

**Université de Cergy-Pontoise**

**RAPPORT**

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

**Projet : Park**

par

**Steven BASKARA Julien RAAD Thomas HORNUNG**



Mars 2024

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Contexte du projet . . . . .	2
1.2	Objectif du projet . . . . .	2
<b>2</b>	<b>Spécification du projet</b>	<b>3</b>
2.1	Notions de base et contraintes du projet . . . . .	3
2.1.1	Fonctionnement général du logiciel . . . . .	3
2.1.2	Notions et terminologies de base . . . . .	3
2.2	Fonctionnalités attendues du projet . . . . .	4
<b>3</b>	<b>Conception et réalisation du projet</b>	<b>5</b>
3.1	Architecture globale du logiciel . . . . .	5
3.2	Conception des classes de données . . . . .	5
3.3	Conception des traitements (processus) . . . . .	6
3.4	Conception de l'IHM graphique . . . . .	6
<b>4</b>	<b>Manuel utilisateur</b>	<b>8</b>
<b>5</b>	<b>Déroulement</b>	<b>12</b>
5.1	Architecture globale du logiciel . . . . .	12
5.2	Réalisation du projet par étapes . . . . .	12
5.3	Répartition des tâches entre membres de l'équipe . . . . .	12
<b>6</b>	<b>Conclusion et perspectives</b>	<b>13</b>
6.1	Résumé du travail réalisé . . . . .	13
6.2	Améliorations possibles du projet . . . . .	13

## Table des figures

1	Diagramme de classes de données . . . . .	5
2	Noyau fonctionnel du programme . . . . .	6
3	Schéma abstrait de la fenêtre d'entrée de l'IHM . . . . .	7
4	Schéma abstrait des fenêtres principales de l'IHM . . . . .	7
5	Image de la fenêtre d'entrée de l'IHM . . . . .	8
6	Capture d'écran d'une partie en cours. On a un menu général nous permettant d'accéder à des actions spécifiques de placement de structures. . . . .	9
7	Capture d'écran du système d'achat du jeu . . . . .	10
8	Capture d'écran de la fenêtre principale de l'IHM . . . . .	11

# **1 Introduction**

Dans cette section, nous allons présenter brièvement l'objectif du projet

## **1.1 Contexte du projet**

Nous avons entrepris ce projet dans le cadre de notre cours de Génie Logiciel. En tant qu'amateurs de jeux vidéo, notre concept de parc s'inspire fortement des jeux de gestion, notamment des "tycoons" tels que RollerCoaster Tycoon. Le choix du thème du parc était l'une de nos premières décisions.

## **1.2 Objectif du projet**

Notre objectif est de concevoir un jeu de gestion de parc d'attractions. L'utilisateur devra générer des revenus pour créer et améliorer son parc en achetant de nouvelles structures et en répondant à la demande des clients. Le jeu sera développé en Java et récompensera une gestion efficace du parc par la génération accrue de revenus, constituant ainsi la base de la progression. De plus, le jeu comportera un objectif final, ce qui permettra aux joueurs de le "terminer".

## 2 Spécification du projet

Nous avons présenté l'objectif du projet dans la section 1. Dans cette section, nous présentons la spécification de notre logiciel réalisé. Ceci 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

**Jeu de Gestion :** Le jeu vidéo de gestion consiste à placer un joueur en charge d'une organisation ou d'un commerce avec des ressources limitées, et lui donne le pouvoir d'utiliser ces ressources pour affecter son fonctionnement et ainsi obtenir plus de ressources. Ici, nous réalisons un jeu de gestion de parc : la ressource a gérée est donc avant tout l'argent, et la progression doit se faire avant tout par la construction de structures pour développer le parc.

**Temps Réel :** Le jeu avancera en temps réel, les processus du jeu utiliseront un intervalle de temps régulier comme unité de temps de base pour réaliser des opérations en respectant des contraintes de temps. Dans la plupart des cas, le temps s'écoulera à vitesse normale pendant que le joueur joue, cela s'oppose aux systèmes de tour par tour. Cet intervalle commencera à 1 seconde, mais pourrait être ajusté si le programme demande plus ou moins de ressources informatiques que prévu. Le temps sera également une ressource pour le joueur, qui devra l'utiliser à bon escient pour générer de l'argent et résoudre les problèmes de son parc.

**Carte Quadrillé :** La carte quadrillée constitue la base du terrain du parc. Chaque case de la grille représente une zone du parc où le joueur peut placer des structures, créer des chemins, et personnaliser l'environnement. Les structures peuvent occuper plusieurs cases, mais une case ne peut contenir plus d'une structure, et est définie par la structure qui l'occupe.

