



---

## Table des matières

1	Analyse préliminaire .....	3
1.1	Introduction .....	3
1.2	Contexte et Architecture .....	4
1.3	Objectifs.....	5
1.4	Planification initiale .....	6
2	Analyse / Conception.....	7
2.1	Concept .....	7
2.2	Use cases scénarios.....	10
2.3	Stratégie de test.....	13
2.4	Risques techniques .....	14
2.5	Planification .....	15
2.6	Dossier de conception .....	16
3	Réalisation.....	17
3.1	Dossier de réalisation .....	27
3.2	Description des tests effectués.....	31
3.3	Erreurs restantes .....	32
3.4	Liste des documents fournis .....	33
4	Conclusions .....	33
4.1	Objectifs atteints .....	33
4.2	Objectifs non-atteints .....	34
4.3	Points positifs .....	35
4.4	Points négatifs .....	35
4.5	Difficultés particulières.....	35
4.6	Suites possibles pour le projet.....	36
5	Annexes.....	37
5.1	Sources – Bibliographie.....	37
5.2	Manuel d'Installation .....	37
5.3	Manuel d'Utilisation.....	38
5.4	Journal de travail .....	40

# **1 Analyse préliminaire**

## **1.1 Introduction**

Sploks est une application desktop développée en Python avec une base MySQL.

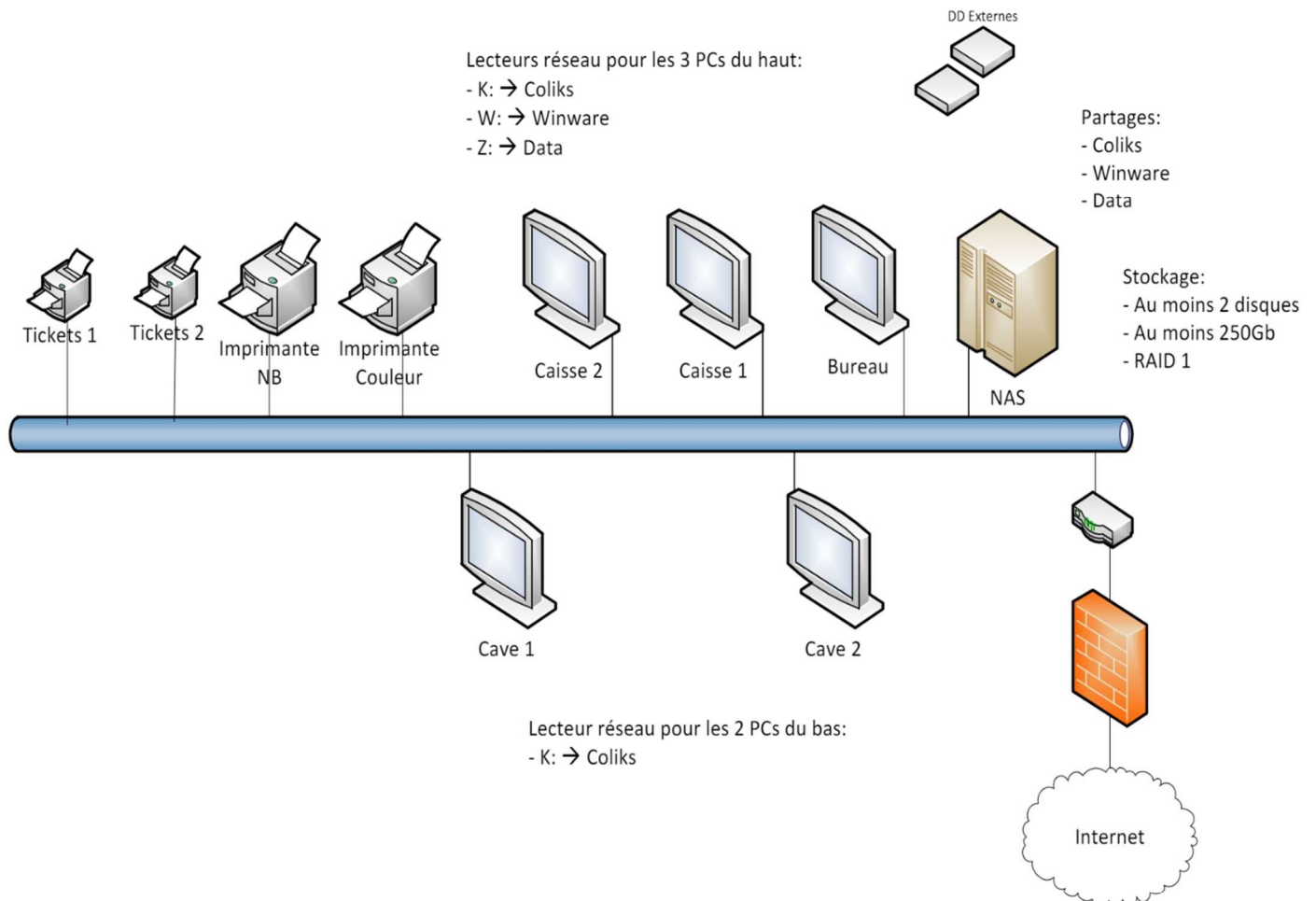
Elle vise à potentiellement remplacer Coliks, une application MSAccess utilisée depuis 17 ans dans le magasin Sports-Time d'Echallens pour gérer la location de matériel de sports d'hiver.

Ce projet est réalisé dans le cadre d'un TPI d'information de 4<sup>ème</sup> année.  
J'ai choisi ce projet pour avoir quelque chose de concret sur mon CV plus tard.

J'ai déjà travaillé sur ce projet durant mon Pré-TPI ce qui implique le fait que la navigation entre les différentes interfaces graphiques a déjà été faite.

## 1.2 Contexte et Architecture

L'application est destinée à fonctionner dans ce contexte réseau.



Une architecture client / serveur est donc obligatoire, avec le serveur hébergé sur la machine « Bureau » (temporairement) et un client sur chaque machine « Caisse 1 », « Caisse 2 », « Cave 1 » et « Cave 2 ».

### 1.3 Objectifs

L'objectif de ce TPI est de réaliser uniquement la partie de gestion de stock de Sploks. La gestion des clients, des contrats de location et du personnel ne font pas partie de ce projet.

Gérer le stock de matériel :

- Pouvoir ajouter du matériel neuf au stock de location en introduisant un article à la fois.
- Pouvoir ajouter du matériel neuf au stock de location en introduisant plusieurs articles à partir d'un formulaire de saisie.
- Pouvoir retirer du matériel trop usagé (matériel retiré doit rester visible dans l'historique).
- Pouvoir mettre à jour l'état dans lequel le matériel se trouve, car le prix de location en dépend.
- Pouvoir montrer les revenus générés par chaque article pour pouvoir savoir lesquels sont rentables. Lors de la consultation des détails d'un article la liste des contrats de location dont il a fait l'objet est accessible et la somme encaissée à travers ceux-ci est affichée.
- 

Gérer les prix de locations :

- Les durées de location doivent être prédéterminées.
- Doit permettre aux gérants de fixer les prix de location pour n'importe quel cas de figure, de la durée courte de matériel haut-de-gamme à la durée longue de matériel très usagé.

## 1.4 Planification initiale

Voici toutes les tâches qui m'ont été attribuées.

En rouges celles qui concernent le stock matériel.

En bleu celles qui concernent les prix de locations



### **Sprint 1 02.05.2022 -> 09.05.2022: Gérer le stock matériel (Lecture)**

- Consulter le stock
- Filtrer le stock
- Consulter les détails d'un article

Sprint review 09.05.2022 à 08h05 jusqu'à 09h05.

### **Sprint 2 09.05.2022 -> 23.05.2022: Gérer le stock matériel (Ajout, Modification)**

- Ajouter des articles
- Ajuster le nombre de pièces d'un article multiple
- Mettre à jour l'état du matériel

Sprint review 23.05.2022 à 08h05 jusqu'à 09h05.

