**Université de Cergy-Pontoise**

**RAPPORT**

pour le projet Génie Logiciel
**Licence d'Informatique deuxième année**

sur le sujet

# Projet GLP : Maritime

rédigé par

**Rayan BACCOUCHE et Amar GUERROUF**


Janvier à Avril 2023

# Table des matières



# Table des figures

## Liste des tableaux

## Remerciements

Nous tenions à remercier Monsieur Tianxio LIU lors de la réalisation de ce projet, pour sa pédagogie et le contenu éducatif fournis. Ainsi que Monsieur Marc LEMAIRE pour sa préparation aux exposés, la réalisation d'un site, la rédaction d'un rapport ainsi que d'un diaporama professionnel.

Le travail demandé était très clair, pas facile mais possible et vous avez été toujours disponible pour nous, que ce soit pour nous apporter des réponses ou pour voir notre progression dans le projet, ainsi que de nous donner des conseils.

Nous vous remercions pour cette année et nous aimerions vous dire à l'année prochaine.

# 1 Introduction

La raison pour laquelle nous avons mis dans notre liste de choix le sujet “Maritime” est parce que celui-ci est très complet et fort intéressant dans son déroulement, qu’il nous inspirait et parce que celui-ci avait 5/5 en matière de difficulté IHM Graphique, un atout sur lequel nous avions quelque peu un retard et sur lequel nous souhaitions nous améliorer.

## 1.1 Contexte du projet

“Maritime” est un jeu de gestion de conquête maritime. Le but du joueur sera de conquérir le monde, cela en gérant une flotte de navires, le joueur devra naviguer à travers les océans pour acheter et vendre des marchandises dans différents ports pour s’enrichir, mais aussi attaquer des bateaux, ports et pays. Le joueur prendra donc le contrôle d’une flotte de navires de marchandises et d’attaque.

## 1.2 Objectif du projet

L’objectif de ce projet aura été de savoir nous organiser dans un travail à deux, nous aider d’outils pour cela, savoir communiquer, reconnaître les forces et faiblesses de chacun, ainsi que nous améliorer en matière de notions IHM graphique. Et plus globalement, notre objectif sera de vous proposer un jeu concret et intéressant, qui saura trouver son public.

## 1.3 Organisation du rapport

Dans ce rapport nous allons vous présenter l’idée générale que nous avions du jeu, son déroulement, les résultats attendus et les modifications de base que nous avons dû faire, les contraintes auxquelles nous avons dû faire face ainsi que leurs solutions.

Nous vous présenterons l’architecture du programme, de l’IHM graphique et de comment nous avons pu raisonner dans nos choix de classes, de traitements et d’organisation.

Enfin, nous vous présenterons le résultat final, le déroulement d’une partie et nous finirons par notre ressenti au cours de ce projet ainsi que les éventuelles mises à jour que nous pourrions y ajouter.

# 2 Spécification du projet

Nous avons présenté l'objectif du projet dans la section 1. Dans cette section, nous vous présenterons la spécification du logiciel réalisé. Cela correspond principalement au document de spécification du projet (cahier des charges).

## 2.1 Notions de base et contraintes du projet

### 2.1.1 Fonctionnement général du logiciel

Maritime est un jeu de conquête maritime en temps réel, qui implique la gestion d'une flotte de navires, l'achat et la vente de marchandises dans différents ports pour s'enrichir, ainsi que l'attaque de bateaux, ports et pays.

Les objectifs de ce projet sont donc de proposer un jeu de gestion, de commerce et d'échanges. Des actions qui offriront la possibilité au joueur de conquérir ce monde maritime grâce à ses compétences stratégiques.

Les spécifications du jeu incluent la mise en place d'un système de temps, de déplacements de navires, d'améliorations des bateaux, de conquête de ports et de pays. Le jeu doit être conçu pour être joué en trois niveaux de difficulté différents. Le développement du jeu doit tenir compte des notions de base et des contraintes d'un jeu de gestion, ainsi que des fonctionnalités spécifiques de ce que donnerait un jeu de conquête maritime.

### 2.1.2 Notions et terminologies de base

**Jeu de gestion :** Dans un jeu de gestion, le joueur est chargé de diriger un ensemble, dans notre cas un ensemble de bateaux. L'objectif est de prendre des décisions stratégiques qui permettront de se développer tout en affrontant les obstacles et les défis. La simulation, la stratégie et la gestion des ressources peuvent toutes être présentes dans certains types de jeux de gestion.

**Temps réel :** Dans ce jeu en temps réel, il y a une contrainte temporelle à respecter dans l'exécution des traitements et ces résultats. Ainsi, l'avancement du temps doit respecter les contraintes du jeu, au lieu d'avoir une exécution du programme qui se termine « instantanément ».

**Sauvegarde du jeu :** La fonction de sauvegarde d'un jeu permet de conserver l'état actuel du jeu pour une utilisation ultérieure. Ainsi, si vous devez quitter le jeu avant de l'avoir terminé, votre progression ne sera pas perdue. Pour notre projet, les sauvegardes sont faites manuellement, et elles sont conservées localement sur, par exemple, un fichier texte.