**Point de vue externe omniscient :** Le joueur n'incarne pas un personnage dans le parc, à la place il est placé à un point de vue extérieur au parc, en ayant la possibilité de le contrôler à sa guise, de le voir à n'importe quel moment dans son entièreté, et de récupérer des informations sur certaines parties spécifiques du parc. L'interface graphique sera adaptée à ce point de vue, offrant une vision d'en haut d'une carte en 2d.

#### 2.1.2 Notions et terminologies de base

**Argent :** L'argent est la ressource principale du jeu, et également l'objectif principal. Il est représenté par une valeur à l'intérieur du code, visible à tout moment au joueur, qui évolue selon l'état du jeu et les actions du joueur. Il forme le moteur de la progression du jeu, permettant au joueur de réaliser les actions qui constituent le jeu tout en étant quelque chose à obtenir par ces mêmes actions.

**Structure :** Les structures sont les objets pouvant être placés par le joueur pour faire progresser sa partie. Elles sont l'essence du jeu de gestion de parc, et seront présentes sous différentes formes en héritant d'une classe Structure générale. Elles auront diverses fonctionnalités selon leur forme, mais permettent globalement de répondre aux problèmes imposés par le jeu et de gagner de l'argent.

**Besoins et demandes :** Pour indiquer au joueur le genre de structure qui devront être placés à un moment ou un autre, des informations sur les besoins du parc et les demandes des clients, générées de manière partiellement aléatoire, seront communiquées. Elles permettront au joueur de faire de bonnes ou de mauvaises décisions et ainsi permettre à l'essence du jeu d'émerger.

Outils de développement :

1. Java
2. Eclipse
3. Latex
4. Visual Studio Code

## 2.2 Fonctionnalités attendues du projet

**Fonctionnalités** Fonctionnalités du programme :

- **Générer une partie** : Le joueur crée ainsi une carte vierge, ou comportant seulement les éléments de base du parc. C'est à partir de cela que l'utilisateur va pouvoir réaliser les différentes actions proposées par le jeu.
- **Agrandir l'espace disponible** : Agrandir son parc en achetant de nouvelles parcelles de terrain, en offrant ainsi plus d'espace pour de nouvelles attractions et installations. L'utilisateur commence avec un terrain plutôt restreint mais suffisant pour un début de partie. Chaque nouvelle portion de terrain acheté coûtera une importante somme d'argent, et cette somme augmentera pour chaque autre portion de terrain déjà acheté. La carte aura une taille maximale prédéfinie, et donc un nombre maximum de portion de terrain pouvant être acheté.
- **Acheter et placer des structures** : Intégrer un système permettant au joueur d'acheter des manèges, boutiques, et stands alimentaires, et de les placer stratégiquement dans son parc. Elles pourront si nécessaire être détruites ou déplacées. Elles occupent une ou plusieurs cases de la carte quadrillée. Ces structures sont la base de la construction du jeu, ce sont surtout elles qui vont coûter et rapporter de l'argent.
- **Placer des chemin** : Créer des chemins permettant de connecter différentes zones du parc (ex : attractions) et ainsi de permettre la navigation des visiteurs à ces zones. Toute structure doit être reliée aux autres par un chemin pour interagir avec elles.
- **Système de temps** : Affichage de la progression du temps avec des cycles jour et de nuit et une date qui s'actualise automatiquement après 24 heures . Une minute dans la vie réelle correspond à une heure dans le jeu. Il aura aussi un contrôle sur le temps, en ayant la possibilité d'arrêter et reprendre le temps mais aussi de l'accélérer.
- **Gestion financière** : Visualisation en temps réel de son total d'argent disponible, ainsi que la valeur de son parc (le total de la valeur des structures construites). Ces valeurs peuvent être négatives dans certaines circonstances.
- **Personnages représentant des visiteurs** : Avoir des personnages virtuels représentant les visiteurs du parc, avec leur nombre augmentant selon le nombre de visiteurs présent dans le parc. Les personnages virtuels ne sont pas chacun modélisés individuellement, mais sont une représentation visuelle de la statistique indiquant le nombre de visiteurs dans le parc.
- **Sauvegarde de la partie** : Effectuer des sauvegardes de sa partie, lui permettant de reprendre là où il s'était arrêté lors de sessions ultérieures. L'utilisateur pourra sélectionner l'option Sauvegarder à tout moment pour stocker localement les informations de sa partie. Il sera possible d'implémenter une fonction de sauvegarde automatique.

