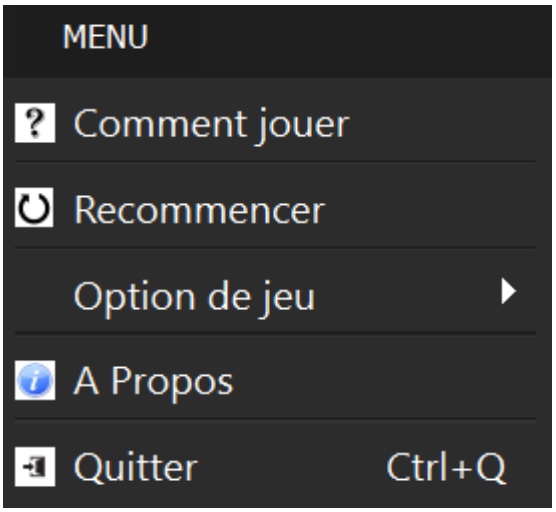
L'écran de jeu

Il est constitué d'une zone centrale contenant les 3 pyramides de cartes ; le talon(pile) en bas ; le score en haut et la barre de menu. Les cartes du bas des pyramides doivent être visibles et la carte défaussée aussi.



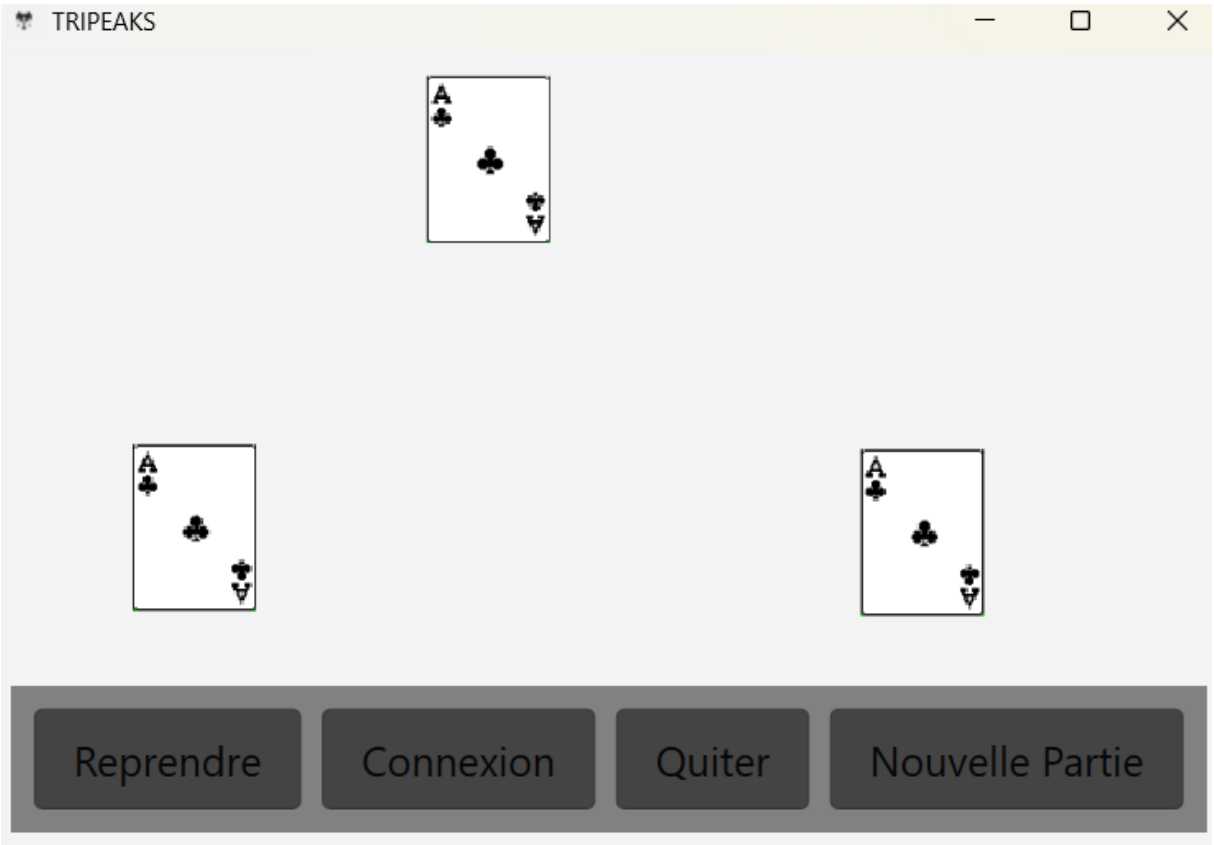
La barre de menu

C'est celle qui présente toutes les options proposées par le jeu pour le rendre plus compréhensible. Elle sera constituée de MENU qui contiendra les menus recommencer (pour recommencer le jeu à zéro), option (pour pouvoir désactiver/activer la musique et ou les effets sonores), Quitter (pour sortir du jeu), comment jouer (pour expliquer le jeu et les différentes possibilités), A propos (pour présenter le jeu et les développeurs) et/ou de menus sauvegarder et charger (pour la sauvegarde et la récupération de partie) ou retour (pour retourner au menu principal).



Le menu principal

C'est lui qui doit nous accueillir à l'ouverture du jeu. Il doit être assez simple et présenter les options tels que : reprendre pour reprendre une partie, quitter pour sortir du jeu, connexion pour une possible connexion, nouvelle partie pour débuter une partie et/ou des options supplémentaires tel que « option » (pour régler le volume depuis l'interface), comment jouer (pour comprendre le jeu avant de le commencer.).



3. Informations supplémentaires pertinentes

Le jeu pourrait évoluer vers un mode multijoueur en ligne compétitif plu tard ;

Le jeu pourrait offrir un menu principal interactif et contenant plusieurs variantes du jeu et aussi un tutoriel dynamique qui montrerai comment jouer au début d'une nouvelle partie ainsi que d'une option conseil ;

Possibilité d'intégrer des thèmes visuels personnalisables dans le futur ;

Adaptation possible sur appareil mobile pour les versions ultérieures ;

Le jeu respectera les bonnes pratiques de programmation en C++ (modularité, encapsulation, design patterns) ;

Le jeu fera usage de la librairie Qt recommandée pour l'interface graphique multiplateforme ;

Le jeu pourrait évoluer et offrir une prise en charge multilingue dès le lancement, notamment en français et en anglais.