

Projet Sploks

Table des matières

1	Analyse préliminaire	3
1.1	Introduction	3
1.2	Objectifs.....	3
1.3	Planification initiale	3
2	Analyse / Conception.....	4
2.1	Concept	4
2.2	Stratégie de test.....	12
2.3	Risques techniques	12
2.4	Planification	13
2.5	Dossier de conception	13
3	Réalisation.....	14
3.1	Dossier de réalisation	14
3.2	Description des tests effectués	Erreur ! Signet non défini.
3.3	Erreurs restantes	18
3.4	Liste des documents fournis	18
4	Conclusions	18
5	Annexes.....	19
5.1	Résumé du rapport du TPI / version succincte de la documentation	19
5.2	Sources – Bibliographie.....	19
5.3	Journal de travail	20
5.4	Manuel d'Installation	20
5.5	Manuel d'Utilisation.....	20
5.6	Archives du projet.....	Erreur ! Signet non défini.

1 Analyse préliminaire

1.1 Introduction

Le projet a pour but de pouvoir gérer stock de matériel hiver (ski, chaussures, etc..). Il est réalisé dans le cadre du pré-TPI d'informaticien en 4^{ème} année. J'ai choisi ce projet car il pourrait m'apporter une touche de plus sur mon CV.

1.2 Objectifs

L'objectif principal serait d'avoir une application fonctionnelle. Mais il faut savoir que ce projet continuera sur le TPI.

- Pour le pré-TPI les objectifs seraient d'avoir une application qui va chercher les données dans la base de données MYSQL.
- Pouvoir modifier les informations dans les champs qui sont attribués pour cela.
- Les points au-dessus sont dédiés à l'interface « clients ».

Planification initiale

Sprint 1 : 07/02/2022 → 14/02/2022

- Rédiger une user story
- S'accorder sur les modalités de travail
- Livraison de documents de suivi

Sprint 2 : 15/02/2022 → 06/03/2022

- Navigation entre les menus
- Utilisation du clavier pour naviguer entre les menus

Sprint 3 : 07/03/2022 → 20/03/2022

- Mise en place de la base de données
- Importer la base de données dans l'application

Sprint 4 : 21/03/2022 → 01/04/2022

- Consultation des contrats de location existants
- Gérer le stock de matériel

Analyse / Conception

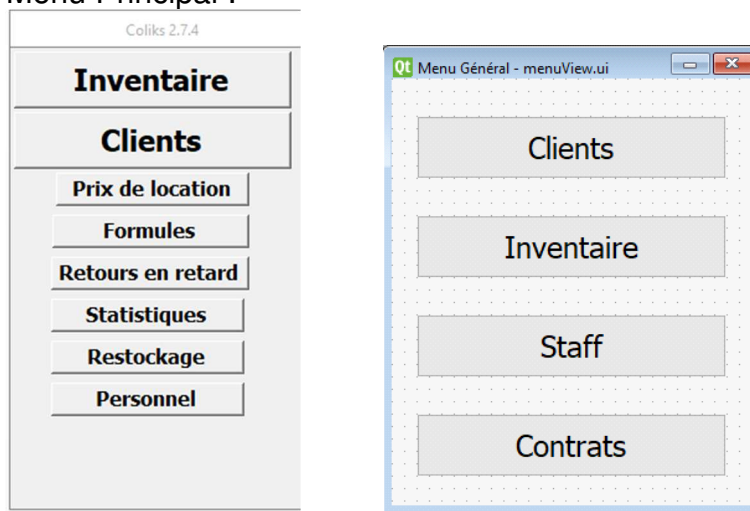
1.3 Concept

Sploks est la rénovation d'une application appelée Coliks, donc disons que les maquettes sont l'interface graphique de l'ancienne application.

