Rapport de projet : Développement Python

5 JANVIER

Michaud loann
El Moussaoui Ramzi
Giana Matteo

Table des matières

INTRODUCTION

Ce rapport présente notre projet de développement d'un jeu de Puissance 4 en Python, jouable sur un serveur local. Notre objectif était de créer une version numérique du jeu classique, avec une touche moderne : les victoires accordent des avantages aux gagnants, augmentant ainsi l'aspect compétitif. Nous avons mis l'accent sur une interface utilisateur attrayante et une expérience de jeu fluide, en utilisant les capacités de Python pour la programmation réseau et la création d'une expérience de jeu interactive. Ce projet vise à offrir une nouvelle dimension au Puissance 4, en le rendant plus dynamique et accessible dans l'environnement numérique actuel.

ANALYSE DU CODE

Le code de notre jeu de Puissance 4 en Python est structuré pour optimiser la jouabilité, la maintenance et l'extensibilité. Il se divise en plusieurs composants clés, chacun remplissant un rôle spécifique dans le fonctionnement du jeu.

Gestion de la grille de jeu : Le cœur du jeu repose sur une grille 2D, représentée par une liste de listes en Python. Cette structure permet une manipulation aisée des jetons dans la grille et simplifie la vérification des conditions de victoire.

Logique de jeu : Des fonctions dédiées gèrent la logique du jeu, telles que les mouvements des joueurs, la vérification des conditions de victoire et le suivi des tours. Ces fonctions permettent de rendre le code plus lisible et facilitent les modifications ou ajouts de nouvelles règles.

Interface utilisateur : L'interface utilisateur est conçue pour être simple et intuitive. En utilisant les bibliothèques graphiques de Python, nous avons créé une interface visuellement agréable, permettant aux joueurs de voir la grille de jeu et de recevoir des feedbacks clairs sur les actions en cours.

Réseau et serveur local : Le jeu est conçu pour fonctionner sur un serveur local, permettant aux joueurs de se connecter et de jouer en temps réel. Cette partie utilise les fonctionnalités réseau de Python pour établir la connexion et gérer les échanges de données entre les joueurs.

Gestion des avantages : Pour augmenter l'aspect compétitif, le jeu intègre un système d'avantages pour le gagnant. Ce système est programmé pour suivre les victoires et attribuer des bonus ou avantages, rendant chaque partie unique et dynamique.

Extensibilité : Le code est écrit en gardant à l'esprit l'extensibilité. Cela signifie que de nouvelles fonctionnalités, telles que de différents modes de jeu ou des améliorations de l'interface, peuvent être ajoutées sans restructuration majeure.

En conclusion, le code du jeu Puissance 4 a été développé avec une attention particulière à la jouabilité, à l'expérience utilisateur et à la robustesse de la conception. L'utilisation de Python a facilité la mise en œuvre de fonctionnalités complexes tout en maintenant le code clair et maintenable.