### 3 Conception et réalisation du projet

#### 3.1 Architecture globale du logiciel

L'architecture du projet est organisée en plusieurs modules principaux, chacun se trouvant dans le répertoire `src` du projet. Voici la fonction de chaque module :

- **config** : Ce module contient toutes les configurations utilisées dans le jeu, telles que les paramètres de dimensionnement des éléments graphiques et la vitesse de simulation.
- **data** : Gère les structures de données essentielles du jeu, incluant la gestion des dates et des statistiques actives des parcs.
- **engine** : Ce module encapsule toute la logique du jeu, responsable de la gestion des cartes, des entités mobiles telles que les visiteurs et les curseurs, ainsi que de la logique de progression du jeu.
- **gui** : Dédié à l'interface utilisateur, ce module gère la présentation visuelle du jeu et l'interaction avec l'utilisateur à travers divers composants graphiques.
- **images** : Contient toutes les images et sprites utilisés dans le jeu.
- **test** : Inclut des tests destinés à vérifier la fiabilité et la performance des différentes composantes du jeu.

#### 3.2 Conception des classes de données

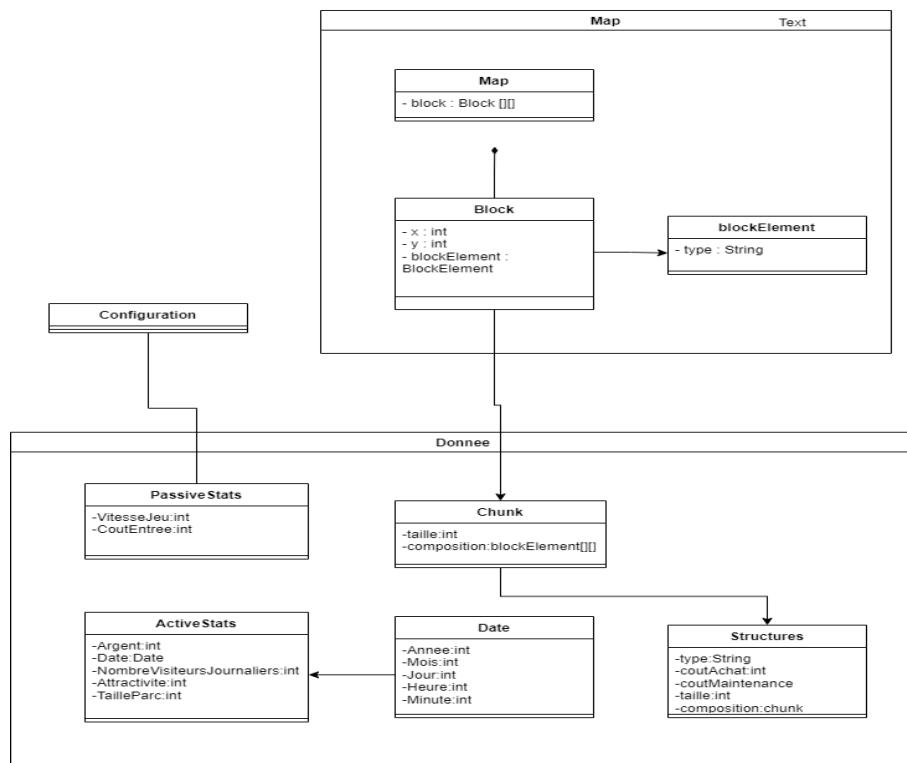


FIGURE 1 – Diagramme de classes de données

### 3.3 Conception des traitements (processus)

Dans la figure 2, on peut voir un schéma représentant le noyau fonctionnel de notre programme.

