



Rapport final de SAé S4B01

# Rapport du projet SecureWin

Réalisé par

AUXILIEN Katia, JACQUEMIN Paul, SALA-MOCHIZUKI Yûki,  
TREMOULET-LEBRETON, VACHALDE Rémi

Projet encadré par

LAGUILLAUMIE Fabien et GARCIA Francis

Pour l'obtention du BUT Informatique, parcours déploiement d'applications communicantes  
sécurisées



# Remerciements

En premier lieu, nous tenons à exprimer notre profonde gratitude envers notre client fictif et responsable de parcours, M. Fabien LAGUILLAUMIE. Nous le remercions sincèrement pour le temps précieux qu'il a consacré à répondre à nos questions, que ce soit par courriel ou lors de visioconférences. Ses explications ont grandement enrichi notre compréhension des protocoles de sécurisation.

Nous souhaitons par ailleurs exprimer notre reconnaissance envers M. Antoine CHOLLET, notre professeur de Management des Systèmes d'Information durant le troisième semestre. Ses enseignements nous ont dotés des outils indispensables à la gestion de projet agile, tandis que son suivi attentif du projet durant cette période a été bénéfique pour notre équipe.

Nos remerciements vont également à M. Francis GARCIA pour son expertise dans les domaines du réseau et de la sécurité, qui ont été cruciaux pour la réussite de notre projet.

Nous tenons à adresser nos remerciements à Mme Cécile MANN et Mme Anita MESSAOUI, qui nous ont fourni les outils nécessaires à la rédaction de ce rapport.

Pour finir, nous voulons également exprimer notre reconnaissance envers M. Simon ROBILLARD pour ses enseignements précieux autour des tests d'application, ainsi qu'à M. Petru VALICOV pour son expertise et son savoir-faire en qualité de code.

# Résumé

## Résumé en français

Ce rapport présente le projet de refonte de l'application d'enchères électroniques SecureWin, tout en faisant référence au travail initial effectué lors de sa création. L'objectif principal est d'améliorer la sécurité, la fiabilité et la facilité de gestion de l'application tout en respectant les besoins croissants de l'utilisateur. Le projet vise également à mettre en place une architecture logicielle robuste, à renforcer la sécurité des connexions réseau, à améliorer le protocole d'enchères, à utiliser Docker pour le déploiement et à évaluer la sécurité de l'application. Il adopte une approche itérative avec des revues techniques régulières pour évaluer les progrès et planifier les étapes suivantes.

## Mots clés

Enchères électroniques, Sécurité des données, Chiffrement, Architecture logicielle, Réalisation itérative, Gestion de projet, Interface utilisateur, Docker, Protocole d'enchères, Sécurité réseau, Déploiement, Évaluation de sécurité.

## Abstract

This report presents the redesign project of the SecureWin electronic auction application, while also referring to the initial work done during its creation. The main objective is to enhance the security, reliability, and ease of management of the application while meeting the growing user needs. The project also aims to implement a robust software architecture, strengthen network connection security, improve the auction protocol, use Docker for deployment, and evaluate the security of the application. It adopts an iterative approach with regular technical reviews to assess progress and plan next steps.

## Key word

Electronic auctions, Data security, Encryption, Software architecture, Iterative development, Project management, User interface, Docker, Auction protocol, Network security, Deployment, Security evaluation.

## Table des figures

Figure 1 - Diagramme de cas d'usage : Application gestionnaire.....	12
Figure 2 - Diagramme de cas d'usage : Application vendeur.....	13
Figure 3 - Diagramme de cas d'usage : Application enchérisseur.....	14
Figure 4 - Diagramme d'activité.....	18
Figure 5 - Diagramme d'état transition d'une offre.....	19
Figure 6 - Diagramme de classe des objets utiles à l'enchère.....	20
Figure 7 - Diagramme de classe du package signedPack.....	21
Figure 8 - Diagramme de classe des classes dédié à la sécurité, provenant du package com.projetenchere.common.util.....	21
Figure 9 - Diagramme de classe du package com.projetenchere.common.view et com.projetenchere.bidder.view.....	24
Figure 10 - Diagramme de classe du package com.projetenchere.common.view et com.projetenchere.seller.view.....	25
Figure 11 - Diagramme de classe du package com.projetenchere.common.view et com.projetenchere.manager.view.....	26
Figure 12 - Capture d'écran de la liste de tâches du semestre 4.....	30
Figure 13 - Capture d'écran serveur discord.....	32
Figure 14 - Capture d'écran salon réunion du serveur discord.....	32
Figure 15 - Capture d'écran du product backlog sur GoogleDoc du semestre 3, sprint 4.....	33
Figure 16 - Capture d'écran de l'issueboard sur GitLab du semestre 3, sprint 2..	34
Figure 17 - Capture d'écran GoogleDrive de la SAE.....	35
Figure 18 - Détails d'un sprint en développement itératif.....	37
Figure 19 - Burndown chart de quatrième Sprint.....	38
Figure 20 - Burnup chart de résultat du quatrième Sprint.....	38
Figure 21 - Graphique de vélocité du quatrième Sprint.....	39
Figure 22 - BurnUp Chart de Production du quatrième Sprint.....	39

# Sommaire

<b>Remerciements.....</b>	<b>2</b>
<b>Résumé.....</b>	<b>3</b>
<b>Table des figures.....</b>	<b>4</b>
<b>Sommaire.....</b>	<b>5</b>
<b>Glossaire.....</b>	<b>6</b>
<b>Introduction.....</b>	<b>7</b>
<b>1. Analyse.....</b>	<b>8</b>
1.1. Présentation du sujet et analyse du contexte.....	8
1.1.1 Analyse de l'existant.....	8
1.1.2 Analyse du contexte.....	9
1.2. Analyse des besoins fonctionnels.....	9
1.2.1 Analyse des besoins du gestionnaire.....	11
1.2.2 Analyse des besoins du vendeur.....	12
1.2.3 Analyse des besoins de l'enchérisseur.....	13
1.3 Contraintes techniques, légales et ergonomiques.....	14
1.3.1 Contraintes légales.....	14
1.3.2 Contraintes techniques.....	15
1.3.3 Contraintes ergonomiques.....	15
<b>2. Rapport technique.....</b>	<b>16</b>
2.1 Conception.....	16
2.1.1 Conception de la partie métier.....	19
2.1.3 Conception des protocoles de sécurité.....	20
2.1.4 Conception du réseau.....	21
2.1.5 Conception de l'interface graphique.....	22
2.2 Réalisation.....	25
2.2.1 Réseau.....	25
2.2.2 Interfaces.....	25
2.3 Résultats.....	26
2.3.1 Tests.....	26
2.3.1 Manuels.....	26
<b>3. Gestion de projet.....</b>	<b>27</b>
3.1 Gestion d'équipe.....	27
3.2 Méthode de développement et outils.....	33
3.3 Bilan critique.....	38
<b>Conclusion.....</b>	<b>40</b>
<b>Bibliographie.....</b>	<b>42</b>
<b>Annexes techniques.....</b>	<b>43</b>
Annexe 1 - Système de Damgård-Jurik.....	43

# Glossaire

**Socket** : Une socket est une extrémité de communication bidirectionnelle entre deux programmes s'exécutant sur un réseau. Elle permet à un processus d'envoyer et de recevoir des données via une connexion réseau.

**Socket sécurisé/SSL** : Un socket sécurisé est une socket qui utilise des méthodes de cryptage pour sécuriser les données transmises entre les programmes. Il assure ainsi une communication sécurisée sur un réseau.

**Cryptographie** : La cryptographie est une méthode de protection de l'information en la transformant en un code indéchiffrable pour les personnes non autorisées. Elle utilise des algorithmes pour chiffrer et déchiffrer les données.

**Cryptographie asymétrique** : La cryptographie asymétrique, aussi appelée cryptographie à clé publique, est une méthode de cryptage utilisant une paire de clés distinctes : une clé publique pour chiffrer les données et une clé privée pour les déchiffrer.

**Clé publique/privée** : Dans la cryptographie asymétrique, la clé publique est accessible à tous et est utilisée pour chiffrer les données. La clé privée, quant à elle, est gardée secrète et est utilisée pour déchiffrer les données.

**Signature informatique** : Une signature informatique est un mécanisme cryptographique qui permet de garantir l'intégrité et l'authenticité d'un document électronique. Elle est créée en utilisant la clé privée d'un utilisateur et peut être vérifiée par n'importe qui en utilisant la clé publique correspondante.

**Certificat** : Un certificat est un document électronique qui lie une identité à une paire de clés publique/privée. Il est délivré par une autorité de certification et permet de vérifier l'identité d'une entité dans une communication sécurisée.

**Broadcast** : Le broadcast est une méthode de communication réseau où un message est envoyé à tous les nœuds du réseau simultanément.

**Sérialisation/Désérialisation** : La sérialisation est le processus de conversion d'un objet en une forme qui peut être facilement stockée ou transmise. La désérialisation, quant à elle, est le processus inverse, où les données sérialisées sont reconstituées en un objet.

**Enchère de Vickrey** : L'enchère de Vickrey est un type d'enchère scellée où les enchérisseurs soumettent leurs offres sans connaître les offres des autres. Le plus offrant remporte l'enchère, mais paie le prix de la deuxième offre la plus élevée.

**Autorité d'enchères** : L'autorité d'enchères est l'entité responsable de la gestion et de l'administration d'une enchère. Elle définit les règles de l'enchère, recueille les offres et détermine le gagnant.

# Introduction

Les enchères de Vickrey, ou enchères à plis fermés, sont des enchères durant lesquelles les enchérisseurs ne connaissent pas les offres de leurs concurrents. Ces offres doivent bien sûr ne pas être communiquées d'un point de vue métier, mais cela implique également un haut niveau de sécurisation : un seul acteur compromis, ou des communications mal chiffrées, compréhensibles par un potentiel attaquant, peuvent à eux seuls mettre à mal le déroulement du protocole. L'application dont il est question dans ce rapport, SecureWin, a été développée précédemment durant une première partie de ce projet pour implémenter ce protocole d'enchères.

L'application client-serveur d'enchères électroniques en Java est actuellement opérationnelle, mais présente quelques lacunes en termes de structure, de sécurité et de facilité de déploiement. Notre client souhaite donc procéder à une refonte complète pour répondre à ses besoins croissants en termes de fiabilité, de sécurité et de facilité de gestion de l'application.

Le projet vise à restructurer l'application d'enchères électroniques en Java en suivant les bonnes pratiques de conception, à renforcer la sécurité des connexions réseau, à mettre en place des tests à grande échelle, à finaliser l'interface graphique, à améliorer le protocole d'enchères, à utiliser Docker pour le déploiement et à évaluer la sécurité de l'application. Le projet sera organisé en sprints de développement itératifs, avec des revues techniques à la fin de chaque sprint pour évaluer les progrès et planifier les étapes suivantes.

Après avoir exposé notre analyse des besoins fonctionnels, les spécifications supplémentaires et les contraintes de cette application, nous présenterons dans le rapport technique les choix de conception que nous avons faits pour développer une architecture logicielle respectant la qualité du code et ses principes. Par la suite, nous aborderons notre méthode de réalisation de cette architecture, pour enfin présenter les résultats obtenus. Ensuite, nous exposerons notre approche dans la gestion du projet sur différents aspects, notamment la gestion d'équipe, de sorte à expliquer nos méthodes de développement et nos outils, pour finir sur un bilan critique sur notre gestion de travail.



# 1. Analyse

## 1.1. Présentation du sujet et analyse du contexte

Dans cette première section, nous fournissons une vue d'ensemble de notre analyse dans le contexte des ventes aux enchères en ligne. Nous examinerons également les contraintes et les défis spécifiques auxquels notre projet d'application, SecureWin, sera confronté.

### 1.1.1 Analyse de l'existant

Pour analyser l'existant dans le domaine des ventes aux enchères en ligne, nous avons adopté une approche méthodologique basée sur plusieurs critères d'observation. Nous avons examiné l'accessibilité, l'ergonomie de l'interface, les options de personnalisation, des fonctionnalités de suivi et des fonctionnalités avancées en termes de sécurité des sites web et applications existantes.

Depuis l'avènement d'Internet, le marché des ventes aux enchères en ligne s'est considérablement développé, offrant un large éventail d'options aux utilisateurs. Parmi les plateformes les plus populaires, on retrouve des sites web tels que Interencheres qui propose en plus de la vente en salle, VPauto spécialisé dans la vente de voitures d'occasion, Vavabid spécialisé dans les loisirs et voyages, mais vendent aussi un large panel de produit, Catawiki spécialisé dans les objets spéciaux et des objets de collection, Enchères-domaine, et enfin le plus connu eBay. Ce qui distingue chacun de ces sites web sont surtout leur interface ergonomique, leurs spécialisations en vente. L'avantage principal d'un site web est qu'il est facilement accessible et ne requiert aucune installation, ils offrent une grande mobilité et accessibilité. Le désavantage principal d'un site web est un accès restreint à certaines fonctionnalités.

De plus, des applications comme AuctionWorx, optimisée pour les appareils mobiles, BiddingOwl et Merkeleon offrent des fonctionnalités complètes pour la gestion des enchères, avec des options de personnalisation avancées autant sur l'interface que sur la création d'une enchère et un suivi des enchères organisé dans un calendrier.

Nos résultats mettent en évidence plusieurs points positifs : l'accessibilité facile via des sites web et des applications mobiles, des interfaces ergonomiques offrant une expérience utilisateur agréable, des options de personnalisation avancées permettant une adaptation aux besoins spécifiques des utilisateurs et la présence de fonctionnalités avancées pour une gestion efficace.

