Software Requirements Document: White House Defense

BONAERT Gregory
COLSON Thibaut
ENGELMAN Benjamin
ENGELMAN David
GAVRILOAIA Edouard
GJINI Jurgen
PIERROT Arthur
VAN HAUWAERT Alexandre

Groupe 4

5 mars 2016

Contents

1 Introduction **3**
- 1.1 But du projet . . . 3
- 1.2 Glossaire . . . 4
- 1.3 Historique . . . 6

2 Besoins utilisateur: Fonctionnels **7**
- 2.1 Connexion . . . 7
  - 2.1.1 Se connecter . . . 7
  - 2.1.2 S'enregistrer . . . 7
- 2.2 Menu Principal . . . 8
  - 2.2.1 Lancer une partie . . . 8
  - 2.2.2 Gérer son profil . . . 9
  - 2.2.3 Consulter un profil . . . 9
  - 2.2.4 Consulter le classement . . . 9
  - 2.2.5 Gérer une liste d'amis . . . 10
  - 2.2.6 Visionner une partie en cours . . . 10
- 2.3 En partie . . . 11
  - 2.3.1 Achat des tours . . . 11
  - 2.3.2 Placer des tours . . . 11
  - 2.3.3 Vente des tours . . . 12
  - 2.3.4 Améliorer des tours . . . 12
  - 2.3.5 Lancer le "Ad spell" . . . 12
  - 2.3.6 Lancer le "Nuke spell" . . . 12
  - 2.3.7 Lancer le "Freeze spell" . . . 13
  - 2.3.8 Lancer le "Airstrike" . . . 13
  - 2.3.9 Lancer le "Team heal" . . . 13
  - 2.3.10 Envoyer des messages aux autres joueurs . . . 13

3 Besoins utilisateur: Non fonctionnels **13**

4 Besoin système: Fonctionnels **14**
- 4.1 Connexion . . . 14
  - 4.1.1 Vérifier les infos . . . 14
- 4.2 Gestion des comptes . . . 14
  - 4.2.1 Création d'un compte . . . 14
  - 4.2.2 Suppression d'un compte . . . 15
- 4.3 Gestion des profils . . . 15
  - 4.3.1 Affichage des profils . . . 15
  - 4.3.2 Modifier son profil . . . 15
  - 4.3.3 Consulter les classements . . . 15
- 4.4 Créer une partie . . . 16
  - 4.4.1 Créer une file d'attente . . . 16
  - 4.4.2 Ajouter des joueurs dans une file d'attente . . . 16
- 4.5 Gestion d'une partie: . . . 17
  - 4.5.1 Gestion des PNJ . . . 17
  - 4.5.2 Faire disparaître le PNJ . . . 17
  - 4.5.3 Faire apparaître les PNJ . . . 17
  - 4.5.4 Gestion de l'argent . . . 19
  - 4.5.5 Gérer l'argent . . . 19
  - 4.5.6 Gestion du score . . . 20
  - 4.5.7 Gestion de la difficulté . . . 20
  - 4.5.8 Choix du vainqueur de la partie . . . 20

5 Besoins système: Non fonctionnels **20**
5.1 Système d'exploitation . . . 20
5.2 Réseau . . . 20
5.3 Disponibilité . . . 21
5.4 Performances . . . 21
5.5 Capacité . . . 21
5.6 Sécurité . . . 21

6 Design et fonctionnement du Système **21**
6.1 Design du système . . . 21
6.1.1 Explications complémentaires . . . 26
6.1.2 Design de la partie Server . . . 26
6.1.3 Design de la partie Client . . . 26
6.1.4 Base de donnée . . . 27
6.1.5 Design des interfaces . . . 27
6.1.6 Diagramme complet du système . . . 28
6.2 Fonctionnement du système . . . 30
6.2.1 Inscription . . . 30
6.2.2 Connexion . . . 31
6.2.3 Matchmaking: Client . . . 32
6.2.4 Matchmaking: Serveur . . . 32
6.2.5 Boucle du jeu: Client . . . 33
6.2.6 Boucle du jeu: Serveur . . . 34
6.2.7 Diagramme d'activité : Login . . . 35

7 Index des termes utilisés **36**

1 Introduction

1.1 But du projet

L'objectif de ce projet est de réaliser un jeu de type "Tower Defense". L'objectif d'un jeu de ce type est simple : Un joueur doit défendre sa base face à des attaques successives de vagues de personnages non-joueur (PNJ). Dans le cadre de ce projet, il nous est demandé de réaliser une version multi-joueurs. Plusieurs joueurs s'affronteront donc en même temps sur la même carte.

Le thème de notre jeu est Donald Trump, président des États-Unis. Notre objectif sera donc de rappeler au travers du jeu différents éléments qui ont marqué sa campagne dans une optique humoristique. Les PNJ seront donc représentés notamment par des mexicains, des musulmans et des communistes. La base d'un joueur sera représentée par la maison blanche.

Pour pouvoir jouer, l'utilisateur doit se connecter à l'aide d'un pseudonyme. Il faut donc s'être inscrit au préalable afin de pouvoir jouer à *White House Defense*.

Trois modes de jeu sont disponibles : classique, par équipe et contre la montre. Un mode "extra" supporter est également présent. Dans ce mode, l'utilisateur observe une partie tout en faisant profiter un joueur de plus d'argent que les autres. Il a aussi la possibilité d'influencer les parties des joueurs adverses en leur envoyant des pubs qui gênent leurs écrans de jeu.

En dehors d'une partie, un joueur a la possibilité de personnaliser son profil, de consulter le profil d'un autre joueur et le classement. Ce dernier est classé par ordre décroissant du nombre de victoires, c'est à dire que le premier a plus de victoires que le deuxième.

Au cours d'une partie, un joueur doit placer des tours de manière stratégique sur la carte pour vaincre les PNJ attaquant sa base. Pour acquérir des tours, le joueur a à sa disposition de l'argent qu'il dépense dans une boutique virtuelle. Tous les modes de jeu se jouent à quatre joueurs et la carte est découpée en 4 parties symétriquement identiques : Nord, Sud, Est, Ouest.

1.2 Glossaire

PNJ : personnage non-joueur.

Vague : ensemble de PNJ.

Matchmaking : système qui met en relation des joueurs qui souhaitent jouer au même moment, de façon à créer une partie et lancer un jeu.

Pseudo : nom d'emprunt utilisé par les utilisateurs.

Frame : intervalle de temps pendant lequel l'état du jeu est mis à jour et une nouvelle image représentant l'état du jeu est dessinée.

1.3 Historique

Numéro de version	Nom	Modifications	Date
0.1	David Engelman/Benjamin Engelman	Besoins utilisateur	3/12/16
0.2	Gregory Bonaert	Besoins systeme	4/12/16
0.3	David Engelman	Gestion pnj + gestion argent	4/12/16
0.4	Benjamin Engelman	Gestion fin de partie et score	8/12/16
0.5	Benjamin Engelman/Edouard Gavriloaia	Besoin utilisateur: Non fonctionnels	11/12/16
0.6	David Engelman	But du projet + détail besoin utilisateur	11/12/16
0.7	Benjamin Engelman/David Engelman	Ajout use case diagram : utilisateur + système (connexion, matchmaking)	17/12/16
0.8	Jurgen Gjini	Ajout use case diagram : besoin systeme	17/12/16
0.9	Gregory Bonaert	Ajout du glossaire	18/12/16
0.10	Gregory Bonaert	Gestion de la difficulté + Classements + Revente des tours	18/12/16
0.11	David Engelman	Inclusion diagrammes séquences	18/12/16
0.12	Benjamin Engelman	Commentaires diagrammes séquences	18/12/16
0.13	David Engelman	Ajouts commentaires diagrammes séquences	18/12/16
0.14	David Engelman/Benjamin Engelman	Design système	18/12/16
0.15	Arthur Pierrot	Diagramme d'activité matchmaking	19/12/16
0.16	Edouard Gavriloaia/Alexandre van Hauwaert	Besoin système: non-fonctionnels	19/12/16
1.1	Alexandre van Hauwaert	Mise à jour après feedback	20/02/17
1.2	Edouard Gavriloaia	Ajout de l'index	28/02/17
1.3	Edouard Gavriloaia	Documentation et ajustement des use-cases après feedback	02/03/17
1.4	Benjamin Engelaman/David Engelman	Ajout section 6.1, 6.1.2, 6.1.3	04/03/17
1.5	Benjamin Engelman/Alexandre van Hauwaert	Diagramme de classe complet	04/03/17
1.6	Benjamin Engelman	Diagramme d'activité: Communication client/server	04/03/17
1.7	Benjamin Engelman/David Engelman	Ajout section UI et corrections	29/03/17

2 Besoins utilisateur: Fonctionnels

2.1 Connexion

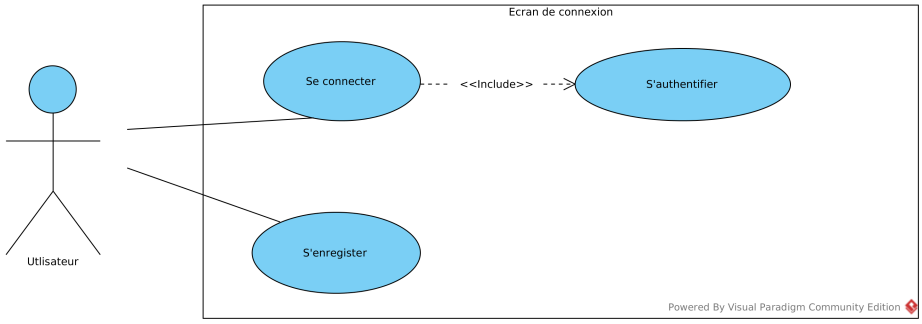


Figure 1: Use case : Connexion

Lors du lancement du programme client, une fenêtre de connexion s'ouvre pour permettre à l'utilisateur de se connecter à son compte. Si celui-ci ne possède pas encore de compte, il a la possibilité de s'en créer un.

2.1.1 Se connecter

- **Acteur :** Client.
- **Relations avec d'autres cas d'utilisation :** S'authentifier.
- **Pré-conditions :** Le serveur doit être en ligne et le client doit posséder au préalable un compte utilisateur reconnu par le serveur du jeu.
- **Post-conditions :** Le client est connecté à son compte utilisateur et accède au menu principal une fois authentifié.
- **Cas général :** Le client se connecte au serveur du jeu pour accéder à son compte utilisateur.
- **Cas exceptionnels :** Les informations entrées ne correspondent pas à un compte préalablement enregistré. Le client est invité à réintroduire ses identifiants tant que cela se produit.

2.1.2 S'enregistrer

- **Acteur :** Client.
- **Relations avec d'autres cas d'utilisation :** Néant.
- **Pré-conditions :** Le serveur doit être en ligne.
- **Post-conditions :** Le client possède désormais un compte utilisateur et il accède à la fenêtre de connexion.
- **Cas général :** Le client envoie une requête au serveur afin de se créer un compte utilisateur reconnu par les serveurs du jeu.
- **Cas exceptionnels :** Si le nom d'utilisateur choisi est déjà sélectionné ou que les informations entrées ne correspondent pas aux contraintes imposées, un message d'erreur est affiché et l'utilisateur à l'occasion de réintroduire sa requête d'enregistrement tant que cela se produit.

