Compte Rendu du projet ECOM

• Jahna (chef de projet) • Aurélien (scrumaster) • François-Xavier • Morgane

Septembre 2021

Table des matières

1 Introduction

Pendant trois semaines, nous avons travaillé sur le projet ECOM qui ouvre notre master Génie Informatique. Nous avons choisi de concevoir un site de vente de vins appelés la Piquette dédié aux bons vins français. Nous n'avons pas été sponorisés par des lobby mais apprécieront tout investissement financier.

Nous hébergeons notre projet sur Git et le déployons via Heroku.

Ce projet a pour but de mettre en place des transactions avec une gestion des images et des stocks. Il s'appuie en grande partie sur JHipster et l'autogénération d'application. Une grande partie de notre travail était de compléter les appels du front (service web) au back (service de gestion de données) pour mettre en place de nouvelles fonctionnalités et sélectionner les informations pertinentes à garder ou à mettre en place pour avoir un site web efficace.

2 Partie Système

2.1 Architecture et technologies choisies

Les technologies ont été choisies tôt : nous avons opté pour Angular et Typescript sur le *frontend* qui ont des systèmes performants d'analyse pré-compilation perfomant et étaient déjà maitrisés par une des membres de l'équipe. Et pour le *backend*, nous utilisons Java avec Spring Boot, JPA, REST et Maven.

L'achitecture a été générée par Jhipster et nous l'avons respectée au maximum car elle est claire. Cela a pourtant été une difficulté supplémentaire lors de la prise en main car l'architecture nous était inconnue, la profusion de fichiers rendait l'ensemble confus et plusieurs fonctionnalités étaient difficilement lisibles et compréhensibles.

Ce que nous appelons *backend* est la partie Java dédiée à la gestion des données en particulier vis à vis de la base de données. Cette partie a plusieurs couches, dans plusieurs *packages* qui correspondent chacun à un niveau de profondeur. La couche la plus basse traite directement avec la base de données alors que la couche la plus haute implémente une API REST qui permet d'offrir une communication efficace via les requêtes HTTP et HTTPS.

Le *frontend* en complément permet de gérer la partie entièrement dédiée au site web ; cela inclut le routage des pages, leur design et esthétique, les informations communiquées, et la configuration des interfaces utilisateur (ce qu'elles font, quelles requêtes sont appelées, comment sont traités les retours)...

Finalement pour la base de données nous avons choisi d'utiliser MongoDB qui est en NoSql. Nous avons fait ce choix d'une part pour consolider nos connaissances en NoSql et d'autre part car le modèle s'y prêtait car il y a peu de liens entre les tables.

Nous avons donc dû faire plusieurs essais avant d'avoir un projet JHipster fonctionnel.

2.2 Modèle de données

Nous présentons le modèle à la figure 1 .

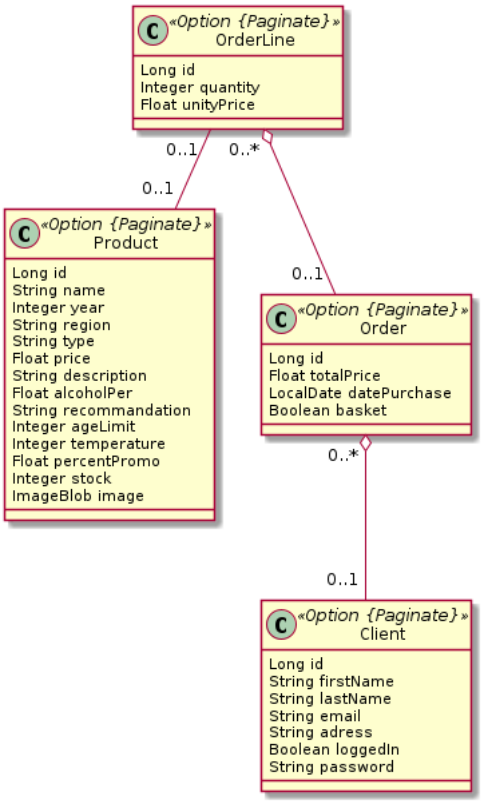

FIGURE 1 – Modèle de données via JHipster JDL

Le modèle est composé d'objets à stocker en base de données. Nous avons définis quatre types. La classe Order représente une commande ou un panier (définis suivant le champ *basket*, un boolean mis à true ou false). Elle contient en plus de cela des informations sur le prix total de la commande et la date d'achat.

