

Rapport de Planification

Etienne ALLAIN, Aymen BERRAJAA, Noha DOAIF, Alp JAKOP,
Giorgi KUCHUKHIDZE, Aymane MENFAA, Ezgi OZEL

Encadrante: Laurence Rozé
Novembre 2023



Table des matières

1	Introduction	2
2	Contexte	2
2.1	Acteurs	2
2.2	Périmètre fonctionnel	3
2.2.1	Interface Utilisateur	3
2.2.2	Intelligences Artificielles	3
2.2.3	Communication Client-Serveur	3
2.2.4	Tests et Validation	3
2.3	Eléments en entrées	3
2.4	Calendrier	4
3	Analyse de risques	4
3.1	Identification des risques	4
3.1.1	Les risques liés à la réalisation du projet	4
3.1.2	Les risques liés au déroulement du projet	5
3.2	Classification des risques	5
3.2.1	Probabilité des Risques	6
3.2.2	Répercussions des risques les plus critiques	6
3.3	Plan d'action	6
4	Organisation	7
4.1	Production	7
4.2	Qualification	8
4.3	Répartition du travail	8
4.4	Mode de pilotage	9
5	Approche d'une estimation du temps	9
5.1	Découpage des tâches	9
5.2	Conseils des personnes expérimentées/concernées	10
5.3	Assurer une marge de sécurité	11
5.4	L'étude de ce qui est parallélisable	11
5.5	Affectation d'un nombre d'heures par tâche	11
6	Planification	12
6.1	Hierarchie des tâches	13
6.2	Affectation des ressources par tâches	13
6.3	Le planning avec les jalons	14
7	Conclusion	14

1 Introduction

Ce projet s’articule autour de la création d’une application mobile dédiée au jeu des ”Cœurs”, se décomposant en deux parties parallélisables distinctes. La première partie se concentre sur le développement de l’application, offrant aux utilisateurs la possibilité de jouer contre des intelligences artificielles. La seconde partie est axée sur la création de ces intelligences artificielles, deux intelligences artificielles distinctes spécifiquement conçues pour exceller dans le jeu des Cœurs.

La première partie du projet consiste à concevoir une interface utilisateur en Flutter qui sera le client. Elle doit être capable de gérer différentes options de jeu et de fournir une interface attrayante pour l’utilisateur.

La seconde partie du projet se concentre sur le développement de deux intelligences artificielles en Python qui seront hébergées sur un serveur Flask, spécifiquement conçues pour performer dans le jeu des Cœurs. La décision de créer deux IA distinctes, plutôt qu’une seule, découle de notre volonté d’explorer diverses approches et de diversifier nos solutions.

Les deux intelligences artificielles envisagées s’appuient sur des algorithmes distincts, à savoir les réseaux de neurones et le Monte Carlo Tree Search (MCTS). Cette diversification d’approches a été choisie stratégiquement. Les réseaux de neurones sont idéaux pour apprendre à partir de données complexes en entrée et pour prendre des décisions, une compétence essentielle pour le jeu des Cœurs. D’un autre côté, le MCTS est couramment utilisé dans les jeux de cartes en raison de sa capacité à explorer efficacement les différentes possibilités.

Enfin, pour assurer le bon fonctionnement de l’ensemble, il est essentiel de connecter les intelligences artificielles côté serveur au client Flutter. Pour cela, une communication HTTP sera établie entre le client et le serveur. Cette interaction permet au serveur de collaborer avec les intelligences artificielles pour déterminer la meilleure carte à jouer, puis de transmettre cette décision au client.

2 Contexte

Dans cette section nous allons évoquer le contexte, c’est une partie importante puisqu’elle permet de fournir les bases nécessaires à la compréhension globale du projet et saisir le cadre global du projet.

2.1 Acteurs

Le projet du jeu des Cœurs rassemble une équipe de sept étudiants aux compétences variées, certains étant plus spécialisés en développement mobile et d’autres en intelligence artificielle, grâce à nos expériences dans des projets antérieurs au cours des années précédentes. Quatre d’entre nous se concentrent sur le développement des intelligences artificielles en Python, cherchant à élaborer une intelligence artificielle utilisant des réseaux de neurones et le Monte-Carlo Tree Search. L’objectif est d’enrichir l’expérience de l’utilisateur lorsqu’il joue contre des adversaires virtuels. Il est à noter que deux membres de cette équipe se retireront à partir de la phase de conception, car ils partiront en mobilité à l’étranger, soulignant ainsi l’importance de planifier soigneusement et de gérer la transition des responsabilités au sein de l’équipe. Du côté du développement Flutter, trois étudiants travaillent sur la création du client Flutter afin de fournir à l’utilisateur une interface de jeu agréable et d’assurer une communication bien synchronisée avec le serveur. Nous avons veillé à équilibrer la répartition des tâches pour compenser ces départs et préserver la cohésion de l’équipe. Le tout est encadré par

Mme Laurence Rozé, enseignante et chercheur dans le domaine de l'intelligence artificielle dans le laboratoire de l'IRISA, qui nous offre une supervision attentive et un soutien technique.