### **Sprint 3 23.05.2022 -> 31.05.2022: Gérer les prix de location (Lecture, Modification)**

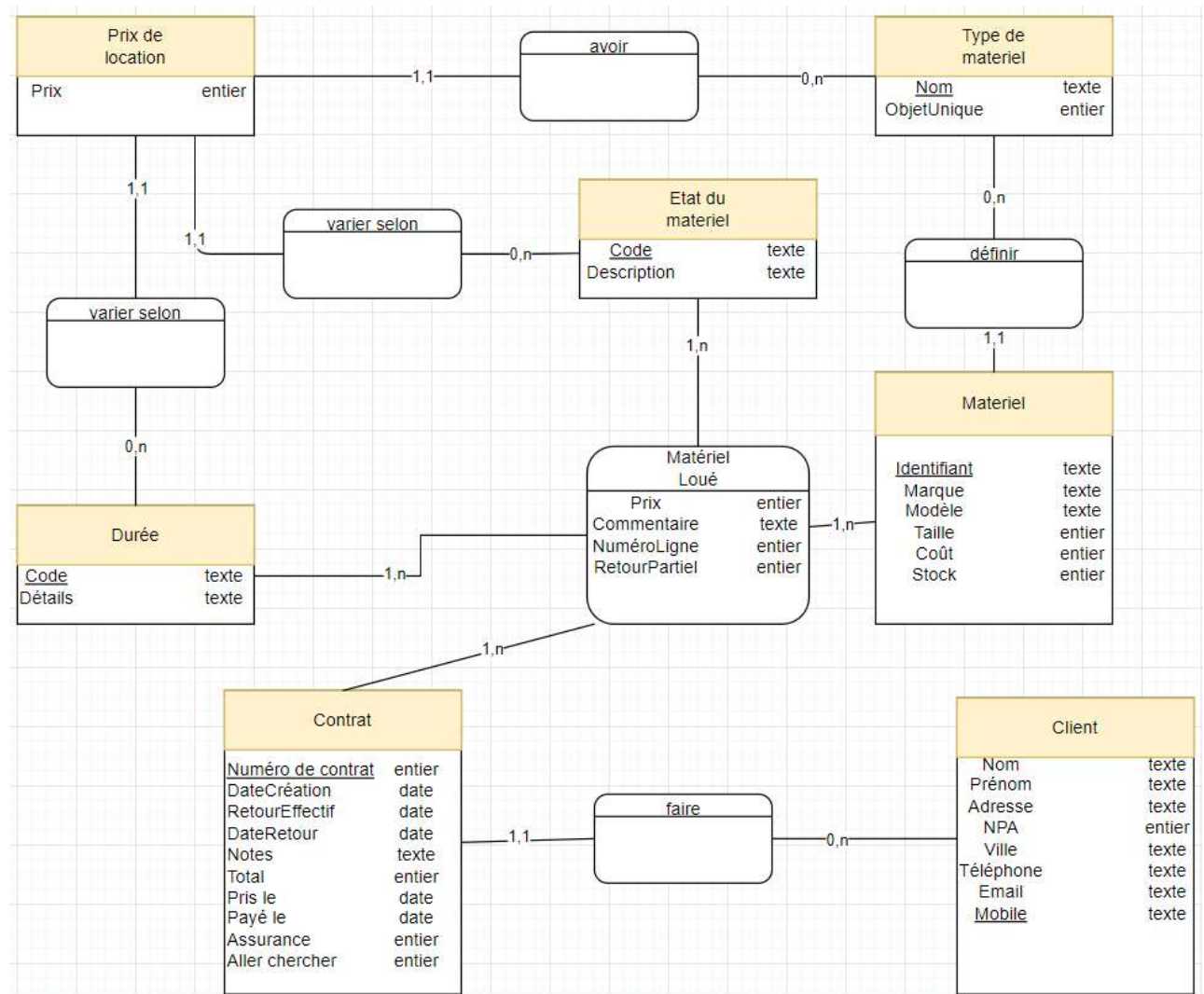
- Afficher les prix
- Travailler par type
- Modifier les prix de location d'un article
- Modifier les prix de location d'un type d'article

Sprint review 31.05.2022 à 13h30 jusqu'à 14h30.

## 2 Analyse / Conception

### 2.1 Concept

#### MCD



Toutes ces entités auront un impact dans mon application et donc doivent être comprises.

Voici certains points qui ne sont pas forcément évident :

**Prix de location :**

C'est une entité qui permet d'avoir tous les scénarios possibles entre le type de matériel, l'état du matériel et la durée. Pour chaque scénario un prix sera attribué, voilà un exemple :

Si on prend un état de matériel **neuf**, en type de matériel **ski** et tout ça pour **2 semaine**. Disons que cela coûte 100 francs, si on prend exactement le même scénario mais avec en état de matériel du **vieux**, forcément que ça coutera moins cher que 100 francs.

**Matériel Loué :**

Matériel Loué est l'intermédiaire de beaucoup d'entité. Un matériel loué est forcément associé à un matériel, ce matériel a aussi un état. Etant donné qu'il est loué, il est forcément loué pour une certaine durée et il appartient à un contrat.





## 2.2 Use cases scénarios

### Sprint 1

#### Consultation du stock

Identifiant	Consultation du stock
En tant que	Utilisateur

Action	Conditions	Réaction
Lancer l'exécutable		Le programme se lance
		Le menu s'affiche
Cliquer sur stock		Une nouvelle fenêtre s'ouvre dans laquelle on peut consulter le stock.
	L'utilisateur a un problème avec sa base de données	Une boîte de dialogue s'ouvre avec le message d'erreur

#### Filtrer le stock

Identifiant	Filtrer le stock
En tant que	Utilisateur

Action	Conditions	Réaction
L'utilisateur remplit les champs de filtrage et clique sur filtrer		Le stock est filtré par rapport aux données entrées
	L'utilisateur a un problème avec sa base de données	Une boîte de dialogue s'ouvre avec le message d'erreur

#### Consulter les détails d'un article

Identifiant	Consulter les détails d'un article
En tant que	Utilisateur

Action	Conditions	Réaction
L'utilisateur double-clique sur un article		Une fenêtre s'ouvre avec les détails de l'article
	L'utilisateur a un problème avec sa base de données	Une boîte de dialogue s'ouvre avec le message d'erreur

## Sprint 2

### Ajouter des articles

Identifiant	Ajouter des articles
En tant que	Utilisateur

Action	Conditions	Réaction
L'utilisateur clique sur ajouter des articles		Une nouvelle fenêtre s'ouvre avec différents champs à remplir
	L'utilisateur à un problème avec sa base de données	Une boîte de dialogue s'ouvre avec le message d'erreur
L'utilisateur remplit les champs et clique sur le bouton encore		Un message de succès s'affiche
	L'utilisateur a entré des valeurs interdites	un message d'erreur s'affiche
L'utilisateur quitte la page en cliquant sur le bouton quitter		La page se ferme et l'utilisateur se retrouve sur le stock

### Ajuster le nombre de pièces d'un article multiple

Identifiant	Ajuster le nombre de pièces d'un article multiple
En tant que	Utilisateur

Action	Conditions	Réaction
L'utilisateur se trouve dans la vue détaillée d'un article et clique sur le bouton Editer.		Tous les champs deviennent éditables sauf le Code article et les revenus générés .
L'utilisateur modifie le nombre de pièces d'un article		La fenêtre se ferme et le nombre de pièce est mis à jour.
	La valeur entrée ne correspond pas au type stock	Un message expliquant l'erreur est affiché en bas à droite.

**Mettre à jour l'état du matériel**

Identifiant	Mettre à jour l'état du matériel
En tant que	Utilisateur

Action	Conditions	Réaction
L'utilisateur se trouve dans la vue détaillée d'un article et clique sur le bouton Editer.		Tous les champs deviennent éditables sauf le Code article et les revenus générés .
L'utilisateur modifie l'état du matériel		La fenêtre se ferme et l'état du matériel est mis à jour.

## 2.3 Stratégie de test

### Test de la base de donnée

Il est important de dire que la base de données est testée avant de pouvoir consulter le stock. C'est-à-dire que si la base de donnée n'est pas atteignable, cela peut être dû à plusieurs raisons :

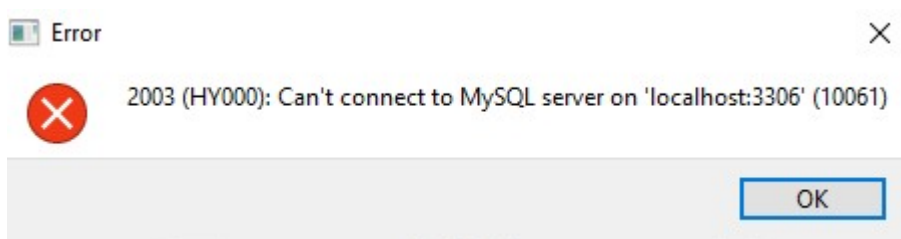
- L'utilisateur a mis les mauvais identifiants dans le fichier. env
- L'utilisateur n'a pas activé le service MySQL
- L'utilisateur ne contient pas la base de données etc.

Si un de ces scénarios arrive, l'application va donner le signal qu'une erreur s'est produite.

```
except mysql.connector.Error as err:
    if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
        raise Exception("Something is wrong with your user name or password")
    elif err.errno == errorcode.ER_BAD_DB_ERROR:
        raise Exception("Database does not exist")
    else:
        raise Exception(err)
```

Sur le bout de code, on peut apercevoir qu'en python pour signaler une erreur on utilise « raise Exception » comme terme.

S'il y a bien un signal d'erreur, on va donc afficher dans un « messageBox » l'erreur obtenue. Voilà un exemple où le service MySQL a été désactivé



### Testeurs externes

Je fais tester mon application une fois par jour par mon collègue Kaarththigan. Grâce à lui je peux voir des choses qui ne sont pas forcément évidentes pour moi en tant que développeur.

