Document de rendu du projet ACOL

Rédigé par Chilou Paul, Meng-Yang Hubert GUO, Point Antonin et Sadoun Titouan.

Table des Matières

Bonne Lecture !

1) Cahier des Charges

Pour répondre aux besoins de notre client, nous avons identifiédifférentes fonctionnalités nécessaires au fonctionnement de notre plateforme.

Rôle du système :

1. Connexion / Déconnexion.
2. Quitter application.
3. Switcher d'un mode à l'autre (achat vente, visu portefeuille).
4. Afficher l'historique des transactions.
5. Avancer d'un jour dans la simulation.
6. Afficher la date à laquelle en est la simulation.
7. Afficher la valeur actuelle du portefeuille.
8. Afficher la valeur associée à une action dans le portefeuille.
9. Afficher la courbe de la valeur d'une action aux cours du temps
10. Permettre le dépot / retrait de l'argent du capital investi.
11. Permettre d'acheter / vendre des actions.

12. Afficher le bilan utilisateur à la fin du jeu.

Environnement système

1. Interaction du client avec l'interface graphique sur sa machine personnelle

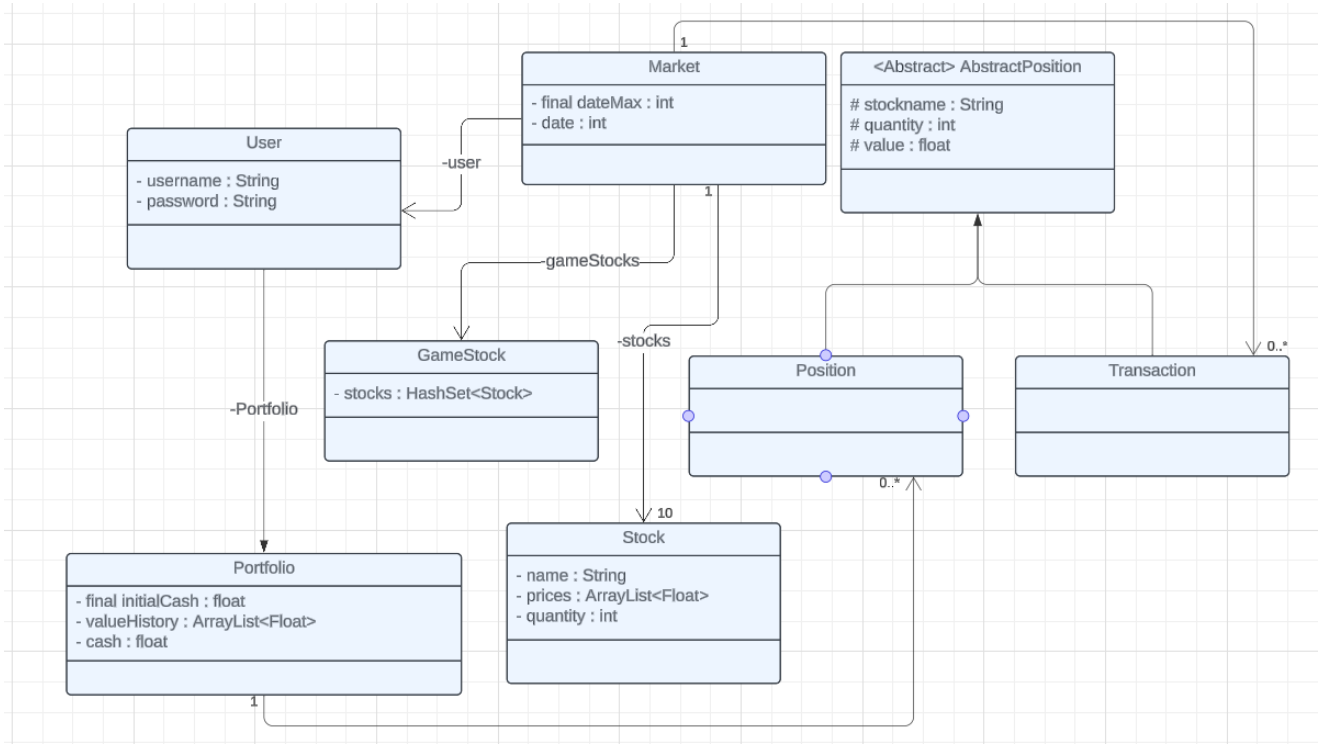
Besoin non fonctionnels

-> Onglet de demande de confirmation de fermeture
-> Message d'erreur si mauvaises informations rentrées

2) Analyse

1) Diagramme de classe d'analyse

Voici les différentes classes que nous avons analysées pour répondre aux besoins du client :



2) Cas d'utilisation

Voici différents cas liés à l'utilisation de notre plateforme de trading.

1) Connexion

1 . Le système demande à l'utilisateur ses identifiants
1.1 . L'utilisateur rentre ses identifiants (clique sur la case Identifiant et rentre son identifiant puis sur la case mot de passe et rentre son mot de passe dans un ordre quelconque et autant de fois que souhaité) et appuie sur "valider".
1.2 . Le système vérifie que les identifiants rentrés en paramètre sont corrects :
1.2.1 . Si les 2 sont corrects, accès à son tableau de bord correspondant.
1.2.2 . Si l'un des identifiants est incorrect (clé (id,mdp) non reconnue), le système affiche un message d'erreur et l'utilisateur reste sur la page de connexion.

