Équipe 209

PolyDiff
Document d'architecture logicielle

Version 1.0

Historique des révisions

Date	Version	Description	Auteur
2024-01-13	0.1	élaboration des objectifs et contraintes	Yanis Halouani et Rayan Fellah
2024-01-24	0.2	Ajout des diagrammes de cas d'utilisation	Rayan Fellah
2024-01-26	0.3	Ajout d'une partie des diagrammes de séquence et diagrammes de paquetage	Rayan Fellah et Yanis Halouani
2024-01-30	0.4	Révision des parties 1 à 5 et mise en page	Olivier Tremblay-Noël
2024-01-30	0.5	Ajouts des descriptions aux diagrammes de paquetage, Ajout du diagramme de déploiement	Yanis Halouani
2024-02-01	1.0	Relecture et mise en page finales	Olivier Tremblay-Noël

Table des matières

Document d'architecture logicielle

1. Introduction

L'architecture logicielle du jeu "PolyDifférences" est un élément essentiel de son développement. Ce document vise à définir les objectifs et contraintes architecturaux, à présenter les différents diagrammes tels que la vue des cas d'utilisation, la vue logique, la vue des processus et la vue de déploiement, et enfin à discuter de la taille et des performances du système.
La conception de l'architecture logicielle est cruciale pour assurer la robustesse, la scalabilité et l'efficacité du jeu, tout en répondant aux besoins spécifiques du projet.

2. Objectifs et contraintes architecturaux

Tout d'abord, on doit prendre en compte les objectifs et les contraintes architecturales liées au projet:

-Confidentialité: Les informations confidentielles des utilisateurs doivent être encapsulées afin d'éviter de les compromettre. On déduit que tout ce qui est lié à la gestion des mots de passe ainsi que le clavardage privé doit être géré par le serveur.

-Sécurité: La validation de l'authentification doit être faite par le serveur afin d'éviter des possible faille de sécurité, la communication HTTP doit être conforme au REST API et il faut s'assurer que le serveur n'envoie jamais un mot de passe au clients il fait seulement valider.

-Réutilisation: avoir des modules évolutifs (qui peuvent s'agrandir facilement) afin de pouvoir les réutiliser et les adapter au besoin pour les 2 clients ce qui n'est pas possible pour des modules très spécifiques.

-Échéancier: La gestion du temps est cruciale pour la réussite du projet. Il est important d'élaborer un échéancier complet et représentatif contenant les dates de début et de fin des principaux lots de travail, ainsi que l'effort estimé en heure pour chaque travail. Cette planification minutieuse est cruciale afin de réaliser le projet dans les temps et de remettre les livrables attendus aux dates requises.

-Portabilité: Le système doit être conçu de sorte à ne pas nécessiter d'énormes modifications dans la logique de conception entre le développement du client lourd et du client léger. En effet, l'architecture logicielle du projet doit être développée de telle sorte à être aussi bien compatible sur Windows 10 que sur Android 11. De plus, puisque l'application bureau et mobile doivent être cohérentes, ce qui rajoute un défi supplémentaire de portabilité dans le développement du projet afin de se faciliter la tâche et de pas avoir à faire un double travail.

- Outils de développement :

- Langages de développement :

3. Vue des cas d'utilisation

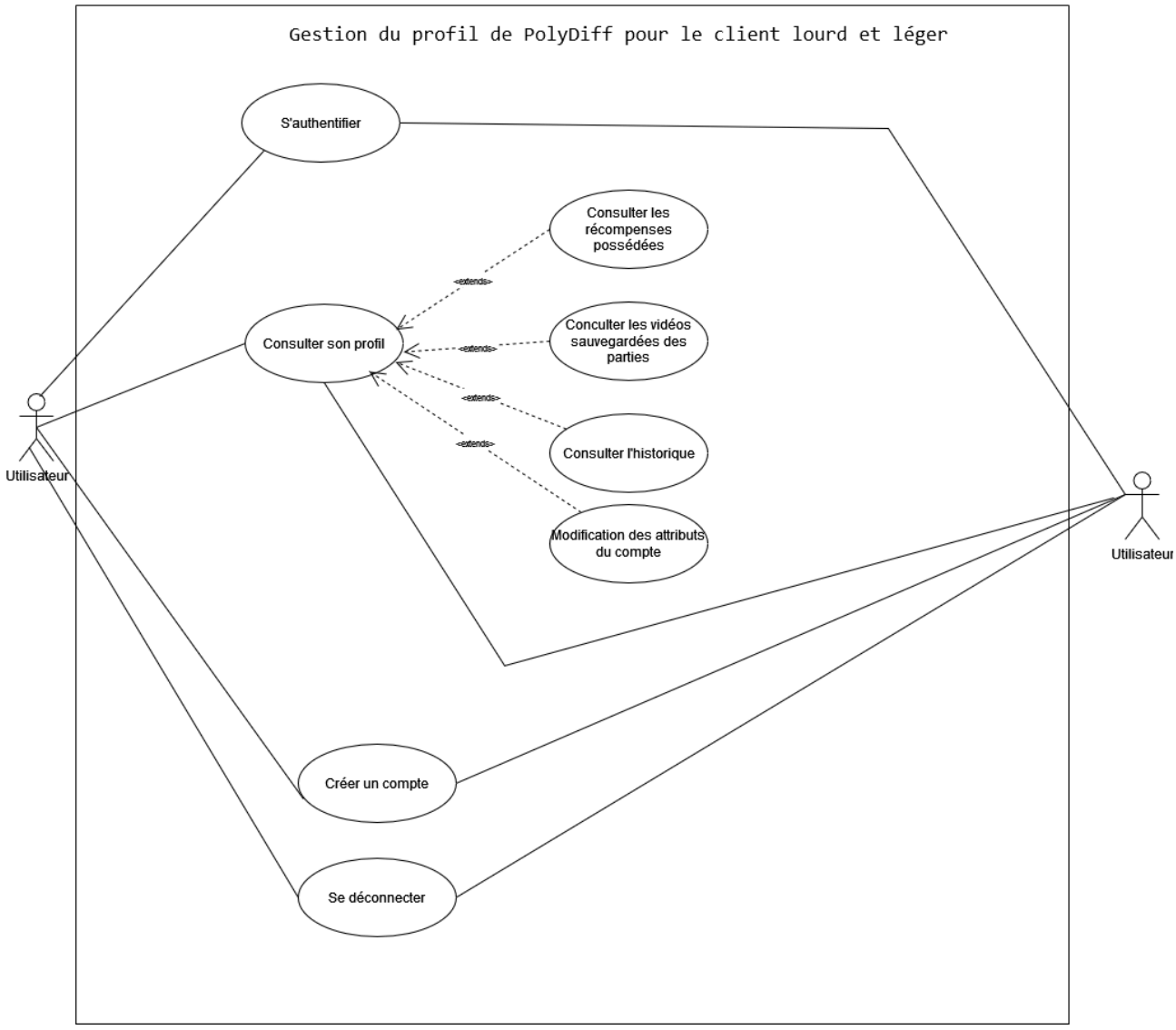


Figure 3.1 Diagramme de cas d'utilisation pour la gestion de compte

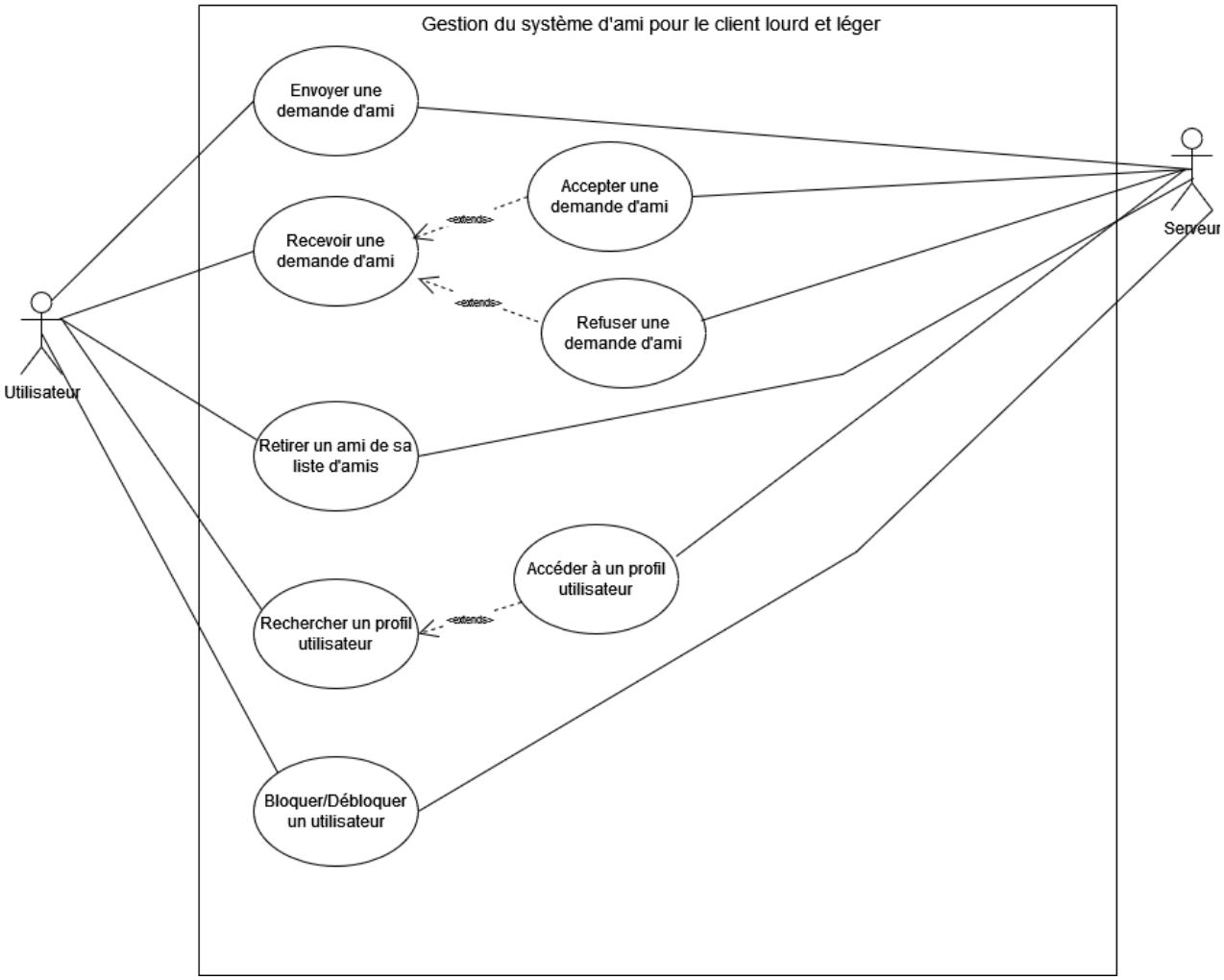
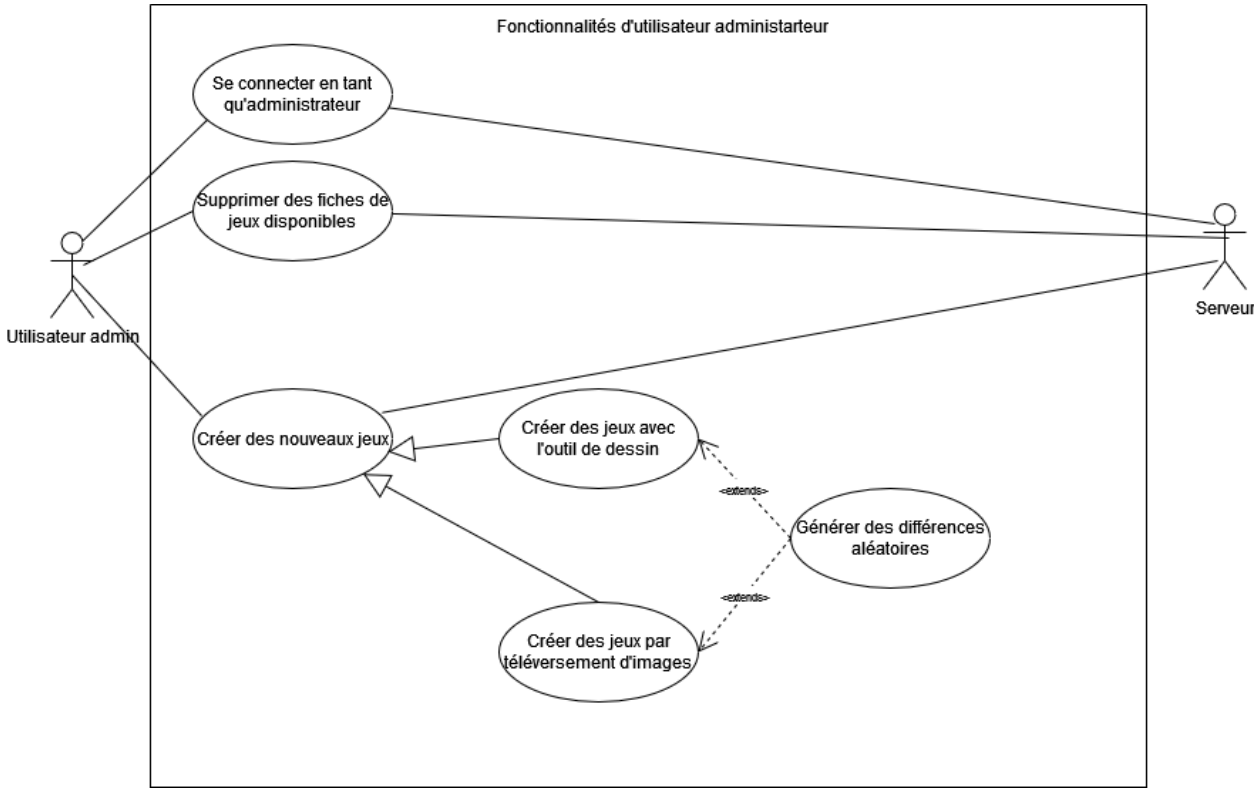