Les OrderLine représentent les différents produits présents dans une commande (une ligne de la facture). Il y a donc un OrderLine par produit ajouté au panier. Celui-ci contient la quantité du produit commandé et le prix unitaire de celui-ci.

Nous avons aussi créé la table Client qui est un utilisateur possédant un compte. Un Client contient des informations de base sur l'utilisateur lui permettant de se connecter, de lui envoyer une facture par mail et de lui faire livrer les produits achetés.

Enfin, nous avons une classe Product qui correspond aux vins que nous vendons sur le site. Celle-ci contient de nombreuses informations sur le vin, comme son année de fabrication, son nom, ses recommandations de dégustation mais aussi son stock actuel.

2.3 Fonctionnalités

Notre site étant un site de vente de vins, nos fonctionnalités permettent de réaliser toutes les étapes allant de la recherche d'un vin à l'achat de celui-ci pour un client. Notons qu'il est possible de constituer des paniers avant de créer un compte, ceci sont stockés en cache et non en base. Nous avons aussi travaillé sur le côté administrateur afin de permettre à celui- ci d'ajouter de nouveaux produits, de modifier un produit mais aussi de gérer les stocks de ses produits.

Les fonctionnalités présentes sont donc les suivantes :

- consulter l'ensemble des produits disponibles à la vente
- trier et rechercher les produits
- acheter des vins en diverse quantité
- créer un panier, le stocker et l'enregistrer
- ajouter et modifier un produit
- changer le stock d'un produit
- créer un compte client/admin

Les fonctionnalités en cours de réalisation (non abouties) :

- Vérification du panier avant paiement (vérifier que les prix sont conformes au prix courant)
- Non-chargement du dernier panier en cas de changement de client
- Non-suppression des filtres sélectionnés après leur affectation

2.4 Choix d'implémentation

2.4.1 Gestion des images

Afin de stocker les images de nos produits nous avons décidé de passer par la base de données. Nous stockons donc les images en base sous forme de tableau de byte car celles ci sont petites (maximum 5MB).

De plus étant donné que notre site ne vend que des vins français, le nombre d'articles sera relativement bas en pratique, en effet, le vignoble français produit 3240 vins différents. Le nombre d'images stockées en base sera donc relativement petit (maximum 3240) car nous n'avons choisi de ne permettre qu'une image par vin.

2.4.2 Gestion des transactions concurrentes

Concernant la gestion des stocks, nous avons choisi de mettre un verrou transactionnel sur la base seulement au moment de l'achat et non à la mise dans le panier.

Cela signifie qu'un produit n'est pas réservé par le client lorsqu'il le met dans son panier. Nous sommes conscient que cela peut apporter de la frustration à l'acheteur si celui-ci ne peut finalement pas commander un produit qu'il avait ajouté à son panier.

Pour contrer cela nous avons réfléchi à des améliorations futures tel que relancer un client quand le produit est proche de la rupture, ou de mettre un encart "dépêchez vous il ne reste plus que ... bouteilles" dans le panier si un des produits est sur le point d'être écoulé.

3 Partie IHM

Le fichier K-MADe .kxml est disponible dans le dépôt Git sous le dossier EIHM.

3.1 Procédure de conception

3.1.1 Persona

Thierry

Thierry, 35 ans, amateur de vins à ses heures perdues, il souhaite découvrir de nouveaux vins mais aussi en offrir à ses proches aussi passionés que lui. Il est donc prêt à s'acheter quelques bouteilles à moyen prix (de 20 à 40€) et occasionnellement à acheter une bonne bouteille ou deux (jusqu'à 150€) pour en faire cadeau à ses amis ou les faire decouvrir lors de diner chez lui.

Stéphanie

Stéphanie, 25 ans, ne connait pas encore le monde de l'oenologie mais souhaite découvrir celui-ci et être capable de choisir un bon vin pour en faire profiter sa famille et ses amis lorsqu'ils viendront manger chez elle.