2.2 Menu Principal

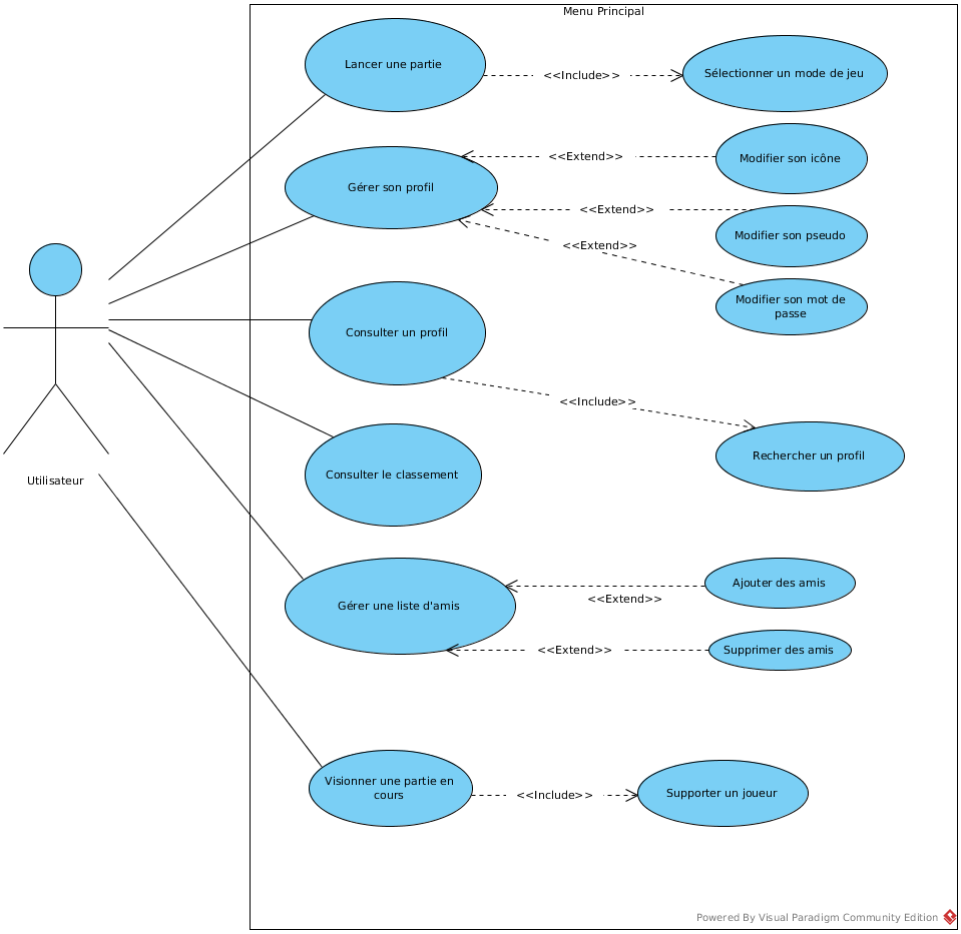


Figure 2: Use case : Menu Principal

2.2.1 Lancer une partie

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Sélectionner un mode de jeu.
- **Pré-conditions :** L'utilisateur doit au préalable avoir sélectionner l'option "Lancer une partie" dans le menu principal du jeu.
- **Post-conditions :** Une fois un mode de jeu sélectionné, le joueur est placé dans une file d'attente correspondant au mode de jeu sélectionné en attendant qu'une partie soit trouvée. Lorsque il y a assez de joueurs pour lancer une partie, ils sont alors redirigés par le programme pour rejoindre la partie dédiée et ils sont retirés de la file d'attente par la même occasion.
- **Cas général :** L'utilisateur lance une partie et sélectionne le mode de jeu auquel il souhaiterait participer. Il à la possibilité de choisir parmi plusieurs modes de jeu:
  1. Mode classique
  2. Mode contre la montre
  3. Mode par équipe
- **Cas exceptionnels :** Néant.

2.2.2 Gérer son profil

Le joueur a la possibilité de modifier son nom d'utilisateur, son mot de passe et sa photo de profil.

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Modifier son icône / Modifier son pseudo / Modifier son mot de passe.
- **Pré-conditions :** L'utilisateur doit-être connecté à son compte utilisateur.
- **Post-conditions :** Le profil du joueur a été mis à jour.
- **Cas général :** L'utilisateur a la possibilité de personnaliser son profil de différentes façons:
  - Modifier son pseudo : le joueur est invité à saisir un nouveau pseudo. Si celui-ci est disponible son pseudo est modifié.
  - Si le joueur utilise l'interface graphique du jeu, celui-ci peut modifier sa photo de profil.
- **Cas exceptionnels :** Lorsque l'utilisateur entre un pseudo indisponible ou un mot de passe incorrect, le programme le lui signale et le joueur est invité à recommencer tant que cela se produit.

2.2.3 Consulter un profil

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Rechercher un profil.
- **Pré-conditions :** L'utilisateur doit-être connecté à son compte utilisateur et se trouver dans le menu principal du jeu.
- **Post-conditions :** L'utilisateur accède au profil de l'utilisateur pour lequel la requête de consultation du profil a été faite (en mode lecture seulement).
- **Cas général :** Le joueur peut consulter un profil en envoyant une requête au serveur afin d'obtenir des informations sur le profil d'un utilisateur en le recherchant grâce à son pseudo.
- **Cas exceptionnels :** L'utilisateur pour lequel la requête de consultation du profil a été faite n'existe pas. Le programme préviens alors l'utilisateur avec un message d'erreur.

2.2.4 Consulter le classement

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Néant.
- **Pré-conditions :** L'utilisateur doit-être connecté à son compte utilisateur et se trouver dans le menu principal du jeu.
- **Post-conditions :** Le classement est affiché et le joueur peut alors le consulter.
- **Cas général :** Le joueur peut consulter un classement unique dans lequel les joueurs sont classés selon leur nombre de victoires, tous modes de jeu confondus.
- **Cas exceptionnels :** Néant.

2.2.5 Gérer une liste d'amis

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Ajouter des amis / Supprimer des amis.
- **Pré-conditions :** L'utilisateur doit-être connecté à son compte utilisateur et se trouver dans le menu principal du jeu..
- **Post-conditions :** L'utilisateur accède soit à l'option de suppression d'un ami soit à l'option d'ajout d'un ami selon son choix.
- **Cas général :** L'utilisateur accède à sa liste d'amis via le menu du jeu. Des amis sont des autres joueurs ayant été ajoutés à cette liste. Il est donc la possibilité d'ajouter et de supprimer des amis selon leur nom d'utilisateur.
- **Cas exceptionnels :** Lorsque l'utilisateur entre un pseudo indisponible, le programme le lui signale avec un message d'erreur.

2.2.6 Visionner une partie en cours

Lorsqu'un supporter demande de se connecter à une partie, le serveur cherche si une ou plusieurs parties sont en cours, si c'est le cas, le supporter peut se connecter à l'une d'elle, sinon, le serveur envoie un message lui signalant l'absence de partie en cours.

- **Acteur :** Utilisateur.
- **Relations avec d'autres cas d'utilisation :** Supporter un joueur.
- **Pré-conditions :** L'utilisateur doit-être connecté à son compte utilisateur et au moins une partie doit être en cours.
- **Post-conditions :** L'utilisateur accède à la partie en cours sélectionnée en tant que spectateur et le joueur supporté reçoit un gain d'argent bonus par seconde.
- **Cas général :** L'utilisateur souhaite accéder à une partie en cours en tant que spectateur et sélectionne un joueur à supporter qui bénéficiera d'un bonus en jeu.
- **Cas exceptionnels :** Il n'y a pas de partie en cours et il n'est alors pas possible de rejoindre une partie en tant que spectateur.

2.3 En partie

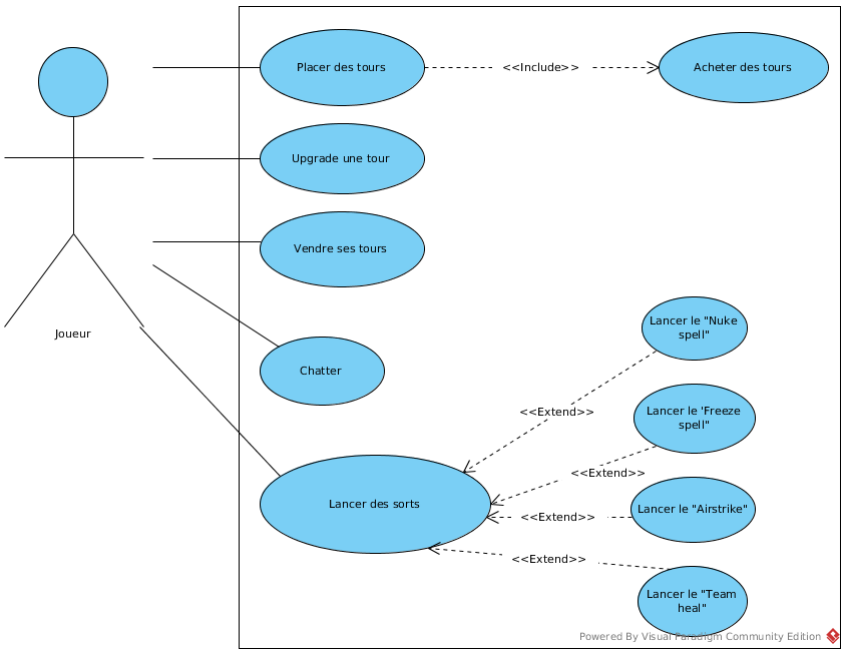


Figure 3: Use case : En partie

2.3.1 Achat des tours

Si le joueur possède l'argent nécessaire, celui-ci peut acheter différents types de tours et doit les placer sur la carte. L'argent dépensé lui est alors décompté. Il ne peut pas garder de tours en réserve pour les placer par-après.

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Placer des tours.
- **Pré-conditions :** Le joueur doit posséder assez d'argent et des positions disponibles à cet effet sur sa section de carte.
- **Post-conditions :** Le joueur se voit retirer la somme correspondant au prix de la tour.
- **Cas général :** Le joueur souhaite acheter une tour.
- **Cas exceptionnels :** Le joueur ne possède pas assez d'argent, l'action est donc refusée.

2.3.2 Placer des tours

Afin de placer des tours, le joueur sélectionne un emplacement de la carte approprié à la construction d'une tour, il peut alors choisir le type de tour qu'il souhaite construire.

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Acheter des tours.
- **Pré-conditions :** Le joueur doit avoir cliqué sur l'emplacement futur de la tour.
- **Post-conditions :** Le joueur voit sa tour apparaître sur la carte.
- **Cas général :** Le joueur souhaite placer une tour pour améliorer sa défense.
- **Cas exceptionnels :** Le joueur clique sur un endroit inadéquat sur la carte ou entre des coordonnées invalides, l'action est donc refusée.

2.3.3 Vente des tours

Si le joueur possède des tours sur la carte, il peut revendre des tours déjà placées et la somme correspondante qui est un pourcentage du prix d'origine de la tour lui est alors reversé.

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Néant.
- **Pré-conditions :** Le joueur doit posséder au moins une tour sur la carte.
- **Post-conditions :** La tour disparaît de la carte et le joueur se voit reversé un pourcentage du prix de la tour.
- **Cas général :** Le joueur souhaite revendre une tour pour gagner de l'argent ou car il n'est pas satisfait du travail de celle-ci.
- **Cas exceptionnels :** Le joueur sélectionne des coordonnées où aucune tour n'est présente. L'action est donc refusée.

