

|  |
| --- |
| **SPEAPING**  APICHESS  https://bouyakasha.github.io/ |

|  |
| --- |
| **Rapport de Soutenance**  **2** |

|  |  |
| --- | --- |
| **N° de version** : 1.2 | **Auteur** : Meliani Keyvan |

**SOMMAIRE**

[1. PRESENTATION GENERALE 2](#_Toc480993340)

[1.1 Introduction 2](#_Toc480993341)

[1.2 Joies et peines 2](#_Toc480993342)

[1.3 Répartition du travail 2](#_Toc480993343)

[2. Problèmes rencontrés 3](#_Toc480993344)

[2.1 Déplacement 3](#_Toc480993345)

[2.2 Structure 3](#_Toc480993346)

[2.3 interface graphique 3](#_Toc480993347)

[2.4 inteligence artificielle 4](#_Toc480993348)

[3. Annexes 5](#_Toc480993349)

[3.1 Capture d’ecran 5](#_Toc480993350)

[3.2 Structures 7](#_Toc480993351)

[3.3 Evaluation 8](#_Toc480993352)

# PRESENTATION GENERALE

## Introduction

A la fin de la première soutenance nous avons décidé de nous écarter de la gestion de projet initialement prévue afin d’avoir le meilleur produit fini possible lors de la dernière soutenance.

Nous avions convenu lors de la réalisation du cahier des charge de se pencher sur l’intelligence artificielle majoritairement en fin de projet.

Il fut décidé de s’y pencher le plus tôt possible afin de pouvoir avoir une marge de manœuvre en cas de mauvaise modélisation.

De plus alors qu’un affichage console avait été prévu, il a été choisi d’implémenter une interface graphique afin d’améliorer le confort visuel de la partie.

## Joies et peines

Quasiment l’entièreté du code a dû être revue.

Nous avons décidé de nous focaliser sur un changement de la majorité des structures, des fonctions utilisées et de modéliser de nouveaux types de données afin d’optimiser un maximum le code et d’avoir à y retourner dessus le moins possible.

## Répartition du travail

La répartition des tâches a été la suivante même si le travail a majoritairement été fait en groupe et un suivi régulier collectif du projet a été fait.

*Keyvan Meliani*

Modélisation, Gestion, Refacto, IA, IATree

*Julien Hautier*

Factorisation du code, modélisation des fonctions de jeux, Développement

*Jean Charles Gravaillac*

Interface graphique

# Problèmes rencontrés

Nous avons rencontré plusieurs problèmes au cours de cette période. La gestion du temps et l’organisation des tâches a nécessitée d’être refaite.

La complexité du code rendait l’utilisation de debugger tel que GDB fastidieuse. Il fallait remonter plusieurs fonctions afin de repérer les erreurs que l’on rencontrait.

Une segmentation et factorisation correcte en plusieurs fichiers a été faite.

## Déplacement

Nous avons décidé de créer des tableaux de chaque coup possible pour chaque pièce.

Un système permettant d’empêcher les Segfault correspondant aux mouvements à erreurs.

Cela a permis d’optimiser le code en testant au préalable le mouvement que la pièce devait faire avant de l’effectuer. Cela a permis une réduction de 50 % du code du projet et une meilleur lisibilité et élasticité de celui-ci.

## Structure

L’accès aux propriétés de chaque case était difficile. Il fallait à chaque fois savoir quelle pièce était sur une case donnée afin de savoir quelle réaction adopter. Pour ce faire, il fallait récupérer la bonne structure qui correspondait à une seule pièce donnée.

Pour résoudre ce problème, nous avons décidé de créer une structure unique pour toutes les pièces. Cela nous a permis pour chaque case de les faire pointer sur un seul type de structure et plus sur un type void qui pouvait représenter n’importe quelle pièce.

## interface graphique

Le développement de l’interface graphique a posé quelques problèmes. Il fut compliqué de relier le code du projet à son interface car ce dernier a subi de nombreux changements. La conversion des coordonnées graphiques en coordonnées utilisables (envoyer la position de départ et d’arrivé correspondant à une chaine de caractères) par l’utilisation des fonctions de déplacements.

Ce problème a été résolu par la segmentation de la surface de jeu en sous blocs représentant des coordonnées et donc de savoir où l’action a été faite et où le joueur ou l’intelligence artificielle veut déplacer sa pièce.

L’implémentation du menu n’a pas pu être terminée et doit actuellement se faire par console.

## inteligence artificielle

La gestion de la mémoire lors de la génération de l’arbre général représentant toutes les possibilités du jeu était inefficace. Le programme ne cessait de s’arrêter à partir du tour 5.