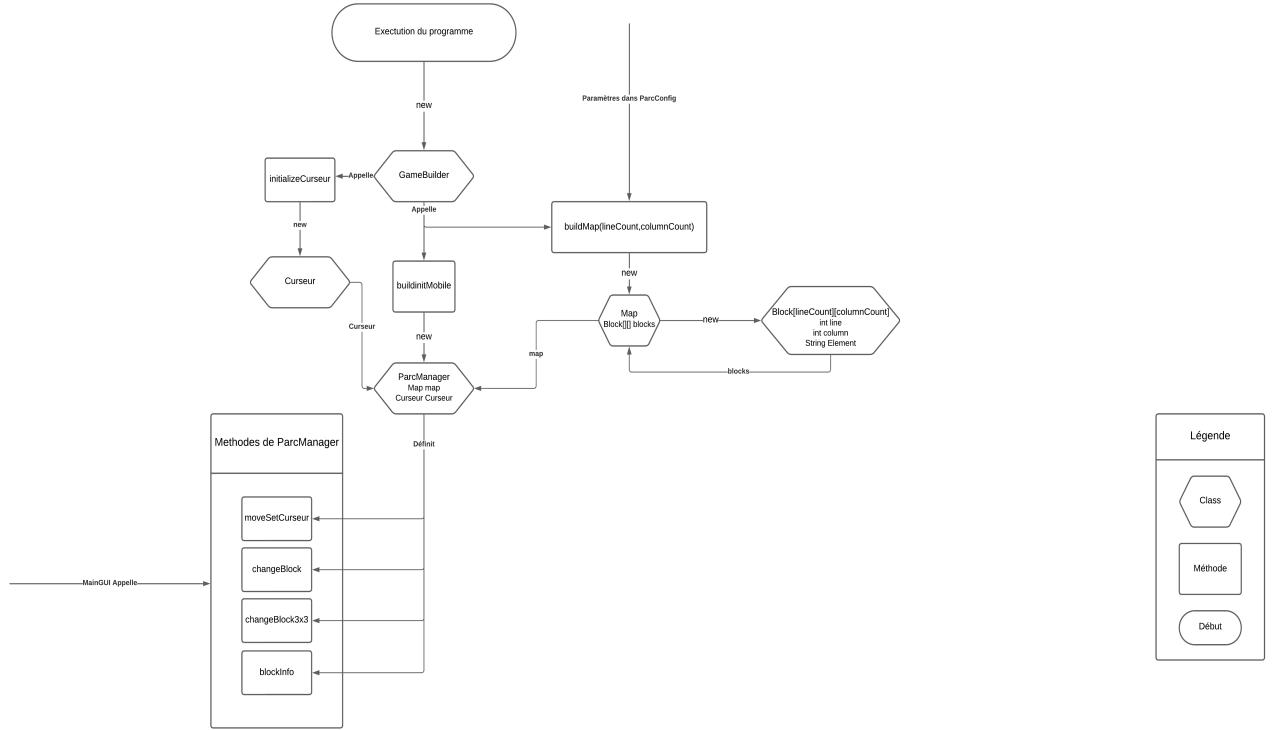


FIGURE 2 – Noyau fonctionnel du programme

### 3.4 Conception de l'IHM graphique

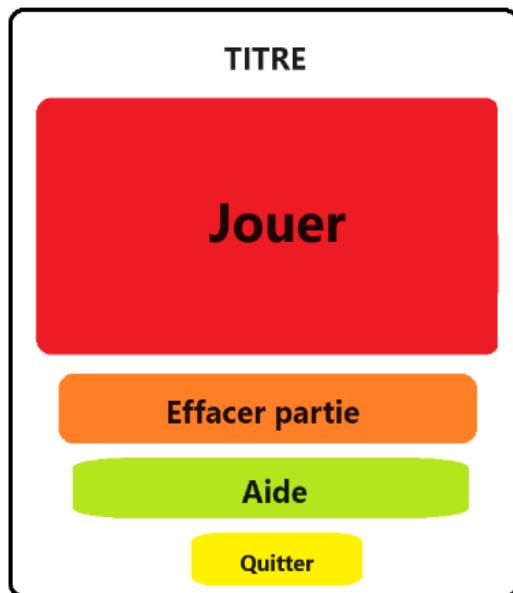


FIGURE 3 – Schéma abstrait de la fenêtre d'entrée de l'IHM

On a une interface avec 4 boutons une pour chaque action : - Jouer : Lancer la partie  
 - Effacer partie : Supprime la sauvegarde - Aide : Affiche les consigne du jeu - Quitter : Ferme l'interface