3.1.2 Scénario

Scénario 1 : Thierry – Passionné de vins

Thierry souhaite acheter un bon vin à faire decouvrir à ses proches lorsque ils viendront manger la semaine prochaine. Un soir de semaine en rentrant il décide donc de se rendre sur la-piquette.herokuapp.com afin de trouver la bonne bouteille.

Pour cela il parcourt les différentes options. Il cherche un rouge qui accompagne bien de la viande rouge car c'est ce qu'il a choisi de cuisiner. Une fois le vin sélectioné il regarde le détail de celui-ci afin de se rendre mieux compte si le vin correspondra au repas. Une fois son choix fait il commande le produit après s'être connecté en habitué.

Scénario 2 : Stéphanie - Amatrice d'oenologie

Stéphanie aimerait découvrir quelques vins à présenter à ses proches mais ne connaissant que peu le vin, elle souhaite pouvoir etre conseillée et avoir facilement accès à des informations adaptées pour ses connaissances. Elle decide donc d'acheter une dizaine de bouteilles en ligne afin d'avoir du choix et de découvrir des goûts nouveaux.

Pour prendre le temps de bien choisir, elle décide de s'accorder quelques heures en fin d'après-midi pour fouiller le site et trouver ce qui lui correspond. Elle va tout d'abord chercher des recommandations avant d'affiner sa recherche par type de vins, rouge, blanc, rosé... Elle decide de prendre deux bouteilles de chaque en se référant aux recommandations et une bouteille un peu plus cher trouvée dans les promotions. Elle crée ensuite un compte avant de commander afin de pouvoir recommander facilement.

3.1.3 Modèle de tâches

A partir des deux scenarii, nous avons créé un modèle de tâches [1] sur K-MADe correspondant à l'action utilisateur "acheter du vin". Nous avons testé ce modèle avec ProtoTask pour vérifier l'enchainement des actions et garantir les possibilités des utilisateurs.

Avec plus de temps, nous aurions pu diversifier nos clients pour prendre en compte leur spécificité. Nous aurions aussi mis en place une même routine pour l'utilisateur administrateur et ses tâches les plus courantes.

3.1.4 Maquette du site

Nous avons aussi produit une maquette sur Figma pour délimiter les espaces de navigation et la manière de naviguer entre eux. Cette maquette n'a pas été entièrement suivie mais elle prenait notament en compte le fait d'ouvrir une boite de dialogue pour confirmer l'achat.

La maquette se concentrait sur l'expérience utilisateur client.