**La difficulté :** Selon la difficulté choisie et le temps écoulé dans le jeu, la complexité augmente. À mesure que le joueur avance dans le jeu, la probabilité de rencontrer des pirates augmente, ainsi que leur flotte. Le joueur ne doit donc pas être trop en retard dans la gestion de sa flotte et dans les choix qu'il entreprend s'il souhaite gagner, et nc conquérir le monde maritime.

**IHM :** L'IHM ("Interaction Humain Machine") est l'ensemble des composants visuels et des commandes qui aident les utilisateurs à interagir avec le jeu et à choisir des actions pour atteindre des objectifs. Elle est essentielle pour une expérience de jeu amusante et efficace, car elle permet aux joueurs de se déplacer dans le jeu avec facilité et d'effectuer des actions rapidement. Pour ce jeu il est attendu 5 étoiles en matière d'IHM, cela ne signifie pas nécessairement une qualité élevée ou de la 3D, mais de contenu variable et en quantité. (plusieurs interfaces : menu, en jeu, en combat).

**Contrôle :** Toutes les actions se font à la souris. Elle est utilisée pour naviguer, sélectionner et s'engager avec divers éléments de l'interface liés au jeu.

### 2.1.3 Contraintes et limitations connues

**Langage de programmation :** nous avons constaté que Java possède tous les packages requis pour créer notre jeu, y compris les bibliothèques graphiques. En utilisant ces outils, nous avons pu accélérer le processus de développement et nous concentrer sur les éléments essentiels de notre jeu.

**Portabilité du jeu :** Grâce à la portabilité de Java, les programmes créés dans ce langage peuvent être utilisés sur une variété de plateformes sans nécessiter d'ajustements majeurs. Cela peut simplifier l'adaptation du jeu à de nombreuses plates-formes.

**Limitations de langue :** Le fait que le jeu que nous avons créé ne soit disponible qu'en français peut empêcher certains joueurs d'y jouer. Il pourrait être difficile pour les joueurs qui ne parlent pas français de comprendre les règles et les éléments du jeu. Il serait intéressant de proposer le choix depuis l'interface d'accueil de mettre le jeu en français ou en anglais, la langue universelle qui rendrait accessible le jeu aux utilisateurs du monde entier.

**Limitation techniques :** Nous avons assuré à ce que le jeu fonctionne parfaitement sur toutes les plateformes. Nous n'avons pas rencontré de contraintes techniques importantes lors de la conception du jeu, bien que nous ayons utilisé une interface utilisateur facile à utiliser. Quel que soit le niveau d'expertise technologique du joueur, nous avons veillé à ce que le jeu soit aussi accessible que possible et à ce que l'utilisateur puisse vivre une expérience de jeu agréable.

## 2.2 Fonctionnalités attendues du jeu

### 2.2.1 Le jeu doit pouvoir :

- Disposer d'un menu de départ intuitive, d'une interface simple et compréhensible entourant une carte fixe en vue aérienne, d'images de bateaux, d'une scène "zoomée" pour les combats entre le joueur et des pirates ou navires neutres.
- Stocker les informations d'une partie de jeu dans un fichier texte : numéro de partie, difficulté choisie, temps dans le jeu, emplacements des bateaux, ports conquis, pays conquis et missions de commerce en cours.
- Incrémenter continuellement le temps dans la partie de jeu à chaque seconde, sauf quand un combat a lieu. Et ainsi, à chaque 60 secondes (valeur choisie arbitrairement) augmenter d'un jour. L'on arrêtera d'incrémenter lorsqu'un combat a lieu ou que le joueur met en pause la partie de jeu.
- Se mettre en pause lorsque le jeu le décide, et ainsi le temps et la progression n'augmenteront pas.
- Afficher les possibilités de déplacement pour chaque bateau sélectionné, et afficher explicitement un message d'erreur en cas ou le joueur choisit un déplacement impossible.
- Tracer un chemin visuel du parcours des bateaux envoyés en mission d'un port A à un port B.
- Disposer d'une page aide ou d'un site internet pour expliquer les différentes règles de jeu.

### 2.2.2 Le joueur doit pouvoir :

- Se retrouver sur un menu à choix lorsqu'il lance le jeu, avec :
  - [Continuer] : (grisé si aucune partie n'existe) continuer une de ses parties
  - [Nouvelle partie] : pour lancer une nouvelle partie
  - [Options] : pour modifier les paramètres de préférences sonores et le mode plein écran
  - [Quitter] : pour quitter le jeu

  Il peut donc reprendre sa partie, en créer une nouvelle, changer certaines options. Lorsqu'il a plusieurs parties et qu'il clique sur [Continuer], il peut choisir parmi la liste de ses parties et en supprimer certaines. Le nom de la partie est sa date de création et, il y figure la date à laquelle la partie a été laissée pour la dernière fois.
- Créer et gérer une flotte de navires, chacun ayant ses propres compétences et qualités

— Parcourir la mer pour acheter et échanger des marchandises, s'arrêter dans des ports pour faire réparer et réapprovisionner leurs navires. Et aussi, s'engager dans des combats tactiques contre un ou plusieurs adversaires, détruisant leurs navires pour conquérir les différents ports.
— Améliorer sa flotte et rendre son armée plus efficace

# 3 Conception et réalisation du projet

Une conception bien réfléchie implique une réalisation dans les meilleures conditions, c'est pourquoi nous avons accordé toute l'attention nécessaire à cette phase, durant laquelle nous avons défini la structure du projet et identifié les fonctionnalités en partant des plus importantes aux rudimentaires

## 3.1 Architecture globale du logiciel

Lors de la conception d'un projet informatique, il est courant de commencer par créer un diagramme de classes afin de représenter les différentes entités et relations du système. Dans la figure 1, on peut voir le diagramme d'architecture initial de Maritime.