Diagramme de séquence système pour une connexion réussie

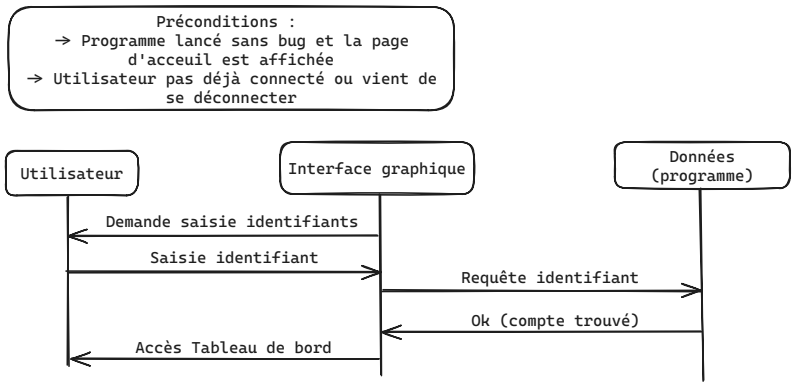
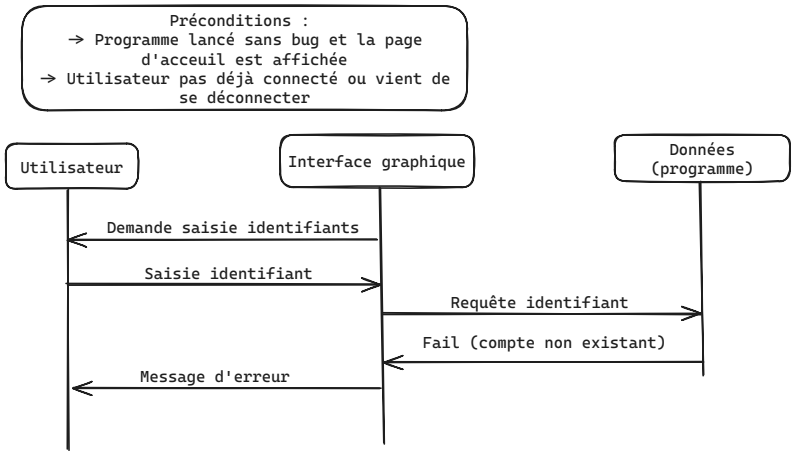


Diagramme de séquence système pour une connexion ratée

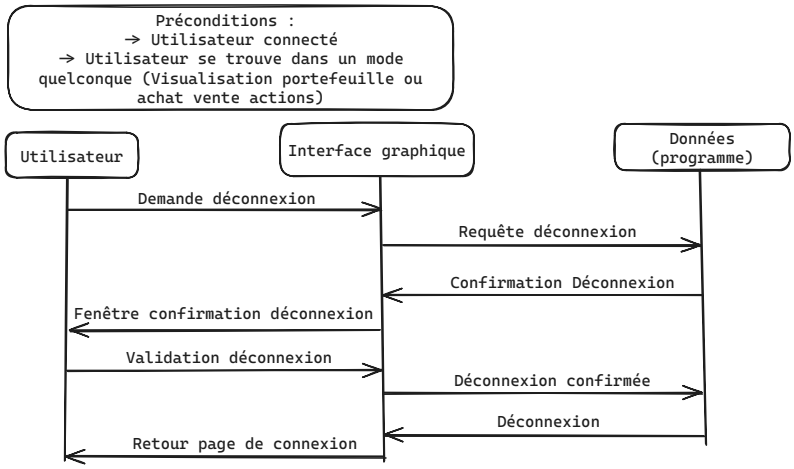


2) Déconnexion

1. L'utilisateur clique sur le bouton "déconnexion" (quelque soit le mode dans lequel il se trouve).
2. Le système envoie un message de confirmation de déconnexion.
3. L'utilisateur clique sur "ok".

4. Le système déconnecte l'utilisateur et le renvoie à la page d'accueil à l'étape de connexion

Diagramme de séquence pour une déconnexion utilisateur



3) Mode Achat / Vente Action (AVA)

1) Achat d'action

1. L'utilisateur entre un entier positif dans le champ quantité
2. L'utilisateur sélectionne un titre
3. L'utilisateur clique sur le bouton "Acheter"
    1. La quantité renseignée est un entier inférieure à la quantité disponible
        1. Le programme modifie le contenu du portefeuille et la quantité d'actions disponibles
        2. La quantité disponible affichée est actualisée
    2. La quantité renseignée n'est pas au bon format :
        1. Le programme n'effectue aucune transaction
        2. Une fenêtre avec un message d'erreur est affichée
            1. L'utilisateur clique sur le bouton "Ok"
            2. L'utilisateur clique sur la croix en haut de la fenêtre
        3. La fenêtre pop-up est fermée

Diagramme de séquence pour un achat réussi

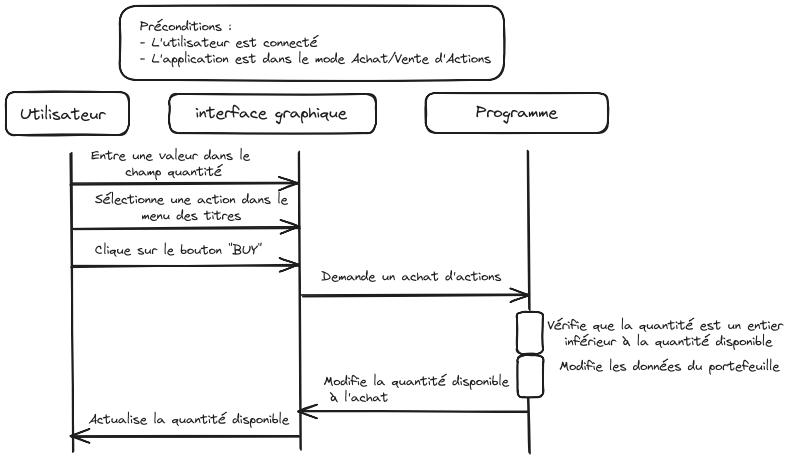
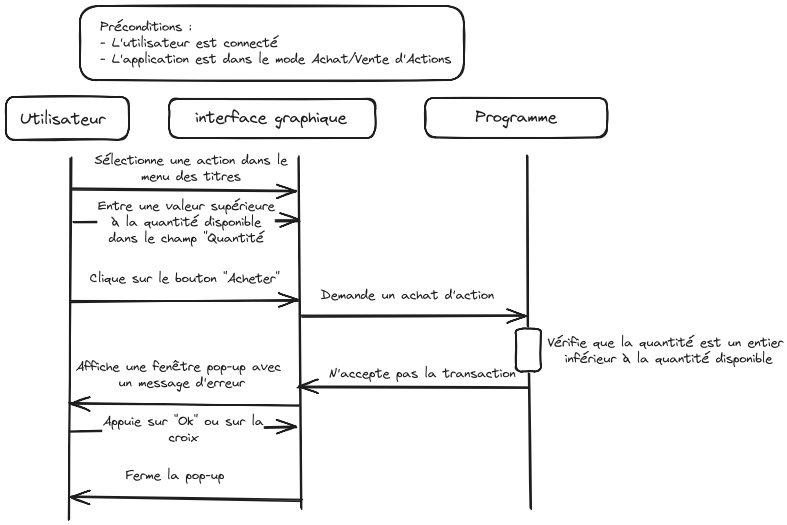