Par exemple il m'a dit que les filtres n'étaient pas forcément explicites donc j'ai rajouté du texte pour bien me faire comprendre, voilà ci-dessous les précisions

ajoutées :

Inventaire

Filtrer par : <input type="text" value="Marque"/> <input type="text" value="Modèle"/> <input type="text" value="N/S"/> <input type="text" value="Etat"/> <input type="button" value="Filtrer"/>										
ID	Code	Marque	Modèle	N/S	Taille	Etat	Coût	Retour	Type	Stock
1	09003	TecnoPROx	Rocket Jr	0134013	145	2	0	195	Ski150FS	1
2	1b150	Chaussure 1 ...	taille mondo	ski	150	1	0	94	1b150	102
3	1b155	Chaussure 1 ...	taille mondo	ski	155	1	0	929	1b155	91
4	1b160	Chaussure 1 ...	taille mondo	ski	160	1	0	434	1b160	97
5	1b165	Chaussure 1 ...	taille mondo	ski	165	1	0	2920	1b165	69
6	1b170	Chaussure 1 ...	taille mondo	ski	170	1	0	461	1b170	96

C'est seulement du texte ainsi que des placeholders ajoutés, mais moi aussi je trouve que c'est beaucoup plus simple à comprendre.

Dans mon cahier des charges, il est demandé de tester la robustesse de mon programme, notamment face aux accès concurrents d'au moins quatre postes clients.

Ce test sera effectué la dernière semaine de TPI parce que c'est important de faire ce test lorsque l'application est presque finie et avec la version la plus complète.

## Test

Le programme Sploks a été ouvert 4 fois sur le même exécutable. Les 4 clients ont créé / modifié des articles, tout a bien été mis à jour. Donc nous pouvons en conclure que l'application marche bien avec quatre postes client en même temps.

## 2.4 Risques techniques

### Planning

Mon plus gros risque technique est le planning. 5 Tâches consistent en la mise à jour de la base de donnée et 5 autres tâche ont un rapport juste avec l'affichage.

Je ne pense pas pouvoir finir toutes les tâches, tout simplement parce que il y en a trop. Je pense plutôt pouvoir finir les tâches du sprint 1 et celles du sprint 2. Si tout se passe bien peut-être en prendre une du sprint 3.

## 2.5 Planification

Nous pouvons maintenant comparer la planification initiale du projet et celle qui a réellement été réalisée.

### Voilà la planification initiale :

#### **Sprint 1 02.05.2022 -> 09.05.2022: Gérer le stock matériel (Lecture)**

- Consulter le stock
- Filtrer le stock
- Consulter les détails d'un article

Sprint review 09.05.2022 à 08h05 jusqu'à 09h05.

#### **Sprint 2 09.05.2022 -> 23.05.2022: Gérer le stock matériel (Ajout, Modification)**

- Ajouter des articles
- Ajuster le nombre de pièces d'un article multiple
- Mettre à jour l'état du matériel

Sprint review 23.05.2022 à 08h05 jusqu'à 09h05.

#### **Sprint 3 23.05.2022 -> 31.05.2022: Gérer les prix de location (Lecture, Modification)**

- Afficher les prix
- Travailler par type
- Modifier les prix de location d'un article
- Modifier les prix de location d'un type d'article

Sprint review 31.05.2022 à 13h30 jusqu'à 14h30.

### Et voilà ce qui a réellement été réalisé

#### **Sprint 1 02.05.2022 -> 09.05.2022: Gérer le stock matériel (Lecture)**

- Consulter le stock

Sprint review 09.05.2022 à 08h05 jusqu'à 09h05.

#### **Sprint 2 09.05.2022 -> 23.05.2022: Gérer le stock matériel (Ajout, Modification)**

- Filtrer le stock
- Consulter les détails d'un article
- Ajuster le nombre de pièces d'un article multiple
- Mettre à jour l'état du matériel

Sprint review 23.05.2022 à 08h05 jusqu'à 09h05.

#### **Sprint 3 23.05.2022 -> 31.05.2022: Gérer les prix de location (Lecture, Modification)**

- Ajouter des articles

Sprint review 31.05.2022 à 13h30 jusqu'à 14h30.

On peut en conclure que aucune tâche du sprint 3 ont été effectuées, et qu'il y a eu des tâches qui ont été décalées vers le sprint suivant.

Par rapport aux risques technique cela avait déjà été anticipé que les tâches du sprint 3 vont être compliqué à être faite à cause du temps.

## 2.6 Dossier de conception

Les logiciels qui ont été utilisés :

### **Visual Studio Code (Version: 1.66.2 user setup) :**

Environnement de travail utilisé pour programmer l'application Sploks.

### **Python (Version : 3.10.2) :**

Langage de programmation utilisé pour le projet Sploks.

La librairie utilisé pour l'interface graphique est PyQt5.

La librairie utilisé pour faire la liaison avec la base de donnée est mysql connector

La librairie utilisé pour mettre à part les identifiants de la base de donnée.

### **Git Bash :**

Permet d'effectuer le versioning du projet Sploks, et a aussi permis de mettre à jour le repository sur GitHub (<https://github.com/GruberAdam/TPI-Sploks>).

GitHub est une plateforme de versioning qui permet aussi d'assurer une certaine sécurité en cas de problème avec nos fichiers en local.

Et pour bien structurer mes fonctionnalités avec des branches, j'ai utilisé git-flow.

### **Suite Office :**

Utilisé pour cette documentation, et pour le journal de bord.

([https://docs.google.com/spreadsheets/d/1SHghMPKg4\\_q\\_OUBYOwOJVStz3ETPuOh1Vprxkh7PrmY/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1SHghMPKg4_q_OUBYOwOJVStz3ETPuOh1Vprxkh7PrmY/edit?usp=sharing))

### **MySQL Workbench :**

Utilisé pour pouvoir consulter la base données Sploks et exécuter des requêtes de test.

### **Qt Designer :**

Utilisé pour pouvoir modifier / créer des interfaces graphiques qui seront directement compatibles avec PyQt5.

### **IceScrub :**

Application développée par Mr. Carrel qui nous aide à gérer les aspect spécifiques d'IceScrum dans une interface simplifiée. C'est avec cet application que j'ai fait mon journal de travail. (<https://github.com/XCarrel/IceScrub>)



### 3 Réalisation

#### **Sprint 1 :**

Rappel du sprint 1

Le sprint 1 consiste à :

- Consulter le stock
- Filtrer le stock
- Consulter les détails d'un article

Fin du sprint 1 le 09.05.2022 à 08h05 (sprint review)

Les interfaces graphiques qui ont été utilisées, ont majoritairement déjà été faites au préalable. Cependant il est possible que quelques modifications ont été effectuées sur certaines d'entre elles.

#### **Analyse**

Je compte commencer ce sprint en finissant la consultation du stock parce que je l'avais déjà commencé durant mon pré-tpi et donc la tâche sera donc rapidement terminée.

Ensuite j'effectuerai la consultation du détail d'un article qui n'est rien d'autre que de l'affichage cherché dans la base de données par rapport à un seul article

Pour finir je finirai par le filtre parce que je pense que c'est la tâche la plus du sprint 1 à mon avis car je n'ai jamais fait de filtre.

#### **Conception**

Être apte à consulter le stock sera utile lorsque l'utilisateur voudra voir la liste complète de tous les articles.

De plus s'il veut chercher par exemple tous les skis qui sont de la marque Rossignol il lui suffira seulement, d'entrer « Rossignol » dans le filtre de la marque.

L'utilisateur souhaite maintenant avoir les revenus générés d'un article avec le numéro de série « 43212 », il suffit seulement d'effectuer un filtre avec ce numéro de série et ensuite d'ouvrir la vue détaillée de cet article.

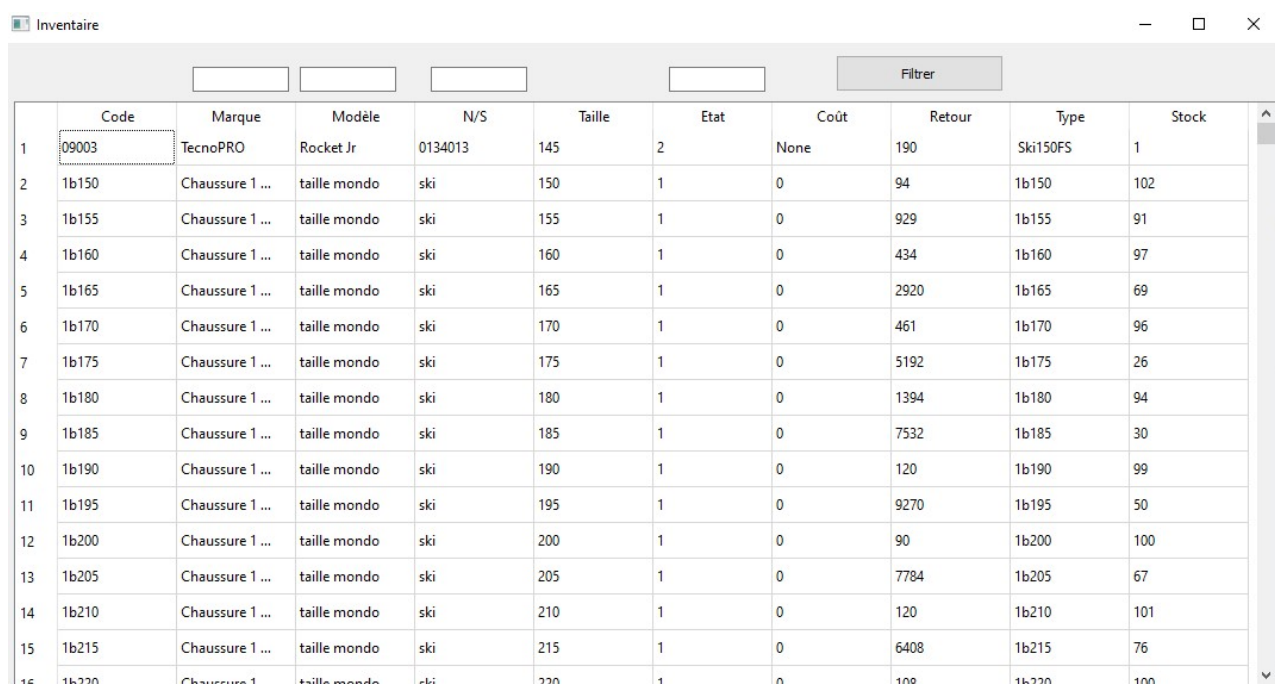
## Réalisation du sprint 1

### Consulter le stock

En ce qui concerne la consultation du stock, ou en reformulant : Chercher des données en lien avec le stock dans la base de donnée et les placer dans un tableau.