2.2 Périmètre fonctionnel

La définition précise du périmètre fonctionnel est cruciale pour garantir le succès du projet en répondant aux besoins des utilisateurs. Cette partie détaille les éléments inclus dans le périmètre fonctionnel du projet qui répondront aux besoins de l'utilisateur de l'application.

2.2.1 Interface Utilisateur

L'utilisateur a besoin d'avoir une interface de jeu agréable et simple d'utilisation donc la première composante majeure du périmètre fonctionnel est la conception d'une interface utilisateur conviviale et attrayante. Le client Flutter sera chargé de fournir une interface facilitant l'interaction de l'utilisateur avec le jeu. La gestion des cartes, des scores, et des informations contextuelles seront intégrées.

2.2.2 Intelligences Artificielles

L'utilisateur a aussi besoin de jouer contre des adversaires virtuels performants afin de pouvoir s'améliorer lui même au jeu, le périmètre fonctionnel inclut donc le développement d'algorithmes d'intelligence artificielle en Python performants. Deux approches distinctes seront mises en œuvre : une IA basée sur des réseaux de neurones et une autre utilisant le Monte-Carlo Tree Search (MCTS). Ces IA seront intégrées au serveur Python pour permettre aux utilisateurs de jouer contre des adversaires virtuels compétitifs.

2.2.3 Communication Client-Serveur

L'utilisateur a besoin d'avoir des temps de réponses courts afin d'avoir la meilleure expérience de jeu possible et donc cela induit une communication efficace entre le client Flutter et le serveur Python. Cela implique la synchronisation des actions de jeu, la gestion des requêtes et des réponses, assurant ainsi une expérience de jeu fluide et réactive.

2.2.4 Tests et Validation

L'utilisateur a aussi besoin d'avoir un jeu qui fonctionne très bien et donc sans bugs, c'est pour cela que dans le périmètre fonctionnel il faut prendre en compte les tests unitaires et d'intégration qui seront développés pour garantir la fiabilité et la stabilité du jeu.

2.3 Éléments en entrées

Les éléments en entrée pour le projet du Jeu des Cœurs sont constitués d'un ensemble diversifié de ressources, dont les bibliothèques, les projets personnels que l'on a réalisés les années précédentes, les documentations, et des articles pertinents. Au niveau des bibliothèques, on dispose de plusieurs bibliothèques existantes qui facilitent le développement, notamment la bibliothèque HTTP de Dart pour la communication via les requêtes HTTP, les bibliothèques intégrées à Flutter pour la construction de l'interface utilisateur, PyTorch pour la mise en œuvre des intelligences artificielles, Poetry pour la gestion des dépendances, et Flask pour créer le serveur Python. Les projets précédemment réalisés permettent aussi d'accélérer la compréhension des objectifs attendus. Parmi ces projets figure un client-serveur réalisé pour le jeu du morpion avec une IA en Python

et un client Flutter, ainsi que des projets d'intelligence artificielle de l'année dernière, tels qu'un réseau de neurones pour le jeu de backgammon et une implémentation d'intelligence artificielle pour le jeu Puissance 4. De plus, il existe divers mini-projets disponibles sur GitHub traitant du jeu des Cœurs pour bénéficier de pratiques existantes fonctionnelles.

La disponibilité de documentations détaillées sur les différentes bibliothèques utilisées constitue une ressource importante. De plus, l'équipe a accès à des articles scientifiques, dont un article détaillant le développement d'un jeu de Cœurs dans sa version néerlandaise avec une IA basée sur Monte-Carlo Tree Search (MCTS). Cette source d'information complémentaire fournira des perspectives utiles pour la mise en œuvre des intelligences artificielles.