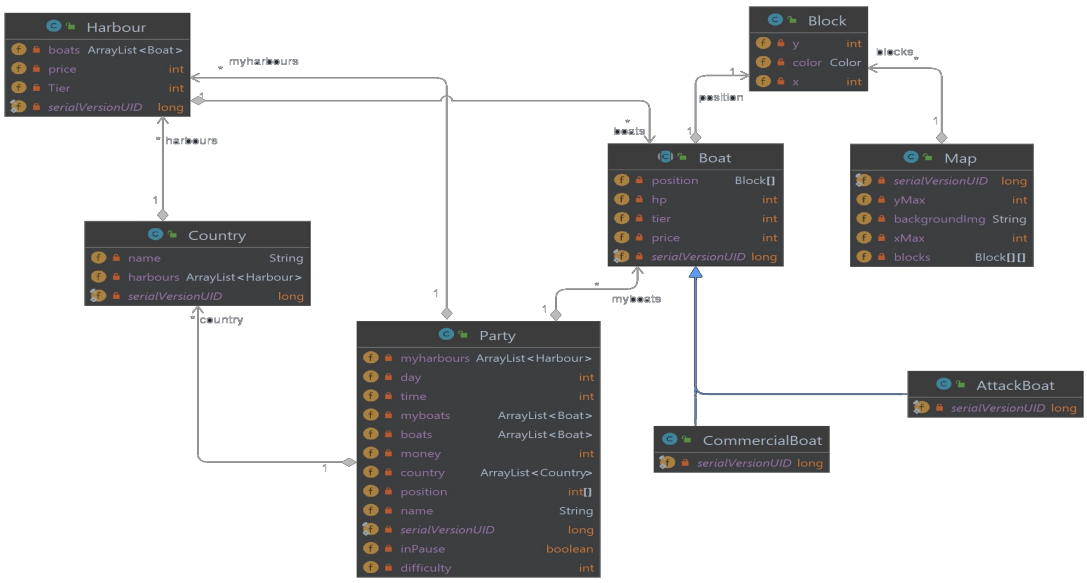



FIGURE 1 – Le diagramme d'architecture initial de Maritime

Cependant, une fois que le développement réel commence, il est possible que des ajustements et des améliorations soient nécessaires. Cela peut donc entraîner des changements dans la structure du projet, ce qui peut se refléter dans le diagramme de classes initial. Ainsi, il est important de mettre à jour régulièrement le diagramme de classes tout au long du développement, afin de s'assurer que la représentation du système reste précise et à jour. À la fin du projet, le diagramme de classes final doit être une représentation fidèle de la structure réelle du système tel qu'il a été développé, il est présenté dans la figure 2 suivante :