Figure 3.2 Diagramme de cas d'utilisation pour la gestion du système d'amis



3.3 Diagramme de cas d’utilisation pour l’administrateur

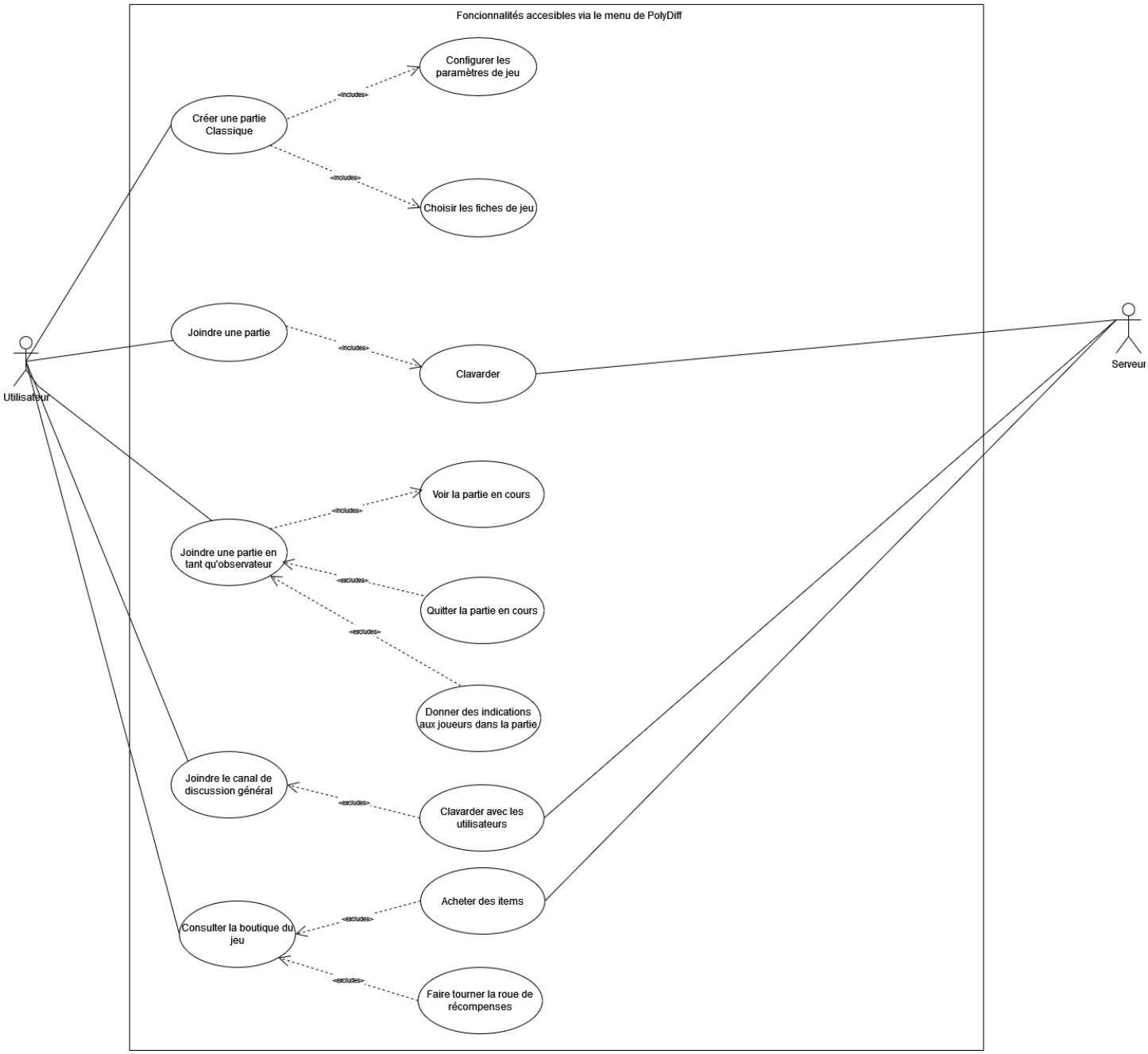


Figure 3.4 Diagramme de cas d'utilisation des fonctionnalités à partir du menu principal