2.4 Calendrier

La gestion du calendrier est essentielle pour assurer la réalisation efficace du projet du jeu des Cœurs. Les principales étapes du calendrier de livraison sont définies comme suit :

Rendu	Date Limite
Rapport de Conception	15 février 2024
Page Web	28 mars 2024
Rapport Final	02 mai 2024
Showroom et Livraisons	06 mai 2024
Soutenance Finale	07 mai 2024

Concernant la disponibilité des ressources, étant donné que les ressources nécessaires sont principalement des ressources informatiques, elles sont généralement disponibles en permanence. Ainsi, nous ne rencontrons pas de contraintes majeures liées à la disponibilité des ressources, ce qui permet une flexibilité dans la planification des différentes phases du projet.

La répartition des tâches tout au long du calendrier doit être soigneusement planifiée pour garantir une progression régulière et atteindre les jalons clés aux dates fixées. Ceci sera abordé plus loin dans le rapport.

3 Analyse de risques

Dans cette section dédiée à l'analyse des risques pour le projet de notre jeu des Cœurs, nous explorerons les divers défis et incertitudes qui pourraient influencer le déroulement du projet. Cette étape cruciale permettra d'anticiper les obstacles potentiels, de mettre en place des stratégies d'atténuation. En identifiant ces risques, l'équipe peut mieux se préparer à surmonter les éventuelles difficultés.

3.1 Identification des risques

3.1.1 Les risques liés à la réalisation du projet

Notre projet est confronté à divers risques liés à sa conception et à son développement, notamment dans les domaines de la performance, de la compatibilité, de la stabilité, et de la faisabilité technique.

Tout d'abord, le risque de performance peut être un énorme enjeu pour notre projet. Puisque dans notre cas, on pourrait avoir des pertes de performances liées à une communication trop lente

entre le client et le serveur, des pertes de performances liées à des réponses trop longues de la part des intelligences artificielles, ou bien des pertes de performances liées à l'intelligence artificielle qui peut ne pas être assez performante ce qui affecterait les besoins de l'utilisateur.

En ce qui concerne la compatibilité et la régression, le projet pourrait faire face à des problèmes d'adaptabilité sur différentes plateformes. La gestion de la compatibilité doit être abordée avec attention, en s'assurant que le jeu reste fonctionnel sur divers environnements (par exemple le jeu doit fonctionner quelque soit le navigateur utilisé par l'utilisateur, le jeu doit être compatible sur tous les systèmes d'exploitation).

Le risque de stabilité et de pérennité met en lumière la possibilité que les bibliothèques utilisées deviennent obsolètes ou ne soient plus maintenues, pouvant ainsi impacter le bon fonctionnement de notre application.

Enfin, la faisabilité technique, particulièrement liée au développement des intelligences artificielles en Python et à leur intégration avec le client Flutter, représente un défi potentiel.

3.1.2 Les risques liés au déroulement du projet

Dans le cadre du déroulement du projet, deux risques majeurs sont identifiés.

Tout d'abord, le risque de charge et de délais pourrait compromettre la progression globale. Il s'agit de la possibilité que les membres de l'équipe se trouvent confrontés à une surcharge de travail due à des imprévus techniques, impactant potentiellement la qualité des livrables et la capacité à respecter les délais prévus pour chaque phase du projet.

Ensuite, le risque lié au périmètre du projet due à la possibilité de modifications imprévues, susceptibles d'entraîner des ajustements dans les fonctionnalités de notre application.

3.2 Classification des risques

L'estimation de la probabilité des risques est essentielle pour évaluer leur impact potentiel sur le projet.

3.2.1 Probabilité des Risques

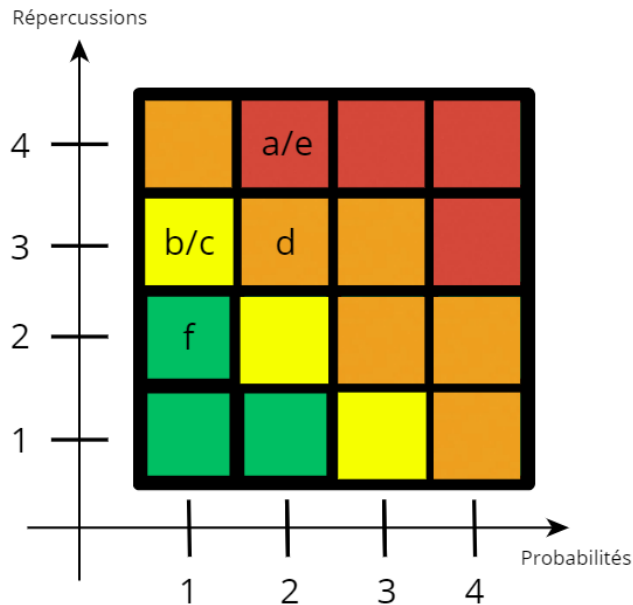


FIGURE 1 – Classification des risques liés à notre projet

- (a) Risque lié à la performance
- (b) Risque lié à la compatibilité
- (c) Risque lié à la stabilité
- (d) Risque lié à la faisabilité technique
- (e) Risque lié aux délais et la charge
- (f) Risque lié au périmètre d'objet