Ce n'était pas très dur étant donné que durant mon pré-tpi je devais effectuer la même manipulation sauf que c'était destiné aux clients.

Voilà à quoi ressemble le stock de tous les objets. Le tableau a déjà été fait, mais j'ai ajouté l'option qui va pouvoir filtrer.



	Code	Marque	Modèle	N/S	Taille	Etat	Coût	Retour	Type	Stock
1	09003	TecnoPRO	Rocket Jr	0134013	145	2	None	190	Ski150FS	1
2	1b150	Chaussure 1 ...	taille mondo	ski	150	1	0	94	1b150	102
3	1b155	Chaussure 1 ...	taille mondo	ski	155	1	0	929	1b155	91
4	1b160	Chaussure 1 ...	taille mondo	ski	160	1	0	434	1b160	97
5	1b165	Chaussure 1 ...	taille mondo	ski	165	1	0	2920	1b165	69
6	1b170	Chaussure 1 ...	taille mondo	ski	170	1	0	461	1b170	96
7	1b175	Chaussure 1 ...	taille mondo	ski	175	1	0	5192	1b175	26
8	1b180	Chaussure 1 ...	taille mondo	ski	180	1	0	1394	1b180	94
9	1b185	Chaussure 1 ...	taille mondo	ski	185	1	0	7532	1b185	30
10	1b190	Chaussure 1 ...	taille mondo	ski	190	1	0	120	1b190	99
11	1b195	Chaussure 1 ...	taille mondo	ski	195	1	0	9270	1b195	50
12	1b200	Chaussure 1 ...	taille mondo	ski	200	1	0	90	1b200	100
13	1b205	Chaussure 1 ...	taille mondo	ski	205	1	0	7784	1b205	67
14	1b210	Chaussure 1 ...	taille mondo	ski	210	1	0	120	1b210	101
15	1b215	Chaussure 1 ...	taille mondo	ski	215	1	0	6408	1b215	76
16	1b220	Chaussure 1 ...	taille mondo	ski	220	1	0	108	1b220	100

Les champs que vous apercevez au-dessus des colonnes « Marque », « Modèle », etc... Sont des champs textuels (TextEdit), qui vont donc pouvoir contenir ce que l'on veut filtrer.

Une fois les champs textuels remplis, il y a un bouton Filtrer sur la droite, qui permet d'exécuter le filtrage.

(Le champ « Etat », qui représente l'état des objets, a été gardé numériquement car sur Coliks il est aussi représenté numériquement)

## Consulter les détails d'un article

Voilà à quoi ressemble le détail d'un article :



The screenshot shows a window titled "Détail Article" with a close button (X) in the top right corner. The window contains two columns of text input fields. The left column has four fields: "Code d'article:" with value "1b150", "Numéro de série:" with value "ski", "Prix d'achat:" with value "0", and "Revenues générés:" with value "80". The right column has four fields: "Marque:" with value "Chaussure 1 boucle", "Modèle:" with value "taille mondo", "Type:" with value "1b150", "Stock:" with value "102", and "Taille:" with value "150". Below these fields, there is a button labeled "Liste des Contrats" and a text label "Nb Contrats: 7".

La seule modification qui a été faite sur cette interface graphique est au niveau des champs textuels.

L'ancien champ textuel s'appelait un « **Label** », le problème avec ce champ c'est qu'il n'est pas fait pour être modifié. C'est-à-dire que si on fait une modification sur un champ label, la variable qui contient ce champ ne sera pas mis à jour et gardera l'ancienne valeur.

C'est pourquoi j'ai mis à la place un champ textuel qui s'appelle « **TextEdit** » qui lui est mis à jour lors d'une modification.

Je l'ai fait en avance parce que je sais que dans le futur les utilisateurs devront être à même à modifier ces champs. Plutôt que de le faire plus tard, je l'ai fait tout de suite.

Pour le reste c'était comme **la consultation du stock**, je l'avais déjà fait auparavant donc ce n'étais pas très compliqué.

Il suffisait de mettre un « event Listener » qui permet d'écouter sur quel ligne l'utilisateur clique. Ensuite la fenêtre qui comporte les détail d'un article s'ouvre et affiche les données plus précise.

En ce qui concerne les revenus générés, il suffisait d'effectuer une addition dans la table qui contient tous les articles qui ont été loués.

## Filtrer le stock

Pour filtrer le stock c'était une histoire différente, car je n'avais jamais effectué de filtre. J'ai donc décidé de garder les choses simples et de juste ajouter des

champs textuels modifiable au-dessus des colonnes qui doivent être filtrée, comme je vous l'ai montré sur l'image ou je consulte le stock.

Accompagné d'un bouton au côté droite qui permet d'exécuter le filtrage.

### **Pour finir**

Je n'ai quand même pas décidé de mettre la story consulter les détails d'un article en review car il y avait un problème avec le nombre de contrats qui s'affiche.

J'ai n'ai pas eu le temps de régler ce problème, donc j'ai juste décidé de review les 2 première stories et finir celle l'a dans la suivante.

### **Sprint review**

La sprint review c'est passée moins bien que prévu.

- J'ai oublié de mettre les 2 stories en « in-review » dans IceScrum.
- Les identifiants de la base de donnée ne devraient pas être présent sur GitHub.
- Je code sans comprendre ce que je fais (par exemple dans la consultation du stock un colonne « Retour » est présente, pourtant je ne savais pas quel était son utilité.)

La story « consulter le stock » a cependant été validée.

En ce qui concerne la story « Filtrer le stock », elle n'a pas été validée pour différentes raisons

- Lorsqu'on effectue un filtrage la touche entrer n'est pas fonctionnelle
- Lorsqu'on veut passer d'un champ texte à un autre, plutôt que d'appuyer avec notre souris sur le prochain champ texte, la touche tab devrait être fonctionnel.
- Il manque une redirection en haut du tableau lorsqu'on effectue un filtrage.

D'autres points qui ont été ajouté :

- Mettre la colonne « Etat » textuellement et non numériquement.
- Mettre au tâches effectué des tags (Mais le journal de travail était bien exécuté en général)
- Faire tester son application avant de la présenter

En somme, ce sprint n'était pas forcément positif, mais je compte m'améliorer pour le suivant.

## Sprint 2

Rappel du sprint 2 selon la planification initiale :

- Ajouter des articles
- Ajuster le nombre de pièces d'un article multiple
- Mettre à jour l'état du matériel

Fin du sprint 2 le 23.05.2022 à 08h05 (sprint review)

Contenu du sprint 2 :

- Filtrer le stock (fonctionnalités à ajouter)
- Consulter les détails d'un article (à finir)
- Ajouter des articles
- Ajuster le nombre de pièces d'un article multiple
- Mettre à jour l'état du matériel

Fin du sprint 2 le 23.05.2022 à 08h05 (sprint review)

## Analyse

Je compte commencer ce sprint en finissant ce que je devais faire durant le sprint 1. Donc en résumant, finir la consultation des détail d'un article. Ainsi que le Filtrage du stock ou plus précisément la fonctionnalité d'appuyer sur « **entrer** » pour valider le filtrage et la fonctionnalité de pouvoir faire « **tab** » pour passer au prochain champ textuel.

Ensuite, la tâche **ajouter** des articles est un peu à part des 2 autres tâches qui consistent à **modifier** soit, l'état ou le nombre de pièces d'un article multiple.

Je vais commencer par l'**ajout** des articles puis poursuivre avec l'**ajustement** du nombre de pièces d'un article multiple pour finir avec la **mise à jour** de l'état du matériel.

## Conception

L'ajout des articles sera utile lorsqu'un manager veut pouvoir ensuite les mettre en location.

Ajuster le nombre de pièces d'un article multiple sera utile lorsqu'un manager veut que le nombre dans l'application corresponde à la réalité du stock.