2.3.4 Améliorer des tours

Si le joueur possède des tours sur la carte, il peut améliorer des tours déjà placées, ce qui renforce leurs capacités. Le niveau maximum d'une tour est 5 (4 améliorations).

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Néant.
- **Pré-conditions :** Le joueur doit posséder au moins une tour sur la carte.
- **Post-conditions :** La tour augmente d'un niveau. Ses dégâts, sa portée, et son prix sont augmentés d'un pourcentage respectif.
- **Cas général :** Le joueur souhaite améliorer les performances d'une tour
- **Cas exceptionnels :** Le joueur souhaite améliorer une tour dont le niveau maximum est atteint. Dans ce cas, l'action est refusée.

2.3.5 Lancer le "Ad spell"

Un supporter peut utiliser le "Ad spell" faisant apparaître des pubs chez tous les joueurs excepté celui qu'il supporte.

- **Acteur :** Supporter.
- **Relations avec d'autres cas d'utilisation :** Lancer des sorts.
- **Pré-conditions :** Néant.
- **Post-conditions :** Un pop-up est affiché chez les joueurs sauf chez le joueur supporté par le supporter qui a lancé le sort.
- **Cas général :** Le supporter souhaite déstabiliser les autres joueurs pour donner l'avantage à celui qu'il supporte.
- **Cas exceptionnels :** Néant.

2.3.6 Lancer le "Nuke spell"

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Lancer des sorts.
- **Pré-conditions :** Une vague doit être entrain d'avancer et le sort doit être disponible.
- **Post-conditions :** Tous les PNJ de la vague sont morts.
- **Cas général :** Le joueur souhaite éliminer tous les PNJ de la vague en un coup. L'utilisation de ce sort est limitée à un usage unique pour chaque joueur de la partie.
- **Cas exceptionnels :** Néant.

2.3.7 Lancer le "Freeze spell"

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Lancer des sorts.
- **Pré-conditions :** Une vague doit être en train d'avancer et le sort doit être disponible.
- **Post-conditions :** Les PNJ de la vague sont à l'arrêt.
- **Cas général :** Le joueur gèle tous les PNJ de la vague pendant un court instant pour les empêcher d'avancer. L'utilisation de ce sort est limitée à un usage unique pour chaque joueur de la partie.
- **Cas exceptionnels :** Néant.

2.3.8 Lancer le "Airstrike"

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Lancer des sorts.
- **Pré-conditions :** Le sort doit être disponible.
- **Post-conditions :** La base du joueur touché a perdu 20 points de vie.
- **Cas général :** Le joueur souhaite infliger des dégâts à une base ennemie. L'utilisation de ce sort est limitée à un usage unique pour chaque joueur de la partie.
- **Cas exceptionnels :** Le joueur ne peut pas envoyer un airstrike sur lui-même, ni sur son coéquipier dans le cas d'un match en équipe.

2.3.9 Lancer le "Team heal"

- **Acteur :** Joueur.
- **Relations avec d'autres cas d'utilisation :** Lancer des sorts.
- **Pré-conditions :** Le sort doit être disponible et la partie doit être une partie en équipe.
- **Post-conditions :** Le coéquipier qui fait soigner récupère 20 points de vie .
- **Cas général :** Le joueur souhaite soigner son coéquipier pour l'aider . L'utilisation de ce sort est limitée à un usage unique pour chaque joueur de la partie.
- **Cas exceptionnels :** Néant.

2.3.10 Envoyer des messages aux autres joueurs

- **Acteur :** Joueur / Supporter.
- **Relations avec d'autres cas d'utilisation :** Néant.
- **Pré-conditions :** Néant.
- **Post-conditions :** Un message est envoyé dans le chat.
- **Cas général :** Le joueur / supporter souhaite communiquer avec les autres personnes présentent dans la partie. Le joueur / supporter peut écrire lui-même un message ou bien choisir parmi messages pré-définis.
- **Cas exceptionnels :** Le joueur / supporter envoie une insulte, celle-ci est alors masquée.

3 Besoins utilisateur: Non fonctionnels

Le jeu doit proposer une interface agréable et facile d'utilisation. Les parties doivent être rapides. En effet, le temps d'une partie ne devrait pas dépasser 10 minutes. De plus quand le joueur recherche une partie, celui ci ne devrait pas devoir attendre plus de 5 minutes dans la file d'attente (sauf cas exceptionnel). Bien entendu, l'accès au jeu est gratuit.

4 Besoin système: Fonctionnels

4.1 Connexion

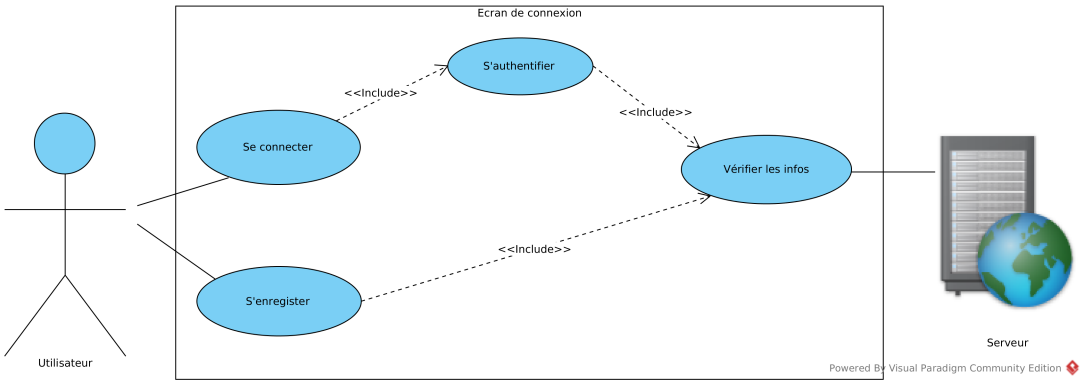


Figure 4: Use case : Connexion system

4.1.1 Vérifier les infos

Au démarrage du jeu, l'utilisateur peut s'authentifier via un formulaire qui lui demande son nom d'utilisateur et son mot de passe. Les données entrées sont envoyées au serveur pour que celui-ci effectue des vérifications.

- **Acteur :** Serveur
- **Relations avec d'autres cas d'utilisation :** S'authentifier / S'enregistrer.
- **Pré-conditions :** Le serveur doit être en ligne.
- **Post-conditions :**
  - Si l'utilisateur s'est authentifié correctement, il est alors redirigé vers le menu principal du jeu.
  - Si l'utilisateur s'est enregistré correctement, il est alors redirigé vers la fenêtre de connexion.
- **Cas général :** Le serveur traite les requêtes de connexions des différents clients et vérifie leur validité dans la base de données.
- **Cas exceptionnels :** Si l'utilisateur introduit des données erronées, un message d'erreur renvoyé par le serveur est affiché.

4.2 Gestion des comptes

4.2.1 Création d'un compte

Au démarrage du jeu, le joueur peut créer un nouveau compte. Un formulaire sera affiché qui lui demande un nom d'utilisateur et un mot de passe.

Le client envoie le pseudo désiré et le mot de passe au serveur. Si le pseudo n'est pas disponible, le serveur renvoie un code d'erreur, qui permet au client d'afficher un message d'erreur à l'utilisateur.

Si le pseudo est disponible et correct, le serveur crée le compte et l'indique au client, sinon le programme renverra un message d'erreur. Si tout s'est bien passé, le joueur peut maintenant se connecter à son compte comme expliqué à la section 3.1.

4.2.2 Suppression d'un compte

Sur son profil, le joueur a la possibilité de supprimer son compte. Avant d'envoyer un message au serveur pour que celui ci supprime le compte, le client affiche un message de confirmation pour être certain que l'utilisateur veut bien supprimer son compte. Si l'utilisateur confirme, un message est envoyé au serveur pour lui demander de supprimer ce joueur de la base de données. Le compte est alors supprimé. Le joueur est déconnecté.

4.3 Gestion des profils

4.3.1 Affichage des profils

Le joueur peut voir les informations sur des profils (le sien ou celui d'autres utilisateurs). Le client envoie alors une requête au serveur pour que celui-ci lui fournisse les informations du profil recherché par l'utilisateur. Le serveur renvoie les informations suivantes:

- Son nom d'utilisateur/pseudo
- Son icône
- Le nombre de PNJ tués
- Le nombre de victoires

Si l'utilisateur veut afficher le profil d'un joueur qui n'existe pas, le serveur renvoie un message d'erreur qui est affiché par le client.

4.3.2 Modifier son profil

Le joueur a la possibilité de modifier certaines informations de son profil:

- Son icône
- Son nom d'utilisateur/pseudo
- Son mot de passe

Pour vérifier la disponibilité du pseudo, le client envoie le pseudo désiré au serveur, qui lui indique si le nom est disponible. Si il n'est pas disponible, le serveur renvoie un message d'erreur que le client affiche à l'utilisateur. Celui ci est alors invité à proposer un autre pseudo.

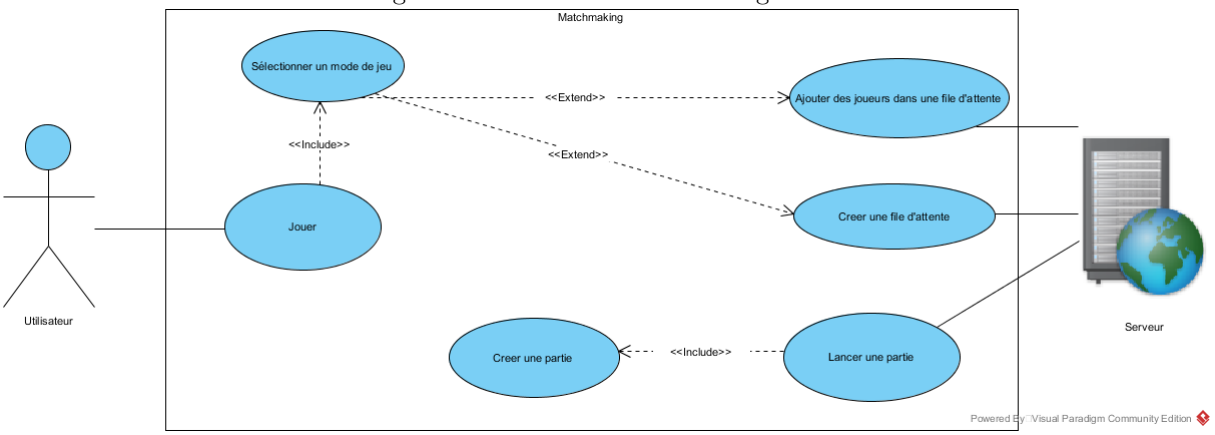
4.3.3 Consulter les classements

Le joueur peut voir le classement de meilleurs joueurs. Le client envoie alors une requête au serveur, qui lui renvoie les informations suivantes:

- Le pseudo des joueurs
- Leur nombre de victoires

4.4 Créer une partie

Figure 5: Use case : Matchmaking



Lorsqu'un joueur demande de se connecter à une partie, le serveur sera responsable du matchmaking.