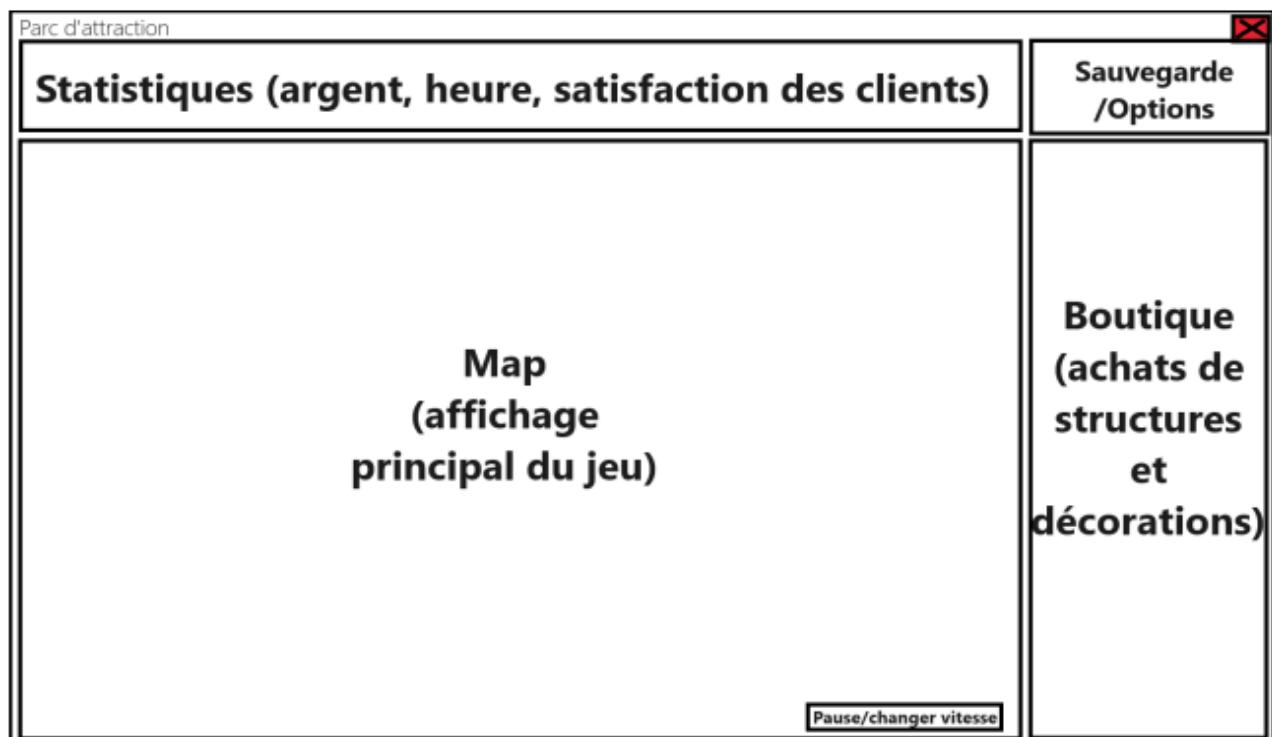


FIGURE 4 – Schéma abstrait des fenêtres principales de l'IHM

## 4 Manuel utilisateur

Cette section est dédiée au manuel utilisateur. C'est la seule section dans laquelle vous pouvez utiliser des captures d'écran de votre logiciel.

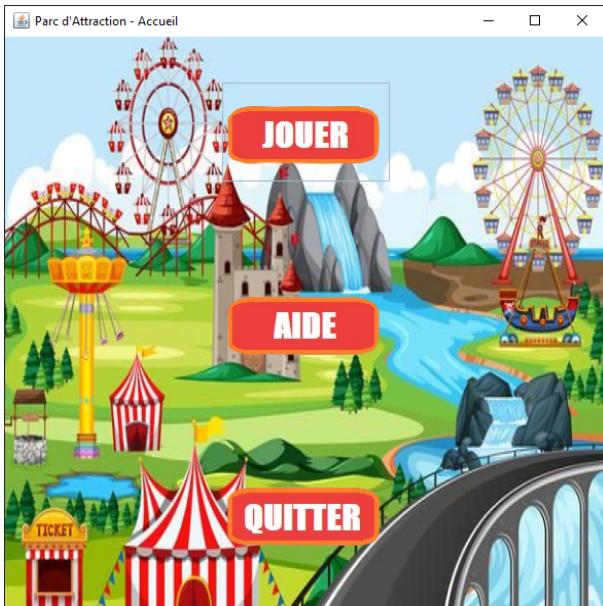


FIGURE 5 – Image de la fenêtre d'entrée de l'IHM

On a une interface avec 3 boutons une pour chaque action : - Jouer : Lancer la partie  
- Aide : Affiche les consigne du jeu - Quitter : Ferme l'interface