Cependant, nous avons également identifié certains points négatifs à prendre en compte : certaines fonctionnalités peuvent être limitées sur les sites web par rapport aux applications, une application sur site web implique de gérer des pics de trafic et des

problèmes de latences. Dans les deux cas, il faudra faire attention à se prémunir face aux attaques par déni de services.

### 1.1.2 Analyse du contexte

Le projet d'application SecureWin doit faire face à de nombreux concurrents sur le marché de l'enchère en ligne et d'applications d'enchères. Notre application doit s'intégrer dans un environnement multiplateforme pc et smartphone et combinant application web et logiciel, tout en offrant une expérience utilisateur de qualité.

L'intégration de l'application SecureWin dans un environnement multiplateforme, combinant à la fois une application web et un logiciel destiné aux ordinateurs et aux smartphones, génère certaines contraintes spécifiques en termes de choix technologiques.

Tout d'abord, pour garantir une expérience utilisateur fluide et cohérente, il est essentiel de sélectionner des technologies de développement compatibles et évolutives. Cela pourrait nécessiter l'utilisation de bibliothèques qui permettent le développement d'interface.

Ensuite, la sécurité est une préoccupation majeure dans le contexte des transactions en ligne. Le choix des technologies doit donc se concentrer sur des solutions robustes et bien établies en matière de sécurité des données, telles que des protocoles de chiffrement forts et des pratiques de développement sécurisé. De plus, la gestion efficace du trafic et des performances constitue un défi important dans un tel environnement. Les technologies choisies doivent être capables de gérer des charges de trafic variables et de maintenir des performances optimales même lors de pics d'activité.

Enfin, la compatibilité avec une variété de systèmes d'exploitation et de dispositifs doit être prise en compte lors du choix des technologies. Cela peut nécessiter des tests approfondis et une optimisation spécifique pour chaque plateforme cible.

En résumé, l'intégration de SecureWin implique des contraintes spécifiques en termes de choix technologiques, notamment en matière de compatibilité, de sécurité, de gestion du trafic et des performances. Ces contraintes doivent être soigneusement prises en compte lors de la conception et du développement de l'application pour garantir son succès et sa fiabilité sur le marché.

## 1.2. Analyse des besoins fonctionnels

Dans le cadre de notre projet, nous avons conçu trois applications distinctes pour répondre aux besoins spécifiques de trois types d'utilisateurs différents. Chaque application a été développée en tenant compte des besoins et des préférences uniques de son utilisateur cible. Dans cette analyse, nous examinerons les besoins de chaque groupe d'utilisateurs pour répondre à ces besoins de manière efficace et adaptée.

### 1.2.1 Analyse des besoins du gestionnaire

Premièrement, l'utilisateur qui a le rôle de gestionnaire, ou autorité de gestion, est chargée de réceptionner les enchères, les transmettre aux futurs enchérisseurs : les enchères en cours et leurs informations, ainsi qu'une clé publique permettant le chiffrement des prix. Ce rôle sera surtout destiné aux entreprises ou organisations.

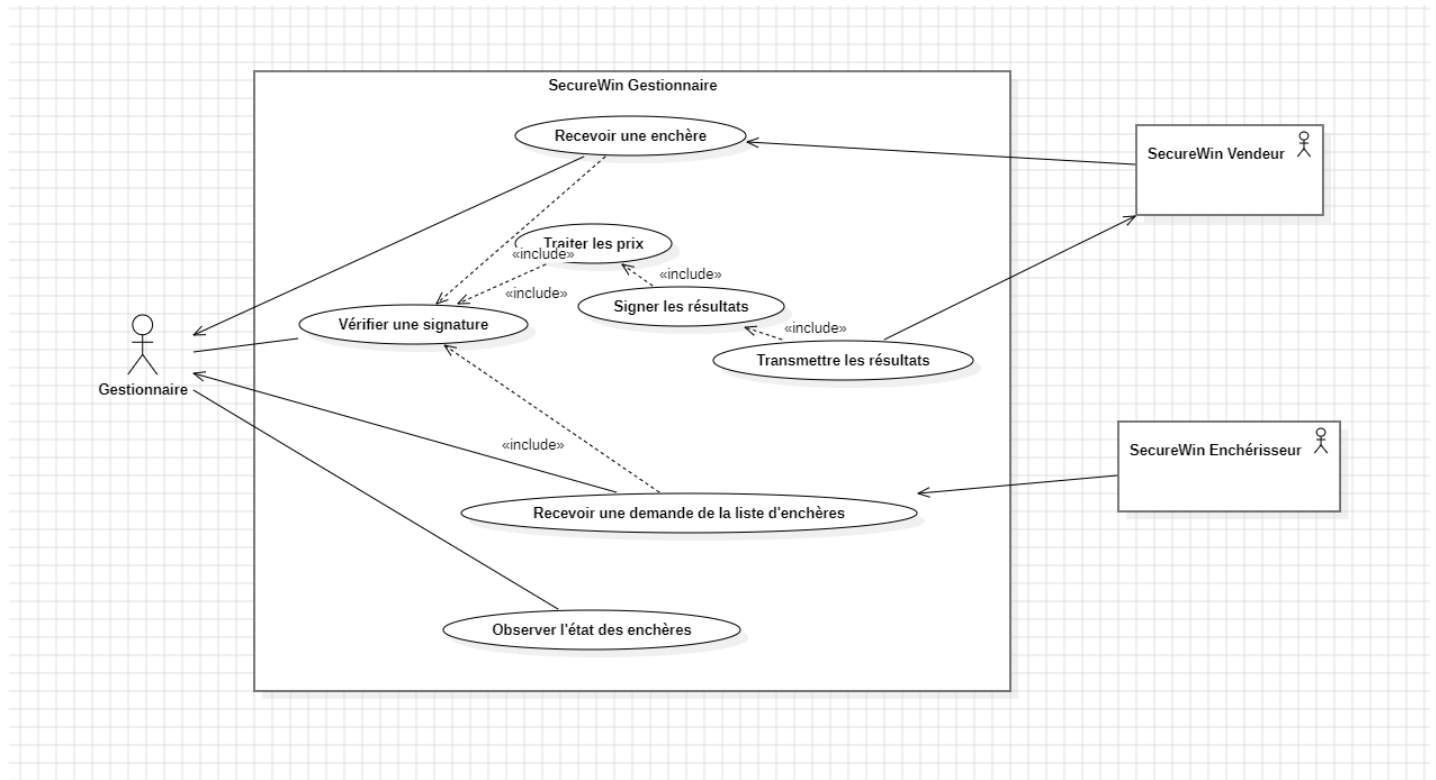


Figure 1 - Diagramme de cas d'usage : Application gestionnaire.

### 1.2.2 Analyse des besoins du vendeur

Deuxièmement, l'utilisateur dont le rôle est vendeur correspond aux utilisateurs a besoin de proposer son enchère, il est chargé de recueillir les participants, mettre fin aux enchères, et transmettre le tout à l'autorité de gestion pour avoir les résultats, il transmet les résultats aux enchérisseurs et attend la manifestation d'un gagnant.

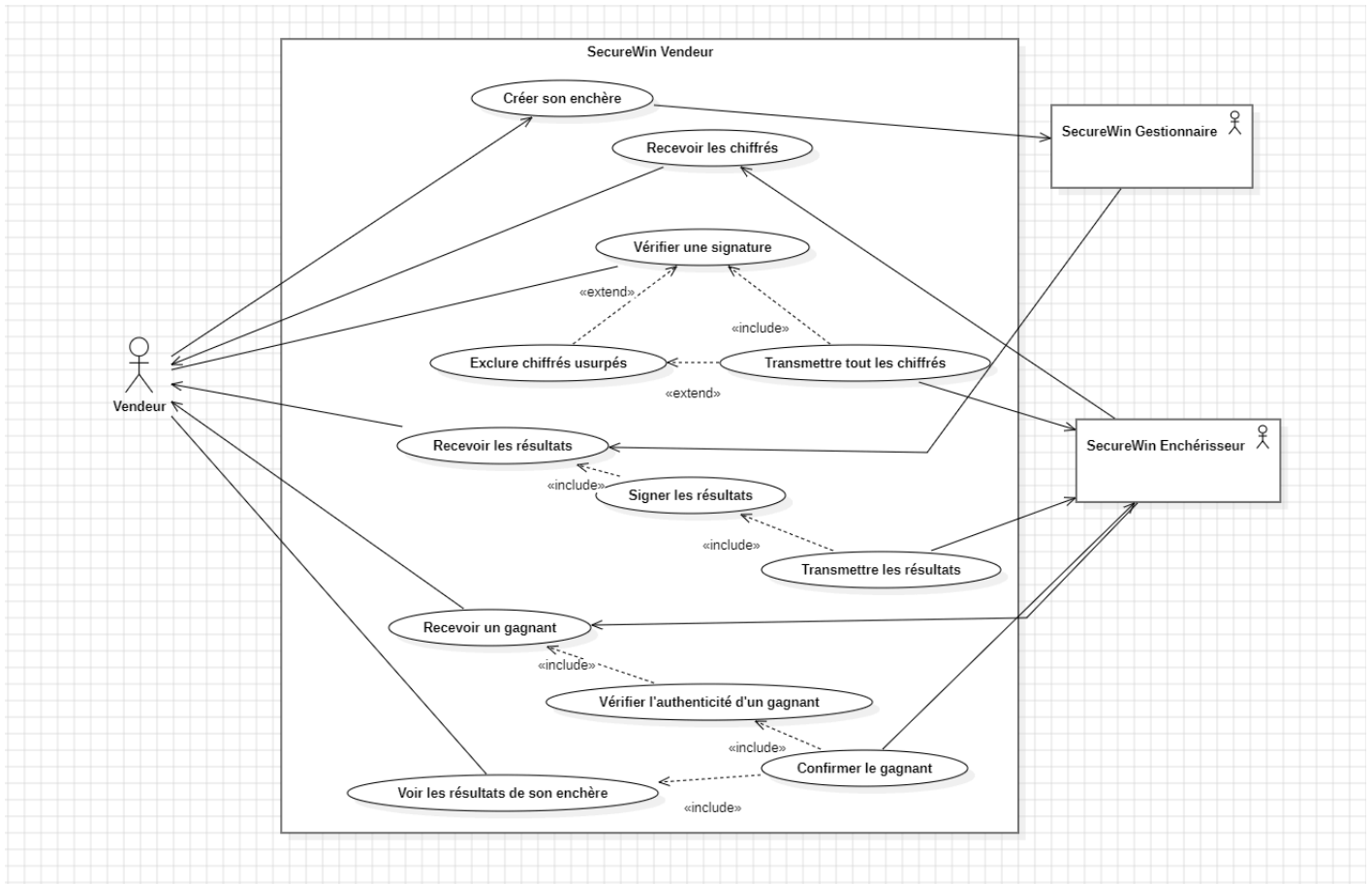


Figure 2 - Diagramme de cas d'usage : Application vendeur.

### 1.2.3 Analyse des besoins de l'enchérisseur

Troisièmement, l'enchérisseur est l'utilisateur qui a le rôle de participant, il est un potentiel futur acheteur. Il peut obtenir toutes les enchères auprès du gestionnaire, prendre connaissance des informations et descriptions des enchères, et enchérir sur celle qui l'intéresse.

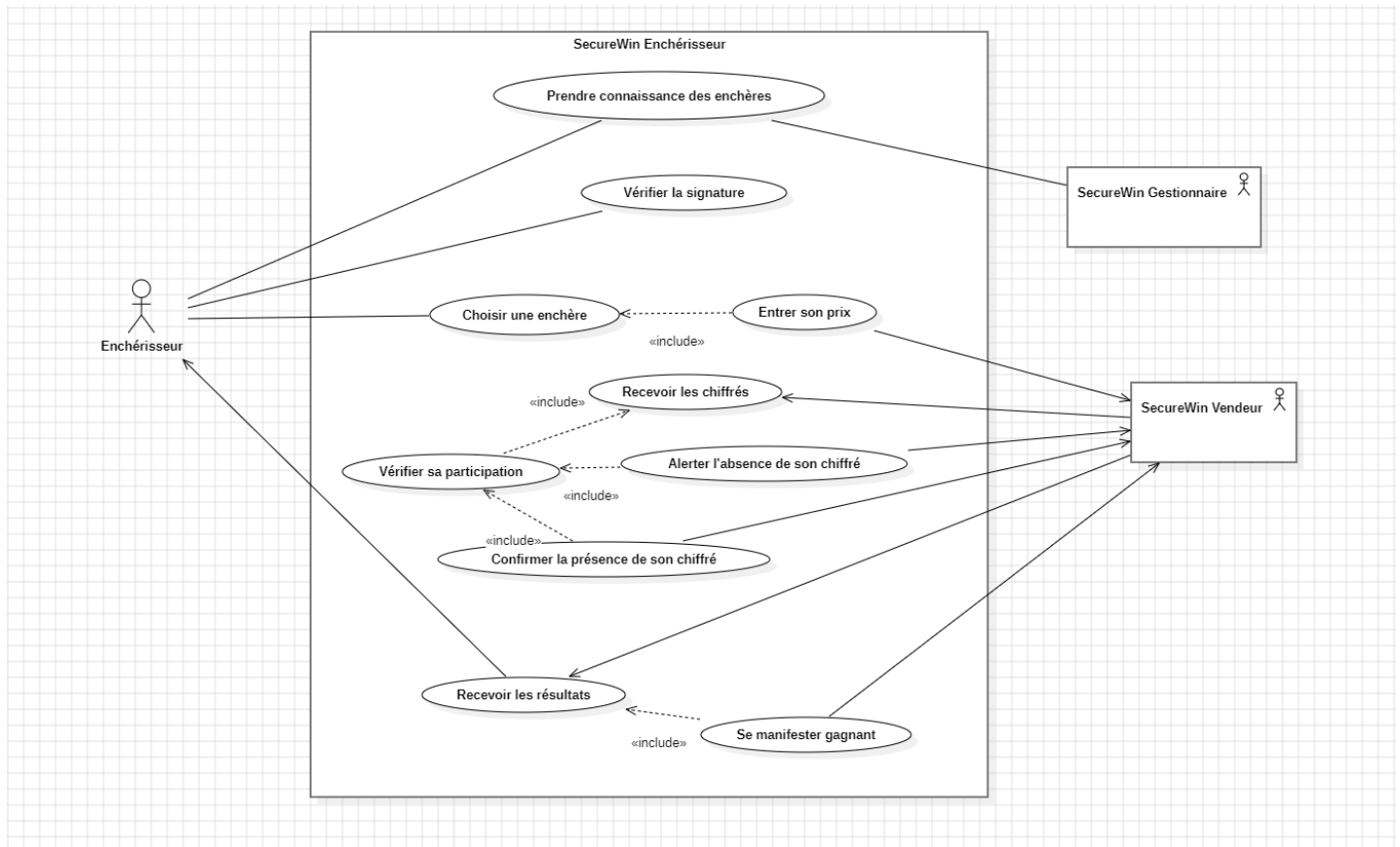


Figure 3 - Diagramme de cas d'usage : Application enchérisseur.