3.2.2 Répercussions des risques les plus critiques

Les risques les plus graves identifiés dans notre projet sont liés à la performance, à la faisabilité technique et aux délais, et les prévenir est essentielle pour garantir le succès du projet. Tout d'abord, les pertes de performances pourraient compromettre l'expérience utilisateur.

En ce qui concerne la faisabilité technique, notamment dans le développement des intelligences artificielles en Python et leur intégration avec Flutter, des difficultés techniques pourraient entraîner des retards importants et une dégradation de la qualité du produit final.

Enfin, le risque lié aux délais pourrait avoir des répercussions sur la planification globale du projet, mettant en danger la livraison du produit final dans les délais attendus. Des retards pourraient également entraîner des impacts sur d'autres phases du projet, augmentant la charge de travail et créant un effet domino sur l'ensemble du calendrier.

3.3 Plan d'action

Face aux risques les plus graves identifiés, il faut mettre en place un plan d'action pour contrer ces risques. Pour le risque lié à la performance, la stratégie adoptée sera axée sur une optimisation continue du code, des tests réguliers et une surveillance constante de la communication entre le

client et le serveur. Des actions spécifiques incluront l'optimisation des algorithmes pour améliorer les temps de réponse. En ce qui concerne le risque de faisabilité technique, la stratégie sera d'identifier les points critiques du développement des intelligences artificielles, et de mettre en place des sessions de revue de code entre les membres de l'équipe. Pour le risque lié aux délais, la stratégie consistera à mettre en place une gestion de projet rigoureuse avec des jalons clairs, des réunions régulières de suivi.

4 Organisation

4.1 Production

Le modèle choisi pour gérer ce projet est le cycle en V comme décrit ci-dessous (cf. Figure 1).

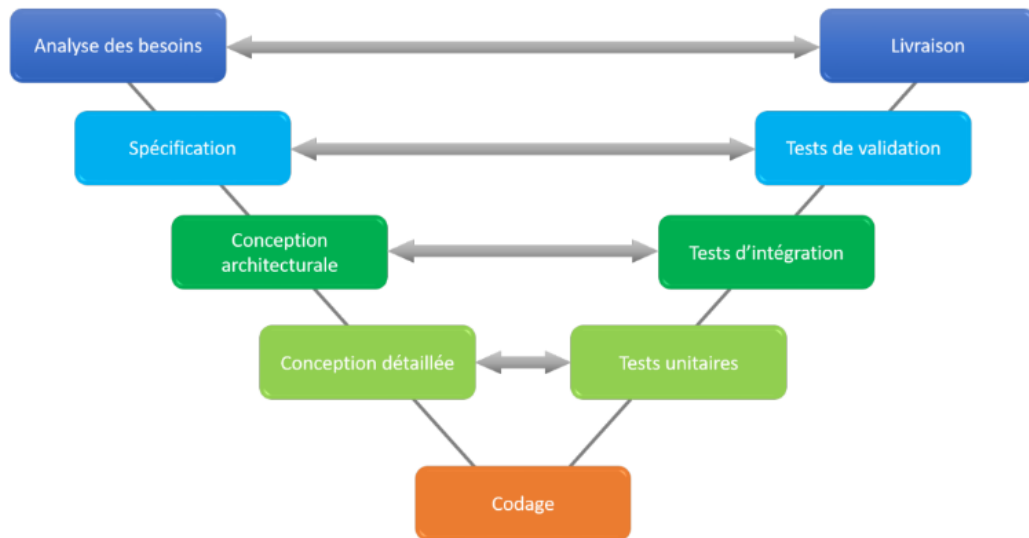


FIGURE 1 – Cycle en V

Nous avons opté pour le cycle en V pour ce projet parce qu'il nous permet de planifier étape par étape de manière claire et d'attraper les erreurs dès le début. Cela nous aide à garder le développement cohérent et de qualité, avec des réunions régulières pour suivre l'avancement et résoudre les problèmes rapidement.