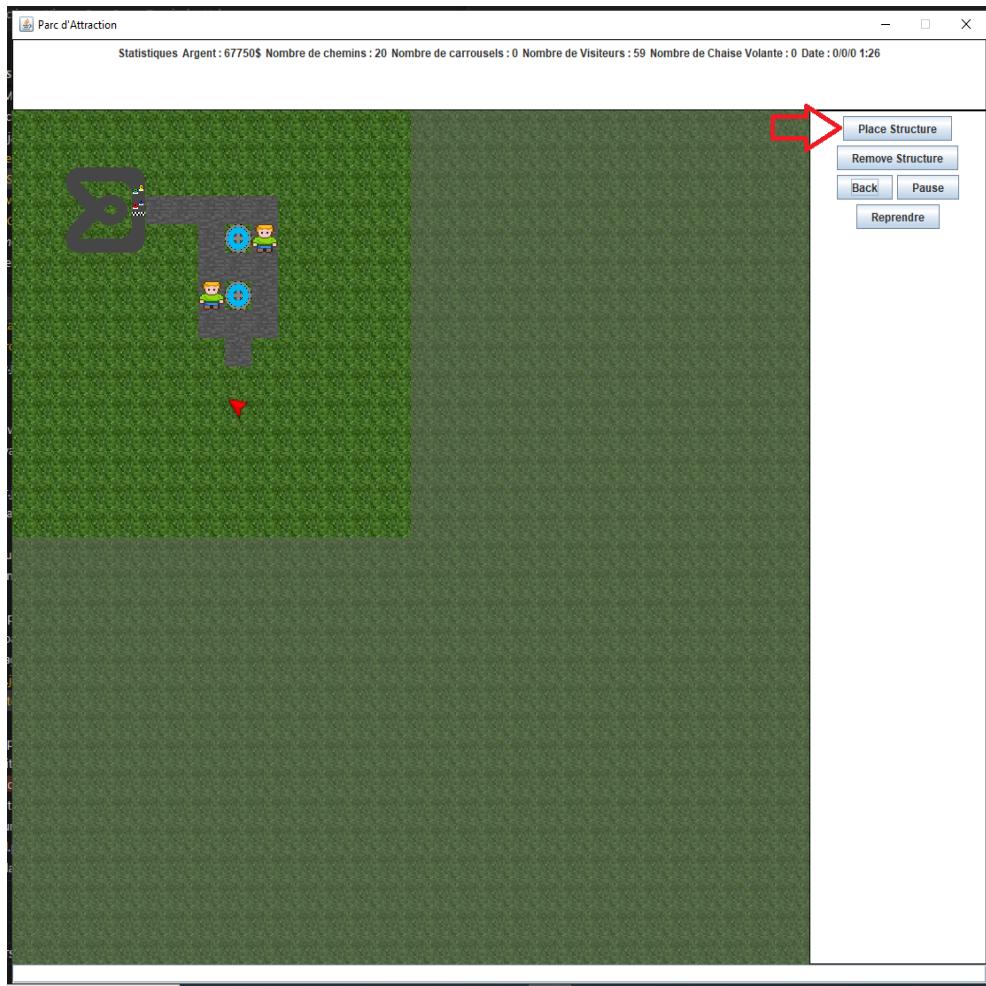


FIGURE 6 – Capture d'écran d'une partie en cours. On a un menu général nous permettant d'accéder à des actions spécifiques de placement de structures.

On a le bouton "Placer Structures" permettant de voir toutes les structure qu'on peut placer sur la carte

