

UFR

de mathématique

et d'informatique

Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

PROJET INTÉGRATEUR - POKERGANG

RAPPORT FINAL



ÉQUIPE 6B - RoundCoders

Table des matières

1	Introduction.....	2
2	Réalisation du projet	2
2.1	Architecture du projet	2
2.2	Points à privilégier	3
2.2.1	Fluidité	3
2.2.2	Ergonomie	3
2.2.3	Sécurité.....	3
2.3	Description du travail effectué	4
2.3.1	Base de données.....	4
2.3.2	Frontend : Site et menu.....	4
2.3.3	Serveur : Mise en place du serveur de base.....	4
2.3.4	Jeu : Implémentation du jeu de base	5
2.3.5	Frontend : Interface du jeu.....	5
2.3.6	Serveur : Mise en place des interactions client / serveur	5
2.3.7	Frontend : Rôles.....	6
2.3.8	Jeu : Rôles	6
2.3.9	Serveur : Rôles.....	6
2.3.10	Débogage.....	7
2.3.11	Configurations des machines virtuels pour jouer en réseau.....	7
2.3.12	Passage du jeu en web en version bureau	7
2.4	Axes d'améliorations.....	7
2.4.1	Lobby (20 heures)	7
2.4.2	Intelligence artificielle (40 heures)	7
2.4.3	Un système d'amis (15 heures).....	8
2.4.4	Nouveaux skins de table et de carte (40 heures)	8
2.4.5	Avatar (5 heures)	8
2.4.6	Evènement avec règles spéciaux (40 heures).....	8
2.4.7	Mode sans rôles (15 heures)	8
2.4.8	Pass Gang (20 heures)	8
3	Organisation de l'équipe.....	8
3.1	Les rôles des membres de l'équipe	8
3.1.1	Yannick LECAM : équipe serveur	9
3.1.2	Johary RALAMBO : équipe frontend.....	9
3.1.3	Corentin STEINER : chef de projet	9

3.1.4	Florian WU : équipe jeu	9
3.1.5	Shixiang XU : équipe serveur	9
3.1.6	Ahmed ZANED : équipe frontend	9
3.1.7	Hayk ZARIKIAN : équipe jeu	10
3.1.8	Ekaterina ZAITCEVA : équipe BDD / jeu / serveur.....	10
3.1.9	Fares ZID : équipe BDD / frontend.....	10
3.2	Rôle du chef de projet	10
3.3	Mode de travail / Outils.....	11
3.3.1	Politique Git	11
3.3.2	Modèle de développement	11
3.3.3	Charte documentaire.....	11
3.3.4	Réunion.....	12
3.3.5	Trello	12
3.3.6	Diagramme de Gantt	12
3.3.7	Serveur discord.....	12
3.3.8	Code Together	12
3.3.9	Timesheet.....	12
3.4	Déroulement	12
3.5	Difficultés et solutions	14
3.6	Implication de chaque membre de l'équipe.....	14
3.6.1	Yannick LECAM	15
3.6.2	Johary RALAMBO.....	15
3.6.3	Corentin STEINER.....	16
3.6.4	Florian WU	17
3.6.5	Shixiang XU	18
3.6.6	Ahmed ZANED	18
3.6.7	Hayk ZARIKIAN.....	19
3.6.8	Ekaterina ZAITCEVA	20
3.6.9	Fares ZID	21
4	FAQ équipe	22
4.1	Comment évaluer-vous l'ensemble du projet ?.....	22
4.2	Êtes-vous satisfait de la qualité du travail accompli par l'équipe ?	22
4.3	Comment évaluer-vous l'organisation et la planification du projet ?.....	22
4.4	Comment décrivez-vous le niveau de communication au sein de l'équipe de projet ?	22
4.5	Avez-vous toutes les ressources nécessaires pour mener à bien votre travail sur ce projet ?	
	22	

4.6	Y'a-t-il des domaines où vous pensez que l'équipe pourrait s'améliorer pour mieux accomplir ses tâches ?	22
4.7	Quels sont les aspects les plus stimulants ou les plus gratifiants de travailler sur ce projet ? 23	
4.8	Quels sont les objectifs futurs de l'équipe pour ce projet ?.....	23
5	Conclusion	23
6	Annexes	23
6.1	Document technique	23
6.2	Document de communication.....	24
6.3	Document Interface Homme-Machine.....	24
6.4	Document utilisateur.....	24
6.5	Politique Git.....	24
6.5.1	Branches	24
6.5.2	Politique de développement	25
6.5.3	Politique de fusion de branches	25
6.6	Diagramme de Gantt	25

PRÉAMBULE

Ce rapport est un document permettant de clôturer notre projet « PokerGang » qui indique plusieurs aspects comme l'organisation, les difficultés, les compétences acquises, les motivations des membres de l'équipe et bien d'autres aspects.

L'organisation du rapport est décrite ci-dessous :



La couleur Jaune représente les parties générales qui englobe tout le projet dans son ensemble.



La couleur bleue représente les parties spécialisées qui englobe la partie générale en jaune correspondant.



La couleur rouge représente les parties les plus spécialisées dans ce document qui englobe la partie spécialisée bleue correspondant.

Listes des membres de l'équipe :

- Yannick LECAM :
- Johary RALAMBO
- Corentin STEINER
- Florian WU
- Shixiang XU
- Ahmed ZANED
- Hayk ZARIKIAN
- Ekaterina ZAITCEVA
- Fares ZID

1 Introduction

Le projet PokerGang est un projet conçu par notre équipe lors de l'UE Projet Intégrateur en 3^{ème} année de licence informatique à l'UFR mathématique et Informatique de l'Université de Strasbourg.

PokerGang est un jeu de Poker, qui a la particularité de proposer un concept original qui est l'ajout de rôles. Les rôles permettent dans le cadre du poker de rendre les parties plus ludiques et décupler les possibilités de bluff dans une partie de poker, le bluff étant l'essence-même du poker. De plus, cet ajout casse les codes du jeu de poker classique car il expose un aspect qui identifie notre jeu comme unique et original.

Dans ce rapport, nous allons présenter l'ensemble de notre travail qui s'est déroulé durant 4 mois, que ce soit sur le plan technique ou au niveau organisationnel.

2 Réalisation du projet

2.1 Architecture du projet

Le projet PokerGang est un jeu de sur table en ligne qui est jouable de 2 à 6 joueurs directement sur le navigateur web ou bien sur le bureau.

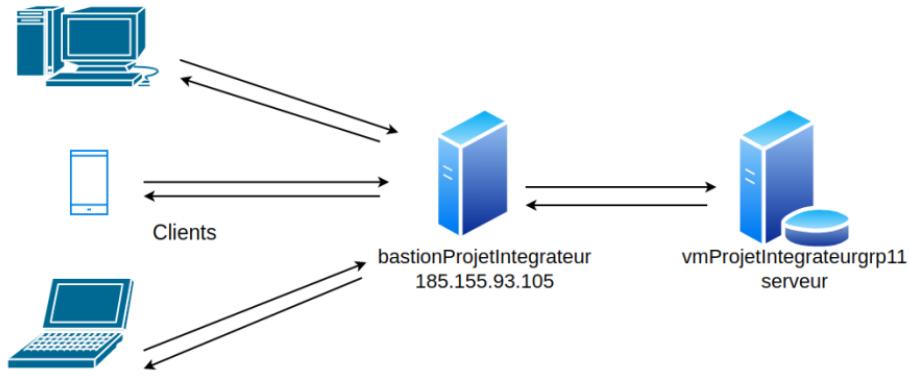
Le projet s'est construit autour de 4 parties majeures qui ont composées les équipes au sein du projet : le frontend, le réseau, la base de données et le jeu.

Le langage Javascript a été choisi comme langage principal car le jeu a été initialement développé en web, puis par la suite en version bureau en se servant du framework « ElectronJs » pour n'avoir qu'un seul code mais plusieurs plateformes. De plus, Javascript possède un grand nombre de framework, dont certains ont été utilisés lors de ce projet et a une grande popularité, ce qui fait qu'il est bien documenté et que des solutions pour des problèmes d'implémentation pour ce langage peuvent être facilement trouvées.

La partie jeu, du fait qu'elle constitue le jeu en lui-même, est implémentée uniquement avec du Javascript.

Le frontend a utilisé le framework « VueJs » car il apportait une fluidité, une bonne réactivité et il reste très accessible.

Pour la partie réseau, nous avons utilisés « NodeJs » et « Express » pour faire le serveur du jeu et faciliter le développement d'une application web. Enfin, pour améliorer les échanges, nous nous sommes servis de Socket.io, ce qui est essentiel au vu de notre configuration serveur / client. La base de données se basait sur SQLite car rapide et simple à intégrer à un programme.



2.2 Points à privilégier

2.2.1 Fluidité

« ElectronJs » est réputé comment étant plutôt lent, un des aspects du projet à privilégier était donc la fluidité du jeu. Ainsi nous obtenons un résultat un jeu fluide qui ne souffre d'aucun ralentissement ou autre problème de cet acabit.

2.2.2 Ergonomie

Du fait qu'une évaluation de l'IHM avait lieu sur notre projet, nous avons donc en ce sens mis un accent sur l'ergonomie de l'application. Nous avons donc cherché à optimiser l'expérience utilisateur, en accordant une attention particulière à des éléments tels que la mise en place des jetons misés, l'ordre de priorité des boutons dans l'interface utilisateur et une navigation simple et rapide.

2.2.3 Sécurité

La sécurité était également une priorité pour le projet, les équipes ont travaillé à garantir la sécurité des données, en utilisant des techniques d'authentification sécurisées et en transmettant les données sensibles via des connexions sécurisées telles que HTTPS. Dans la situation actuelle le TLS est fait avec un certificat auto-signé. Si le projet était déployé sur un vrai serveur et pas sur la VM de la faculté, on aurait mis un certificat signé par une autorité de certification vérifiée. Les échanges de données via des sockets personnalisés ont également été utilisés pour renforcer la sécurité du projet et éviter des problèmes de triches. Les mots de passe sont aussi hachés dans la base de données, les inputs client sont vérifiés.

2.3 Description du travail effectué

2.3.1 Base de données

Ekaterina ZAITCEVA et Fares ZID se sont initialement occupé de cette partie, ils l'ont conçu les schémas E/A, puis ils ont implémenté les tables et les contraintes. Ils se sont ensuite séparés pour rejoindre d'autres équipes pour pouvoir faire la liaison avec la BDD. Ils ont plus tard rajouté d'autres fonctions en lien avec la BDD dans les différents groupes qu'ils ont rejoints par la suite.

La base de données est opérationnelle, elle a pour fonction d'enregistrer les comptes d'utilisateurs, comme leurs pseudo, mail, mot de passe et nombre de jetons.

Temps : 60 heures

2.3.2 Frontend : Site et menu

Ahmed ZANED et Johary RALAMBO se sont occupés du frontend de l'application. Ils ont d'abord créé les wireframes du site, le design de l'interface et la maquette graphique.

Ils ont ensuite implémenté le site qui accueillerait le jeu. Initialement conçu en HTML et CSS, il est apparu plus tard que le site devait finalement être en VueJS pour obtenir de meilleure performance et une meilleure compatibilité avec le serveur, ce qui a été fait par la suite avec l'aide de Yannick LE CAM de l'équipe Réseau.

Fares ZID a ensuite rejoint l'équipe frontend, il a participé avec les deux autres membres à la traduction en anglais du site. Avec Johary, ils ont conçu la page sur le classement des joueurs.

Le site fait office d'interface pour la création de compte, la connexion, avoir accès au différentes pages et fonctionnalité et entrer dans le jeu.

Temps : 65 heures

2.3.3 Serveur : Mise en place du serveur de base

Yannick LE CAM et Shixiang XU ont implémentée la base du serveur, en se basant sur les différents modules utilisés, puis implémenté les fonctions pour se connecter à la BDD et pouvoir récupérer les informations du client. Ils ont aussi implémenté les fonctions pour les inscriptions et les connections sur le site.

Plus tard, Hayk Zarikian a développé un script pour envoyer des mails automatiquement et qui a été intégré par la suite par l'équipe serveur.

Le serveur de base est là pour gérer tout ce qui touche au site en backend.

Temps : 60 heures

2.3.4 Jeu : Implémentation du jeu de base

Hayk ZARIKIAN a d'abord réalisé le diagramme du jeu. Après cela il a implémenté le squelette de base. Après cela ils se sont répartis les différentes fonctions chacun de leurs côtés.

Hayk a implémenté les différentes fonctions qui lui étaient attribuées, il a aussi effectué les tests unitaires sur son code. Ekaterina ZAITCEVA a par la suite rejoint l'équipe Jeu et pu faire le lien entre la partie Jeu et la partie BDD.

Il est apparu vers la fin de la période de cette tâche dans le Gantt (Implémentations des fonctions), que Florian avait menti sur l'avancement et la qualité de ses fonctions, ce qui a engendré du retard et du travail supplémentaire pour les autres membres de son équipe, qui ont dû refaire les fonctions dont il avait la charge.

Cette partie correspond au jeu et permet donc d'avoir le jeu de poker de base sans les rôles.

Temps : 60 heures

2.3.5 Frontend : Interface du jeu

Johary RALAMBO a implémenté l'interface représentant la table du Jeu, puis avec Fares ZID et Ahmed Zaned, ils ont implémenté les animations. Johary et Fares ont par la suite amélioré l'interface notamment sur les boutons.

La suite de cette partie est une recherche d'amélioration constante pour garantir le meilleur résultat et la meilleure ergonomie possible, comme rajouter des options, des boutons, mettre en place un timer, mettre un chat en place ainsi qu'une liste avec les infos de la partie en temps réel.

L'interface du jeu constitue l'élément principal du front, c'est la table de poker. C'est une partie qui a demandé beaucoup d'effort du fait que cela constitue la face la plus visible de notre projet.

Temps : 135 heures

2.3.6 Serveur : Mise en place des interactions client / serveur

Après avoir relié le serveur à la partie site du projet, il fallait faire le lien avec l'interface du jeu et le jeu lui-même via le serveur. Yannick LE CAM et Shixiang XU ont donc implémenté un algorithme pour la connexion des clients à la table permettant un affichage efficace pour la partie front et des modifications rapides et claires côté backend.

Shixiang a fait ensuite les requêtes AJAX pour les récupérations de données, il a aussi fait les requêtes nécessaires avec la BDD pour le classement et le Log

out, ainsi que le squelette des algorithmes majeur, il a aussi conçu l'algorithme pour permettre aux utilisateurs qui arrivent dans le jeu de transmettre ses données, ou quitter et rejoindre la table. Il a aussi implémenté un mode spectateur.

Yannick a fait le lien entre la nouvelle version du site (en « VueJS ») via Node, il a aussi écrit un script pour installer la bonne version de Node et des composantes sur chaque ordinateur. Il a aussi plus tard ajouté un moyen de rester connecté si la fenêtre se ferme et qu'on revient sur le jeu.

Durant l'assemblage, avec l'aide de l'équipe Front et Jeu, Yannick s'est occupé de tout ce qui permettait de lier les différentes parties, avec l'aide de Ahmed et Johary pour le front, Hayk pour le Jeu et Ekaterina pour la BDD et le Jeu.

Cette partie correspond à la suite de la « Mise en place du serveur de base » avec des ajouts de fonctionnalité mais elles sont plus axées sur la mise en commun entre les différentes parties.

Temps : 200 heures

2.3.7 Frontend : Rôles

Après avoir fait l'interface pour le jeu, il fallait aussi faire la partie qui correspondait aux rôles, pour cela Ahmed ZANED, Johary RALAMBO et Fares ZID ont fait les animations et les modifications nécessaire de l'interface. Après cela, Ahmed et Johary ont participé aux assemblages pour intégrer leurs parties aux autres.

Temps : 40 heures

2.3.8 Jeu : Rôles

Ayant de l'avance dans sa partie, Hayk ZARIKIAN a pu profiter de cela pour pouvoir concevoir et implémenter en avance. Il a donc pu implémenter le squelette pour les classes, puis avec cela les 3 premiers rôles, avec les tests unitaires qui vont avec. Ekaterina ZAITCEVA en a implémenté un 4eme.

Ils ont plus tard été modifié par le Yannick et le Johary pour pouvoir être intégré au reste du projet. Les 2 autres avait besoin de la partie Frontend pour être implémentée, qui n'était pas encore prête, vu que cela a eu lieu bien en avance par rapport à ce qui était prévu.

Ils ont donc été implémentée par Ekaterina, Yannick et Johary pendant l'assemblage des rôles.

Temps : 15 heures

2.3.9 Serveur : Rôles

Avec l'aide de Ekaterina et Johary, Yannick LE CAM a effectué l'assemblage des rôles, ils ont donc comme dit précédemment les modifications. Ils ont aussi implémenté l'algorithme de l'attribution des rôles aux joueurs.

Temps : 30 heures

2.3.10 Débogage

Les résultats issus des assemblages avaient de nombreux bugs, cette partie ne correspond pas à une partie du projet en elle-même, mais le débogage constitue une partie non-négligeable du travail effectué durant de projet. La majorité des membres de l'équipe ont participé à rectifier les problèmes du code. Ekaterina ZAITCEVA a effectué énormément de tests en jouant au jeu pour détecter les problèmes ou tester les tentatives de débogages. Yannick LE CAM et Johary Ralambo, Ahmed ZANED et Ekaterina ont été les principaux membres qui ont le plus débogué. Hayk ZARIKIAN, Shixiang Xu et Fars ZID ont aussi été aidés à résoudre certains problèmes.

2.3.11 Configurations des machines virtuels pour jouer en réseau

Afin de pouvoir jouer en réseau, nous avions besoin d'utiliser les machines virtuelles qui étaient à notre disposition. Ekaterina ZAITCEVA s'est donc occupée de configurer cette partie et d'y ajouter le code du jeu. C'est elle tous les aspects liés au VM.

2.3.12 Passage du jeu en web en version bureau

Afin d'obtenir un jeu « en version lourde », nous nous sommes basés sur « ElectronJS ». Ekaterina ZAITCEVA a donc effectué les configurations Electron nécessaire pour pouvoir y jouer sur Linux. Plus tard, avec l'aide de Johary RALAMBO, une configuration a été faite pour Windows.

Ahmed ZANED avait aussi tenté de faire une configuration « ElectronJS » de son côté, mais n'était parvenu à rien de concluant.

2.4 Axes d'améliorations

2.4.1 Lobby (20 heures)

Il était prévu, si nous avions le temps, de créer un lobby pour avoir différentes tables de poker jouables en même temps afin que plus de 6 personnes puissent jouer, mais cela aurait nécessité de réorganiser une partie du code serveur.

2.4.2 Intelligence artificielle (40 heures)

Florian WU avait débuté du code pour faire une IA afin de pouvoir avoir des bots pour remplir les tables. Il avait commencé, mais il a sollicité l'aide du reste du groupe bien trop tard avant le rendu pour faire un assemblage.

2.4.3 Un système d'amis (15 heures)

Un système d'amis pourrait être ajouté afin de pouvoir rajouter des interactions entre joueurs, pour pouvoir permettre de jouer avec des gens qu'on apprécie.

2.4.4 Nouveaux skins de table et de carte (40 heures)

L'ajout de skins de table et de cartes pour le poker pourrait offrir une expérience de jeu plus personnalisée pour les joueurs. En proposant une variété de skins, les joueurs pourraient choisir un thème qui correspond à leur style personnel.

2.4.5 Avatar (5 heures)

On pourrait ajouter une fonction pour avoir un avatar personnalisé afin de pouvoir choisir son image de profil.

2.4.6 Evènement avec règles spéciaux (40 heures)

L'ajout d'événements spéciaux, tels que les modes de jeux spéciaux ou des tournois, pourrait améliorer l'expérience de jeu et fidéliser des joueurs.

2.4.7 Mode sans rôles (15 heures)

Malgré l'ajout des rôles, certains veulent peut-être de temps en temps faire une partie de poker classique, il serait donc possible d'implémenter un choix pour un mode sans les rôles

2.4.8 Pass Gang (20 heures)

Pass Gang est un système de récompenses où les joueurs peuvent débloquer des assets (avatar, skin de carte, ...) en accumulant des victoires. Cela peut augmenter la motivation et l'engagement des joueurs en leur donnant un but à long terme à atteindre et des objectifs.

3 Organisation de l'équipe

3.1 Les rôles des membres de l'équipe

3.1.1 Yannick LECAM : équipe serveur

Il a développé des fonctions pour le serveur mais il était surtout orienté dans les assemblages entre les différentes parties ce qui l'a amené à être en communication constante avec les autres groupes et être au centre du projet. De fait, il gérait aussi les merges entre les différentes branches et était aussi présent au cœur des débogages.

3.1.2 Johary RALAMBO : équipe frontend

Il a commencé par créer le wireframe et la maquette puis implémenter les pages de base en HTML et CSS. Ensuite, il a intégré « VueJS » ajouté un système de traduction et commencé à développer l'interface, il a également travaillé sur les animations de certains rôles du jeu. Il a également participé aux différents assemblages et aidé à résoudre de nombreux problèmes dû au bug.

3.1.3 Corentin STEINER : chef de projet

S'occupe de l'organisation globale du projet, définit les priorités, supervise les sprints et s'adapte selon les contraintes si nécessaire afin de garantir les délais fixés pour chaque étape du projet. Il a également géré la communication dans le groupe afin que tout le monde soit informé de l'avancement global du projet, il maintient la cohésion du groupe et tranche les dilemmes si besoin, il doit aussi rester à l'écoute et se remettre en question si nécessaire. A rédigé les comptes rendus de réunion et hebdomadaire et à participer et superviser la rédaction du cahier des charges et des rapports et la documentation.

3.1.4 Florian WU : équipe jeu

Initialement, il devait implémenter les fonctions du jeu de base mais devant le fait qu'il ne rendait pas son travail dans les temps et qu'il mentait dans son avancement et la qualité de son travail, il lui a été demandé d'implémenter une IA qui joue au jeu pour avoir à terme des bots.

3.1.5 Shixiang XU : équipe serveur

Membre de l'équipe serveur il était de son côté plus axée sur le développement de fonction ou bien des requêtes pour le serveur, il a aussi implémenté d'autres fonctionnalités en plus comme le mode spectateur.

3.1.6 Ahmed ZANED : équipe frontend

Il a conçu les wireframes de l'application et a travaillé sur toutes les pages du site. Il a notamment travaillé sur les pages de connexion et d'inscription, l'intégration de « VueJs », la mise en place d'un système de traduction, ainsi que l'interface utilisateur en effectuant de nombreuses modifications pour la rendre simple et compréhensible pour les utilisateurs. Il a contribué à l'assemblage du projet et a aidé à déboguer.

3.1.7 Hayk ZARIKIAN : équipe jeu

Il s'est occupé de la conception de la partie jeu de base puis plus tard celle des rôles, il a par la suite implémenté ces différentes parties et aidés à les assembler aux restes du projet et à déboguer. Il a aussi généré beaucoup de documentations, dont certaines ont servi pour la rédaction du cahier des charges et certains rapports. Il a rédigé une certaine partie du rapport IHM, a géré la communication et a mis en page le cahier des charges et les rapports.

3.1.8 Ekaterina ZAITCEVA : équipe BDD / jeu / serveur

Elle s'est vu devoir travailler sur plusieurs aspects du projet, notamment la conception et l'implémentation de la base de données avec son binôme. Après avoir terminé son travail dans l'équipe BDD, elle a rejoint l'équipe jeu pour lier le jeu à la base de données et écrire la fonction principale du jeu. Elle a également géré la VM pour le serveur, transformé la version web en version desktop avec « ElectronJs » et effectué des tests. Enfin, elle a participé à des sessions de débogage pour aider l'équipe à résoudre les problèmes techniques rencontrés.

3.1.9 Fares ZID : équipe BDD / frontend

Au départ, il a travaillé sur la base de données avec son binôme pour concevoir et mettre en place les fonctions nécessaires à la manipulation de la base de données. Par la suite, il a rejoint l'équipe frontend pour aider à la conception des pages du jeu et des animations en y ajoutant ses compétences en production musical pour créer des bandes sonores pour le jeu et l'accueil. Il a aussi aidé son équipe à déboguer sa partie.