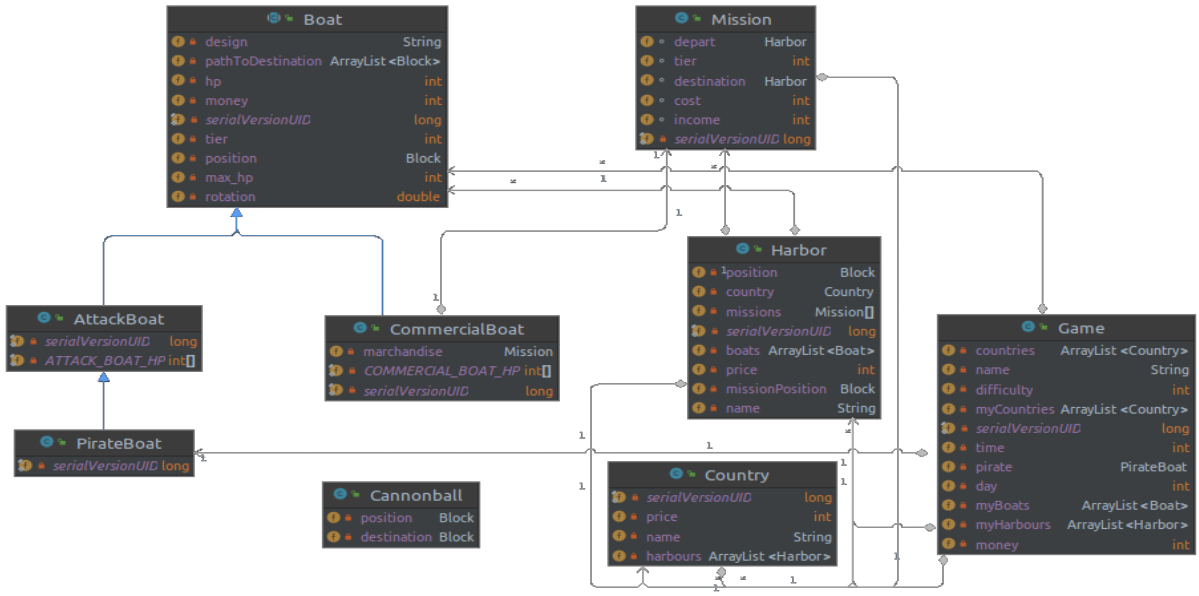



FIGURE 2 – Le diagramme d'architecture final de Maritime

Chaque objet contenu dans "Game" (la partie de jeu) contient une clé de sérialisation ("serialVersionUID") qui correspond à la version du jeu affichée sur l'écran d'accueil, lorsqu'elle change dans le programme, celui-ci rendra les parties précédemment enregistrées non lisibles avec notre version courante.

## 3.2 Conception des classes de données

Dans le programme nous retrouvons principalement la classe maîtresse "Game", ainsi que ses objets la composant, tous nécessaires pour une sauvegarde de la partie, et ainsi pouvoir la reprendre plus tard sans rien y perdre.

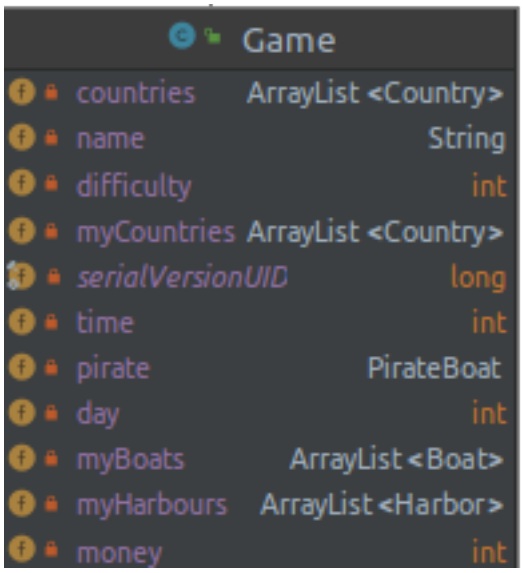



FIGURE 3 – La classe Game et ses attributs

La liste "countries", sera composée des pays sur la carte, chacun composés de ports, qui eux-mêmes comportent des bateaux.

## 3.3 Conception des traitements (processus)

### 3.3.1 Noyau de traitement

Pour contrôler la logique et faire les différents traitements du jeu, nous utilisons la classe "Engine" (voir figure 4). Cette classe gère et stocke seulement temporairement les informations : trajectoires de bateaux ou de boulets de canons envoyés, mais aussi le bateau ou port actuellement sélectionné.

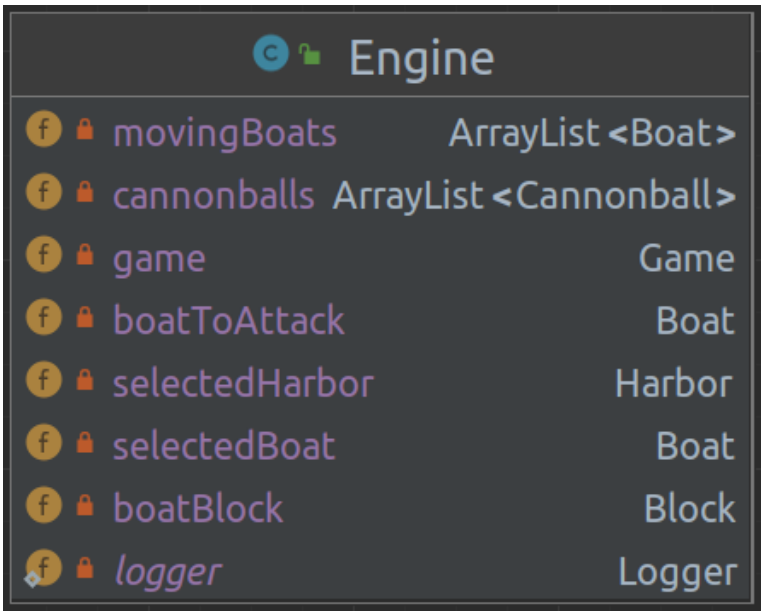



FIGURE 4 – La classe Engine et ses attributs

Voici une liste des principales fonctions de la classe "Engine" (voir figure 4) :

1. Gestion des bateaux :
   - Garder la trace de la position, de l'orientation et de l'état du bateau en mouvement.
   - Modifier les informations relatives au bateau en fonction des activités du joueur (par exemple, déplacer ou faire pivoter un bateau).
   - Organiser les bateaux en mouvement dans la liste des bateaux en mouvement.
   - Trouver le chemin le plus court et le plus rapide pour arriver à la destination.
2. Gestion des boulets de canon :
   - Stockage des données relatives aux boulets de canon, y compris leur position et leur état.
   - La modification des données relatives aux boulets en réponse aux actions du joueur, telles que le tir d'un canon.
3. Gestion de la sélection utilisateur :
   - Stockage du bateau et du port que le joueur a choisi.
   - Manipulation de la sélection en fonction des activités du joueur (par exemple, changement de sélection).

