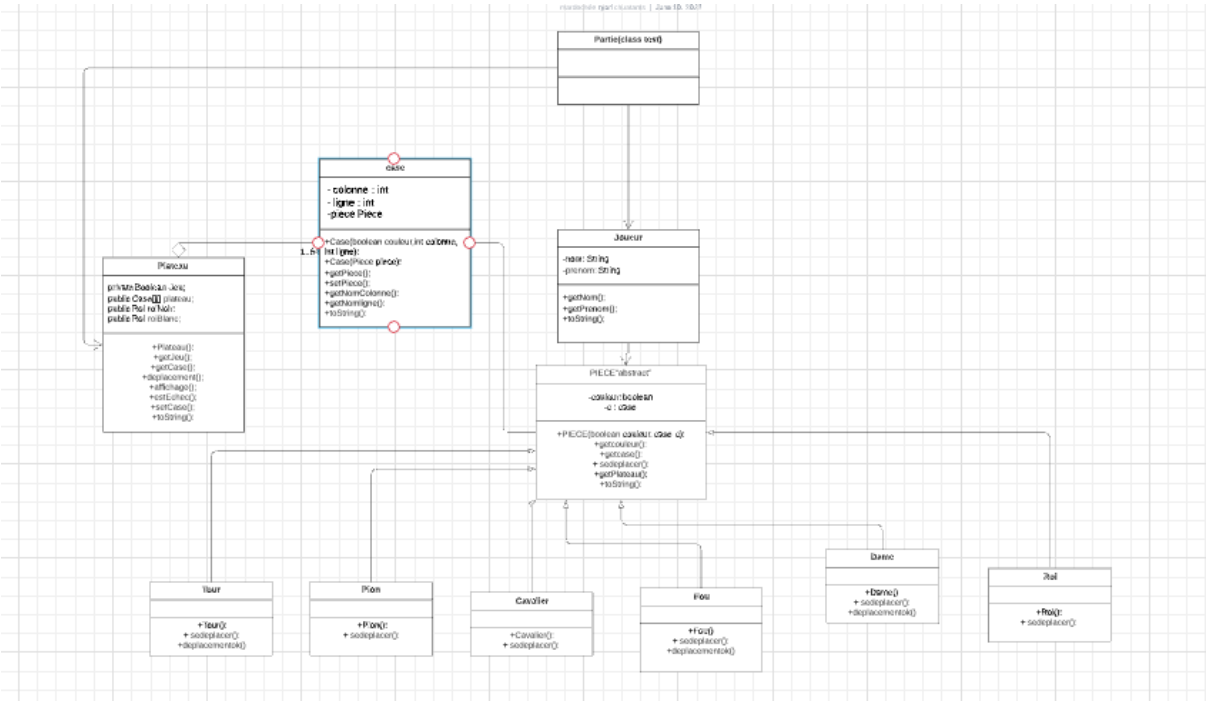
Tamimoul Chuaibe
Castin Mardochée
Messai Ryad

Aigle
Projet Tutoré

RAPPORT FINAL:

Tout d'abord nous avons étudié le sujet en groupe. Puis nous avons listé toutes les tâches nécessaires à la réalisation de ce jeu d'échecs. Ensuite, avec notre groupe nous avons rempli le tableau d'avancement, qui nous permet de savoir où nous en sommes. Ce tableau nous permet de réaliser des tâches dont un diagramme de classe et un diagramme de cas d'utilisation.

Diagramme de classe:



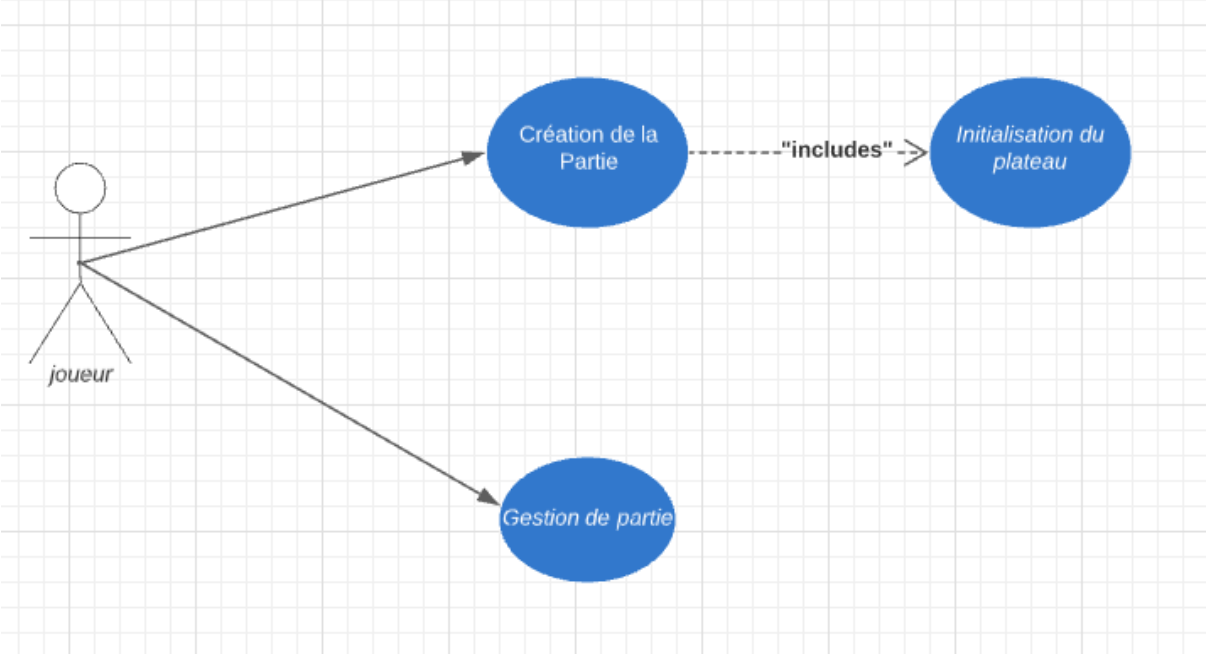
Pour le diagramme de classe nous avons énuméré toutes les classes qu'on estimait essentiel pour ensuite les coder.

Diagrammes de cas d'utilisation:

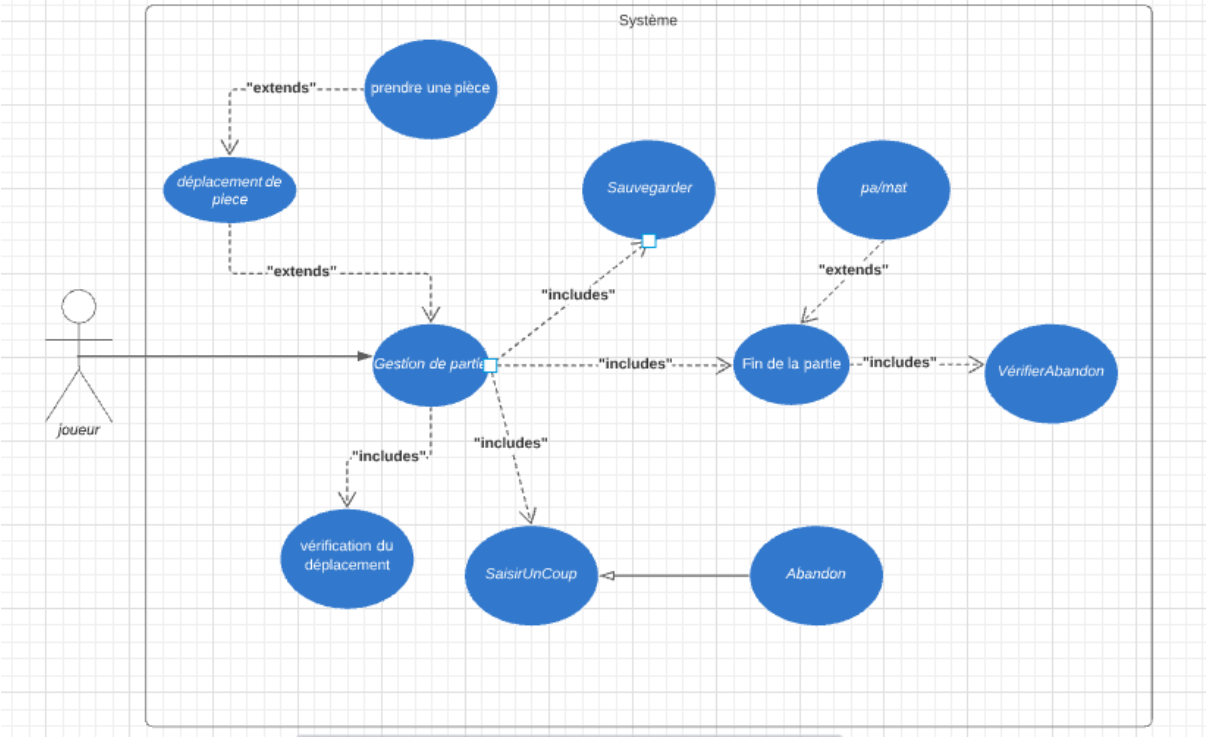
Nous avons fait un diagramme de cas d'utilisation pour nous permettre d'avoir une représentation fonctionnelle du programme.