3.2 Rôle du chef de projet

Le projet a été découpé en différentes phases allant de 2 à 4 semaines : à l'exception de la première, une phase a 2 semaines d'implémentations et 2 semaines d'assemblages qui font office de jalons, c'est-à-dire la mise en commun des codes avec les différentes équipes afin de finaliser une étape et de passer à la suivante. Ces différentes phases faisaient alors office de sprint, des objectifs étaient alors fixés sur 1 ou 2 semaines selon les cas, par une discussion avec le chef de projet et les différents membres du groupe. En cas de problème ou de retard, les objectifs et les délais étaient redéfinis pour respecter la date finale de l'étape.

A chaque début de séance, le chef de projet passait voir les différents membres du groupe, pour se renseigner sur l'avancement et les éventuelles difficultés. Durant certaines séances, un stand-up meeting était organisé ou alors des entretiens individuels entre le chef de projet étaient mis en place afin de faciliter la communication et de pouvoir avoir un autre point de vue sur l'état du groupe ou de détecter des problèmes qui n'auraient pas pu être identifié autrement ou alors pour pouvoir remettre en question la méthodologie employée.

Lorsque certains problèmes étaient identifiés, le chef de projet cherchait à les résoudre via des nouveaux outils ou en définissant des procédés.

Du fait qu'il y ait eu des problèmes de merge ou d'organisations sur git, une politique git a été mise en place pour les procédures à suivre ou l'utilisation des branches.

Un autre problème pouvait être de trouver du travail pour ceux qui n'avaient pas de tâches : pour répondre à cela, une backlog fut mise en place en se servant de Trello. Elle fut utile au début mais peu à peu abandonnée par la suite. Si jamais un problème de travail se faisait pressentir, le chef de projet les incitait à se référer à lui : soit il se renseignait avec le reste du groupe si une tâche était nécessaire ailleurs, ou bien il leur assignait directement un travail.

3.3 Mode de travail / Outils

3.3.1 Politique Git

Pour le bon déroulement de notre dépôt git, nous avons mis en place une politique git qui indique la procédure à suivre afin de push le travail effectué sans créer des conflits avec les membres de l'équipe. L'un des principaux outils utilisés afin de pouvoir déposer le code et le mettre en commun. Se référer à la partie [6.5 Politique Git](#).

3.3.2 Modèle de développement

Au début du projet nous avions défini un modèle de développement itératif « Exploratoire » qui a été respecté avec succès car tout au long du projet nous avons amélioré le projet suite à plusieurs itérations tout en explorant les différentes technologies en jugeant leur efficacité afin d'opter la meilleure stratégie.

3.3.3 Charte documentaire

Nous avons également défini une charte documentaire afin que les documents fournis suivent une cohérence graphique et de forme. Par exemple, dans nos documents nous avons toujours 3 parties au maximum qui peuvent être imbriquées, la taille de la police utilisée doit être 11 ou encore la police utilisée doit être « Calibri (Corps) » et bien d'autres aspects.

3.3.4 Réunion

Nous avons élaboré de nombreuses réunions tout au long du projet afin de faire un compte rendu du travail effectué la semaine et de ce qu'il faut prévoir de faire dans les prochaines semaines. Les réunions ont été réalisées en présentiel comme en distanciel à l'aide de discord ou bien de réservation de salle dans les bibliothèques universitaire ou encore pendant l'heure de projet intégrateur dédiée.

3.3.5 Trello

Trello est un outil qui nous a permis de gérer la gestion des tâches tout en suivant l'avancement du projet pas à pas en complément du diagramme de Gantt.

3.3.6 Diagramme de Gantt

Le diagramme de Gantt nous a permis d'évaluer les tâches de l'ensemble du projet en spécifiant leur priorité absolue et du temps disposé pour la terminaison du projet. Lors d'un retard, il nous a permis d'agir en conséquence en prenant compte le diagramme de Gantt et des priorités. L'outil qui a plutôt été utilisé par le chef de projet pour pouvoir indiquer les différentes étapes du projet et les jalons et se situer par rapport à l'avancement du projet. Se référer à la section [6.6 Diagramme de Gantt](#).

3.3.7 Serveur discord

Discord a été l'outil principal de communication pour le groupe, ainsi que notre outil le plus utilisé pour les réunions.

3.3.8 Code Together

Code together est un outil permettant à plusieurs développeurs de pouvoir coder en même temps sur le même fichier, il a beaucoup été utilisé en réunion.

3.3.9 Timesheet

Une timesheet a été mis en place pour pouvoir rapporter les différentes tâches faites par les membres du groupe et les heures associés.

3.4 Déroulement

Le projet a été découpé en plusieurs phases, la 1^{ère} tranche de 2 semaines avait pour but de laisser aux différentes équipes le temps de pouvoir monter en niveaux, car nous ne connaissons pas certains langages et frameworks utilisées.

Cette phase avait aussi pour but de permettre aux équipes de faire la spécification et ainsi pouvoir faire les différentes conceptions du jeu, des schémas E/A de la base de données, la maquette graphique du front, ...

Après cela, la 2^{ème} phase de 4 semaines correspondait à l'implémentation du site qui constituerait l'application. Elle a vu l'implémentation de la base de données, du front et la mise en place du serveur de base.

L'assemblage s'est fait sans problèmes durant la 1^{ère} semaine qui était dédié. Cependant, il est apparu la semaine suivante, que le site fait en HTML / CSS devait en fait être implémentée en « VueJS » afin de s'assurer que l'application soit optimale, « ElectronJS » étant réputé pour être lent. La 2^{ème} semaine de l'assemblage ainsi que la 1^{ère} de la phase suivante, cela n'a cependant pas affecté les autres tâches en cours.

La phase 3 avait comme objectif de se finir avec un jeu de poker basique sans les rôles. Comme l'équipe BDD avait terminé d'implémenter de son côté. Ekaterina ZAITCEVA a rejoint l'équipe jeu et Fares ZID a de son côté rejoint l'équipe frontend. L'équipe jeu implémentait le code du jeu depuis la fin de la phase 1 (n'étant pas concerné par la phase 2), Hayk ZARIKIAN ayant fini par avance, il avait commencé l'implémentation de certains rôles en avance. Le frontend a lui de son côté implémentée l'interface représentant la table et enfin la partie serveur implémentait les fonctions nécessaires à la connexion à la table et pour récupérer et envoyer les données nécessaires aux différentes parties.

Une semaine avant le début de l'assemblage, une revue de code était prévue au sein de l'équipe jeu, cependant Florian qui avançait avoir finis ses fonctions et qu'il ne manquait plus que quelques tests, n'avait pas push son code sur git, malgré les demandes répétées, il aura finalement avoué avoir menti dans son avancement, et après un ultimatum, il a finalement push son code sur git sur une autre branche. Certaines fonctions n'était pas implémentée et d'autre n'était pas assez optimisé.

La 1^{ère} semaine de l'assemblage de cette phase a donc vu le reste de l'équipe jeu implémentée le code attribué à Florian, tandis que le frontend et le serveur ont faits leurs assemblages de leurs côtés en attendant. Les fonctions du jeu manquante implémentée, nous avons pu réaliser le reste de l'assemblage. Le poker de base était donc réalisé, mais nous avons eu beaucoup de bug et moins de temps que prévu pour pouvoir les traités.

Nous commençons donc la 4^{ème} phase en cherchant d'abord à rendre le jeu plus stable possible. Ainsi les deux premières semaines prévues pour l'implémentation des rôles se sont basés essentiellement là-dessus. En parallèle, Ekaterina ZAITCEVA a configuré les VMs pour pouvoir jouer en réseau sur différents ordinateurs en même temps et pouvoir effectuer des tests, l'équipe frontend à quant à elle implémenté au fur et à mesure pour le jeu nécessitant l'assistance du frontend pour être réalisé n'ont pas pu être implémentée dans cette période. L'équipe serveur elle s'est principalement exclusivement concentré sur le débogage.

Arrivé à la présupposé 1^{ère} semaine d'assemblage pour les rôles, le jeu était encore bugué mais nettement plus stable. A la fin de cette semaine, les animations avaient quant à elle était terminé, nous pouvions donc assembler les rôles. Il est à noter que durant ces 3 semaines précédentes, beaucoup d'examens et de projets à rendre ont eu lieu, ce qui a pu accentuer notre retard initial dû aux bugs.

La 2^{ème} semaine d'assemblage de la phase 4 ont eu lieu durant les vacances en avril, ce qui nous a permis grâce à l'effort de l'équipe de pouvoir organiser beaucoup de réunion de travail sur des longues durées et ainsi pouvoir débuguer le jeu, implémenté les derniers rôles et tous assemblé et pouvoir ensuite mettre le jeu en version bureau via l'utilisation de « ElectronJS », uniquement sur Linux d'abord puis sur Windows.

La dernière semaine avant le rendu qui faisait office de marge si nécessaire, a donc permis de pouvoir retirer les derniers bugs du jeu, implémentée certaines autres fonctionnalités qui ont pu être développé en parallèle (mode spectateur et script pour envoi de mail) ainsi que la rédaction des rapports.

3.5 Difficultés et solutions

Dans la réalisation de notre projet, nous nous sommes confrontés à quelques difficultés sur le plan technique et organisationnel.

- ❖ Problème de merge : Il y a pu avoir quelque fois des problèmes de merge lorsque le contenu de plusieurs branches était rassemblé, pour résoudre ça, le chef de projet a mis en place une politique git afin de définir les procédures adopté pour éviter les problèmes sur git.
- ❖ Travail étudiant : Un certain nombre d'entre nous ont un travail à côté des études, ce qui pose problème pour pouvoir organiser des réunions, la solution était alors de faire des réunions sur des horaires adaptés, généralement le soir.
- ❖ L'un d'entre nous avait des difficultés avec la langue française, ce qui pouvait rendre par moment la communication difficile. Il fallait donc prendre le temps d'expliquer les choses ou bien communiquer en anglais.
- ❖ Le fait que Florian ait menti sur l'avancement et la qualité de son travail a pu rompre la confiance que certains du groupe avait en lui. Avançant que des problèmes était dû au fait qu'il codait moins vite que les autres, notre tuteur a avancé qu'il devrait donc travailler plus que les autres, il a été convenu qu'il devait atteindre 261,5 heures (soit $150 * 1,75$). Afin qu'il puisse avoir une contribution au projet, il lui a été confié la mission d'implémenter une IA. Ce serait une tâche qui lui aurait permis de faire des heures, d'avoir une contribution au projet et ce ne serait pas contraignant pour les autres et le déroulement du projet. Il n'a cependant pas fait autant d'heure qu'on attendait de lui.

3.6 Implication de chaque membre de l'équipe

3.6.1 Yannick LECAM

Durant notre projet intégrateur, nous avons eu l'idée de concevoir un jeu similaire au poker, et pour le rendre encore plus captivant, nous avons mis en place un système de rôles. Ma tâche principale était de travailler sur la partie réseau, pour laquelle j'ai acquis de nouvelles compétences en « NodeJS » (ayant déjà manipulé du Javascript (côté frontend)).

Dans un premier temps, à l'occasion d'une montée en compétence j'ai pu élaborer un projet qui nous a servi par la suite : un chat textuel en temps réel utilisant les frameworks « Express » et « Socket.io », étant donné que ces modules allaient être utilisés pour la conception du backend du jeu.

En outre, j'ai travaillé sur l'anticipation de l'intégration du frontend dans le serveur ainsi que celle de la base de données. Nous avons également tenté d'organiser notre architecture sur le Git, ce que j'aurais peut-être abordé différemment avec le recul. Ensuite, j'ai mis en place la partie inscription et l'envoi des données utilisateurs une fois que l'utilisateur était connecté.

De plus, j'ai également collaboré avec mon binôme pour anticiper l'arrivée du jeu en créant de nombreux objets pour permettre au frontend de récupérer efficacement les données et pour faciliter l'attribution des données du côté serveur. Une fois le jeu de poker implémenté (sans les rôles), j'ai pu l'intégrer dans le serveur et envoyer les données au frontend déjà intégré. Par la suite, j'ai travaillé avec mon binôme pour gérer dynamiquement l'arrivée des joueurs sur une table, en permettant de la rejoindre ou de la quitter.

Une fois le jeu stable, j'ai pu commencer à travailler sur l'intégration des rôles dans le serveur, et aider le frontend à l'afficher grâce à « Socket.io ».

En tant que membre de l'équipe réseau, j'avais une vue d'ensemble de chaque partie du projet. Mon rôle m'a permis d'aider les parties autres que réseau lorsqu'elles rencontraient des blocages. Ce rôle m'a appris de nombreuses choses, telles que le travail en équipe, la persévérance, la rigueur dans la documentation du code et la gestion du temps.

3.6.2 Johary RALAMBO

Pour ce projet intégrateur qui consistait à développer un jeu de poker revisité, j'ai eu l'occasion de travailler sur la partie frontend en binôme avec ahmed ZANED, ainsi qu'avec Fares ZID qui nous a rejoint dans la deuxième partie du projet. Il a été décidé d'utiliser le framework « VueJs », une technologie très avantageuse pour créer des interfaces et applications web monopages. N'ayant aucune expérience avec cet outil, j'ai pu le découvrir et me familiariser avec lors d'une montée en niveau en début de projet. En parallèle de celle-ci, mon binôme et moi avons créé le wireframe de l'application.

Une fois le wireframe terminé et validé par l'ensemble de l'équipe, nous avons commencé l'implémentation des pages de base, en HTML et CSS

premièrement (sans framework). Ainsi, je me suis occupé de la page d'accueil de la page du menu principal, et de la page règles du jeu. Grâce à l'UE Programmation Web 2 du semestre 4, j'ai pu effectuer cette tâche sans trop de difficultés.

Après l'implémentation des pages de base en HTML et CSS, l'intégration en « VueJS » a pu commencer. Lors de cette intégration, nous avons implémenté avec mon binôme un système de traduction, grâce au plugin « vue-i18n ». Après cette intégration, nous avons commencé le développement de l'interface.

Ensuite, il était convenu de procéder à un premier assemblage avec la partie réseau afin d'intégrer le poker de base. Cette tâche a été très longue et complexe, étant donné le nombre important de données à se transmettre mutuellement entre le frontend et le backend, ainsi que les différents ajustements nécessaires. Il m'arrivait souvent de ne plus en voir le bout.

Par la suite, j'ai pu procéder aux animations de certains rôles du jeu pour préparer le second assemblage avec la partie réseau.

Pour conclure, ce projet a été très enrichissant humainement et techniquement, en effet c'était la première fois que je participais à un projet avec autant de personnes.

Certaines tâches (intégration « VueJS », assemblage) ont pris plus de temps que prévu, entraînant un certain retard. Ainsi, lors des dernières semaines nous avons fourni une grosse charge de travail afin de rattraper ce retard, ce qui a été très épuisant.

Aussi, j'ai fait en sorte d'assister au plus grand nombre de réunions, et de m'impliquer du mieux que je pouvais pour mener à bien ce projet.

Le résultat du projet est satisfaisant, même s'il est évidemment possible d'apporter des améliorations et corrections.

3.6.3 Corentin STEINER

Dans le cadre de ce projet, j'ai eu le rôle de chef de projet de l'équipe 6B. Mes objectifs étaient de gérer l'organisation globale du projet, du découpage via des jalons, du découpage des tâches et d'assurer le bon déroulement du projet.

J'ai rédigé le cahier des charges, les comptes-rendus de réunion et les comptes-rendus hebdomadaires, ainsi que le rapport. Je supervisai le projet tout au long de son déroulement, j'étais à l'écoute et m'assurait que tout le monde soit au courant de l'avancement du projet. J'ai su aussi définir les priorités et m'adapter lorsque les circonstances l'exigeaient afin de garantir les délais pour les différentes étapes du projet. De même, j'ai fait en sorte de maintenir la cohésion du groupe et de trancher les dilemmes si besoin.

J'ai eu la chance d'avoir à diriger une très bonne équipe, travailleuse et dont les membres communiquaient bien entre eux, ce qui a pu me faciliter la tâche

comme chef de projet, mais je pense néanmoins avoir su apporter mon concours pour la réussite du projet. C'était la première fois que j'endossais le rôle de chef de projet : cela a été une expérience enrichissante qui me servira sans aucun doute plus tard, même en tant que développeur, je saurais mieux identifier les attentes du chef de projet et interagir bénéfiquement avec l'ensemble du groupe.

3.6.4 Florian WU

Pour le projet intégrateur notre groupe s'est mis d'accord pour concevoir une variante du poker avec des effets spécifiques à savoir des compétences pour un rôle spécifique donnée. Ma partie consistait principalement à coder la partie backend par défaut n'ayant pas de point fort spécifique.

Tout d'abord, connaissant ma capacité à programmer dans un nouveau langage, j'ai dû faire une montée en compétence pour me renseigner sur le JS Electron, à savoir faire de mini programme consistant à comprendre son fonctionnement, ses limites et ses avantages.

Par la suite, j'ai donc été assigné au côté jeu du projet par défaut, à savoir le fonctionnement du jeu en lui-même. C'est pourquoi mon binôme pour la partie backend, Hayk avait déjà avancé sur le squelette du jeu de base, puis on a reparti les fonctions. Cependant malgré ma contribution j'avais beaucoup de mal à comprendre comment implémenter le code.

Je n'ai donc pas réussi à finir mes fonctions et par stress d'efficacité j'ai donc menti sur mon avancement, je n'y arrivais pas et je voulais absolument finir mes fonctions avant de les partagées. Pour éviter d'avoir des conflits de merge, de ce fait j'ai donc implémenté de mon côté les fonctions.

Le temps passé à coder et à essayer de comprendre le code ne donnait pas grand résultat, ceux à quoi pour la plupart de mes heures de recherches non efficace. Plus tard mon erreur de communication a pu mettre le projet en retard, du manque de ma communication et incapacité à finir les fonctions de test, même si certaine était déjà implémentée donc de ce fait en double (les fonctions de comparaisons de mains de poker) du fait que je n'ai pas push le code sur git. A la fin on m'a donc demandé de push sur une autre branche, ce que j'ai fini par faire et une partie du code que je pensais valide ne l'était même pas.

Enfin en tant que membre backend, j'ai dû me tourner sur une fonctionnalité IA que j'implémente. Sur laquelle, je devais montrer mon implication afin d'avoir contribuer au projet. A la suite petits problèmes personnels, j'ai eu un grand découragement et manque de motivation et dû à mon erreur, je n'ai pas osé demander de l'aide à mes collègues. Ce projet m'a tout de même permis de comprendre qu'il faut communiquer davantage dans un projet de groupe et développer de manière autonome ses connaissances dans un nouveau langage de programmation.

3.6.5 Shixiang XU

Dans ce projet, je suis rattaché à la partie réseau, durant le projet j'ai pu : Tout d'abord, j'ai d'abord compris le principe de fonctionnement de « NodeJS » et « socket.io » à travers le chat textuel en temps réel relié à une base de données, afin de comprendre le fonctionnement des transmissions des données de la base de données vers un front end.

Ensuite j'ai amélioré la lisibilité de server.js en ajoutant "router" pour "inscription" et "connexion" dans server.js. Et puis, j'ai ajouté une session à server.js pour que le serveur se souvienne de l'utilisateur connecté et transmette les données en toute sécurité.

D'ailleurs, j'ai transféré le classement de "classement" vers le front end via les données de BDD et compléter le registre de requêtes ajax vers la page de gestion des utilisateurs (changer le nom d'utilisateur, le mot de passe)

Finalement, j'ai ajouté une fonctionnalité "spectateur" dans la partie back du projet, afin que les joueurs en attentes puissent rejoindre la partie et observer la partie déjà en cours.

Pour mes difficultés : Tout d'abord, l'équipe a choisi javascript, que j'ai peu utilisé, donc beaucoup de contenu est à réapprendre, et certaines logiques sont différentes du langage C, donc ça prend beaucoup de temps à déboguer. La seconde est la langue, parfois je n'arrivais pas à comprendre précisément les attentes de l'équipe, merci à l'équipe de toujours m'expliquer patiemment.

Pour conclure, j'ai appris beaucoup de nouvelles connaissances dans ce projet d'équipe, comme l'utilisation efficace de javascript, etc., et en même temps m'a donné une nouvelle compréhension du réseau client et serveur. C'était aussi la première fois que je réalisais que les soft skills sont si importants, surtout, j'ai amélioré mes compétences en travail d'équipe et en communication

3.6.6 Ahmed ZANED

En ce qui concerne le projet intégrateur, nous avons choisi de développer un jeu de poker revisité avec quelques modifications, en ajoutant des rôles pour plus de fun. J'ai travaillé sur la partie Frontend avec mon binôme Johary RALAMBO et Fares ZID qui nous a rejoint dans la deuxième partie du projet. Nous avons décidé d'utiliser le framework « VueJS » pour créer notre interface, un outil que je n'avais jamais utilisé, ce qui a nécessité un peu de temps pour nous familiariser avec.

En parallèle de notre montée en compétences, mon binôme et moi avons conçu le wireframe de l'application. Nous avons ensuite travaillé sur toutes les pages du site, sauf l'interface du jeu en HTML et CSS, ce qui était une erreur de notre part, vu que nous avons conclu qu'il fallait tout intégrer dans

des composantes « VueJS » pour la suite. Ainsi, j'ai travaillé sur les pages de connexion et d'inscription.

Ensuite, nous avons procédé à l'intégration de « VueJS » et mis en place un système de traduction grâce au plugin « vue-i18n ». La tâche la plus difficile a été l'intégration entre le frontend et le serveur du jeu, ainsi que le débogage qui a suivi. Cette étape a nécessité beaucoup de temps et d'efforts pour assurer que toutes les données étaient correctement transmises et interprétées par les deux parties. Cela a été un vrai défi pour l'équipe, mais nous avons finalement réussi à dépasser les obstacles et à fournir une expérience de jeu fluide pour l'utilisateur final.

Ensuite, j'ai participé à l'implémentation des différents rôles du jeu en préparation de la deuxième phase d'assemblage avec la partie Réseau. Nous avons ensuite travaillé sur l'interface utilisateur et j'ai effectué de nombreuses modifications pour la rendre la plus simple et compréhensible possible pour les utilisateurs. C'était un défi particulièrement difficile étant donné que le jeu de poker contient de nombreuses informations à afficher à l'utilisateur, et il était important de trouver un équilibre entre la présentation de ces informations et la facilité d'utilisation de l'interface. Cependant, avec l'aide de mes collègues et en m'appuyant sur mes compétences en programmation web, j'ai réussi à surmonter ce défi et à fournir une interface conviviale pour les utilisateurs.

En conclusion, le développement de ce jeu de poker revisité a été une expérience très enrichissante pour moi. J'ai pu travailler en équipe sur une technologie que je ne connaissais pas auparavant, et ainsi développer mes compétences en matière de développement frontend. J'ai également eu l'opportunité de participer à toutes les étapes du projet, de la création du wireframe à l'implémentation des fonctionnalités les plus avancées. Ce projet m'a permis de repousser mes limites et de progresser.

3.6.7 Hayk ZARIKIAN

Dans ce projet intégrateur, j'ai eu la charge de m'occuper de la partie backend jeu, des mises en page des documents et de la communication.

Tout d'abord, je me suis chargé de la partie backend jeu qui se décompose en deux parties importantes. Dans un premier temps, la partie fonctionnelle qui est la première étape avant la programmation car il est important de définir comment nous allons programmer, nous organiser avant de passer à une étape plus technique. J'ai donc réalisé un diagramme de classe de base du jeu de poker sans prendre en compte la fonctionnalité des rôles. Par la suite, j'ai construit le squelette de base en me référant sur le diagramme de classe afin de distribuer les fonctions avec mon binôme Florian WU, les fonctions ont été commentées afin de faciliter l'implémentation. Dans un deuxième temps, j'ai implémenté les fonctions qui m'étaient associées avec anticipation tout en faisant des tests unitaires à l'aide de « Jest » afin de valider le fonctionnement de la fonction qui m'a aidé à debuguer plus facilement

durant tout le long du projet. J'ai tenu la partie technique avec rigueur car j'ai push sur la branche dédiée à la partie backend jeu tout en indiquant sur un changelog les changements à chaque push.