Mettre à jour l'état du matériel sera utile lorsqu'un manager veut être prêt à démarrer la saison de location

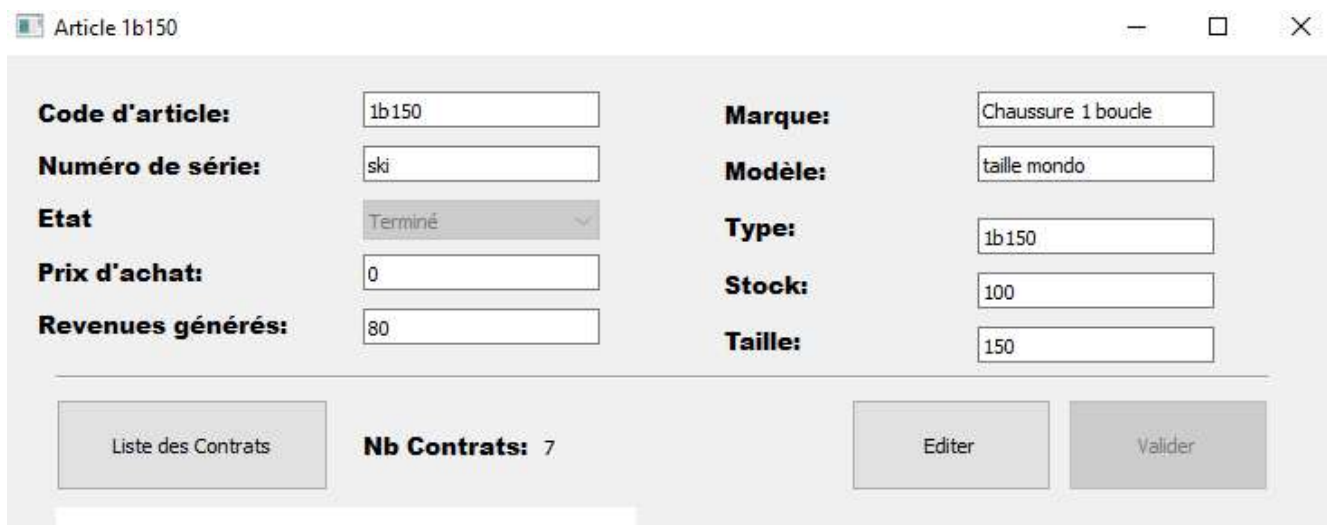
## Réalisation du sprint 2

### Consultation des détails d'un article :

J'ai fait une requête qui compte le nombre de fois que « item id » se trouve dans la table « rented items », pour afficher le nombre de contrats présents.

Il fallait aussi renommer les fenêtres détaillées par rapport au « code article » que l'article comprenais.

Grace à la visite de Mr. Lymberis, j'ai remarqué que l'état n'était pas visible dans la vue détaillée. Donc j'ai rajouté une « comboBox » qui n'est pas modifiable (juste visuelle). Voilà à quoi ça ressemble :



<b>Code d'article:</b>	1b150	<b>Marque:</b>	Chaussure 1 boucle
<b>Numéro de série:</b>	ski	<b>Modèle:</b>	taille mondo
<b>Etat</b>	Terminé	<b>Type:</b>	1b150
<b>Prix d'achat:</b>	0	<b>Stock:</b>	100
<b>Revenues générés:</b>	80	<b>Taille:</b>	150

Liste des Contrats      **Nb Contrats: 7**      Editer      Valider

### Filtrer le stock :

Pour gérer la touche tabulateur, il suffisait seulement de modifier les propriétés des champs textes dans QtDesigner, ensuite on peut choisir l'ordre dans lequel le tabulateur va être exécuté.

Pour gérer la touche entrer c'était un peu plus dur. Je m'explique, comme je vous l'avais dit j'utilisais comme champ textuel des **TextEdits**. Etant donné que les TextEdits peuvent contenir des retours à la ligne, à chaque fois qu'on appuie sur « entrer » dans un TextEdit, on effectue un retour à la ligne.

J'ai essayé de supprimer le texte lorsque l'utilisateur appuie sur entrer, sauf que le retour à la ligne ne voulait pas être supprimé.

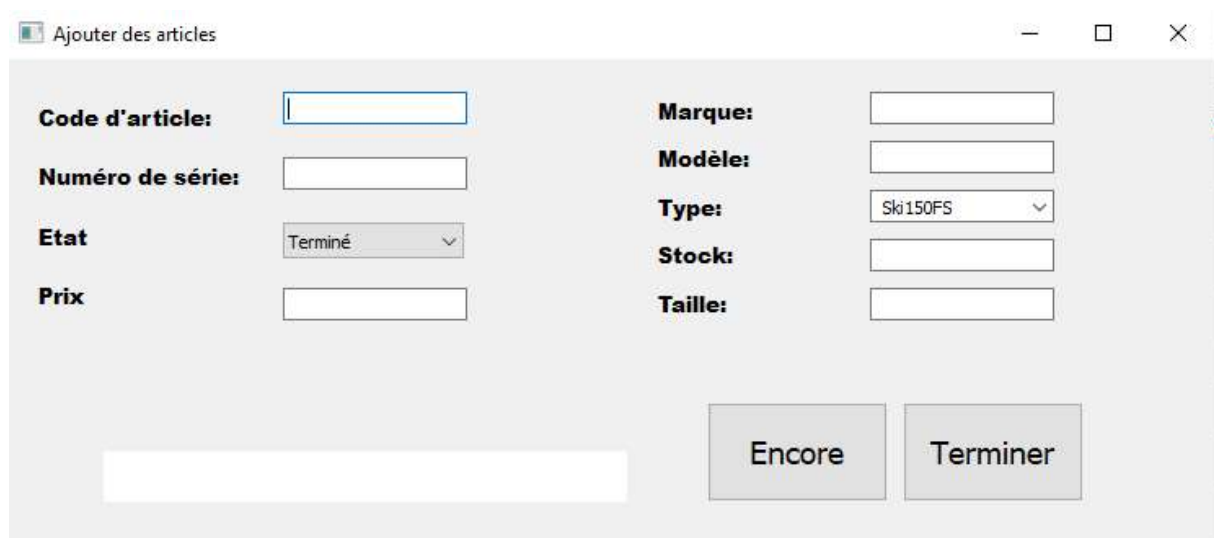
Après avoir effectué plusieurs recherches j'ai trouvé un champ textuel qui s'appelle « **LineEdit** ». Il comporte les mêmes fonctionnalités que le « TextEdit » sauf qu'il reste toujours sur une ligne. J'ai donc modifié tous mes filtres avec champ « TextEdit » en « LineEdit » et tout fonctionne.

**Ajouter des articles :**

Tout d'abord il fallait créer un formulaire d'entrée pour créer des articles. La seule contrainte qui était donnée sur ce formulaire, c'est qu'il y aura 2 radiobox. Une radiobox aura écrit à côté « multiple » et l'autre aura écrit à côté « unique », si l'utilisateur clique sur multiple, on lui demandera combien le nombre d'article qu'il souhaite ajouter, inversement s'il clique sur unique, on lui demandera un numéro de série pour son article.

Durant cette tâche Mr. Carrel a vu comment je programmais et il a trouvé que ma manière de développer n'était pas la plus optimale. Il s'avère que je faisais de la logique métier dans mon contrôleur (Je travaille en MVC). Il m'a donc proposé de refaire tout ce qui se passe au niveau du model, ce que j'ai accepté. C'est pourquoi après cette tâche je vais devoir retravailler mon code et donc devoir consacrer du temps là-dedans.

En continuant sur la tâche d'ajout d'article, voilà à quoi ressemblais mon formulaire : (Je n'avais pas mis les 2 radios box parce que je n'avais pas compris leurs utilité)



On peut apercevoir plusieurs champs qu'il faut remplir. Le champ type est une liste de choix chargé depuis la base de donnée. Donc tous les types possibles sont affichés.

Les seules contraintes par rapport à la base de donnée sont les champs « Prix », « Taille » et « stock » qui doivent impérativement être des nombres. Si ce n'est pas le Cas, un message d'erreur qui indique l'erreur apparaît en bas à gauche du formulaire

De plus si les valeurs de ces nombres sont trop grandes, la base de donnée n'arrive pas à les ajouter. C'est pourquoi j'ai mis une limite de 7 caractères sur ces 3 champs.

J'ai eu une conversation avec Mr. Carrel concernant l'ajout des articles, et voilà ce que j'ai pu en ressortir :

The screenshot shows a web form titled "Ajouter des articles" with a standard window interface (minimize, maximize, close buttons). The form contains the following fields and controls:

- Code d'article:** A text input field.
- Marque:** A text input field.
- Etat:** A dropdown menu currently showing "Neuf".
- Modèle:** A text input field.
- Type:** A dropdown menu currently showing "Choix".
- Prix:** A text input field.
- Taille:** A text input field.
- Type stock:** Two radio buttons labeled "Unique" and "Multiple".
- At the bottom right, there are two buttons: "Encore" and "Quitter".
- Below the "Type stock" section, there is a large, empty text area.

Donc il m'a bien parlé des champs qui étaient obligatoires (avec une astérisque). J'ai donc pu comprendre l'histoire avec les 2 radio box (comme sur le schéma) contient le nom « Unique » et « Multiple ».