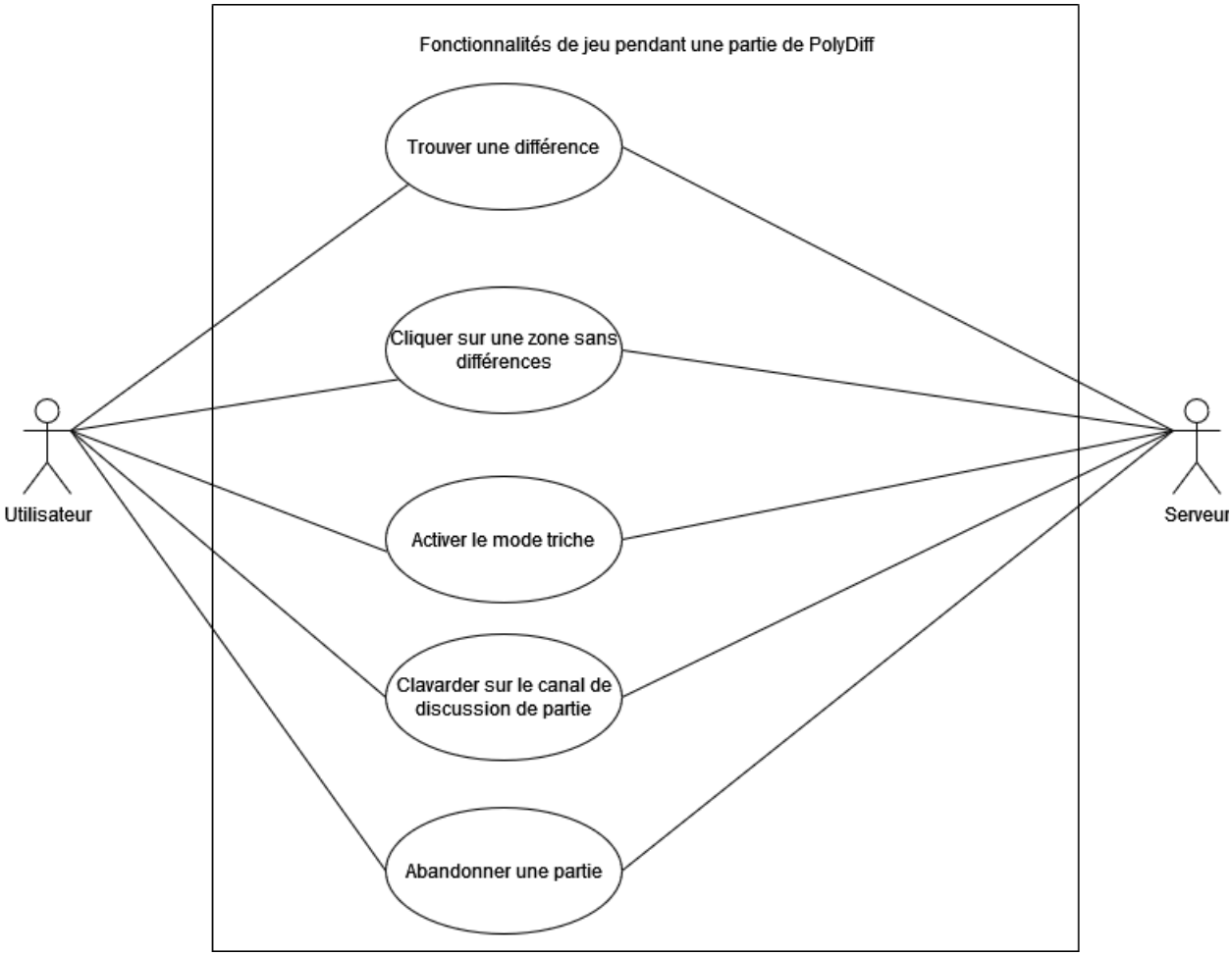


Figure 3.5 Diagramme de cas d'utilisation des fonctionnalités pendant une partie

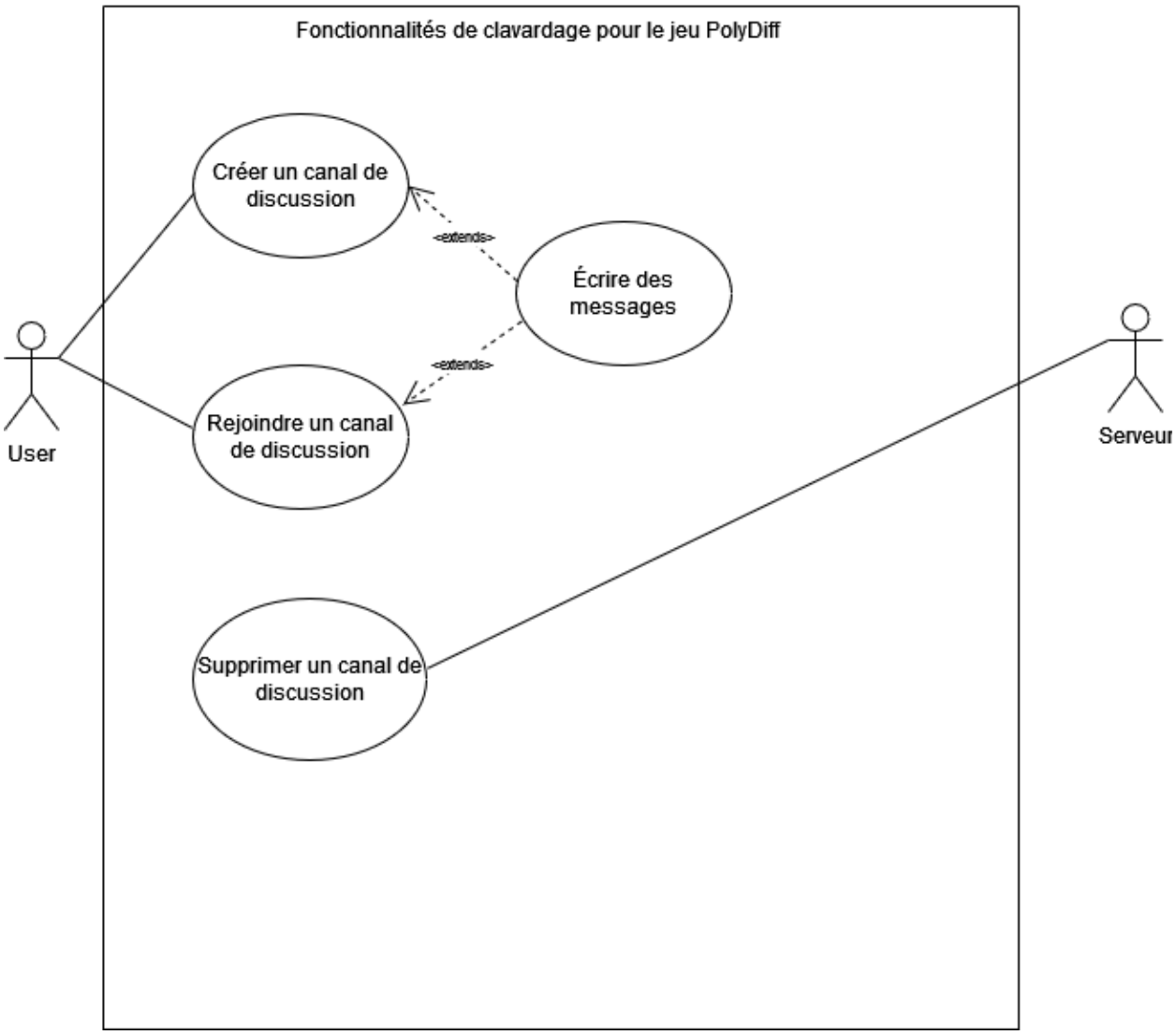


Figure 3.6 Diagramme de cas d'utilisation des fonctionnalités de clavardage

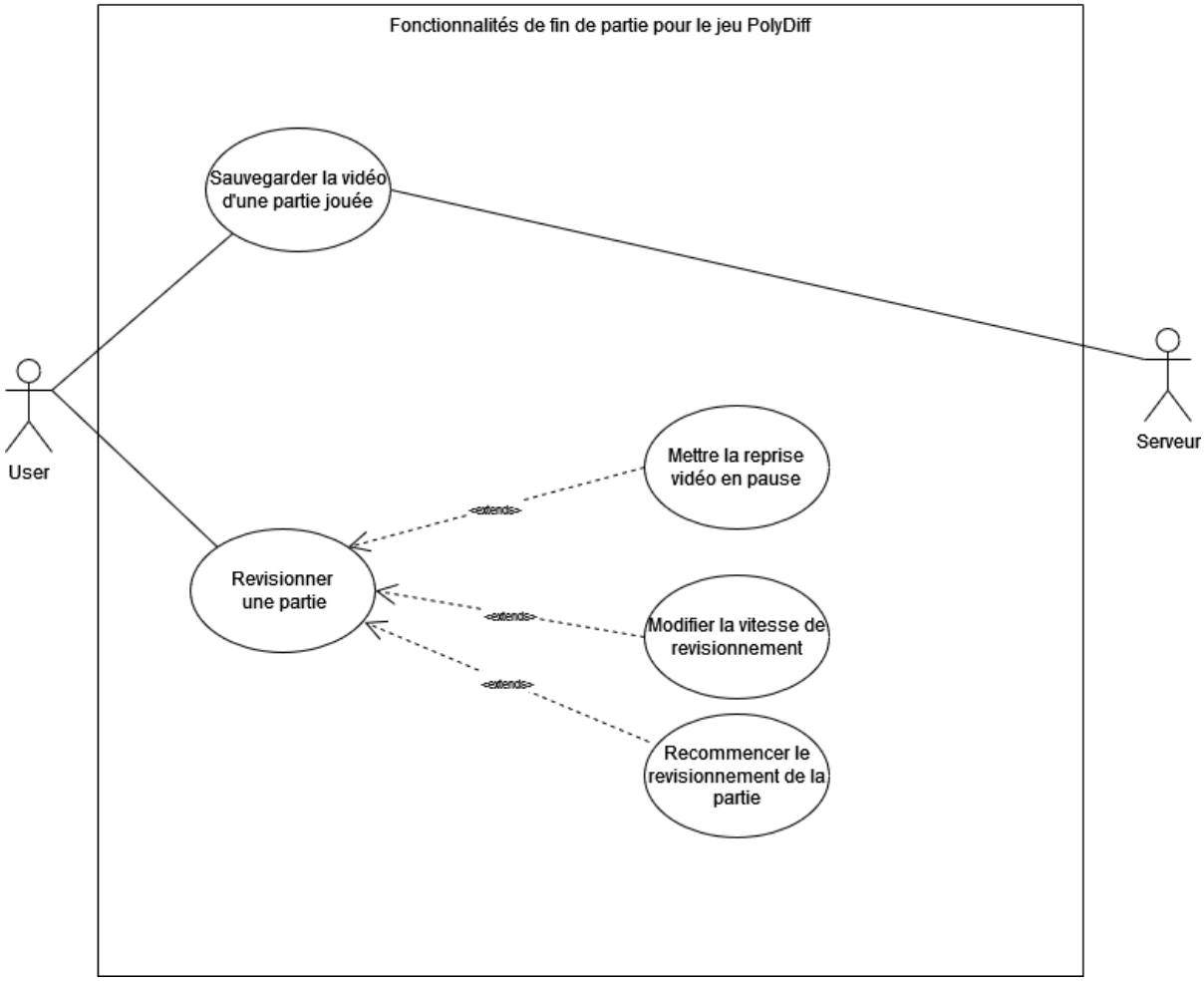
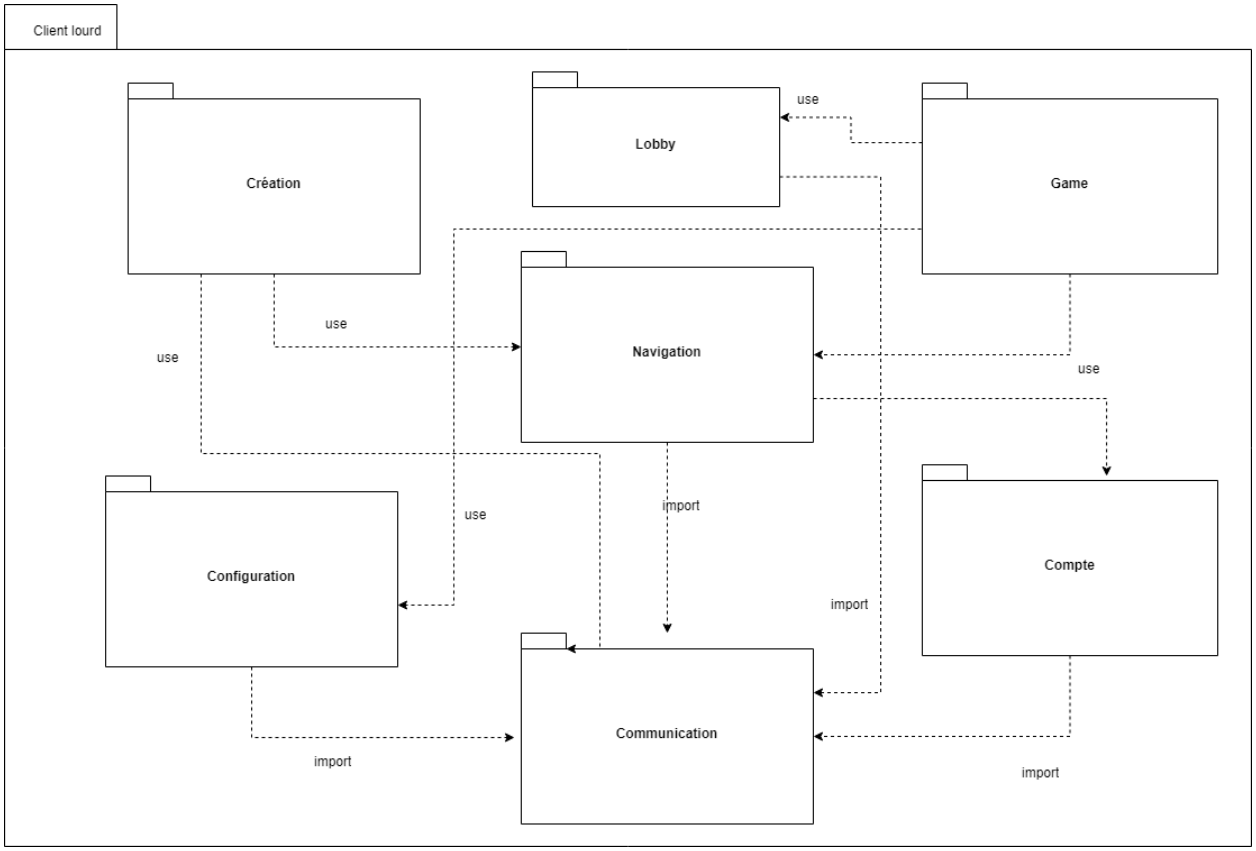


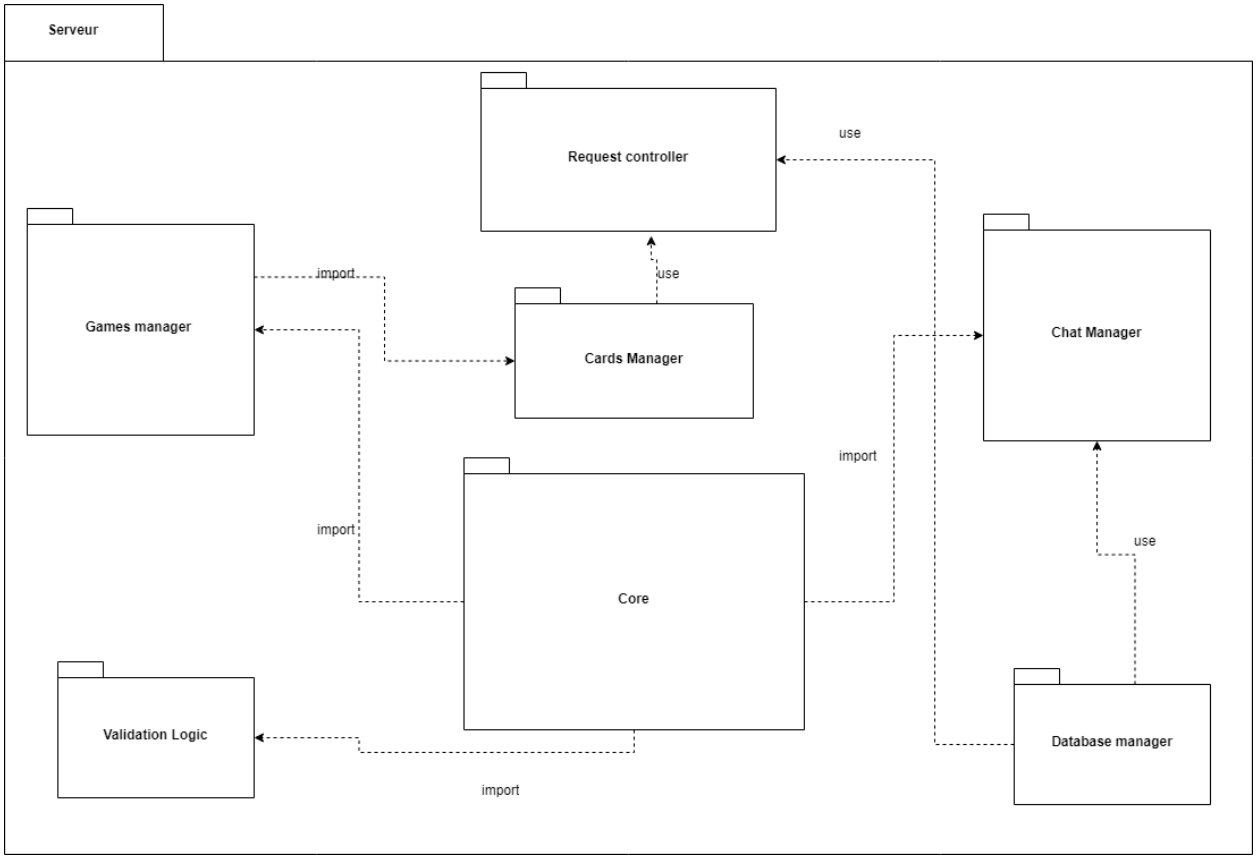
Figure 3.7 Diagramme de cas d'utilisation des fonctionnalités d'une fin de partie

4. Vue logique

Client Lourd
Est responsable de tout ce qui concerne le côté client sur desktop (electron), on y trouve les différents paquetages liés au modes de jeu, à la création de jeux ainsi que la navigation dans l'application.

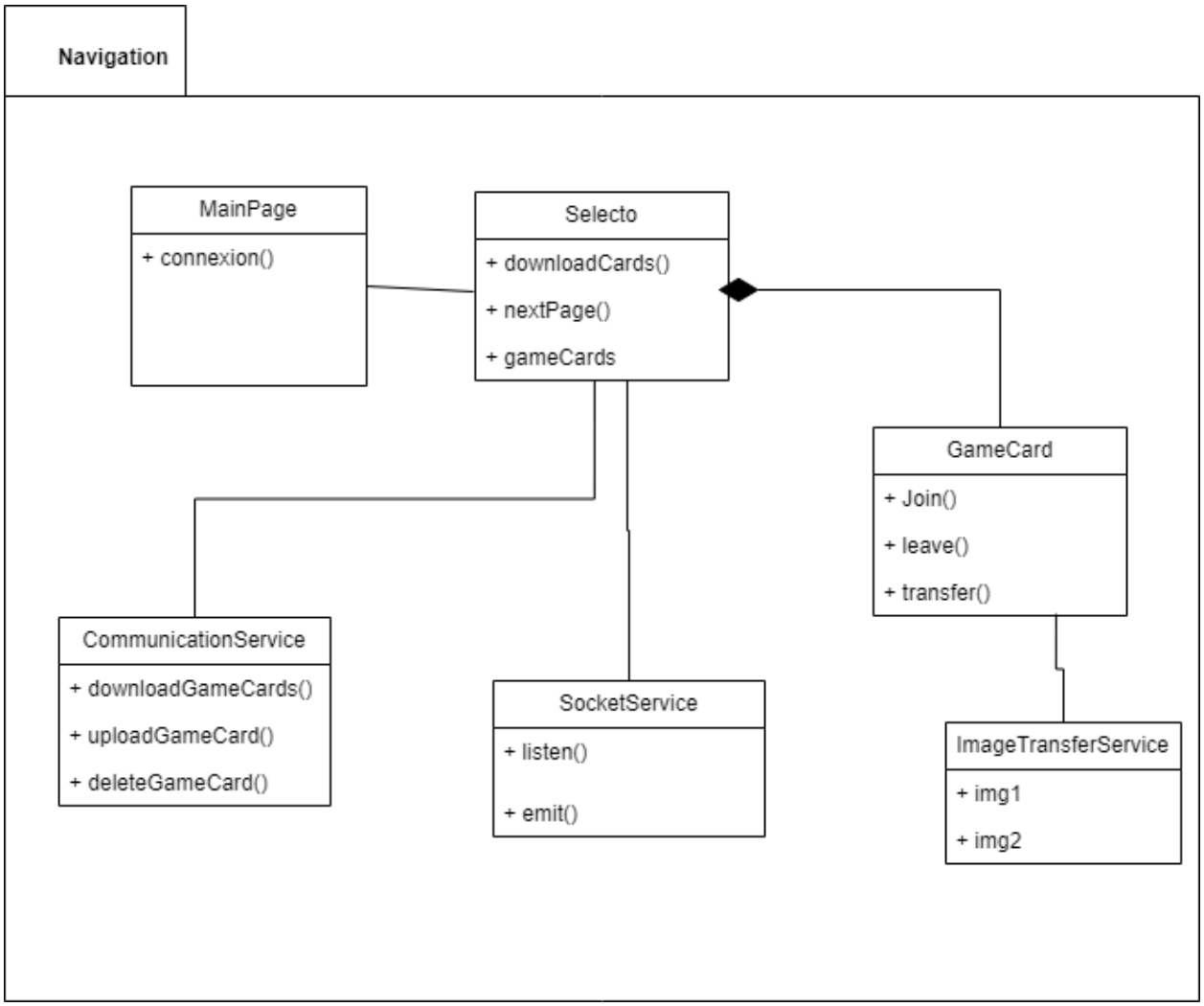


Serveur
Est responsable de tout ce qui concerne le serveur, dont la gestions des parties, la validation et le pointage, la communication entre joueurs ainsi que la communication avec la base de données.