Une fois mes fonctions réalisées, je me suis attardé sur la réalisation du nouveau diagramme de classe en y intégrant la fonctionnalité des rôles, puis la réalisation du nouveau squelette de base, enfin l'implémentation des rôles. Par la suite, j'ai arrêté l'implémentation des rôles dû à un mensonge de mon binôme concernant la réalisation de ses fonctions qui a engendré un retard sur le premier jalon d'assemblage backend, frontend. J'ai alors réalisé l'implémentation des fonctions de mon binôme le plus rapidement possible afin de rattraper le temps de retard, j'ai eu l'aide de Ekaterina ZAITCEVA pour la fonction principale du jeu « jouerTour () » où ils y avaient beaucoup de cas à prendre en compte. Ensuite, j'ai repris l'implémentation des rôles. Par la suite, j'ai répondu aux questions et exigences de Yannick LECAM par des solutions et en lui expliquant l'implémentation du jeu et son fonctionnement afin d'établir avec succès l'assemblage du backend, front. J'ai également procédé aux debuggings du backend jeu tout au long du projet.

Mise à part la partie fonctionnelle / technique, je me suis aussi occupé de la mise en page des documentations, du design, de la charte documentaire. Par conséquent, tous les documents liés au projet intégrateur sont cohérents et possèdent la même charte graphique et les mêmes règles de mises en page (taille, police, type de sommaire, marges, ...).

Également en tant que community manager, je me suis chargé de la communication / marketing du jeu en réalisant tout d'abord un trailer annonçant le jeu avec un montage vidéo / photo, puis des flyers (mains traditionnelles au poker, les rôles). J'ai également réalisé le document de communication.

Pour en conclure, ce projet m'a beaucoup apporté sur plusieurs points. Tout d'abord sur le point technique avec une montée en compétence en langage Javascript mais aussi du framework « Jest » qui m'a permis de réaliser les tests unitaires sur mes fonctions. Ensuite sur le point de la rédaction, de nature à aimer faire de la documentation, j'ai amélioré ma manière de documenter, de faire des rapports clairs et cohérent tout en y intégrant design et clarté. Finalement, sur le point de la communication où j'ai appris à faire des montages vidéo, des flyers ce qui m'a permis de développer mes capacités artistiques. Le résultat du projet est très satisfaisant et très intéressant au niveau de l'originalité car nous avons cassés les codes du jeu de poker afin de créer un jeu qui ressemble à aucun autre ce qui a permis de mettre le jeu sur un point d'unicité et d'identifier notre équipe RoundCoders.

3.6.8 Ekaterina ZAITCEVA

Dans le cadre de ce projet j'ai eu des responsabilités assez diverses. D'abord, j'étais dans l'équipe 'Base de données'. Avec mon binôme Fares ZID on a fait la conception de la BDD pour les différentes versions du jeu (alpha, beta et

une version avec les fonctionnalités complémentaires prévu dans le cas où il y a une marge avant le rendu du projet), ainsi que l'implémentation pour les deux premières versions.

Comme notre équipe a fini l'implémentation plus tôt que les autres, j'ai rejoint l'équipe 'Jeu' pour relier le jeu avec la BDD et écrire la fonction principale du jeu - 'joueur_tour' avec Hayk ZARIKIAN.

Le reste du temps j'étais chargée de gérer la VM pour le serveur (déploiement et inspection des logs d'erreurs), transformer la version web en version desktop avec Electron et faire des tests (d'où vient le pseudo 'Bug Hunter' qui m'a été attribuée par le groupe et 700000000+ jetons sur mon compte du jeu). J'ai aussi participé à différentes sessions de débogage, surtout avec Yannick LE CAM et Johary RALAMBO.

Ce projet m'a permis de monter en compétences de types hard skills et soft skills également, c'était une expérience très positive. Même si la répartition des groupes a été faite au hasard, je suis contente et reconnaissante d'avoir pu travailler avec ces personnes formidables.

En conclusion, je suis très satisfaite du rendu actuel du projet et j'espère pouvoir continuer son développement pendant les vacances d'été avec notre groupe et éventuellement le déployer.

3.6.9 Fares ZID

Pour notre projet intégrateur, on a choisi de concevoir un jeu de poker revisité, en mettant en place un système qui distribue à chaque joueur un rôle afin de rendre ce jeu de cartes plus divertissant.

Durant ce projet, j'ai commencé par travailler sur la base de données avec Ekaterina. Pendant les premières semaines, on s'est occupé de mettre en place la base de données et les fonctions nécessaires à la manipulation de cette BDD. Une fois prête, j'ai intégré l'équipe frontend afin d'aider Johary et Ahmed à la conception des pages du jeu en utilisant le framework « VueJS », ainsi qu'à mettre en place les différentes animations et les améliorations IHM au fur et à mesure. J'ai pu aussi utiliser mes compétences de production de musique en bénéfice du projet afin de produire deux bandes sons pour l'accueil et le jeu en cours.

Pour pouvoir effectuer ces tâches, une montée en compétences pour « NodeJS », « ElectronJS » et surtout « VueJS » était primordiale.

Parmi les difficultés que j'ai rencontré pendant le projet, c'est l'adaptation au framework « VueJS », mais j'ai pu surmonter cet obstacle progressivement. De plus, j'ai eu quelques bugs concernant Git sur ma machine au début du projet, que j'ai pu résoudre par la suite.

Tout au long de ce projet, j'ai pu acquérir des compétences sur les langages cités et j'ai appris à mieux communiquer avec mes collègues. De plus, j'ai eu

l'occasion de mettre en pratique quelques éléments de l'UE Génie Logiciel suivie le semestre précédent.

4 FAQ équipe

4.1 Comment évaluer-vous l'ensemble du projet ?

L'ensemble du projet a été apprécié par tous les membres de l'équipe que ce soit le thème du projet, l'organisation ou la partie technique, la motivation pour mener à bien le projet était au rendez-vous.

4.2 Êtes-vous satisfait de la qualité du travail accompli par l'équipe ?

Le travail est bien évidemment satisfait par l'équipe car nous avons effectué un travail rigoureux et implémenté les fonctionnalités prioritaires. La qualité du travail est visible dans tous les domaines : frontend, backend (serveur et jeu), base de données et interface homme-machine.

4.3 Comment évaluer-vous l'organisation et la planification du projet ?

L'organisation du travail était plus que correcte car malgré les difficultés de certains membres de l'équipe à être disponible et les examens en parallèle, nous avons toute même pu faire beaucoup de réunions. La planification du projet avait bien été définie ce qui nous a permis toujours d'avancer vers les nouvelles tâches.

4.4 Comment décrivez-vous le niveau de communication au sein de l'équipe de projet ?

L'équipe est satisfaite de la communication au sein de l'équipe car nous avons tous participer à l'avancement du projet en voyant le résultat, les bugs, les tâches de chacun sur le serveur discord et lors nombreuses réunions.

4.5 Avez-vous toutes les ressources nécessaires pour mener à bien votre travail sur ce projet ?

Globalement, nous n'avions pas besoin d'énormément de ressource mise à part la VM pour mettre le jeu sur le serveur et de beaucoup de temps. Par conséquent, les ressources ne nous ont pas freiné sur l'avancement du projet. Nous avons utilisé tout le temps dédié au projet.

4.6 Y'a-t-il des domaines où vous pensez que l'équipe pourrait s'améliorer pour mieux accomplir ses tâches ?

Une amélioration des compétences techniques afin d'être plus efficace sur la réalisation des tâches et de permettre d'implémenter plus de fonctionnalités.

4.7 Quels sont les aspects les plus stimulants ou les plus gratifiants de travailler sur ce projet ?

Les aspects les plus stimulants et gratifiants de travailler sur ce projet sont les implantations avec un résultat visible, la fonctionnalité des rôles qui nous a beaucoup motivé et d'avoir un résultat unique et original. Les assemblages ont été très passionnant également avec le debugging pas à pas du code. Les nouvelles technologies employées, nous ont également passionnés.

4.8 Quels sont les objectifs futurs de l'équipe pour ce projet ?

Certains membres de l'équipe aimeraient mettre le jeu en ligne afin de continuer à l'améliorer continuellement, d'autres voudraient s'arrêter avec le projet et passer à la conception de d'autres projets passionnantes et d'autres construire un jeu de société PokerGang. Finalement les objectifs sont variés selon les membres de l'équipe mais constitue un objectif lié à cette expérience.

5 Conclusion

Pour en conclure, à l'issue de ce projet, nous avons donc réussi à implémenter un jeu conformément au cahier des charges établie avec des changements sur certains aspects qui ont eu pour objectif de simplifier le jeu et de l'équilibrer afin de répondre avec succès à notre public visé. Le projet est abouti et plaisant à jouer.

Si nous avons pu parvenir à ce résultat, c'est bien grâce à une excellente communication dans le groupe et l'implication dans le travail apporté par chacun. Dans l'ensemble, chaque membre du groupe s'est pleinement investi et a échangé aussi bien avec le chef de projet qu'entre eux. L'ambiance était donc dynamique et enthousiaste, et elle a permis à tous de finaliser sa tâche grâce à ce soutien permanent.

Ce projet fut donc une expérience très enrichissante pour tout le monde : l'esprit collaboratif, les encouragements mutuels ainsi les discussions et propositions de solutions différentes ou complémentaires, nous ont permis de mener à bien ce projet à l'image du monde du travail.

6 Annexes

6.1 Document technique

Vous trouverez la documentation technique ci-dessous à la suite de la section « Annexes ».

6.2 Document de communication

Vous trouverez le document de communication ci-dessous à la suite de la section « Annexes ».

6.3 Document Interface Homme-Machine

Vous trouverez le document interface Homme-Machine ci-dessous à la suite de la section « Annexes ».

6.4 Document utilisateur

Vous trouverez le document utilisateur ci-dessous à la suite de la section « Annexes ».

6.5 Politique Git

6.5.1 Branches

Les différentes ont pour but de donner à chaque équipe un espace pour travailler son domaine et pour permettent

Main : branche principale qui contient le code stable et fonctionnel

Release : branche de préparation pour les livraisons

Development : branche où les développeurs fusionnent leur code pour les tests d'intégration

Backend-game-feature : branche dédiée aux fonctionnalités du backend pour le jeu

Backend-game-hotfix : branche dédiée aux correctifs urgents du backend pour le jeu

Backend-server-feature : branche dédiée aux fonctionnalités du backend pour le serveur

Backend-server-hotfix : branche dédiée aux correctifs urgents du backend pour le serveur

Database-feature : branche dédiée aux fonctionnalités de la base de données

Database-hotfix : branche dédiée aux correctifs urgents de la base de données

Frontend-feature : branche dédiée aux fonctionnalités du frontend

Frontend-hotfix : branche dédiée aux correctifs urgents du frontend

6.5.2 Politique de développement

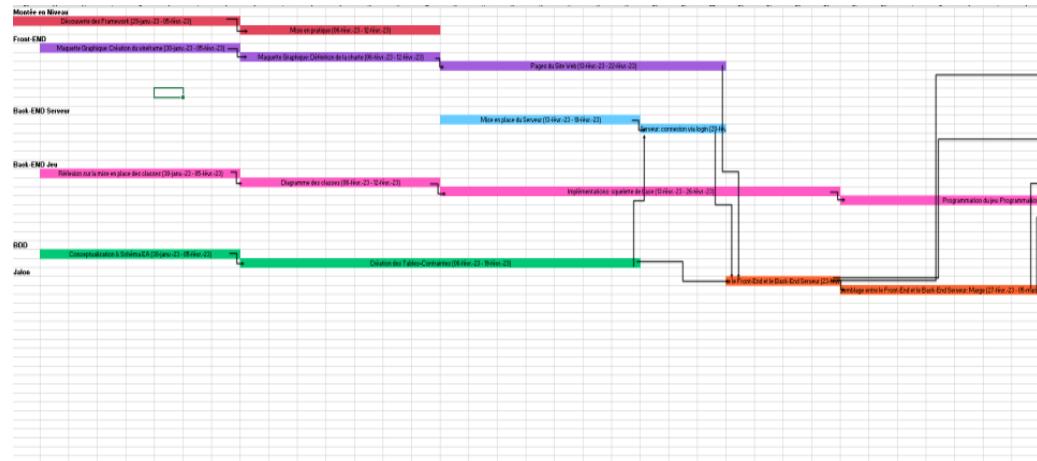
Il est important que chaque membre de l'équipe travaille sur la dernière version du code pour éviter les conflits et les erreurs de merge. Pour cela, il est demandé que chaque membre de l'équipe fasse un "pull" à chaque fois qu'il commence à travailler sur une branche. Cela permet de récupérer les dernières modifications effectuées par les autres membres de l'équipe et de travailler sur une base à jour. Ensuite, dès qu'une session de travail est finie, il est demandé à la fin de faire un "push" pour envoyer ces modifications sur la branche correspondante sur le serveur Git. Cela permet de sauvegarder le travail accompli et de le rendre disponible pour les autres membres de l'équipe.

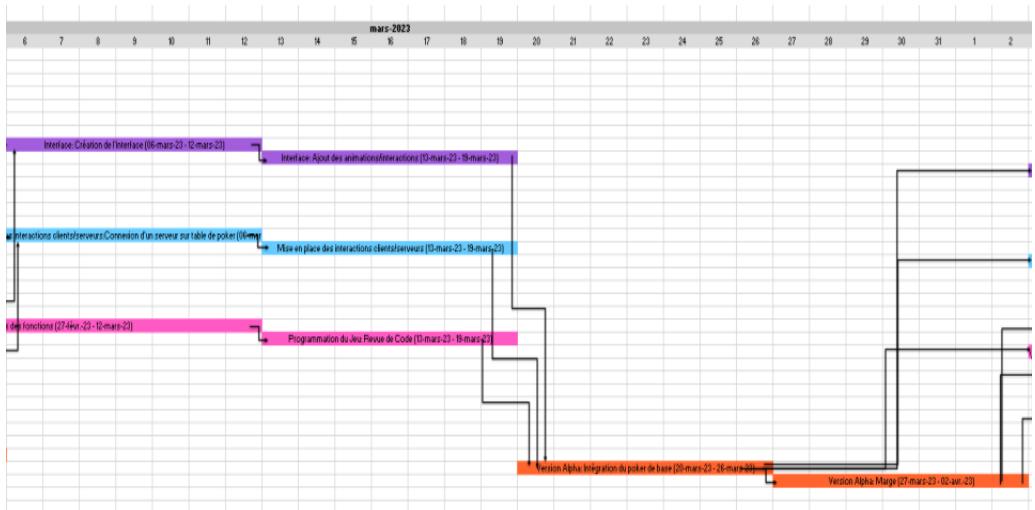
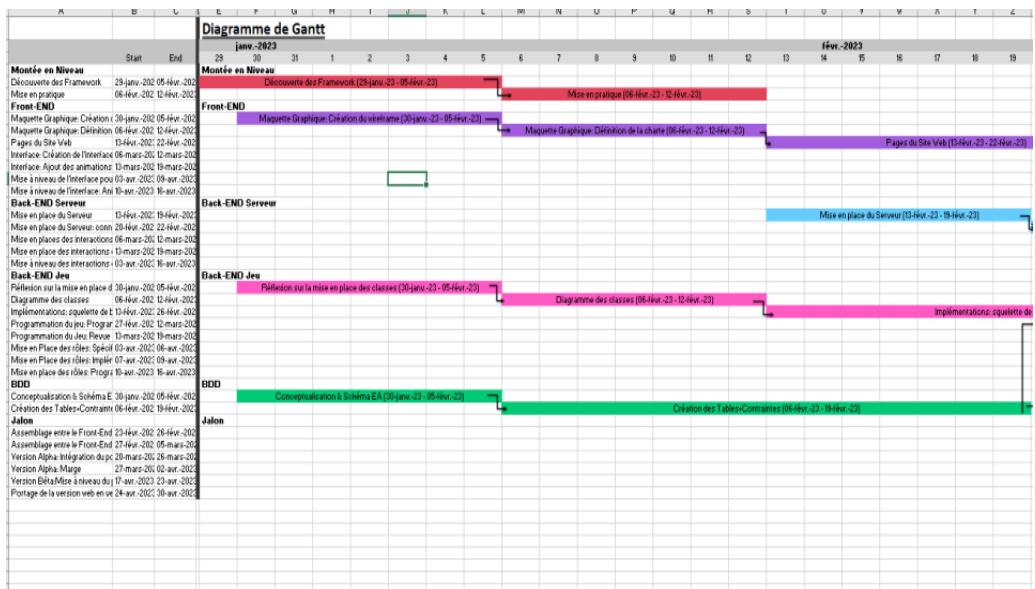
Il est important de respecter cette politique pour éviter les conflits et les pertes de travail, ainsi que pour faciliter la collaboration au sein de l'équipe. Pour pouvoir prendre en compte les modifications de tous les fichiers et prendre en compte le gitignore, il est recommandé d'utiliser git add -A.

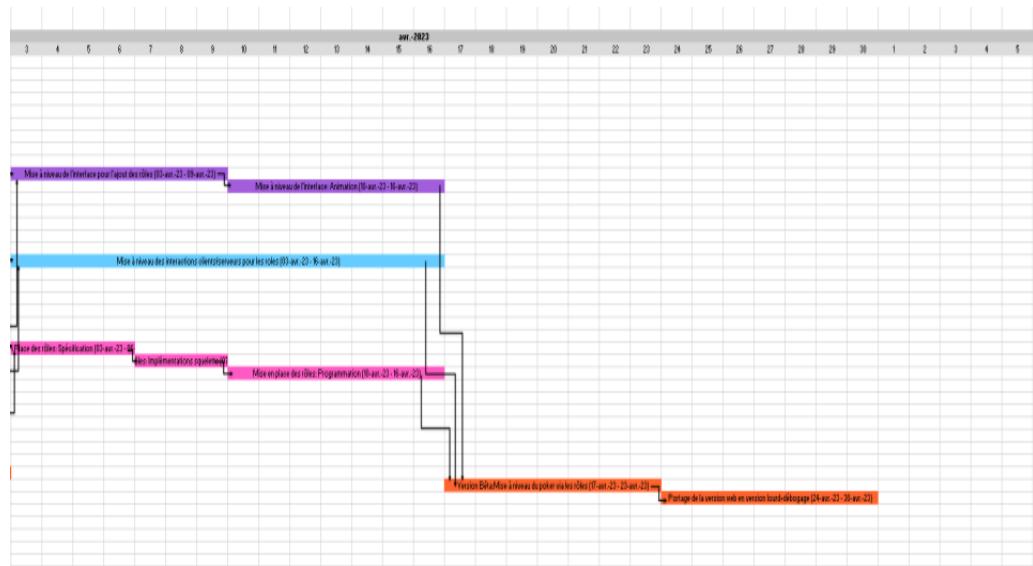
6.5.3 Politique de fusion de branches

Nous avons décidé d'utiliser la méthode de fusion de branches en amont (upstream) pour assurer la qualité du code et la stabilité du projet. Avant de fusionner une branche, le développeur doit s'assurer que toutes les modifications ont été testées et validées. Ensuite, il doit demander une revue de code à un autre développeur avant de fusionner la branche dans la branche cible.

6.6 Diagramme de Gantt







UFR

de mathématique

et d'informatique

Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

PROJET INTÉGRATEUR - POKERGANG DOCUMENTATION TECHNIQUE



Hayk ZARIKIAN – Lead Tech Game Engine

Yannick LECAM – Lead Tech Server

Johary RALAMBO et Ahmed ZANED – Lead Tech Front

Ekaterina ZAITCEVA – Lead Tech Base de données

ÉQUIPE 6B - RoundCoders

Table des matières

1	Socle technique	1
2	Backend Jeu	1
2.1	Diagramme de classe du jeu de base	1
2.2	Diagramme de classe (avec fonctionnalité rôle)	2
2.3	Classe Carte	2
2.3.1	Description attributs.....	2
2.4	Classe Dealer	3
2.4.1	Description Attributs	3
2.4.2	melangerPaquetCarte ()	3
2.4.3	initCarteTable ()	4
2.4.4	initRoles ().....	4
2.4.5	distribue ()	4
2.4.6	revelationCarteJoueur ().....	4
2.4.7	attributionDesGains ()	4
2.5	Classe Mains	5
2.5.1	Description Attributs	5
2.5.2	creeRang ().....	5
2.6	Classe Rang.....	5
2.6.1	Description Attributs	5
2.6.2	quinteFlushRoyale ()	6
2.6.3	quinteFlush ().....	6
2.6.4	quads ()	6
2.6.5	full ()	7
2.6.6	couleur ()	7
2.6.7	quinte ()	7
2.6.8	brelan ()	7
2.6.9	deuxPaires ()	7
2.6.10	unePaire ()	8
2.6.11	chercheRang ().....	8
2.7	Classe Joueur	8
2.7.1	Description Attributs	8
2.7.2	recevoirCarte ().....	9
2.7.3	miser ().....	9
2.7.4	check ()	9

2.7.5	seCoucher ().....	9
2.8	Classe Jeu.....	10
2.8.1	Description Attributs	10
2.8.2	initJoueur ()	10
2.8.3	departJeuPoker ()	11
2.8.4	quitterJoueur ().....	11
2.8.5	jouerTour ().....	11
2.9	Classe Role.....	11
2.9.1	Description attributs.....	11
2.10	Classe Escroc.....	12
2.10.1	recupererMoitierMise ().....	12
2.11	Classe Usurpateur.....	12
2.11.1	copierRoleJoueur ()	12
2.12	Classe Policier	12
2.12.1	mettreEnPrison ()	12
2.13	Classe Insolvable.....	13
2.13.1	emprunterArgentDealer ().....	13
2.14	Classe Indicateur.....	13
2.14.1	indiquerCarteJoueur ()	13
2.15	Classe Voleur	13
2.15.1	volerCarte ().....	13
3	Scripts Python : Mail automatique	13
3.1	Introduction.....	13
3.2	python_launch_mail.js	13
3.3	mail.py	14
4	Base de données.....	15
4.1	Schéma EA	15
4.2	Fonctions	15
4.2.1	check_pseudo_exists.....	15
4.2.2	update_pseudo.....	16
4.2.3	db_open ()	16
4.2.4	create_tables ().....	16
4.2.5	db_close ()	16
4.2.6	inscription ().....	16
4.2.7	connexion ().....	16
4.2.8	change_pwd ().....	17

4.2.9	update_user_jetons ()	17
4.2.10	bet ()	17
4.2.11	reload_jetons ()	17
4.2.12	take_back_half_bet_money_escroc ()	18
4.2.13	update_nb_games ().....	18
4.2.14	update_nb_victories ()	18
4.2.15	add_winner ()	18
4.2.16	get_nb_jetons ()	18
4.2.17	get_user_info ()	18
4.2.18	get_user_stats ().....	19
4.2.19	classement_jetons ().....	19
4.2.20	classement_victoires ().....	19
5	Frontend	19
5.1	Accueil	19
5.2	Connexion.....	20
5.3	Inscription.....	20
5.4	Menu principal	20
5.5	Règles du jeu	21
5.6	Classement	21
5.7	Interface	21
5.8	Changer mot de passe	22
5.9	Contact	22
5.10	Credits.....	22
6	Serveur	23
6.1.1	Les objets.....	23
6.1.2	Les routes	24
6.1.3	Parcours des tableaux.....	25
6.1.4	Gestion des réceptions	26
6.1.5	Gestion des émissions	28
6.1.6	Gestion de la partie jeu	28
6.1.7	Gestion de la déconnexion des joueurs.....	30
6.1.8	Gestion des rôles	30
7	Tests unitaires	31
7.1	Test unitaire Carte	32
7.2	Test unitaire Dealer.....	32
7.3	Test unitaire Jeu.....	32