Pour ce projet, différents éléments sont à produire. Tout d'abord, nous aurons à rédiger des documents liés à la présentation du projet. Chacun de ces documents peut être associé à un jalon pour l'organisation du projet. Il nous faut ainsi produire :

- **Rapport de Pré-étude et Spécification :**
Un document déjà rédigé, couvrant la présentation initiale du projet.
- **Rapport de Planification :**
Un document détaillant la planification initiale du projet.
- **Documentations Utilisateur et Technique :**
Des documents explicatifs détaillant le fonctionnement du projet.

- **Rapport de Conception Finale :**

Un document décrivant l'état final du projet et le travail accompli pendant la phase de développement.

- **Page HTML :**

Une page HTML détaillant le projet et ses fonctionnalités.

- **Présentations :**

Des supports de présentation (PowerPoint, etc.) pour la démonstration finale et les différentes soutenances.

- **Développement :**

- **Interface Client :** Une interface permettant de jouer au jeu de cœurs.
- **Code de l'Application :** Le code source Flutter de l'application.
- **Code des IA :** Le code source Python des intelligences artificielles.

Le code source de ces programmes sera accessible via le gestionnaire de version GitLab de l'INSA Rennes. Ce processus permettra une gestion efficace des versions et une collaboration transparente au sein de l'équipe.

4.2 Qualification

Une autre partie importante dans l'organisation du projet consiste en la vérification de la qualité des différents éléments produits. En effet, il est nécessaire de vérifier que nous répondons aux objectifs que nous nous fixons (algorithmes, code) mais aussi aux besoins du client (cahier des charges). Ainsi, nous avons décidé d'utiliser différents outils et techniques :

- les tests unitaires : ces tests permettent de vérifier le bon fonctionnement d'une partie spécifique d'un programme,
- les tests d'intégration : ces tests permettent de vérifier que les différentes parties du programme que nous allons développer fonctionnent bien ensemble,
- les tests de validation : ces tests permettent de vérifier que notre code répond bien aux spécifications que nous avons établi dans la seconde phase de notre projet.

Enfin, tout au long du développement, il sera important de communiquer et faire tous les tests avec notre encadrante afin de nous assurer que notre implémentation correspond à ses attentes. Cette communication sera importante dans la validation de notre travail mais aussi en cas de changement d'attente ou d'apparition d'un nouveau besoin.

4.3 Répartition du travail

Pour optimiser l'organisation du projet, nous devons répartir efficacement le travail parmi les cinq membres du groupe pour gagner du temps et accroître l'efficacité. Chacun sera assigné à un module tout en collaborant sur les autres. Le responsable d'un module ne travaillera pas exclusivement sur celui-ci, mais restera conscient de son avancement, des besoins et des problèmes, qu'il remontera lors des réunions.

Pour minimiser les conflits pendant le développement, nous adopterons une approche avec une branche principale (master) sur le gestionnaire de version, où aucune modification directe ne sera autorisée. Après le développement, chaque développeur soumettra ses modifications au groupe via une "merge request". Les autres membres réviseront le code, suggéreront des ajustements, et une fois qu'un consensus sera atteint, le code sera fusionné à la branche principale.

Le groupe utilisera GitLab comme plateforme d'hébergement, exploitant la fonctionnalité de MR pour permettre une collaboration transparente. Pour maximiser l'efficacité, chaque personne devra travailler sur des "tranches" spécifiques du code, apportant des modifications à des endroits ou comportements précis sans toucher à trop de fichiers pour éviter les conflits.

4.4 Mode de pilotage

Pour avancer de manière efficace, il est nécessaire de mettre en place un mode de pilotage au sein de l'équipe. Pour s'assurer de l'avancée du projet, des réunions sont organisées de manière hebdomadaire. Celles-ci sont l'occasion de partager les avancées de chacun et de faire part des problèmes vis-à-vis du travail en cours au reste du groupe. Avec Mme Rozé ou au sein du groupe, cela permet d'éclaircir le travail à effectuer. Les réunions suivent généralement le format suivant, qui correspond à un format classique de "daily stand-up" dans une entreprise :

- un tour de table est effectué, durant lequel chacun fait le compte-rendu de ce qu'il a fait au cours de la semaine écoulée et de ses potentiels problèmes
- un échange est fait sur les choses qui sont à faire avant la réunion suivante et sur les problèmes qui ont été évoqués
- enfin un tour de table est fait pour décider ce sur quoi chacun va travailler dans la semaine qui vient.

