Université Grenoble Alpes

Jeu de Dames

Développement Web Mobile

Groupe 7 - M2 MIASHS
Rohini Ackiah, Thomas Foret, Mohamed Taha Maata, Gwendoline Delestre

Sommaire

1. Objectifs

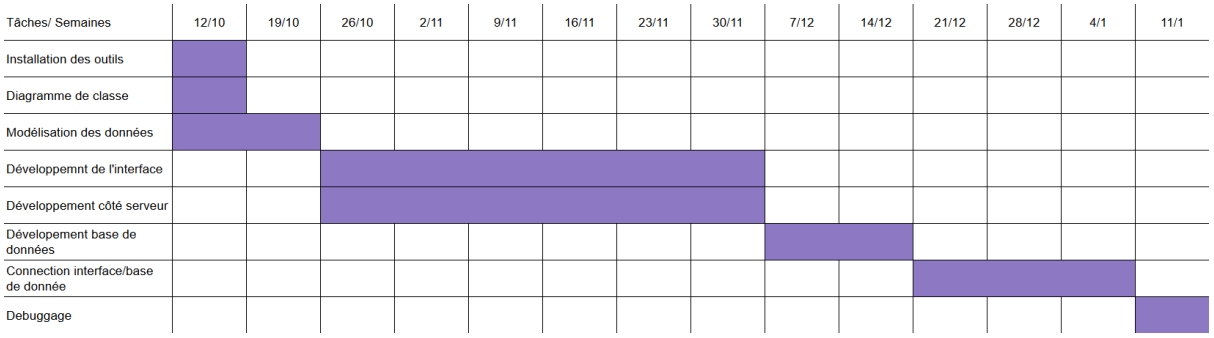
Le but de ce projet est de réaliser une application Web mobile permettant de jouer au jeu de Dames en ligne que ce soit sur une application téléphone ou sur un navigateur sur ordinateur. L'identification avec un email et un mot de passe est obligatoire afin de jouer ce qui permet d'enregistrer les scores des joueurs dans une base de données une fois que la partie est terminée et que le vainqueur est validé.

2. Technologies utilisées

Pour réaliser ce projet, nous utilisons Cordova pour le développement côté client, node.js pour le côté serveur et MongoDB pour la base de données. Afin d'émuler un téléphone sur ordinateur, nous avons aussi la possibilité d'utiliser Android Studio.