7.4	Test unitaire Mains	33
7.5	Test unitaire Rang	33
7.6	Test unitaire Role	33
7.7	Test unitaire Escroc	33
7.8	Test unitaire Usurpateur	33
7.9	Test unitaire Policier	33
7.10	Test unitaire Insolvable	34
7.11	Test unitaire Indicateur	34
7.12	Test unitaire Voleur	34

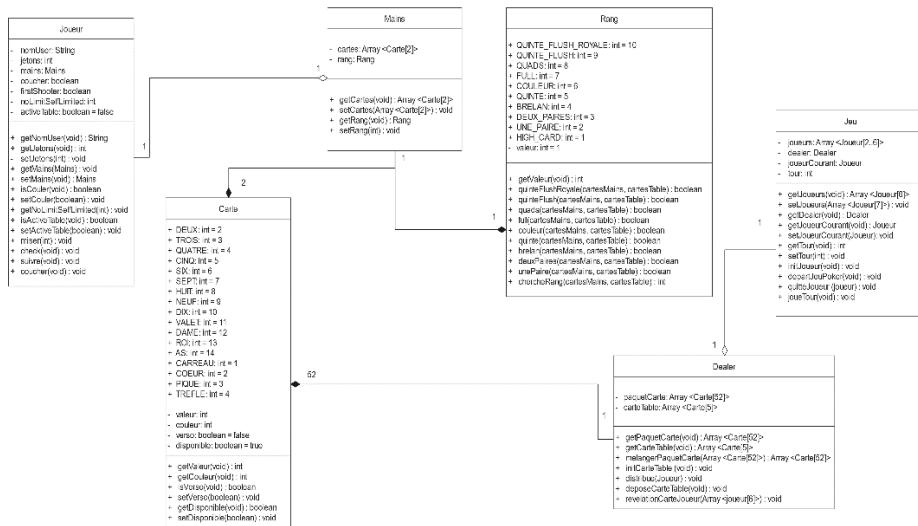
1 Socle technique

Pour que notre projet fonctionne correctement, nous avons utilisés plusieurs dépendances, qui sont les suivants :

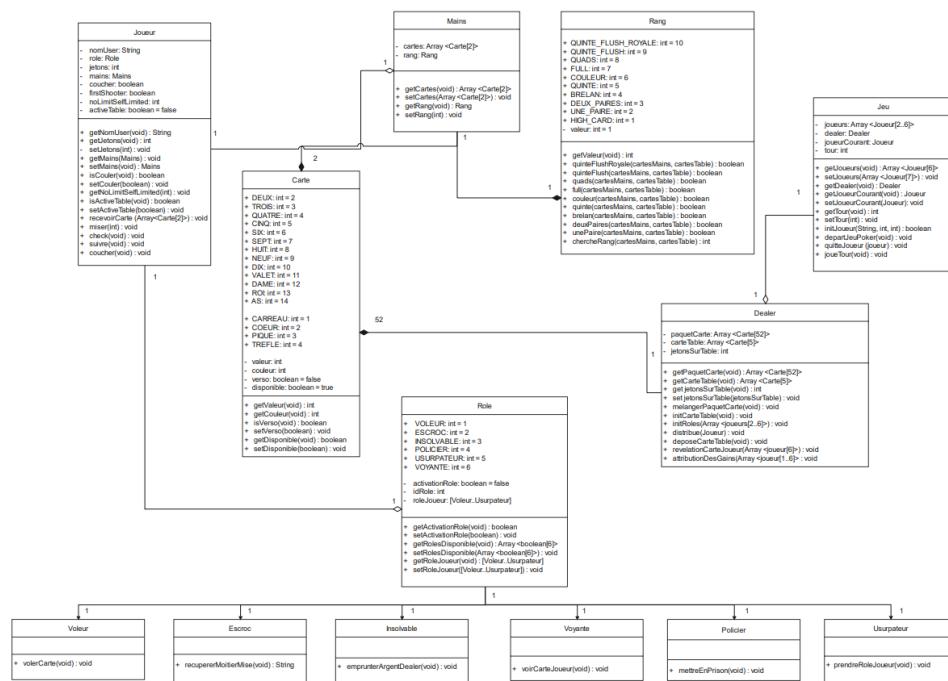
- ❖ Framework VueJs : Utilisation pour le Front-End
 - ❖ Framework ElectronJs : Utilisation de base pour la transformation de l'application web en desktop
 - ❖ Framework JestJs : Utilisation de jest pour effectuer les tests unitaires
 - ❖ Plateforme de développement NodeJs : Utilisation pour le serveur
 - ❖ Langage JavaScript : Langage Back-End principal
 - ❖ Langage HTML, CSS : Langage Front-End principal
 - ❖ Langage SQLite : Langage Base de données principal

2 Backend Jeu

2.1 Diagramme de classe du jeu de base



2.2 Diagramme de classe (avec fonctionnalité rôle)



2.3 Classe Carte

2.3.1 Description attributs

VALEUR DES CARTES		
Type	Nom de l'attribut	Description
int	const DEUX = 2	Carte de valeur 2
Int	const TROIS = 3	Carte de valeur 3
Int	const QUATRE = 4	Carte de valeur 4
Int	const CINQ = 5	Carte de valeur 5
int	const SIX = 6	Carte de valeur 6
Int	const SEPT = 7	Carte de valeur 7
int	const HUIT = 8	Carte de valeur 8
int	const NEUF = 9	Carte de valeur 9
int	const DIX = 10	Carte de valeur 10
int	const VALET = 11	Carte de valeur valet
Int	onst DAME = 12	Carte de valeur dame
Int	const ROI = 13	Carte de valeur roi
int	const AS = 14	Carte de valeur as

VALEUR DES COULEURS		
Type	Nom de l'attribut	Description
int	const CARREAU = 1	Valeur de la couleur CARREAU
Int	const CŒUR = 2	Valeur de la couleur CŒUR

int	const PIQUE = 3	Valeur de la couleur PIQUE
int	const TREFLE = 4	Valeur de la couleur TREFLE

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
int	valeur	Valeur de la carte	La valeur est comprise entre 2-14
int	couleur	Valeur de la couleur	La valeur est comprise entre 1-4
boolean	disponible	Vrai si la carte est disponible, faux sinon	La valeur par défaut est « true »

2.4 Classe Dealer

2.4.1 Description Attributs

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
Carte	paquetCarte = new Array(52)	Tableau de 52 cartes	
Carte	carteTable = new Array(5)	Tableau des 5 cartes sur la table	
int	jetonsSurTable = 0	Jetons présents sur la table	
boolean	rolesDisponible = new Array(6)	Tableau de boolean représentant la disponibilité des rôles	

2.4.2 melangerPaquetCarte ()

Paramètres	Aucun
Description	Mélanger les cartes de manières aléatoires
Retour	Modification de l'attribut « paquetCarte [52] » par l'adresse

2.4.3 initCarteTable ()

Paramètres	Aucun
Dépendances	melangerPaquetCarte ()
Description	Initialise les 5 cartes de la table aléatoirement dans l'attribut
Retour	Modification de l'attribut « carteTable[5] » par l'adresse

2.4.4 initRoles ()

Paramètres	Joueurs [2..6] : Tableau de joueurs ayant au minimum 2 joueurs initialisés et 6 joueurs au maximum initialisés
Dépendances	Constructeur Role
Description	Initialise le rôle des joueurs
Retour	Initialisation de l'attribut « role » dans la classe « Joueur.js ». Modification de la disponibilité du rôle dans l'attribut « rolesDisponible[2..6] »

2.4.5 distribue ()

Paramètres	joueur : Objet de type Joueur
Dépendances	❖ melangerPaquetCarte () ❖ joueur.recevoirCarte ()
Description	Distribution de deux cartes à un joueur aléatoirement
Retour	Initialisation de l'attribut « mains » dans la classe « Joueur.js » et l'attribut « cartes » dans la classe « Mains.js » par l'adresse.

2.4.6 revelationCarteJoueur ()

Paramètres	Joueurs [2..6] : Tableau de joueurs ayant au minimum 2 joueurs initialisés et 6 joueurs au maximum initialisés
Dépendances	attributionDesGains ()
Description	Détermination du / des vainqueur(s) et attribution des gains
Retour	Retourne les indices des joueurs gagnants

2.4.7 attributionDesGains ()

Paramètres	❖ joueurs [2..6] : Tableau de joueurs ayant au minimum 2 joueurs initialisés et 6 joueurs au maximum initialisés.
------------	---

	❖ indicesJoueursGagnants[1..6] : Tableau des indices des joueurs gagnants
Dépendances	BDD.add_winner ()
Description	Permet de mettre à jour la base de données pour les joueurs gagnants
Retour	Met à jour les jetons des joueurs gagnants depuis la base données par l'adresse

2.5 Classe Mains

2.5.1 Description Attributs

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
Carte	Cartes = new Array(2)	Tableau de deux cartes dans la main	
Rang	Rang	Objet de type rang	

2.5.2 creeRang ()

Paramètres	carteTable[5] : Tableau qui représente les 5 cartes de la table
Dépendances	Constructeur Rang
Description	Initialisation du rang du joueur
Retour	Initialise l'attribut « rang » dans la classe « Mains.js » par l'adresse

2.6 Classe Rang

2.6.1 Description Attributs

VALEURS DES RANGS DE MAIN TRADITIONNELLE		
Type	Nom de l'attribut	Description
int	const QUINTE_FLUSH_ROYALE = 10	Main QUINTE_FLUSH_ROYALE
int	const QUINTE_FLUSH = 9	Main QUINTE_FLUSH
int	const QUADS = 8	Main QUADS
int	const FULL = 7	Main FULL

int	const COULEUR = 6	Main COULEUR
int	const QUINTE = 5	Main QUINTE
int	const BRELAN = 4	Main BRELAN
int	const DEUX_PAIRE = 3	Main DEUX_PAIRE
int	const UNE_PAIRE = 2	Main UNE_PAIRE
int	const HIGH_CARD = 1	Main HIGH_CARD

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
int	valeur	Valeur du rang	La valeur est comprise entre 1-10

2.6.2 quinteFlushRoyale ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Dépendances	❖ couleur () ❖ quinteFlush ()
Description	Détermine si le joueur possède un rang de niveau 10 représentant la main « quinteFlushRoyale »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 10.

2.6.3 quinteFlush ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 9 représentant la main « quinteFlush »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 9.

2.6.4 quads ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 8 représentant la main « quads »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 8.

2.6.5 full ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Dépendances	❖ brelan () ❖ unePaire ()
Description	Détermine si le joueur possède un rang de niveau 7 représentant la main « full »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 7.

2.6.6 couleur ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 6 représentant la main « couleur »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 6.

2.6.7 quinte ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 5 représentant la main « quinte »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 5.

2.6.8 brelan ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 4 représentant la main « brelan »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 4.

2.6.9 deuxPaires ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
------------	--

Description	Détermine si le joueur possède un rang de niveau 3 représentant la main « deuxPaires »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 3.

2.6.10 unePaire ()

Paramètres	cartesMainsTable[7] : Tableau de 7 cartes qui représente les 5 cartes de la table et les 2 cartes de la main du joueur
Description	Détermine si le joueur possède un rang de niveau 2 représentant la main « unePaire »
Retour	Retourne vrai ou faux si le joueur possède un rang de niveau 2.

2.6.11 chercheRang ()

Paramètres	<ul style="list-style-type: none"> ❖ cartesMains[2] : Tableau des 2 cartes du joueur ❖ cartesTable[5] : Tableau des 5 cartes de la table
Dépendances	<ul style="list-style-type: none"> ❖ quinteFlushRoyale () ❖ quinteFlush () ❖ quads () ❖ full () ❖ couleur () ❖ quinte () ❖ brelan () ❖ deuxPaires () ❖ unePaire ()
Description	Permet de chercher le rang du joueur
Retour	Retourne le niveau du rang

2.7 Classe Joueur

2.7.1 Description Attributs

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
int	idJoueur	L'identifiant du joueur	L'identifiant doit être cohérent avec celui de la base de données
Role	Role	Objet de type Role	
Mains	Mains	Objet de type Mains	

int	Jetons	Nombre de jeton	
int	deposeJetons	Nombre de jeton déposé dans la partie courante	
boolean	Coucher = false	Vrai si le joueur est coucher, faux sinon	La valeur par défaut est « false »
boolean	firstShooter = false	Vrai si le joueur est le premier joueur à jouer, faux sinon	La valeur par défaut est « false »
int	noLimitSelfLimited	La limite du joueur en jeton	
boolean	jaiDejaJouerMonTour = false	Vrai si le joueur a déjà joué, faux sinon	La valeur par défaut est « false »

2.7.2 recevoirCarte ()

Paramètres	cartes[2] : Tableau des 2 cartes du joueur
Dépendances	Constructeur Mains
Description	Initialise l'attribut « mains »
Retour	Initialisation de mains par l'adresse

2.7.3 miser ()

Paramètres	❖ dealer : Un objet de type Dealer ❖ jetons : Entier représentant le nombre de jeton
Dépendances	BDD.bet ()
Description	Permet de miser des jetons
Retour	Met à jour la base de données

2.7.4 check ()

Paramètres	Aucun
Description	Le joueur passe son tour
Retour	Modification de la variable « jaiDejaJouerMonTour » à true

2.7.5 seCoucher ()

Paramètres	Aucun
------------	-------

Description	Permet au joueur de se coucher
Retour	Modification de la variable « jaiDejaJouerMonTour » et « coucher » à true

2.8 Classe Jeu

2.8.1 Description Attributs

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
Joueur	Joueurs = new Array(6)	Tableau des joueurs	Le nombre de joueur est compris entre 1-6
Dealer	Dealer = new Dealer	Objet de type Dealer	XX
Joueur	joueurCourant	Joueur courant	XX
int	Tour	Tour courant	La valeur est comprise entre 1-5
Joueur	joueursEnCours	Tableau des joueurs en cours de partie	Le nombre de joueur en cours et compris entre 2-6
int	indiceFirstShooter	Indice du premier joueur	XX
int	nbJoueur	Nombre de joueur restant à jouer	La valeur est comprise entre 0-6
boolean	finPartie = false	Vrai si la partie est terminée, faux sinon	La valeur par défaut est « false »

2.8.2 initJoueur ()

Paramètres	<ul style="list-style-type: none"> ❖ idJoueur : Identifiant du joueur ❖ jetons : Jetons mis en place sur la table (modifiable) ❖ noLimitSelfLimited : Jetons mis en place sur la table (non modifiable)
Dépendances	Constructeur Joueur
Description	Initialise le joueur si une place est disponible
Retour	Retourne vrai si l'initialisation du joueur c'est bien passée, faux sinon. Modification du tableau de « joueurs [] » par l'adresse.

2.8.3 departJeuPoker ()

Paramètres	Aucun
Dépendances	<ul style="list-style-type: none">❖ Dealer.initCarteTable ()❖ Dealer.distribue ()❖ Dealer.initRoles ()❖ Joueur.miser ()❖ jouerTour ()
Description	Lance le jeu de poker tout en initialisation les prérequis
Retour	Aucun

2.8.4 quitterJoueur ()

Paramètres	<ul style="list-style-type: none">❖ joueurs : Tableau des joueurs[2..6]❖ index : Indice du joueur qui souhaite quitter
Description	Quitter joueur, libération de la place dans le tableau
Retour	Libère une place dans le tableau par l'adresse

2.8.5 jouerTour ()

Paramètres	
Description	
Retour	

2.9 Classe Role

2.9.1 Description attributs

VALEURS DES ROLES		
Type	Nom de l'attribut	Description
int	const VOLEUR = 0	Role VOLEUR
int	const ESCROC = 1	Role ESCROC
int	const INSOLVABLE = 2	Role INSOLVABLE
int	const POLICIER = 3	Role POLICIER
int	const USURPATEUR = 4	Role USURPATEUR
int	const VOYANTE = 5	Role VOYANTE

ATTRIBUTS A INITIALISER			
Type	Nom de l'attribut	Description	Contraintes
boolean	activationRole	Vrai si le pouvoir a été	X

		activé, faux sinon	
int	idRole	Identifiant du role	La valeur est comprise entre 0-6
[VOLEUR..VOYANTE]	roleJoueur	Objet de type Voleur, Escorc, Insolvable, Policier, Usurpateur ou Voyante	

2.10 Classe Escroc

2.10.1 recupererMoitierMise ()

Paramètres	Joueur : le joueur escroc
Description	Permet de récupérer la moitié de sa mise en se couchant
Retour	Modification de l'attribut « coucher » pour le joueur en question

2.11 Classe Usurpateur

2.11.1 copierRoleJoueur ()

Paramètres	❖ joueur : le joueur voleur ❖ joueurs : Le tableau des joueurs
Description	Copie le rôle d'un des joueurs aléatoirement
Retour	Modification de l'attribut « role » dans la classe « Joueur.js »

2.12 Classe Policier

2.12.1 mettreEnPrison ()

Paramètres	Joueurs : Le tableau des joueurs
Description	Permet de faire coucher un joueur
Retour	Modification de l'attribut « coucher » pour le joueur en question

2.13 Classe Insolvable

2.13.1 emprunterArgentDealer ()

Paramètres	
Description	Permet d'emprunter des jetons au dealer
Retour	

2.14 Classe Indicateur

2.14.1 indiquerCarteJoueur ()

Paramètres	
Description	Permet de voir l'une des cartes d'un des joueurs
Retour	

2.15 Classe Voleur

2.15.1 volerCarte ()

Paramètres	❖ joueur : le joueur voleur ❖ joueurs : Le tableau des joueurs
Description	Échange la carte avec un autre joueur
Retour	Changement de l'attribut « cartes[0 ou 1] » dans la classe « mains.js »

3 Scripts Python : Mail automatique

3.1 Introduction

La création du script python « mail.py <email> » permet d'envoyer un mail via SMTP du compte « round.coders@gmail.com » afin de notifier l'inscription du joueur sur le jeu PokerGang.

3.2 python_launch_mail.js

L'exécution du script python est réalisé par un processus fils qui est créé en langage JS.

3.3 mail.py

Configuration du SMTP :

```
# Configuration SMTP

smtp_server = "smtp.gmail.com"
smtp_port = 587

smtp_login = "round.coders@gmail.com"
to_email = sys.argv[1]
auth_code = "ecujnyocilfwujtk" # Code chiffré

msg = MIMEMultipart ()
msg['From'] = smtp_login
msg['To'] = to_email
msg['Subject'] = "Sujet du message"
body = "Message"

msg.attach(MIMEText(body, 'plain'))

# Fin configuration SMTP
```

Attachement du logo d'équipe :

```
# Attachement du logo équipe

with open('logo.png', 'rb') as f:
    img_data = f.read()
    f.seek(0) # Remettre le curseur à la position de départ
pour éviter l'erreur 'ValueError: seek of closed file'

    # Resize the image to a maximum width of 200 pixels
    img = Image.open(f)
    max_width = 200
    width, height = img.size
    if width > max_width:
        ratio = max_width / width
        img = img.resize((max_width, int(height * ratio)), 
Image.ANTIALIAS)

    # Add the image to the email as an attachment
    img_byte_array = io.BytesIO()
    img.save(img_byte_array, format='PNG')
    img_byte_array = img_byte_array.getvalue()
    img_mime = MIMEImage(img_byte_array, name='logo.png')
    img_mime.add_header('Content-ID', '<logo>')
    img_mime.add_header('Content-Disposition', 'inline',
filename='logo.png')
    msg.attach(img_mime)

# Fin attachement du logo équipe
```

Signature du mail générique :

```
# Ajout de la signature
```

```

signature = """<br>
-- <br>
Bien à vous,<br>
<br>
Equipe RoundCoders<br>
round.coders@gmail.com<br>
Université de Strasbourg<br>
UFR Mathématique et Informatique<br>
Licence Informatique<br>
L3-S6 Equipe 6B"""
signature += f'<br>'
msg.attach(MIMEText(signature, 'html'))

#Fin ajout de la signature

```

Connexion SMTP et envoi du mail :

```

# Connexion SMTP et envoi du mail

with smtplib.SMTP(smtp_server, smtp_port) as server:
    server.starttls()
    server.login(smtp_login, auth_code,
initial_response_ok=False)
    server.docmd("AUTH", "XOAUTH2" + auth_code)

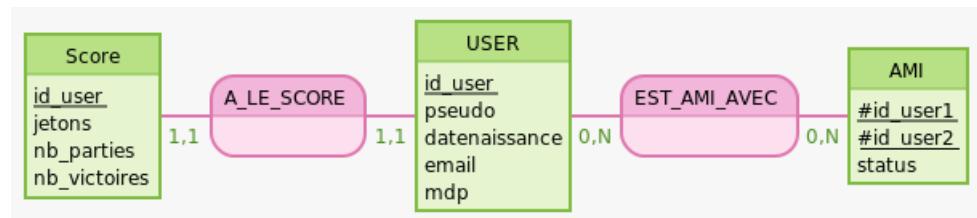
    server.sendmail(smtp_login, to_email, msg.as_string())
    print ("Message envoyé à " + to_email)

# Fin connexion SMTP et envoi du mail

```

4 Base de données

4.1 Schéma EA



4.2 Fonctions

4.2.1 check_pseudo_exists

Paramètres	pseudo : nom de l'utilisateur
Description	Vérifie si le pseudo existe ou non
Retour	Retourne vrai si le pseudo existe sinon faux

4.2.2 update_pseudo

Paramètres	❖ Id_user : identifiant d'un utilisateur ❖ pseudo : nom de l'utilisateur
Description	Met à jour le pseudo de l'utilisateur
Retour	Met à jour la base de données

4.2.3 db_open ()

Paramètres	Aucun
Description	Connexion à la base de données
Retour	Accessibilité à la base ouverte

4.2.4 create_tables ()

Paramètres	Aucun
Description	Permet de créer des tables de la base de données et des déclencheurs nécessaires s'ils n'existent pas encore.
Retour	Création des tables

4.2.5 db_close ()

Paramètres	Aucun
Description	Fermeture de la connexion à la base de données
Retour	Accessibilité à la base fermée

4.2.6 inscription ()

Paramètres	❖ pseudo : pseudonyme d'un utilisateur ❖ datenaissance : sa date de naissance ❖ email : son adresse email ❖ mdp : son mot de passe
Dépendances	❖ genSsalt () ❖ hash () de la bibliothèque bcryptjs
Description	Création d'un utilisateur dans la base de données, table User
Retour	Retourne vrai si l'inscription se passe bien, faux sinon

4.2.7 connexion ()

Paramètres	❖ email : adresse email d'un utilisateur
------------	--

	❖ mdp : son mot de passe
Dépendances	❖ compare () de la bibliothèque bcryptjs
Description	Vérifie si l'email et le mot de passe sont corrects
Retour	Retourne l'identifiant de l'utilisateur ou faux si la connexion se passe bien

4.2.8 change_pwd ()

Paramètres	❖ Id_user : identifiant d'un utilisateur ❖ nouveauMdp : son nouveau mot de passe
Description	Met à jour le mot de passe de l'utilisateur
Retour	Met à jour la base de données

4.2.9 update_user_jetons ()

Paramètres	❖ id_user : identifiant d'un utilisateur ❖ nb_jetons : le nombre de jetons qu'il faut ajouter au utilisateur
Description	Ajoute nb_jetons au compte de l'utilisateur
Retour	Met à jour la base de données

4.2.10 bet ()

Paramètres	Id_user : identifiant d'un utilisateur
Dépendances	❖ update_user_jetons () ❖ reload_jetons ()
Description	Vérifie que l'utilisateur n'essaie pas de miser plus de jeton qu'il a sur son compte
Retour	Met à jour la base de données

4.2.11 reload_jetons ()

Paramètres	Id_user : identifiant d'un utilisateur
Dépendances	❖ update_user_jetons () ❖ reload_jetons ()
Description	Vérifie que le nombre de jetons sur le compte de l'utilisateur est inférieur à 5000 et lui ajoute 25000 jetons si c'est le cas
Retour	Met à jour la base de données

4.2.12 take_back_half_bet_money_escroc ()