### 3.3.2 Boucle de déroulement

Pour faire les différentes animations du jeu, nous utilisons la classe "Thread" qui permet de lancer une méthode "run()", dans notre classe "GameDisplay" qui implémente la classe "Runnable". (voir fonctionnement figure 5)

La méthode comprend une boucle qui continue de tourner tant qu'il y a des boulets de canon ou des navires en mouvement dans le jeu. Cela consiste à faire dormir (sleep) notre programme pendant 50 millisecondes (ms) à chaque itération de la boucle, cela en rafraîchissant l'écran du jeu toutes les

50 ms. La méthode "moveBoat()" de la classe "Engine" est ensuite appelée après avoir déterminé si des bateaux sont actuellement en mouvement. Un bateau est supprimé de la liste des bateaux en mouvement lorsqu'il atteint sa destination, et sa position est alors fixée sur le bloc correct en invoquant la méthode "setBlockXY()" de la classe "Engine".

Si le bateau est un bateau commercial, la méthode "hasMarchandise()" de l'objet de type "BateauCommercial" est appelée pour vérifier s'il transporte des marchandises. Si c'est le cas, le système détermine alors si le bateau a atteint sa destination en comparant la position du bateau à celle des objets. Si le bateau a atteint sa destination, le solde du joueur est augmenté en appelant la méthode "addMoney()" de l'objet de type "Game", la marchandise est retirée en appelant la méthode "setMarchandise(null)" de l'objet de type "BateauCommercial" et le panneau de gauche de l'interface utilisateur ("gameGUI") est mis à jour en appelant la méthode "updateSelected()".

La méthode vérifiera ensuite s'il y a des boulets de canon en mouvement et les déplace en appelant la méthode "moveCannonball()" de "Engine". Ensuite, la méthode, appelée "repaint()", rafraîchit les graphismes du jeu.

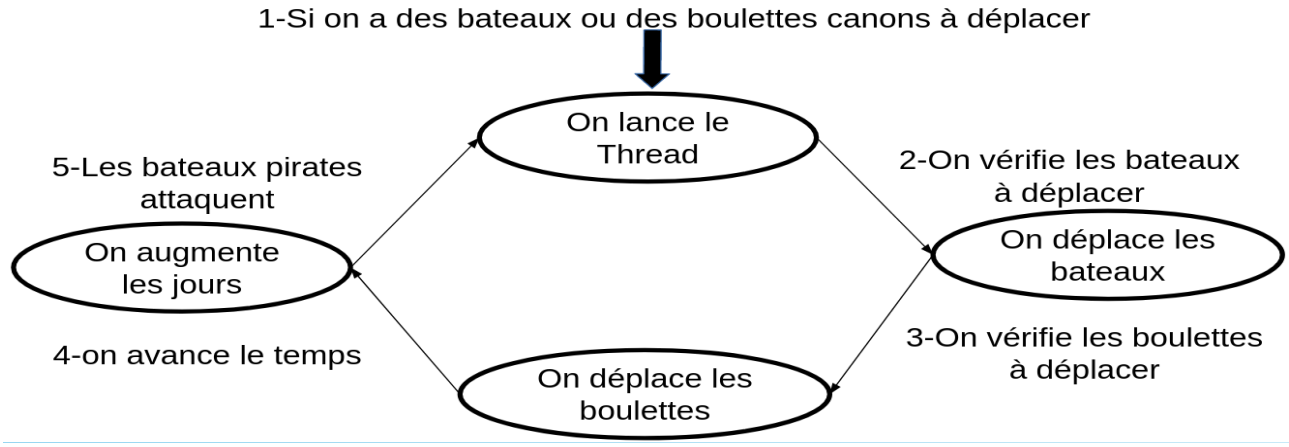



FIGURE 5 – Illustration du déroulement général du jeu

### 3.3.3 Méthodes intéressantes

Lors du développement de ce jeu, plusieurs méthodes ont été mises en œuvre pour garantir la qualité du produit final. Voici quelques-unes des méthodes les plus intéressantes :

| **Methodes** | **Action** |
|---|---|
| **public buildGame(int difficulté) : Game** | Créer une partie de jeu |
| **public findPath() : ArrayList<Block>** | Trouve le chemin de déplacement des bateaux |
| **public moveBoat() : void** | Déplace les bateaux |
| **public moveCannonBall() : void** | Lance et déplace les boulets cannons |
| **public neutralBoatAttack(Cannonball) : void** | Lance une attaque contre un bateau neutre |
| **public pirateAttack() : void** | Lance une attaque de bateau pirate |
| **public save() : void** | Sauvegarde l'état du jeu dans un fichier |
| **public readGame(String path) : Game** | Récupère une partie de jeu choisie |

## 3.4 Conception de l'IHM graphique

Dans cette section, nous allons vous expliquer la conception de l'IHM graphique avec des schémas abstraits qui illustrent l'organisation de différentes fenêtres (sous-fenêtres), ainsi que l'enchaînement qu'il y a entre elles.