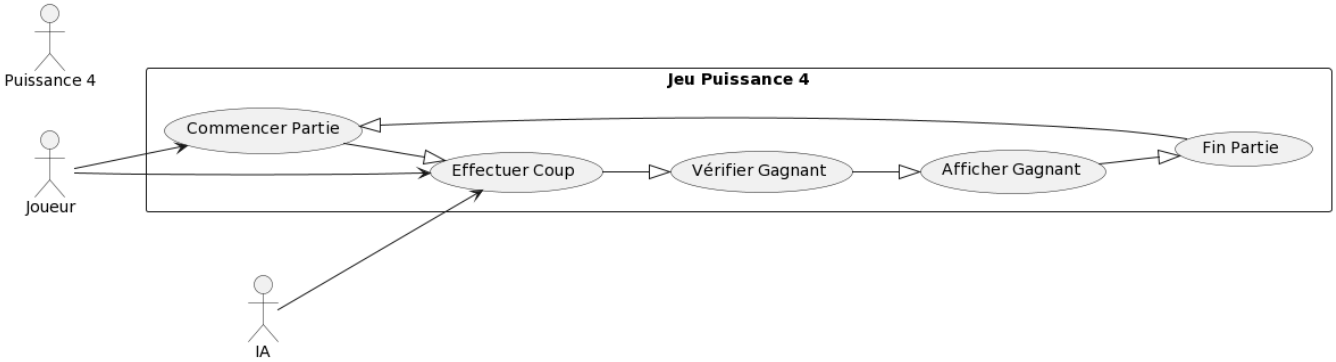
METHODES & FONCTIONS

- **Fonction `create_board`:**
  - **Entrée :** Aucune
  - **Sortie :** Liste de listes représentant la grille de jeu (type `list`)
  - **Description :** Cette fonction initialise la grille de jeu en créant une matrice 6x7, où chaque cellule est initialement vide.
- **Fonction `drop_piece`:**
  - **Entrée :** `board` (la grille de jeu), `row` (numéro de la rangée), `col` (numéro de la colonne), `piece` (le jeton du joueur)
  - **Sortie :** Aucune (modification en place de la grille)
  - **Description :** Place un jeton du joueur dans la grille à l'emplacement spécifié.
- **Fonction `is_valid_location`:**
  - **Entrée :** `board` (la grille de jeu), `col` (numéro de la colonne)
  - **Sortie :** Booléen (type `bool`)
  - **Description :** Vérifie si un coup est jouable dans la colonne donnée (i.e., si la colonne n'est pas déjà pleine).
- **Fonction `winning_move`:**
  - **Entrée :** `board` (la grille de jeu), `piece` (le jeton du joueur)
  - **Sortie :** Booléen (type `bool`)
  - **Description :** Vérifie si le dernier coup joué conduit à une victoire (alignement de quatre jetons).
- **Fonction `draw_board`:**
  - **Entrée :** `board` (la grille de jeu)
  - **Sortie :** Aucune (affichage graphique du plateau)
  - **Description :** Dessine l'état actuel de la grille de jeu, en représentant les jetons des joueurs.
- **Fonction `play_game`:**
  - **Entrée :** Aucune
  - **Sortie :** Aucune
  - **Description :** Gère le déroulement d'une partie, incluant la prise de tours entre les joueurs et la vérification de la condition de victoire.

CONCEPTION DETAILLEE

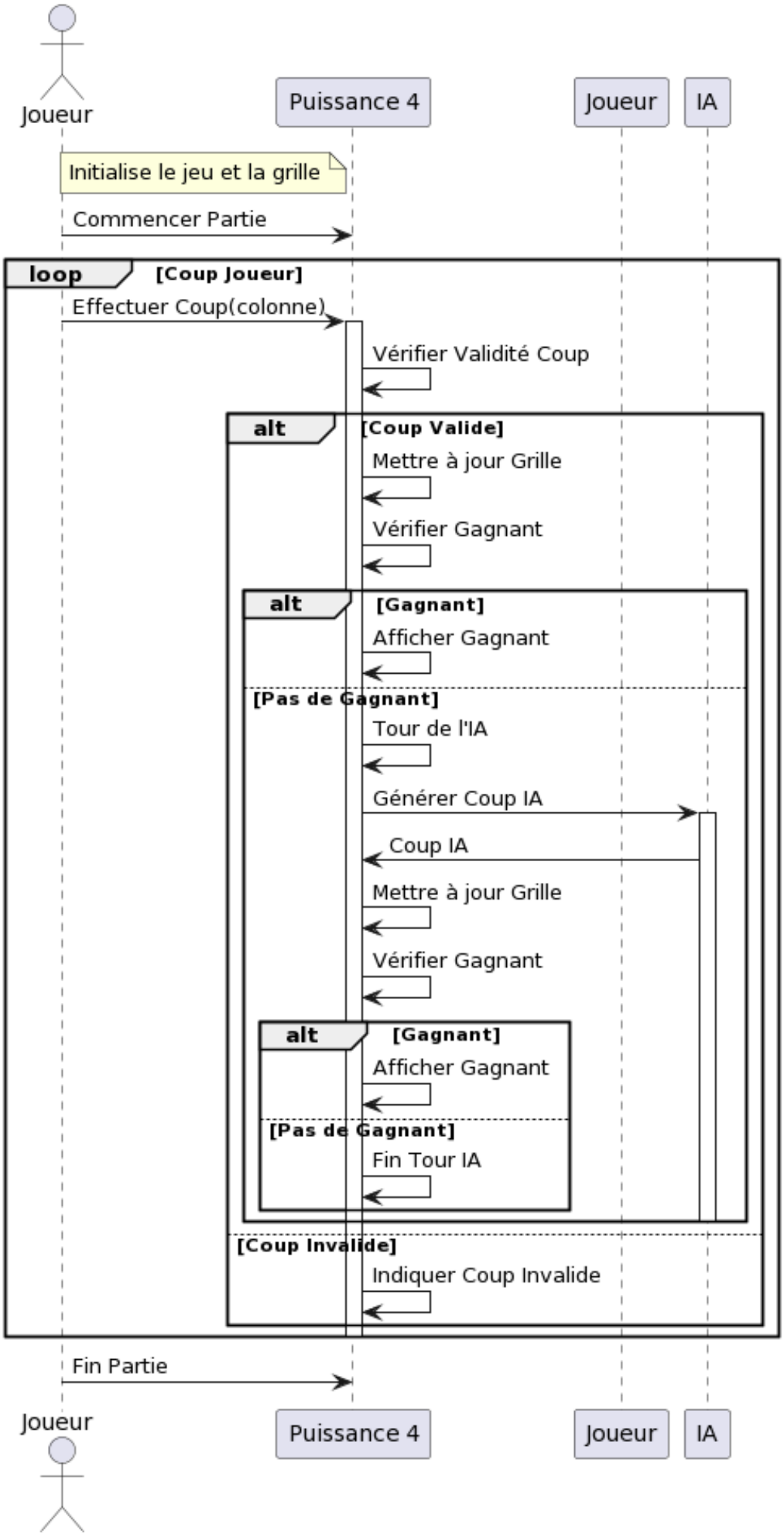
1. Initialisation (Diagramme de Cas d'Utilisation)