## 1.3 Contraintes techniques, légales et ergonomiques

### 1.3.1 Contraintes légales

Pour le futur développement, nous avons dû nous assurer que l'application soit conforme aux réglementations en matière de protection des données personnelles. Ainsi, il faut que nos données utilisateur soient stockées de manière sécurisée et qu'elles ne soient pas utilisées à des fins autres que celles pour lesquelles elles ont été collectées. Cela concernera surtout les entreprises qui utiliseront l'application gestionnaire.

Ensuite, nous avons dû nous assurer que l'application soit sécurisée et qu'elle ne peut pas être piratée. Il faudra que les organisations s'assurent que les données des utilisateurs soient stockées de manière sécurisée et qu'elles ne soient pas accessibles à des tiers non autorisés.

Nous avons aussi dû nous assurer que l'application est conforme aux lois et réglementations en vigueur dans la juridiction où elle sera utilisée. Durant le développement, nous avons fait en sorte que l'application ne viole pas les droits de propriété intellectuelle d'autrui et il faudra se munir de mesures pour qu'elle ne soit pas utilisée à des fins illégales.

Les organisations devront rédiger des contrats d'utilisation clairs et précis pour les utilisateurs de l'application. Ces contrats doivent définir les droits et les obligations des utilisateurs, ainsi que les limites de responsabilité de votre part.

Sachant que l'application sera utilisée pour la vente de biens ou de services, les organisations devront rédiger des conditions générales de vente claires et précises. Ces conditions doivent définir les modalités de paiement, de livraison, de retour des produits et autres. Les conditions générales de vente (CGV) sont un document contractuel qui définit les modalités de vente entre un vendeur et un acheteur passant potentiellement par des mandataires (vous). Les CGV doivent être claires et précises et doivent inclure les informations suivantes :

- **Objet des CGV** : Les CGV doivent définir l'objet de la vente et les produits ou services proposés.
- **Prix et modalités de paiement** : Les CGV doivent préciser le prix des produits ou services proposés, ainsi que les modalités de paiement.
- **Livraison** : Les CGV doivent préciser les modalités de livraison des produits ou services proposés.
- **Garanties** : Les CGV doivent préciser les garanties offertes par le vendeur pour les produits ou services proposés.
- **Droit de rétractation** : Les CGV doivent préciser les conditions de retour des produits et les modalités de remboursement.
- **Propriété intellectuelle** : Les CGV doivent préciser les droits de propriété intellectuelle du vendeur sur les produits ou services proposés.

- Responsabilité : Les CGV doivent préciser les limites de responsabilité du vendeur en cas de dommages causés par les produits ou services proposés.
- Loi applicable et juridiction compétente : Les CGV doivent préciser la loi applicable et la juridiction compétente en cas de litige.

Les organisations devront s'assurer que l'usage de l'application ne viole pas les droits de propriété intellectuelle d'autrui. Elles devront veiller à ce que les marques, les logos et les autres éléments de propriété intellectuelle soient utilisés conformément aux lois et réglementations en vigueur.

Les organisations devront définir les limites de leur responsabilité en cas de dommages causés par l'application. Elles veilleront à ce que les utilisateurs soient informés des limites de responsabilité de votre part.

### 1.3.2 Contraintes techniques

Les applications devront être programmées en Java et, pour les applications clients, devront être dotées d'une interface graphique,

En termes de communication, les sockets SSL devront être mis en place avec la librairie spécialisée Java.

### 1.3.3 Contraintes ergonomiques

Dans le cadre des contraintes ergonomiques, une attention particulière est portée à la clarté et à la lisibilité de l'interface utilisateur des applications. Les éléments visuels tels que le texte, les boutons et les icônes sont soigneusement conçus pour être facilement identifiables et compréhensibles, ce qui garantit une expérience utilisateur fluide. De plus, une navigation intuitive est essentielle pour permettre aux utilisateurs de trouver rapidement les fonctionnalités dont ils ont besoin. L'architecture de l'application est pensée de manière à ce que la navigation entre les différentes sections et fonctionnalités soit intuitive, réduisant ainsi la friction dans l'utilisation quotidienne. En ce qui concerne l'ergonomie des interactions, les actions courantes telles que la saisie de données et la sélection d'options sont optimisées pour être intuitives et peu exigeantes sur le plan cognitif. Les utilisateurs peuvent interagir avec l'application de manière naturelle et efficace, ce qui contribue à une expérience utilisateur positive et satisfaisante.

## 2. Rapport technique

### 2.1 Conception

Étant donné que SecureWin est en réalité trois applications, il a fallu trouver une méthode pour minimiser la duplication de code et de répartir le code spécialisé à chaque application.

Nous avons donc décidé de diviser le code en plusieurs package : un package commun, regroupant tout le code réutilisable entre les trois parties, et trois package spécialisés (enchérisseur, vendeur, autorité).

L'application est généralement conçue en fonction du principe MVC (Model - View - Controller). Ainsi, elle est divisée en trois parties :

- Une partie métier;
- Une partie interface;
- Une partie qui lie les deux précédentes.

À cela s'ajoute une partie réseau, qui s'occupe de la communication entre les trois applications, et une partie sécurité, soit le chiffrement et les signatures.

Pour la gestion d'une enchère de son début à sa fin, les applications communiquent généralement de la manière suivante:



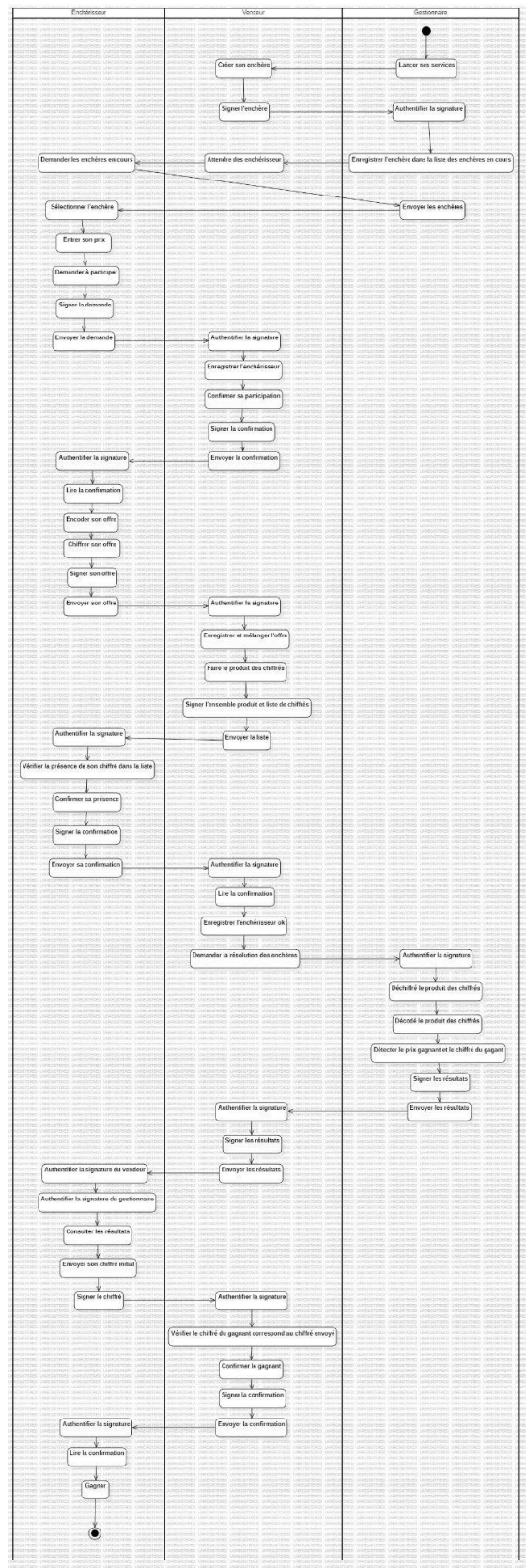


Figure 4 - Diagramme d'activité.

Une enchère, durant sa gestion, passe à travers plusieurs états. Voici un diagramme représentant ces états :

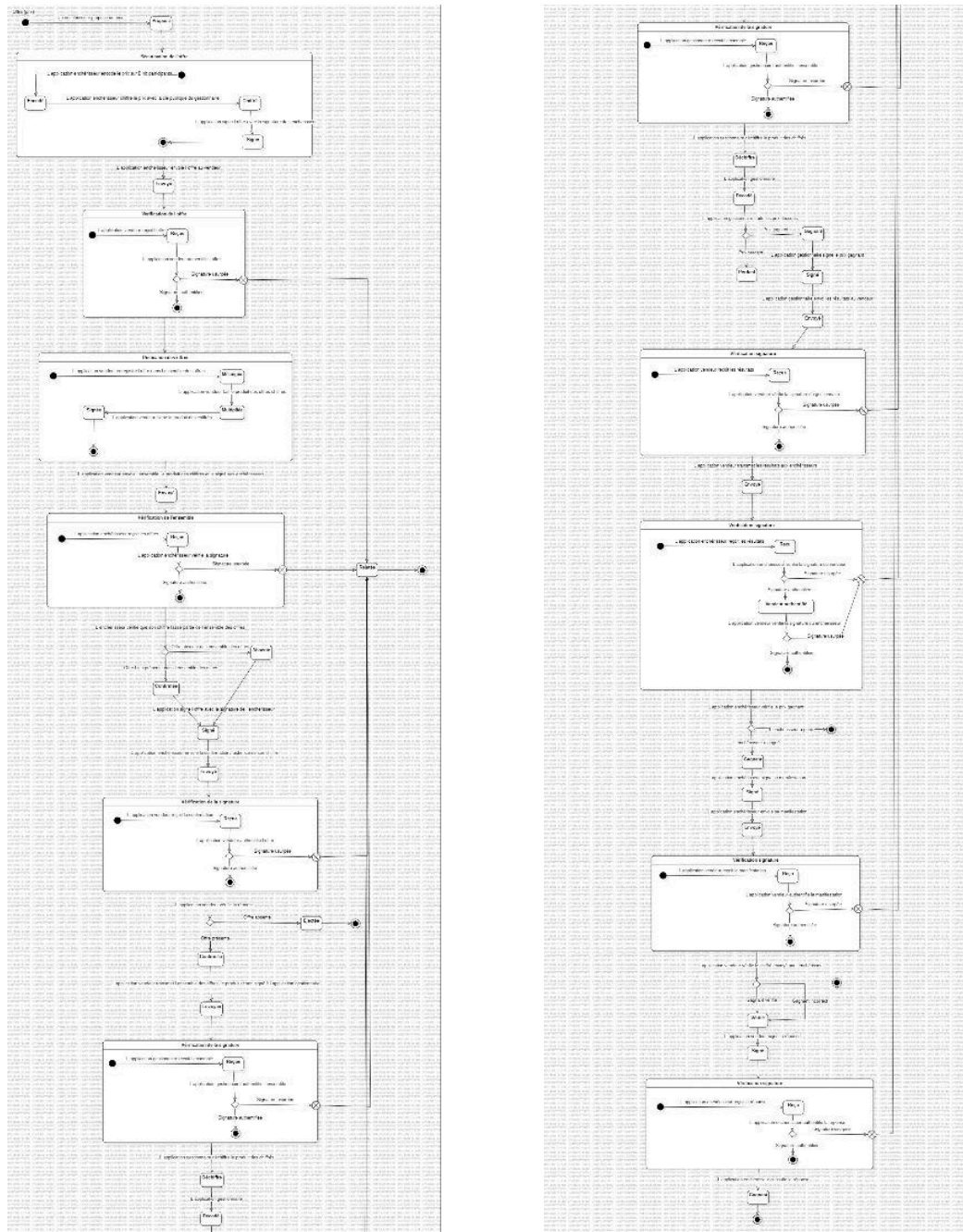


Figure 5 - Diagramme d'état transition d'une offre.