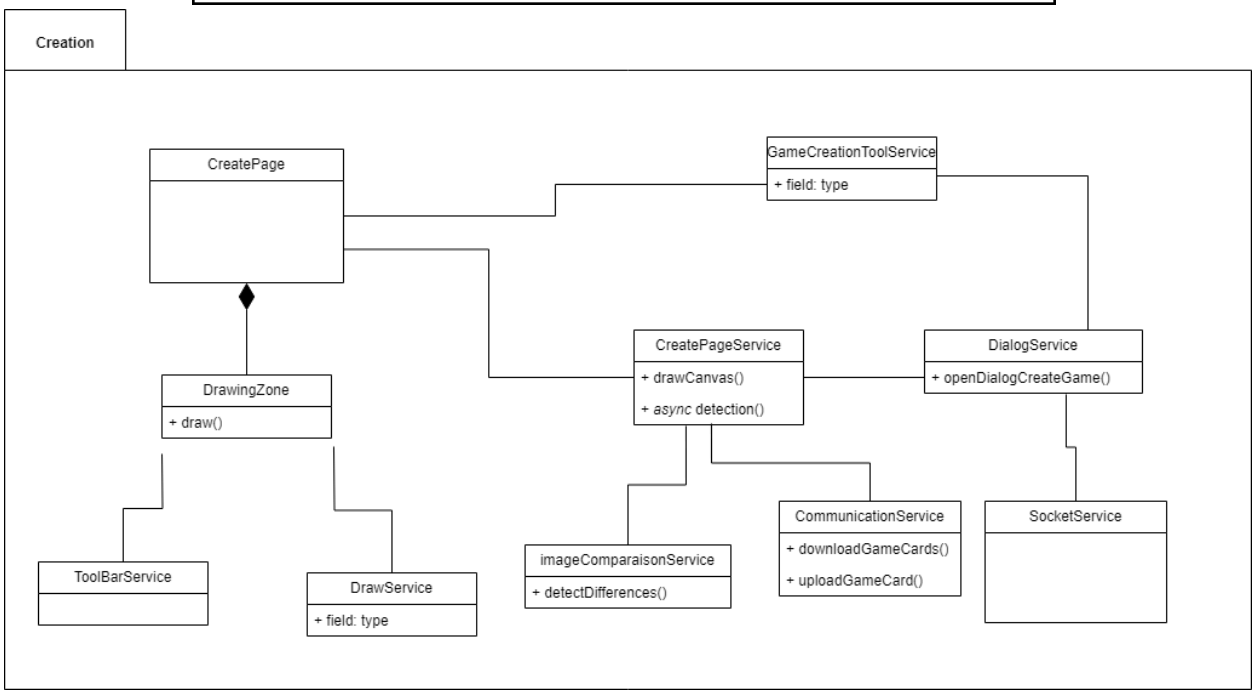


Vues Détaillées Clients:

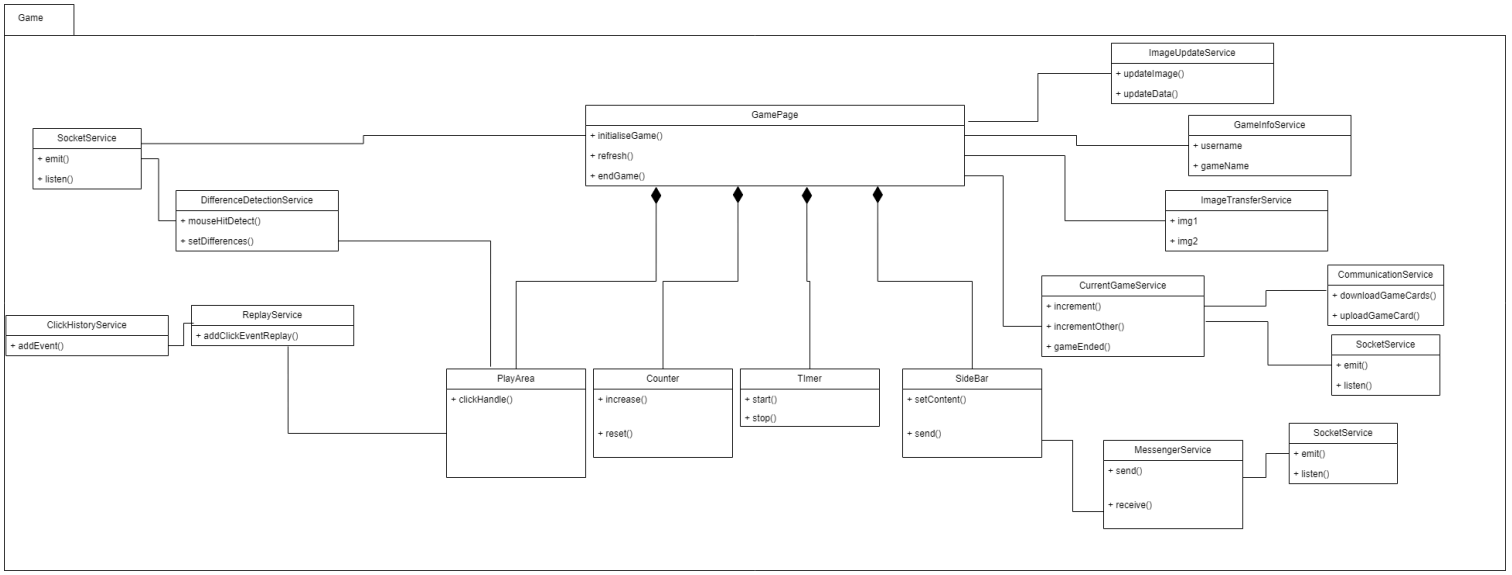
Navigation
Est responsable de la navigation dans le côté client, cela inclut l'affichage des parties disponibles ainsi que la consultation du compte



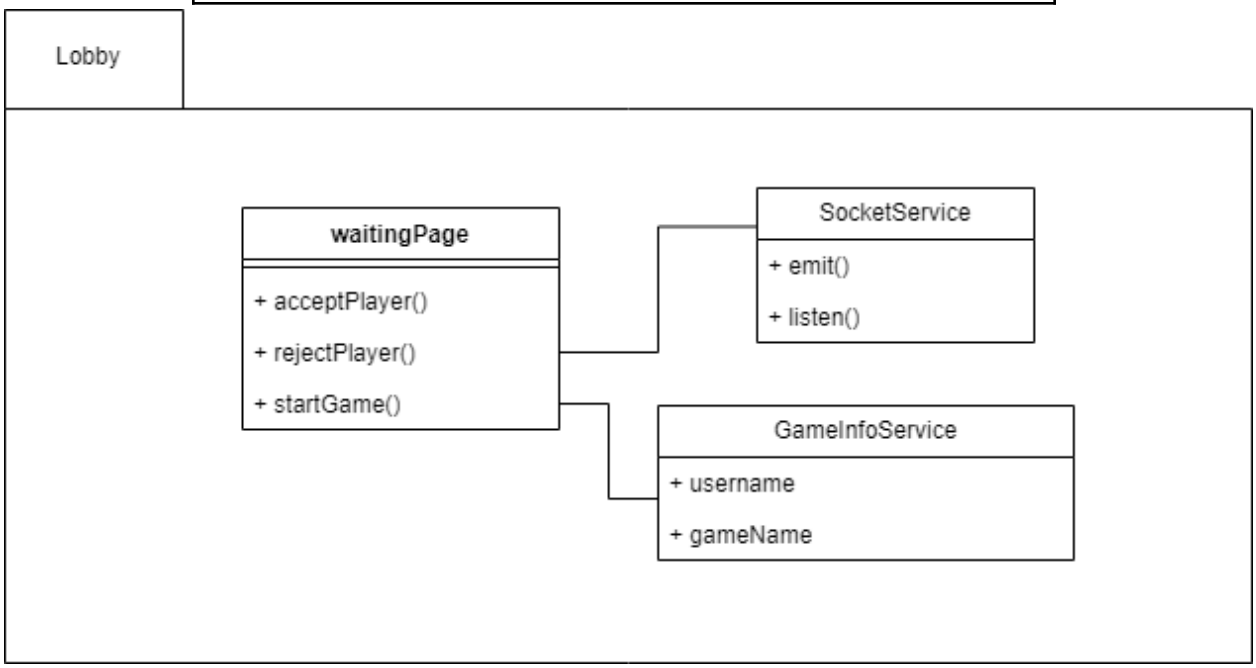
Création
Est responsable de tout ce qui permet de créer une fiche de jeu, cela inclut le dessin et la sauvegarde des fiches.



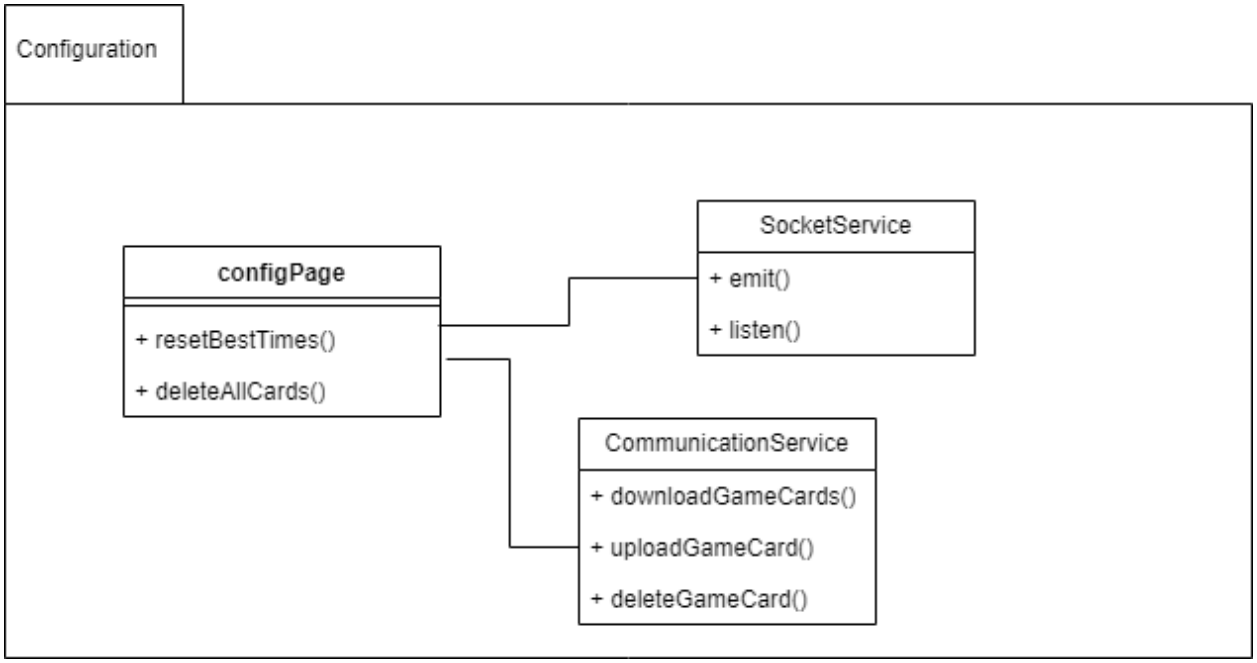
Game
Est responsable d'afficher aux joueurs les bonnes informations lors d'une partie, il est aussi responsable de la détection des événements émis par le joueur et leur traitement.



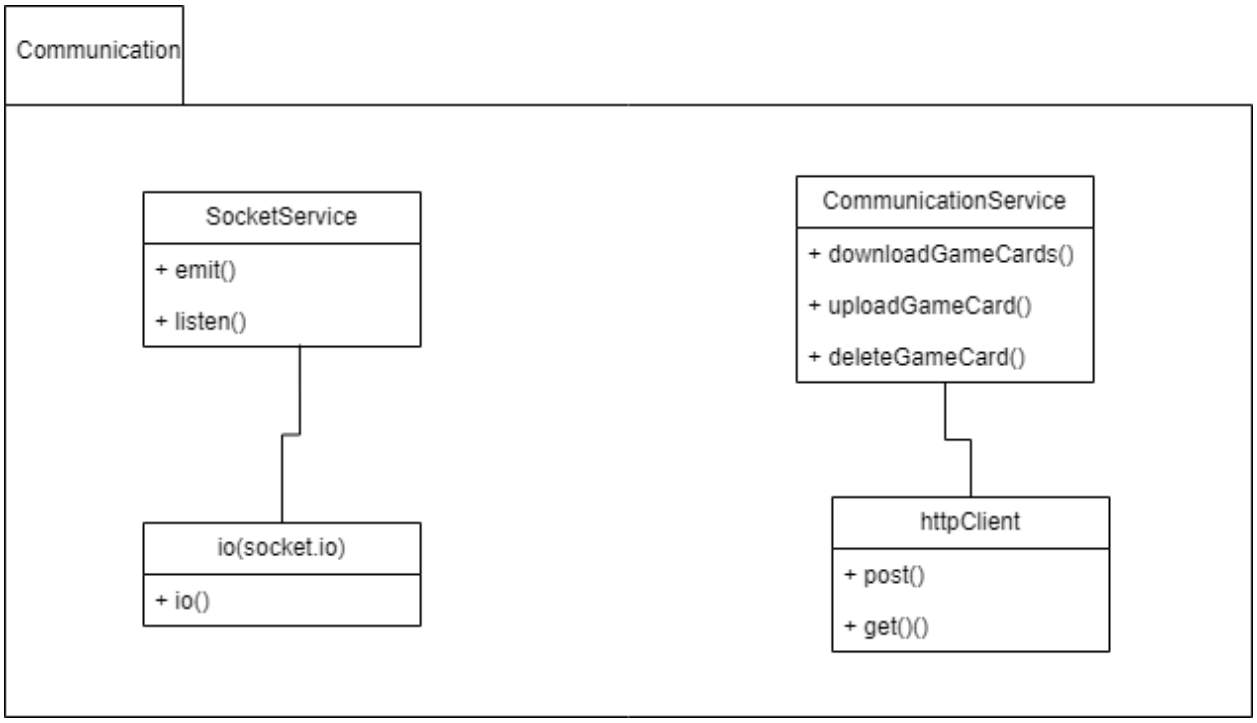
Lobby
Est responsable d'accueillir des nouveaux joueurs dans une partie à venir et de choisir les paramètres de partie.