1. Voir Figure2 : Modèle de taches et fichier .kxml du dépôt

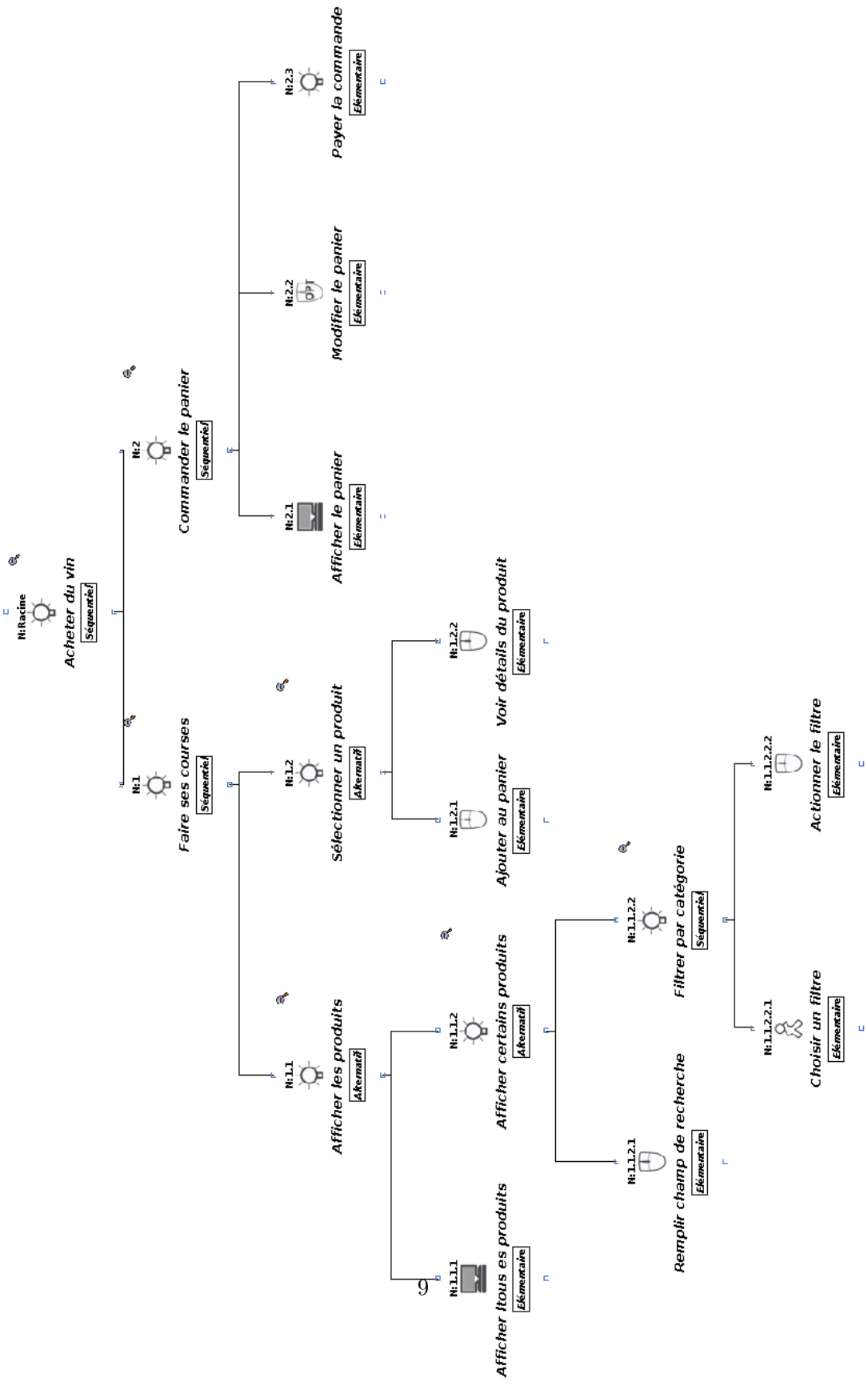

FIGURE 2 – Modèle de tâches sous K-MADe

3.2 Evaluation des heuristiques

3.2.1 Visibilité de l'état du système

Le système devrait toujours tenir informé l'utilisateur de ce qui se passe, en fournissant un « retour » (feedback) approprié, dans un temps raisonnable.

- **Comment est-elle bien mise en application ?** Beaucoup d'alertes sont déjà mises en place dans les cas suivants : modification du panier (ajout et retrait, depuit la liste produits ou depuis la vue du panier), paiement, modification d'un produit.
- **Comment est-elle mal mise en application ?** L'ajout dans le panier pourrait se faire avec une icône +1 sur le panier plutôt qu'un texte avec beaucoup d'informations inutiles.
- **Comment pourrait-elle être appliquée ?** Il n'y a pas d'état de transition ce qui serait utile dans les cas où les requêtes pourraient prendre un peu de temps.

3.2.2 Correspondance du système avec le monde réel

Le système devrait « parler » le langage de l'utilisateur, avec des mots, des phrases et des concepts qui lui sont familiers, plutôt que d'utiliser un langage propre au système. Suivre les conventions du monde réel, en faisant apparaître les informations dans une séquence naturelle et logique.

- **Comment est-elle bien mise en application ?** La clientèle est essentiellement française donc le français est la seule langue disponible. Le site reprend le modèle et les icones courantes des sites marchants.
- **Comment est-elle mal mise en application ?** Certaines icones sont manquantes (on a manqué de temps pour les incruster) et sont remplacés par un mot commun et parlant.