Après avoir finalisé la partie conception de notre outil, le processus de la réalisation des interfaces graphiques a été entamée, en commençant par un prototype présenté dans la figure 6, réalisé à l'aide du logiciel Figma, en utilisant la charte graphique établie.


FIGURE 6 – Illustration de la conception IHM graphique de GameGUI

Une fois celui-ci finalisé, nous commencions la réalisation de l'IHM graphique à l'aide du package "javax.swing".

# 4 Déroulement du projet

Dans cette section, nous allons décrire comment le projet a été réalisé en équipe : la répartition des tâches et la synchronisation du travail en membres de l'équipe.

## 4.1 Réalisation du projet par étapes

Trois phases essentielles qui permettront de structurer la mise en œuvre de notre travail ont été établies :

### 4.1.1 Première phase : Recherche et documentation

Avant de se lancer dans la conception du projet, et pour une bonne prise de connaissance du sujet, des recherches approfondies sur notre thème, suivi par une analyse du cahier des charges, ont été effectuées. Pour la mise en œuvre du progiciel, nous nous sommes également documentés sur des outils techniques mis à notre disposition.

### 4.1.2 Deuxième phase : Conception

Sur la base de la recherche réalisée et la documentation collectée, la conception du projet a été entamée.

Durant cette phase nous avons décidé du paradigme de programmation le plus adéquat, nous avons donc opté pour la programmation orientée objet, car elle offre une meilleure structuration pour le code, ceci garantira la maintenance et l'évolutivité, tel que, grâce à l'indépendance des modules, la modification de l'un des constituants n'entraînera pas d'erreurs et n'affectera pas les autres constituants.

De plus, la structure de données a été définie, suivi par la réalisation d'un découpage modulaire permettant de créer un diagramme de classes qui détaillera la structure future de notre code.

### 4.1.3 Troisième phase : Réalisation

Cette phase a consisté à la concrétisation des éléments définis dans la phase de conception, en utilisant les outils techniques nécessaires et en commençant par la programmation, la création des fichiers de données, leur implémentations et enfin la mise en œuvre de l'interface graphique avec les changements adéquats, selon les cas.

## 4.2 Répartition des tâches entre membres de l'équipe

Notre projet a été mené en collaboration, en binôme, avec une approche de travail d'équipe et de coopération, où nous nous sommes tous deux impliqués de manière égale dans toutes les tâches nécessaires pour mener à bien le projet. Nous avons choisi de ne pas effectuer de répartition des tâches, car nous estimions que cela pourrait limiter notre flexibilité en matière de travail en équipe et créer des déséquilibres dans la charge de travail entre nous.

Nous avons travaillé en étroite collaboration tout au long du projet, en partageant les idées, les connaissances et les compétences de manière égale, et en nous assurant que toutes les tâches étaient effectuées de manière satisfaisante. Cette approche a permis une meilleure communication et une meilleure compréhension mutuelle entre nous, ce qui a conduit à une meilleure qualité du travail réalisé, ainsi qu'à un sentiment de satisfaction et d'accomplissement partagé.

## 4.3 Communication

Les membres de l'équipe ont tous des capacités et des connaissances différentes, et les forces de chaque membre constituent la force de l'équipe. Ainsi, chacun peut compléter les compétences de l'autre et partager les siennes avec lui.

Mais tout cela ne serait pas possible sans une bonne communication et nous en avions parfaitement conscience au sein de notre équipe. On a donc choisi les outils les plus adéquats afin de créer une transparence totale et une collaboration effective entre les membres. Mais ce n'est pas tout, les outils qu'on allait choisir devaient aussi assurer un bon team-building.

Nous effectuions donc régulièrement des réunions afin de vérifier l'avancement de chacun, de se concerter entre les membres et de s'assurer que le projet avance de la manière la plus sereine possible.

# 5 Présentation du jeu

## 5.1 Commencement du jeu

Le joueur commence au “Jour 1” dans un port neutre avec,
- en mode Facile : un bateau de marchandise Tier 1, un bateau d’attaque Tier 1 et 50 pièces
- en mode Normal : un bateau d’attaque Tier 1 et 30 pièces
- en mode Difficile : un bateau d’attaque Tier 1

Il peut directement se déplacer en mer et gagner de l’argent en trouvant des coffres, ou en s’attaquant à un navire de marchandise avec son bateau d’attaque.

## 5.2 Commerce et attaque

Lorsque le joueur a un bateau de marchandise, il peut acheter une marchandise et envoyer son bateau, seul ou avec une flotte, revendre la marchandise dans un autre port, cela selon sa capacité. Chaque port propose une tâche différente. Lorsque le bateau revient au port de départ avec l’argent récolté de la vente, l’argent vous revient directement dans votre solde.