Configuration
Est responsable de la gestion des fiches de jeu, dont la remise à zéro des meilleurs temps ainsi que la suppression de fiches.

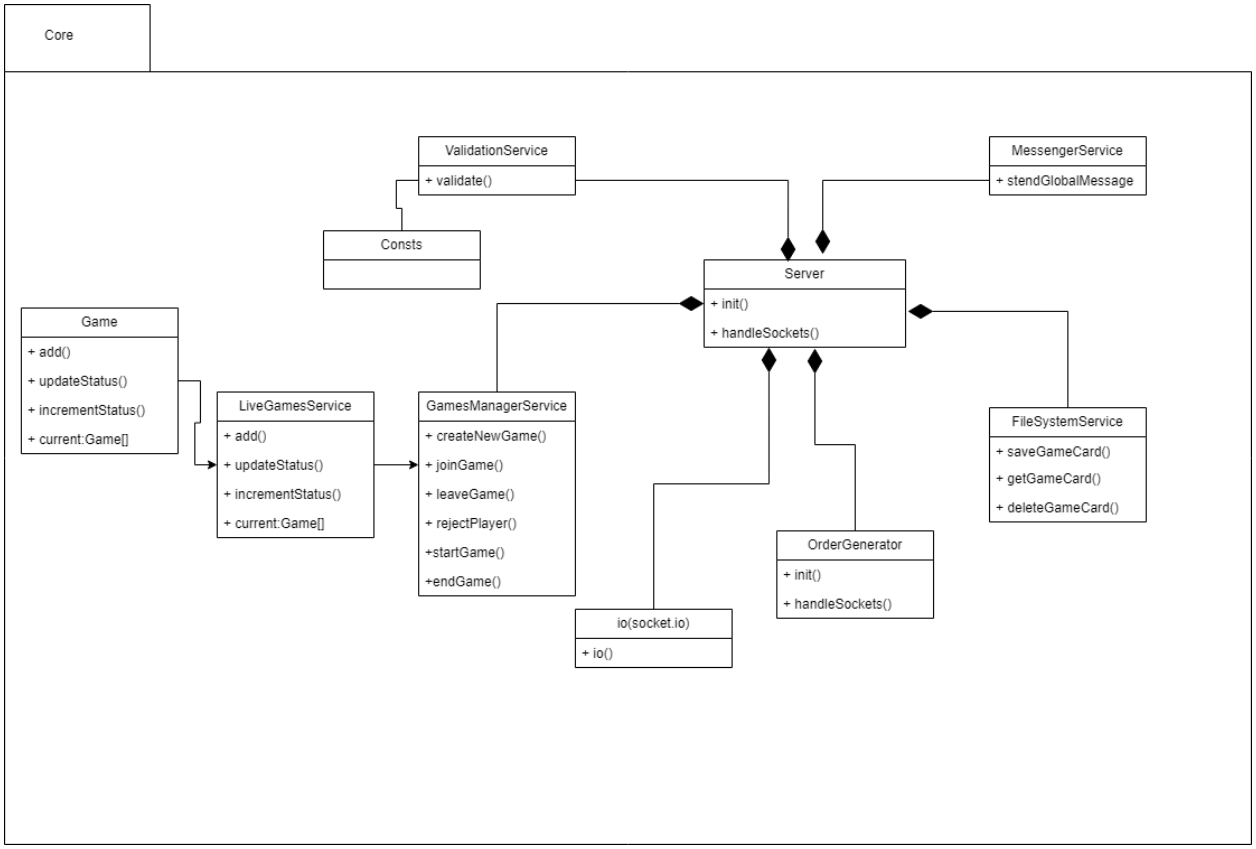


Communication
Est responsable de la communication avec le serveur à travers les protocoles HTTP et websockets

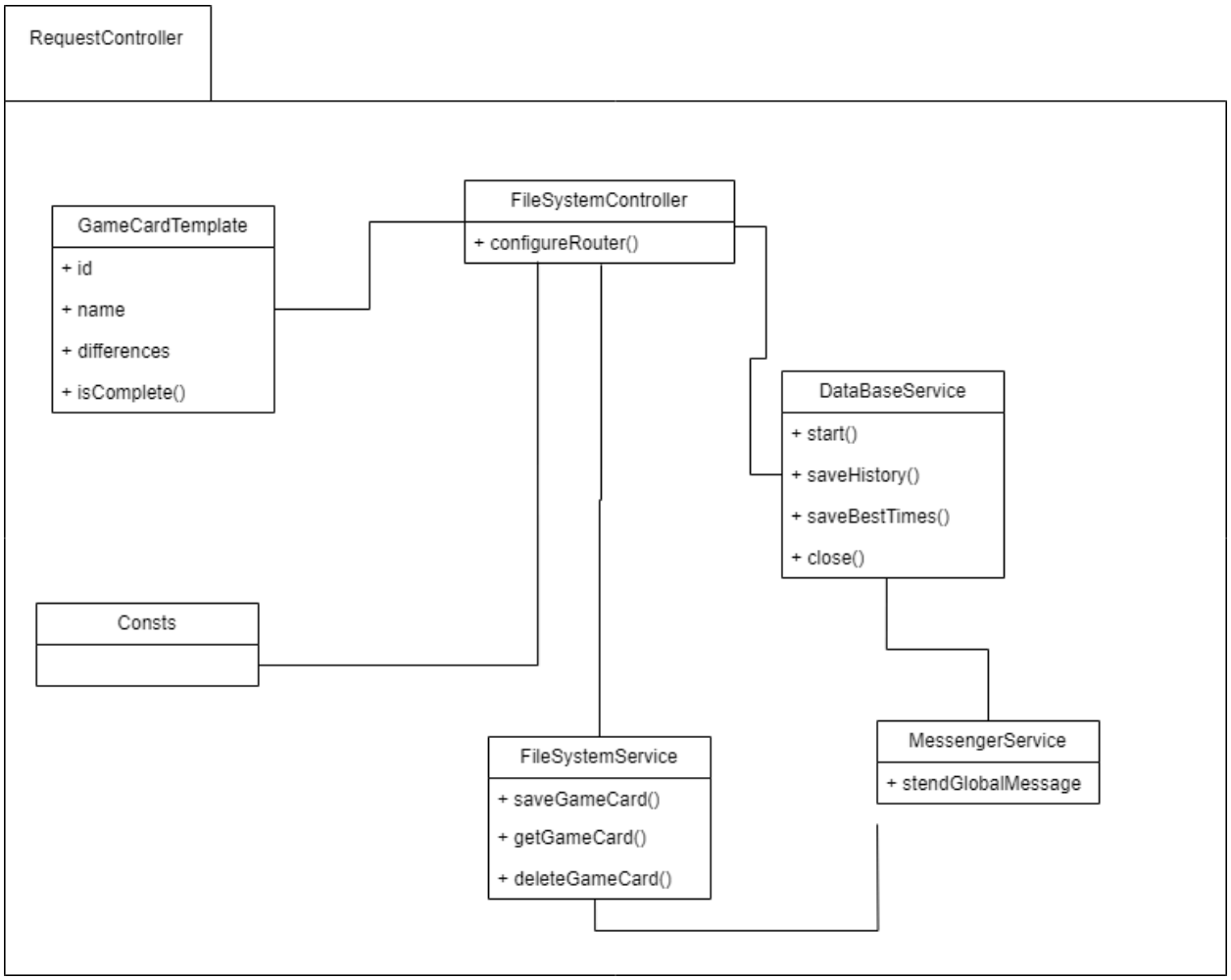


Vues Détaillés Serveur:

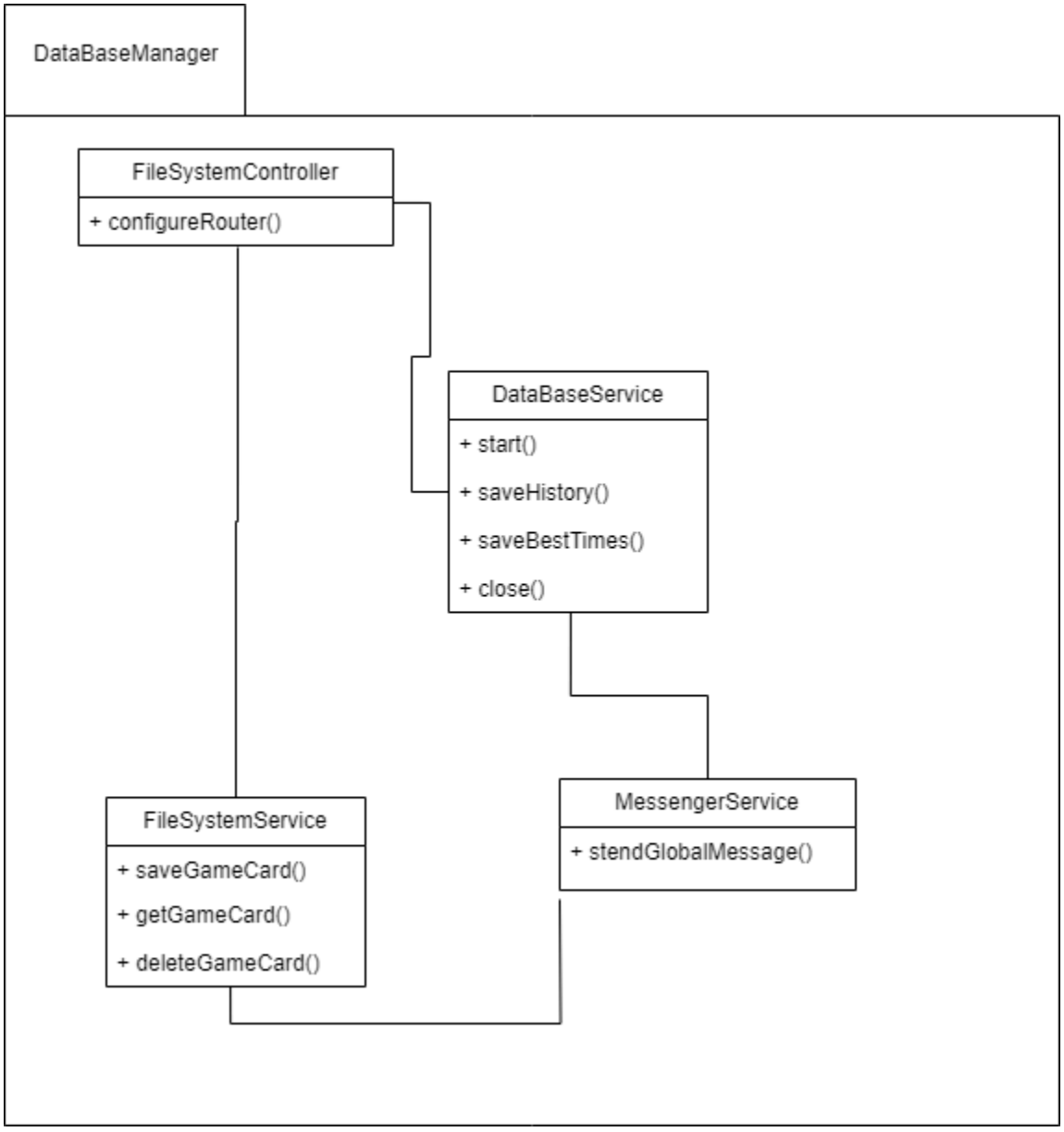
Core
Est responsable de relier les différents paquetages du serveur afin de s'assurer de son bon fonctionnement, traite aussi la communication websocket