4.4.1 Créer une file d'attente

- **Acteur :** Serveur.
- **Relations avec d'autres cas d'utilisation :** Sélectionner un mode de jeu.
- **Pré-conditions :** Une file d'attente n'est créée seulement si aucune autre file d'attente n'existe déjà pour le mode de jeu sélectionné.
- **Post-conditions :** Lorsqu'une file d'attente est créée, le premier joueur a avoir fait la requête pour rejoindre une partie ainsi que les prochains joueurs seront placés dans cette file (par mode de jeu respectif).
- **Cas général :** Le serveur traite les requêtes de connexion à une partie via le matchmaking.
- **Cas exceptionnels :** Néant.

4.4.2 Ajouter des joueurs dans une file d'attente

- **Acteur :** Serveur.
- **Relations avec d'autres cas d'utilisation :** Sélectionner un mode de jeu.
- **Pré-conditions :** Il doit déjà y avoir une file d'attente pour le mode de jeu sélectionné.
- **Post-conditions :** Le joueur est ajouté à la file d'attente et attend le début de la partie.
- **Cas général :** Le serveur traite les requêtes de connexion à une partie via le matchmaking.
- **Cas exceptionnels :** Il est possible qu'un joueur se déconnecte en file d'attente, sa place dans la file est alors libérée.

4.5 Gestion d'une partie:

4.5.1 Gestion des PNJ

Apparitions des PNJ

A chaque début de vague, le serveur fait apparaître les PNJ au centre de la carte. Les vagues de tous les joueurs sont composés du même nombre de PNJ. Tous les joueurs ont donc affaire à des vagues de niveau identique.

Vie des PNJ

Les PNJ possèdent un total de points de vie initialisé à une valeur fixée. Chaque fois qu'un PNJ est touché par une tourelle, celui-ci se voit infliger des dégâts réduisant son nombre de points de vie. Le nombre de dégâts qui lui sont infligés dépend du type de la tour l'ayant touché. Une fois qu'un PNJ a perdu tous ses points de vie, il disparaît de la carte.

Capacités des PNJ

Un PNJ peut tout simplement avancer sur la carte. Son objectif est de se rendre dans la base du joueur. Un PNJ ne possède pas de moyen de faire des dégâts.

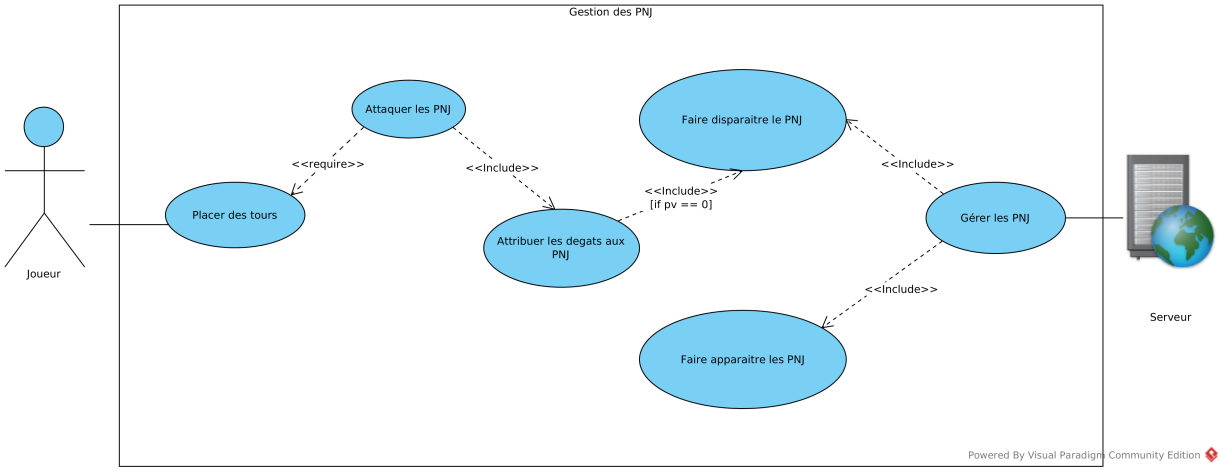


Figure 6: Use case : Gestion des PNJ

4.5.2 Faire disparaître le PNJ

- **Acteur :** Serveur.
- **Relations avec d'autres cas d'utilisation :** Gérer les PNJ / Attribuer les dégâts aux PNJ.
- **Pré-conditions :** Pour que le serveur puisse faire disparaître un PNJ, il faut qu'une vague de PNJ soit présente sur la carte et qu'au moins un des PNJ ait un montant de points de vie inférieur ou égal à 0.
- **Post-conditions :** Le PNJ disparaît de la vague dans laquelle il était.
- **Cas général :** Déroulement du jeu et gestion de la partie par le serveur.
- **Cas exceptionnels :** Néant.

4.5.3 Faire apparaître les PNJ

- **Acteur :** Serveur.
- **Relations avec d'autres cas d'utilisation :** Gérer les PNJ.

- **Pré-conditions :** Pour que le serveur puisse faire apparaître une vague de PNJ, il faut soit que ce soit le début de la partie, soit que toutes les vagues de PNJ apparues en dernier lieu soient éliminées (et ce pour tous les quadrants des joueurs encore en vie de la carte).
- **Post-conditions :** Le PNJ apparaît sur la carte au sein d'une vague de PNJ avec son maximum de vie initial.
- **Cas général :** Déroulement du jeu et gestion de la partie par le serveur.
- **Cas exceptionnels :** Quand un joueur est mort, il n'y a plus de vagues de PNJ qui apparaissent dans sa portion de la carte.

4.5.4 Gestion de l'argent

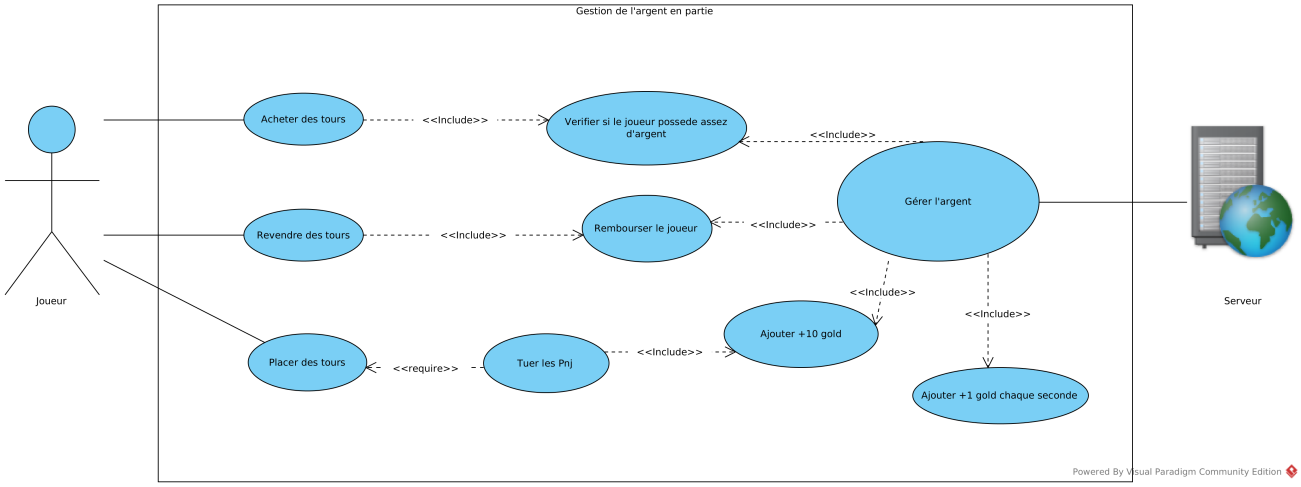


Figure 7: Use case : Gestion de l'argent

Gain d'argent

Chaque joueur possède en début de partie une certaine somme d'argent. Le joueur gagne de l'argent de 2 manières différentes:

1. Toutes les 1 seconde le joueur gagne +1 d'argent. Entre les différentes vagues le joueur ne gagne pas d'argent
2. Le joueur gagne +10 d'argent pour chaque PNJ tué.

Dépenser de l'argent

Le joueur a la capacité de dépenser son argent pour acheter des tours ou faire un upgrade de celles-ci et donc améliorer son système de défense. Ces achats se font au travers d'une boutique.

Une boutique est une fenêtre comportant un listing des différentes tours disponibles à l'achat ainsi que leurs prix. Le joueur peut donc choisir parmi cette liste ce qu'il souhaite acheter. Lors de l'achat d'un objet un message est envoyé au serveur pour que celui puisse décompter la somme dépensée de l'argent possédé par le joueur. Si un joueur tente d'acheter un objet sans posséder l'argent nécessaire un message d'erreur lui est renvoyé par le serveur.

Récupérer de l'argent

Le joueur peut revendre des tours qu'il a déjà placées, récupérant son argent qu'il peut réutiliser pour construire des tours ailleurs. Il récupère une partie de la somme qu'il a dépensé lors de l'achat de la tour.

4.5.5 Gérer l'argent

- **Acteur :** Serveur.
- **Relations avec d'autres cas d'utilisation :**
  - Vérifier si le joueur possède assez d'argent
  - Rembourser le joueur
  - Ajouter +10 gold (PNJ tué)
  - Ajouter +1 gold chaque seconde

- **Pré-conditions :** Pour que le serveur s'occupe de la gestion de l'argent, il faut qu'une partie qui ne soit pas encore finie (càd que le vainqueur n'est pas encore déterminé) soit en cours.
- **Post-conditions :** Le montant d'argent du joueur a été mis à jour.
- **Cas général :** Déroulement du jeu et gestion de la partie par le serveur.
- **Cas exceptionnels :** Lorsque le joueur tente d'effectuer des achats pour lesquels il ne possède pas l'argent nécessaire, le serveur envoie un message pour le signaler au joueur et sa requête d'achat n'est pas effectuée.

4.5.6 Gestion du score

Dans le mode contre la montre

Le but du mode de jeu contre la montre est de tuer le plus de PNJ dans un temps imparti. Le score d'un joueur correspond donc au nombre de PNJ tué.

Dans le mode classique et par équipe

Dans ces deux modes de jeu, le score est également le nombre de PNJ tués mais le score n'intervient pas dans le choix du vainqueur.

4.5.7 Gestion de la difficulté

Au fil du temps, la difficulté de chaque vague augmente progressivement. Le serveur du jeu contrôle cette difficulté de 2 façons différentes:

1. La quantité de PNJ par vague
2. Les types de PNJ présents dans la vague: certains PNJ ont plus de points de vie, les rendant plus résistants aux attaques des tours et augmentant la difficulté de la vague

Pour que la partie ne dure pas trop longtemps, la difficulté augmente relativement rapidement.

4.5.8 Choix du vainqueur de la partie

Lorsqu'une partie se termine, le serveur doit choisir un gagnant. Ce choix dépend du mode de jeu.

Choix du vainqueur dans le mode contre la montre

C'est le joueur avec le plus grand score qui est désigné comme vainqueur, tous les autres joueurs sont alors désignés comme perdants.

Choix du vainqueur dans le mode de jeu classique

Le dernier joueur encore en vie est désigné comme vainqueur, tous les autres sont alors désignés comme perdant.

Choix du vainqueur dans le mode de jeu par équipe

Les joueurs de la dernière équipe encore en vie sont tous désignés comme vainqueur. Les joueurs de toutes les autres équipes sont tous désignés comme perdants.

5 Besoins système: Non fonctionnels

5.1 Système d'exploitation

Le jeu doit être exécutable sur le système d'exploitation «Linux».