Nous avons dû repenser l’entièreté de la gestion mémoire concernant l’échiquier et l’utilisation de ce dernier par l’intelligence artificielle.

Pour pallier à ce problème, nous avons décidé que l’intelligence artificielle ne conserverait que les informations utiles à cette dernière : la couleur de la pièce représentant son appartenance, sa valeur utile à la prise de décision du meilleur coup et le type de la pièce, catégorisée par un code décimal allant de 1 à 9.

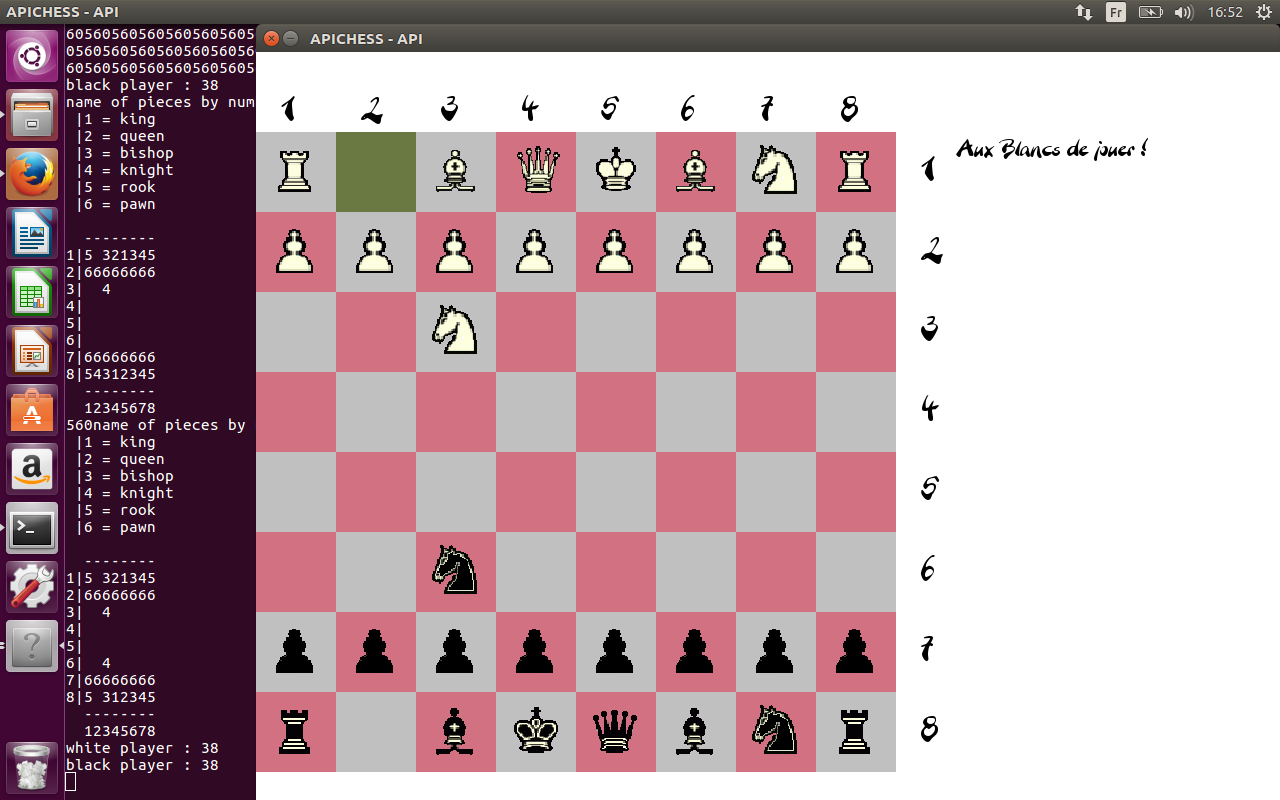
Pour ce faire, l’intelligence artificielle utilisera une structure représentant l’échiquier mais en ne conservant que les informations utiles, allégeant de par ce fait la quantité de mémoire utilisée.