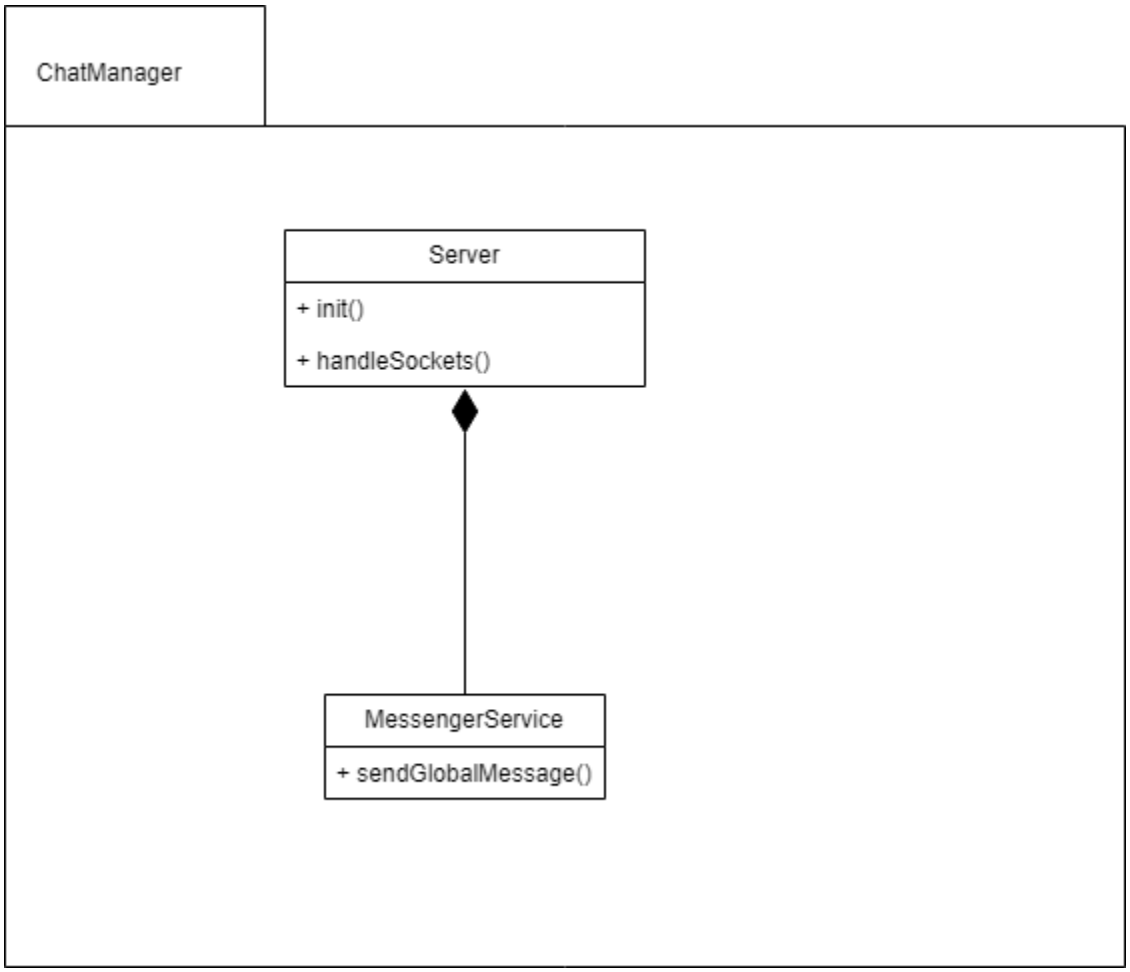
RequestController
Est responsable de la gestion des requêtes HTTP ainsi que leur redirection aux bons services.



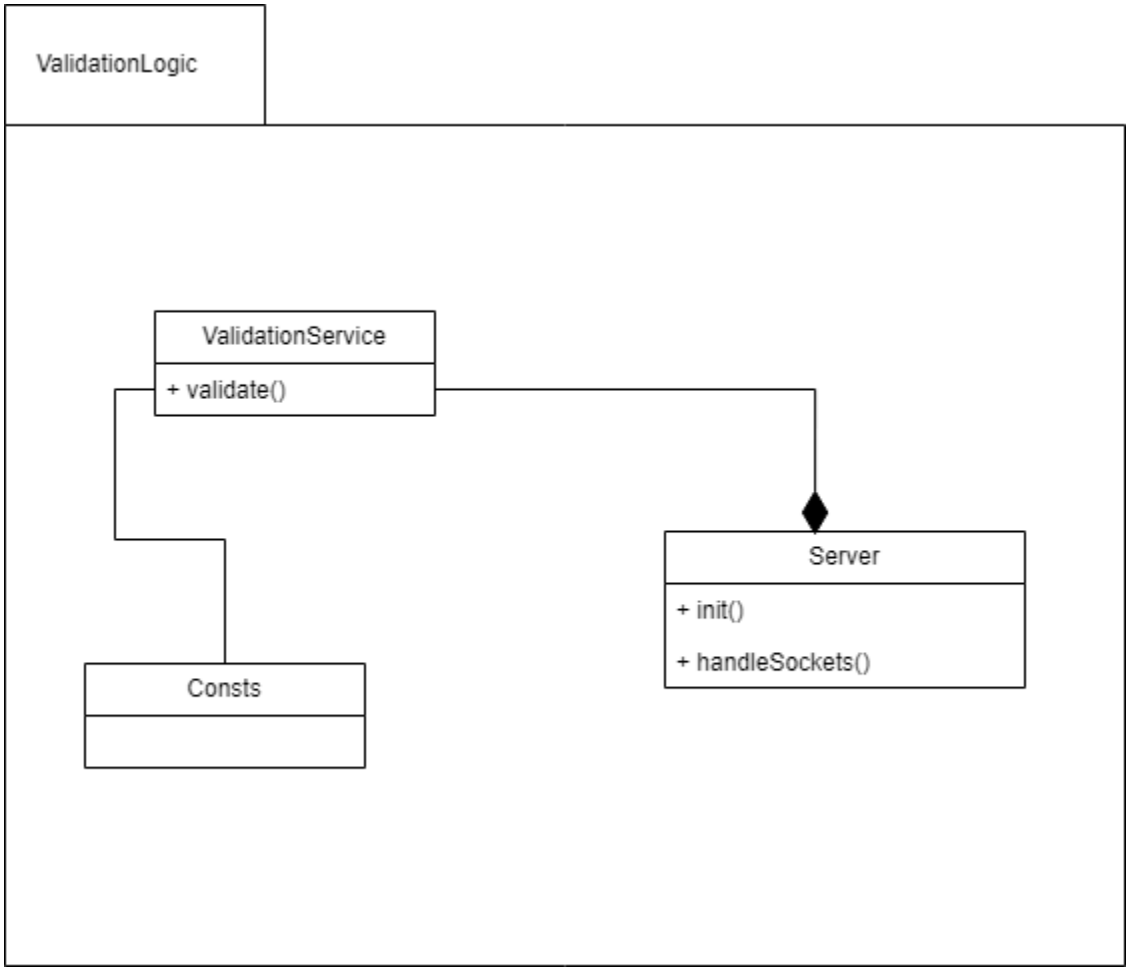
DataBaseManager
Est responsable de la communication avec la base de données. Il doit aussi s'assurer de redonner l'information auz bons services.



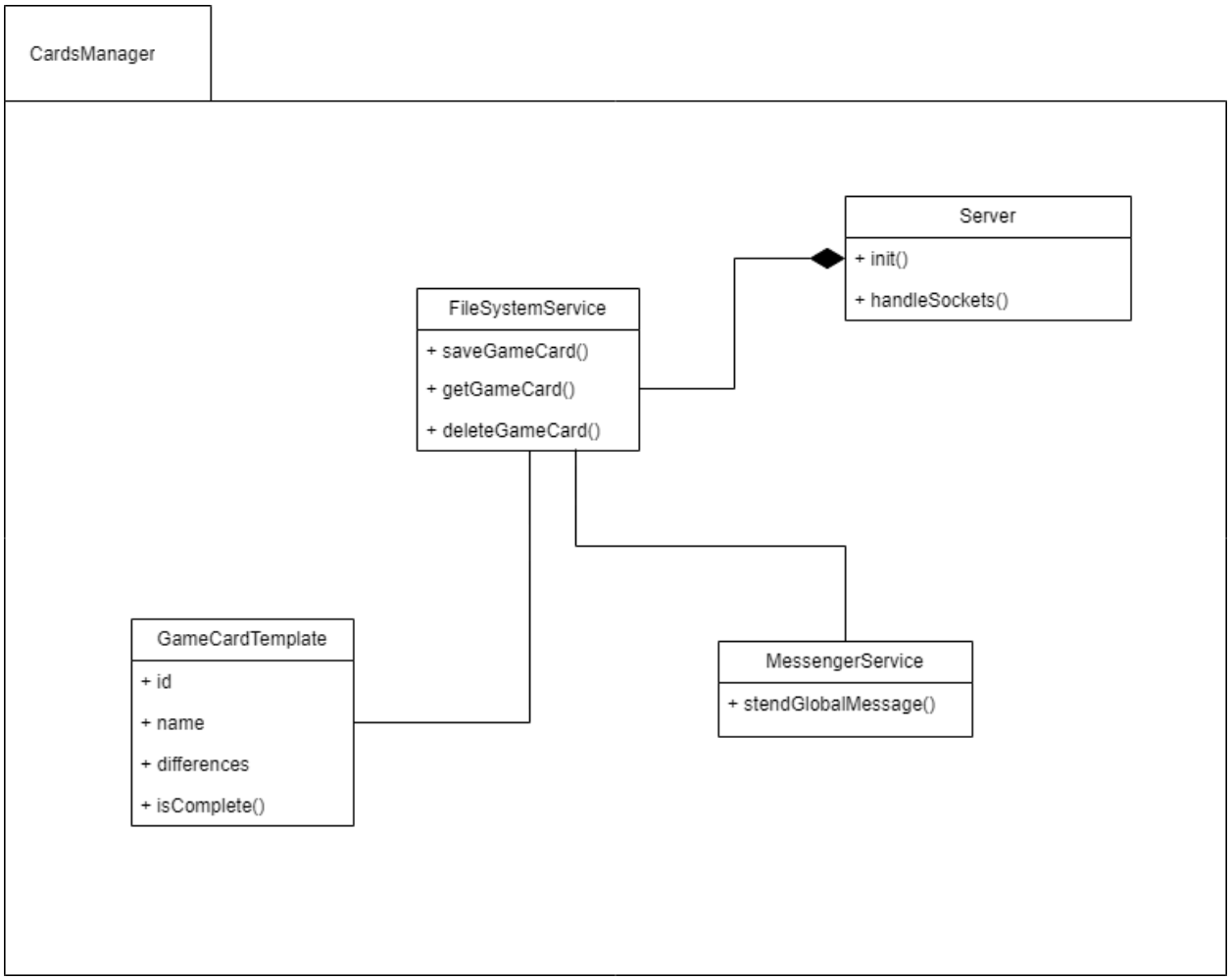
ChatManager
Est responsable de la gestion du clavardage et l'envoie des messages aux bons socket rooms.



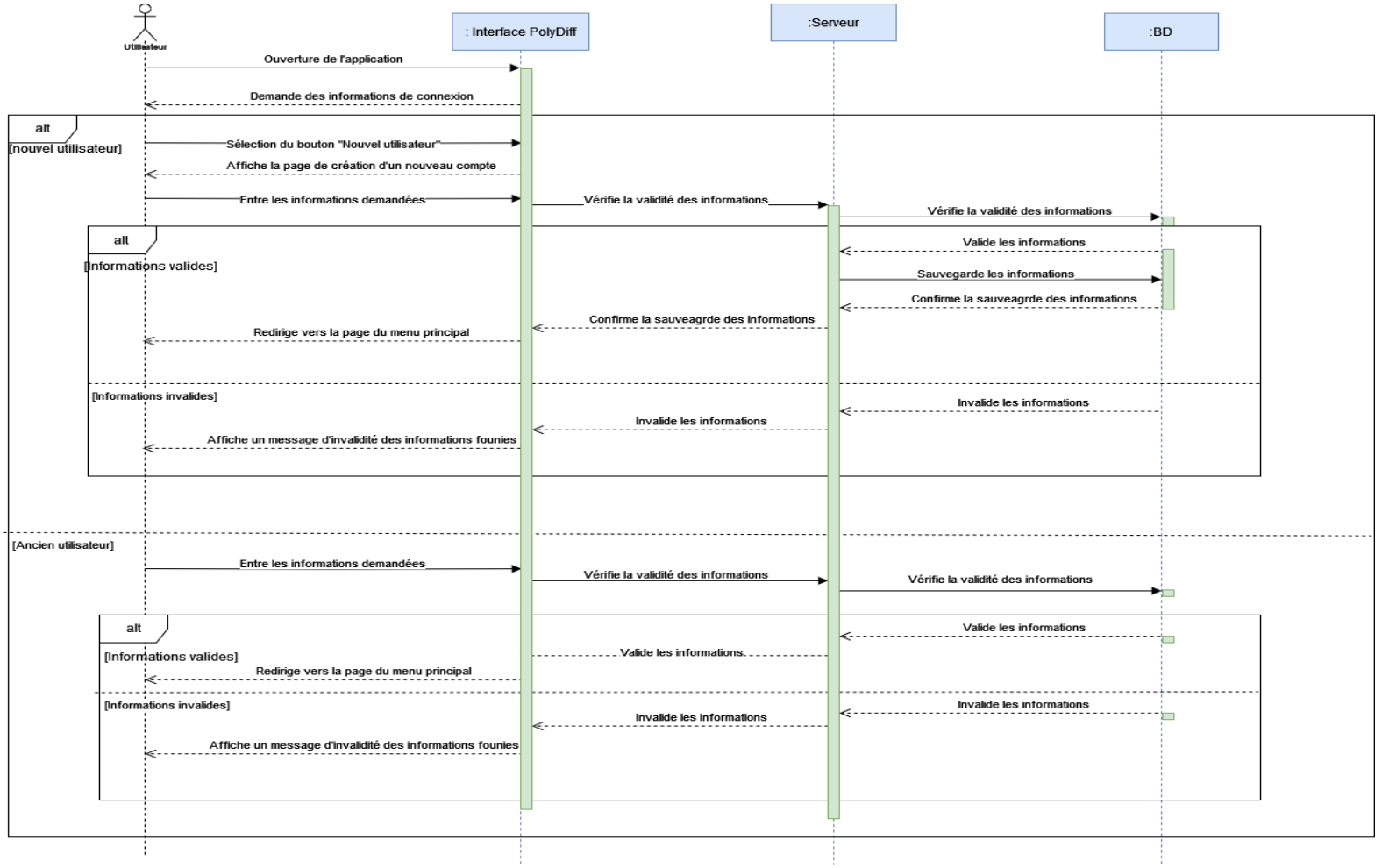
Validation
Est responsable de la validation d'une tentative d'un joueurs et déduire si ce dernier a trouver une différence ou pas.