Diagramme de séquence pour un échec d'achat (quantité disponible insuffisante)



2) Vente d'action

1. L'utilisateur entre un entier positif dans le champ quantité
2. L'utilisateur sélectionne un titre
3. L'utilisateur clique sur le bouton "Vendre"
    1. La quantité renseignée est un entier positif
        1. Le programme modifie le contenu du portefeuille et la quantité d'actions disponibles
        2. La quantité disponible affichée est actualisée
    2. La quantité renseignée n'est pas un entier
        1. Le programme n'effectue aucune transaction

Diagramme de séquence pour un achat réussi

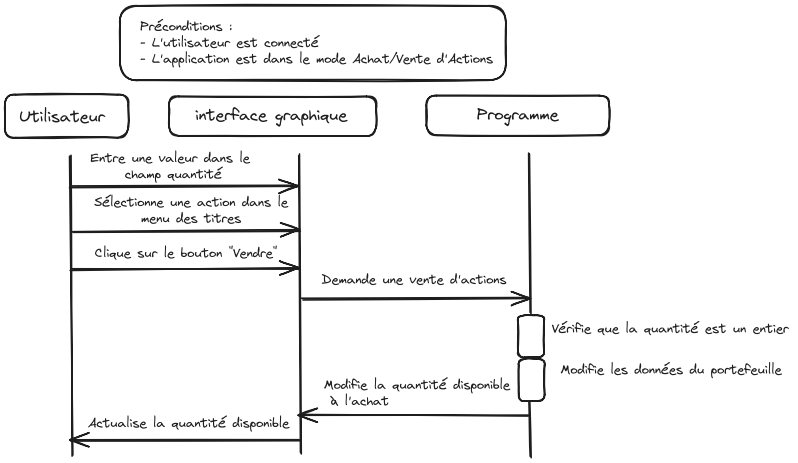
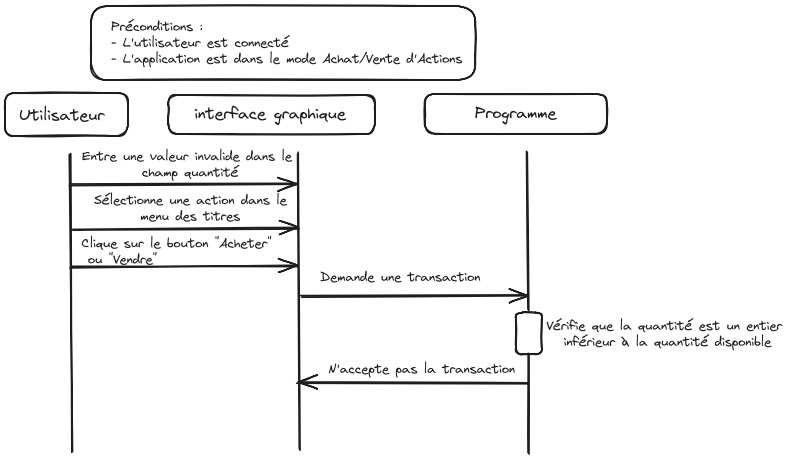


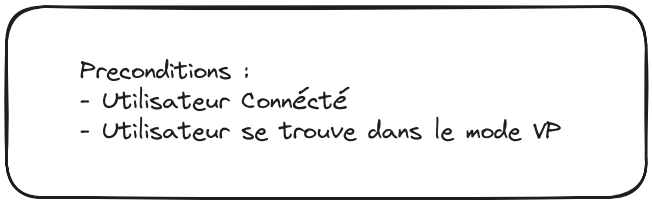
Diagramme de séquence pour un échec d'achat ou de vente (quantité rentrée invalide)

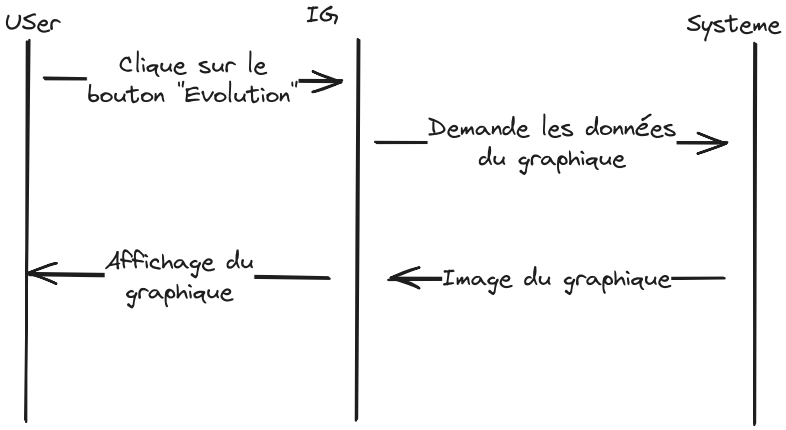


4) Mode Portefeuille

1) Affichage de l'évolution de la valeur du portefeuille

1. L'utilisateur clique sur le bouton "Evolution"
2. La courbe de l'évolution de la valeur du portefeuille s'affiche





2) Visualisation de la date

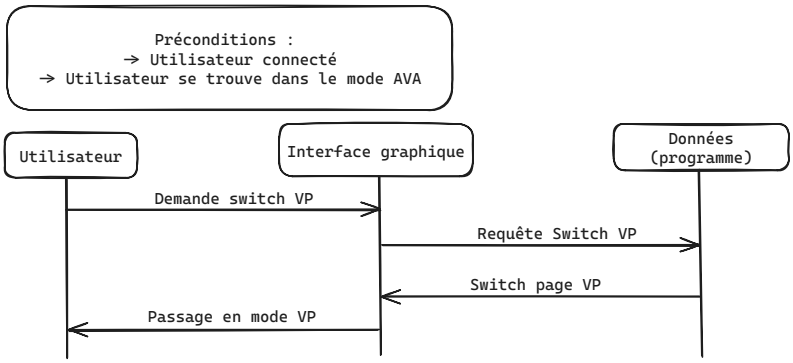
1. L'utilisateur visualise directement la date du jour actuel dans le jeu