5.2 Réseau

Le jeu se joue en réseau, une connexion internet est requise.

5.3 Disponibilité

Pour se connecter, le serveur doit être en ligne. S'il est en ligne, il est accessible sans autres conditions.

5.4 Performances

Les temps de rafraîchissement et les temps de réponses sont de l'ordre des millisecondes.

5.5 Capacité

Le jeu peut gérer quatre joueurs actifs par partie ainsi qu'un nombre indéfini de spectateurs. L'espace de stockage nécessaire pour une partie est de l'ordre des Megabytes.

5.6 Sécurité

Pour participer à une partie, un joueur doit se connecter à son compte. Pour cela, il doit s'authentifier en introduisant un pseudo et un mot de passe corrects.

6 Design et fonctionnement du Système

6.1 Design du système

Le Système est divisé en 9 entités:

1. Connexion et Inscription

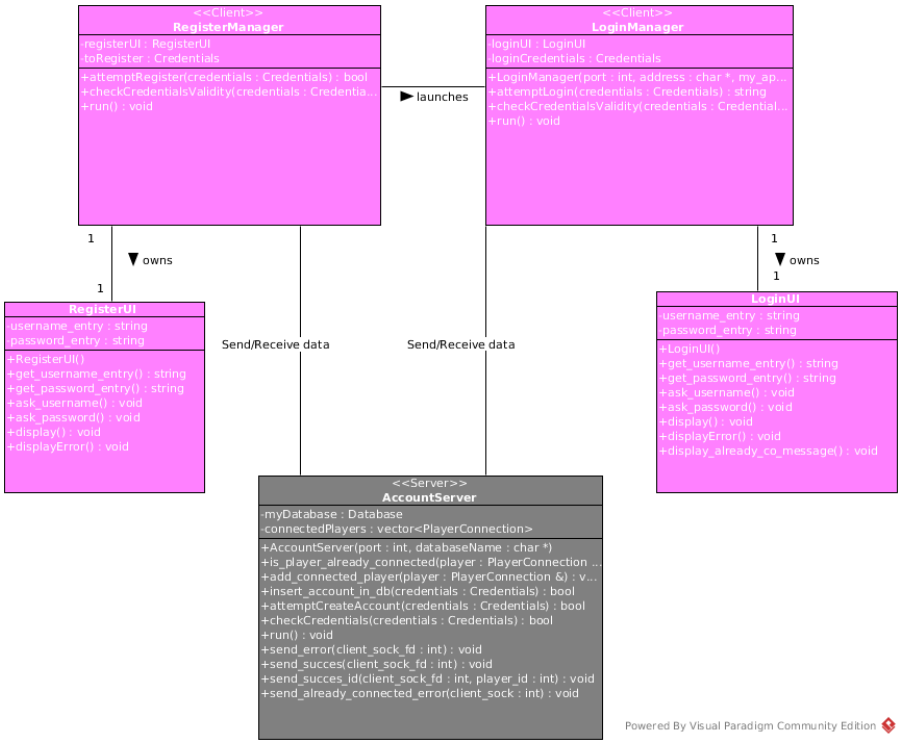


Figure 8: Classes : Connexion et Inscription

2. Gestion des comptes

Figure 9: Classes : Gestion des comptes

3. Menu principal

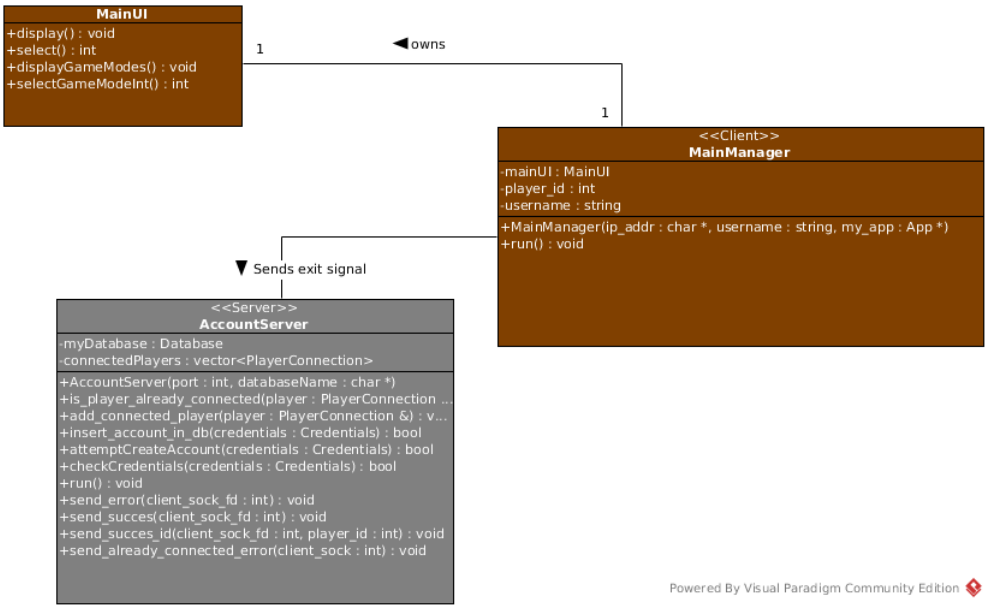


Figure 10: Classes : Menu principal

4. Profil et liste d'amis

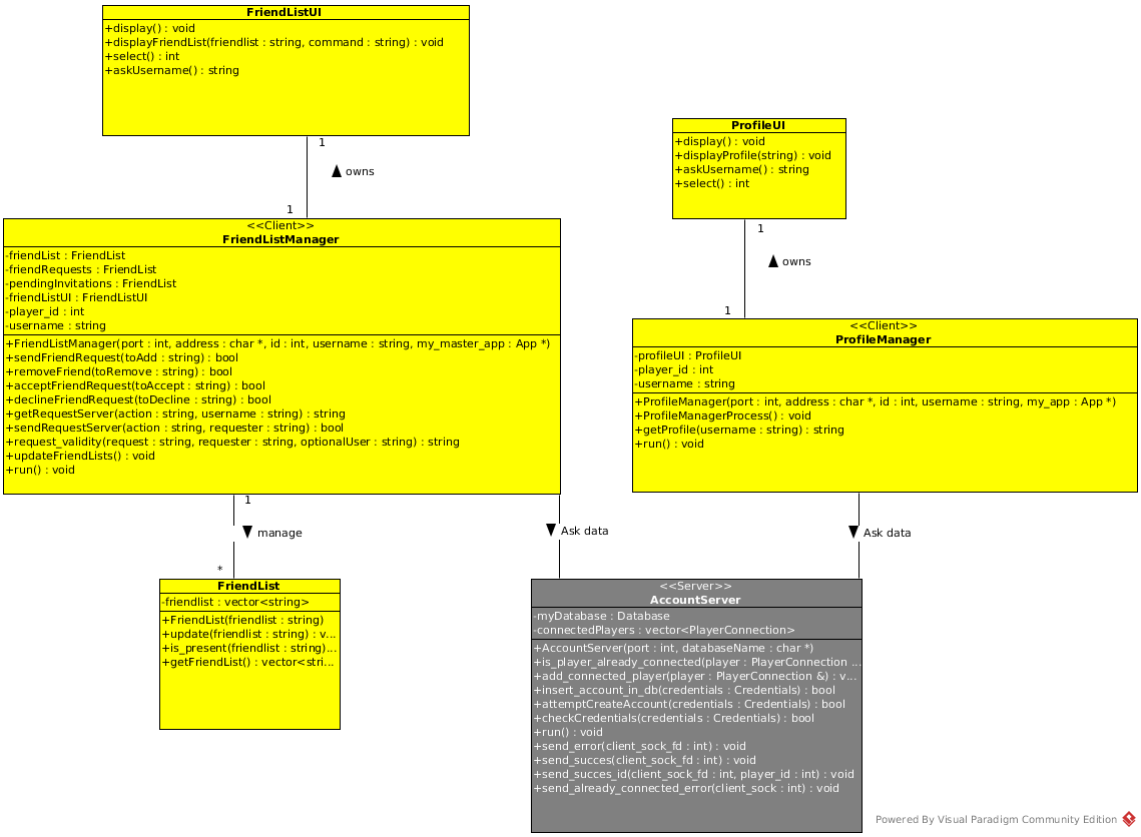


Figure 11: Classes : Profil et liste d'amis

5. Classement

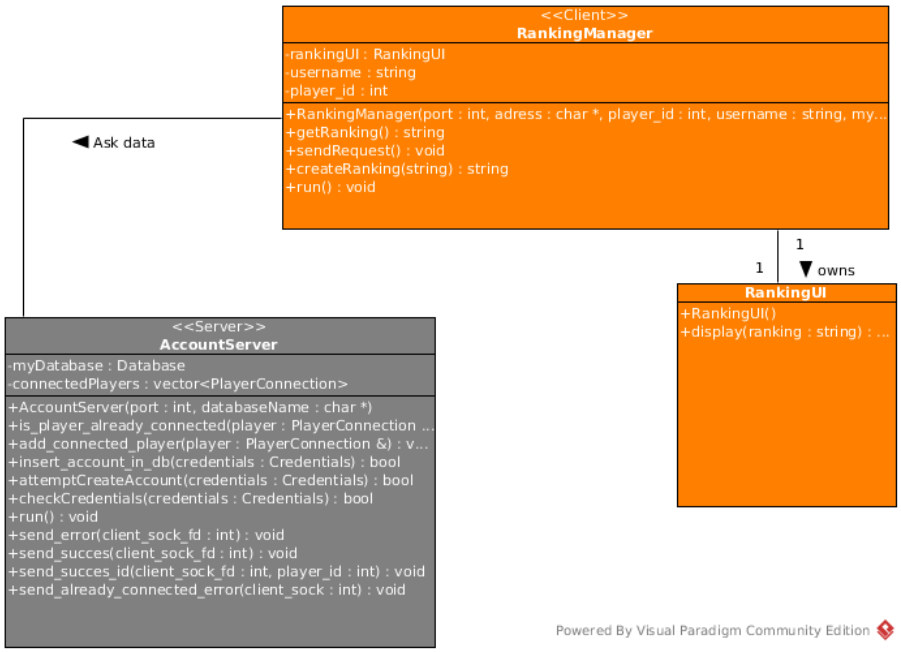


Figure 12: Classes : Classement

6. Matchmaking

Figure 13: Classes : Matchmaking

7. Jeu

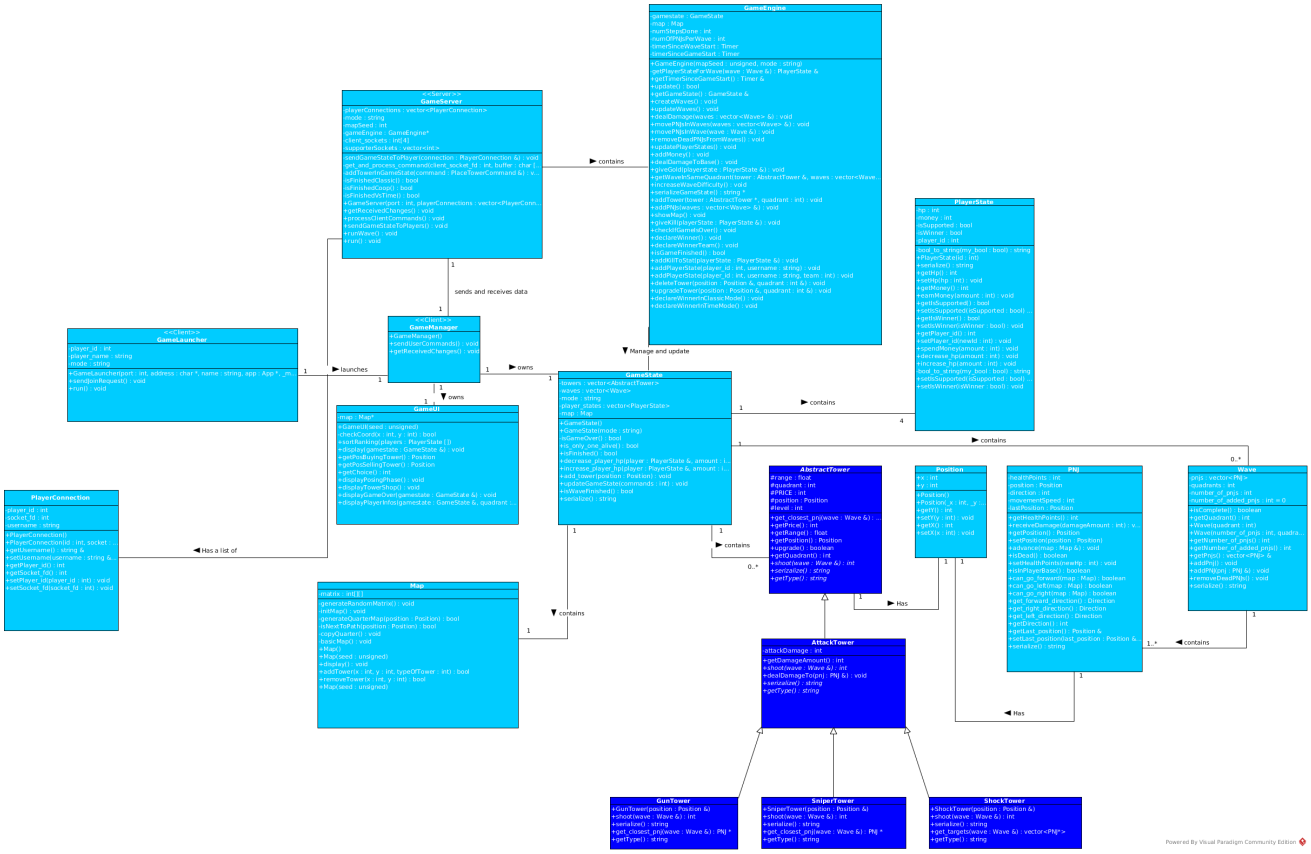


Figure 14: Classes : Jeu

8. Le mode Spectateur

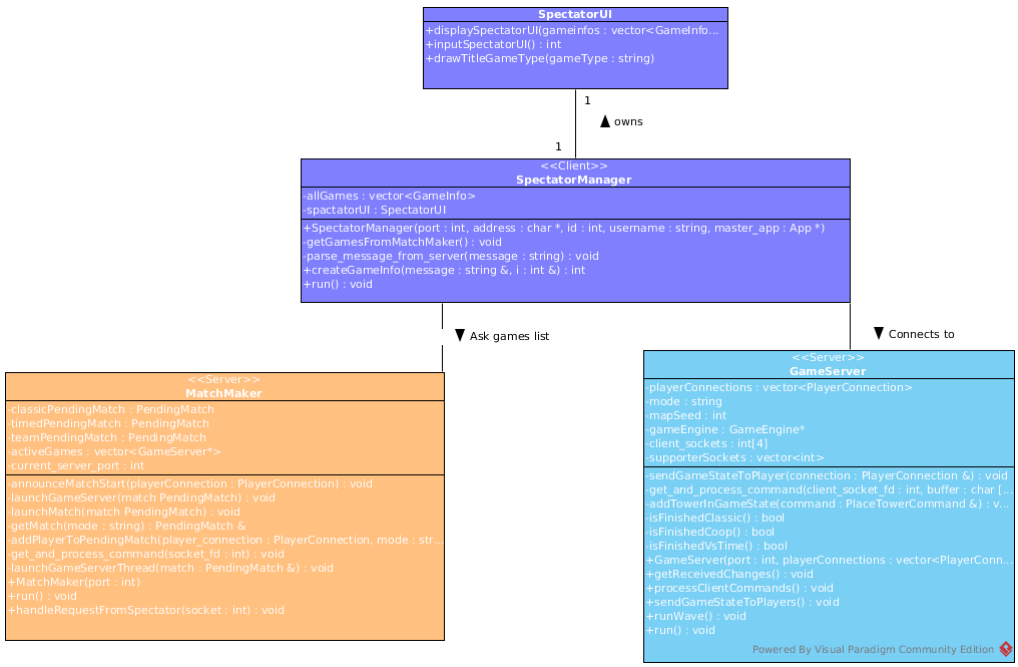


Figure 15: Classes : Spectateur

9. L'écran d'accueil

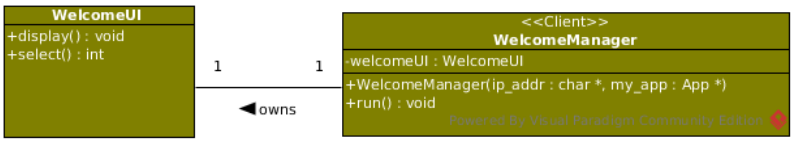


Figure 16: Classes : Ecran d'acceuil

6.1.1 Explications complémentaires

Globalement, chaque entité du système est composée de 3 parties principales:

1. L'interface (UI): Elle sert uniquement à afficher les informations et à recevoir le actions de l'utilisateur qu'elle communique au manager.
2. Le manager: Il gère les requêtes envoyées par l'interface et communique avec le serveur pour échanger des informations.
3. Le serveur: Il se charge d'envoyer les informations demandées par le manager et permet également de mettre en relation plusieurs utilisateurs. En partie, le serveur est le maître du jeu.

En partie, chaque client possède son propre gameState. Le serveur en possède un également, c'est le gameState de référence. Le gameManager est chargé de communiquer avec le GameServer pour mettre à jour les gameState des clients.

6.1.2 Design de la partie Server

Tout d'abord, deux seveurs tournent en **permanence** en parallèle:

- Le AccountServer
- Le MatchMaker

Ensuite, des GameServer sont créés ou détruits au fur et à mesure que des parties sont lancées ou se terminent. Un GameServer se charge de gérer une seule partie, nous avons donc autant de GameServer qui tournent en parallèle que de partie en cours. La partie se déroule sur le server et celui-ci envoi périodiquement l'état du jeu aux clients. Le GameServer gère également l'arrivée des supporter lors d'une partie en cours.

Lorsque qu'une partie est terminée le GameServer se charge d'envoyer les scores des différents joueurs au AccountServer pour que ce dernier puisse mettre à jour les statistiques des joueurs. Ensuite le GameServer signale au MatchMaker que la partie est terminée.