Et donc si l'utilisateur clique sur unique, il doit entrer (s'il y en a un) le numéro de série de l'objet. En échange s'il s'agit de plusieurs articles qu'il ajoute à la fois, il clique sur Multiple et doit remplir le nombre d'article qu'il souhaite ajouter.

### Factorisation du code

J'ai perdu certes du temps sur la factorisation du code mais c'est vraiment important de bien savoir séparer son contrôleur et son model.

Avant j'avais juste des fonctions dans mon model qui communiquaient avec la base de donnée, mais Mr Carrel m'a recommandé de faire des classes. C'est pourquoi j'ai refait toute la structure du model avec des classes.

Dans le stock par exemple, j'ai la classe parente « Stock » qui contient différentes méthodes comme « Obtenir tout le stock » et en classe enfant j'ai la classe « Item » qui elle va permettre d'obtenir des informations sur un seul article.

### Ajuster le nombre de pièces d'un article multiple

Lors de l'ajustement de l'article je me suis demandé si un article n'a pas le droit d'être ajouté à cause d'un champ n'autorise pas les doublons. J'ai donc pensé que c'est le code article qui serait l'identifiant naturel, étant donné que le numéro de série n'est pas obligatoire dans le formulaire d'ajout d'article.

J'ai quand même décidé d'avoir une conversation avec Mr. Carrel et il m'a confirmé mon choix (et a même dit que le numéro de série sera supprimé.), suite à ce questionnement j'ai fait un MCD, qui peut être trouvé plus haut.



## **Sprint review**

Bonne sprint review en général.

Sur les 4 stories présentée, les 4 ont été validée. La seule storie qui n'a pas été présentée est « Ajouter des articles » tout simplement car je n'ai pas eu le temps de la terminer.

Sinon quelques points ont été demandé d'être revu sur les différentes vues.

Le plus gros problème que j'ai, c'est qu'il est possible que je mélange mon code avec les différentes branches que j'ai créé.

## **Sprint 3**

Rappel du sprint 3 selon la planification initiale :

- Afficher les prix
- Travailler par type
- Modifier les prix de location d'un article
- Modifier les prix de location d'un type d'article

Sprint review 31.05.2022 à 13h30 jusqu'à 14h30.

Contenu du sprint 3 :

- Ajouter des articles
- Afficher les prix
- Peaufinassions

Sprint review 31.05.2022 à 13h30 jusqu'à 14h30.

## **Analyse**

Je vais commencer en finissant les problèmes relevés par Mr. Carrel durant la sprint review. Ensuite je finirai l'ajout d'article puis pour finir j'effectuerai l'affichage des prix.

Je ne vais pas faire les autres stories, parce que je n'aurais tout simplement pas le temps de les finir.

De plus la sprint review aura lieu après le rendu du TPI tout simplement avec les horaires qui ne pouvaient pas convenir avec celles de mon chef de projet.

## **Conception**

Afficher les prix sera utile lorsqu'un gérant veut voir les prix de location sous la forme d'un tableau, pour en avoir une bonne vue d'ensemble.

## Réalisation

### Ajouter des articles

Encore une fois, l'ajout d'article était presque terminé, il manquait le fait qu'un utilisateur puisse ajouter un nouveau type. Donc voilà la vue finale de l'ajout d'article :



The screenshot shows a window titled "Ajouter des articles" with the following fields and controls:

- Code d'article:** A text input field.
- Marque:** A text input field.
- Etat:** A dropdown menu with "Neuf" selected.
- Modèle:** A text input field.
- Prix:** A text input field.
- Type:** A dropdown menu with "Choix" selected.
- Taille:** A text input field.
- Type stock:** Radio buttons for "Unique" (selected) and "Multiple". A text input field labeled "Nombre" is next to the "Multiple" option.
- Buttons:** "Encore" and "Quitter" buttons at the bottom right.
- Footer:** A long, empty text input field at the bottom left.

### Afficher les prix

Je n'ai préféré pas commencer cette tâche, parce que je n'aurai pas eu assez de temps pour peaufiner des choses plus importantes.

### Peaufinassions

J'ai eu le temps de retravailler la ligne sélectionnée, qui ne fonctionnait pas dans tous les scénarios, ainsi qu'adapter mon application pour un fichier .exe.

Pour le fichier .exe, j'ai utilisé pyinstaller. J'ai pris beaucoup de temps à réussir à le faire tout simplement à cause des fichiers « .ui ». Lorsqu'un fichier .exe est exécuté, le code python est exécuté dans une arborescence temporaire. J'ai eu du mal à intégrer ces fichiers .ui dans cette arborescence. Au début j'ai trouvé sur stackoverflow, les gens disaient qu'il fallait convertir les fichiers .ui en .py comme ça pyinstaller pourra lire tout ça facilement. Mais c'était très compliqué, donc j'ai cherché sur une autre piste et j'ai trouvé le fait de pouvoir ajouter des data files avec pyinstaller. C'est beaucoup plus simple que convertir chaque fichier ui en py.

### 3.1 Dossier de réalisation

Voilà l'arborescence de mon projet :

#### A la racine

controller	Ajout des chemins relatifs pour le .exe	17 hours ago
doc	Mise à jour des documents	6 days ago
model	Ajout des chemins relatifs pour le .exe	17 hours ago
packages	Package inutile	10 days ago
view	Suppression des fichiers inutiles	17 hours ago
.gitignore	Initial commit	28 days ago
README.md	Table des matières fonctionnelle	15 hours ago
script.sql	Mise à jour du script sql	17 hours ago
sploks.exe	.exe de l'application mis en place	16 hours ago
sploks.py	Ajout des chemins relatifs pour le .exe	17 hours ago

**.gitignore** : Permet d'ignorer des fichiers que l'on ne veut pas ajouter sur notre repository. Par exemple les fichiers pycaches.

**README.md** : Document textuel qui permet de mettre en place l'application avec un manuel d'utilisation

**script.sql** : Script qui contient toutes les données de la base de données.

**sploks.exe** : Exécutable qui lance l'application sploks.

**sploks.py** : Point d'entrée de l'application sploks en python.

## Dans doc

IceScrub-master	projet initial
Documentation_de_projet-Adam_Gruber-24.05.2022.pdf	Mise à jour des documents
Journal_de_travail-Adam_Gruber-24.05.2022.pdf	Mise à jour des documents

Contient tous les fichiers liés à la documentation du projet.

**IceScrub-master** : Permet de mettre en forme le journal de travail sur IceScrum.

**Documentation\_de\_projet-Adam\_Gruber-xxx** : La documentation du projet en PDF.

**Journal\_de\_travail-Adam\_Gruber-xxx** : Journal de travail du projet en PDF.

## Dans packages :

..	
PyQt5	Package inutile
dotenv	Ajout du package dotenv
mysql	projet initial




Contient toutes les librairies du projet

**PyQt5** : Librairie qui s'occupe de l'interface graphique du projet.

**dotenv** : Librairie qui s'occupe de lire le fichier .env à la racine du projet. (Variables globales)

**mysql** : Librairie qui permet de faire des requêtes SQL en python.

### Dans controller :

 contractsController.py	Le totaux des contrats est consultable
 mainMenuController.py	Ajout des chemins relatifs pour le .exe
 stockController.py	Ajout des chemins relatifs pour le .exe

**ContractsController.py** : Permet d'afficher la fenêtre des contrats et de rediriger tous les comportement utilisateur au bon endroit.

**mainMenuController.py** : Permet d'afficher le menu principal et de rediriger tous les comportement utilisateur au bon endroit.

**stockController.py** : Permet d'afficher tout ce qui est en rapport avec le stock et de rediriger tous les comportement utilisateur au bon endroit.

### Dans model :

 Contracts.py	Refactoring du code terminé
 Stock.py	In progress de l'ajout d'un nouveau type
 database.py	Ajout des chemins relatifs pour le .exe

Vous avez peut-être vu, que le fichier database.py n'a pas de majuscule. C'est tout simplement parce que ce n'est pas une classe contrairement à Contracts.py et Stock.py

**Contrats.py** : Effectue toutes les requêtes SQL concernant les contrats.

**Stock.py** : Effectue toutes les requêtes SQL concernant le Stock.

**database.py** : La connexion avec la base de donnée, et l'exécution des requêtes sont fait dans ce fichier.

**Dans view :**

 addItemsView.ui	In progress de l'ajout d'un nouveau type
 contractsView.ui	Le totaux des contrats est consultable
 itemDetailsView.ui	Vue de du détail d'un item mis à jour
 menuView.ui	projet initial
 stockView.ui	L'inventaire est maintenant utilisable sans souris

Les fichiers peuvent être facilement lu et interprété par PyQt5, c'est pour cela que mes vues sont tous en fichier « .ui ». Un fichier « .ui » n'est rien d'autre que du XML.

**addItemsView.ui** : Contient la vue de l'ajout d'un article.

**contractsView.ui** : Contient la vue de la liste des contrats.

**itemDetailsView.ui** : Contient la vue détaillée d'un article.

**menuView.ui** : Contient la vue du menu principal.

**stockView.ui** : Contient la vue du stock en entier.

**Version du projet : 1.0**

### 3.2 Description des tests effectués

Tous les tests ont été effectués le 30.05.2022 par mon collègue Kaarththigan en respectant les uses cases fait au préalable.