3. Conception

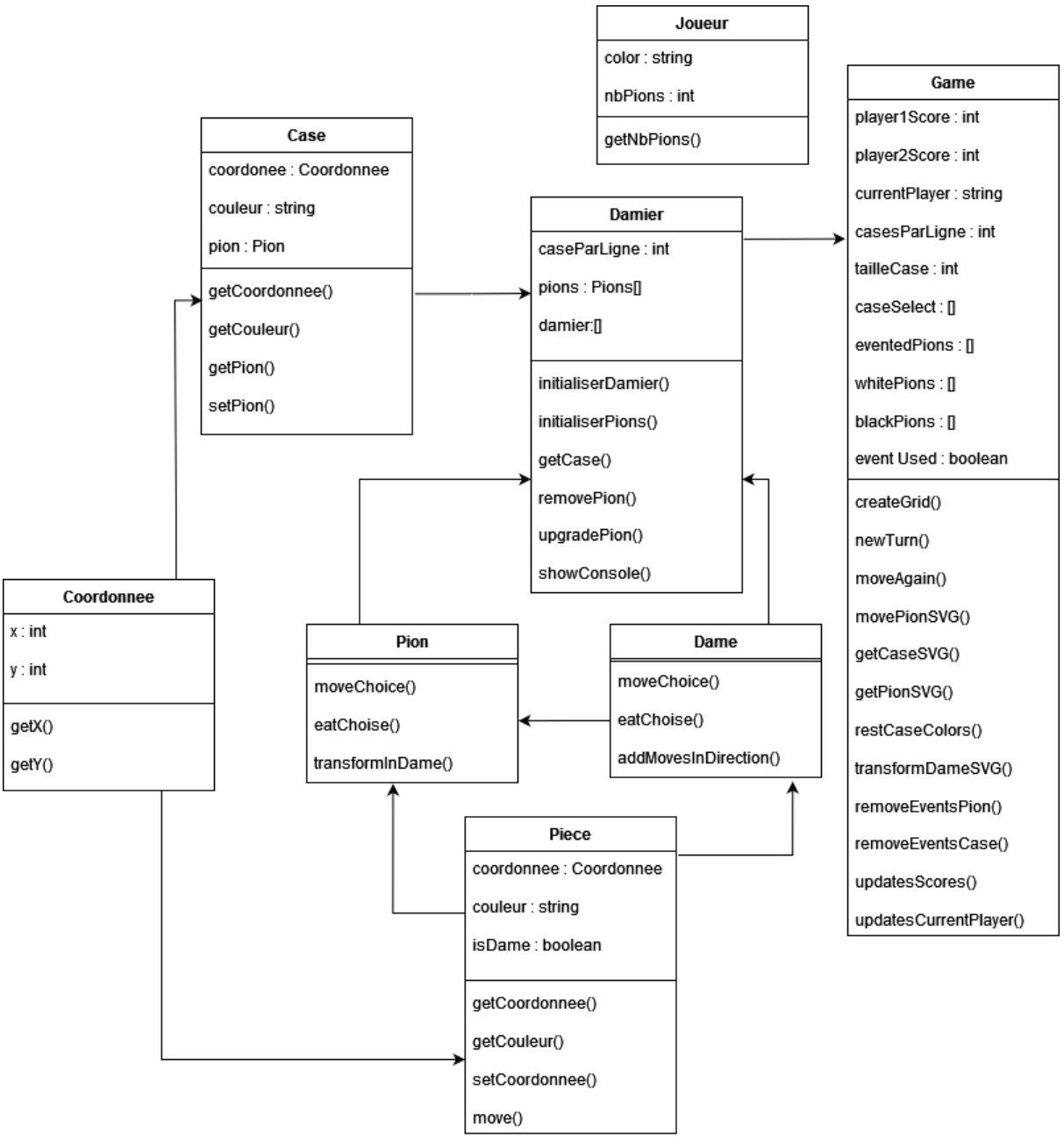
a. Planning prévisionnel



b. La bases de données

Nous utilisons une base de données MongoDB.
Notre base de données comprend une table avec deux collections : une première collection pour les utilisateurs (users : login, mdp, user_id), et une deuxième pour les scores de jeu.

c. Diagramme de classes du jeu en mode solo



d. Règles du jeu

Le but du jeu est de capturer toutes les pièces de l'adversaire ou de bloquer ses pièces de manière à ce qu'il ne puisse plus effectuer de mouvements.

Le jeu se joue sur un plateau de 64 cases, alternant les couleurs, avec chaque joueur plaçant ses pièces sur les cases noires.

Chaque joueur commence avec 12 pièces. Les pièces se déplacent en diagonale vers l'avant.

Le joueur avec les pions blancs commence.

Si une case adjacente est occupée par une pièce adverse et la case suivante est libre, le joueur peut capturer la pièce adverse en effectuant un saut par-dessus elle. La pièce capturée est retirée du jeu.

Si un pion atteint la dernière rangée de l'adversaire, il est couronné et devient une dame. Les dames peuvent se déplacer en diagonale dans n'importe quelle direction sur plusieurs cases.

Si après une capture le joueur a la possibilité de capturer une autre pièce adverse, il peut le faire. Plusieurs sauts peuvent être effectués en un seul tour.

Le jeu se termine lorsque l'un des joueurs capture toutes les pièces de l'adversaire.

4. Développement

a. Coté Client

Le côté client à été codé majoritairement en HTML, CSS et JS.

index.html comprend toutes les différentes “vues” allant de la connexion à la fin d’un jeu.
index.css regroupe les différentes classes utilisées pour le style du jeu.
Nous avons implémenté plusieurs classes javascript afin de configurer le jeu.

- Case.js

Elle permet de définir une case par sa couleur et la présence d’un pion ou non.

- Coordonnee.js

Cette classe permet d'obtenir les coordonnées d’une case ou d’une pièce.

