Rapport projet web mobile

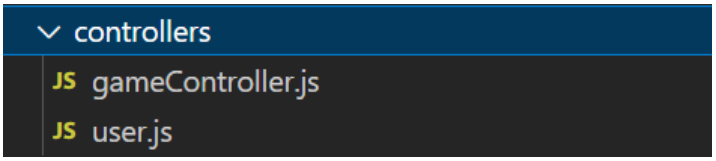
Dartois Juliette
Fauchon Louis-Bastien
Soucheleau Aurore
Lu Yuxuan

1. Introduction

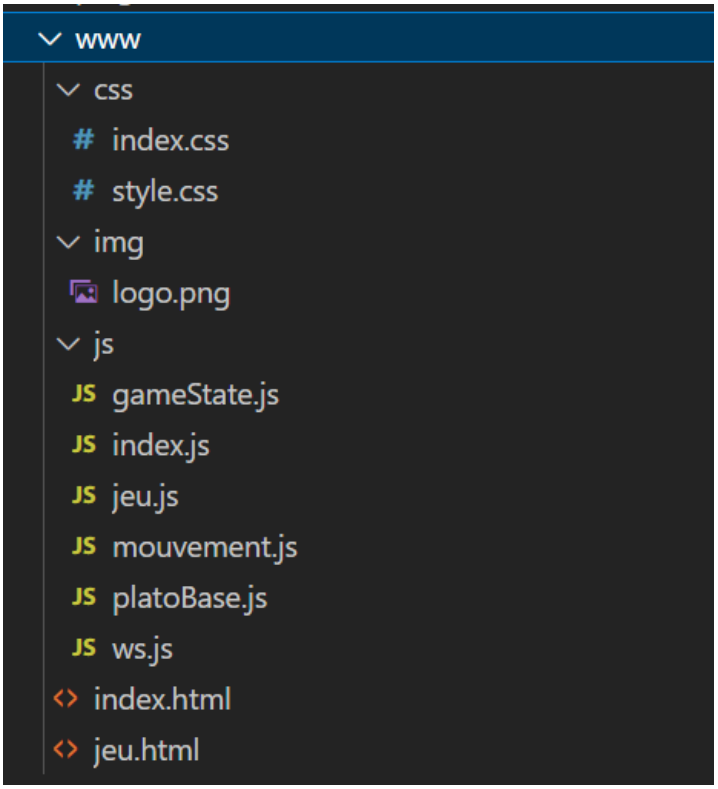
Ce projet a pour objectif de créer une application de jeu de dames, accessible depuis un smartphone ou un ordinateur. Les joueurs pourront s'identifier, rejoindre une partie en ligne et jouer contre un adversaire. Plusieurs parties peuvent avoir lieu simultanément.
Pour réaliser ce projet nous nous sommes appuyés sur des outils comme Cordova, Node.js, MongoDB, Android Studio ainsi que HTML, JavaScript et CSS.

2. Architecture du Projet

Notre code se découpe en deux sous partie la partie serveur dans le dossier controllers et la partie client dans le dossier www.



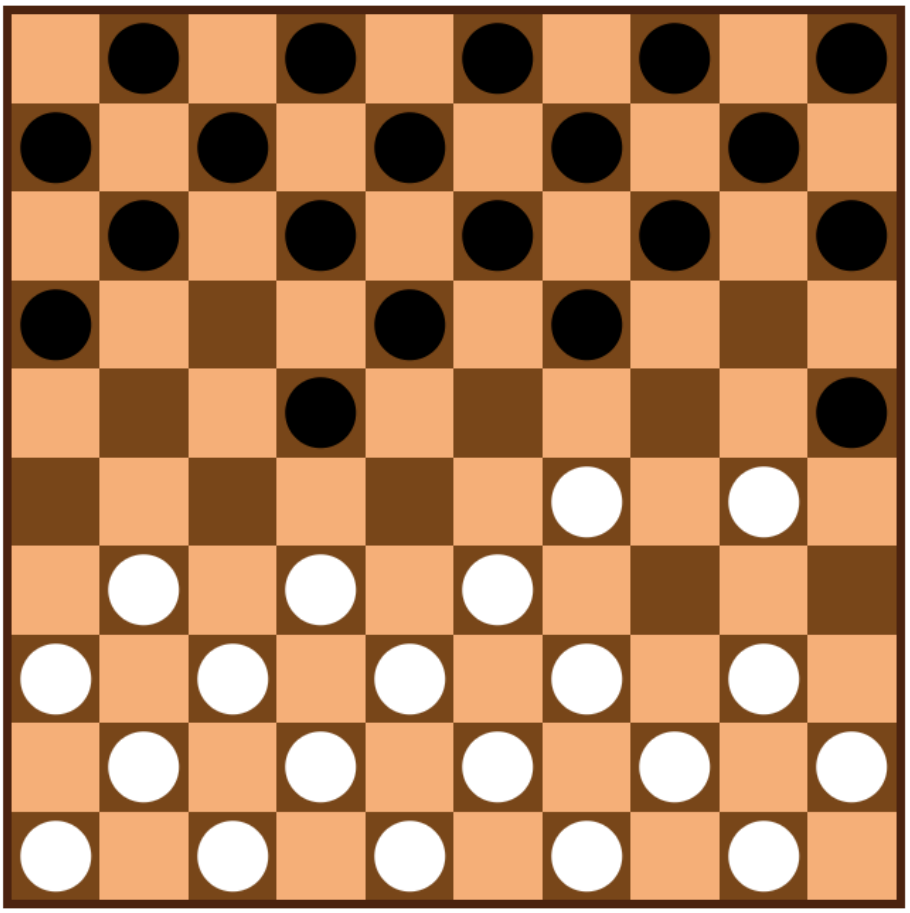
Dans la partie serveur nous avons deux classe :
La classe gameController est responsable de la gestion des événements et des actions dans le jeu. Cette classe permet d'indiquer aux joueurs quand une partie commence, de leur assigner une couleur, de vérifier le tour du joueur et de synchroniser leur mouvement.
La classe User sert à gérer les informations des utilisateurs, lors de leur inscription et de leur connexion. Lors de l'inscription, elle vérifie si un nom d'utilisateur existe déjà et sinon en crée un nouveau. Lors de la connexion, elle permet de vérifier si les informations comme le nom d'utilisateur et le mot de passe sont correcte.