Le AccountServer est responsable de toutes les opérations nécéssitants un accès à la base de donnée du jeu (ex: connexions, profils, friendlist,...).

Le Matchmaker est lui responsable de créer des files d'attentes pour les différents modes de jeu et de les remplir au fur et à mesure que de nouveaux joueurs souhaient rejoindre une partie. C'est aussi lui qui se charge de fournir une liste des parties en cours aux joueurs voulant supporter un autre joueur actuellement en partie.

6.1.3 Design de la partie Client

Les classes faisant office de "clients" sont toutes les classes communiquant avec un des serveurs présentés ci-dessus. Soit:

- LoginManager
  - Communique avec : AccountServer

  - Rôle : Gèrer l'authentfication d'un joueur.

- RegisterManager
  - Communique avec : AccountServer
  - Rôle : Gèrer l'enregistrement d'un nouveau compte utilisateur.

- MainManager
  - Communique avec : AccountServer
  - Rôle : Menu Principal du jeu, il se charge de créer d'autres Manager à la demande de l'utilisateur et de signaler au AccountServer quand un joueur quitte le jeu.

- ProfilManager
  - Communique avec : AccountServer
  - Rôle : Gèrer l'affichage de son propre profil et permet également de rechercher le profil d'un autre joueur.

- RankingManager
  - Communique avec : AccountServer
  - Rôle : Affichage du classement (les joueurs sont classé selon leur nombre de victoires).

- SpectatorManager
  - Communique avec : MatchMaker et GameServer
  - Rôle : Fournir une interface à un utilisateur souhaitant supporter quelqu'un. Une liste des parties en cours est affichée.

- GameLauncher
  - Communique avec : MatchMaker
  - Rôle : Placer un utilisateur dans la queue pour une partie et lancer un GameManager quand la partie démarre.

- GameManager
  - Communique avec : GameServer
  - Rôle : Recevoir les données du GameServer et les traduire pour mettre à jour l'état du jeu. Il offre également une interface au joueur pour placer / supprimer des tours et afficher l'état de la partie.

6.1.4 Base de donnée

La base de donnée utilise le moteur de base de données relationelle sqlite3 accessible par le langage SQL. Celle-çi permet de conserver les informations liées aux comptes et à la liste d'amis.

La base de donnée est composé de 4 tables :

- Accounts
- FriendList
- FriendRequests
- PendingInvitation

6.1.5 Design des interfaces

Le jeu White House Defense est jouable via 2 interfaces : En console et via une interface graphique. La structures du code a donc été pensée de manière à faciliter la coexistence de celles-ci. L'interface graphique à été réalisé en utilisant Qt. Pour des raisons techniques certaines actions in-game ne sont pas disponible en console.

Schéma simplifié du design

Voici la structure utilisée pour les interfaces du menu principal. La structure de toutes les autres "entités" du système décrites précédemment est totalement identique.

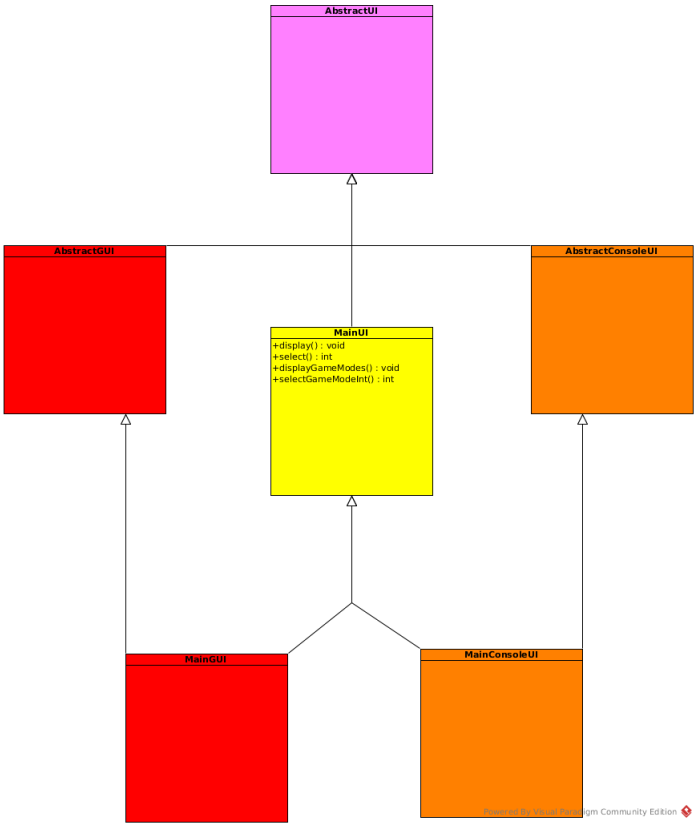


Figure 17: Structure héritage: Interfaces

6.1.6 Diagramme complet du système

Voir page suivante.