- Dame.js

Cette classe est une sous classe de Piece, elle permet de définir les dames par leur mouvements, le choix de mouvements qui leurs sont possibles à un instant du jeu.

- Damier.js

Le damier permet de créer toutes les cases du jeu ainsi que les pions qui s’y trouvent.

- Game.js

Cette classe permet de créer la grille visuelle pour une partie solo (jouer contre soi-même) ainsi que tout changement de celle-ci au fur et à mesure du jeu. De plus, si le joueur à la possibilité de rejouer, elle l’indique puisqu'est donne aussi le joueur actuel.

- GameOrdi.js

De la même façon que la classe Game.js, cette classe permet de jouer contre un ordinateur qui fait des mouvements aléatoires parmi ceux qui lui sont disponibles.

- GameOnline.js

Cette classe permet de jouer en ligne contre un autre adversaire connecté.

- Joueur.js

La classe Joueur.js permet de créer une instance d'un joueur et de lui attribuer une couleur ainsi qu'un nombre de pions.

- Piece.js

La classe piece est une super classe de Dame et Pion, elle permet de définir et de récupérer les coordonnées et de définir la couleur d'une pièce de jeu.

- Pion.js

Pion est une sous classe de Piece et définit les mouvements des pions, le choix possible de déplacement et le choix de possible de manger un pion adverse. Un pion peut aussi devenir une dame.

- index.js et main.js

Ces deux fichiers permettent de définir les événements liés aux différents boutons. Plus particulièrement, c'est index.js qui permet de lancer les différents modes de jeu.

Nous avons donc développé 3 modes de jeu différents, un mode solo qui permet de jouer contre soi-même, un mode Ordi où l'on joue contre un ordinateur qui choisit aléatoirement un mouvement parmi une liste de mouvements possibles et enfin un mode de jeu en ligne qui permet de jouer avec un autre joueur.

b. Côté Serveur et Base de données

- server.js

Nous avons créé une bdd sur MongoDB sous le nom de dbJeu afin de stocker les utilisateurs connectés. On stocke leur username et leur password qu'ils insèrent dans la page de login et à chaque login on vérifie s'ils sont présents dans la bdd afin d'éviter les doublons.

Pour le serveur, la connexion se fait avec des websockets. Quand on reçoit des informations du client, afin de pouvoir différencier les messages reçus, on a prefixé certains messages avec des string. E.g. le message d'authentification est prefixé de "Login_" et aussi "pass_" dans une string comme "Login_mars-rhoverpass_05031998" afin de pouvoir récupérer les identifiants pour toutes les instances d'utilisateurs. On vérifie s'ils sont dans la bdd et on met à jour le document concerné avec le statut "connected". Sinon une insertion est faite.

Pour les jeux 1vs1, une salle d'attente a été créée afin de connecter les 2 utilisateurs concernés dans une salle commune avant de commencer le jeu. Si aucune salle n'est trouvée, on en crée une, sinon on cherche une place de libre.
Si un utilisateur se déconnecte, un message est affiché.

5. Installation

Installation de packages nécessaire avec npm install: websocket, MongoDB
Pour lancer le projet vous devez lancez 3 terminal

1. La bdd avec **mongod --dbpath {lien vers la DBB}**
2. Le serveur avec **node server.js** en étant dans le dossier serveur
3. Le projet avec **cordova build browser** et ensuite **cordova run browser**

6. Répartition du travail

Thomas s'est chargé de coder les règles du jeu ainsi que les mouvements des pions. Après avoir codé le jeu en mode solo, il a adapté le code pour jouer contre l'ordi.

Rohini et Mohamed Taha se sont chargés du backend et de la base de données.

De plus, Rohini et Thomas ont adapté le code du mode de jeu contre ordinateur pour jouer en ligne.

Gwendoline s'est majoritairement chargé du rendu visuel du jeu et des menus.

7. Améliorations pour la soutenance

- Il persiste quelques bugs lorsque l'on joue contre l'ordinateur.
- Affichage des meilleurs scores dans le menu "Meilleurs Scores"
- Amélioration de l'authentification
- Amélioration du 1vs1

8. Améliorations générales

- Amélioration du global code