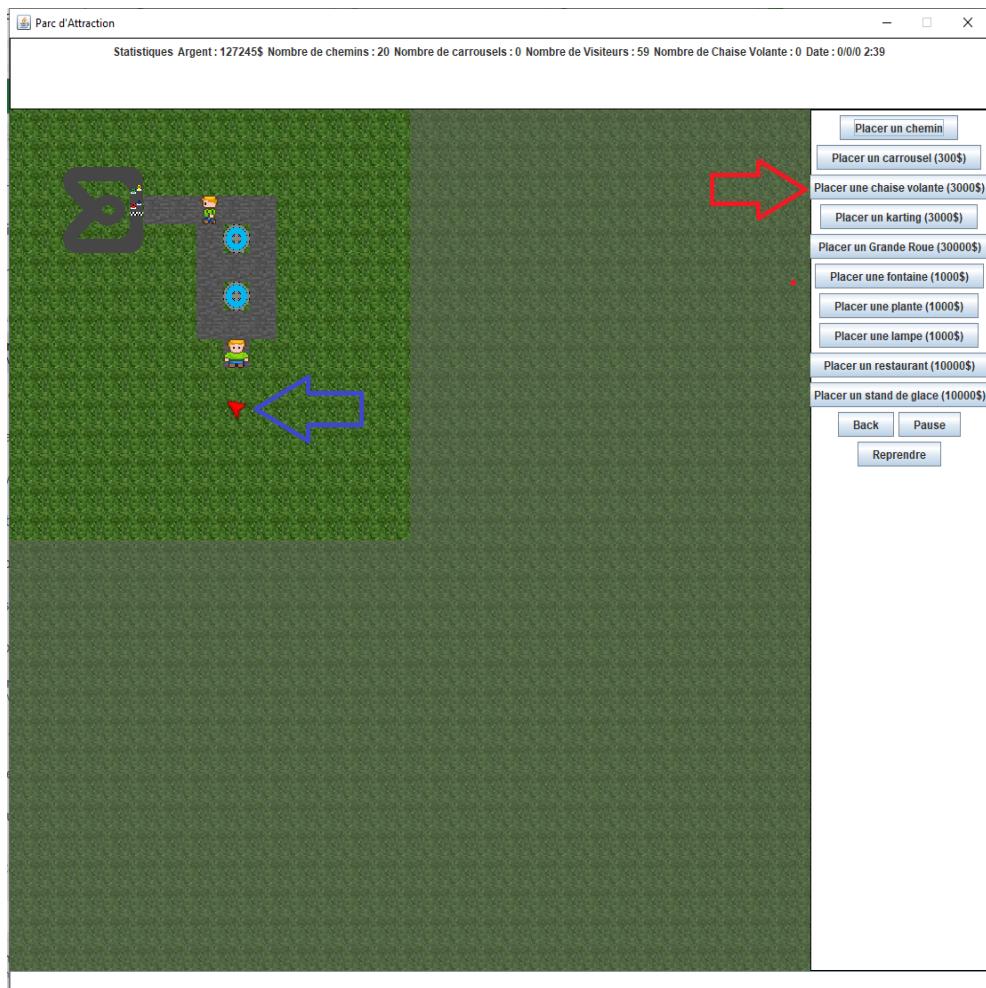


FIGURE 7 – Capture d’écran du système d’achat du jeu

Voici un exemple d’utilisation de cette interface : l’utilisateur positionne son curseur sur un emplacement spécifique de la carte, puis clique sur le bouton ‘Placer Chaise Volante’. Cette action entraîne l’installation du manège à l’endroit sélectionné et la déduction correspondante du montant de son solde.

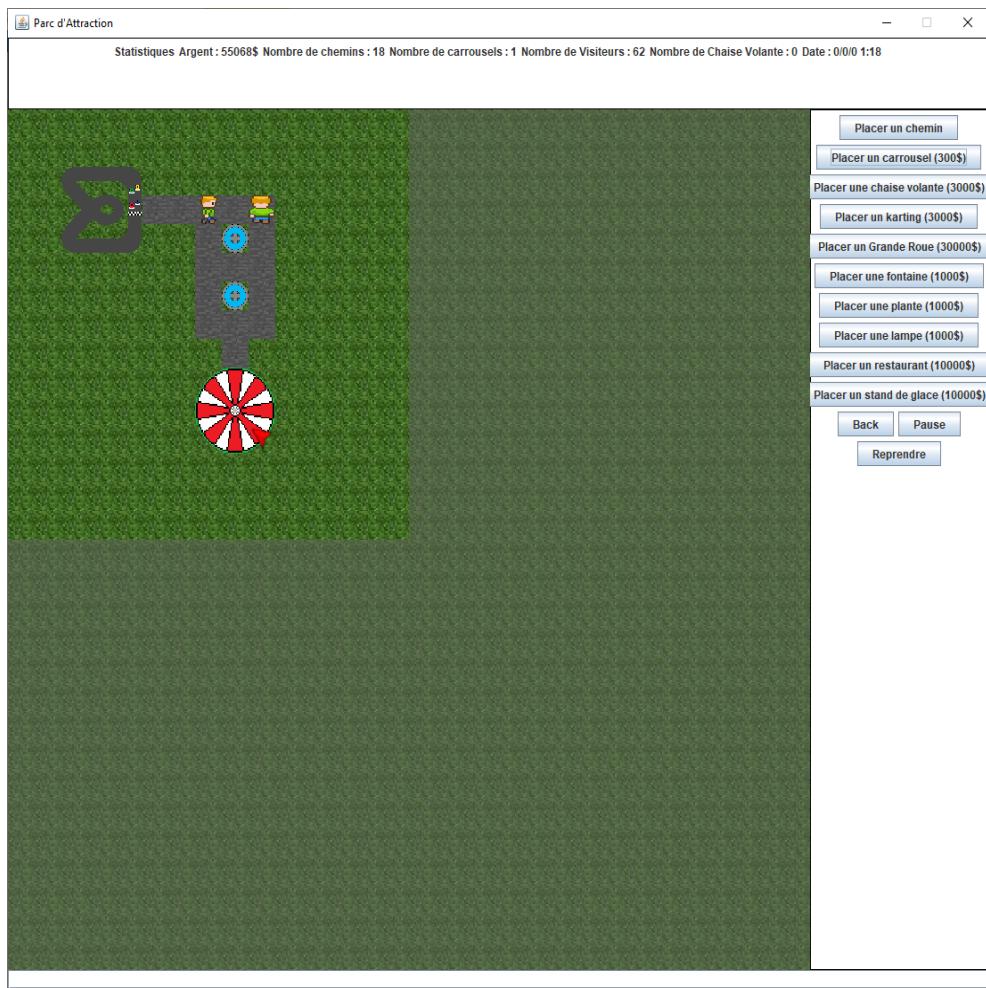


FIGURE 8 – Capture d'écran de la fenêtre principale de l'IHM

Voici le résultat de l'action précédente. Le Carrousel est placé.