### 2.1.1 Conception de la partie métier

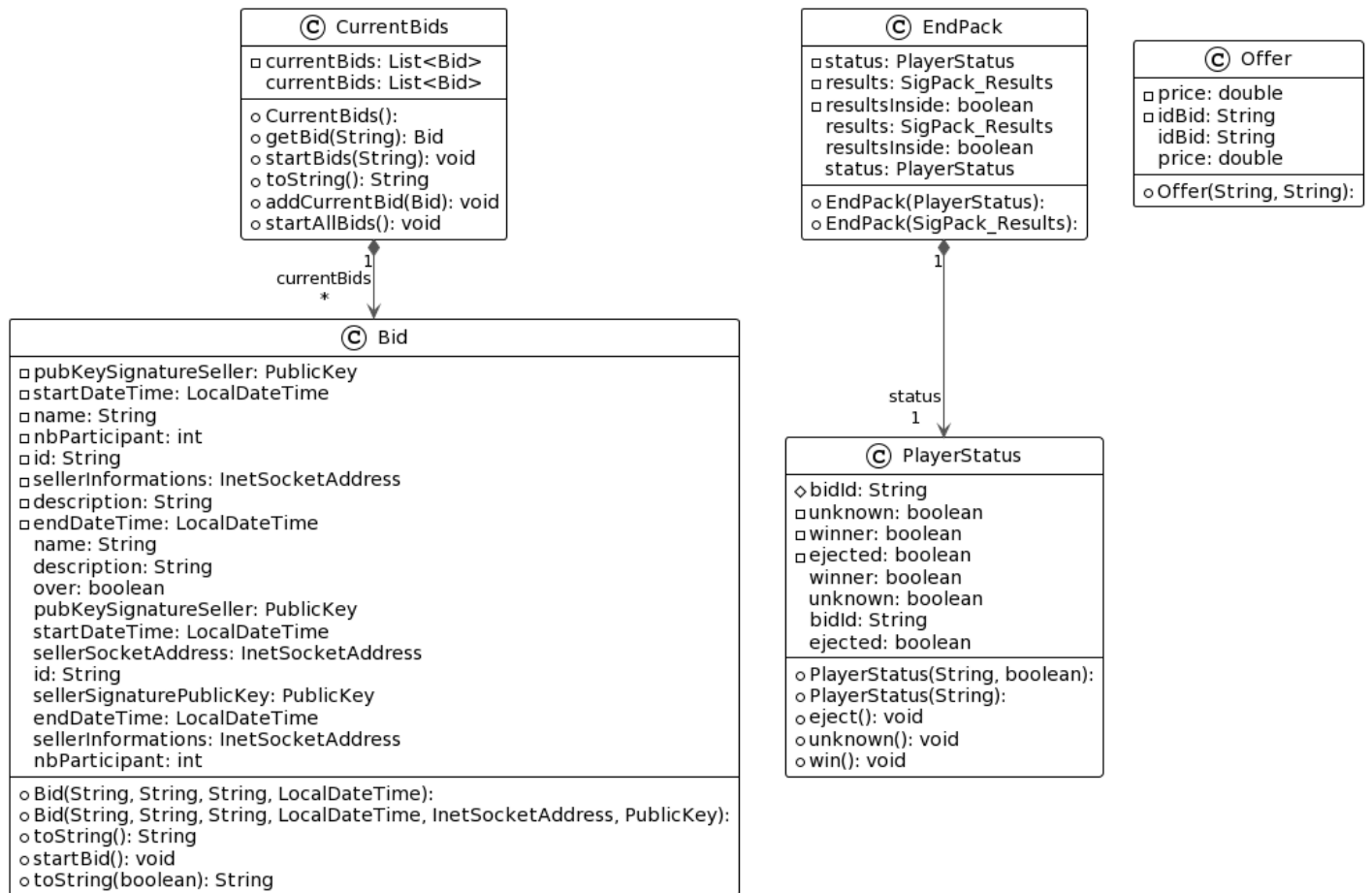


Figure 6 - Diagramme de classe des objets utiles à l'enchère.

### 2.1.3 Conception des protocoles de sécurité

Des classes spéciales appelées *SigPack* sont utilisées pour contenir les signatures et les chiffrés qui doivent être échangés sur le réseau. Elles permettent de vérifier les signatures automatiquement et d'extraire une version signée ou non de l'objet.

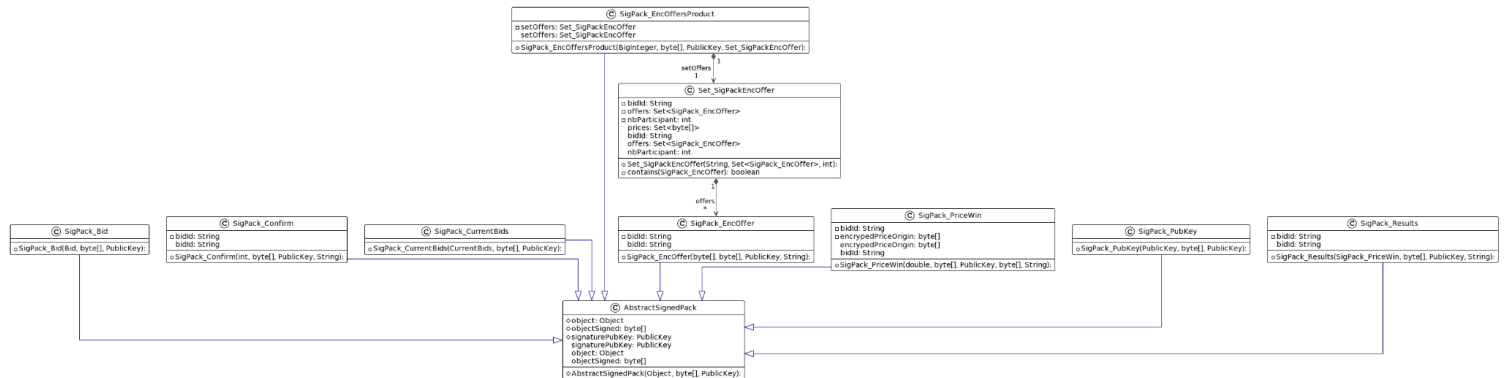


Figure 7 - Diagramme de classe du package signedPack

Le chiffrement utilisé et le système de Damgård-Jurik avec deux itérations ( $s = 2$ ), qui a la particularité d'être linéairement homomorphe : multiplier le chiffré par  $n$  revient à ajouter  $n$  au message. L'algorithme utilisé est disponible en annexe. Les seules instructions qui ne suivent pas directement l'algorithme sont l'encodage des clés. La clé privée contient  $p$  et  $q$  en binaire simplement juxtaposés, et la clé publique contient leur produit.

Les signatures sont en RSA. Elles sont stockées dans des fichiers JKS, dans le dossier `~/home/$USER/.config/securewin`. C'est également dans ce dossier que sont présents l'adresse IP du manager,

Le chiffrement et le réseau sont gérés via des classes statiques Util pour les rendre facilement utilisables.

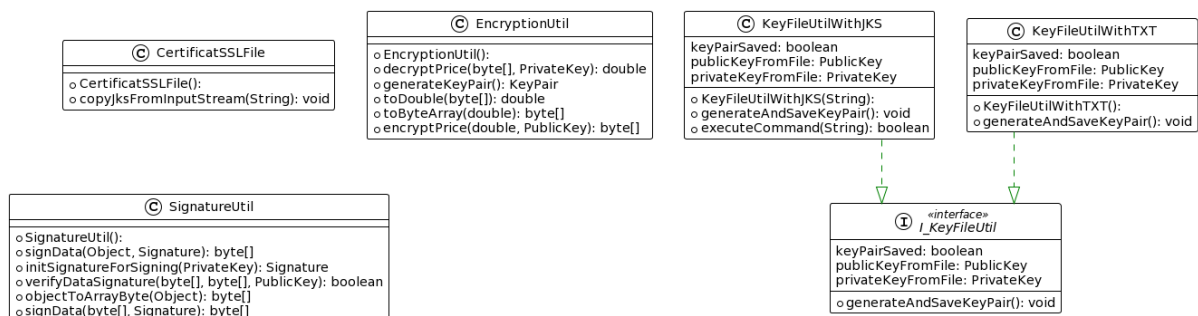


Figure 8 - Diagramme de classe des classes dédié à la sécurité, provenant du package `com.projetenhere.common.util`

## 2.1.4 Conception du réseau

L'échange de paquets est organisé en client/serveur, représentés par les classes Client et Serveur. Les données échangées sont représentées par la classe DataWrapper.

Cette classe contient deux choses:

- La donnée à échanger, sous la forme d'un objet sérialisable,
- Un en-tête qualifiant le paquet.

Ainsi, quand un serveur est démarré, on lui ajoute une liste d'en-tête auxquels répondre. À chaque en-tête est associée une implémentation de l'interface *IDataHandler*, contenant une méthode handle qui spécifie quoi faire avec un paquet, et retourne un autre DataWrapper.

À l'ouverture d'une socket, le serveur ouvre un thread avec un *ClientAcceptor*. Chaque ClientAcceptor contient un socket. Quand il reçoit un paquet dans son socket, il regarde sur la table si le header existe, et si c'est le cas il appelle la méthode handle du handler correspondant, et renvoie le DataWrapper retourné.

Cette classe Serveur est utilisée par le vendeur et le gestionnaire.

La classe Client est plus simple. Elle possède deux méthodes, fetch et fetchWithData. Fetch envoie un DataWrapper avec un simple en-tête de requête, sans données, et fetchWithData envoie des données en plus du header. Ces deux fonctions s'attendent à recevoir un certain header en retour, et renvoient une erreur si ce n'est pas le bon.

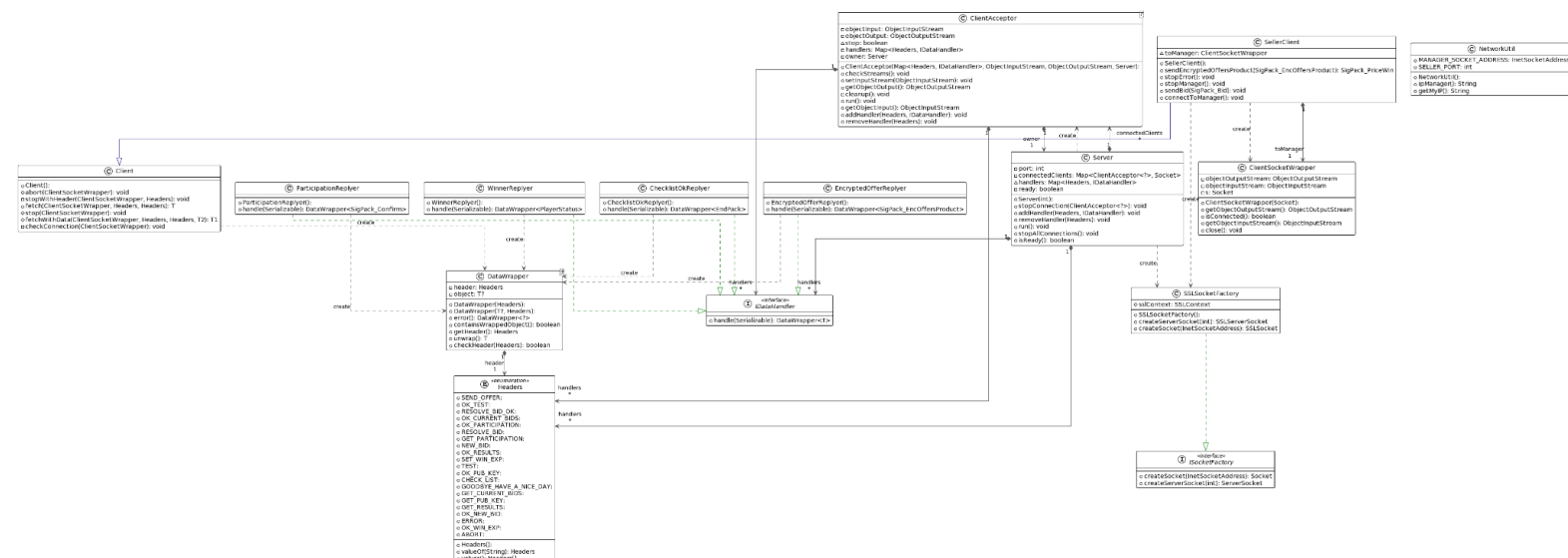


Figure ? : classes réseau

### 2.1.5 Conception de l'interface graphique

L'interface graphique (GUI) est réalisée avec la librairie JavaFX. Étant donné que les applications doivent également pouvoir fonctionner en ligne de commande (CLI), il y a pour chaque application deux classes *loader*, l'une chargeant le GUI l'autre la CLI.

Le comportement de ces deux interfaces utilisateurs est standardisé par des interfaces (au sens objet orienté du terme). Il y a une interface en commun, mettant à disposition des méthodes générales permettant par exemple d'afficher un message. Pour chaque application, il y a une sous-interface mettant à dispositions de méthodes supplémentaires spécifiques.

Des couches intermédiaires implémentent ces interfaces pour la CLI et le GUI. Quand l'application est lancée, un loader instancie la GUI ou la CLI en fonction de si l'argument -g a été passé ou non.