Paramètres	Id_user : identifiant d'un utilisateur
Dépendances	❖ bet_money : le nombre de jetons misés par un utilisateur au cours de la partie
Description	update_user_jetons ()
Retour	Met à jour la base de données

4.2.13 update_nb_games ()

Paramètres	Id_user : identifiant d'un utilisateur
Description	Augmente le compteur de parties jouées de l'utilisateur par 1
Retour	Met à jour la base de données

4.2.14 update_nb_victories ()

Paramètres	Id_user : identifiant d'un utilisateur
Description	Augmente le compteur de victoires de l'utilisateur par 1
Retour	Met à jour la base de données

4.2.15 add_winner ()

Paramètres	❖ id_user : identifiant d'un utilisateur ❖ nb_jetons : le nombre de jetons qu'il a gagné
Dépendances	❖ update_user_jetons () ❖ update_nb_victories ()
Description	Fonction_wrapper pour mettre à jour le joueur gagnant, augmenter son nombre de victoires et ajouter les jetons gagnés sur son compte
Retour	Met à jour la base de données

4.2.16 get_nb_jetons ()

Paramètres	Id_user : identifiant d'un utilisateur
Description	Retourne le nombre de jetons de l'utilisateur extrait de la base de données
Retour	Le nombre de jetons sur le compte de l'utilisateur

4.2.17 get_user_info ()

Paramètres	<code>Id_user : identifiant d'un utilisateur</code>
Description	Retourne les informations personnelles de l'utilisateur issues de la table User de la base de données
Retour	Un objet JSON contenant les informations personnelles non-sensibles de l'utilisateur, ses ids, pseudos, date de naissance et email

4.2.18 `get_user_stats ()`

Paramètres	<code>Id_user : identifiant d'un utilisateur</code>
Description	Retourne les statistiques de l'utilisateur issues de la table Score de la base de données
Retour	Un objet JSON contenant les statistiques de l'utilisateur, son id, nombre de jetons, nombre de parties jouées et nombre de parties gagnées

4.2.19 `classement_jetons ()`

Paramètres	Aucun
Description	Trie tous les utilisateurs en fonction du nombre de jetons et du nombre de victoires (en cas d'égalité de jetons sur leurs comptes) et retourne un array contenant des ids des utilisateurs dans le bon ordre
Retour	Un array de <code>id_users</code> dans l'ordre de classement par le nombre de jetons

4.2.20 `classement_victoires ()`

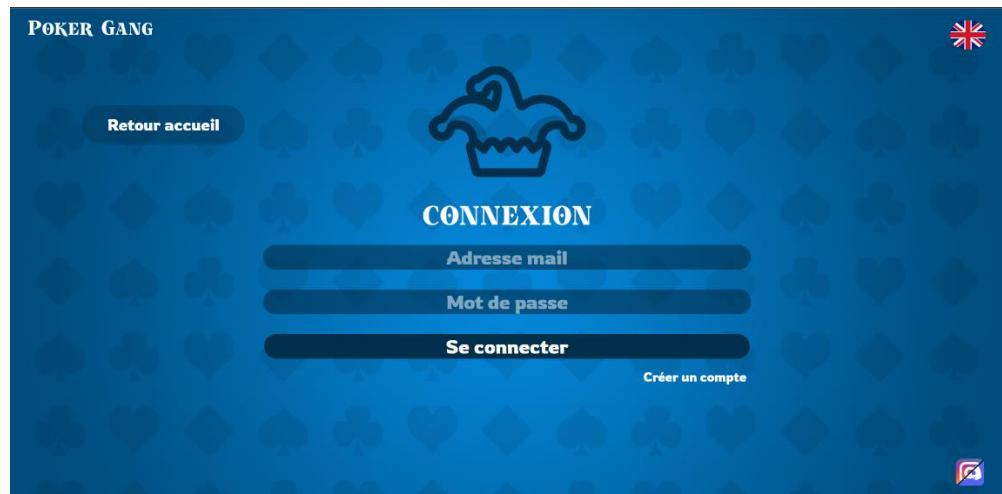
Paramètres	Aucun
Description	Trie tous les utilisateurs en fonction du nombre de victoires et du nombre de jetons (en cas d'égalité de victoires) et retourne un array contenant des ids des utilisateurs dans le bon ordre
Retour	Un array de <code>id_users</code> dans l'ordre de classement par le nombre de victoires

5 Frontend

5.1 Accueil



5.2 Connexion



5.3 Inscription



5.4 Menu principal



5.5 Règles du jeu

Règles du jeu

Le poker est une famille de jeux de cartes comprenant de nombreuses formules et variantes. Il se pratique à plusieurs joueurs avec un jeu généralement de cinquante-deux cartes et des jetons représentant les sommes mises. Les séquences de jeu alternent distribution de cartes et tours d'enchères.

Les règles de Poker Gang ne sont pas tout à fait identiques à celles d'un poker traditionnel, nous avons décidé de revisiter certaines parties afin de construire une variante. Notre jeu implémente des rôles pour chaque joueur, faisant référence à des membres d'un 'gang'.

Les points essentiels:

- Deux cartes, ne pouvant être vues que par le joueur qui les reçoit, sont distribuées à chaque joueur.
- Le croupier révèle cinq cartes - trois d'un coup, puis une quatrième, puis une cinquième - qui peuvent être utilisées par tous les joueurs pour former la meilleure main à cinq cartes possibles.
- Les joueurs misent chacun leur tour avant et après la distribution de chaque carte. Pour rester dans la main et voir la...

5.6 Classement

Rang	Joueur	Victoires	Jetons
1	JackSparrow	0	100000
2	jojo67	0	100000
3	test67	0	100000
4	toto101	0	100000
5	toto71	0	70917
6	toto69	2	59123
7	toto67	55	27762
8	toto79	5	18294

5.7 Interface



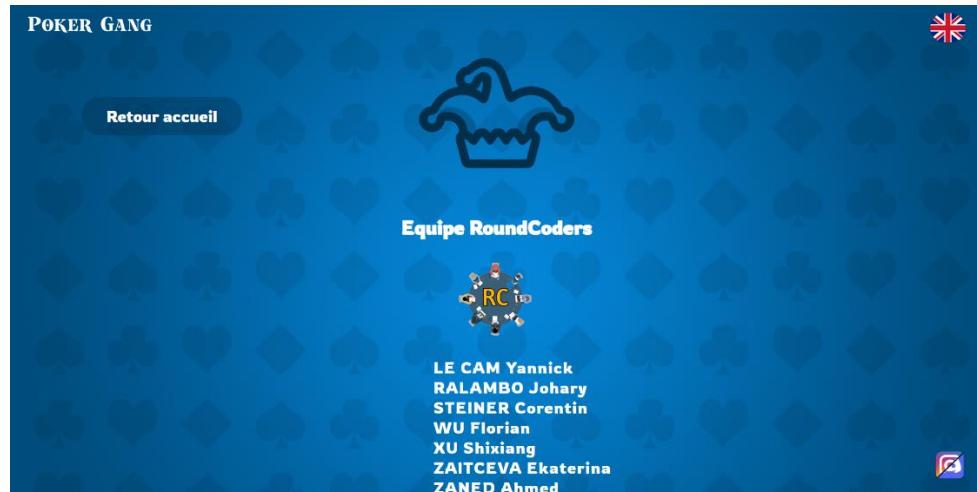
5.8 Changer mot de passe



5.9 Contact



5.10 Credits



6 Serveur

6.1.1 Les objets

Table
this.Mise_total=Mise_total
this.Mise_maximum=0
this.Mise_minimum=Mise_minimum
this.Nb_Joueurs=0
this.CartesSurTable=CartesSurTable
this.blind=0;
this.tour=0;
Cet objet permet de gérer tous les arguments présents dans une table de poker

Joueur
this.pseudo=pseudo
this.nb_jetons=nb_jetons
this.Mise=0
this.Coucher=false
this.isPlaying=false
this.isInGame=false
this.isPremierJoueur=false
Cet objet permet de gérer tous les arguments des joueurs sans les données sensibles du jeu (tel que la main et le rôle)

JoueurDetail
this.IdJoueur=IdJoueur
this.pseudo=pseudo
this.nb_jetons= nb_jetons
this.Mise=0

this.Mise_Tour=0
this.Coucher=false //Bool
this.Main=0
this.Role=-1//wip
this.Cd_Pouvoir=-1
this.isPlaying=false
this.isInGame=false
this.IdSocket=IdSocket
this.isPremierJoueur=false
Cet objet permet de gérer tous les arguments des joueurs avec toutes les informations essentielles pour la partie.

Update_Data
this.Table=Table//table
this.MonJoueur=MonJoueur//Joueur Detail
this.Joueur1=Joueur1//Joueur
this.Joueur2=Joueur2
this.Joueur3=Joueur3
this.Joueur4=Joueur4
this.Joueur5=Joueur5
Cet objet est fait pour que le front puisse afficher sans se compliquer la tâche, présent dans un tableau et envoyer chaque update_data de façon personnalisée à chacun. L'algorithme permettant de mettre en place le tableau d'Update_Data permet un affichage avec le bon emplacement du joueur parmi l'ordre de jeu.

Main
this.carte1={carte1Valeur,carte1Couleur}
this.carte2={carte2Valeur,carte2Couleur}
Cet objet permet de stocker les mains des joueurs de façon fluide pour le front, un élément de Joueur_Detail.

Spectateur_Update_Date
this.Table=Table
this.MonJoueur=Joueur1
this.Joueur1=Joueur2
this.Joueur2=Joueur3
this.Joueur3=Joueur4
this.Joueur4=Joueur5
this.Joueur5=Joueur6
Cet objet est semblable à Update_Data (mais pour les spectateurs) cependant il n'est composé que de Joueur et n'a pas de JoueurDetail pour ne pas que le spectateur n'est accès au données sensible de jeu tel (la Main et le Rôle)

6.1.2 Les routes

Routes 'get'
Route '/accueil' : Est une sécurité à une connexion par l'url a la page accueil sans s'être connecté, si l'utilisateur n'est pas connecté il est donc redirigé sur la page de connexion.
Route '/jeu' : Est une sécurité à une connexion par l'url a la page accueil sans s'être connecté, si l'utilisateur n'est pas connecté il est donc redirigé sur la page de connexion.
Route 'classement' : Est une sécurité à une connexion par l'url a la page accueil sans s'être connecté, si l'utilisateur n'est pas connecté il est donc redirigé sur la page de connexion.
Route '/lobby' : Est une sécurité à une connexion par l'url a la page accueil sans s'être connecté, si l'utilisateur n'est pas connecté il est donc redirigé sur la page de connexion.
Route '/connexion' : Si l'utilisateur est déjà connecté alors il est directement envoyé sur la page de l'accueil.
Route '/inscription' : Si l'utilisateur est déjà connecté alors il est directement envoyé sur la page de l'accueil.
Route 'logout' : Permet à l'utilisateur de se déconnecter (Par exemple pour changer de compte).
Route 'getMoreToken' : permet à l'utilisateur de récupérer des jetons une fois qu'il passe sous la barre des 5000 jetons.
Route '/getUserStat' : Permet au front d'afficher les statistiques de l'utilisateur.
Route '/getUserInfo' : Permet au front de récupérer les informations élémentaires de l'utilisateur (Pseudo , nombre de jetons , id_user , ...).
Route '/getClassementInfo' : Permet au front d'afficher le classement total des joueurs en fonction des jetons, cependant on peut voir le nombre de parties gagnées par le joueur.

Routes 'post'
Route '/connexion' : Demande à l'utilisateur d'envoyer un mail et un mot de passe et vérifie que le compte est juste, si juste alors l'utilisateur se fait rediriger sur la page d'accueil et le serveur enregistre les informations de jeu dans une session, Si l'utilisateur transmet des informations erronées alors il reste sur la page d'accueil.
Route '/inscription' : Demande à l'utilisateur de fournir un pseudo, un mail, un mot de passe, sa date de naissance. Après la véracité des informations le site transmet les informations a la BDD, or si il y a un refus de la BDD alors un message est envoyé pour annoncer à l'utilisateur qu'il n'a pas réussi à se connecter lui indiquant pourquoi (Sans trop de détail).
Route '/changeusername' : Demande à l'utilisateur son id et le nouveau username, cette requête permet de changer le pseudo de l'utilisateur.
Route '/verifierusername' : Demande à l'utilisateur un username, cette requête vérifie si l'username existe déjà dans la BDD.

6.1.3 Parcours des tableaux

findIndicePlayerParID (idJoueur)

Retourne l'indice du joueur dans les tableaux Joueur et JoueurDetail en prenant comme argument l'idjoueur , renvoie -1 si l'id n'existe pas

updateDesDonnees()

Actualise les tableaux Joueur et JoueurDetail avec les données présentes dans le jeu.

findIndicePlayer(sock)

Retourne l'indice du joueur dans les tableaux Joueur et JoueurDetail en prenant comme argument l'id_socket, renvoie -1 si le socket n'existe pas dans les tableaux.

trouveUnePlaceLibre()

Retourne l'indice dans les tableaux Joueur et JoueurDetail, l'indice qui contient un emplacement vide, s'il n'y a pas d'emplacement vide la fonction renvoie -1.

6.1.4 Gestion des réceptions

Socket connection

Ce type de socket permet au serveur de réceptionner les joueurs entrant dans le socket.

Socket disconnect

Ce type de socket permet au serveur de savoir quand un utilisateur se déconnecte de la partie jeu, par défaut il enclenche le mode déconnexion sauvage.

Socket initConnection

Ce type de socket permet au serveur, d'initialiser avec les données envoyées par le front (id_user , nb_jeton , pseudo) d'initialiser les tableaux joueur et joueurDetail et initialiser ou non en spectateur selon le nombre de joueurs déjà présent autour de la table. Autre condition c'est que si à l'arrivée du joueur il n'y a que 2 joueurs et qu'une partie n'est pas en train de se relancer alors la partie va être lancée.

Socket miser

Ce type de socket permet au serveur quand le front utilise la fonction "miser" en amont. La fonction émit fait un filtrage des informations transmises, dans le côté serveur il y a une lecture et une transmission des données au côté Jeu. Puis renvoie les données de la table avec le prochain joueur qui joue.

Socket message

Ce type de socket permet au serveur de récupérer un message venant d'un utilisateur afin de le transmettre à tous les autres utilisateurs autour de la table transformée en forme d'objet (Pseudo + message).

Socket se coucher

Ce type de socket permet au serveur de récupérer le joueur qui se couche envoyé par le front et le serveur le transmet à la partie jeu. Puis renvoie les données de la table avec le prochain joueur qui joue.

Socket passer

Ce type de socket permet au serveur de récupérer le joueur qui passe (checker) envoyé par le front et le serveur le transmet à la partie jeu. Puis renvoie les données de la table avec le prochain joueur qui joue.

Socket RejoindrePartie

Ce type de socket permet au serveur de détecter les joueurs qui rejouent en les mettant dans un tableau temporaire (réinitialiser dans la fonction RecuperationDesJoueurs).

Socket deconnexion

Ce type de socket est un message du front quand l'utilisateur se déconnecte de façon "propre", fait appel à la fonction quitter joueur (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket voyanteVisuCarte

Ce type de socket permet au serveur de récupérer les informations pour utiliser le pouvoir de la "voyante", renommé indicateur (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket voleurAction

Ce type de socket permet au serveur de récupérer les informations nécessaires pour l'utilisation du pouvoir du voleur (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket policeAction

Ce type de socket permet au serveur de récupérer les informations nécessaires pour l'utilisation du pouvoir du policier (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket usurpateurAction

Ce type de socket permet au serveur de récupérer les informations nécessaires pour l'utilisation du pouvoir de l'usurpateur (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket insolvableAction

Ce type de socket permet au serveur de récupérer les informations nécessaires pour l'utilisation du pouvoir de l'insolvable (les informations nécessaires sont détaillées dans la fonction l'utilisant).

Socket escrocAction

Ce type de socket permet au serveur de récupérer les informations nécessaires pour l'utilisation du pouvoir de l'escroc (les informations nécessaires sont détaillées dans la fonction l'utilisant).

6.1.5 Gestion des émissions

Socket escrocAction

Fonction permettant d'envoyer à chaque utilisateur les informations du(des) gagnant(s) (Pseudo avec sa main). Les infosGagnant sont pour la pluspart des cas en lien avec findPseudoGagnant()

Socket escrocAction

Fonction permettant d'envoyer à chaque joueur autour de la table la version personnalisée qui lui est destinée une version de l'objet Update_Data. Prend également en compte les joueurs spectateur en envoyant un objet Update_Data_Spec, contenant pas les infos sensibles pour le jeu (Les Mains des joueurs et le role).

Socket escrocAction

Fonction permettant d'envoyer l'action du joueur au front afin de faire l'animation adaptée.

Socket escrocAction

Fonction permettant de signaler qu'un joueur vient de se connecter avec son pseudo.

Socket escrocAction

Fonction permettant de signaler au front a quel la partie en cours est actuellement

6.1.6 Gestion de la partie jeu

prochainJoueurAJouer(indiceJoueurCourant)

Renvoie l'indice du prochain joueur à jouer dans la partie en prenant comme argument l'indice du joueur courant

initJoueurPresent()

Initialise dans les tableaux Joueur et JoueurDetail tous les joueurs qui vont commencer la partie.

resetStatutJoueur()

Remet à zéro tous les statuts des joueurs.

recuperationMainsJoueurs()

Récupération de la main de chaque joueur dans l'objet "serveur" MAIN (qui est dans "./Objet.js"), en sélectionnant chaque attribue

recuperationCartesTable(tour)

Permet de récupérer les cartes présentes sur la table afin de l'envoyer le nombre exact de carte à afficher pour le front pour ne pas que les joueurs puissent l'intercepter des informations pouvant faciliter la win en récupérant le tour actuel comme argument.

initNewTour(tour)

Quand un changement de tour s'opère dans la partie alors cette fonction récupère le tour en argument elle est déclenchée afin d'actualiser les actions correspondante

envoieCompteur()

Fonction qui émet un compteur au front entre 2 parties afin qu'il soit synchro

recuperationLeNbDeJetonLePlusBas()

Parcours de la liste entière afin de retourner la valeur de jetons le plus bas autour de la table actuellement

initPremierJoueur()

Se sert de l'attribut dans le jeu "First Shooter" afin de retrouver l'id du joueur pour le comparer à toute la table afin de trouver le bon premier joueur

isAlreadyHere(idJoueur)

Cette fonction regarde si le joueur est autour de la table grâce à son id(en argument), marche avec la feature "déconnexion sauvage"

findPseudoGagnant()

Renvoie les informations du gagnant (Pseudo + Mains)

resetMiseTour()

Reset les Mise_Tour de chaque joueur (est appelé entre chaque dans la fonction InitNewTour)

initBlind(blind,indiceJoueurPremier)
Initialisation des blinds et actualisation pour le front et dans le jeu en récupérant en argument la blind et indice du premier joueur

SuiteDeFinDePartie()
Est la suite de la détection de fin de partie, est utile de la séparer en 2 car la fonction SetTimeout permet de lancer un fonction après un temps donné

recuperationDesJoueurs()
Cette fonction laisse dans le tableau de joueur et joueursDetail tous les joueurs ayant cliqué sur rejouer, fait quitter ceux qui n'ont pas cliqué , et ajoute les spectateurs dans l'ordre d'arrivée de ces dernier

6.1.7 Gestion de la déconnexion des joueurs

Paramètres	idJoueurQuiQuitte : l'id du joueur qui quitte
Description	Supprime l'existence d'un joueur qui part en prenant comme argument l'id du joueur qui quitte, à la fois dans le jeu ainsi que dans les données faites pour transmettre au front. et Prends en considération toutes les situations possibles de chaque joueur avant de quitter.
Retour	Aucun

Paramètres	idJoueurQuiQuitte : l'id du joueur qui quitte
Description	Cette fonction détecte si un joueur a déjà été passer dans quitter joueur en prenant comme argument l'id du socket du joueur qui quitte, mais se fait appeler dans déco joueur.
Retour	Aucun

6.1.8 Gestion des rôles

envoiVisuVoyante(data)
Paramètres
Data - data[0] : pseudo du joueur ciblé - data[1] : socketid du joueur provenant
Description
Envoyer une carte de la main du joueur souhaité par la voyante (renommé indicateur par la suite).
Retour
Aucun

broadcastJoueurPrison(data)
Paramètres
Data - data[0] : pseudo du joueur ciblé

	- data[1] : socketid du joueur provenant
Description	Récupère le joueur ciblé et l'interdit d'utiliser son pouvoir jusqu'à la fin de la partie.
Retour	Aucun

broadcastJoueurUsurpateur(data)	
Paramètres	Data - data[0] : pseudo du joueur ciblé - data[1] : socketid du joueur provenant
Description	Récupère le joueur ciblé et clone son rôle jusqu'à la fin de la partie.

broadcastJoueurVoleur(data)	
Paramètres	Data - data[0] : pseudo du joueur ciblé - data[1] : socketid du joueur provenant - data[2] : l'indice dans la main de la carte du voleur qu'il veut se débarasser
Description	Récupère le joueur ciblé et l'indice de la carte du voleur dont il veut se séparer. Ensuite la fonction échange cette dernière avec l'une du joueur ciblé
Retour	Aucun

broadcastJoueurEscroc(data)	
Paramètres	Data - data[0] : socket.id du joueur courant
Description	Ajoute au joueur la moitié de ses mises depuis le début de la partie et le front se charge de le coucher
Retour	Aucun

broadcastJoueurInsolvable(data)	
Paramètres	Data - data[0] : socket.id de l'insolvable - data[1] : somme pour suivre
Description	Ajoute la moitié des jetons de la mise lorsque ce dernier suit avec son pouvoir
Retour	Aucun

recuperationRole(data)	
Paramètres	Aucun
Description	Initialise aléatoirement un rôle à chaque joueur en début de partie en faisant appel à la fonction du jeu
Retour	Aucun

7 Tests unitaires

7.1 Test unitaire Carte

Dans ce test unitaire, nous testons la bonne création des cartes et la bonne initialisation des valeurs.