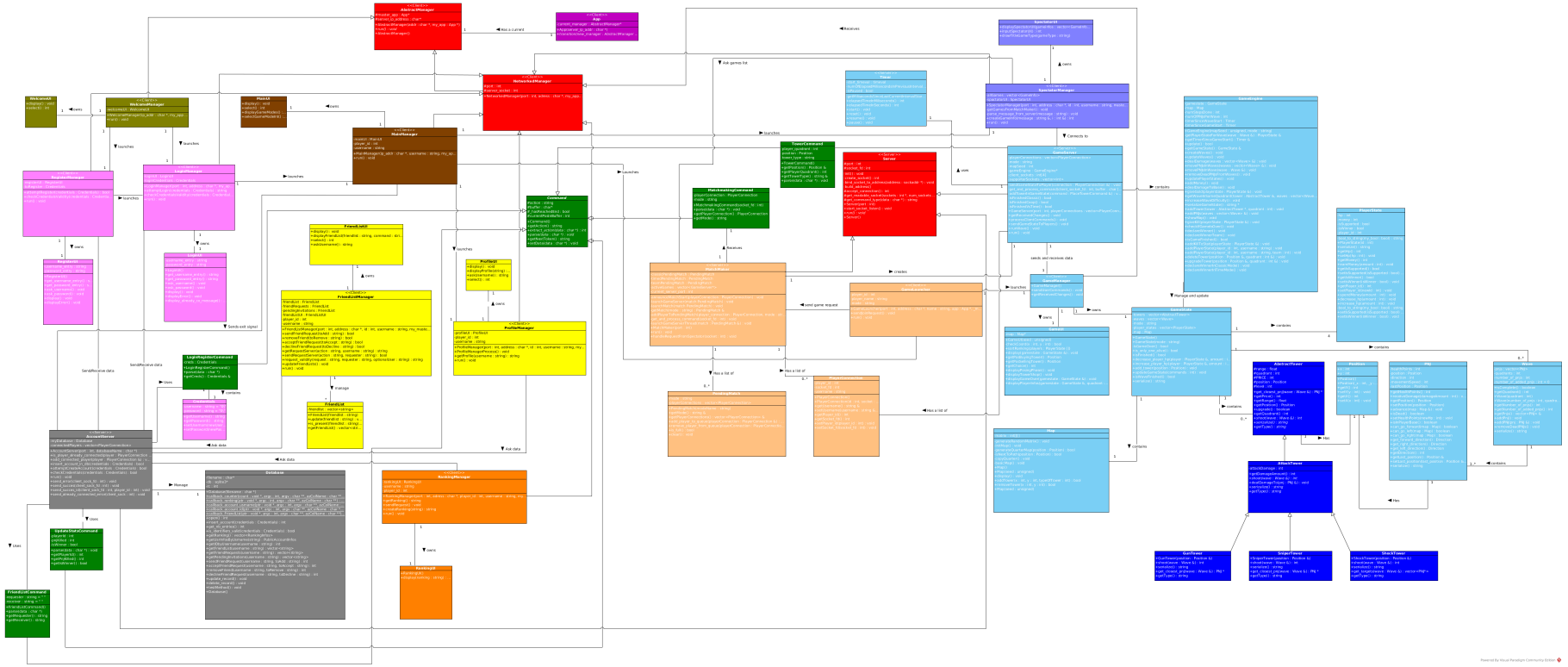


Figure 18: Diagramme de classe

6.2 Fonctionnement du système

Voici les diagrammes de séquence illustrant le fonctionnement de différentes parties du système.

6.2.1 Inscription

Ce diagramme décrit la séquence d'action réalisées lorsqu'un utilisateur souhaite créer un compte. Les données entrées par l'utilisateur seront une première fois vérifiées par le registerManager (vérification de champs vide, nombre de caractères ...) avant des les envoyer à l'accountServeur. Celui-ci vérifiera que le nom d'utilisateur est disponible.

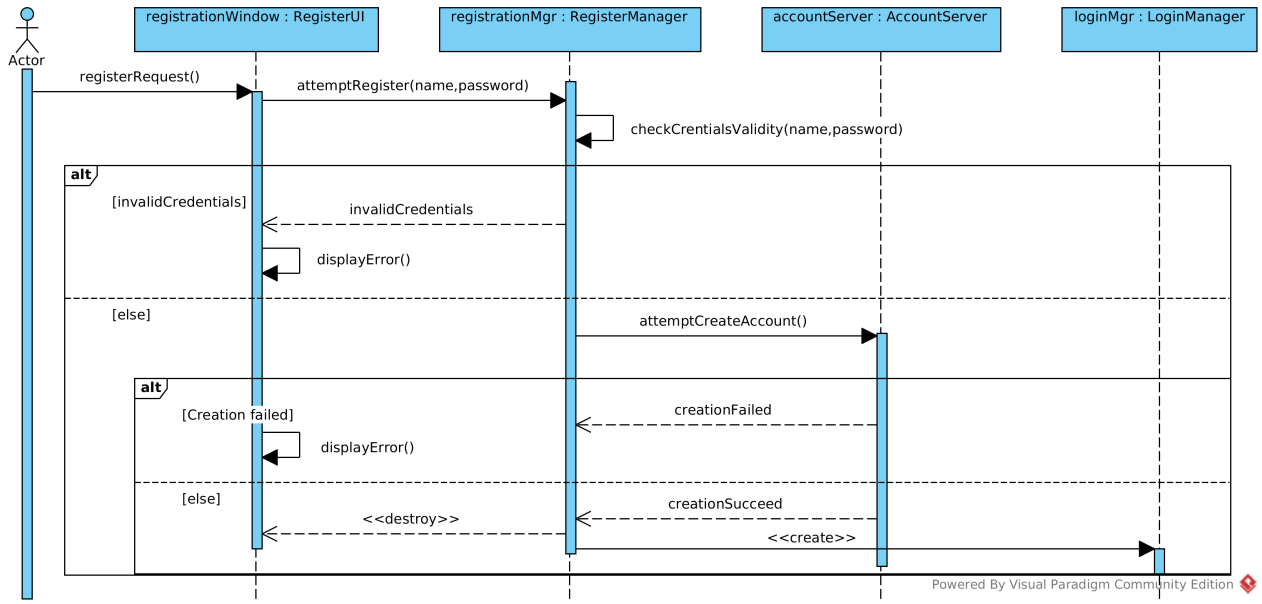


Figure 19: Sequence : Inscription

6.2.2 Connexion

Ce diagramme décrit la séquence d'actions réalisées lorsqu'un utilisateur veut se connecter à son compte utilisateur. Comme pour l'inscription, les données sont d'abord vérifiées en interne avant des les envoyer au serveur. Celui-ci vérifiera cette fois que le nom d'utilisateur et le mot de passe concordent.

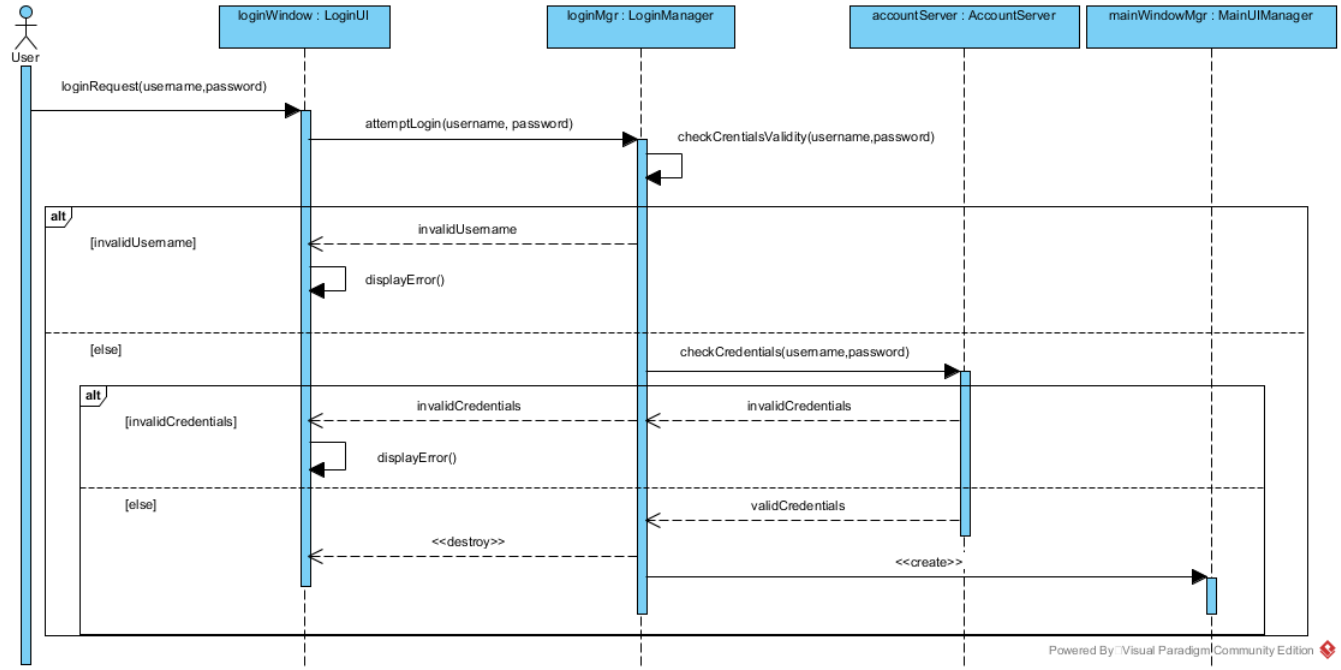


Figure 20: Sequence : Connexion

6.2.3 Matchmaking: Client

Ce diagramme décrit la séquence d'actions réalisées quand un utilisateur décide de rejoindre une partie (coté client).

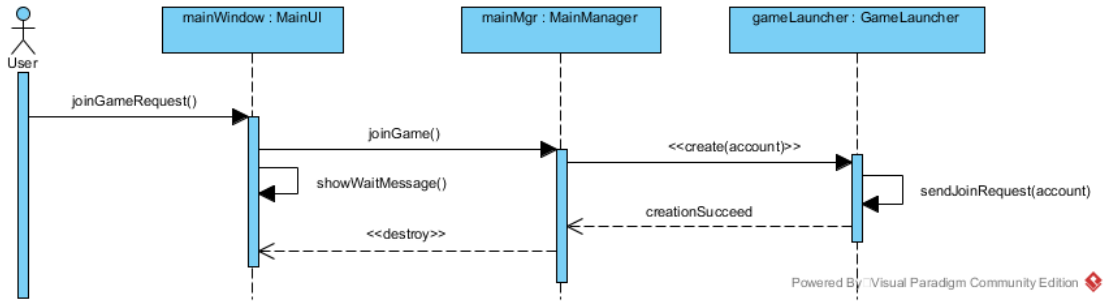


Figure 21: Sequence : Matchmaking client

6.2.4 Matchmaking: Serveur

Ce diagramme décrit la séquence d'actions réalisées quand un utilisateur décide de rejoindre une partie (coté serveur). Les pendingMatches sont des matchs en construction (tous les 4 joueurs n'ont pas encore été trouvés).