5) Changer de mode

1) Passage du mode Achat / Vente Actions (AVA) au mode Visualisation Portefeuille (VP)

1. L'utilisateur clique sur le bouton "Changer mode"
2. Le système passe en mode VP et affiche la nouvelle page VP

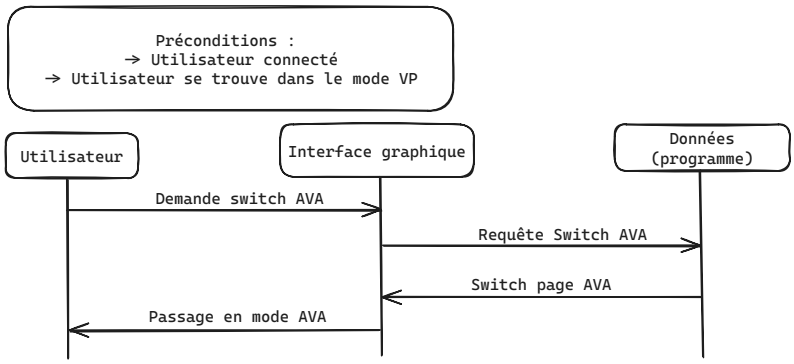
Diagramme de séquence pour un passage du mode AVA au mode VP



2) Passage du mode VP au mode AVA

1. L'utilisateur clique sur le bouton "Changer Mode"
2. Le système passe en mode AVA et affiche la nouvelle page AVA

Diagramme de séquence pour un passage du mode VP au mode AVA



6) Quitter Application

1. L'utilisateur clique sur "quitter" ou tente de fermer l'application
2. Le système envoit un message de demande de confirmation
   2.1. L'utilisateur clique sur "oui"
   2.1.1. Le système stoppe le programme
   2.2. L'utilisateur clique sur "non" et retourne à la page actuelle

Diagramme de séquence pour une fermeture réussie

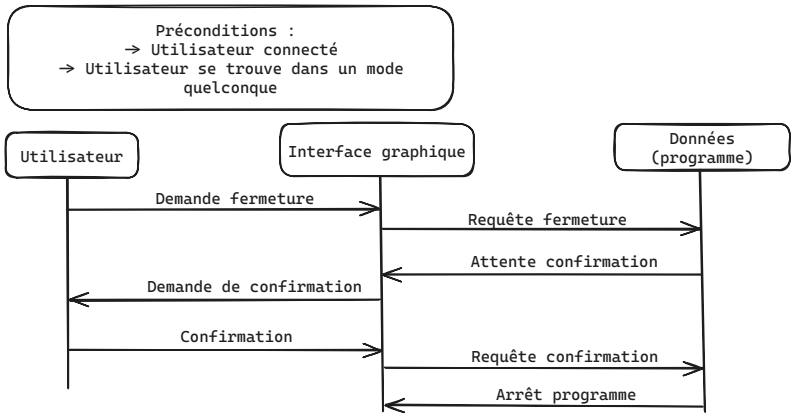
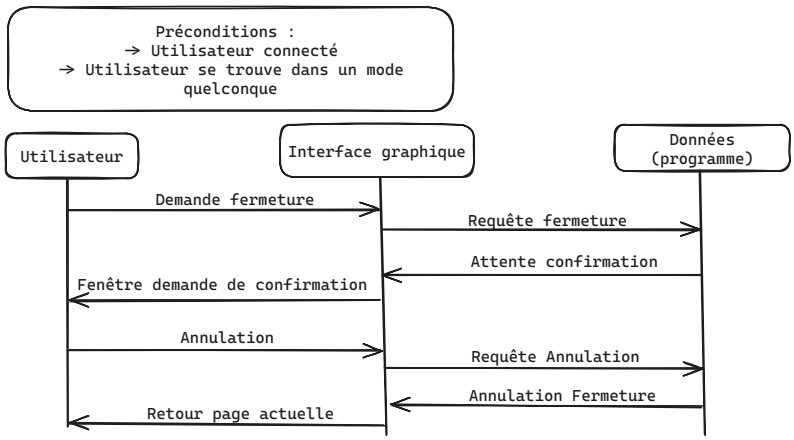


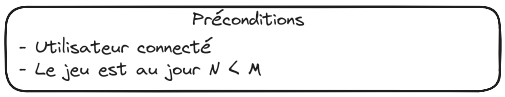
Diagramme de séquence pour une fermeture annulée

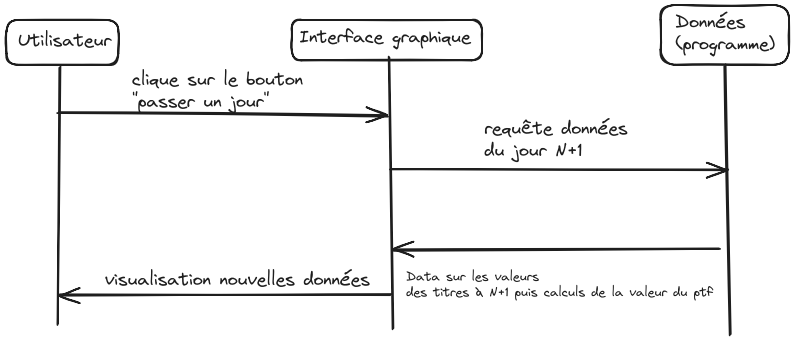


7) Passer au jour suivant

Le jeu s'arrête quand le jour M+1 est atteint (où M est le nombre de jour associés à la simulation).
Ici, disons que l'utilisateur est au jour N $\leq$ M.