#### Use case du sprint 1 :

Action	Réussi ?
Consultation du stock	Réussi
Filtrer le stock	Réussi
Consulter les détails d'un article	Réussi

#### Use case du sprint 2

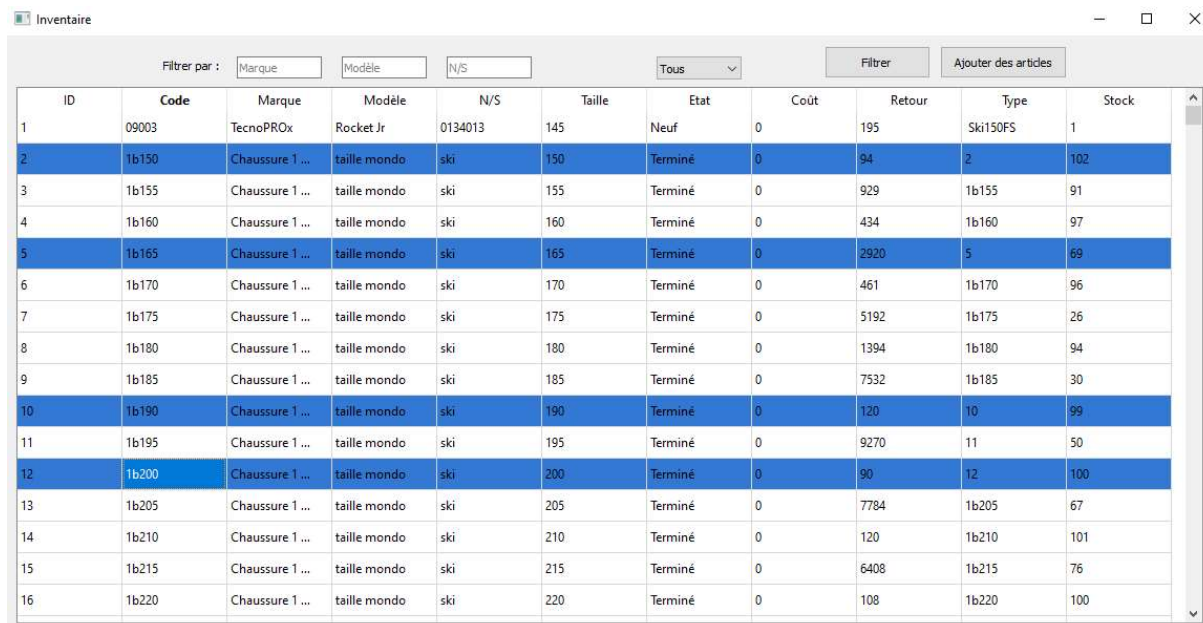
Action	Réussi ?
Ajouter des articles	Réussi
Ajuster le nombre de pièces d'un article multiple	Réussi
Mettre à jour l'état du matériel	Réussi

### 3.3 Erreurs restantes

#### Le clic droit

Par rapport aux cas qui ne sont pas géré, il y a le clic droit.

Si l'utilisateur fait des doubles clics droits sur le tableau qui affiche tout l'inventaire, plusieurs lignes seront sélectionnées.



The screenshot shows a web application window titled 'Inventaire'. It contains a table with 12 columns: ID, Code, Marque, Modèle, N/S, Taille, Etat, Coût, Retour, Type, and Stock. The table lists 16 items. Rows 2, 5, 10, 12, and 13 are highlighted in blue, indicating they are selected. Above the table, there are filters for 'Marque', 'Modèle', and 'N/S', a dropdown menu set to 'Tous', and buttons for 'Filtrer' and 'Ajouter des articles'.

ID	Code	Marque	Modèle	N/S	Taille	Etat	Coût	Retour	Type	Stock
1	09003	TecnoPROx	Rocket Jr	0134013	145	Neuf	0	195	Ski150FS	1
2	1b150	Chaussure 1 ...	taille mondo	ski	150	Terminé	0	94	2	102
3	1b155	Chaussure 1 ...	taille mondo	ski	155	Terminé	0	929	1b155	91
4	1b160	Chaussure 1 ...	taille mondo	ski	160	Terminé	0	434	1b160	97
5	1b165	Chaussure 1 ...	taille mondo	ski	165	Terminé	0	2920	5	69
6	1b170	Chaussure 1 ...	taille mondo	ski	170	Terminé	0	461	1b170	96
7	1b175	Chaussure 1 ...	taille mondo	ski	175	Terminé	0	5192	1b175	26
8	1b180	Chaussure 1 ...	taille mondo	ski	180	Terminé	0	1394	1b180	94
9	1b185	Chaussure 1 ...	taille mondo	ski	185	Terminé	0	7532	1b185	30
10	1b190	Chaussure 1 ...	taille mondo	ski	190	Terminé	0	120	10	99
11	1b195	Chaussure 1 ...	taille mondo	ski	195	Terminé	0	9270	11	50
12	1b200	Chaussure 1 ...	taille mondo	ski	200	Terminé	0	90	12	100
13	1b205	Chaussure 1 ...	taille mondo	ski	205	Terminé	0	7784	1b205	67
14	1b210	Chaussure 1 ...	taille mondo	ski	210	Terminé	0	120	1b210	101
15	1b215	Chaussure 1 ...	taille mondo	ski	215	Terminé	0	6408	1b215	76
16	1b220	Chaussure 1 ...	taille mondo	ski	220	Terminé	0	108	1b220	100

Sur cette image on peut voir plusieurs lignes sélectionnées en bleu, ce qui ne fait donc aucun sens.

Ce bug est présent, parce que si l'on souhaite qu'une ligne soit désélectionnée, il faudrait faire un clic gauche sur une cellule qui ne l'est pas. Vu que l'utilisateur fait un clic droit, la cellule reste sélectionnée, d'où la provenance du problème.

Pour y remédier il faudrait donc vérifier si l'utilisateur fait un clic droit sur une autre cellule, et effectuer la même tâche que le clic gauche (désélectionner la ligne).

Cette erreur a-t-elle vraiment des conséquences sur l'utilisation du produit ? Elle pourrait juste avoir des conséquences dans lesquels l'utilisateur peut être confus, parce qu'il ne sait pas quel est la vraie sélection.



### 3.4 Liste des documents fournis

A la fin de mon projet, mon chef de projet aura toute ma documentation en format électronique et s'il le souhaite en format papier.

Mes experts auront les 2 le format électronique de ma documentation, mais mon expert numéro 1 l'aura aussi en format papier et mon expert 2 s'il le souhaite, je lui en fournirais un aussi.

Dans ma documentation sera compris :

- Toute ma documentation de projet
- Mon journal de travail
- Mon journal de bord

## 4 Conclusions

### 4.1 Objectifs atteints

Par rapport à ce qui est demandé dans le cahier des charges :

*En italique = ce que le cahier des charges demande*

*En tant que gérant, je veux ajouter du matériel neuf au stock de location en introduisant un article à la fois.*

C'est totalement possible si l'on se trouve sur le stock et que l'on clique sur ajouter des articles.

Un formulaire d'ajout s'ouvrira et nous permettra de choisir l'état du matériel et tout ce qui est nécessaire pour compléter le matériel en question.

*En tant que gérant, je veux ajouter du matériel neuf au stock de location en introduisant plusieurs articles à partir d'un formulaire de saisie.*

C'est la même chose que le point au-dessus sauf que dans le formulaire on coche la case multiple, pour savoir combien d'articles doivent être introduits.

*En tant que gérant, je veux mettre à jour l'état dans lequel le matériel se trouve, car le prix de location en dépend.*

Cet objectif a été parfaitement atteint. Pour effectuer une mise à jour de l'état il suffit d'aller dans la vue détaillée du matériel désiré, puis à l'aide de la comboBox, choisir le nouvel état du matériel.

*En tant que gérant, je veux que l'application me montre les revenus générés par chaque article pour que je puisse savoir lesquels sont rentables.*

L'application montre les revenus générés pour chaque article, mais il n'est pas écrit explicitement s'ils sont rentables ou pas. Pour voir les revenus générés par un article précis il faut se diriger dans sa vue détaillée. Un champ non modifiable qui s'appelle revenus générés est présent.

*Lorsque je consulte les détails d'un article, la liste des contrats de location dont il a fait l'objet est accessible et la somme encaissée à travers ceux-ci est affichée.*

La liste des contrats de location d'un article est disponible lorsque l'utilisateur clic sur le bouton « contrats » disponible dans la vue détaillée de celui-ci. Il peut ensuite visualiser la somme encaissée par ceux-ci.

## **4.2 Objectifs non-atteints**

*En tant que gérant, je veux retirer du matériel trop usagé (mais le matériel retiré doit rester visible dans l'historique).*

Cet objectif n'a malheureusement pas été atteint. Tout simplement parce que j'ai priorisé les tâches présentes sur iceScrum et non celles sur le cahier des charges.