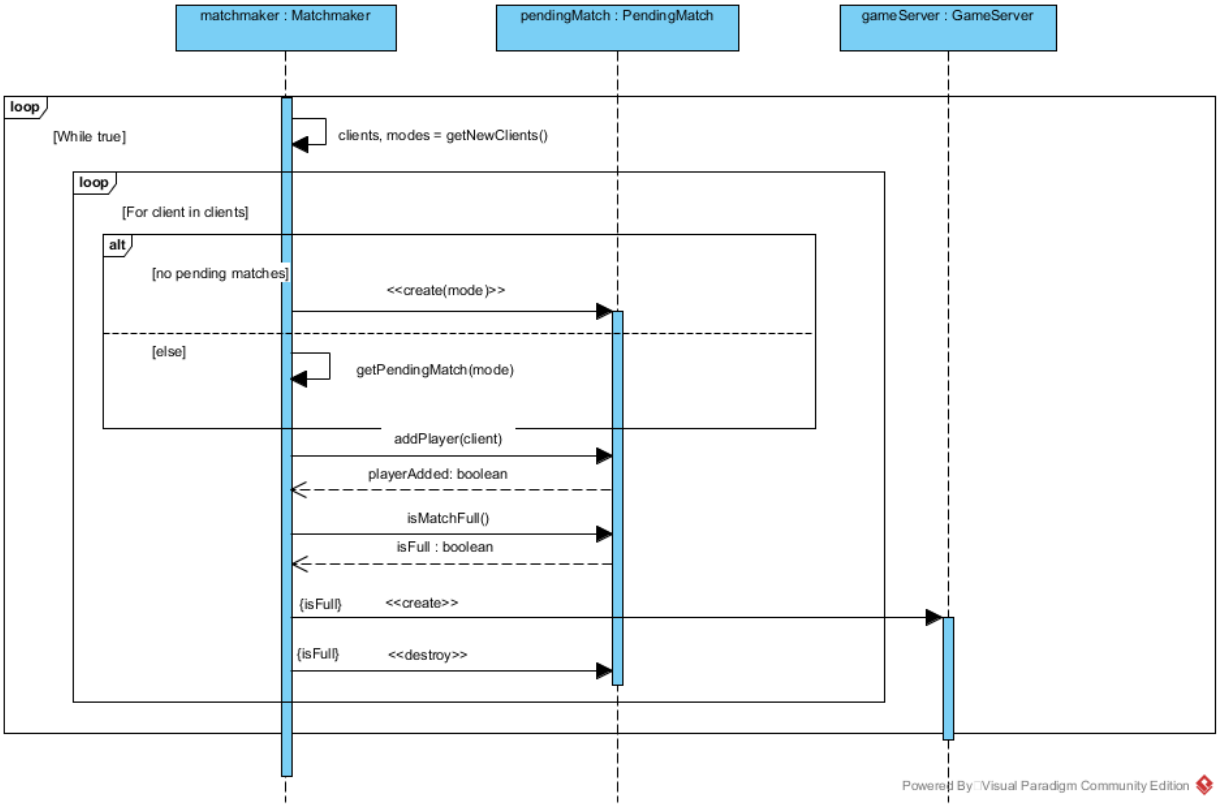


Figure 22: Sequence : Matchmaking serveur

6.2.5 Boucle du jeu: Client

Ce diagramme décrit la séquence d'actions réalisées lors d'une "frame". En effet cette séquence d'actions se répète un nombre important de fois par seconde tant que la partie n'est pas terminée.

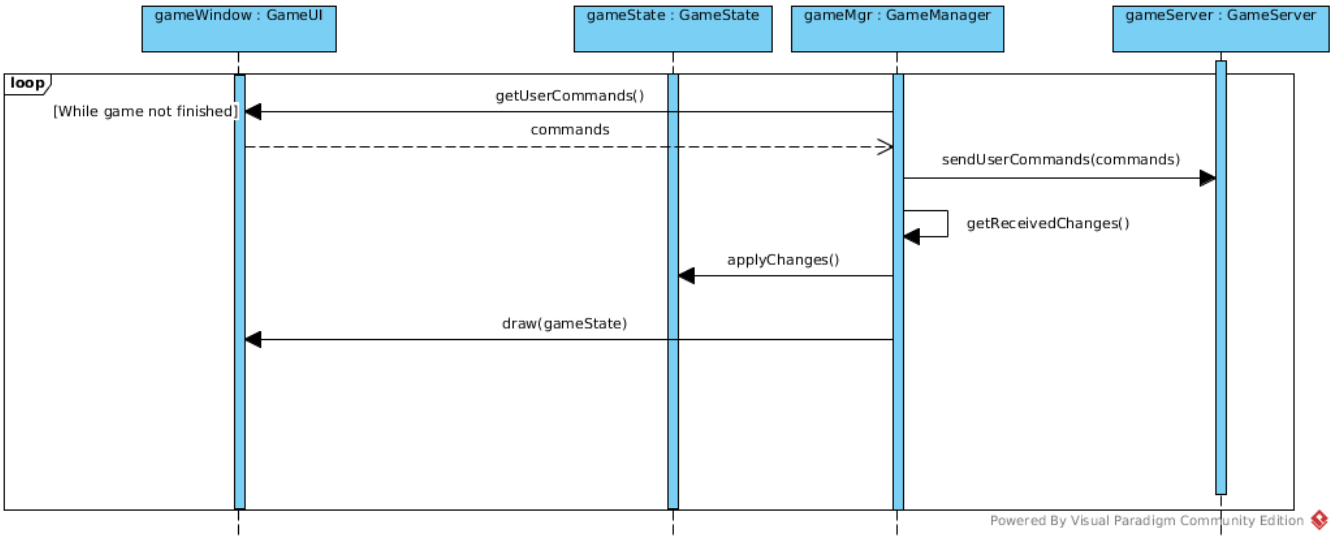


Figure 23: Sequence : Boucle du jeu client

6.2.6 Boucle du jeu: Serveur

Ce diagramme décrit la séquence d'actions réalisées par le gameServer, comme pour le diagramme précédent, cette séquence d'actions se répète un très grand nombre de fois par seconde.

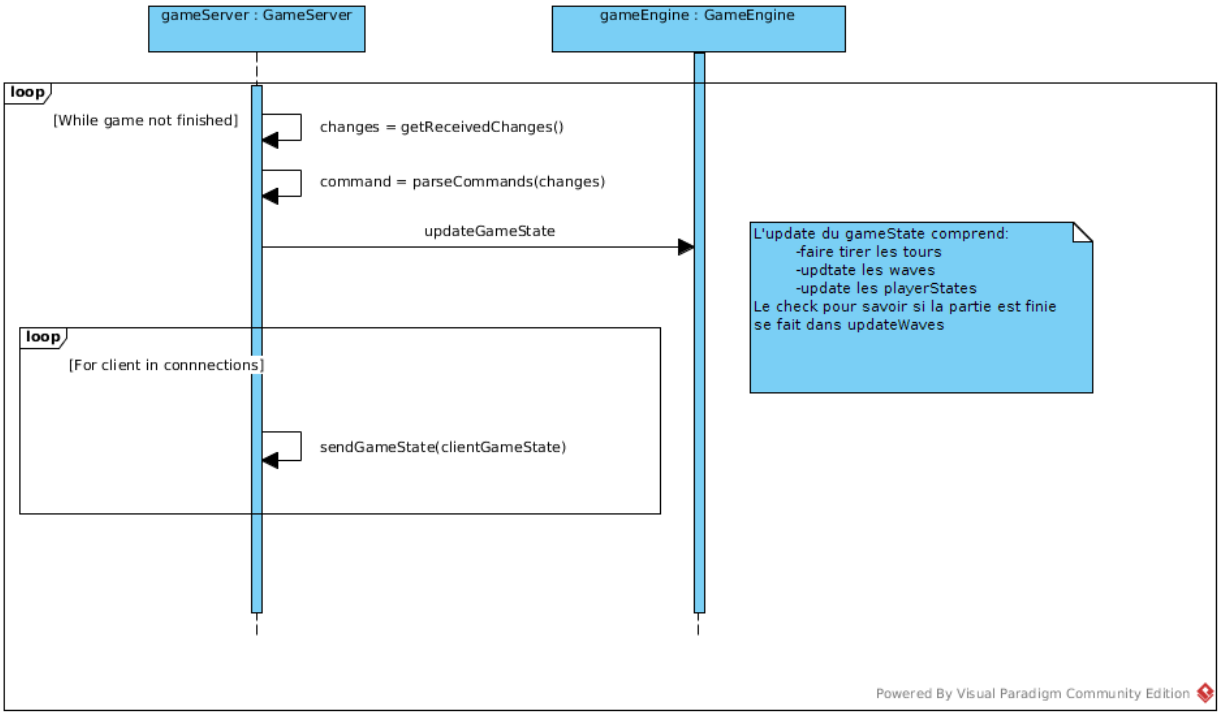


Figure 24: Sequence : Boucle du jeu serveur

6.2.7 Diagramme d'activité : Login

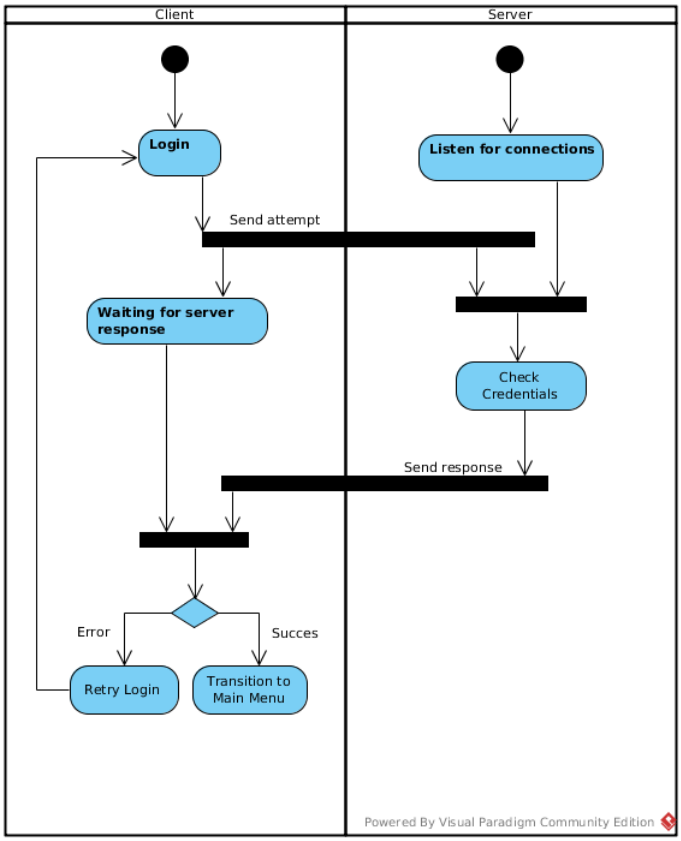


Figure 25: Activité : Matchmaking

7 Index des termes utilisés

- PNJ : page 3
- Vague : page 3
- Matchmaking : page 9
- Pseudo : page 6
- Frame : page 18