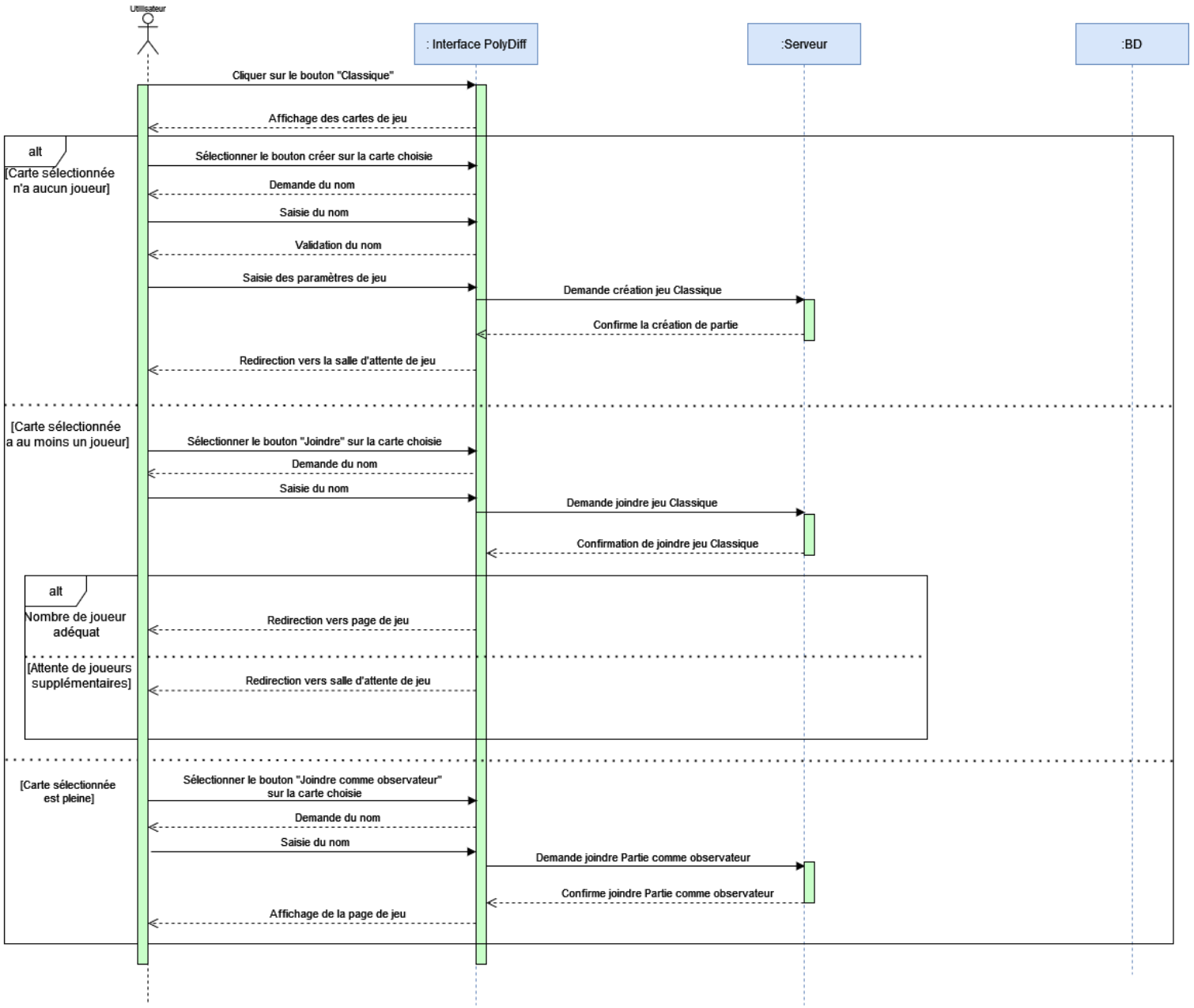
CardsManager
Est responsable de la gestion des fiches de jeux, leur création ainsi que leur suppression.



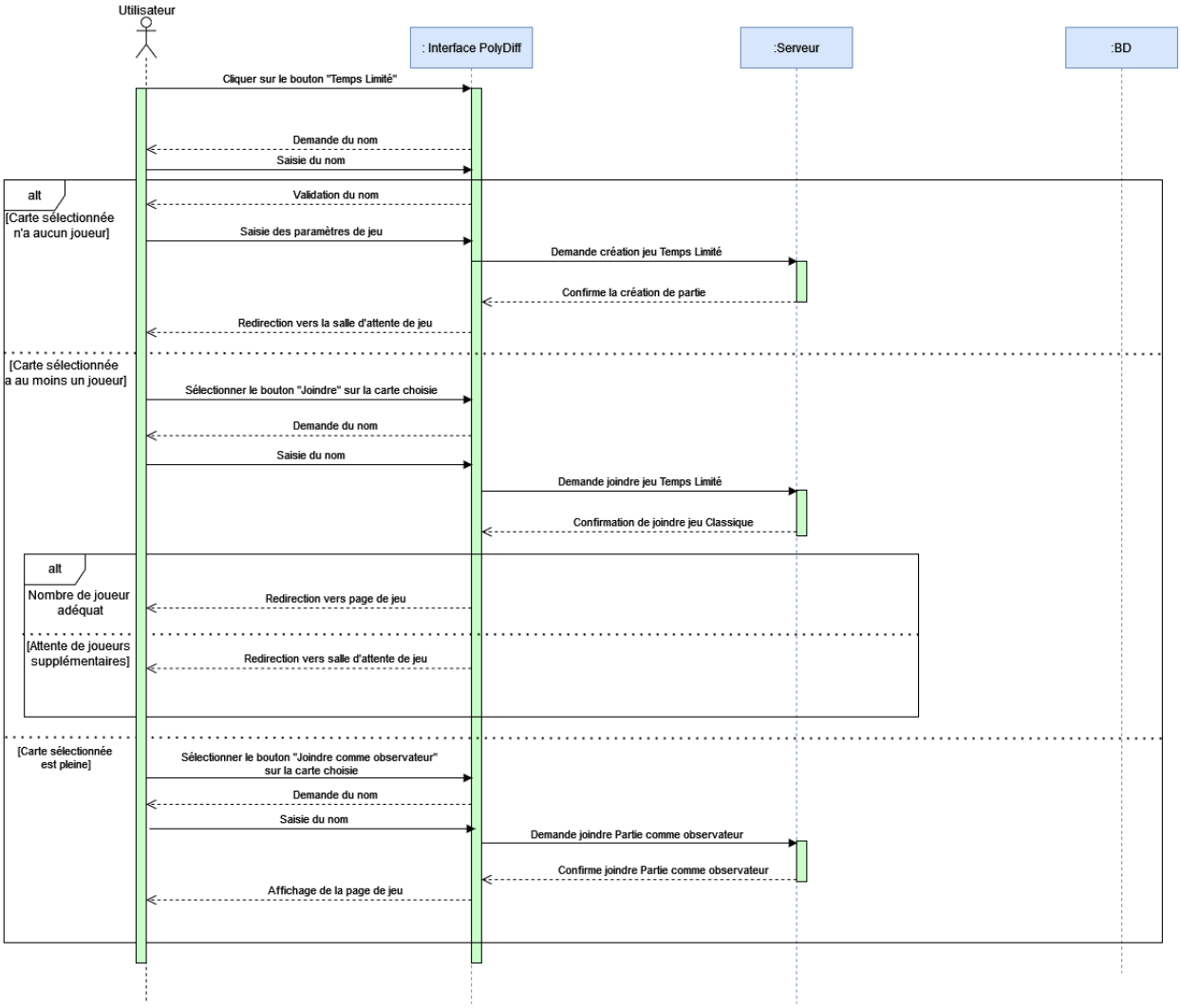
5. Vue des processus



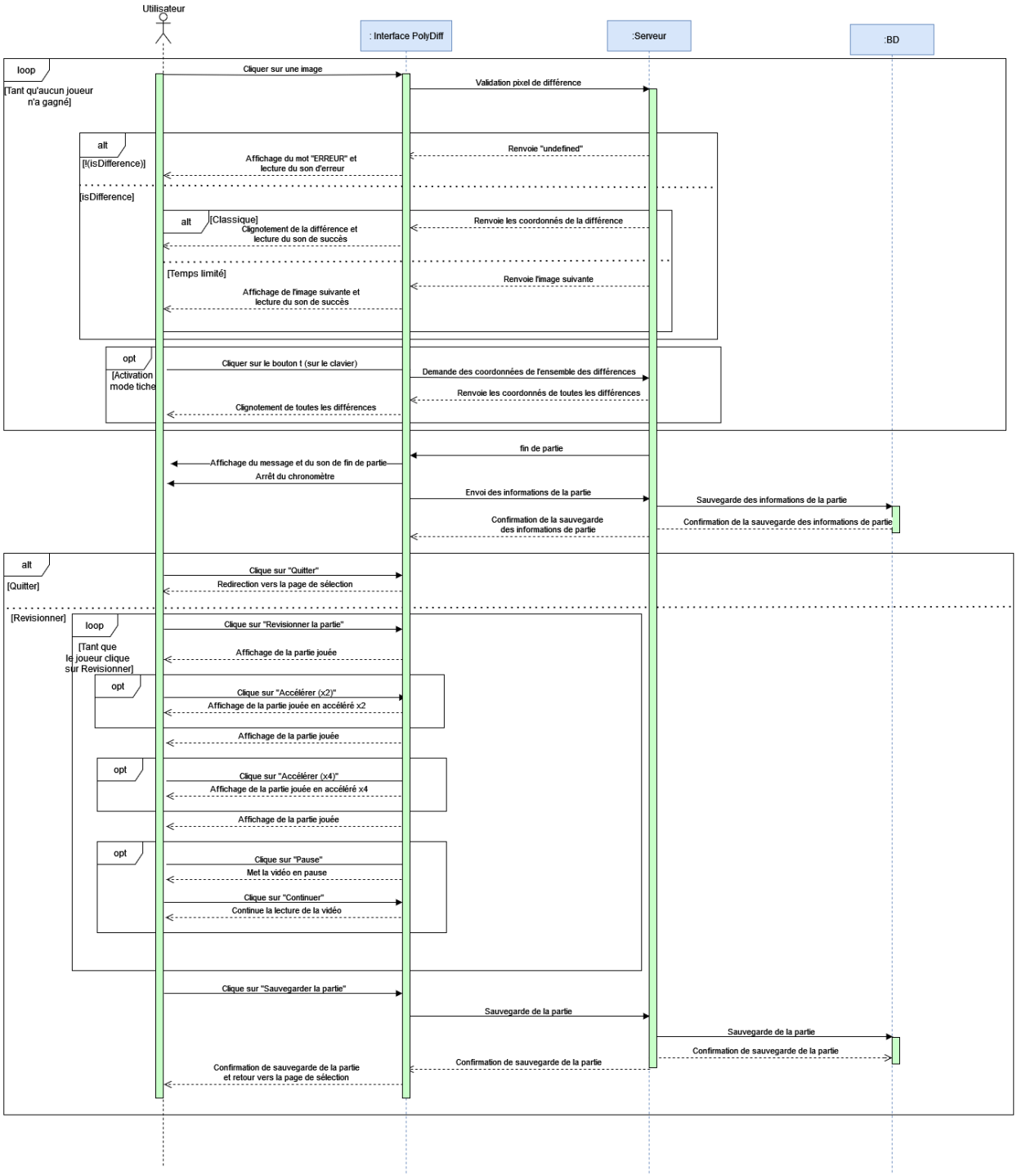
5.1 Diagramme de processus pour l'authentification au système