*L'application doit permettre aux gérants de fixer les prix de location pour n'importe quel cas de figure, de la durée courte de matériel haut-de-gamme neuf à la durée longue de matériel très usagé.*

Cet objectif fait partit du sprint 3, donc je n'ai pas eu le temps de le commencer.

#### 4.3 Points positifs

Le plus grand point positif que m'a apporté ce projet, est le fait que j'ai appris à me connaître. Maintenant je sais très bien que j'ai besoin de bien comprendre précisément ce que je dois programmer avant d'agir.

Ce genre de projet est très utile pour voir à quoi ressemble un cadre dans lequel on doit programmer dans mon cas quelque chose de coqueret.

#### 4.4 Points négatifs

Etant donné que j'ai besoin d'avoir une idée claire avant de programmer, j'avais souvent besoin de mon chef de projet, malheureusement il ne pouvait pas toujours être là pour moi. Ce qui implique que parfois j'essayais de faire les choses comme je les pensais justes, mais par la suite je devais les modifier car elles ne l'étaient pas.

#### 4.5 Difficultés particulières

##### **Temps**

Malheureusement je n'ai pas forcément bien géré mon temps, j'aurai dû prendre plus de temps de bien réaliser ce que je dois programmer, avant d'avancer sans avoir une idée très claire de ce que je dois faire.

Même si cela aurait pris un temps conséquent sur mon travail, il aurait été bénéfique sur la suite de mon projet.

##### **. Exe**

J'ai trouvé particulièrement difficile à mettre le projet en .exe. J'ai utilisé pyinstaller pour la conversion. Le souci que j'ai eu, c'est à cause des fichier « .ui », qui ne sont donc pas des fichiers python, et donc j'ai dû écrire des lignes de code pour qu'ils soient ajoutés dans le .exe.

#### 4.6 **Suites possibles pour le projet**

##### **Tâches non terminées**

Les choses les plus importantes à faire pour le projet serait de finir toutes les tâches qui n'ont pas été terminée.

##### **Application responsive**

Une application responsive signifie qu'elle s'adapte à toutes circonstances, que ce soit une résolution différente ou lorsque l'utilisateur agrandi une fenêtre. L'application n'est pas du tout responsive et réagit très mal aux résolutions différentes pour l'instant.

## 5 Annexes

### 5.1 Sources – Bibliographie

Les sources utilisées pour les questions concernant python :

<https://stackoverflow.com/>

<https://www.w3schools.com/>

Les sources utilisés pour les questions concernant PyQt5

<https://doc.qt.io/qt-5/>

<https://stackoverflow.com/>

### 5.2 Manuel d'Installation

Comment faire fonctionner Sploks ?

#### **Télécharger les fichiers nécessaires**

Dirigez vous sur ce sur lien GitHub de sploks (<https://github.com/GruberAdam/TPI-Sploks/tree/develop>).

Ensuite, vous aurez seulement besoin de télécharger 2 fichiers à la racine du projet. Sploks.exe qui va être l'exécutable, et script SQL qui va contenir toutes les données de la base de données.

#### **Installer la base de données**

En ce qui concerne la base de données, je vous invite à installer MySQL workbench, et le service qui va avec, pour pouvoir exécuter le script que vous avez téléchargé.

Dans MySQL il faudra créer un compte qui s'appelle « root » et qui a comme mot de passe « root1234 ». Bien évidemment, ce compte aura besoin d'avoir les droits de modification sur la base de données.

#### **Lancer l'application**

Une fois tout cela effectué, il faut lancer le fichier qui s'appelle « Sploks.exe » téléchargé précédemment. Un menu devra apparaître, lorsque vous cliquez sur « Inventaire », si toutes les étapes ont bien été effectuées auparavant, vous devriez avoir un menu qui s'ouvre.

### 5.3 Manuel d'Utilisation

Maintenant que vous êtes sur l'application, dans le menu principal, seul le bouton "Inventaire" fonctionnera. S'il n'y a pas d'erreur avec la base de données, une grande liste qui contient tout le stock doit s'ouvrir.

De là, vous avez 3 possibilités,

1. Filtrer le stock
2. Ajouter des articles
3. Modifier des articles

#### **Filtrer le stock**

Si vous souhaitez effectuer un filtre, les champs au-dessus des colonnes du stock sont à disposition pour cela. Veuillez entrer les données souhaitées pour un filtre puis cliquez sur le bouton situé sur la droite "Filtrer" pour lancer le filtre

#### **Ajouter des articles**

Si vous souhaitez ajouter des articles, vous devez tout simplement cliquer sur le bouton en haut à droite du stock qui s'appelle "Ajouter des articles". Une nouvelle fenêtre s'ouvrira avec différents champs à remplir. Si une erreur est présente sur le remplissage des champs l'application vous expliquera votre erreur.

Le bouton "Encore" permet de créer un article, tandis que le bouton "Quitter" permet seulement de quitter la page.

#### **Modifier des articles**

Si vous souhaitez modifier des articles, vous devez double-cliquer sur la ligne que vous souhaitez modifier dans le stock. La vue détaillée de cet article s'ouvrira et vous aurez maintenant la possibilité de modifier cet article.

Le bouton "Editer" permet de modifier les champs modifiables, oui parce que pas tous les champs sont modifiables, le code article par exemple. L'autre bouton, donc "Valider" permet d'effectuer les changements effectués. Bien évidemment, si une valeur est interdite, l'application vous le fera comprendre.

**Petites subtilités**

Voilà les petites subtilités lorsqu'on souhaite utiliser l'application sans souris.

Dans le menu principal, les touches "a" "s" "d" "f", représente chaque menu dans l'ordre. Donc "a" = Clients, "s" = Inventaire, "d" = Staff, "f" = Contrats.

Dans le stock, si vous appuyez sur "entrer" en ayant une ligne sélectionnée, cela simule un double-clic.

Si vous êtes en train de naviguer avec les flèches directionnelles dans le stock et que vous souhaitez effectuer un filtre, appuyez sur "echap" ou "escape".

## 5.4 Journal de travail

31/05/2022 08:39

Projet TPI Adam Gruber - Sploks

# Projet TPI Adam Gruber - Sploks

story	tâche	remarque	terminée_le	Totals
Ajouter des articles	Ajout d'un article schéma		2022-05-13	45.00
	Ajout d'un type		2022-05-24	200.00
	Ajouter des articles Vue		2022-05-12	180.00
	Documentation		2022-05-10	150.00
	Logique de l'ajout des articles		2022-05-13	340.00
	Pouvoir naviguer avec le clavier		2022-05-16	180.00
	Vue de l'ajout des articles		2022-05-24	50.00
Ajuster le nombre de pièces d'un article multiple	Ajuster le nombre de pièce dans le stock logique		2022-05-19	150.00
	Vue Ajuster le nombre de pieces		2022-05-17	60.00
Consulter le stock	Ajout de la base de donnée		2022-05-03	80.00
	Mise à jour de la vue		2022-05-03	30.00
Consulter les détails d'un article	Ajout du nombre de contrats		2022-05-09	40.00
	Documentation		2022-05-09	30.00
	Etat d'un article ajouté		2022-05-19	30.00
	Titres des fenêtres		2022-05-09	45.00
Filtrer le stock	Ajout de la touche entrer		2022-05-10	120.00
	Ajout du tabulateur		2022-05-10	60.00
	Documentation		2022-05-10	30.00
	Travail sur la logique du filtrage		2022-05-05	120.00
	Travail sur la vue du filtrage		2022-05-03	60.00
Mettre à jour l'état du matériel	Mise en place de la vue		2022-05-20	60.00

file:///Z:/TPI/dv/IceScrub-master (1)/IceScrub-master/Timesheet.html



31/05/2022 08:39

Projet TPI Adam Gruber - Sploks

story	tâche	remarque	terminée_le	Totals
	Travail sur la logique de la mise à jour de l'état du matériel		2022-05-20	120.00
Récurrence	Sprint review		2022-05-09	60.00
			2022-05-23	60.00
Urgente	Afficher les contrats		2022-05-06	150.00
	Afficher les détails d'un article		2022-05-06	120.00
	Ajustement de la vue détaillée d'un article		2022-05-24	60.00
	Analyse de la base de donnée		2022-05-06	90.00
			2022-05-09	100.00
	Analyse du sprint 2		2022-05-09	60.00
	Analyse sprint 3		2022-05-24	60.00
	Documentation		2022-05-02	260.00
			2022-05-17	360.00
			2022-05-24	510.00
	Garder le texte après avoir effectué un filtrage		2022-05-24	60.00
	MCD Sploks		2022-05-17	90.00
	Meeting avec Alain Roy		2022-05-02	90.00
	Meeting avec Mr Carrel		2022-05-02	30.00
	Mise en place de l'environnement		2022-05-02	90.00
	Planification initiale		2022-05-02	90.00
	Refactorisation du code		2022-05-16	230.00
	Réunion avec Mr. Carrel		2022-05-20	60.00
	Test de l'application en sa globalité		2022-05-19	60.00
	Trouver comment mettre le programme en .exe		2022-05-30	240.00
	Viste de l'expert Dimitros Lymberis		2022-05-16	60.00
			Totals	5,170.00

file:///Z:/TPI/jdt/IceScrub-master (1)/IceScrub-master/Timesheet.html