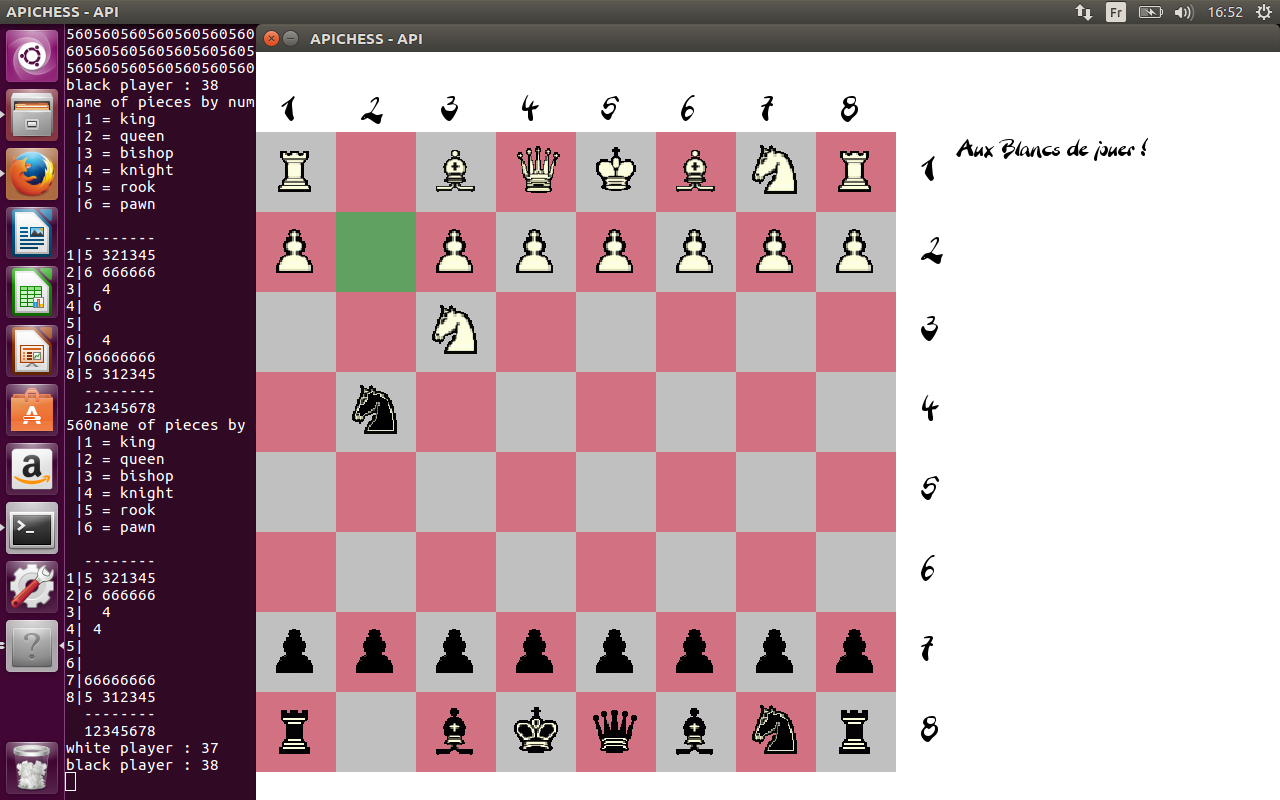
Pour cette soutenance, l’intelligence artificielle ne choisira que le meilleur déplacement sur toutes les possibilités offertes au prochain tour.

# Annexes

## Capture d’ecran







## Structures

**struct** piece {

**int** type;

**int** id;

**int** color;

**int** square;

**float** value;

**int** isDead;

**int** firstMove;

**struct** piece **\***next;

};

**struct** square {

**int** id;

**int** color;

**struct** piece **\***isEmpty;

};

**struct** chessBoard {

**int** turn;

**int** end;

**struct** square **\*\***squareMat;

**struct** piece **\***nextDeadPiece;

**struct** piece **\***nextAlivePiece;

**int** **\*\***pawnMoves;

**int** **\*\***knightMoves;

**int** **\*\***bishopMoves;

**int** **\*\***rookMoves;

**int** **\*\***kingMoves;

};

**struct** AISquare {

**int** type, color, value;

};

**struct** AITree {

**struct** AISquare **\*\***AIMat;

**int** vBlack, vWhite, height;

**struct** AITree **\***nodes;

};

Keyvan Meliani | Julien Hautier | Jean-Charles Gravaillac

## Evaluation

L’évaluation des pièces qui a été choisie est celle définie par Hans Berliner ex champion de jeu d’échec ayant participé à la participation de l’ordinateur HiTech

* pion : 1.0
* [cavalier](https://fr.wikipedia.org/wiki/Cavalier_(jeu_d%27%C3%A9checs)) : 3.2
* [fou](https://fr.wikipedia.org/wiki/Fou_(jeu_d%27%C3%A9checs)) : 3.33
* [tour](https://fr.wikipedia.org/wiki/Tour_(jeu_d%27%C3%A9checs)) : 5.1
* [dame](https://fr.wikipedia.org/wiki/Dame_(jeu_d%27%C3%A9checs)) : 8.8