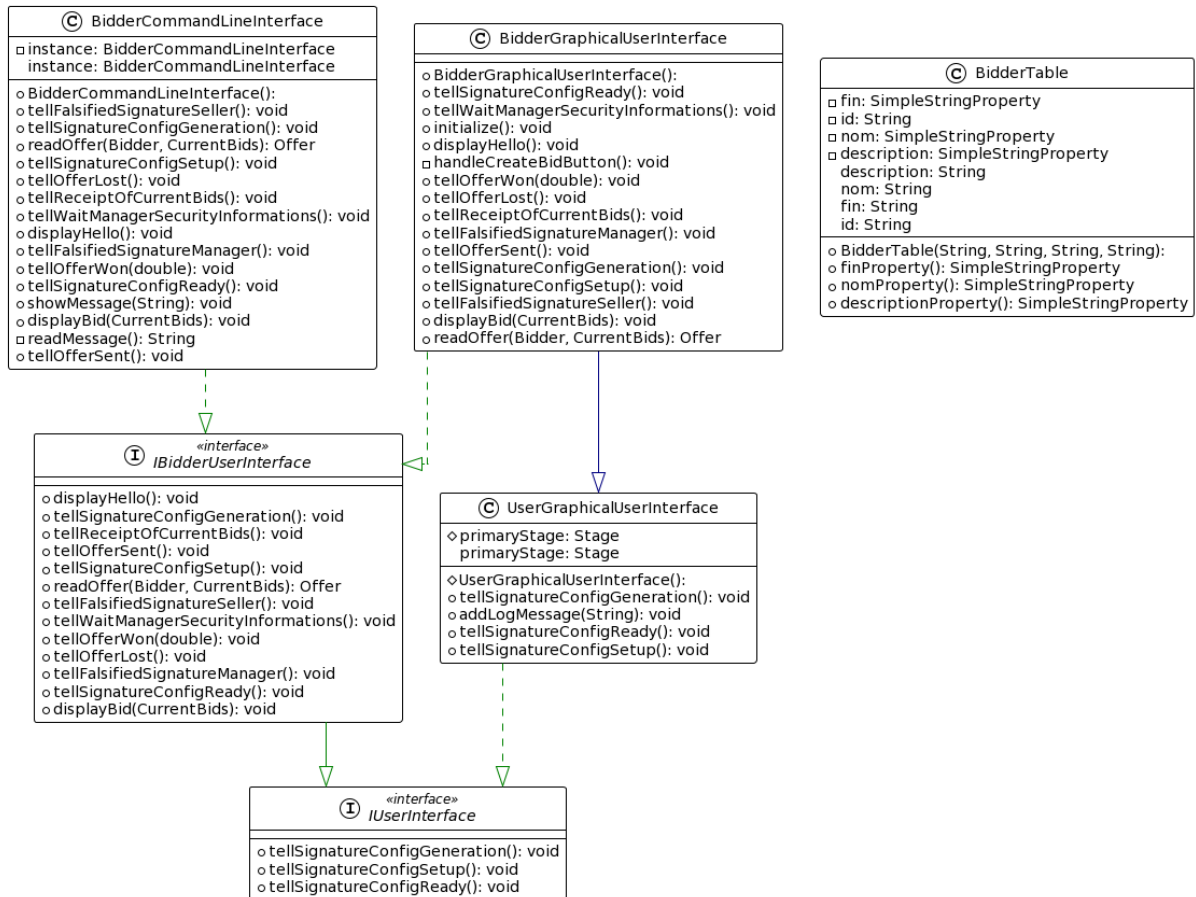


Figure 9 - Diagramme de classe du package com.projetenchere.common.view et com.projetenchere.bidder.view

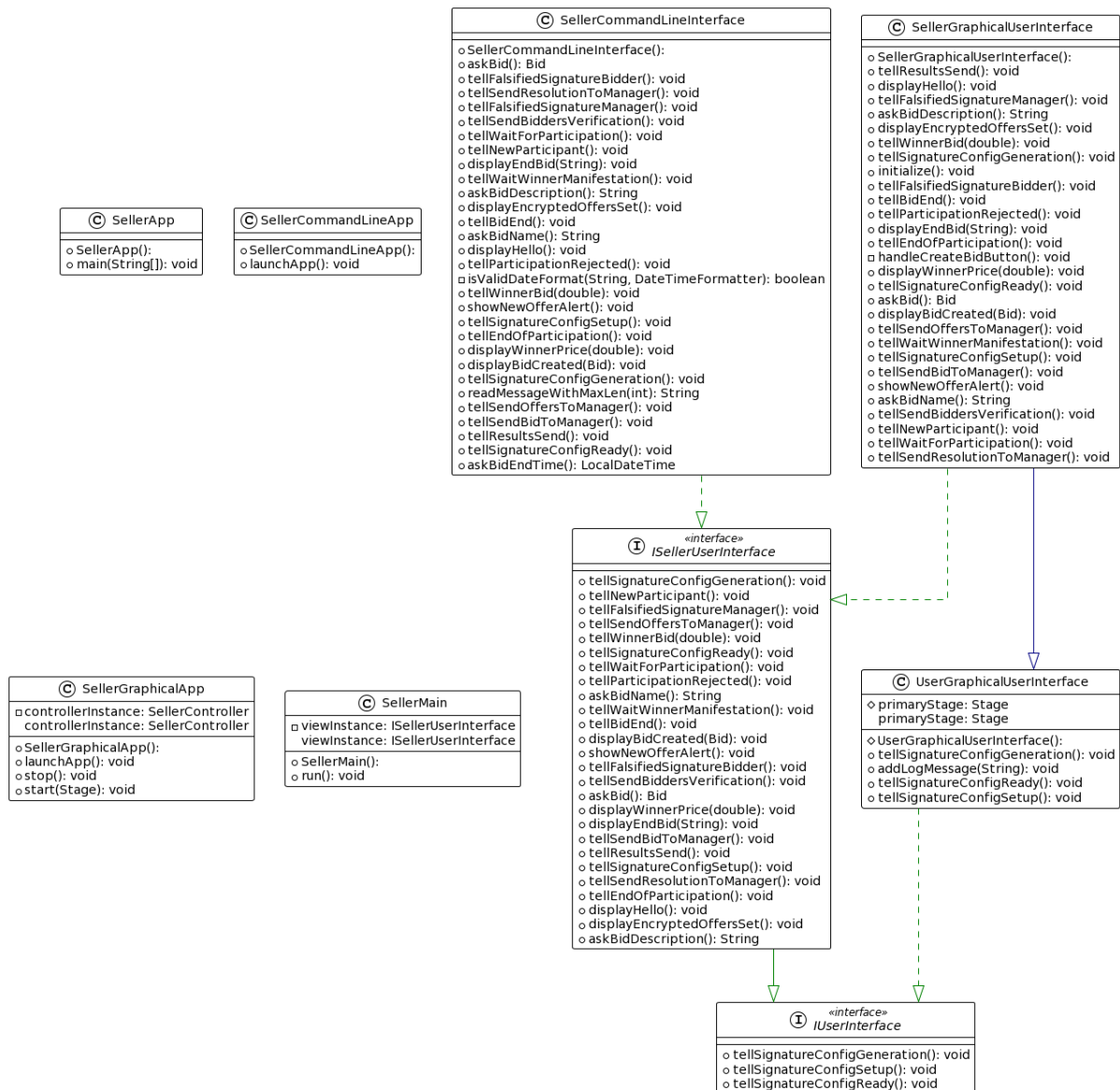


Figure 10 - Diagramme de classe du package `com.projetenchere.common.view` et `com.projetenchere.seller.view`



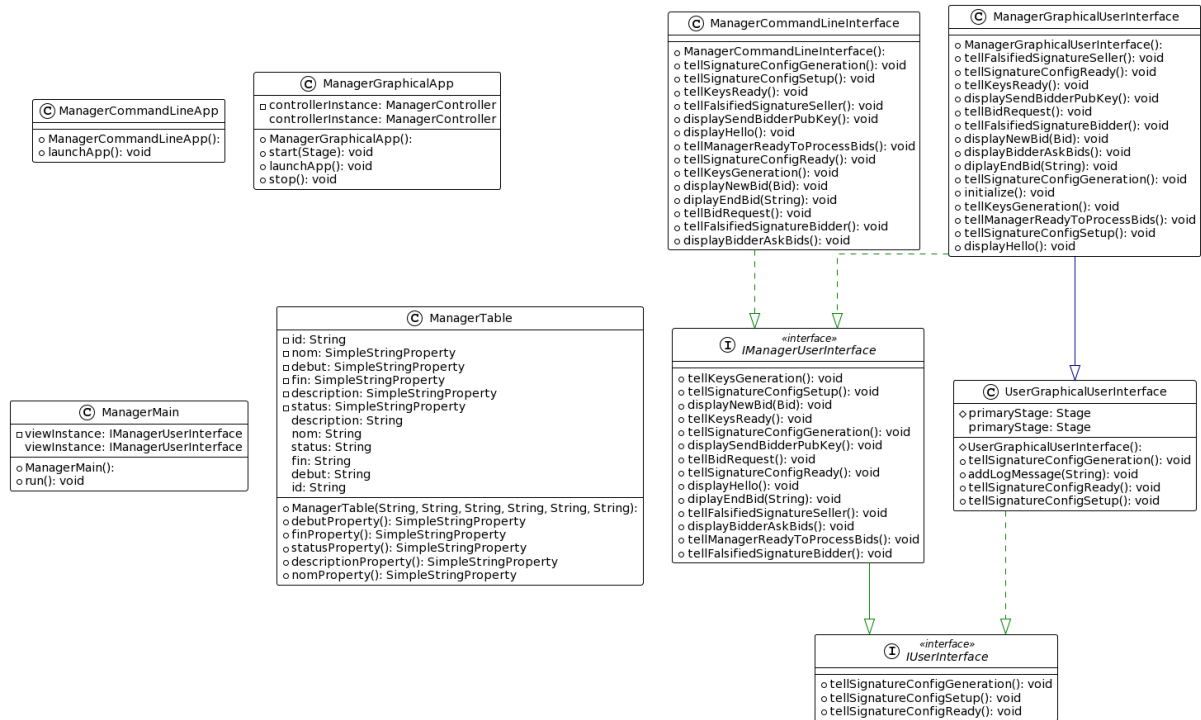


Figure 11 - Diagramme de classe du package `com.projetenchere.common.view` et `com.projetenchere.manager.view`

## 2.2 Réalisation

Le développement s'est déroulé principalement sans problème majeur. Des difficultés ont cependant été rencontrées.

Les principaux ralentissements dans la réalisation ont été rencontrés en mettant en place la synchronisation des différents acteurs avec le vendeur et dans la gestion de l'interface graphique.

### 2.2.1 Réseau

Pour le gestionnaire, tous les échanges sont gérés par les handlers du serveur, chacun indépendants, permettant un développement modulaire et fluide. De manière analogue, l'enchérisseur utilise seulement le client, nécessitant seulement de juxtaposer les appels les uns après les autres, tels que décrits par le diagramme de séquence.

Le vendeur, quant à lui, utilise à la fois un serveur et un client: l'un pour gérer les requêtes des enchérisseurs, l'autre pour faire des demandes au gestionnaire, typiquement pour enregistrer ou résoudre une enchère.

Cela implique de synchroniser plusieurs éléments: il faut retirer et ajouter des handlers en fonction de l'avancement de l'enchère, attendre au sein même de ces handlers que certaines étapes soient atteintes, et d'autres obstacles similaires.

L'application utilise de simples attributs booléens et des boucles while pour se synchroniser. Cela ajouté aux nombreux threads concurrents a rendu l'implémentation et le débogage difficiles.

À posteriori, un moyen plus formel d'assurer la bonne succession des étapes de l'enchère aurait été nécessaire. Cela aurait permis d'obtenir un aperçu plus clair sur le déroulé du programme, ce qui manquait cruellement lors de l'implémentation.

### 2.2.2 Interfaces

La coexistence de la CLI et de la GUI est l'autre obstacle qui a beaucoup ralenti le développement. L'appel à la méthode de lancement de la thread GUI est bloquant, et il réagit mal avec les autres threads.

Durant les premières itérations, la CLI a été plus ou moins retirée du code, jusqu'à ce qu'elle devienne à nouveau nécessaire quand SecureWin Manager a commencé à être conteneurisé.

La solution que nous avons trouvée est de faire en sorte que la CLI se comporte similairement à la GUI et non l'inverse. Au moment où le loader appelle la CLI, une thread similaire à celle du GUI est lancée et peut recevoir des appels.

## 2.3 Résultats

### 2.3.1 Tests

Des tests ont été mis en place pour garantir la stabilité du système d'enchères, et aussi et surtout la sécurité. Ces tests sont un témoin du bon fonctionnement de l'application en réussissant.

Des vérifications sont mises en place pour vérifier que le chiffrement s'exécute correctement, i.e. le chiffré est effectivement indéchiffrable sans la clé, il se déchiffre correctement, etc.

Il en va de même pour les signatures : les objets signés doivent être vérifiables et cette vérification doit échouer si la signature est falsifiée et réussir si elle est conforme. La lecture et l'enregistrement des signatures doivent se comporter comme attendu.

Pour ce qui est des enchères en elles-mêmes, une suite de tests s'assure qu'elles sont bien créées, qu'elles se lancent et s'arrêtent au moment adéquat, et que le gestionnaire les résout correctement.

### 2.3.1 Manuels

Des manuels d'installation et d'utilisation sont disponibles.