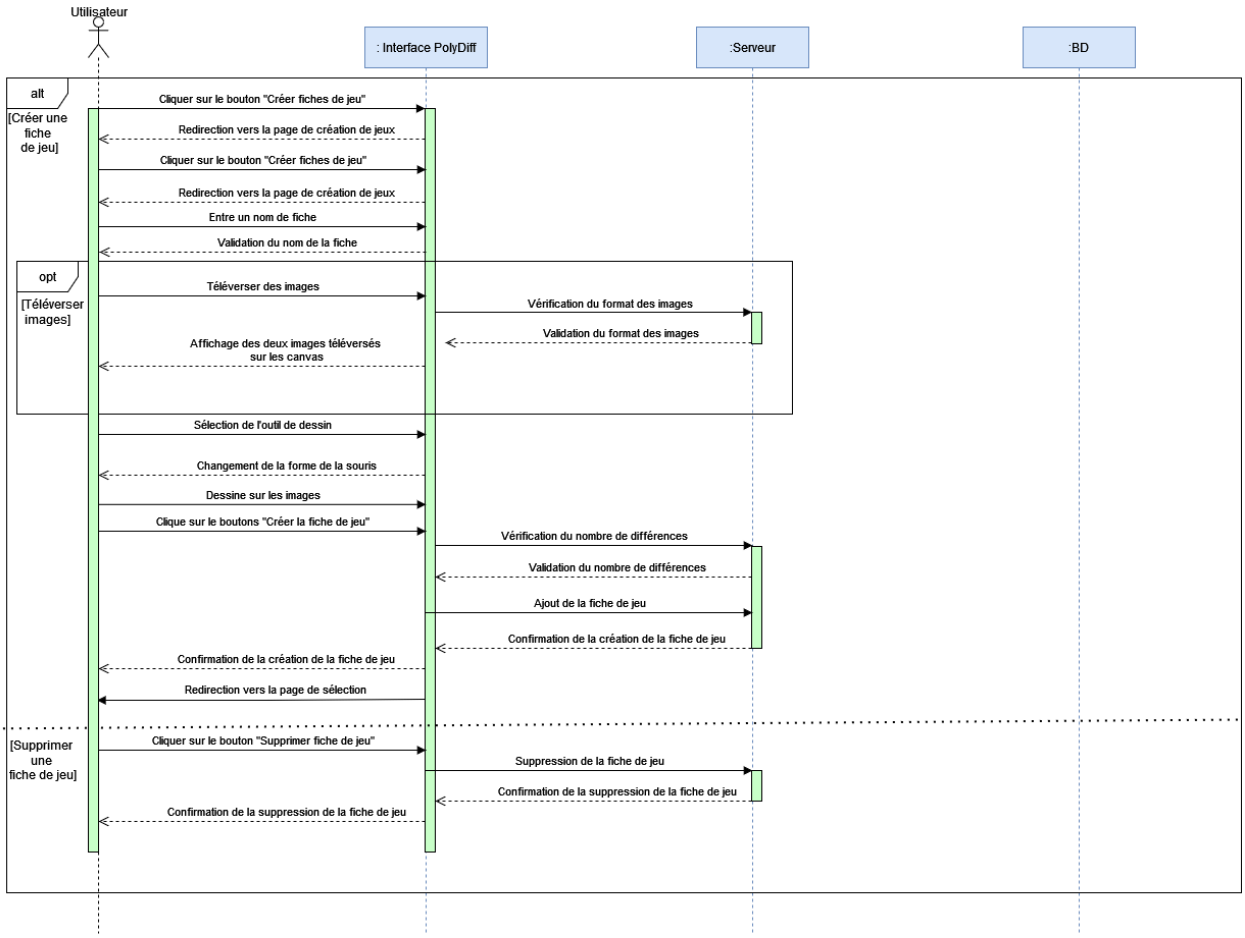
5.2 Diagramme de processus pour la création et la connexion d'une partie classique



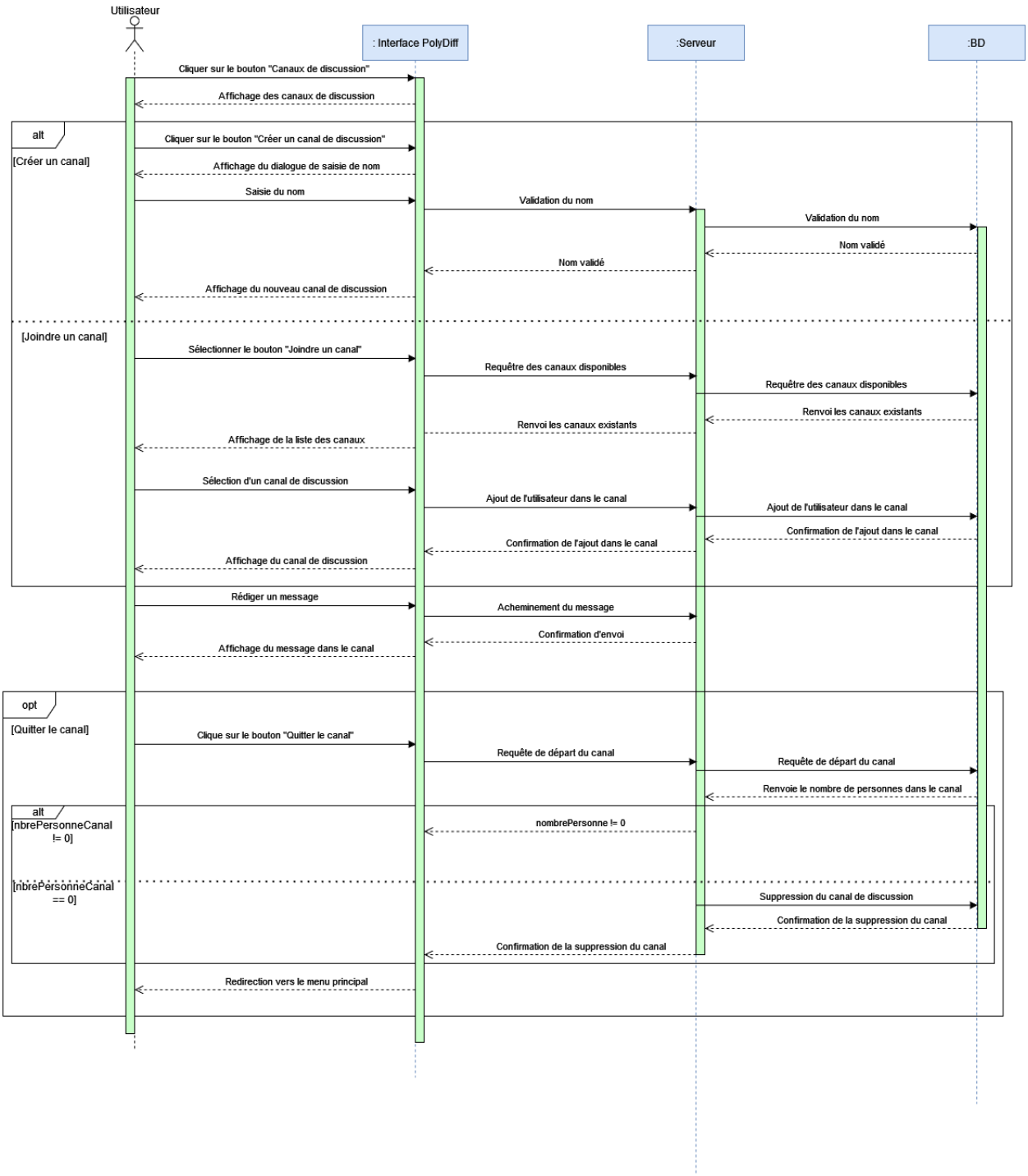
5.3 Diagramme de processus pour la création et la connexion d'une partie temps limité



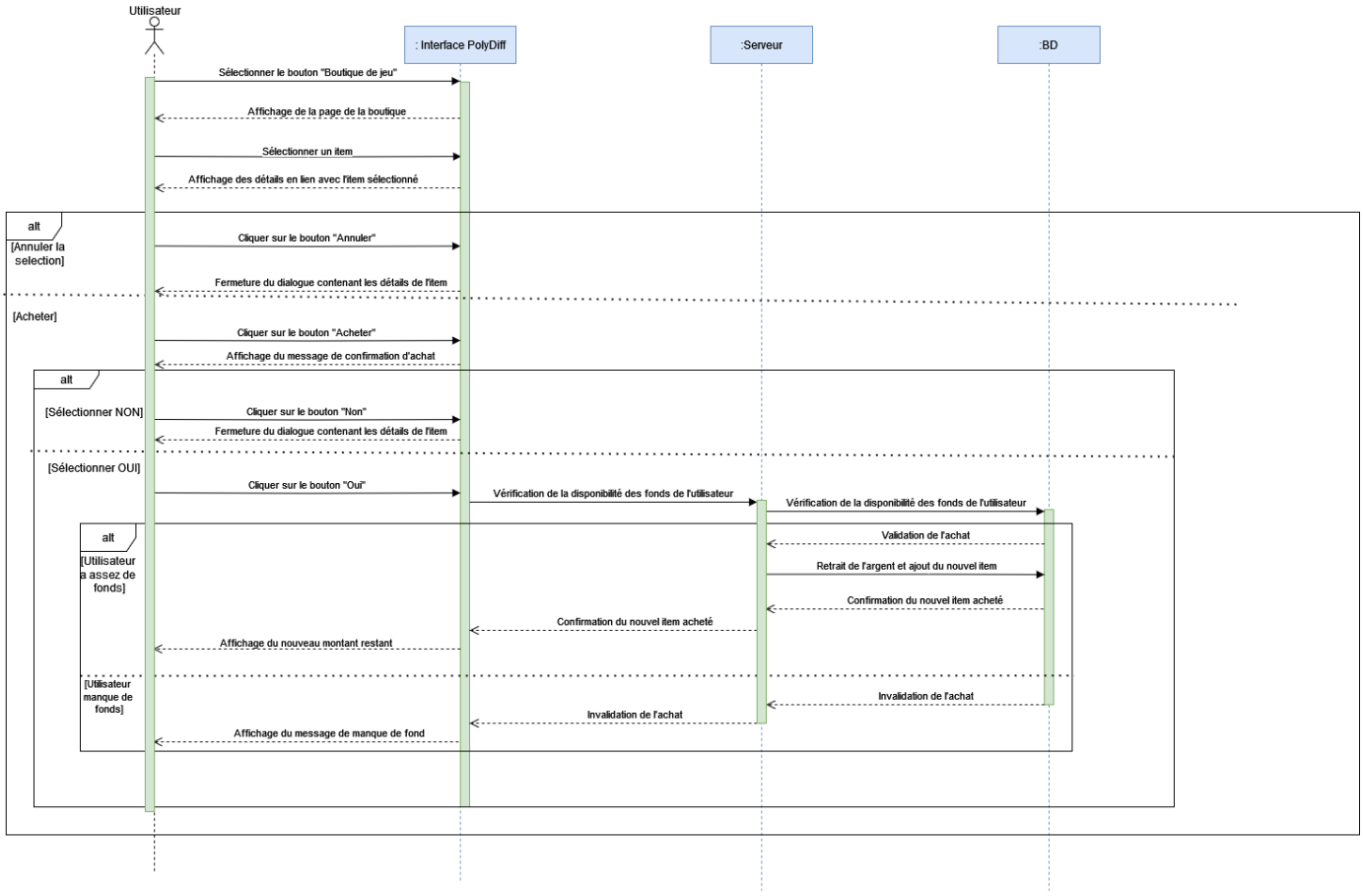
5.4 Diagramme de processus pour le déroulement d'une partie



5.5 Diagramme de processus pour les fonctionnalités du compte administrateur

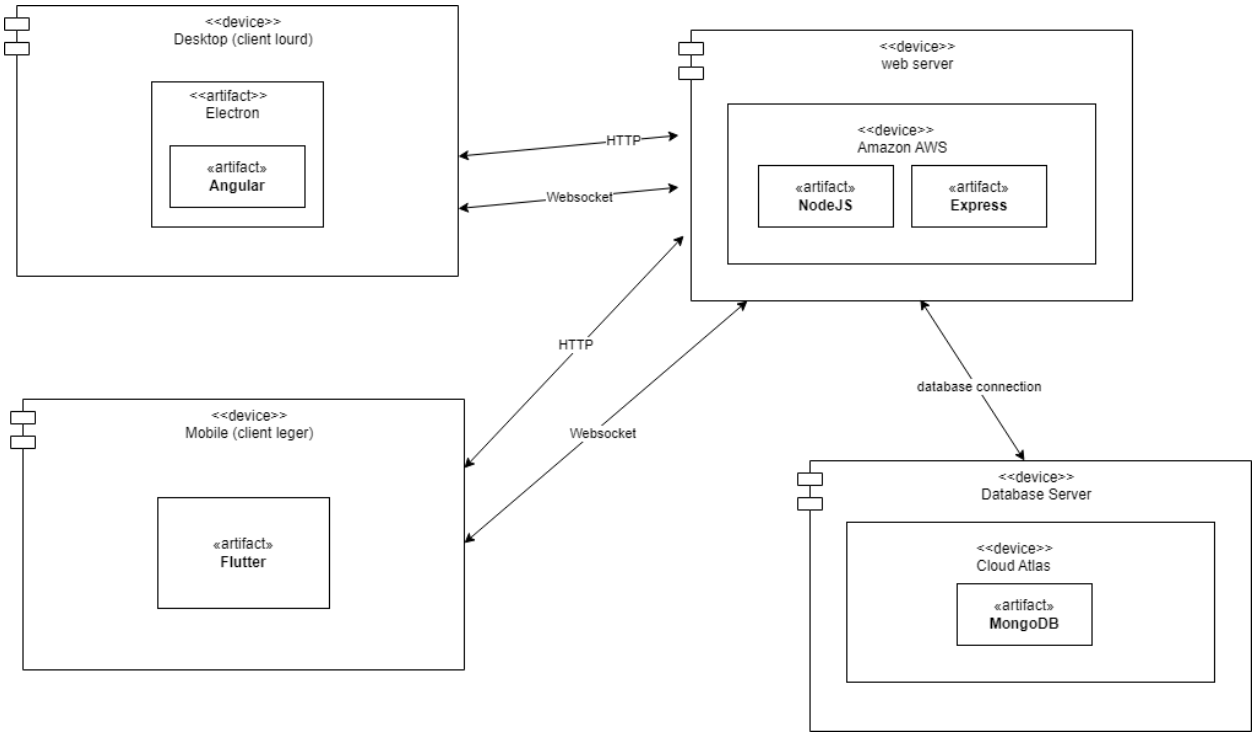


5.6 Diagramme de processus pour la gestion des canaux de discussion



5.7 Diagramme de processus pour la boutique de jeu

6. Vue de déploiement



7. Taille et performance

Il est important d'analyser ces aspects afin de dimensionner adéquatement le système, d'optimiser ses performances et de garantir une expérience utilisateur fluide.

Cela inclut des éléments tels que la quantité de données à traiter, le nombre d'utilisateurs simultanés prévus, les exigences en matière de temps de réponse, et les contraintes de ressources matérielles.