- **Le jeu commence par l'initialisation de la grille et des paramètres de jeu. Une instance du jeu est créée, définissant l'état initial et préparant l'interface pour les joueurs et l'IA.**



2. Déroulement de la Partie (Diagramme de Séquences)

- **Commencer Partie :**
  - **Le joueur déclenche le début de la partie. L'interface utilisateur affiche la grille de jeu vide et indique que c'est au tour du joueur humain de jouer.**
- **Boucle de Jeu :**
  - **Coup du Joueur :**
    - **Le joueur choisit une colonne pour y déposer un jeton. La validité du coup est vérifiée. Si le coup est valide, le jeton est placé dans la grille.**
    - **Vérification du Gagnant :**
      - **Après chaque coup, le jeu vérifie si le coup conduit à une condition de victoire. Si un gagnant est détecté, le jeu procède à l'affichage du gagnant.**
    - **Tour de l'IA :**
      - **Si aucun gagnant n'est détecté et que le coup est valide, le tour passe à l'IA qui génère son coup.**
      - **L'IA effectue son coup, et le jeu répète la vérification pour un éventuel gagnant.**
  - **Coup Invalide :**
    - **Si un coup du joueur est invalide, un message est affiché pour informer le joueur, et il est invité à jouer à nouveau.**


Joueur
Puissance 4
Joueur
IA
Initialise le jeu et la grille
Commencer Partie
loop
[Coup Joueur]
Effectuer Coup(colonne)
Vérifier Validité Coup
alt
[Coup Valide]
Mettre à jour Grille
Vérifier Gagnant
alt
[Gagnant]
Afficher Gagnant
[Pas de Gagnant]
Tour de l'IA
Générer Coup IA
Coup IA
Mettre à jour Grille
Vérifier Gagnant
alt
[Gagnant]
Afficher Gagnant
[Pas de Gagnant]
Fin Tour IA
[Coup Invalide]
Indiquer Coup Invalide
Fin Partie
Joueur
Puissance 4
Joueur
IA

3. Fin de la Partie

- **Le jeu se termine lorsque l'un des joueurs gagne ou si la grille est complète sans gagnant, auquel cas la partie se termine par une égalité. L'interface fournit une option pour recommencer une nouvelle partie.**