5 Approche d'une estimation du temps

Pour bien planifier un projet, il est crucial d'estimer correctement le temps requis pour accomplir les différentes tâches. Ces estimations jouent un rôle clé dans l'organisation du projet, orientant sa progression. Dans cette section, nous explorerons les méthodes et outils utilisés pour évaluer le temps nécessaire à la réalisation des travaux prévus. Une estimation précise du temps permet de mieux anticiper les étapes à venir, facilitant ainsi une planification efficace.

5.1 Découpage des tâches

Afin de déterminer avec précision la durée nécessaire pour mener à bien notre projet, nous avons entrepris de le découper méthodiquement en différentes tâches. Cette démarche nous a permis d'attribuer à chaque tâche une estimation temporelle plus précise. Initialement, le projet a été subdivisé en 9 tâches distinctes, puis chacune de ces tâches a été décomposée en sous-tâches. Cette approche détaillée vise à rendre les estimations plus précises, en considérant que des tâches plus petites facilitent une meilleure évaluation du temps nécessaire. Voici le résultat de notre découpage du projet en tâches et sous-tâches pour optimiser l'estimation du temps requis à chaque étape :

- Développement de l'interface graphique (UI/UX)

- Diagramme UML
- Intégration des maquettes dans le code
- Intégration de la logique du jeu
- Mise en place d'une communication avec le backend

- Architecture Backend
 - Diagramme UML
 - Configuration de l'environnement serveur
 - Intégration de la logique du jeu
 - Création des services et contrôleurs pour communiquer avec le client
- Implémentation de l'algorithme MCTS
 - Prise en main des ressources de MCTS
 - Implémentation de l'algorithme MCTS
 - Entraînement de l'algorithme MCTS
 - Tests unitaires pour valider le comportement de l'IA
- Intégration des réseaux de neurones
 - Prise en main des ressources
 - Implémentation de l'algorithme
 - Entraînement de l'IA
 - Tests unitaires
- Rapport
 - Rapport de Conception
 - Rapport Final
 - Page html
- Qualification
 - Tests d'intégration
 - Tests de validation
- Documentation et Finalisation
 - Rédaction de la documentation technique et des commentaires dans le code
 - Evaluation de la performance du jeu
- Préparation de la soutenance
- Préparation de la démonstration

5.2 Conseils des personnes expérimentées/concernées

Pour évaluer la durée nécessaire à chaque tâche, nous avons recherché des conseils auprès de personnes ayant déjà travaillé sur des projets similaires. Cependant, ces projets ne présentaient que peu de similitudes avec le nôtre, limitant ainsi les informations que nous pouvions en tirer.

Néanmoins, nous avons pu bénéficier de l'expertise de notre encadrant, possédant une expérience avérée dans ce domaine spécifique.

En parallèle, nous avons également pris en compte l'avis des personnes impliquées directement dans l'exécution des tâches pour estimer plus précisément les durées requises.

5.3 Assurer une marge de sécurité

Inclure une marge de sécurité dans l'évaluation du temps permet de flexibiliser le déroulement du projet. Quelle que soit la précision de l'estimation, des difficultés, des imprévus ou des changements de direction sont souvent inévitables. Avoir une marge supplémentaire augmente donc les chances de terminer le projet dans les délais impartis. Cependant, définir cette marge doit prendre en considération le coût supplémentaire pour le client par rapport au temps prolongé

5.4 L'étude de ce qui est parallélisable

Dans le cadre de ce projet, certaines contraintes temporelles se manifestent, limitant la possibilité de mener à bien plusieurs tâches de manière simultanée. Près de la moitié des activités prévues dans le plan ne peuvent être initiées en parallèle. En effet, la progression du projet est structurée de manière séquentielle, nécessitant l'achèvement des 5 premières tâches pour entreprendre les 4 dernières. Ces dernières, incluant les tests et le débogage, la documentation et la finalisation, la préparation de la soutenance et la préparation de la démonstration, sont interdépendantes des réalisations précédentes. Cette organisation séquentielle impose une planification rigoureuse pour garantir une exécution fluide et efficace du projet, en tenant compte des dépendances entre les différentes étapes.