### 3. Gestion de projet

Dans un premier temps, nous aborderons la gestion d'équipe, en détaillant nos processus de planification des tâches, de répartition des responsabilités et d'utilisation d'outils de communication. Ensuite, nous discuterons de notre méthodologie de développement agile SCRUM et des outils tels que nous avons adoptés. Enfin, nous dresserons un bilan critique de notre gestion de projet, mettant en évidence nos points forts, nos faiblesses et les leçons apprises pour l'avenir.

#### 3.1 Gestion d'équipe

Premièrement, à la réception de la note d'intention du client, nous nous sommes réunis pour échanger autour des nouveaux besoins pour l'application. Par la suite, nous avons effectué une liste de tâches sous forme de tableau avec l'outil Google Docs. Cet outil nous a permis de répartir les tâches, en se basant à la fois sur notre champ de compétences et sur les préférences de chacun et chacune.

Mission et sous-tâches	Responsables	État
<b>Restructurer votre application en utilisant les bonnes pratiques de conception</b>	@?	En cours
Restructurer les classes du package Encrypted pour respecter le principe de ségrégation des interfaces ( <a href="https://mgasquet.github.io/R304-QualiteDeveloppement/assets/TP3/JSP2.svg">https://mgasquet.github.io/R304-QualiteDeveloppement/assets/TP3/JSP2.svg</a> )	@Katia	Implémenté (A test...)
Conception sur starUML	@Katia	Implémenté (A test...)
		Pas commencé
<b>Implémenter des connexions sécurisées entre des machines connectées sur un réseau (si ce n'est pas fait)</b>		Terminé
<b>Mettre en œuvre des tests unitaires à grande échelle et de tests d'intégration (S1 &amp; S2)</b>	@Loan	En cours
		Pas commencé
<b>Finaliser l'interface graphique (S1 &amp; S2)</b>	@Yûki	Terminé
Préparer une maquette (avoir une meilleure idée de comment doit être l'interface de l'application)		Pas commencé
<b>Modifier le protocole d'enchères en remplaçant l'algorithme de chiffrement utilisé pour chiffrer les mises à prix. (S1)</b>	@Paul	En cours
		Pas commencé
<b>Analyser le nouveau protocole et trouver un ensemble de paramètres adéquats pour son implémentation. (S1)</b>	@Katia	En cours
Vendeur : Produit des chiffrés c	@Katia	Implémenté (A test...)
Vendeur calcule ensuite sa signature z du chiffré c et définit C comme l'ensemble des chiffrés {ci}i=1,N	@Katia	Implémenté (A test...)
Vendeur : Transmet (C,c,z) aux enchérisseurs.	@Katia	Implémenté (A test...)
Vendeur : S vérifie la signature du gestionnaire et publie p2 avec sa propre signature et celle de A.	@Katia	Implémenté (A test...)
Vendeur : Attends un gagnant.		Implémenté (A test...)
Enchérisseur : Obtient la liste précédemment mentionnée, vérifie la signature et que son chiffré appartient bien à C.	@Katia	Implémenté (A test...)
Enchérisseur : et, le cas échéant, fait remarquer l'absence de son chiffré.	@Katia	Implémenté (A test...)
Enchérisseur se manifeste s'il gagne	@Katia	Implémenté (A test...)
Enchérisseur : L'enchérisseur qui a offert un prix plus important se manifeste pour être déclaré vainqueur.	@Katia	Implémenté (A test...)

Figure 12 - Capture d'écran de la liste de tâches du semestre 4

L'application de messagerie instantanée, Discord, nous a permis de créer un serveur dédié au projet afin de faciliter les échanges et le suivi sur la progression du projet. Ainsi,

nous effectuions des sessions de développement en groupe, d'environ 4h à 5h, le samedi ou le dimanche, en télétravail en visioconférence sur ce serveur Discord. En semaine, chaque membre de l'équipe travaillait sur la tâche qui lui était assignée, nous profitions de nos pauses entre nos cours pour échanger sur l'avancement de nos tâches respectives. Nous prenions le temps dans la semaine pour faire des réunions, nommées "Stand up meeting", afin de faire le point sur nos avancements, les difficultés rencontrées et aussi sur le bien-être de chaque membre de l'équipe, le tout était résumé dans un rapport de réunion rédigé par le Scrum Master, Yûki SALA-MOCHIZUKI. Tout ceci nous a permis d'assurer une bonne communication au sein du groupe, et une autonomie à chaque membre de l'équipe, mais de tout de même avoir des moments pour échanger et s'entraider.

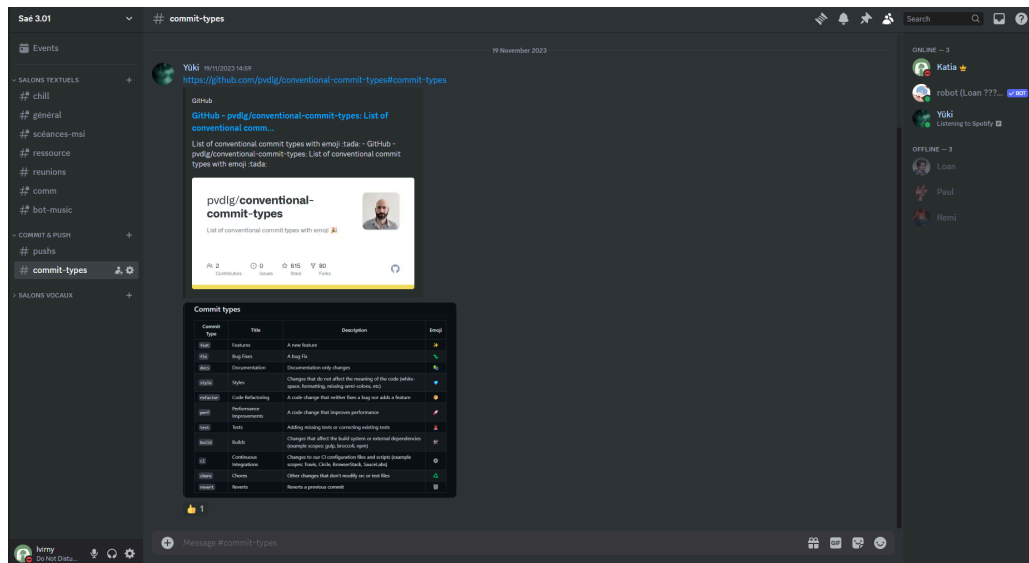


Figure 13 - Capture d'écran serveur discord

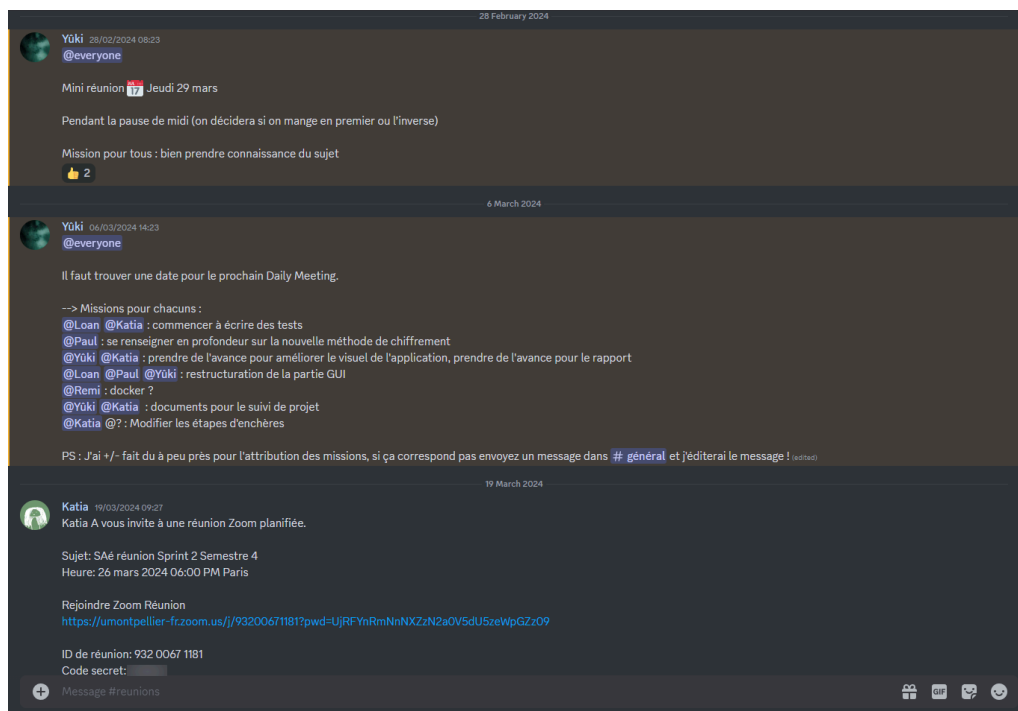
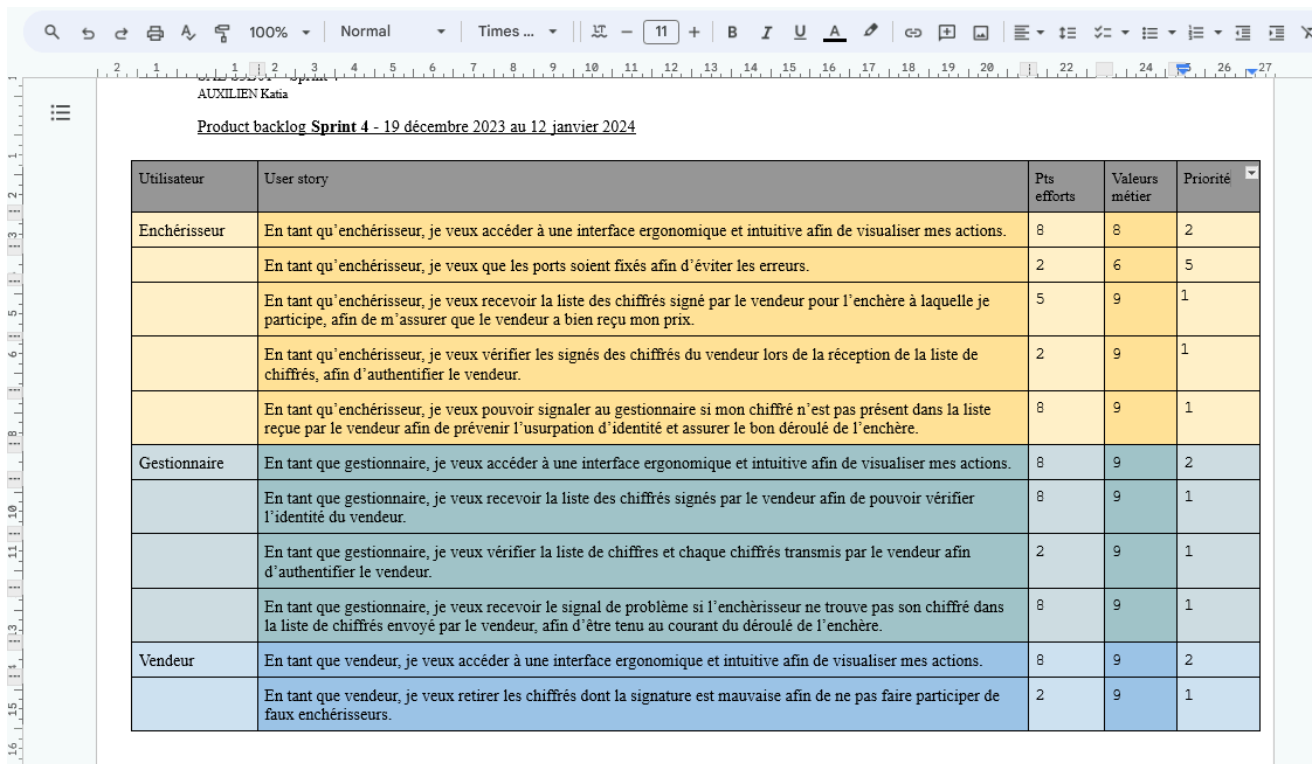


Figure 14 - Capture d'écran salon réunion du serveur discord

Nous avons pris l'initiative de contacter le client par mail à chaque début de sprint, afin de connaître ses besoins pour le sprint en cours, et de prendre un rendez-vous si nécessaire, pour échanger autour des points qui n'étaient pas clairs pour l'équipe. C'est la product owner Katia AUXILIEN qui était chargée de fluidifier le dialogue entre le client et l'équipe, de représenter le client auprès de l'équipe de développement dont elle fait partie, et de représenter l'équipe auprès du client, notamment pour fournir une réponse au client sur la faisabilité de certaines fonctionnalités.

Avant de développer, nous effectuions beaucoup de recherches et de consultation dans la documentation des outils que l'on voulait et devait utiliser.

En plus des réunions, pour assurer le suivi de chaque tâche, nous avions à notre disposition de nombreux outils. Notre équipe suivant une démarche agile, la Product Owner établissait des "Users Story", qui est une description d'exigences pour toute fonctionnalité ou "tâche" nécessaire au fonctionnement du produit en développement. Lors des premières réunions, avait lieu le "planning poker", qui est une méthode pour attribuer en équipe des points d'efforts nécessaire pour valider une User Story. Après ce planning poker, la Product Owner rédigeait les tests d'acceptation des Users Story dans le Product Backlog et les publiait sur GitLab avec l'outil "Issue".