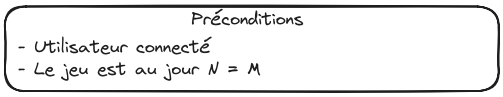
1. L'utilisateur clique sur le bouton "passer au jour suivant"
2. 1. Si N = M : le jeu est fini, affichage d'une fenêtre "jeu terminé" ainsi que du bilan utilisateur.
3. 2. Sinon, si le joueur est mode VP ou AVA : il reste toujours dans ce mode et les données sont actualisées au jour N+1.
   2.2.1 mise à jour des valeur des titres disponible sur le marché (affichage des nouvelles valeurs des actions détenus dans le portefeuille)
   2.2.2 calcul de la nouvelle valeur du portefeuille + mise à jours du graphique de l'évolution du portefeuille

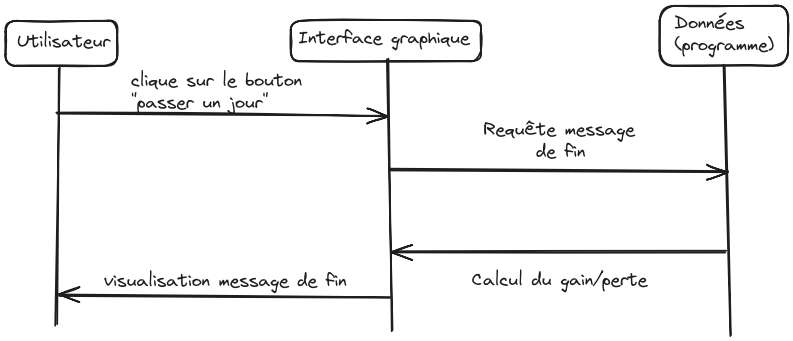




8) Bilan Fin du jeu

1. L'utilisateur est au dernier jour de la simulation et passe au jour suivant
2. Le système termine la simulation et renvoit le bilan utilisateur associé
   2.1 Si (valeur du portefeuille initiale - valeur du portefeuille au dernier jour) > 0
   Affichage du message: " vous avez gagné X "
   2.2 Si (valeur du portefeuille initiale - valeur du portefeuille au dernier jour) <= 0
   Affichage du message: " vous avez perdu X "
   Avec X la valeur absolue de la différence entre le cash final et initial.





3) Conception

Dans cette partie, nous entrons en détail dans l'implémentation de la simulation du jeu.

1) Architecture Logicielle pour l'interface graphique

Pour répondre à la demande d'interface graphique de la part du client, nous avons opté pour une Architecture **Modèle Vue Controleur**.

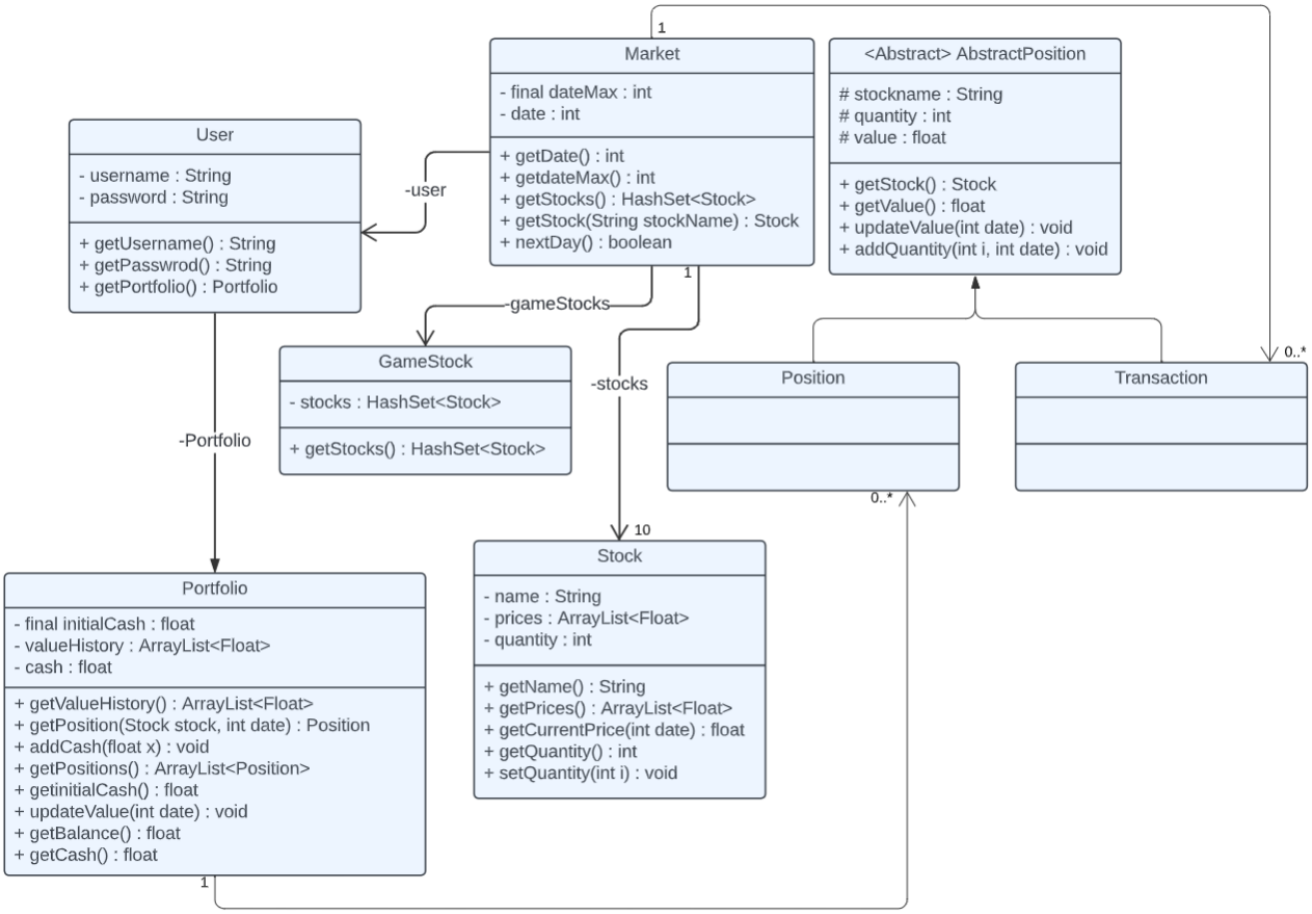
Diagramme d'Architecture Logicielle

Le diagramme associé étant assez conséquent, nous affichons séparement les deux parties de ce dernier (partie simulation du jeu et partie interface graphique).
La réalisation des graphiques a été faite à l'aide du logiciel **Lucidchart**. Les ressources de ce dernier étant limitées, il ne nous a pas été possible d'afficher l'entièreté des dépendances.
Nous avons choisi de ne représenter les dépendances qu'avec les classes déjà présentes dans le diagramme. Pour un exemple (à nouveau simplifié pour une bonne lisibilité) de schéma intégrant des dépendances, merci de jeter un coup d'oeil au fichier **"Example-Dépendances"** situé dans l'annexe.