Création de carte :

- ✓ Valeur() = undefined, couleur() = undefined
- ✓ Valeur(2) = 2, couleur(2) = 2
- ✓ Valeur(0) = undefined, couleur(0) = undefined
- ✓ Valeur(5) = undefined, couleur(0) = undefined
- ✓ Valeur(0) = undefined, couleur(2) = undefined

7.2 Test unitaire Dealer

Dans ce test unitaire, nous avons plusieurs groupes de test :

Création du paquet de carte :

- ✓ Le paquet contient exactement 52 cartes
- ✓ Pas de doublons, les cartes sont uniques
- ✓ La valeur des cartes sont comprise entre 0-52 et la couleur entre 1-4
- ✓ Production de paquet identique à chaque fois

Mélanger les cartes :

- ✓ Le paquet mélangé contient exactement 52 cartes
- ✓ Le paquet mélangé est différent du paquet original
- ✓ Toutes les cartes du paquet original se trouvent dans le paquet mélangé
- ✓ Pas de doublons
- ✓ Produit des paquets différents à chaque fois

Distribuer les cartes :

- ✓ Distribution des cartes aux joueurs
- ✓ Carte indisponible après la distribution

Initialisation des cartes sur la table :

- ✓ Initialise les cartes de la table
- ✓ Carte indisponible après l'initialisation des cartes sur la table

Révélation des cartes du joueur :

- ✓ Création rang et récupération des joueurs gagnants correcte

7.3 Test unitaire Jeu

Création joueur :

- ✓ Initialisation joueur
- ✓ Plus de place disponible

- ✓ Quitter joueur

Départ du jeu :

- ✓ Initialisation des cartes sur la table
- ✓ Initialisation des cartes du joueur

7.4 Test unitaire Mains

Création rang :

- ✓ Le rang est défini
- ✓ Le rang est valide

7.5 Test unitaire Rang

Création bonne valeur du rang :

- ✓ Une paire
- ✓ Deux paires
- ✓ Brelan
- ✓ Quinte
- ✓ Couleur
- ✓ Full
- ✓ Quads
- ✓ Quinte Flush
- ✓ Quinte Flush Royale

7.6 Test unitaire Role

Création du rôle joueur :

- ✓ Assignation du rôle joueur aléatoire

7.7 Test unitaire Escroc

Pouvoir Escroc :

- ✓ Récupération de la moitié de la mise

7.8 Test unitaire Usurpateur

Pouvoir Usurpateur :

- ✓ Copier rôle

7.9 Test unitaire Policier

Pouvoir Policier :

- ✓ Mettre en prison

7.10 Test unitaire Insolvable

Pouvoir Insolvable :

- ✓ Emprunter argent

7.11 Test unitaire Indicateur

Pouvoir Indicateur :

- ✓ Indiquer une info

7.12 Test unitaire Voleur

Pouvoir Voleur :

- ✓ Voler une carte

UFR

de mathématique

et d'informatique

Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

PROJET INTÉGRATEUR - POKERGANG DOCUMENT DE COMMUNICATION



Hayk ZARIKIAN – Community Manager

ÉQUIPE 6B - RoundCoders

Table des matières

1	Introduction.....	1
2	Public visé	1
3	Technique marketing	1
3.1	L'offre et la demande.....	1
3.2	Marketing avant lancement.....	1
3.3	Marketing pendant le déploiement en production.....	1
4	Rémunération envisageable	2
4.1	Rémunération par publicité.....	2
4.2	Rémunération par achats intégrés	2
4.3	Rémunération par dons.....	2
4.4	Rémunération par prix de vente.....	2
5	Communication avant déploiement en production	2
5.1	Introduction.....	2
5.2	Lancer un trailer pour annoncer le jeu.....	2
5.3	Participation à des salons de jeu vidéo	2
5.4	Publicités (réseaux sociaux, google, ...)	3
6	Communication pendant production	3
6.1	Introduction.....	3
6.2	Communication de maintenance	3
6.3	Communication d'une mise à jour importante	3
6.4	Partenariats / Collaboration	3
6.5	Axe d'amélioration pour faire vivre le jeu	3
7	Nous contacter	4
7.1	Introduction.....	4
7.2	Réseaux sociaux.....	4
7.3	Mail.....	4
8	Conclusion	4

1 Introduction

Ce document de communication est nécessaire afin d'établir la communication que nous allons entreprendre tout au long du jeu PokerGang avant et après son déploiement en production.

2 Public visé

Le public visé pour notre jeu est un public tout d'abord jeune car le thème « gang » du jeu est son style « cartoon » correspondent bien à des jeunes adultes, mais le jeu de poker est aussi bien adapté à un public plus âgé. Nous avons donc pu réaliser un jeu de poker original qui casse les codes du jeu de poker authentique qui est plus sérieux, tandis que PokerGang porte plus l'accent sur le poker « fun ».

3 Technique marketing

3.1 L'offre et la demande

Le poker est un jeu connu et apprécié depuis sa création dans les années 1800. Son caractère classique et intemporel ont fait de lui l'un des jeux les plus populaires dans le monde. Par conséquent, la demande est constante ce qui a engendré notre projet de PokerGang. Ce qui veut dire qu'en fonction des fonds récoltés nous pourrons améliorer le jeu, nous permettre de lancer une campagne publicitaire ou bien participer à des salons de jeux vidéo.

3.2 Marketing avant lancement

Avant le lancement, le projet Poker Gang est un projet sous forme de bénévolat ce qui engendre seulement un temps de travail, de la motivation et de la communication. Par ailleurs, nous priorisons d'abord l'augmentation de notre audience afin que la communauté puisse découvrir le jeu. Ensuite, la question est de savoir combien les utilisateurs sont prêts à payer pour y jouer ou sont-ils prêts à payer pour y jouer simplement et prendre en compte également notre implication.

3.3 Marketing pendant le déploiement en production

Tout d'abord, le jeu sera proposé gratuitement à la communauté mais selon son évolution au fil du temps, nous pourrons espérer intégrer une des rémunérations proposées dans la section [4 Rémunération envisageable](#). Par la suite, nous pourrions utiliser cette rémunération pour développement avancé du jeu et récompenser l'équipe RoundCoders.

4 Rémunération envisageable

4.1 Rémunération par publicité

Une des premières rémunérations possibles serait la publicité intégrée dans notre jeu, la meilleure manière d'être rémunéré indirectement lors du lancement du jeu.

4.2 Rémunération par achats intégrés

Un autre type de rémunération est l'achat intégré dans le jeu qui corresponds à des achats de jetons de poker par exemple.

4.3 Rémunération par dons

Nous pouvons aussi nous baser sur une rémunération par la communauté via des dons pour les personnes qui souhaitent contribuer au développement du jeu.

4.4 Rémunération par prix de vente

La rémunération par prix de vente reste aussi une possibilité mais elle restreint le public à envisagé si le jeu connaît une expansion importante.

5 Communication avant déploiement en production

5.1 Introduction

Cette partie est consacrée à la façon dont nous allons communiquer la sortie de notre jeu avant de le mettre en production pour cela nous avons plusieurs méthodes décrites ci-dessous.

5.2 Lancer un trailer pour annoncer le jeu

La réalisation d'un trailer pour annoncer le jeu est indispensable pour donner la nouvelle tendance et annoncer l'originalité de notre jeu. Nous avons donc réalisé un montage vidéo afin de proposer un trailer du lancement du jeu.

5.3 Participation à des salons de jeu vidéo

Nous pourrions envisager de participer à des salons de jeu vidéo afin de présenter notre jeu, le faire tester, avoir des avis, remarques afin de l'améliorer avant de mettre en

production. Nous avons pour cela conçu des flyers que nous pourrions distribuer dans ces salons afin de présenter les règles du jeu plus convenablement possible.

5.4 Publicités (réseaux sociaux, google, ...)

Il est aussi important de mettre en place les réseaux sociaux pour être plus proche de la communauté et répondre à leur commentaire et publier les actualités. Nous pourrions également faire de la publicité sur les réseaux ou bien utiliser google.

6 Communication pendant production

6.1 Introduction

Cette partie est consacré à la façon dont nous allons communiquer pendant la production notamment les maintenances, les collaborations que nous serions amenés à entreprendre.

6.2 Communication de maintenance

Lors d'une nouvelle version ou d'une mise à jour corrective, nous avons besoin de mettre le jeu en maintenance. Nous devons alors communiquer l'information en utilisant les mails des utilisateurs, l'annoncer sur le jeu avec une journée d'avance ou encore utiliser les réseaux sociaux.

6.3 Communication d'une mise à jour importante

Lors d'une mise à jour importante qui concerne un changement important nous devons faire une annonce de type vidéo comme un trailer des nouvelles fonctionnalités (nouveaux rôles, nouveaux modes, ...).

6.4 Partenariats / Collaboration

En fonction de l'évolution du jeu nous pourrions envisager des partenariats ou des collaborations avec d'autres compagnies.

6.5 Axe d'amélioration pour faire vivre le jeu

Nous avons énoncé de nombreux point d'amélioration dans le jeu ce qui va permettre à la communauté d'être toujours actif et rendre le jeu vivant. Pour cela, nous devront élaborer des publicités des maintenances comme annoncé dans les parties précédentes ci-dessus. Prévoir des nouvelles fonctionnalités dans le jeu est important dans la communication car elle permet d'exprimer de manière indirecte que le jeu est toujours d'actualité et dans les tendances.

7 Nous contacter

7.1 Introduction

Dans les sections précédentes, nous avons établi la manière dont nous allions communiquer avec la communauté, dans cette section nous allons étudier la manière dont nous allons permettre à la communauté de communiquer avec RoundCoders.

7.2 Réseaux sociaux

Les réseaux sociaux sera une manière à la communauté de nous contacter, de donner leurs avis.

Notre Instagram :



Notre serveur discord dédié à la communauté :

<https://discord.gg/b3bP997k>

7.3 Mail

Les utilisateurs peuvent nous contacter également via le mail « round.coders@gmail.com ».

8 Conclusion

Pour en conclure, nous avons assez bien établie notre projet de communication tout au long du jeu que ce soit avant ou pendant la production. Notre communication va évidemment varier selon les circonstances de notre audience et les demandes de la communauté.

UFR

de mathématique

et d'informatique

Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

PROJET INTÉGRATEUR - POKERGANG INTERACTION HOMME-MACHINE



ÉQUIPE 6B – RoundCoders

Table des matières

1	Introduction.....	1
2	Wireframes	1
2.1	Introduction.....	1
2.2	Wireframe par page.....	1
2.2.1	Page de départ.....	1
2.2.2	Page d'authentification (inscription / connexion)	2
2.2.3	Page d'accueil	2
2.2.4	Page de jeu	3
2.2.5	Page règles du jeu.....	3
2.2.6	Page classement	4
2.2.7	Page profil.....	4
2.2.8	Lien figma des wireframes.....	5
3	Design et Inspiration.....	5
4	Interactions et IHM.....	5
4.1	Inscription et connexion	5
4.2	Menu principal	7
4.3	Interface du jeu	8
4.4	Actions	10
4.4.1	Action miser.....	10
4.4.2	Action checker	11
4.4.3	Action se coucher	11
4.5	Pouvoir des rôles	12
4.6	Fin d'une partie.....	12
5	Performance et fluidité.....	13
6	Palette de couleur.....	13
7	Gestion d'erreurs	15
8	Accessibilité et Internationalisation	15
9	Sécurité.....	16
10	Conclusion	16

1 Introduction

Il est important de définir le contexte de la prise en main du jeu PokerGang qui est un jeu de poker avec un « look and feel » différent des standards utilisés dans les jeux de poker qui s'avère plutôt un style sérieux, festive. Dans notre cas, le jeu se présente comme un jeu fun pour s'amuser avec un style plutôt « cartoon » qui casse les codes et rend le style de jeu original par rapport à d'autres jeux de poker standards.

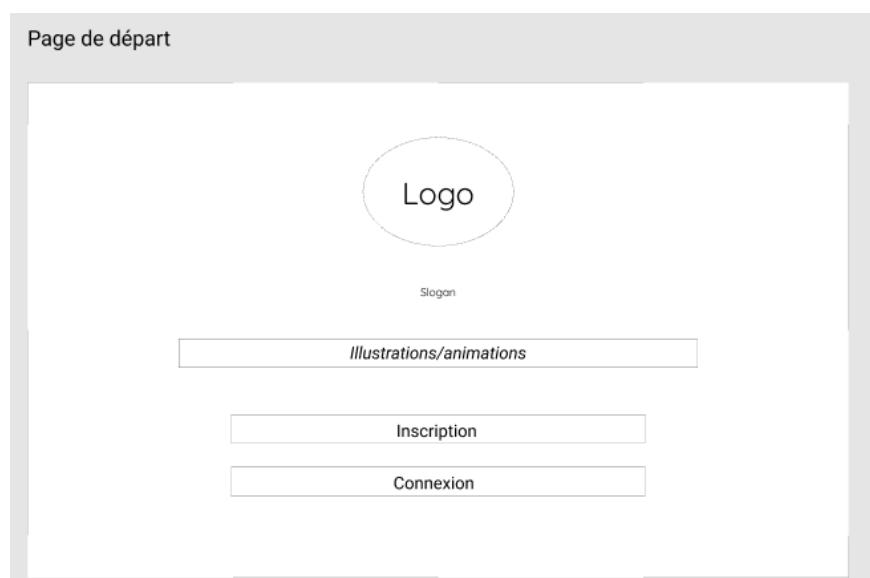
2 Wireframes

2.1 Introduction

Avant l'implémentation de notre interface graphique, nous avons dû élaborer de nombreux wireframes avec des designs différents afin d'évaluer le pour et le contre de chacune d'elles. Par la suite nous avons choisi un thème de design, constitué une palette de couleur, les actions à entreprendre, les formes afin de constituer l'identité du jeu PokerGang. Les images, les sons, les bruitages sont également sélectionnés précisément afin d'apporter une cohérence à l'ensemble car oui même les objets non visibles comme le son joue un rôle essentiel dans l'interface.

2.2 Wireframe par page

2.2.1 Page de départ



2.2.2 Page d'authentification (inscription / connexion)

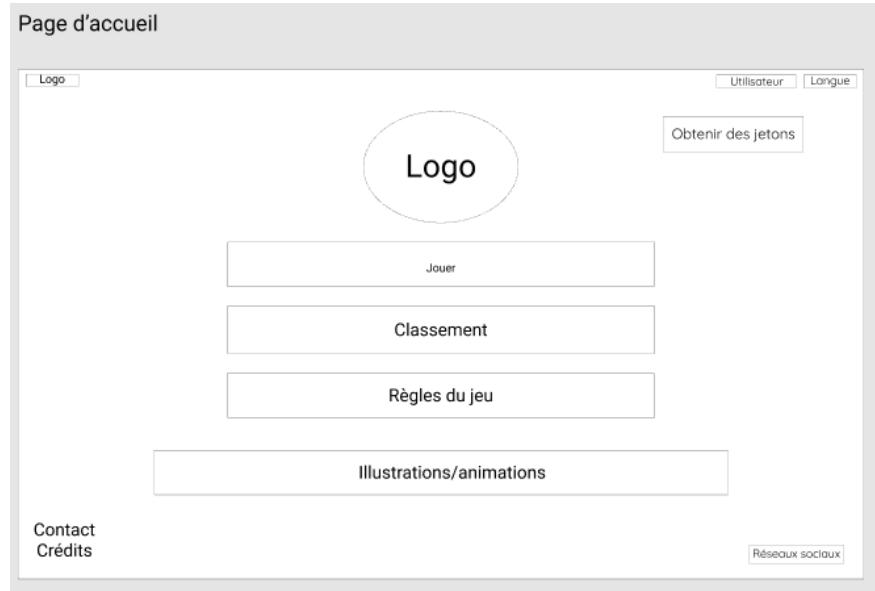
Page d'inscription

The registration page wireframe includes a logo placeholder, a "Retour accueil" button, and a "Lien" button. It features a large central logo placeholder with the word "Logo". Below it are four input fields labeled "Nom utilisateur", "Email", "Mot de passe", and "Date de naissance". A final input field at the bottom is labeled "Inscription".

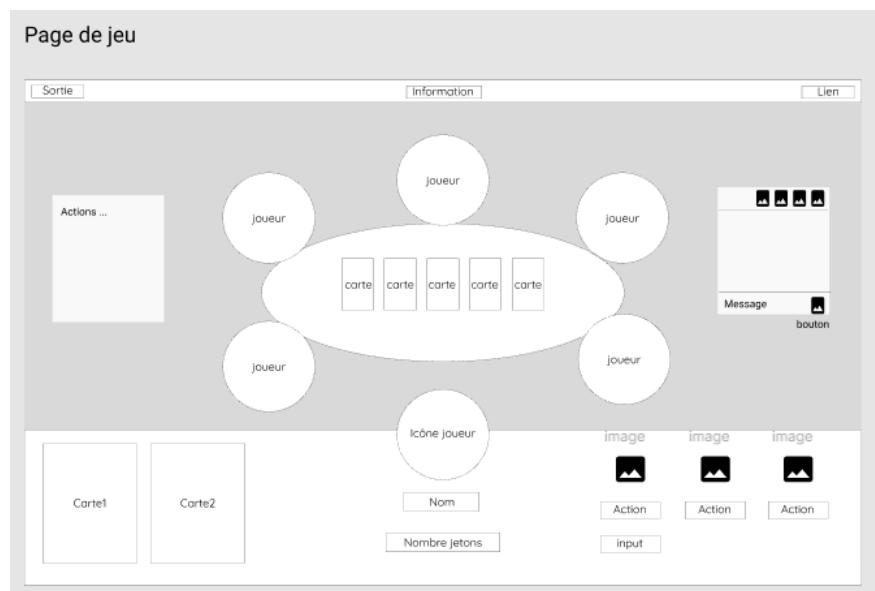
Page de connexion

The login page wireframe includes a logo placeholder, a "Retour accueil" button, and a "langue" button. It features a large central logo placeholder with the word "Logo". Below it are two input fields labeled "Nom utilisateur" and "Mot de passe". A final input field at the bottom is labeled "Se connecter".

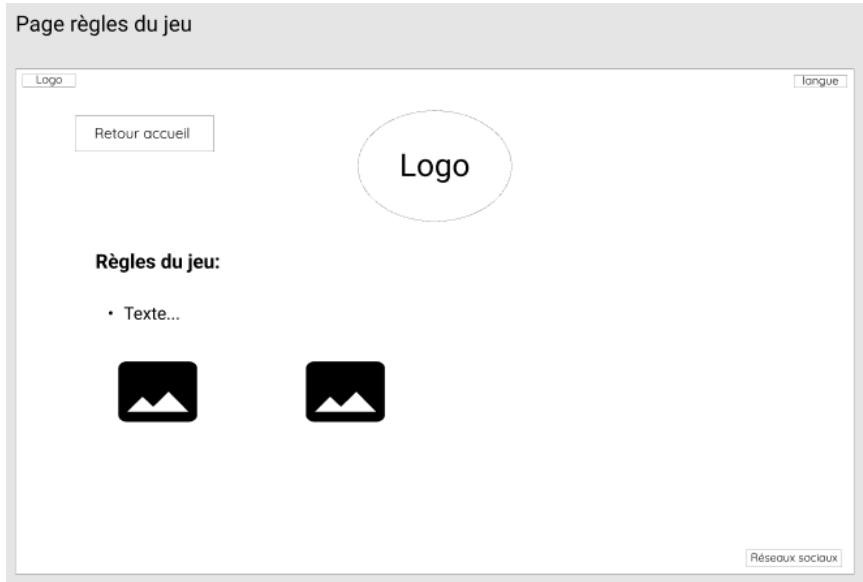
2.2.3 Page d'accueil



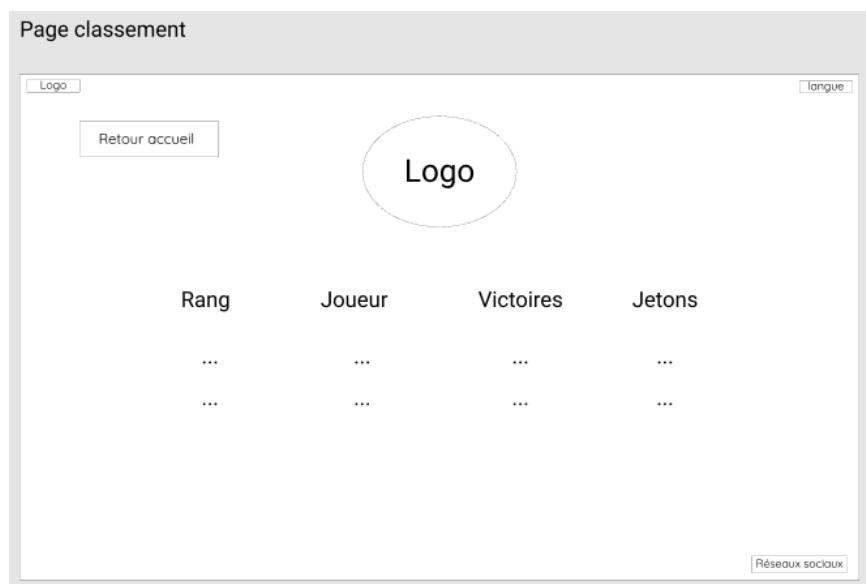
2.2.4 Page de jeu



2.2.5 Page règles du jeu



2.2.6 Page classement



2.2.7 Page profil



2.2.8 Lien figma des wireframes

Rendez-vous sur le lien suivant : [figma](#)

3 Design et Inspiration

Pour la réalisation de notre interface, nous avons choisi de nous inspirer du thème « Cartoon » qui constitue des images de type « dessin animé ». Par conséquent, le design principal reste très simple car les personnages sélectionnés apportent de la couleur et de l'animation ce qui s'immisce bien dans le décor simple. De plus, les cartes de poker apportent également de la couleur et des formes dans notre interface ce qui permet d'équilibrer l'ensemble.

4 Interactions et IHM

4.1 Inscription et connexion

La première étape à faire pour jouer à notre jeu est de créer un compte en cliquant sur le bouton « Rejoindre le GANG ».

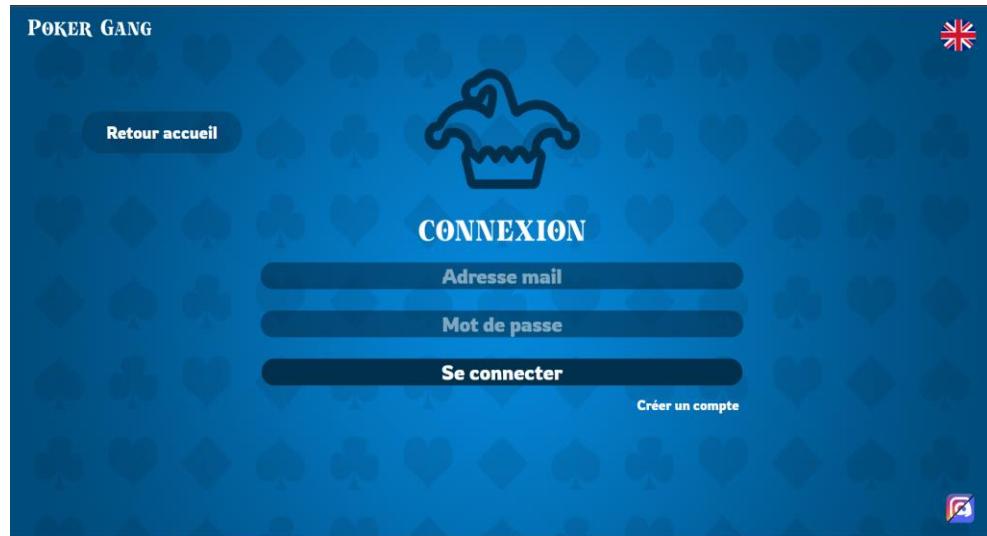


Une fois votre compte crée vous pourrez ensuite accéder à notre jeu en cliquant directement sur « Se connecter », après avoir rempli vos informations de connexion.

Le formulaire d'inscription se présente sous la forme suivante :

The screenshot shows the sign-up form for the Poker Gang website. The background is dark blue with a card suit pattern. At the top left is a "Retour accueil" button. In the center is the same stylized crown logo. Below it, the word "INSCRIPTION" is written in a bold, white, sans-serif font. The form consists of five input fields, each with a dark blue header bar and a white input area. The headers are "Nom d'utilisateur", "Adresse mail", "Mot de passe", "Date de naissance: jj/mm/aaaa", and "Inscription". To the right of the "Inscription" field is a link "J'ai déjà un compte" next to a user icon. In the bottom right corner of the form area, there is a small British flag icon.

Renseignez les différents champs, et si aucun message d'erreur est affiché, votre compte a bien été créé. Vous serez ensuite redirigé vers la page de connexion pour vous connecter avec le compte que vous venez de créer :



4.2 Menu principal

Après vous êtes connecté, vous serez redirigé vers le menu principal :



Sur cette page s'affiche votre pseudo, ainsi que votre nombre de jetons actuel (5). Pour les recharger, vous pouvez cliquer sur le bouton situé en dessous (4). Cependant, votre nombre de jetons devra être inférieur à 5000 pour en obtenir des nouveaux.

Sur cette page, vous pouvez accéder à différentes pages : les règles du jeu (3), la page de classement (2), la page de votre profil (5), une page de formulaire pour nous contacter, et une page de crédits (6).

En cliquant sur le drapeau en haut à droite, vous pouvez traduire le site en anglais. Pour traduire en français, vous n'aurez qu'à cliquer sur le drapeau français qui aura remplacé l'anglais.

En dessous du bouton Règles du jeu (3), vous avez un aperçu des six rôles disponibles. Le nom de chaque personnage est disponible lors du survol de la souris.

Un lien Linktree est disponible en cliquant sur le logo en bas à droite de la page. Ce lien vous permettra de nous rejoindre sur Instagram et Discord.

4.3 Interface du jeu

En cliquant sur « Jouer », depuis le menu principal, vous serez amené vers une page intermédiaire afin de rejoindre une partie. Cliquez sur le bouton, vous rejoindrez alors une partie.