5.5 Affectation d'un nombre d'heures par tâche

Nom de la tâche	Nombre d'heures
1. Développement de l'interface	310 h
Diagramme UML	10 h
Intégration de maquettes dans le code	150 h
Intégration de la logique du jeu	100 h
Mise en place de la communication client/serveur	50 h
2. Architecture backend	190 h
Configuration de l'environnement serveur	20 h
Intégration de la logique du jeu	100 h
Diagramme UML	10 h
Création de service pour communiquer avec le client	60 h
3. Implémentation de l'algorithme MCTS	200 h
Prise en main des ressources	20 h
Implémentation de l'algorithme	120 h
Entraînement de l'IA	30 h
Tests unitaires	30 h
4. Implémentation de réseau de neurones	200 h
Prise en main des ressources	20 h
Implémentation de l'algorithme	120 h
Entraînement de l'IA	30 h
Tests unitaires	30 h
5. Rapports	115 h
Rapport de conception	50 h
Rapport final	50 h
Page HTML	15 h
6. Qualification	60 h
Tests d'intégration	40 h
Tests de validation	20 h
7. Documentation et finalisation	40 h
Rédaction de la doc technique et commentaires du code	20 h
Evaluation de la performance du jeu	20 h
8. Préparation de la soutenance	40 h
9. Préparation de la démonstration	15 h
Total	1 170 h

TABLE 1 – Tableau des tâches

6 Planification

Après avoir décidé de ce qu'il fallait faire avant la toute fin de ce projet, nous avons divisé les différents objectifs et tâches en catégories et estimé le temps nécessaire pour les réaliser. Nous avons travaillé sur MS Project pour la planification de ces tâches et leur visualisation sous forme de diagramme de Gantt.