3.2.3 Liberté, contrôle de l'utilisateur

Les utilisateurs choisissent souvent par erreur des fonctions du système et ils ont besoin d'une « sortie de secours », clairement libellée pour quitter la fonction non désirée, sans qu'il y ait besoin de passer par de multiples dialogues pour le faire. Le système doit permettre d'annuler/refaire (undo/redo) une action

- **Comment est-elle bien mise en application ?** Toutes les pages ou presque proposent un bouton retour (libellé retour avec en icône une flèche vers la gauche). Il est possible notament de vider tous les filtres des produits d'un bouton.
- **Comment est-elle mal mise en application ?** On n'a pas de fonctionnalité refaire. Le paiement déclenche une action non-interruptible et non-annulable mais n'ouvre pas de dialogue pour en informer l'utilisateur et demander sa confirmation.
- **Comment pourrait-elle être appliquée ?** Ajouter des boites de dialogue pour confirmer le déclenchement d'action non-interruption (comme le paiement ou la mise en place d'une promo).

3.2.4 Cohérence et standards

L'utilisateur ne doit pas avoir à se poser des questions pour savoir si différents mots situations ou actions signifient la même chose. Suivre les conventions liées à la plate-forme.

— **Comment est-elle bien mise en application ?** Les seuls objets dont on parle sont les commandes, les paniers et les produits, aucun autre mot vient les remplacer. La différence entre les commandes et les paniers est le statut de paiement, cette différence est déjà commune dans les autres sites marchants. Un code couleur suit les boutons : le retour est en bleu, la sauvegarde de modification en vert et la suppression en rouge.

3.2.5 Prévention des erreurs

Au-delà de la conception de messages d'erreur clairs, il faudra en premier lieu être attentif à ce que le design permette de prévenir les problèmes que pourrait rencontrer l'utilisateur.

— **Comment est-elle bien mise en application ?** Mise en place à l'achat, un client ne peut pas mettre dans son panier un produit qui n'est pas en stock. Mais aussi via les formulaires (inscription et ajout/modification de produit) : par exemple les champs mail doivent contenir un arrobase, un stock ou un prix ne peut pas être négatif, la région est choisie parmi une énumération, etc...

— **Comment est-elle mal mise en application ?** Les formulaires pourraient être plus stricts. La vérification de mail a été déactivée momentanément. La vérification du panier n'a pas été complétement mise en place (pour vérifier l'adéquation entre les prix enregistrés dans le panier et les prix courants).

3.2.6 Reconnaître plutôt que se souvenir

Rendre visible les objets, les actions et les options. L'utilisateur ne devrait pas avoir à se souvenir d'une information, d'une séquence de dialogue à l'autre. Les instructions pour utiliser le système devraient être immédiatement visibles ou facilement accessibles, à chaque fois que l'utilisateur en a besoin.

— **Comment est-elle bien mise en application ?** Le panier contient toutes les informations nécessaires pour le paiement (y compris les prix unitaires et le prix total).

— **Comment est-elle mal mise en application ?** Le panier n'affiche pas le nombre de produits déjà sélectionnés dans la barre de navigation. Les filtres de produit ne sont pas persistants : ils s'effacent après avoir été utilisées.

— **Comment pourrait-elle être appliquée ?** La barre de navigation pourrait se servir de product.service pour avoir le nombre de produits sélectionnés et l'afficher tout le temps. La non-persistence des filtres est due à un bug dans le rafraichissement des pages (qu'on n'a pas eu le temps de régler).

3.2.7 Flexibilité dans l'utilisation

Les raccourcis - ignorés par des utilisateurs novices – permettent souvent d'accélérer les interactions pour les utilisateurs expérimentés. Ainsi le système peut convenir à la fois aux utilisateurs inexpérimentés et expérimentés. Autoriser les utilisateurs à personnaliser les actions récurrentes.

- **Comment est-elle bien mise en application ?** Du côté client, il est possible de mettre un produit dans le panier de deux manières (via la liste ou la vue détaillée), et le retirer de plusieurs manières (via le panier ou la vue détaillée du produit). Du côté administrateur, on peut avoir la vue client tout en ayant la possible de modifier les produits ou alors passer directement par la gestion des stocks.
- **Comment est-elle mal mise en application ?** Aucun raccourci clavier n'a été prévu. Aucun changement de palette de couleurs non plus. La taille des écritures n'est pas configurable.