Cet affichage correspond à la visualisation d'attente de partie. Dans le cas ci-dessus, aucune partie n'est actuellement en cours, vous devez donc attendre des joueurs pour jouer avec eux et démarrer une partie.

Sur cette page, plusieurs informations sont disponibles. Dans la partie basse de l'interface, sont affichées les informations liées à votre joueur. Les cartes (1) actuellement faces cachées correspondent aux deux cartes possédées par votre joueur, et invisibles pour les autres joueurs.

Les informations situées dans le cadre (2) correspondent aux données de votre joueur : votre pseudo, vos jetons en jeu, et vos jetons restants. L'emplacement avec le point d'interrogation vous permettra de visualiser le rôle obtenu, dès que la partie commence. La boule de cristal située en-dessous vous servira pour activer le pouvoir de ce rôle.

Les trois boutons présents dans le cadre (3) sont les trois actions disponibles pour votre joueur : miser, checker, se coucher. L'utilisation de ces boutons sera détaillée dans la partie ci-dessous.



Vous pouvez désormais visualiser le rôle qui vous a été attribué (l'insolvable dans cet exemple). Lorsque c'est à votre tour de jouer, le chronomètre (1) est enclenché, et l'image de votre rôle (2) est mise en valeur. Vous devez donc effectuer votre action durant un temps imparti. La boule de cristal est également illuminée si vous pouvez utiliser votre pouvoir. Cependant, il n'est utilisable qu'une seule fois par partie. Après son utilisation, la boule de cristal apparaîtra grisée.

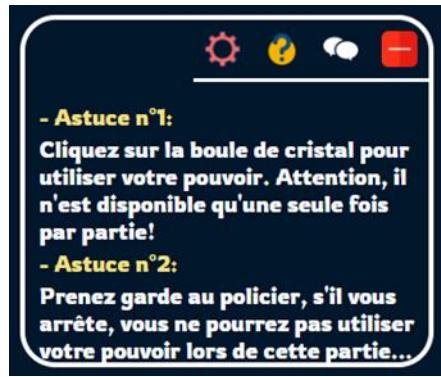
Sur la partie gauche de l'interface (3) est affichée une fenêtre qui affiche dynamiquement le déroulement du jeu (actions des joueurs, changement de tour, ...)

Sur la partie droite de l'interface (4) une fenêtre contenant plusieurs onglets vous propose :

- Un onglet chat (pour discuter en direct avec les autres joueurs) :



- Un onglet d'astuces (utile pour les débutants) :



- Un onglet paramètres (pour changer le volume du jeu et changer la langue)
-



Si cette fenêtre vous dérange, vous pouvez la réduire à tout moment en cliquant sur l'icône rouge la plus à droite.

Ensuite, vous pouvez observer cinq cartes placées faces cachées situées sur la table (5). Elles constituent les cartes « ouvertes » et seront retournées au fur et à mesure du déroulement de la partie.

Autour de cette table, sont placés les joueurs actuellement en jeu (6). La photo du joueur s'illuminera lorsque ce sera à son tour de jouer. Pour chaque joueur est affiché son nombre de jetons pour le tour actuel, ainsi que son nombre de jetons restants.

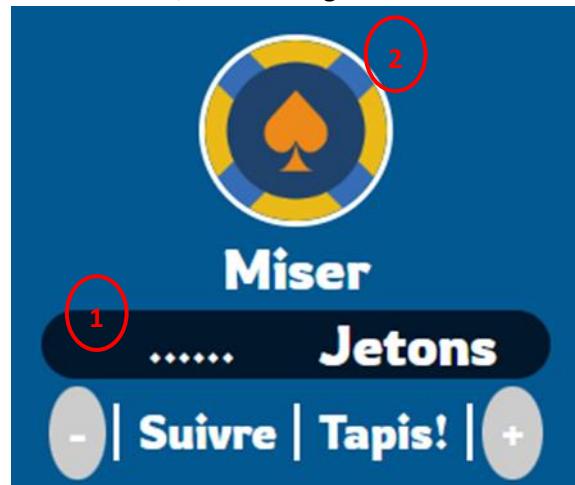
Le nombre de total de jetons en jeu, dans le pot est affiché en haut de l'interface (7), et correspond au montant total de jetons que le gagnant remportera à la fin de la partie.

4.4 Actions

4.4.1 Action miser

Lorsque c'est à votre tour de joueur, trois choix s'offrent à vous : miser, checker, se coucher.

Si vous choisissez de miser, vous interagissez avec les boutons ci-dessous :



Lors de la mise, vous mettez en jeu le nombre de jetons que vous avez renseigné dans le champ de saisie (1). Si vous souhaitez suivre (miser le même montant que le joueur précédent), ou faire tapis (miser tous vos jetons), vous pouvez cliquer sur les deux boutons associés. Il prérempliront le champ de saisie. Une fois le champ de saisie rempli, cliquez sur le bouton miser (2) pour mettre les jetons en jeu.

4.4.2 Action checker

L'autre action disponible, « checker » se situe à droite du bouton miser :



Si aucune mise n'a été effectuée, vous pouvez choisir de « checker » et ne pas miser de jetons. Cependant, vous ne pouvez pas checker si des joueurs ont misé des jetons avant vous.

4.4.3 Action se coucher

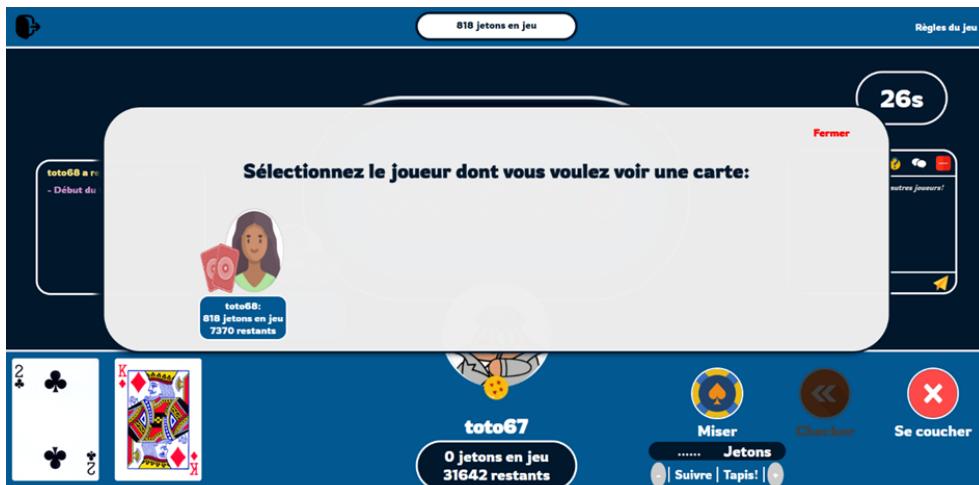
Il peut arriver lors d'une partie que votre main ne vous convienne pas. Il est alors possible d'abandonner la partie actuelle et perdre le nombre de jetons en jeu :



Pour se coucher vous cliquez alors sur le bouton correspondant. Attention, cette action est irréversible.

4.5 Pouvoir des rôles

Notre jeu de poker intègre des rôles avec des pouvoirs spécifiques. Lors de votre tour, vous pouvez utiliser ce pouvoir en cliquant sur la boule de cristal. Ensuite, selon le pouvoir, un message apparaîtra pour vous demander confirmation, ou pour choisir une victime de votre pouvoir. Par exemple pour « l'infiltrer », voici la fenêtre qui vous permet de choisir le joueur dont vous voulez voir une carte.



Vous pouvez aussi changer d'avis et décider de ne pas utiliser votre pouvoir en cliquant sur « fermer ». Si au contraire vous utilisez le pouvoir, celui-ci n'est plus disponible, et vous devez attendre la prochaine partie pour obtenir un nouveau pouvoir.

4.6 Fin d'une partie

Une fois le cinquième tour achevé, ou que tous les joueurs se sont couchés et ne laissant qu'un seul joueur « actif », la partie se termine. Le gagnant, ses cartes, ainsi que le montant remporté est affiché :



Ensuite, vous pouvez choisir de rejouer ou non une partie. Si vous souhaitez rejouer cliquez sur le bouton « prêt » durant le temps imparti. Si vous n'effectuez aucune action durant le temps imparti, vous serez redirigé vers le menu principal :



Pour quitter le jeu et revenir au menu principal, vous pouvez soit attendre la fin du compteur, ou soit directement cliquer sur la porte de sortie en haut à gauche (1).

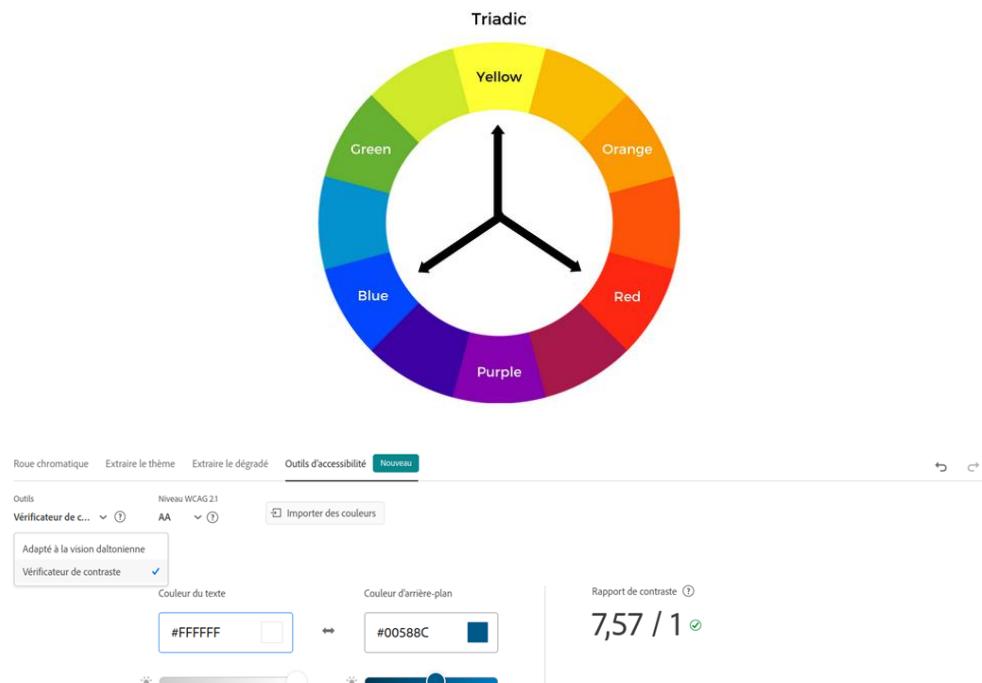
5 Performance et fluidité

La performance et la fluidité est une phase importante lors de la prise en compte de l'interface Homme-Machine car sans la fluidité le jeu ne peut pas être ergonomique malgré une interface digne de la perfection. Il est donc important pour nous d'optimiser la fluidité afin que le jeu soit jouable et appréciable.

6 Palette de couleur

Nous avons réalisé notre palette de couleur en se basant sur 3 couleurs avec un agencement triadique qui sont les suivants : bleu, jaune et rouge. Elles peuvent être plus ou moins claire. Nous avons décidé d'avoir seulement 1 couleur principale qui est le bleu car nous avions trois options

possibles : vert foncé, rouge et bleu, ce sont les couleurs généralement associées avec le poker. Comme notre but est de créer un jeu plutôt amusant, nous avons considéré que le rouge était trop agressif. Le vert foncé est le choix classique, mais aussi trop répandu. Nous avons donc choisi le bleu. Cependant, nous avons également choisi 2 couleurs secondaires le jaune et le rouge afin que l'interface ne soit pas fatigant au niveau des yeux, reste cohérent dans l'ensemble et sobre. Nous avons également deux couleurs complémentaire le noir et le blanc.



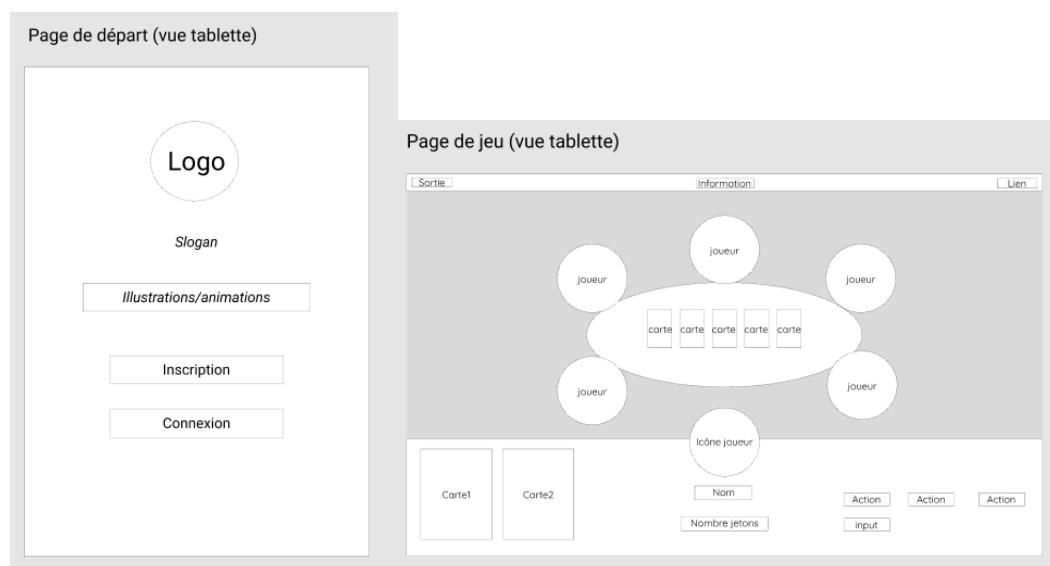
7 Gestion d'erreurs

Notre interface aborde également le sujet de la gestion d'erreur car en effet lors d'une saisie incorrecte lors de la connexion ou bien de l'inscription, nous affichons un message d'erreur en indiquant le problème en question. De la même manière, lorsque nous nous retrouvons en pleine partie et que nous voulons déposer des jetons supérieurs de ce que nous avons.

8 Accessibilité et Internationalisation

L'accessibilité a bien été prise en compte dans notre jeu car nous avons beaucoup réfléchi à comment placer nos boutons, dans quelle forme afin de ne pas impacter l'accessibilité notamment la visibilité. Par exemple, lorsque nous affichons les règles du jeu en pleine partie nous faisant attention de ne pas rendre cette action bloquant afin d'éviter de gâcher l'expérience de jeu et de nuire à la visibilité. Le jeu est également responsive design ce qui permet de réduire la fenêtre si besoin ou encore de jouer sur une autre plateforme avec un accès à internet. Nous avons également internationalisé notre jeu en y intégrant la possibilité de changer de langue (anglais ou français) pour viser les joueurs internationaux et de ne pas restreindre la compréhension de notre jeu pour les joueurs qui ne comprennent pas le français. Nous faisons aussi attention aux personnes atteintes d'handicap visuel car notre palette de couleur est adaptée au daltonisme.

Quelques wireframes pour tablette :



Quelques résultats pratiques sur mobile (Android) :



9 Sécurité

La sécurité peut aussi être abordée par l'interface Homme-Machine car nous avons de bloquer certaines actions qui ne sont pas autorisées à un certain temps dans le jeu. Par exemple, le joueur ne doit pas pouvoir faire une action tant que le jeu n'a pas débuté ou tant que ce n'est pas son tour de jouer.

10 Conclusion

Pour en conclure, l'interface de notre jeu est très importante car c'est la première image que nous donnons à notre jeu. Le design est la première chose que l'utilisateur prête attention en analysant les couleurs, la forme, les interactions possibles et bien d'autres aspects interactifs, il est donc essentiel que le design ne soit pas provocateur, reste simple afin que les utilisateurs s'adaptent car les utilisateurs auront l'habitude de voir pendant des heures et des jours le même design s'ils jouent activement au jeu. Mise à part le design, les interactions d'objets est une autre partie importante car c'est une partie qui interroge la fluidité, la facilité à comprendre, l'ergonomie, la mise en forme et selon les cas ces caractéristiques peuvent varier. Dans une autre partie, nous avons aussi pensé à l'accessibilité, la gestion d'erreur et la sécurité. Tous les paramètres décrit ci-dessus font partie de l'identité de l'interface homme-machine de notre jeu.

2 – 6 Joueurs



18 ans et plus



POKER GANG

Notice d'utilisation

&

Règles du jeu



Sommaire

Poker Gang	3
- Les membres du Gang	3
Inscription & Connexion	4
Menu principal	6
Interface du jeu	7
Comment jouer ?	8
- Action miser	10
- Action checker.....	10
- Action Se coucher.....	11
- Pouvoir des rôles	11
- Fin d'une partie	12
Rappel des règles du Poker.....	13
- Les points essentiels.....	13
- Actions disponibles pour chaque joueur	14
- Déroulement d'une partie	14
- Le classement des mains	14



Poker Gang

Poker Gang est un jeu de poker revisité, proposant une expérience ludique et amusante grâce à l'ajout de rôles, tous en rapport avec le thème « Gang » (d'où le nom du jeu). Ses règles et son fonctionnement sont ceux du poker Texas hold'em, avec quelques variantes. Pour faire simple, le but du jeu est d'avoir de meilleures cartes que ses adversaires, et ainsi remporter les jetons en jeu. Utilisez alors stratégiquement les pouvoirs des rôles pour mettre toutes les chances de votre côté.

Cette notice va vous permettre de comprendre comment jouer à notre jeu et présenter toutes ses fonctionnalités.

- Les membres du Gang



L'escroc : peut récupérer la moitié de sa mise en se couchant.



Le flic : Peut mettre en prison un joueur afin qu'il ne puisse pas utiliser son pouvoir durant la partie.



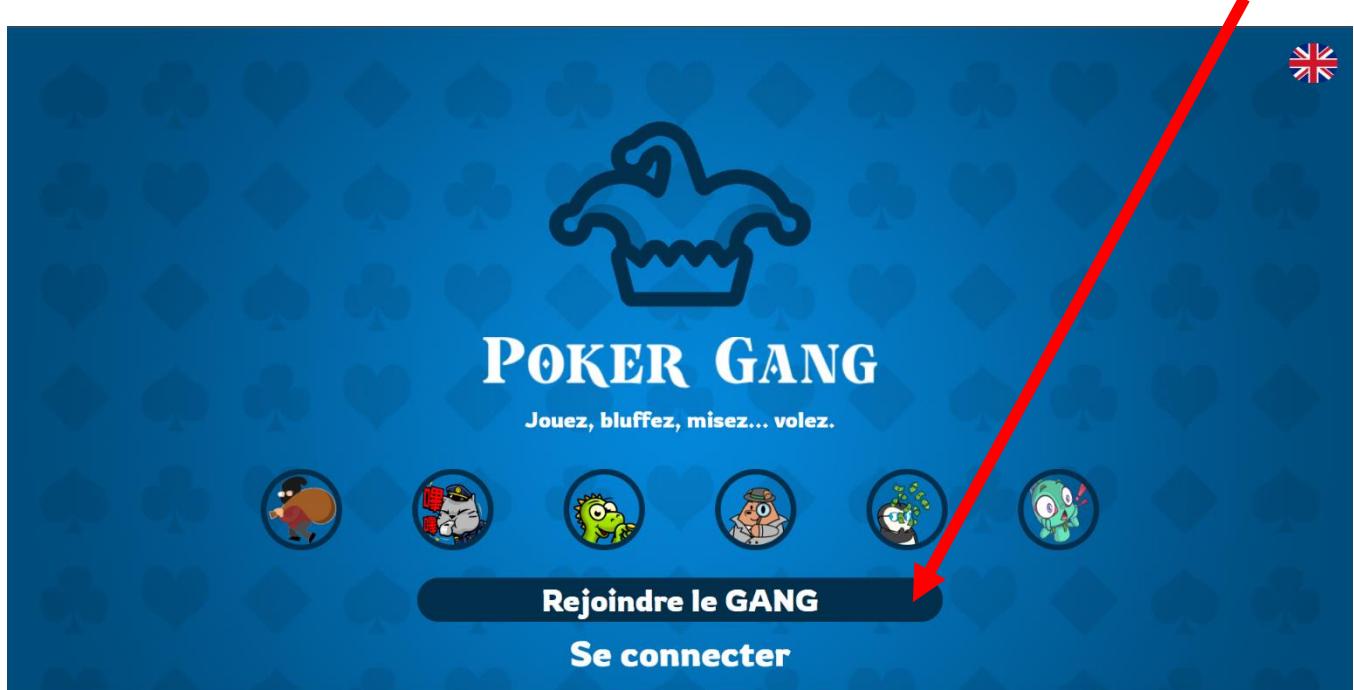
L'infiltré : Peut voir la carte d'un des joueurs.



Le voleur :
Peut échanger une de ses cartes contre l'une d'un de ses adversaires, sans la connaître.

Inscription & Connexion

La première étape à faire pour jouer à notre jeu est de créer un compte :



Une fois votre compte créé vous pourrez ensuite accéder à notre jeu en cliquant directement sur « Se connecter », après avoir rempli vos informations de connexion.

Le formulaire d'inscription se présente sous la forme suivante :

The screenshot shows the registration page for POKER GANG. At the top left is the logo "POKER GANG" and at the top right is the British flag icon. Below the logo is a large crown icon. The main title "INSCRIPTION" is centered above four input fields: "Nom d'utilisateur", "Adresse mail", "Mot de passe", and "Date de naissance: jj/mm/aaaa". There is also a checkbox labeled "Inscription". At the bottom left is a link "J'ai déjà un compte" and a small "Facebook" icon.

Renseignez les différents champs, et si aucun message d'erreur est affiché, votre compte a bien été créé. Vous serez ensuite redirigé vers la page de connexion pour vous connecter avec le compte que vous venez de créer :

The screenshot shows the connection page for POKER GANG. At the top left is the logo "POKER GANG" and at the top right is the British flag icon. Below the logo is a large crown icon. The main title "CONNEXION" is centered above three input fields: "Adresse mail", "Mot de passe", and "Se connecter". At the bottom right is a link "Créer un compte" and a small "Facebook" icon.

Menu principal

Après vous être connecté, vous serez redirigé vers le menu principal :



Sur cette page s'affiche votre pseudo, ainsi que votre nombre de jetons actuel (5). Pour les recharger, vous pouvez cliquer sur le bouton situé en dessous (4). Cependant, votre nombre de jetons devra être inférieur à 5000 pour en obtenir des nouveaux.

Sur cette page pouvez accéder à différentes pages : les règles du jeu (3), la page de classement (2), la page de votre profil (5), une page de formulaire pour nous contacter, et une page de crédits (6).

En cliquant sur le drapeau en haut à droite, vous pouvez traduire le site en anglais. Pour traduire en français, vous n'aurez qu'à cliquer sur le drapeau français qui aura remplacé l'anglais.

En dessous du bouton Règles du jeu (3), vous avez un aperçu des six rôles disponibles. Le nom de chaque personnage est disponible lors du survol de la souris.

Un lien Linktree est disponible en cliquant sur le logo en bas à droite de la page. Ce lien vous permettra de nous rejoindre sur Instagram et Discord.

Interface du jeu

En cliquant sur « Jouer », depuis le menu principal, vous serez amené vers une page intermédiaire afin de rejoindre une partie. Cliquez sur le bouton, vous rejoindrez alors une partie :



Cet affichage correspond à la visualisation d'attente de partie. Dans le cas ci-dessus, aucune partie n'est actuellement en cours, vous devez donc attendre des joueurs pour jouer avec eux et démarrer une partie.

Sur cette page, plusieurs informations sont disponibles. Dans la partie basse de l'interface, sont affichées les informations liées à votre joueur. Les cartes (1) actuellement faces cachées correspondent aux deux cartes possédées par votre joueur, et invisibles pour les autres joueurs.