Dans la partie client nous avons deux fichiers html pour la page de connexion et de jeu ainsi que leur style css et le js qui leur est associé.
gameState permet d'indiquer au joueur son tour.
index permet de récupérer les différents éléments du formulaire et d'indiquer au joueur si son inscription pose un problème.
jeu initialise la connexion initiale web-socket.
mouvement permet au joueur d'effectuer les mouvements dans le jeu, vérifie la validité des actions et crée des dames au besoin.
platoBase permet de créer le plateau de jeu initial ainsi que les cellules qui le composent.
ws gère l'envoie de message web-socket.

3. Partie Client

3.1 Interface graphique



L’interface graphique permet de visualiser les pièces et le plateau du jeu de dames, de sélectionner une pièce de départ et une case d’arrivée pour faire bouger la pièce. Les pièces ne peuvent pas effectuer de déplacement illégal, et sont obligées de manger la pièce de l’adversaire quand c’est possible. Une alerte est envoyée au joueur si celui-ci tente de faire un autre mouvement que de manger l’adversaire.

3.2 Fonctionnalité

Le côté client de l’application permet différentes fonctionnalités, notamment se connecter ou créer un compte. Des messages d'erreur indiqueront au joueur si le mot de passe et le nom d’utilisateur sont incorrects, ou s' il sont corrects que celui-ci va être redirigé vers l’interface de jeu. Lors de la création d’un nouveau compte on indiquera au joueur si son nom d’utilisateur est déjà pris, ou si le compte a bien été créé.

L’application redirige automatiquement les utilisateurs connectés vers l’interface de jeu, où, soit ils attendent un autre joueur, soit un autre joueur est déjà présent et la partie commence. Le joueur se voit attribuer une couleur en fonction de son ordre d'arrivée. L’application vérifie le tour des joueurs et s’assure que ceux-ci ne peuvent pas jouer lorsque ce n’est pas leur tour. Les déplacement d’un pion d’un joueur modifie le visuel de l’autre joueur afin que les deux joueurs voient le même plateau de jeu.

Les déplacements sont également à vérifier, uniquement en diagonale d’une case pour les pions, ou de plusieurs cases en avant et en arrière pour les dames. Ils ne peuvent pas aller sur une case ou une pièce est déjà présente. Si une pièce peut être mangée le joueur sera obligé de la manger, si plusieurs pièces sont mangeables alors le joueur aura le choix de la pièce à manger. De plus, quand un pion vient manger une pièce, celui-ci peut remanger une pièce et ce jusqu'à ce qu’il n’en ai plus la possibilité. De même pour les dames qui fonctionnent de la même manière à la différence qu' elles peuvent se déplacer de plusieurs cases. Les dames sont créées automatiquement dès lors qu’un pion arrive sur la dernière case, et indiquez par un symbole couronne sur la pièce.

La partie s'arrête quand un joueur n’a plus de pièce, un message s’affiche alors à l'écran indiquant au joueur si il a gagné ou perdu. La partie peut également être interrompue par l’un des joueur qui quitte la partie ou perd sa connexion, dans ces cas là le joueur restant reste dans la seule de jeu et attend un nouveau joueur.

4. Partie Serveur

Actuellement, notre serveur utilise WebSocket pour gérer la communication en temps réel et pour traiter en toute sécurité l'authentification et l'enregistrement des utilisateurs dans une base de données. Notre côté client interagit également avec le serveur via WebSocket pour effectuer les opérations de connexion et d'enregistrement.

4.1. Connexion webSocket

Le fichier ws.js initialise une connexion webSocket avec le serveur, c'est là bas que les messages envoyés au serveur vont être gérés. Tout au long de la connexion au serveur il y a une vérification de la disponibilité de la connexion grâce à une fonctionnalité de ping pong.