4. Affichage des Résultats

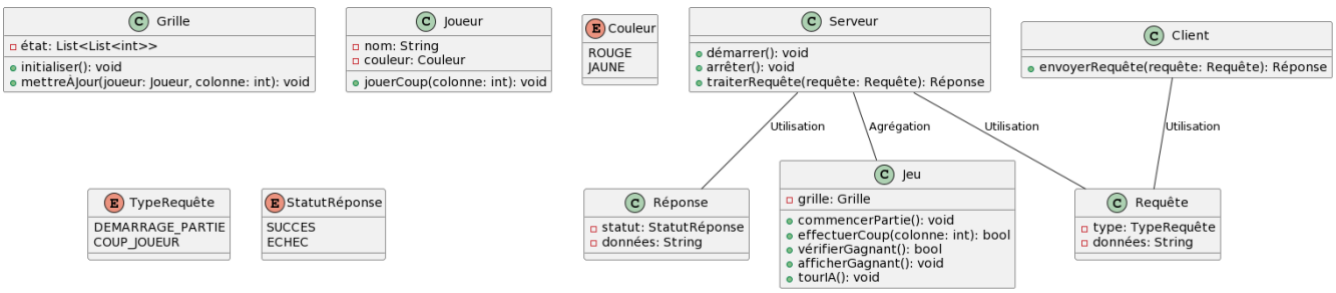
- **À la fin de la partie, l'interface utilisateur affiche le gagnant ou déclare une égalité. Les scores peuvent être mis à jour si le jeu inclut un système de suivi des scores sur plusieurs parties.**

5. Diagramme de Classe

Le diagramme de classe détaille les composants suivants :

- **Grille : Gère l'état du plateau de jeu.**
- **Joueur : Représente les joueurs et gère les couleurs des jetons.**
- **Jeu : Coordonne le déroulement de la partie, y compris le démarrage, le jeu des coups et l'affichage des résultats.**
- **Serveur : Gère la logique de traitement des requêtes du jeu.**
- **Client : Interface pour que les utilisateurs interagissent avec le jeu.**
- **Requête et Réponse : Pour la communication entre le client et le serveur, avec des requêtes et des réponses structurées.**
- **Les énumérations TypeRequête et StatutRéponse : Définissent les types de requêtes possibles et les statuts de réponse pour une communication claire.**

Cette conception détaillée fournit une vue d'ensemble du fonctionnement interne du jeu Puissance 4, depuis l'interaction utilisateur jusqu'au traitement côté serveur. Elle décrit le rôle de chaque classe et comment elles s'intègrent dans le processus global du jeu.



TESTS & VALIDATION

Pour garantir la fiabilité et la robustesse de notre application de jeu Puissance 4, une série de tests unitaires a été mise en place. Chaque composant clé a été soumis à des tests rigoureux pour valider son fonctionnement correct.

Exemples de Tests Unitaires :

1. **Test de la méthode initialiser de la classe Grille :**
    - **Objectif : S'assurer que la grille est correctement initialisée avec des dimensions 6x7 et ne contient que des valeurs nulles.**
    - **Code de Test :**

```
def test_initialiser_grille():
    grille = Grille()
    grille.initialiser()
    assert grille.etat == [[0]*7 for _ in range(6)]
```

2. **Test de la méthode mettre Ajour de la classe Grille :**

- **Objectif : Confirmer que les jetons sont correctement placés dans la colonne choisie.**
- **Code de Test :**

```
def test_mettre_a_jour():
    grille = Grille()
    grille.initialiser()
    joueur = Joueur('Test', 'ROUGE')
    colonne = 3
    grille.mettreAJour(joueur, colonne)
    assert grille.etat[5][colonne] == joueur.couleur
```

Ces tests sont exécutés automatiquement à l'aide d'un framework de test, tel que unittest ou pytest en Python, à chaque modification du code pour s'assurer qu'aucune régression n'est introduite.

Importance des Tests Unitaires : Les tests unitaires sont essentiels pour prévenir les bugs, faciliter les mises à jour et améliorer la qualité du code. Ils servent de documentation et fournissent une assurance que le code se comporte comme prévu, même après des modifications ou des extensions.

DEROULEMENT

Le projet Puissance 4 a été conduit par une équipe de trois membres, avec une stratégie d'exécution simultanée des tâches pour maximiser l'apprentissage.