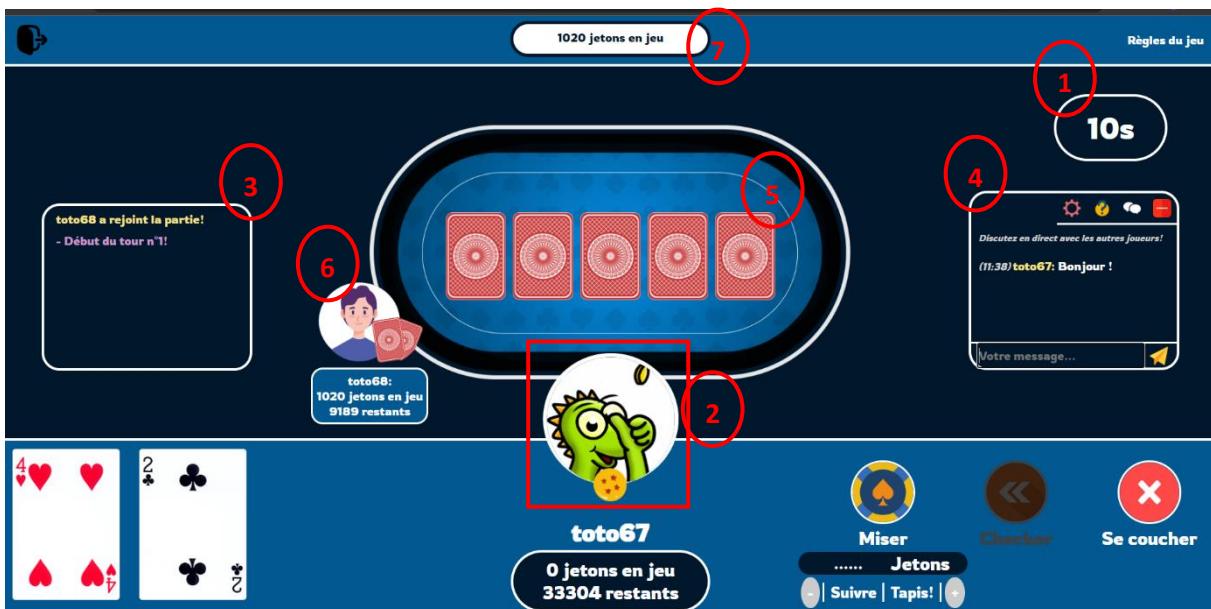
Les informations situées dans le cadre (2) correspondent aux données de votre joueur : votre pseudo, vos jetons en jeu, et vos jetons restants. L'emplacement avec le point d'interrogation vous permettra de visualiser le rôle obtenu, dès que la partie commence. La boule de cristal située en-dessous vous servira pour activer le pouvoir de ce rôle.

Les trois boutons présents dans le cadre (3) sont les trois actions disponibles pour votre joueur : Miser, checker, se coucher. L'utilisation de ces boutons sera détaillée dans la partie ci-dessous.

Comment jouer ?

Pour qu'une partie commence, un joueur doit rejoindre votre table, ou vous devez rejoindre la table d'un joueur en attente de joueurs. Si vous rejoignez une table dont une partie est déjà en cours, vous serez spectateur, jusqu'à la fin de celle-ci.

Voici l'interface affichée lorsque la partie démarre :



Vous pouvez désormais visualiser le rôle qui vous a été attribué (l'insolvable dans cet exemple). Lorsque c'est à votre tour de jouer, le chronomètre (1) est enclenché, et l'image de votre rôle (2) est mise en valeur. Vous devez donc effectuer votre action durant un temps imparti. La boule de cristal est également illuminée si vous pouvez utiliser votre pouvoir. Cependant, il n'est utilisable qu'une seule fois par partie. Après son utilisation, la boule de cristal apparaîtra grisée.

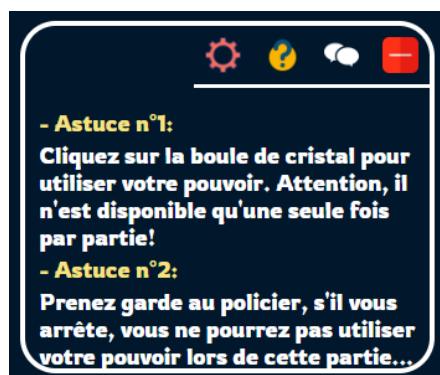
Sur la partie gauche de l'interface (3) est affichée une fenêtre qui affiche dynamiquement le déroulement du jeu (actions des joueurs, changement de tour, ...).

Sur la partie droite de l'interface (4) une fenêtre contenant plusieurs onglets vous propose :

- un onglet chat (pour discuter en direct avec les autres joueurs) :



- un onglet d'astuces (utile pour les débutants) :



- un onglet paramètres (pour changer le volume du jeu et changer la langue) :



Si cette fenêtre vous dérange, vous pouvez la réduire à tout moment en cliquant sur l'icône rouge la plus à droite.

Ensuite, vous pouvez observer cinq cartes placées faces cachées situées sur la table (5). Elles constituent les cartes « ouvertes » et seront retournées au fur et à mesure du déroulement de la partie.

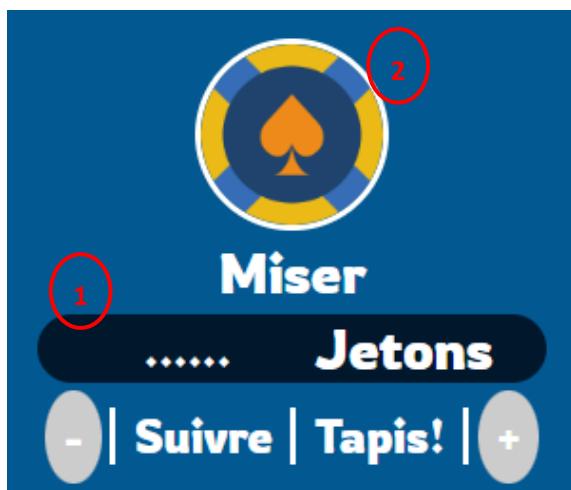
Autour de cette table, sont placés les joueurs actuellement en jeu (6). La photo du joueur s'illuminera lorsque ce sera à son tour de jouer. Pour chaque joueur est affiché son nombre de jetons pour le tour actuel, ainsi que son nombre de jetons restants.

Le nombre de total de jetons en jeu est affiché en haut de l'interface (7), et correspond au montant total de jetons que le gagnant remportera à la fin de la partie.

- Action miser

Lorsque c'est à votre tour de jouer, trois choix s'offrent à vous : miser, checker, se coucher.

Si vous choisissez de miser, vous interagissez avec les boutons ci-dessous :



Lors de la mise, vous mettez en jeu le nombre de jetons que vous avez renseigné dans le champ de saisie (1). Si vous souhaitez suivre (miser le même montant que le joueur précédent), ou faire tapis (miser tous vos jetons), vous pouvez cliquer sur les deux boutons associés. Ils prérempliront le champ de saisie. Une fois le champ de saisie rempli, cliquez sur le bouton Miser (2) pour mettre les jetons en jeu.

- Action checker

L'autre action disponible, « checker » se situe à droite du bouton miser :



Si aucune mise n'a été effectuée, vous pouvez choisir de « checker » et ne pas miser de jetons. Cependant, vous ne pouvez pas checker si des joueurs ont misé des jetons avant vous.

- Action Se coucher

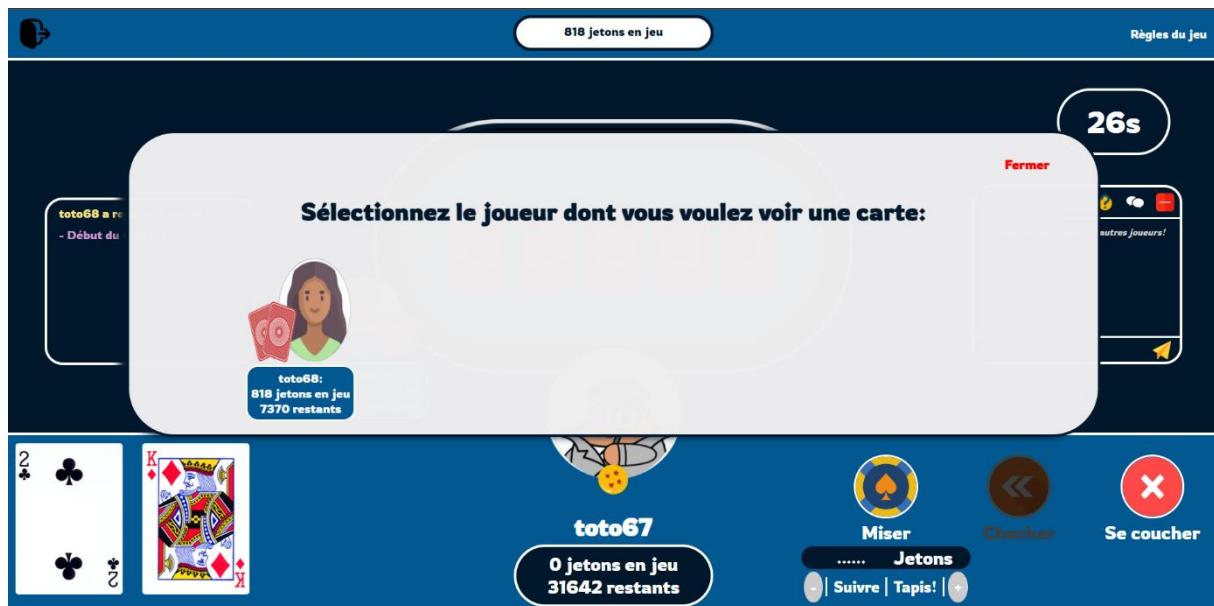
Il peut arriver lors d'une partie que votre main ne vous convienne pas. Il est alors possible d'abandonner la partie actuelle et perdre le nombre de jetons en jeu :



Pour se coucher vous cliquez alors sur le bouton correspondant. Attention, cette action est irréversible.

- Pouvoir des rôles

Comme énoncé précédemment, notre jeu de poker intègre des rôles avec des pouvoirs spécifiques. Lors de votre de tour, vous pouvez utiliser ce pouvoir en cliquant sur la boule de cristal. Ensuite, selon le pouvoir, un message apparaîtra pour vous demander confirmation, ou pour choisir une victime de votre pouvoir. Par exemple pour « l'infiltré », voici la fenêtre qui vous permet de choisir le joueur dont vous voulez voir une carte :



Vous pouvez aussi changer d'avis et décider de ne pas utiliser votre pouvoir en cliquant sur « fermer ». Si au contraire vous utilisez le pouvoir, celui-ci n'est plus disponible, et vous devez attendre la prochaine partie pour en obtenir un nouveau.

- Fin d'une partie

Une fois le cinquième tour achevé, ou que tous les joueurs se sont couchés et ne laissant qu'un seul joueur « actif », la partie se termine. Le gagnant, ses cartes, ainsi que le montant remporté est affiché :



Ensuite, vous pouvez choisir de rejouer ou non une partie. Si vous souhaitez rejouer cliquez sur le bouton « prêt » durant le temps imparti. Si vous n'effectuez aucune action durant le temps imparti, vous serez redirigé vers le menu principal :



Pour quitter le jeu et revenir au menu principal, vous pouvez soit attendre la fin du compteur, ou soit directement cliquer sur la porte de sortie en haut à gauche (1).

Rappel des règles du Poker

Le poker est une famille de jeux de cartes comprenant de nombreuses formules et variantes. Il se pratique à plusieurs joueurs avec un jeu généralement de cinquante-deux cartes et des jetons représentant les sommes misées. Les séquences de jeu alternent distribution de cartes et tours d'enchères.

Les règles de Poker Gang ne sont pas tout à fait identiques à celles d'un poker traditionnel, nous avons décidé de revisiter certaines parties afin de construire une variante. Notre jeu implémente des rôles pour chaque joueur, faisant référence à des membres d'un « gang ».

Ces règles sont disponibles sur notre site depuis la page « Règles du jeu », ou directement accessibles sur l'interface en cliquant sur le bouton « Règles du jeu ».

- Les points essentiels

- Deux cartes, ne pouvant être vues que par le joueur qui les reçoit, sont distribuées à chaque joueur.
- Au début de la partie, cinq cartes sont présentes sur la table. Lors du début du second tour trois cartes sont révélées d'un coup, puis une quatrième, puis une cinquième – qui peuvent être utilisées par tous les joueurs pour former la meilleure main à cinq cartes possibles.
- Les joueurs misent chacun leur tour avant et après la distribution de chaque carte. Pour rester dans la main et voir la prochaine carte, tous les joueurs doivent avoir placé la même montant de jetons dans le pot.
- La meilleure main de poker remporte le pot.
- 2 joueurs minimum et 6 joueurs maximum.

Le premier tour commence toujours par une mise automatique de la personne (First Shooter) qui se trouve à gauche du croupier (appelé « Dealer » dans Poker Gang) qui correspondra à 10 pourcents du joueur le moins riche. Par la suite le « First Shooter » change dans le sens des aiguilles d'une montre.

- Actions disponibles pour chaque joueur

- Se coucher : abandonne le tour en perdant sa mise.
- Checker : si personne n'a misé quoi que ce soit, un joueur peut alors checker et ne pas miser de jetons.
- Miser : miser davantage de jetons.
- Relancer : suivre la mise précédente.

- Déroulement d'une partie

- Premier tour : aucune carte n'est dévoilée, la mise continue jusqu'à que tous les joueurs 'check'.
- Deuxième tour : trois cartes sont dévoilées par le croupier, le jeu continue de manière classique de droite vers la gauche.
- Troisième tour : une quatrième carte est dévoilée, le jeu continue de manière classique de la droite vers la gauche.
- Quatrième tour : la dernière carte est dévoilée, le jeu continue de manière classique de la droite vers la gauche.
- Tour tapis : si dans un tour chaque joueur non couché effectue un tapis, les cinq cartes sont alors directement dévoilées.
- Tour couché : si tous les joueurs se couchent alors la partie est terminée.

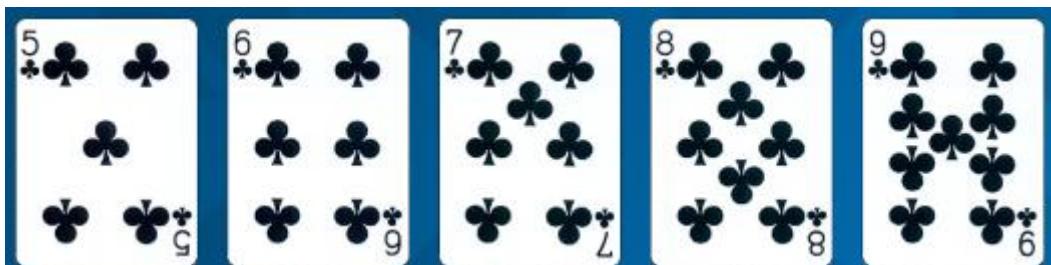
- Le classement des mains

- **Quinte Flush Royale :**



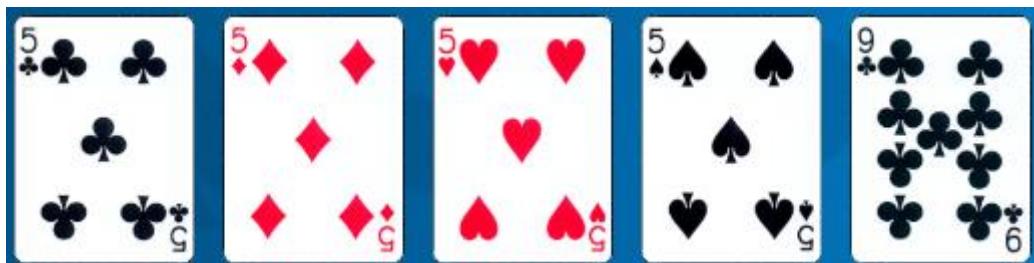
La plus forte main possible du poker. Elle est composée de : As, Roi, Dame, Valet, 10, tous de la même couleur (pique, carreau, coeur ou trèfle). Si deux joueurs ont une quinte flush royale, ils se partageront le pot.

- **Quinte Flush :**



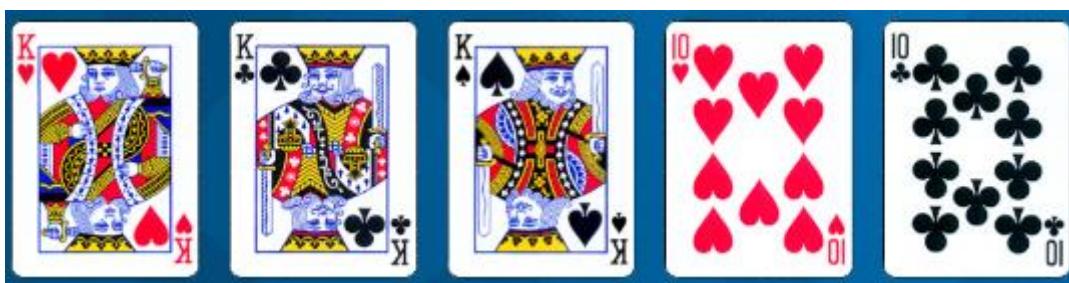
Une quinte flush est similaire à la quinte flush royale. Il s'agit d'une suite combinée à une couleur, mais sans l'As comme plus haute carte. Par exemple : 9, 8, 7, 6 et 5, tous d'une même couleur. Lorsque deux joueurs ont une quinte flush, le joueur qui a la carte la plus haute de la suite l'emporte. Lorsque les deux mains sont identiques, le pot est divisé entre les deux joueurs.

- **Carre :**



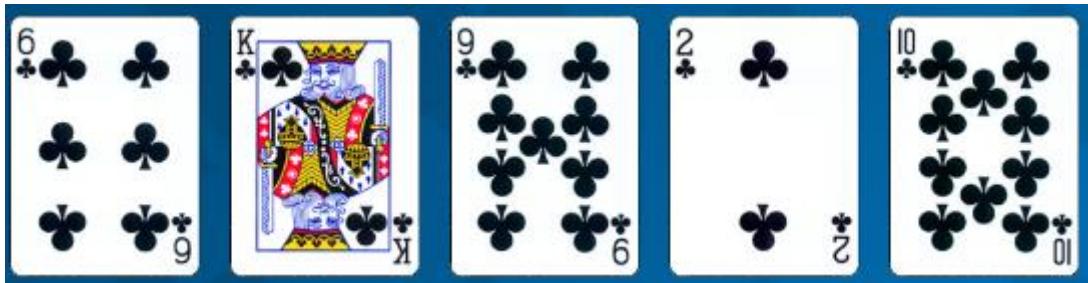
Un carré est une main qui contient quatre cartes d'un même rang comme quatre As ou quatre Valets. Lorsque deux joueurs ont un carré, celui qui a le plus haut l'emporte. Si les deux joueurs ont le même carré, le pot est remis à celui qui a la cinquième carte la plus haute, appelée aussi acolyte.

- **Full :**



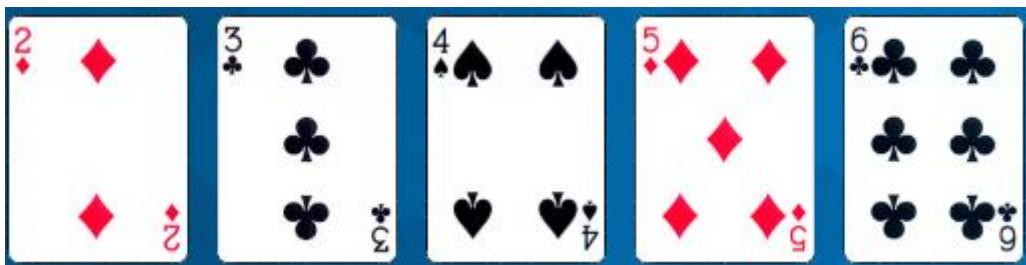
Trois cartes d'un même rang combinées à deux autres d'un même rang. Exemples de full : trois Rois et deux 10, ou trois 4 et deux As. Lorsque deux joueurs ont un full, celui qui détient les trois cartes du même rang les plus hautes gagne. Dans les deux exemples proposés auparavant, le joueur qui a les trois Rois gagne.

- **Couleur :**



Une couleur est une main qui contient cinq cartes de même couleur : cinq piques, cinq trèfles, cinq carreaux ou cinq coeurs. Lorsque deux joueurs ou plus ont une couleur, celui qui a la carte la plus haute remporte la main. Si deux mains ont la même carte la plus haute, la seconde carte la plus haute déterminera le gagnant, et ainsi de suite.

- **Quinte :**



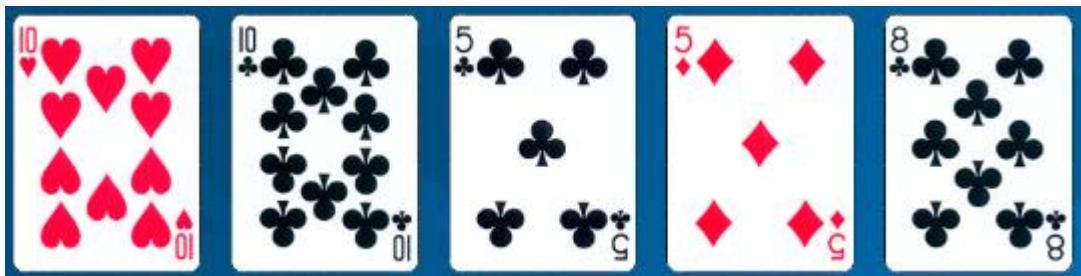
Cinq cartes qui se suivent numériquement et forment une suite, comme 6, 5, 4, 3 et 2. Lorsque deux joueurs ou plus ont une quinte, celui qui a la carte la plus haute l'emporte. Si deux joueurs ont la même carte la plus haute, ils partageront le pot. L'As peut jouer comme la plus haute carte mais aussi comme la plus faible. Par exemple : As, Roi, Dame, Valet, 10 ou As, 2, 3, 4, 5. L'As est la seule carte qui peut s'utiliser de cette façon.

- **Brelan :**



Trois cartes de même valeur, comme trois Valets ou trois 8. Lorsque deux joueurs ont un brelan, celui qui a le brelan dont la valeur est la plus élevée, l'emporte. Si deux joueurs ont le même brelan, le joueur qui a la quatrième carte la plus haute l'emporte.

- **Deux paires :**



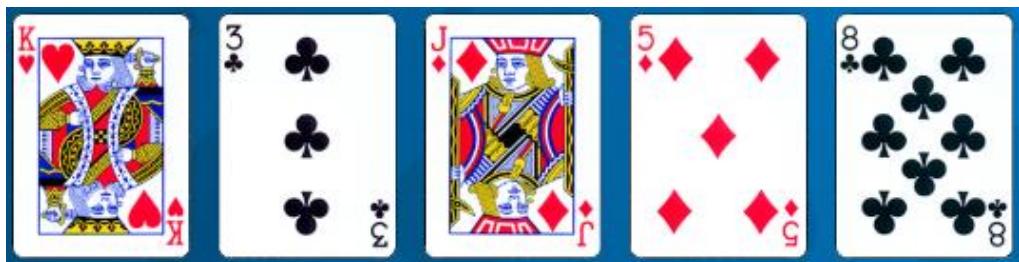
Une main qui consiste à avoir deux paires, comme deux 10 et deux 6. Si deux joueurs ont deux paires, celui avec la paire la plus haute l'emporte. Si les deux joueurs ont la même paire, celui avec la seconde paire la plus haute l'emporte. Si les deux paires sont identiques pour les deux joueurs, celui dont l'acolyte est le plus haut l'emporte.

- **Une paire :**



Main d'une seule paire. C'est-à-dire deux cartes de même valeur, comme deux As ou deux Valets, avec trois autres cartes qui ne forment aucune autre combinaison. Lorsque deux joueurs ont une paire, celui qui a la plus haute l'emporte ; s'ils ont tous les deux la même paire, celui qui détient la carte la plus haute l'emporte.

- **Hauteur :**



Une main qui ne présente aucune des combinaisons mentionnées ci-dessus est appelée une main à carte la plus haute. Lorsque deux joueurs ont tous les deux une carte, la plus haute l'emporte. Si la plus haute carte est la même pour les deux, c'est la seconde carte la plus haute qui détermine le gagnant, et ainsi de suite.