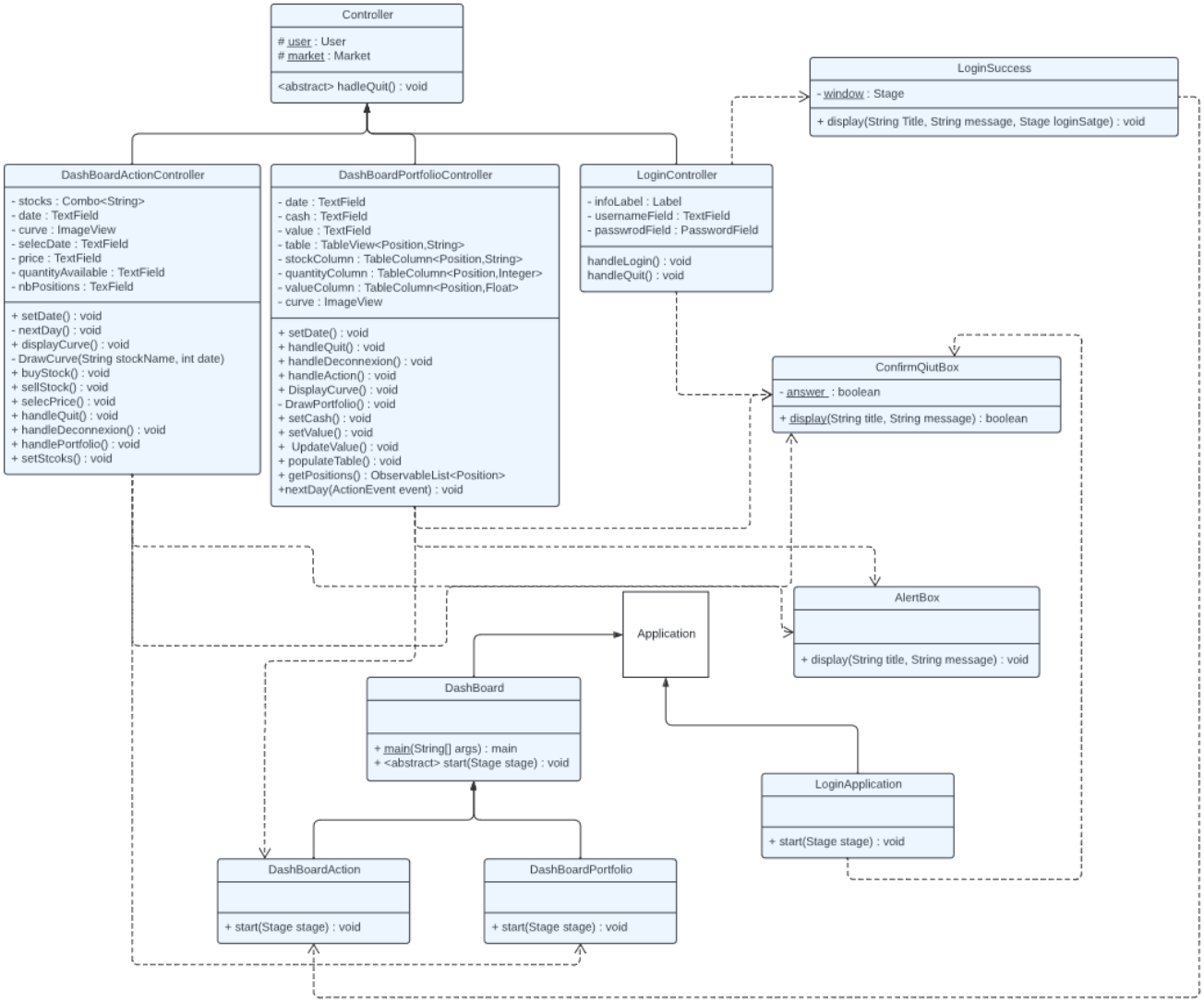
Afin de pouvoir décrire notre architecture, nous avons donc intégré certains attributs dépendants d'un autre classe au sein de la classe qui l'instancie.

Voici les deux diagrammes en question :

1) Diagramme de classes logicielles pour la partie simulation du jeu



2) Diagramme de classes logicielles pour la partie interface graphique



Ces 2 diagrammes sont également disponibles dans l'annexe.

2) Incrément Choisi

Afin de répondre au besoin de simulation d'un jeu de Bourse, nous avons choisi les incréments suivants :

- Il n'y a qu'**un seul compte** créé pour accéder à la plateforme, dont les logins sont : username = "a" et password = "p".
- La simulation se déroule sur **un seul marché** de manière **linéaire** : il faut terminer la simulation pour pouvoir la recommencer.
- Nous avons sélectionné un ensemble de **10 cours d'actions** sur une période de **20 jours** à partir de données réelles pour simuler les cours du marché. Les quantités disponibles sont fixées au début de la simulation.
- Les **achats et ventes d'actions** se font de manière **instantanée** : une opération affecte directement les stocks disponibles.

La simulation démmarre par une interface de connexion, où le joueur peut :

- **Insérer ses identifiants et se connecter** (un message est levé si les identifiants sont incorrects)
- **Quitter l'application**

Une fois la connexion effectuée, l'interface graphique se découpe en 2 fenêtres : le mode **action** et le mode **portefeuille**.

Au sein du **mode action**, le joueur peut :

- **Visualiser la date** de simulation (Ex. Jour 1)
- **Visualiser l'évolution du cours d'une action** depuis le début de la simulation
- **Visualiser le prix d'une action** sur une date égale ou antérieure à celle de la simulation actuelle, à condition que le format de la date (un entier) soit bien respecté.
- **Acheter des actions**, à condition que les stocks disponibles et le crédit (ou cash) de l'utilisateur soient suffisants.
- **Vendre des actions**, en quantité quelconque; l'utilisateur peut vendre les stocks qu'il possède déjà.
- **Passer au jour suivant** : cela met à jour la date, le prix des actions, les courbes et la valeur du portefeuille du joueur conformément au jour suivant.
- **Changer de mode** : passer en visualisation de portefeuille.
- **Se déconnecter** ou **quitter** l'application

Au sein du **mode protefeuille**, le joueur peut :

- **Visualiser la date** de simulation, le crédit (ou cash) disponible sur son compte ainsi que la valeur réelle de son portefeuille (cash + valeur des actifs possédés)
- **Se déconnecter** ou **quitter** l'application
- **Visualiser la composition de son portefeuille** : pour chaque position sur laquelle est placée l'investisseur, le nom de l'action, la quantité d'actions possédées (ou dûes si négatif) ainsi que sa valeur totale sont affichés.