Stratégie d'Équipe :

- **Une approche uniforme a été adoptée où chaque membre a participé à toutes les phases du projet.**

Répartition des Tâches :

- **Les sessions étaient organisées pour permettre à chaque membre d'exécuter la même tâche simultanément, renforçant ainsi la compréhension collective.**

Communication :

- **Des réunions hebdomadaires ont assuré le suivi du projet et la résolution collaborative des problèmes.**

Développement et Tests :

- **Le pair programming et les tests croisés ont été utilisés pour améliorer la qualité du code et la cohésion de l'équipe.**

Leçons Apprises :

- **Cette méthode a renforcé la polyvalence de l'équipe et a mis en évidence l'importance de la communication.**

CONCLUSION

Le projet de jeu Puissance 4 nous a offert l'occasion unique d'appliquer des concepts avancés de développement logiciel en Python. Nous avons enrichi notre compréhension de la programmation orientée objet, de la gestion d'état dans les jeux, et des bases de la communication réseau. Les tests unitaires ont été intégrés au cœur de notre processus de développement, ce qui nous a permis de construire un code fiable et de qualité.

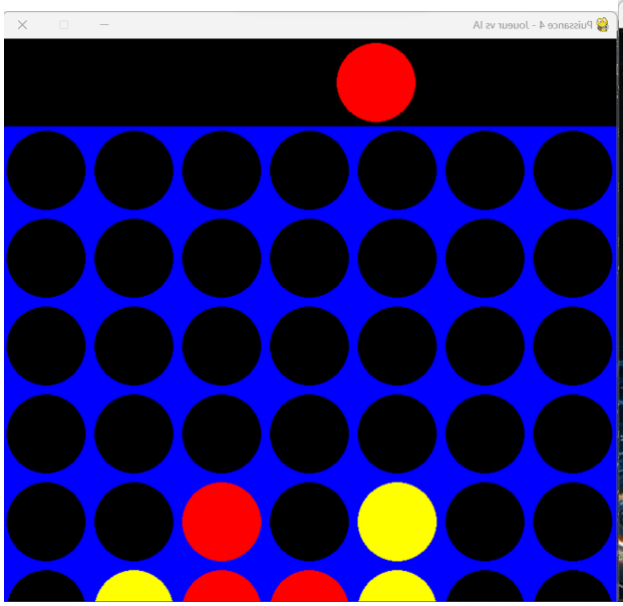
Cependant, le projet a également été une leçon d'humilité et de réalisme. Malgré nos efforts, nous avons rencontré des obstacles significatifs. La construction d'un serveur fonctionnel, une entreprise que nous n'avions jamais tentée auparavant, s'est avérée plus complexe que prévu. De même, le développement du jeu Puissance 4, bien qu'il ait été une tâche enrichissante, a exigé une quantité de temps et de ressources que nous avions sous-estimée.

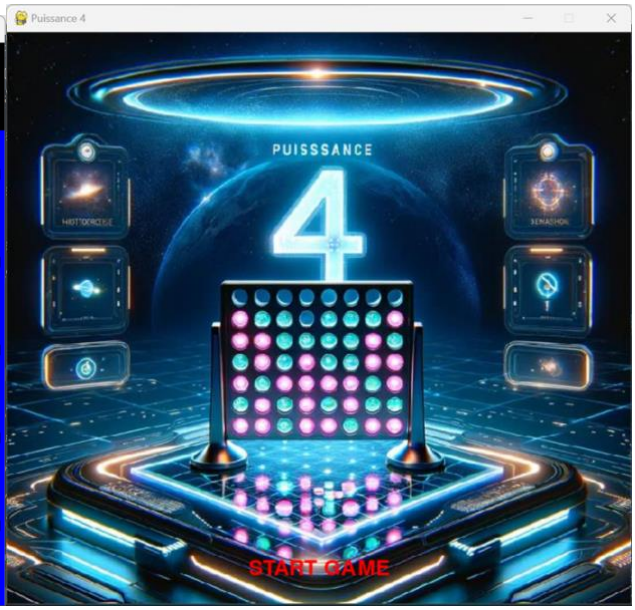
En définitive, bien que nous n'ayons pas accompli tous les objectifs que nous nous étions fixés, les leçons tirées de ce projet ont été inestimables. Nous avons appris la valeur d'une architecture logicielle soignée, l'importance de la clarté du code et des tests automatisés, et surtout, nous avons compris qu'un ambitieux projet de développement est un parcours semé de défis, d'apprentissage continu et d'adaptation.

APERCU


CHOOSE GAME MODE
RED WARRIOR vs Golden AI Synth
RED WARRIOR vs YELLOW WARRIOR


FÉCLIITÍCTATIONS!
RED WARRIOR EMERGES VICTORIOUS!
CONNECT FOUR! ...
5:444 45




Puissance 4
PUISSSANCE
4
START GAME