Le bateau ou la flotte de bateaux envoyés peuvent se faire attaquer par des pirates en chemin (aller ou retour), vous aurez alors plusieurs possibilités :
- Attaquer : s’il n’y a pas de bateau d’attaque dans la flotte, la case est grisée et on ne peut pas cliquer dessus
- Céder : donner toutes les marchandises de la flotte aux pirates, ou toute l’argent s’il revenait du port
- S’enfuire : lancer un dé pour tenter de s’enfuir, chaque bateau devra lancer un dé et saisir sa chance (affichée à l’écran), cela en fonction de son Tier, cependant s’il rate il sera détruit.

## 5.3 Déplacements en mer

Les déplacements s’effectuent cases par cases sur une grille quadrillée (presque invisible). La distance de déplacement ainsi que le rayon de tirs varies tout deux selon le tier du bateau. Voici l’exemple dans la figure 7 suivante.

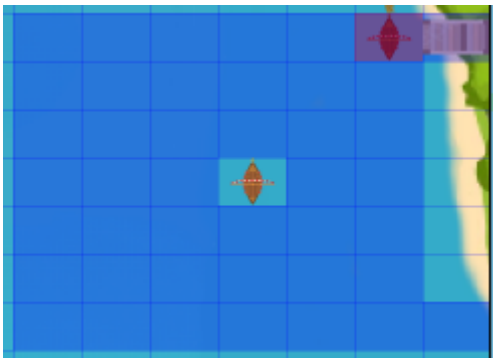


FIGURE 7 – Exemple de déplacement de notre bateau d’attaque tier 3

Dans cette position, notre navire tier 3 est bien plus avantagé pour attaquer le navire ennemi, qui est de tier 2.

## 5.4 Le système de bateaux

Le joueur possède un ensemble de bateaux représentant sa flotte, selon chaque niveau du bateau on a une capacité et un rayon de déplacement différents, ou même un rayon de tir pour les bateaux d'attaques. Ces caractéristique sont illustrés dans le tableau 1 suivant.

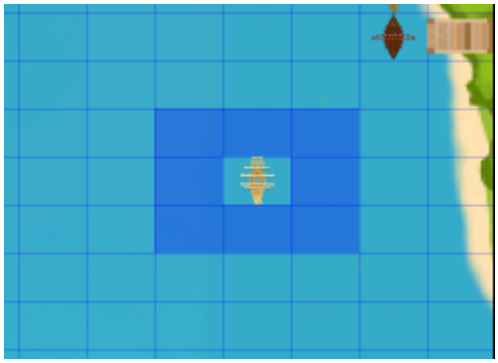
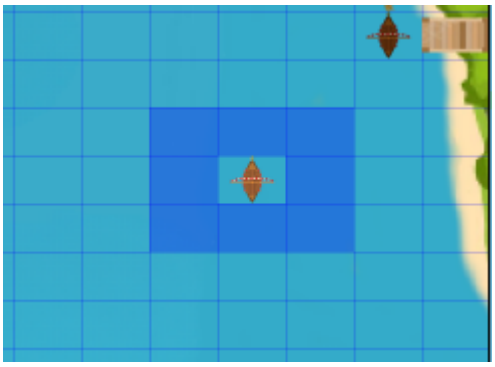
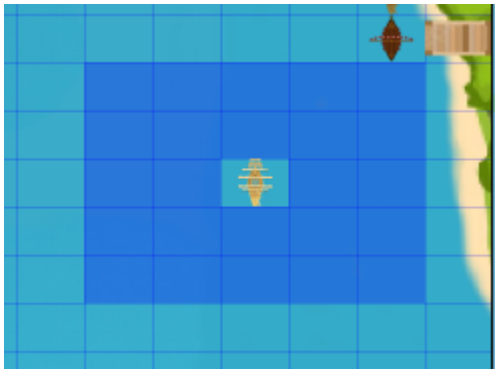
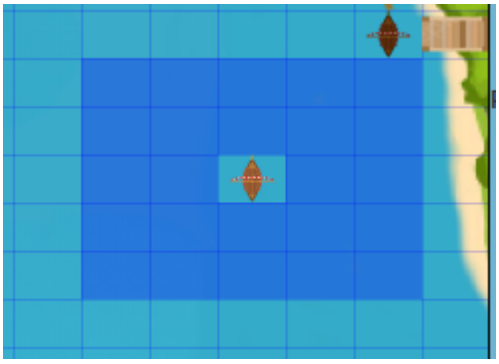
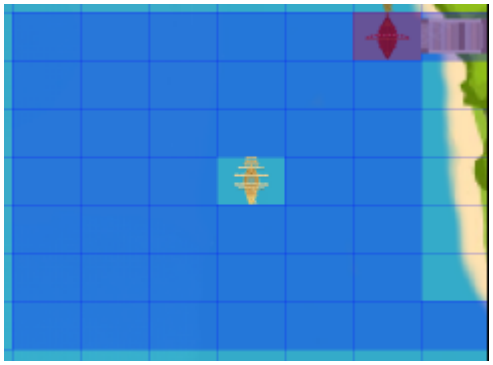
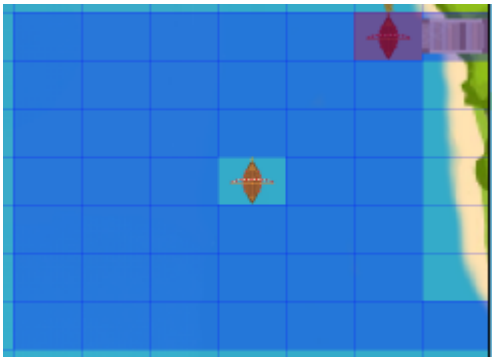
| | |
|---|---|
| Bateau de Marchandise Tier 1<br>  <br>Capacité : Tier 1<br>Résistance : 100HP | Bateau d'Attaque Tier 1<br>  <br>Résistance : 120HP |
| Bateau de Marchandise Tier 2<br>  <br>Capacité : Tier 2<br>Résistance : 150HP | Bateau d'Attaque Tier 2<br>  <br>Résistance : 170HP |
| Bateau de Marchandise Tier 3<br>  <br>Capacité : Tier 3<br>Résistance : 200HP | Bateau d'Attaque Tier 3<br>  <br>Résistance : 250HP |