3.2.8 Esthétique et design minimaliste

Les dialogues ne devraient pas proposer d'informations qui ne sont pas pertinentes ou qui ne sont que rarement nécessaires. Chaque information dans un dialogue entre en concurrence avec les autres informations – et en particulier celles qui sont pertinentes – et diminue leur visibilité relative.

- **Comment est-elle bien mise en application ?** La liste produit est conçue pour comporter uniquement les informations pertinentes, en partie elle comporte moins d'informations que la vue détaillée (pour alléger l'interface).
- **Comment est-elle mal mise en application ?** Les alertes divulgent les identifiants des objets (commande et produit), cette information est inutile à l'utilisateur.

3.2.9 Faciliter l'identification, le diagnostic et la "récupération" des erreurs par l'utilisateur

Les messages d'erreur devraient être formulés en langage clair (pas de codes), indiquer précisément le problème et suggérer une solution pour le résoudre.

- **Comment est-elle bien mise en application ?** Cette fonctionnalité est mise en place pour le formulaire d'inscription où les retours sont claires, pour les erreurs provenant du back (les messages sont en français, compréhensibles et peu détaillés, pas orienté développeur).
- **Comment est-elle mal mise en application ?** Pour le formulaire de modification de produits, les erreurs ne sont pas communiquer.
- **Comment pourrait-elle être appliquée ?** Il faudrait plus de détails sur le formulaire de modification produit.

3.2.10 Aide et documentation

Bien qu'il soit préférable que le système puisse être utilisé sans le recours à une documentation, il peut cependant être nécessaire de fournir de l'aide et de la documentation. Les informations de ce type devraient être faciles à trouver, centrées sur la tâche de l'utilisateur, indiquer concrètement les étapes à suivre et ne pas être trop longues.

— **Comment est-elle mal mise en application ?** Non appliquée.

— **Comment pourrait-elle être appliquée ?** Il faudrait rajouter des points d'aide (des boutons qui survolés expliquent ce qui est attendu). On aurait voulu ajouter de la documentation spécifique au choix des vins pour aider et guider les clients novices sur le sujet.

4 Partie Agile

Les méthodes agiles nous ont offert différents avantages lors de la réalisation de ce projet.

4.1 Une structure

La nécessité du backlog nous a poussé très tôt à définir les fonctionnalités, leur importance et le temps qu'on pensait y consacrer. Ce document nous servait à ne pas nous dispercer lors de l'avancée des travaux mais aussi à voir rapidement les fonctionnalités qui avaient une implémentation similaire et pouvaient donc être réalisées rapidement ou par la même personne.

4.2 Un élan de communication

La démarche agile insiste sur l'importance de la communication que ce soit par les daily ou les rétro. Cela empêche de manière très efficace l'effet tunnel où chacun est occupé sur sa partie. Nous avons pu être ouvert sur le travail des autres, s'échanger des conseils lors de difficultés et souffrir ensemble lors de nos tentatives infructueuses.

Tous ces efforts de communication nous ont permis de nous sortir d'une situation où tout le monde était perdu et commençait à empiéter sur le travail des autres de manière non constructive. La rétro a permis de soulever rapidement ce problème et de repartir sur de bonnes bases avec encore davantage de communication et d'attribution des tâches.

4.3 Une démarche limitée par le contexte

Cependant, nous pensons à l'unanimité que ce projet n'était pas forcément adapté à la mise en place de toutes les méthodes présentées. Etre à la fois *Product Owner*, *Scrum master* et équipier ne permet pas toujours des échanges productifs.

La démarche agile se veut flexible pour faire face aux changements d'exigences client mais les notre étaient fixes dès le début. Il n'y a pas eu à discuter longuement des

fonctionnalités dans notre groupe donc l'essentiel de nos conversations portait sur leur implémentation.

De plus, la difficulté liée aux nouvelles technologies abordées a complètement balayé toute tentative d'estimation de temps sur une si courte période, nous donnant plutôt l'impression d'être toujours en retard.

En conclusion, la méthode agile fournit de précieux outils qui facilitent la construction et la réalisation d'un projet en mettant l'accent sur la communication et en définissant un ensemble de fonctionnalités à monter semaine par semaine. Cependant notre projet était trop court et peu adapté pour que ces outils aient un impact significatif sur le projet.