- **Tracer la courbe d'évolution de son portefeuille** : la joueur peut analyser l'évolution de la valeur de son portefeuille depuis le début de la simulation.

Une fois la **simulation terminée** (dernier jour passé), le jeu affiche les résultats du joueur (gain ou perte par rapport à capital initial).

Un détail concernant les transactions

Nous avons intégré dans notre architecture la possibilité d'**enregistrer les différents échanges réalisés** sur le marche sous la forme de **transactions**. Par soucis de temps, nous n'avons pas implémenté cette fonctionnalité, mais il serait intéressant à l'avenir de la compéter pour offrir plus d'informations sur la simulation au joueur.

Un détail sur l'architecture Graphique

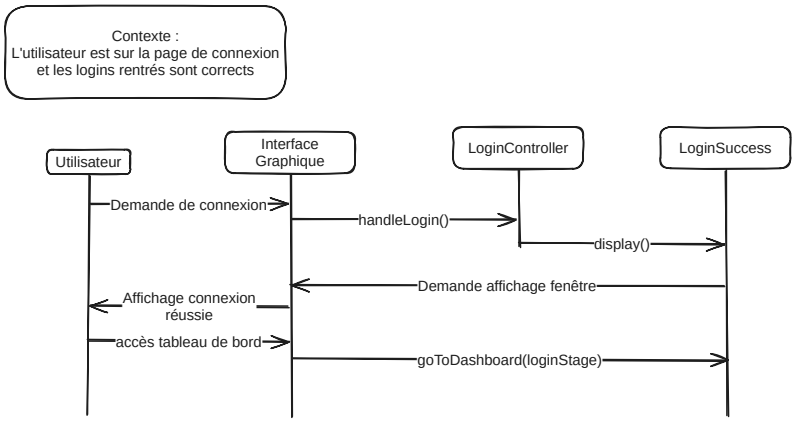
Chaque classe de contrôle (controller) est associé à un fichier fxml, d'où la dépendance via le package "java.fxml".

3) Diagrammes de séquences et communication architecturale

Afin d'illustrer les fonctionnement des différentes parties de notre système, nous proposons les quelques diagrammes de séquences suivants (les relations ne sont représentées qu'entre les objets que nous avons créés) :

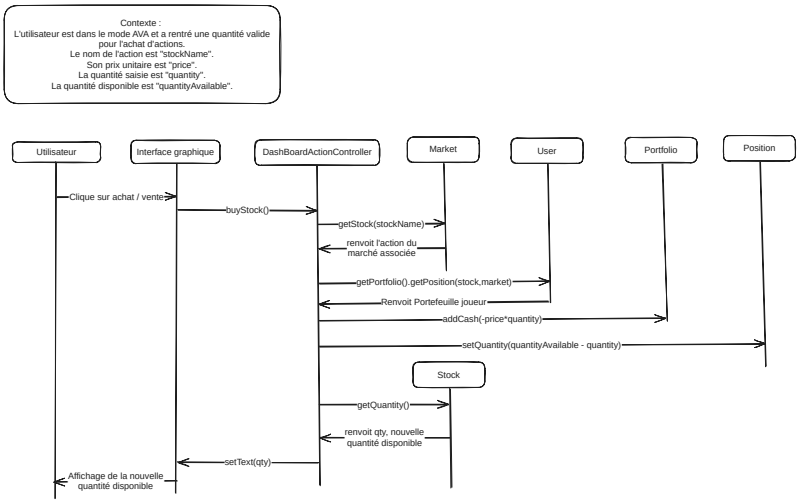
1) Diagramme de séquence logiciel pour une connexion réussie

Lors d'une **connection réussie**, le système crée tout d'abord une fenêtre de confirmation de connexion réussie, ferme les fenêtres existantes et ouvre une nouvelle fenêtre **DashBoardAction** (via la fonction goToDashboard).



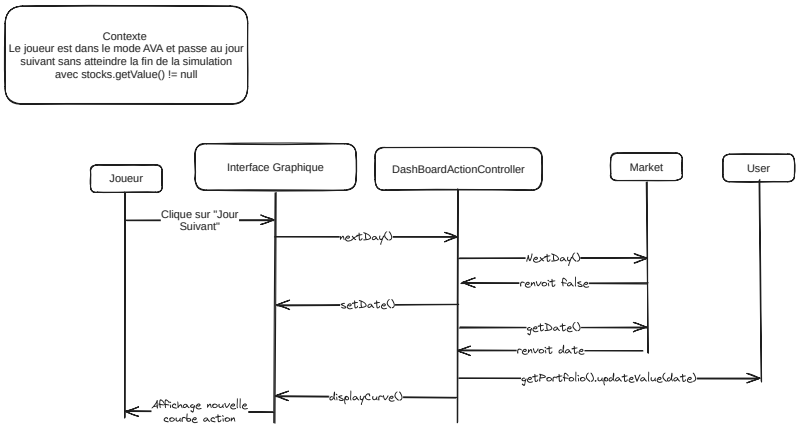
2) Diagramme de séquence pour un achat d'actions réussi

Lors d'un achat d'actions réussi, le système accède à la position du portefeuille de l'utilisateur et met à jour la quantité d'actions possédée pour cette position, la valeur du portefeuille et la quantité disponible restante. Cette dernière valeur est enfin mise à jour dans l'**affichage graphique**.