TABLE 1 – Caractéristiques d'un bateau pour chaque niveaux

## 5.5 Conquérir la carte et gagner une partie

- Pour conquérir un pays, il faut posséder tous ses ports.
- Pour conquérir un port, il faut neutraliser tous ses bateaux puis payer une somme d'argent pour le "rénover" / "reconstruire" afin qu'il nous appartienne.
- Il y a un nombre de bateaux et une somme d'argent différents selon l'influence de celui-ci.

— Lorsque l'on possède un port, celui-ci nous rapporte de l'argent, selon son influence, et ceux toutes les minutes (exemple : 8 pièces/minutes), ainsi que d'autres avantages tels que le coût de marchandises et l'amélioration de ses navires moins chers.

# 6 Conclusion et perspectives

Dans cette section, nous vous résumerons la réalisation du projet puis nous vous présenterons les extensions et améliorations possibles du jeu.

## 6.1 Résumé du travail réalisé

Durant les semaines passées à travailler sur ce projet, nous avons su user de nos compétences et les joindre afin d'aboutir à un produit final qui satisfaisait les objectifs fixés dans notre cahier des charges et ce grâce à une motivation et une détermination sans pareille ainsi qu'une excellente communication qui a fait que nous avons pu atteindre la plupart de nos objectifs en temps et en heure.

Nous avons fait face à la contrainte du temps ainsi qu'à des contraintes techniques mais nous avons très rapidement pu s'en défaire grâce à la collaboration entre nous et à l'esprit de partage qui s'est instauré au sein de notre binôme. Ainsi, à l'issue de ce projet nous avons réalisé un jeu qui répond à la problématique posée.

Pour conclure, ce projet nous a permis de vivre une immersion modeste dans la vie d'un ingénieur logiciel ce qui nous a permis d'étendre nos perspectives et connaissances ainsi que de nous trouver diverses nouvelles passions concernant le développement de solutions informatiques, découvrant où est-ce qu'on est doué et où est-ce qu'on l'est un peu moins.

## 6.2 Améliorations possibles du projet

Plusieurs idées nous sont venues au cours de la réalisation de ce jeu afin de le rendre mieux équilibré, plus attrayant et maintenir l'intérêt des joueurs à long terme. Parmi ces idées, nous retiendrons les améliorations suivantes, lesquelles pourraient être ajoutées dans des mises à jour futures (dans l'ordre chronologique d'ajouts) :

- Le changement d'apparition de la scène de combat "zoomée" (qui est à part de la carte), qui sera désormais constament présente lorsqu'un combat aura lieu et non uniquement lorsque l'on riposte face à une attaque de pirate. Une fois dans cette scène de combat lorsque le joueur essaiera de fuir, s'il échoue, alors son bateau recevra un boulet de cannon et des dégats multipliés par 2, puis il pourra réessayer jusqu'à ce qu'il s'enfuit ou qu'il perde son bateau explosé. Également, si le combat s'intensifie de son côté, qu'il pense pouvoir perdre et qu'il le juge donc nécéssaire, il pourra donc essayer de s'enfuir de la même manière et risquer de se perdre des dégats doublés.
- La possibilité pour un navire ennemi de se déplacer en combat (dans la scène de combat zoomé), et donc de pouvoir se défendre face à un de nos navire qui serait d'un tier supérieur et qu'il n'aurait pas pu atteindre, il pourrait ainsi s'avancer jusqu'à être à portée de tir.
- La possibilité de pouvoir réparer son bateau dans un port.
- La possibilité de pouvoir améliorer son bateau (ex : Tier 2 -> Tier) dans son port.

## Références

[Jai11] Sonoo Jaiswal. javax.swing (java platform se 8). https://docs.oracle.com/javase/8/docs/api/index.html?javax/swing/package-summary.html, 2011. Visité le : 2 Février 2023.

[Ope21] OpenAI. Dall-e. https://openai.com/product/dall-e-2, 5 janvier 2021. Visité en : Mars 2023.

[Tha20] Nguyen Nam Thai. Baeldung. https://www.baeldung.com/, 2020. Visité en : Janvier 2023.