## 5 Déroulement

### 5.1 Architecture globale du logiciel

L'architecture du projet est organisée en plusieurs modules principaux, chacun se trouvant au sein du répertoire `src` du projet. Voici la fonction de chaque module :

- **config** : Ce module contient toutes les configurations utilisées dans le jeu, comme les paramètres de dimensionnement des éléments graphiques et de vitesse de simulation.
- **data** : Gère les structures de données essentielles pour le jeu, incluant la gestion des dates et des statistiques actives des parcs.
- **engine** : Ce module contient toutes la logique du jeu, responsable de la gestion des cartes, des entités mobiles telles que les visiteurs et les curseurs, ainsi que de la logique de déroulement du jeu.
- **gui** : Dedié à l'interface utilisateur, ce module gère la présentation visuelle du jeu et l'interaction avec l'utilisateur à travers différents composants graphiques.
- **images** : Contient toutes les images et sprites utilisés pour le jeu.
- **test** : Inclut des tests destinés à vérifier la fiabilité et la performance des différentes composantes du jeu.

### 5.2 Réalisation du projet par étapes

Le développement du projet s'est articulé en plusieurs étapes clés, débutant par la conception de l'architecture du système, suivie par le développement des différentes fonctionnalités, et enfin, la phase de test et d'optimisation. Chaque étape a été réalisée avec des délais précis pour assurer la progression régulière du projet.

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

La répartition des tâches a été conçue pour optimiser les compétences individuelles des membres de l'équipe, tout en assurant une distribution équilibrée du travail. Voici le détail de la contribution de chaque membre :

- **Thomas Hornung** : Thomas a été chargé de développer le système de génération d'argent. Cette tâche incluait la conception des algorithmes qui simulent les flux financiers du parc en fonction des attractions et des visiteurs, ainsi que l'intégration de ce système avec les autres modules du jeu pour assurer une gestion fluide des ressources financières du parc.
- **Steven Baskara** : Steven a travaillé sur le système de déplacement des visiteurs. Son rôle était de mettre en place des algorithmes pour le mouvement autonome des visiteurs dans le parc, en s'assurant que les interactions avec les attractions soient logiques et enrichissantes. Steven a également intégré des éléments de l'intelligence artificielle pour que les comportements des visiteurs soient réalistes et prévisibles.
- **Julien Raad** : Julien a été responsable des graphismes et des sprites du jeu. Cela comprenait la conception graphique des interfaces utilisateur, des sprites pour les attractions et les visiteurs, ainsi que de l'ensemble de l'aspect visuel du jeu. Sa contribution a été cruciale pour garantir que le jeu soit visuellement attrayant et engageant pour les utilisateurs.

## 6 Conclusion et perspectives

Dans cette section, nous résumons la réalisation du projet et nous présentons également les extensions et améliorations possibles du projet.

### 6.1 Résumé du travail réalisé

Dans le cadre de ce projet, nous avons conçu et développé un jeu de simulation où l'utilisateur gère un parc d'attractions. L'objectif principal était de créer une application interactive permettant aux utilisateurs de construire et d'optimiser leur parc, en prenant en compte divers facteurs tels que les finances, les envies des visiteurs, et l'expansion du parc.

**Développements et Implémentations :** Nous avons implémenté plusieurs fonctionnalités clés, y compris :

- **Interface utilisateur graphique (GUI) :** Une interface conviviale permettant aux utilisateurs de naviguer facilement dans le jeu, de placer des attractions et de visualiser les statistiques du parc.
- **Système de gestion financière :** Un module pour gérer les finances du parc, incluant les coûts des attractions, les revenus, et les dépenses opérationnelles.
- **Simulation de visiteurs :** Un algorithme pour simuler l'arrivée et les mouvements des visiteurs dans le parc, ainsi que leur interaction avec les différentes attractions.

### 6.2 Améliorations possibles du projet

Plusieurs améliorations sont envisageables. Ces améliorations visent à enrichir l'expérience utilisateur.

**Idée d'amélioration :**

- **Ajout de nouvelles fonctionnalités :** Intégrer des éléments tels que des événements spéciaux saisonniers qui peuvent attirer plus de visiteurs et augmenter les revenus de manière temporaire.

**Améliorations de l'Interface Utilisateur :** Adapter l'interface pour qu'elle soit responsive et assure une expérience utilisateur fluide sur tous les dispositifs .

**Optimisations Techniques :**

- **Performance :** Améliorer les algorithmes de simulation pour accélérer les temps de réponse et augmenter l'efficacité de la gestion des ressources du jeu.