4.2. Inscription et Gestion des joueurs

C'est le fichier user.js qui gère l'inscription et la connexion des joueurs. Leurs identifiants ainsi que leurs mots de passe sont stockés dans une base de données MongoDB (db.js). Pour rejoindre une partie, les joueurs doivent s'authentifier.
Le schéma MongoDB est défini pour utiliser bcryptjs afin de stocker les informations des utilisateurs avec des mots de passe hachés.

4.3 Salles

Après leur connexion, les utilisateurs sont stockés dans une file d'attente (avec des messages d'attente côté serveur).

Si le premier utilisateur connecté clique sur son interface alors qu'il est encore dans la file d'attente un message d'erreur lui est renvoyé par le serveur le notifiant que la partie n'a pas encore commencé.

C'est lorsqu'au moins 2 utilisateurs se connectent qu'une même salle de jeu se créée pour les 2 premiers connectés si leurs noms sont bien valides (sinon message d'erreur). Les utilisateurs sont maintenant joueurs (player1 et player2) et sont supprimés de la file d'attente. Chaque joueur se voit alors attribuer une couleur : blanc pour le premier connecté et noir pour le second.
Chaque salle est identifiée par un ID unique et stocke l'information des deux joueurs et l'état actuel du jeu.

4.4 Déplacements

Lorsqu'un joueur effectue un mouvement, le serveur vérifie si c'est bien à son tour (avec un message d'erreur renvoyé au côté client si ce n'est pas le cas).

Si le joueur actif effectue un mouvement validé par le côté client, le mouvement est envoyé au serveur via un webSocket. Le serveur renvoie le déplacement aux deux joueurs.

Permettant ainsi la synchronisation des plateaux des deux joueurs de la même salle. A chaque mouvement effectué, il y a des messages côté serveur.

Pareil pour la promotion d'un pion ou la capture d'un pion, dans les deux cas le serveur renvoie l'information aux deux joueurs grâce à des webSockets pour mettre l'état du plateau à jour.

A chaque fin de tour (défini côté client), le serveur change le joueur actif et informe les deux joueurs du changement.

4.5 Fin de partie

Arrivé à la fin d'une partie (lorsqu'un des joueurs n'a plus de pions dans notre cas), le serveur récupère l'information du gagnant et informe les joueurs de la salle du gagant. Ensuite le serveur supprime la salle.

4.6 Déconnexion

Si lors de la partie un des deux joueurs se déconnecte alors le joueur est supprimé de la salle.
Le deuxième joueur n'est pas averti du départ de son adversaire. La salle est supprimée seulement si le deuxième joueur se déconnecte aussi.
Si le joueur se déconnecte alors qu'il était dans la file d'attente alors il est supprimé de la file d'attente.

5. Répartition des tâches

Développement Frontend : Juliette et Louis-Bastien
Création d'interfaces utilisateurs : index.html et jeu.html
Implémentation des interfaces utilisateurs et les interactions de jeu : index.js, index.css, style.css et jeu.js

Développement Backend : Yuxuan et Aurore
Gestion des parties entre deux joueurs : gameController.js
Gestion de la logique d'authentification utilisateurs : user.js et User.js

Logique partagée : Yuxuan et Louis-Bastien
Création de la logique de jeu : mouvement.js gameState.js
Gestion mise en place des websockets et pour la connexion des joueurs : ws.js et ServerWS.js

Le projet a été réalisé grâce à chaque membre de notre groupe, chacun ayant apporté des idées dans leurs parties.

Toutefois, nous tenons à souligner que Louis-Bastien et Yuxuan ont su intervenir sur les tâches plus complexes pour la réussite du projet.

6. Pour aller plus loin

Bien que notre projet soit fonctionnel, voici certaines fonctionnalités que nous aurions pu implémenter afin de rendre l'expérience utilisateur plus agréable et que l'ensemble des règles du jeu de dames soit implémenté.

- une connexion plus élaborée avec un système de récupération de mots de passe quand il a été oublié.
- une intelligence artificiel qui permet aux joueur de jouer contre l'ordinateur
- la possibilité pour les pions de manger en arrière
- etablir une fin de partie quand les joueur n'ont plus assez de pièce ou font 3 fois de suite les même mouvement
- un système de classement des meilleur joueur ainsi qu'un affichage de leur score
- Lorsqu'une partie est interrompue par un joueur qui quitte la partie, cela est considéré comme une défaite pour celui-ci et une victoire pour l'autre joueur.
- lorsqu'une partie est interrompue car le joueur a perdu sa connexion il a 30s pour se reconnecter avant que cela soit considéré comme un forfait.
- lorsqu'un joueur ne peut plus jouer alors que c'est son tour cela entraîne la défaite.
- un système de minuteur qui oblige le joueur à jouer dans un temps imparti.
- permettre au joueur de proposer l'égalité.
- un bouton permettant au joueur de rejouer a la fin d'une partie, soit contre un nouvel adversaire, soit contre le même celui-ci est d'accord.