Voici les 2 diagramme qu'on a réalisé:

Premier Diagramme



Deuxième Diagramme



Description:

2 joueurs vont créer une partie pour s'affronter. Les joueurs auront la possibilité de déplacer une pièce et chaque avancement dans la partie entraînera une sauvegarde obligatoire. De plus, il y a différentes manières de finir une partie, soit par abandon d'un joueur soit par match nul (PAT) ou soit par la capture du roi du joueur adverse (MAT).

Le codage des classes:

Après avoir fait les diagrammes de cas et de classe, nous avons par la suite commencé le codage des classes qui vont servir au fonctionnement du programme.
Nous nous sommes répartis les différentes classes à coder sauf pour les plus complexes qu'on a décidé de réaliser en groupe.

Description des classes:

Classe Piece:
Cette classe est une classe abstraite dans laquelle il y a deux méthodes, une qui vérifie si une pièce quelconque peut se déplacer d'une case à une autre et une autre qui permet de vérifier s' il n'y a pas pièces sur le chemin notamment pour la Tour, le Fou et la Dame . Cette méthode est utilisée par toutes les classes qui héritent de Pièce c'est-à-dire le Roi, la Dame, le Fou, le Cavalier, la Tour et le pion.

méthode abstraite:

```
public abstract boolean sedeplacer(Case depart ,Case arrivee);
public abstract boolean deplacementok(Case depart ,Case arrivee);
```

Classe Case:
Cette classe est liée à la classe Pièce et elle a une méthode qui permet à une pièce de savoir dans quelle case elle se situe sur le plateau.

Getters:

```
public int getLigne(){
    return this.ligne;
}

public int getColonne(){
    return this.colonne;
}
```

Classe Plateau:
Cette classe permet entre autres la mise en place des pièces avec la méthode setPièce dans les cases. On l'utilise notamment lors de l'initialisation du plateau.

```
public void setPiece(Piece p)
{
    this.piece = p;
}
```

Classe Partie:
Cette classe permet de gérer le déroulement du jeu, c'est-à-dire elle effectue des affichages et permet de vérifier c'est à quel joueur de jouer.
Au blanc de jouer :

```
boolean Tours = true;
System.out.println("C'est au tour des blanc a vous monsieur "+ joueur1.toString());
while (gameover == false) {
```

Au noir de jouer :

```
System.out.println("C'est au tour des noir a vous monsieur "+ joueur2.toString());
toursdejeux = false;
Tours=false;
```

Comment lancer notre programme ?

Après avoir compilé toutes nos classes, on exécute la classe "Partie" pour pouvoir commencer à jouer.

Déroulement du programme:
Tout d'abord le programme vous proposera de jouer, entrez "o" ou "O" pour pouvoir commencer le jeu. Ensuite, il demandera les noms et prénoms des 2 joueurs pour les affecter à une couleur(blanc ou noir). Puis, c'est au tour des blancs de commencer et il demandera ensuite de rentrer les coordonnées (colonne de départ/ligne de départ et colonne de d'arrivée et ligne d'arrivée). Ainsi il affichera la position de la case de départ et celle de la case d'arrivée. Par la suite, c'est au tour des noirs et ça sera le même principe que pour les blancs.

Difficultés rencontrées :

Nous n'avons pas pu finir notre projet au niveau de la programmation car nous avons eu du mal à adapter les classes entre elles. De plus, même si nous avons codé certaines méthodes nous n'avons pas réussi à les faire fonctionner dans le programme. Enfin, nous avons privilégié beaucoup de méthodes qui nous semblaient essentielles au programme mais nous n'avons pas fait notamment le chargement de partie et la sauvegarde.