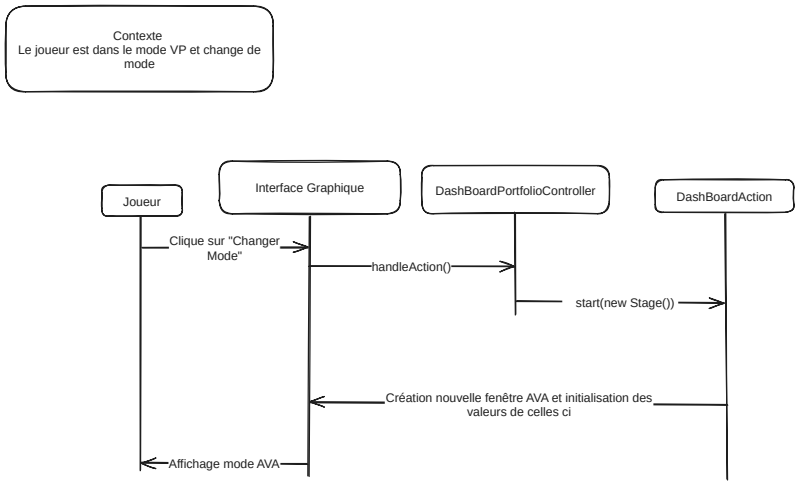
3) Passage au jour suivant (mode AVA) sans atteindre la fin de la simulation avec une action sélectionnée.

Lors d'un tel passage au jour suivant, le système met à jour la date du marché, puis met à jour la valeur du portefeuille et affiche la nouvelle courbe d'action.



4) Changement du mode VP vers le mode AVA

Lors d'un tel changement de mode, le système ferme la fenêtre actuelle et ouvre une nouvelle fenêtre du mode Action, en initialisant les valeurs qui lui sont liées.



4) Manuel Utilisateur

1) Connexion

Pour se connecter, il suffit d'entrer son nom d'utilisateur et son mot de passe puis de cliquer sur le bouton "Connexion."

Pour notre simulation, les logins sont les suivants :

- "a" pour l'**identifiant**
- "p" pour le **mot de passe**

Lors de sa 1ère connection, l'utilisateur commence avec un **cash initial** de 10 000 $

Il dispose alors de **M = 20 jours** pour réaliser le meilleur profit possible, c'est à dire obtenir la plus grosse valeur de portefeuille.

2) Mode Achat / Vente d'actions (AVA)

Une fois l'utilisateur connecté, l'application s'ouvre dans le mode achat et vente d'action (AVA). C'est dans ce mode que l'utilisateur peut effectuer des transactions et consulter l'évolution des prix des différents titres.

1) Sélection d'action

Lorsque l'utilisateur arrive dans ce mode, aucun titre n'est sélectionné. Pour en sélectionner un, l'utilisateur doit utiliser le menu défilant de sélection. Une fois, le titre sélectionné la courbe de l'évolution du prix depuis le début de la simulation s'affiche. L'utilisateur peut changer l'action sélectionnée de la même manière.

2) Achat d'action

Pour acheter une action, l'utilisateur doit entrer une valeur entière positive dans le champ quantité puis sélectionner un titre via le menu déroulant. Ces deux actions peuvent être effectuées dans n'importe quel ordre.
L'utilisateur doit ensuite cliquer sur le bouton "Acheter" pour valider et effectuer la transaction. Si la quantité est valide et disponible et que le capital de l'utilisateur est suffisant alors l'achat est pris en compte par l'application et les données du portefeuille sont modifiées en conséquences.

3) Vente d'action

Pour vendre une action, l'utilisateur doit entrer une valeur entière positive dans le champ quantité puis sélectionner un titre via le menu déroulant. Ces deux actions peuvent être effectuées dans n'importe quel ordre.
L'utilisateur doit ensuite cliquer sur le bouton "Vendre" pour valider et effectuer la transaction.
Si la quantité est valide alors la vente est prise en compte par l'application et les données du portefeuille sont modifiées en conséquences.

4) Consultation d'un prix

Pour consulter le prix d'une action à un certain jour, l'utilisateur doit sélectionner l'action qui l'intéresse et renseigner le numéro du jour pour lequel il veut obtenir le prix de cette dernière dans le champ jour.
Ces deux actions peuvent être effectuées dans n'importe quel ordre.
L'utilisateur doit ensuite cliquer sur le bouton "Ok" et si le jour renseigné est au bon format alors l'information cherchée est affichée dans le champ prix.

2) Mode Portefeuille

Dans le mode Portefeuille l'utilisateur peut consulter son capital ainsi que ses positions ouvertes.
Il peut aussi consulter l'évolution de son portefeuille depuis le début de la simulation en cliquant sur le bouton "Evolution".

3) Passer d'un mode à l'autre

Dans les modes portefeuille et AVA, un bouton "Changer de mode" est disponible pour passer d'un mode à l'autre.

4) Avancer la simulation

Dans les modes portefeuille et AVA, un bouton "Jour suivant" est disponible pour faire avancer la simulation d'un jour.
Le prix des différents actifs est alors mis à jour et la valeur du portefeuille est recalculée selon ces nouveaux prix.

5) Fin de la simulation

Une fois le 20ème jour atteint, la simulation se termine et le bilan de la valeur du portefeuille est affichée à l'utilisateur sous la forme d'un gain ou d'une perte d'argent.

5) Bilan

Ce travail de groupe a demandé beaucoup d'organisation.
La principale difficulté rencontrée est le manque de connaissance (théorique et code) et d'implication de la plupart des membres pour produire un outil fonctionnel et original.
Pour faire face à ce problème, nous avons élu **Antonin** comme **directeur de projet** chargé des tâches suivantes :

- Organiser les séances de travail en groupe pour produire l'analyse principale.
- Répartir le travail de chacun sur la documentation.
- Réaliser la conception de la simulation du jeu et de l'interface graphique ainsi que de leur implémentation.
- Préparer les rendus finaux.

Lors de la phase d'analyse, le **modeleur UML Lucidchart** a été particulièrement utile mais nous limitait en termes de composants que nous pouvions ajouter au schéma. Nous nous sommes donc concentrés sur les éléments principaux à afficher au client, un détail de l'architecture peut être trouvé dans le code source disponible dans le rendu.

Un autre problème rencontré est le manque de confort de certains membres du groupe avec Latex. Pour répondre à ce problème, **Antonin** à proposé une rédaction des documents sous l'éditeur de texte **Obsidian**. Cela s'est avéré être un bon compromis pour rendre un rapport correct.

Pour la partie code, la principale difficulté a été de générer un projet qui puisse interpréter les dépendances sur un autre ordinateur (ceux de l'ENSIMAG). L'utilisation de **Idea Intellij** a permis de mettre en place un projet **Maven** (pour la migration des dépendances) tout en offrant la structure principale de l'interface graphique sous **JavaFX**.

Nous espérons enfin que votre simulation vous plaira autant qu'elle nous a apporté en connaissances.