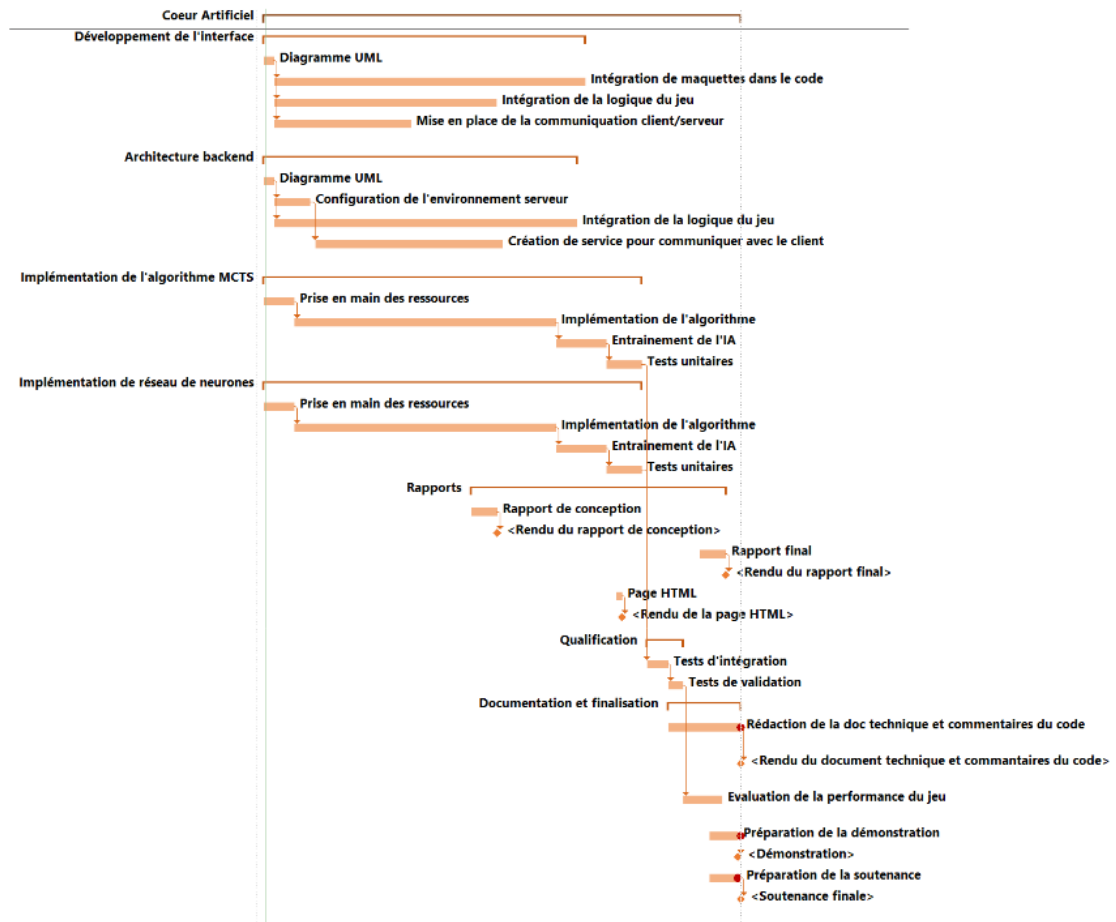


FIGURE 2 – Diagramme de Gantt

6.1 Hiérarchie des tâches

Afin de bien planifier notre projet, nous avons divisé nos objectifs en de nombreuses catégories différentes contenant des tâches respectives. Certaines de ces tâches ont dû être hiérarchisées car elles ne pouvaient pas démarrer avant la fin de la dernière. Par exemple, les "tests d'intégration" et "tests de validation" n'ont pu démarrer avant la finalisation de toutes les sous-catégories du projet (les IA, l'interface-client et la communication entre elles).

6.2 Affectation des ressources par tâches

Pour chaque tâche, nous avons estimé le temps qu'il nous faudrait pour réaliser la tâche. De plus, pour chaque tâche respective, nous avons décidé du temps que chaque membre ou équipe du projet consacrerait au travail sur la tâche en vue de sa réalisation.

Dans notre projet, nous avons quatre équipes qui travailleront sur certaines parties du projet. Mais comme il y a cinq membres dans notre projet mais quatre équipes, un membre travaillera pour deux équipes différentes, l'équipe *Backend* et l'équipe *Interface Graphique*. Les équipes *MCTS* et

Réseaux de Neurones ne contiendront qu'un seul membre qui sera responsable de toutes les tâches de cette équipe. En plus de cela, toutes les équipes du projet travailleront ensemble sur les points qui concernent l'ensemble du groupe comme les rapports et la soutenance.

	Nom de la ressource ▼	Unités max ▼	Type ▼	Base Calendar ▼	Initials ▼
	Equipe Interface Graphique	150%	Work	Standard	IG
	Equipe Backend	150%	Work	Standard	B
	Equipe MCTS	100%	Work	Standard	MCTS
	Equipe Réseaux de Neurones	100%	Work	Standard	RN

FIGURE 3 – Tableau des Ressources

6.3 Le planning avec les jalons

Nous avons utilisé différents jalons tout en travaillant sur la planification. Tous les jalons sont définis grâce à une contrainte de type “*Fin au plus tard le*” avec la date précise qui contraint le jalon. Nous avons les jalons suivants pour notre projet :

- **”Rendu du rapport de conception”** correspond à la date obligatoire de rendu du rapport de conception, c’est-à-dire le 15/02/2024,
- **”Rendu de la page HTML”** correspond à la date obligatoire de rendu de la page HTML, c’est-à-dire le 28/03/2024,
- **”Rendu du rapport final”** correspond à la date obligatoire de rendu du rapport final, c’est-à-dire le 02/05/2024,
- **”Démonstration”** correspond à la date de showroom-livraison du projet, c’est-à-dire le 06/05/2024,
- **”Soutenance finale”** correspond à la date de la soutenance finale du projet, c’est-à-dire le 07/05/2024,
- **”Rendu du document technique et commentaires du code”** correspond à la date de rendu du document technique du projet, c’est-à-dire le 07/05/2024.

7 Conclusion

Dans ce rapport, nous avons établi l’organisation et la planification de notre travail pour notre projet. Nous avons dans un premier temps rappelé le contexte de nos travaux et identifié les risques (comme les risques dans le développement des différentes intelligences artificielles utilisant Python et l’utilisation de Flutter pour l’application mobile) qui pourraient avoir un impact sur l’avancement du projet. Ensuite, nous avons choisi notre méthode de travail et les outils que nous utiliserions pour assurer un bon développement. Nous avons défini qu’il nous fallait quatre grandes catégories pour répartir notre travail en tâches : *l’interface graphique*, *le backend*, *MCTS* et *réseaux de neurones* et constitué les équipes nécessaires pour être en charge desdites tâches. Enfin, nous avons estimé le temps nécessaire pour réaliser chaque tâche et planifié la réalisation de ces tâches au fil du temps.

Ce rapport peut donc servir de référence lors de la phase de développement. Le diagramme de Gantt sera particulièrement utile, car il nous permettra de nous rendre compte des éventuels retards dans notre projet. Cela nous permettra de répondre rapidement à tout problème dès qu'il survient. Sans ces outils, un retard dans le développement du projet ne pourrait être détecté qu'à l'approche des dates de livraison. L'équipe n'aurait donc pas suffisamment de temps pour réagir et remédier à ces éventuels problèmes. Sachant organiser notre travail, nous pouvons désormais passer à la phase suivante de notre projet.