Utilisateur	User story	Pts efforts	Valeurs métier	Priorité
Enchérisseur	En tant qu'enchérisseur, je veux accéder à une interface ergonomique et intuitive afin de visualiser mes actions.	8	8	2
	En tant qu'enchérisseur, je veux que les ports soient fixés afin d'éviter les erreurs.	2	6	5
	En tant qu'enchérisseur, je veux recevoir la liste des chiffrés signé par le vendeur pour l'enchère à laquelle je participe, afin de m'assurer que le vendeur a bien reçu mon prix.	5	9	1
	En tant qu'enchérisseur, je veux vérifier les signés des chiffrés du vendeur lors de la réception de la liste de chiffrés, afin d'authentifier le vendeur.	2	9	1
	En tant qu'enchérisseur, je veux pouvoir signaler au gestionnaire si mon chiffré n'est pas présent dans la liste reçue par le vendeur afin de prévenir l'usurpation d'identité et assurer le bon déroulé de l'enchère.	8	9	1
Gestionnaire	En tant que gestionnaire, je veux accéder à une interface ergonomique et intuitive afin de visualiser mes actions.	8	9	2
	En tant que gestionnaire, je veux recevoir la liste des chiffrés signés par le vendeur afin de pouvoir vérifier l'identité du vendeur.	8	9	1
	En tant que gestionnaire, je veux vérifier la liste de chiffres et chaque chiffrés transmis par le vendeur afin d'authentifier le vendeur.	2	9	1
	En tant que gestionnaire, je veux recevoir le signal de problème si l'enchérisseur ne trouve pas son chiffré dans la liste de chiffrés envoyé par le vendeur, afin d'être tenu au courant du déroulé de l'enchère.	8	9	1
Vendeur	En tant que vendeur, je veux accéder à une interface ergonomique et intuitive afin de visualiser mes actions.	8	9	2
	En tant que vendeur, je veux retirer les chiffrés dont la signature est mauvaise afin de ne pas faire participer de faux enchérisseurs.	2	9	1

Figure 15 - Capture d'écran du product backlog sur GoogleDoc du semestre 3, sprint 4

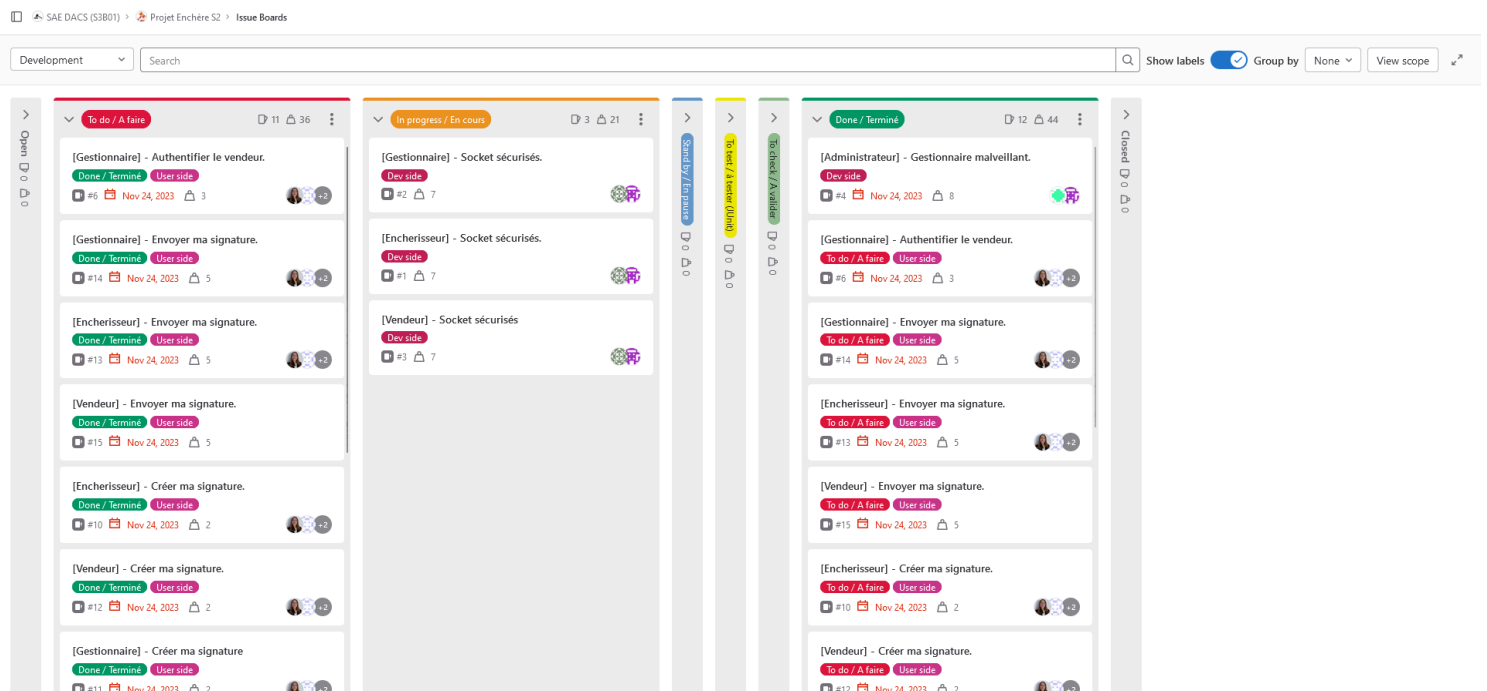


Figure 16 - Capture d'écran de l'issueboard sur GitLab du semestre 3, sprint 2

En environnement collaboratif pour le code, nous avons utilisé **GitLab**, permettant un versionnement et un suivi du code durant toute la durée du projet. Ainsi, nous avons utilisé la méthode **Feature branching**. Le projet était structuré par une branche principale, et chaque nouvelle fonctionnalité était sur une branche. Une branche se base sur la branche principale pour sa création. Lorsqu'un membre de l'équipe commence le développement d'une fonctionnalité, il crée une nouvelle branche, et lorsque le développement de cette fonctionnalité est terminé et testé, cette branche fusionne avec la branche principale (merge). Cela permet d'assurer que le contenu de la branche principale est conforme aux tests et validé par l'équipe.

Durant le développement, nous avons utilisé comme environnement de développement IntelliJ IDEA sous licence étudiante. Cet IDE permettant d'utiliser git à travers son interface, cela a facilité nos "commits" et le suivi des "branches" de notre projet. De plus, cela a permis de faciliter le respect des bonnes pratiques de développement et de créer une arborescence de projet structurée.



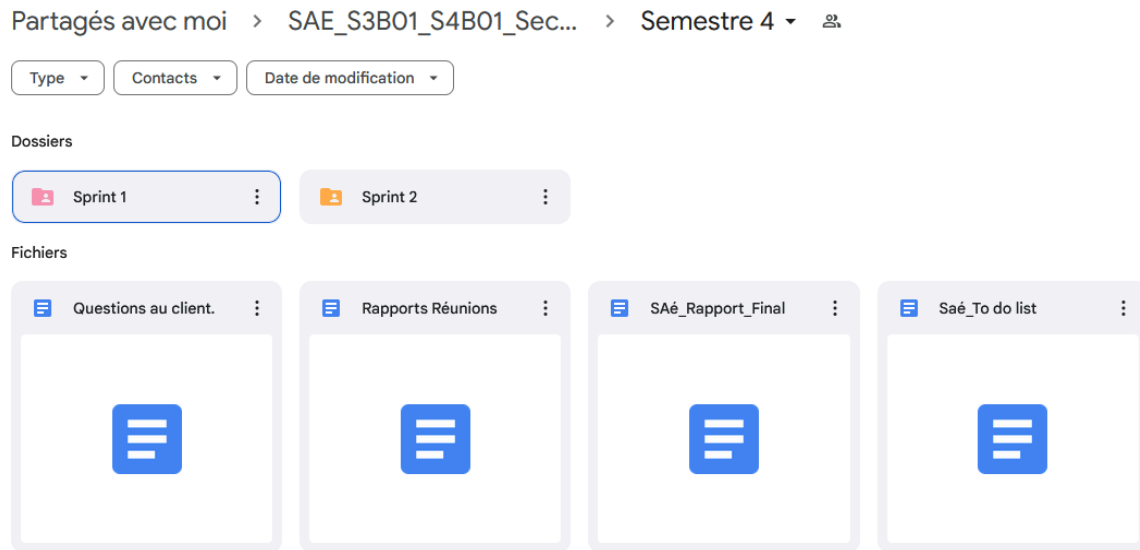


Figure 17 - Capture d'écran GoogleDrive de la SAE

De plus, dès le début du projet, nous avons créé un répertoire Google Drive pour centraliser tous ces documents, classés par sprint, et par phase dans le développement du projet. Dans la partie Analyse, nous centralisons les diagrammes de cas d'usages que nous réalisons au début du sprint. La partie Conception contient les diagrammes de classes repris du sprint précédent pour améliorer l'architecture du logiciel en fonction des nouveaux besoins clients. Enfin, le dossier suivi contenait les graphiques et autres notes utiles.

## 3.2 Méthode de développement et outils

Comme précisé auparavant, le développement du projet a suivi la méthodologie agile SCRUM. Nous l'avons choisie pour son adaptabilité, en effet, elle est conçue pour être adaptable aux changements. Les équipes peuvent rapidement réagir aux retours et aux nouveaux besoins du client, ce qui permet de mieux répondre aux exigences changeantes du projet. Cette méthode encourage la transparence à tous les niveaux. De ce fait, nous avons tenu des réunions régulières qui ont permis de discuter de l'avancement, des obstacles et des plans futurs, ce qui favorise la confiance et la collaboration.

La méthodologie Scrum divise le projet en itérations appelées "sprints", généralement de 1 à 4 semaines. À la fin de chaque sprint, une version potentiellement livrable du produit est présentée, ce qui permettait au client de voir rapidement les progrès et de fournir des commentaires. La méthodologie Scrum intègre des mécanismes de rétroaction réguliers, tels que les revues de sprint et les rétrospectives, qui nous ont permis de réfléchir à notre fonctionnement et d'identifier des moyens d'améliorer nos processus.

Enfin, Cette méthodologie nous a encouragé dans l'auto-organisation des équipes et favorisait notre engagement envers le succès du projet. Chaque membre de l'équipe étaient responsables de la planification de leur travail et étaient encouragés à prendre des décisions pour atteindre les objectifs du sprint.

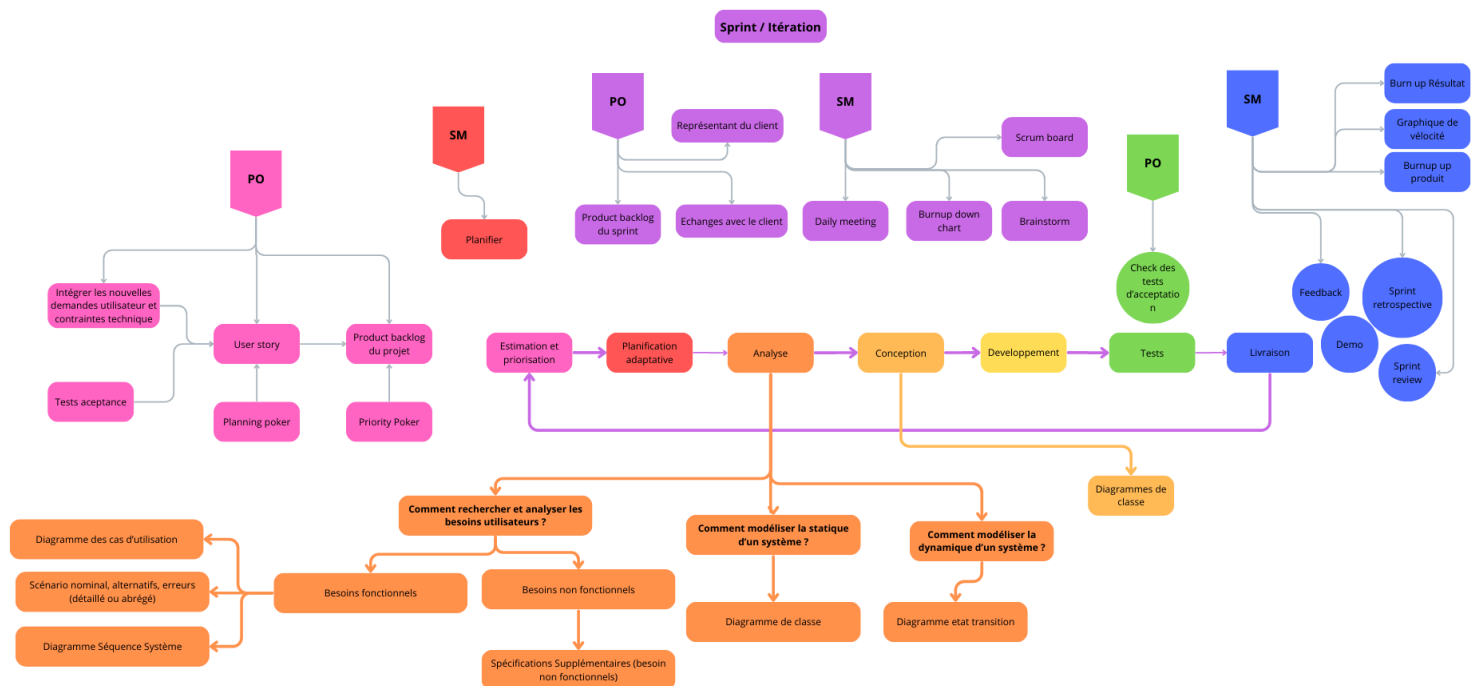


Figure 18 - Détails d'un sprint en développement itératif.

Comme précisé précédemment, nous utilisons le Product Backlog et l'outil Issue de GitLab pour suivre nos Users Story. GitLab permettant de planifier une deadline à chaque issu, cela nous a permis de nous organiser en priorisant les Users Story qui avaient le plus d'importance pour le client. De plus, lors du quatrième semestre nous avons utilisé une To Do List maintenue par la Product Owner.

Au cours du projet, le Scrum Master a mis en place différents types de graphiques (burndown chart de Sprint, burnup chart de résultat, graphique de vélocité, burnup chart de production) permettant de refléter la progression et les performances de l'équipe dans le cadre de la méthodologie SCRUM.

Le burndown chart de Sprint fournit une visualisation claire de la quantité de travail restant à accomplir durant le sprint en cours par rapport au temps restant. La figure attachée reflète spécifiquement la progression du Sprint quatre, dans laquelle on peut détecter une chute du nombre de points d'efforts vers les derniers jours du Sprint. Cela peut aider l'équipe à identifier des tendances, à apprendre des erreurs et à ajuster ses méthodes de travail pour les Sprints suivants.



cohérence suggère une bonne compréhension et une réponse efficace aux attentes du client pendant ces périodes. Cependant, lors du deuxième sprint, un écart significatif est observable, révélant ainsi une divergence par rapport aux attentes du client. Le graphique de vélocité reflète l'importance de la compréhension des besoins du client.

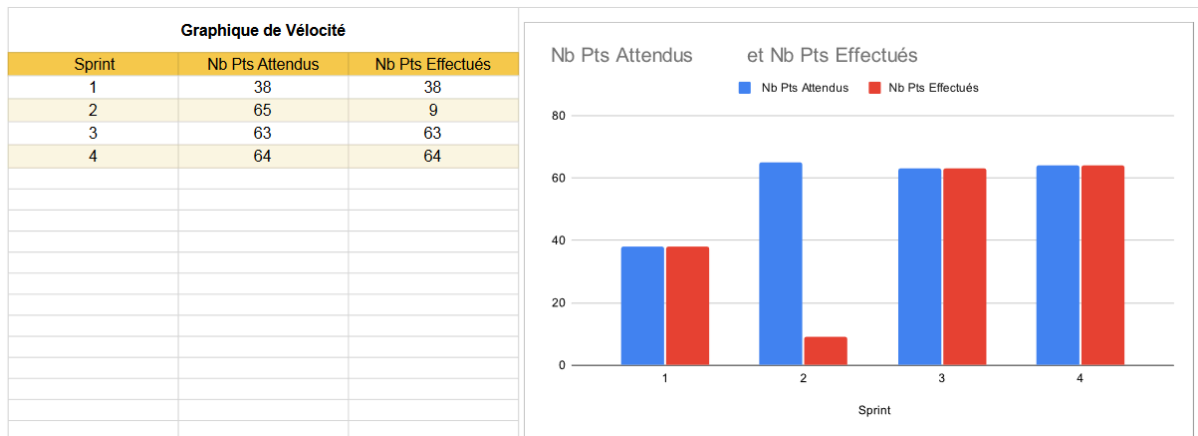


Figure 21 - Graphique de vélocité du quatrième Sprint.

Enfin, le burnup chart de production offre une vue d'ensemble de la progression de la livraison des fonctionnalités dans le temps. Cela permet à l'équipe de suivre l'évolution du projet par rapport aux objectifs fixés et d'identifier les éventuels retards ou défis à surmonter.

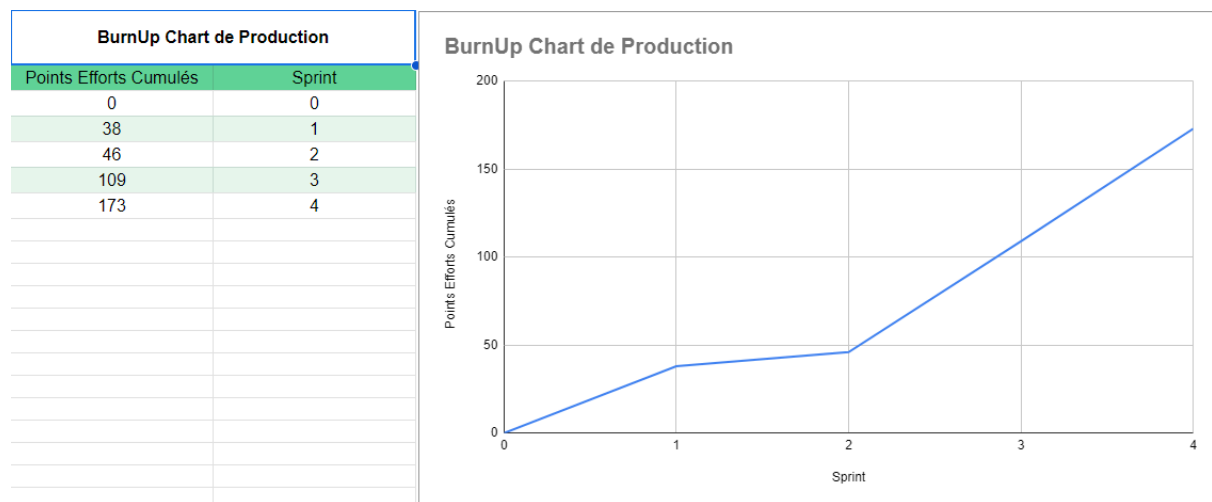


Figure 22 - BurnUp Chart de Production du quatrième Sprint.

Ces graphiques ont été essentiels pour évaluer la performance de l'équipe, identifier les tendances et les problèmes potentiels, et ajuster les stratégies en conséquence. En les utilisant, l'équipe a pu prendre des décisions éclairées tout au long du projet, favorisant ainsi la transparence, la collaboration et l'efficacité.

### 3.3 Bilan critique

Par ailleurs, à chaque fin de sprint nous avons effectué un bilan du sprint réalisé. Nous faisons la liste des difficultés rencontrées au sein de l'équipe, nous avons identifié l'origine de ces difficultés et fait le point sur les éventuels conflits et points faibles au sein de l'équipe. Nous avons aussi listé nos points forts et nos bonnes pratiques durant le sprint. Ces revues de sprints ont permis de nous améliorer car nous trouvions des solutions afin que ces difficultés rencontrées ne persistent pas, nous avons aussi précisé qu'il fallait continuer certaines pratiques pour conserver nos points forts dans cette démarche d'amélioration.

De manière générale, durant les sprints du troisième semestre et quatrième semestre nous avons eu des difficultés à démarrer les débuts de sprint correctement par manque de temps avec nos cours, cela provoquait une obligation de s'empresseur sur certaines étapes du projet ce qui peut avoir affecté la qualité de notre travail.

De plus, au sein de l'équipe il y avait une mauvaise répartition des tâches et certains membres se sont plus investis ou étaient obligés de faire plus de tâches que d'autres ce qui provoquait un déséquilibre dans la compréhension du projet pour certains membres de l'équipe. Pour nous améliorer, nous avons pris différentes mesures :

- Assurer une communication transparente sur les modifications apportées au code de l'autre, favorisant ainsi l'apprentissage au sein de l'équipe.
- Mieux énumérer nos intentions pour éviter des malentendus similaires à l'avenir.
- Avoir une meilleure répartition des tâches.
- S'assigner des tâches à partir de l'analyse jusqu'aux tests.
- Faire une To Do list

Lors du deuxième sprint, nous n'avons pas assez communiqué avec le client ce qui a engendré une mauvaise compréhension de l'objectif principal du sprint qui était l'implémentation des signatures cryptographiques. Il a fallu près d'une semaine et demie pour cerner clairement l'intention du client. Cette période initiale a été un défi pour nous. Nous avons décidé par la suite de prendre régulièrement des rendez-vous avec le client pour être sûr d'aller dans la bonne direction.

Nous nous sommes améliorés concernant les démonstrations de fin de sprint au client. En effet, lors du premier sprint nous étions dans le flou sur les attendus des démonstrations de sprint. Pour nous améliorer nous avons pris en compte les retours de nos professeurs :

- Nous avons décidé de ne pas s'engager vis-à-vis du client pour la suite du projet sans en discuter au préalable en équipe.
- Contextualiser la présentation pour permettre au client de mieux comprendre le travail accompli.
- Préparer des supports de présentation tels que des slides ou un résumé pour faciliter la compréhension.
- Inclure les diagrammes que nous avons effectués pour une communication visuelle plus efficace.

- Mieux répartir la parole lors de la présentation pour permettre à chaque membre d'avoir un temps de parole équilibré.

Parmi les points positifs, nous avons remarqué que les membres de l'équipe se sont fait confiance mutuellement tout au long des sprints, nous avons su résoudre nos conflits au début du projet ce qui n'a pas impacté la suite du développement.

En bref, tout au long du projet, nous avons progressé en adoptant une approche flexible et en tirant des leçons de nos expériences. Nous avons mieux géré notre emploi du temps, amélioré la communication en équipe, et affiné nos interactions avec le client. En résolvant les problèmes dès leur apparition, nous avons renforcé la confiance mutuelle et maintenu un environnement de travail positif. Ces ajustements continus ont été essentiels pour la réussite du projet et nous ont préparés à relever de nouveaux défis à l'avenir.

# Conclusion

Dans notre projet de développement des applications SecureWin, nous avons poursuivi plusieurs objectifs définis dès le départ. Tout d'abord, nous cherchions à créer une plateforme d'enchères électroniques sécurisée et conviviale, répondant aux besoins du client tout en respectant les normes de sécurité. Ensuite, nous nous sommes engagés à mettre en place une architecture logicielle robuste et évolutive, permettant une réalisation itérative et une gestion efficace du projet. Enfin, nous avons visé à fournir une interface utilisateur intuitive et agréable, favorisant une expérience fluide et satisfaisante pour tous les utilisateurs de l'application.

Grâce à notre collaboration et à notre engagement, nous avons réussi à concrétiser ces objectifs avec succès. SecureWin est désormais une application fonctionnelle, offrant un environnement sécurisé pour les enchères électroniques et garantissant la confidentialité des données des utilisateurs. L'architecture logicielle mise en place permet une évolutivité aisée et une maintenance simplifiée, tandis que l'interface utilisateur épurée offre une navigation intuitive et une expérience agréable à chaque interaction.

Pour l'avenir de SecureWin, nous prévoyons d'intégrer des fonctionnalités de gestion des mots de passe pour renforcer la sécurité des données des utilisateurs. De plus, nous envisageons de déployer l'application sur plusieurs systèmes d'exploitation, notamment sur Windows, pour élargir sa portée. Enfin, nous chercherons à améliorer l'interface graphique de l'application afin d'offrir une expérience utilisateur encore plus fluide et intuitive.

En conclusion, ce projet nous a apporté une multitude d'enseignements sur le plan technique et humain. Sur le plan technique, nous avons acquis une expertise approfondie en matière de développement logiciel, de sécurité des données et d'architecture système. Nous avons surmonté divers défis techniques tout au long du processus de développement, ce qui nous a permis de renforcer nos compétences et notre compréhension des concepts clés. Sur le plan humain, nous avons appris l'importance de la collaboration, de la communication et de la gestion efficace du temps. En travaillant ensemble, nous avons pu surmonter les obstacles et atteindre nos objectifs avec succès, ce qui témoigne de notre engagement et de notre détermination à réussir. En fin de compte, ce projet nous a préparés à relever de nouveaux défis dans notre parcours professionnel et nous a enrichis tant sur le plan personnel que professionnel.



# Bibliographie

<https://stocklear.fr/conseil/les-7-meilleurs-sites-d-encheres-en-ligne>  
<https://geekflare.com/fr/online-auction-software/>

# Annexes techniques

## Annexe 1 - Système de Damgård-Jurik

### KeyGen

1. Choisir  $p$  et  $q$  deux nombres premiers de 2048 bits chacun et calculer  $N = p \times q$ .
2. Calculer  $\varphi(N) = (p - 1) \times (q - 1)$ .
3. Calculer  $d = \varphi(N) - 1 \bmod Ns$ .
4. La clé publique est  $pk = Ns$  et la clé secrète est  $sk = (p, q, \varphi(N), d)$  pour un entier  $s \geq 1$  de votre choix.

### Encrypt( $m, pk$ )

1. Tirer  $r \in (\mathbb{Z}/Ns+1\mathbb{Z})^*$
2. Calculer  $c = (1 + N)^m \times r^{Ns} \bmod Ns+1$
3. Le chiffre est  $c$ .

### Decrypt( $c, sk$ )

Pour décrire le déchiffrement, commençons par définir la fonction  $L(b) = (b - 1)/N$  pour un entier  $b$ .

1. Calculer  $M = c\varphi(N) \bmod Ns+1$ . La valeur de  $M$  est alors  $(1 + N)^{m \times \varphi(N)} \bmod Ns+1$  (vous pouvez le vérifier sur des exemples). Il faut donc arriver à extraire  $m$ .
2. Il s'agit alors d'appliquer l'algorithme suivant, que vous pouvez trouver dans l'article "A Generalisation, a Simplification and some Applications of Paillier's Probabilistic Public-Key System", par Ivan Damgaard et Mads Jurik, publié à PKC en 2001.

### Algorithm 1:

Input:  $M, N, s$

Result:  $m \times \varphi(N) \bmod Ns$

```
i := 0;
for j := 1 to s do
  t1 := L(M mod N^(j+1)) ;
  t2 := i ;
  for k := 2 to j do
    i := i - 1 ;
    t2 := t2 × i mod N^j ;
    t1 := t1 - t2 × N^(k-1) / (k! mod N^j) ;
  end
  i := t1 ;
end

return i
```