La première image est la maquette de l'ancienne application « Coliks » et la deuxième sera la maquette de la nouvelle application « Sploks »

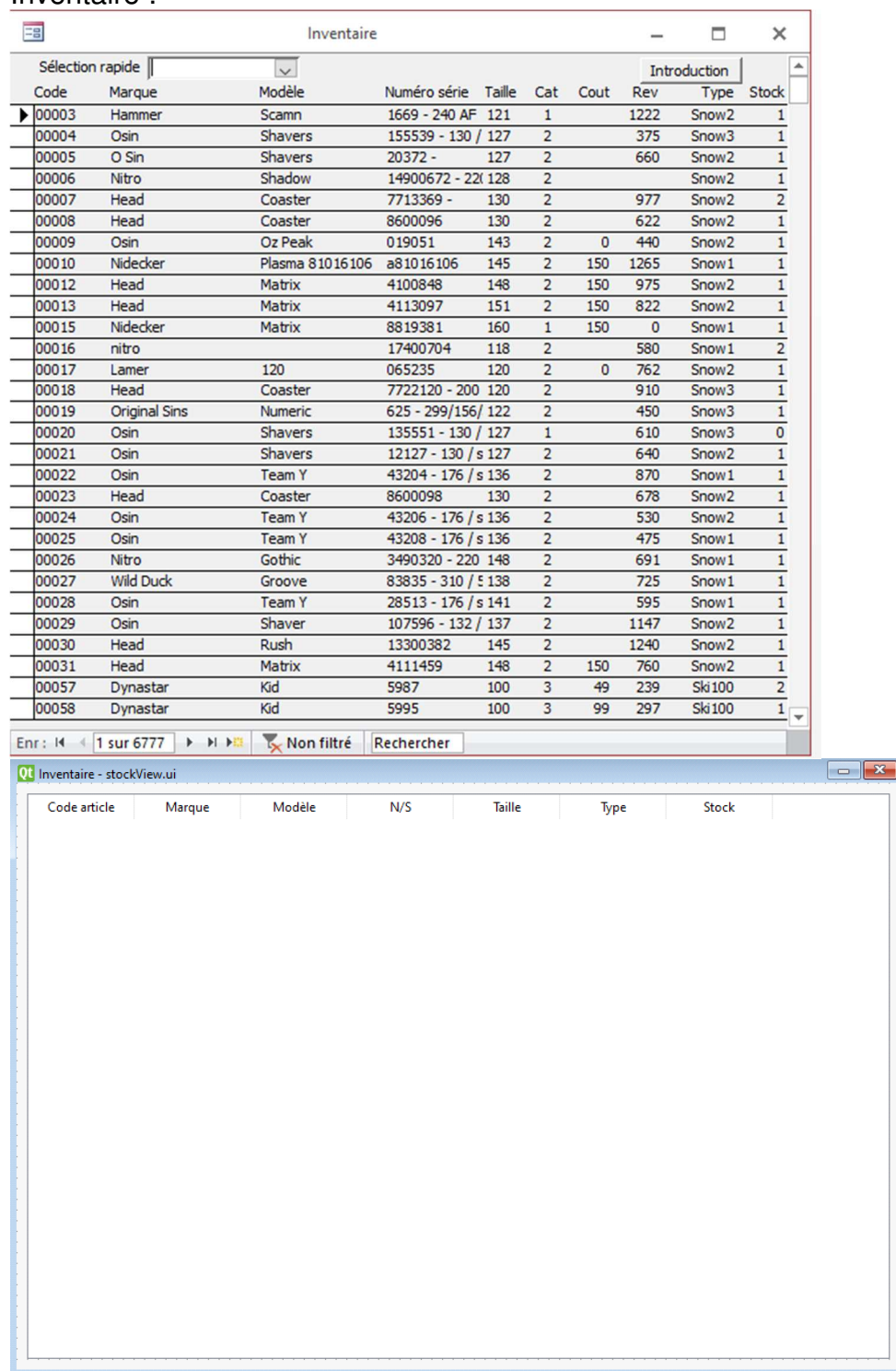
En dessous des images sera laissé un commentaire sur ces maquettes.

Menu Principal :



Le nouveau menu est du même type que l'ancien sauf qu'il est très simplifié au niveau des options disponibles. Bien évidemment pas toutes les interfaces ont été ajoutées dans la nouvelle applications, parce que le but est de simplifier un maximum celle-ci.

Inventaire :



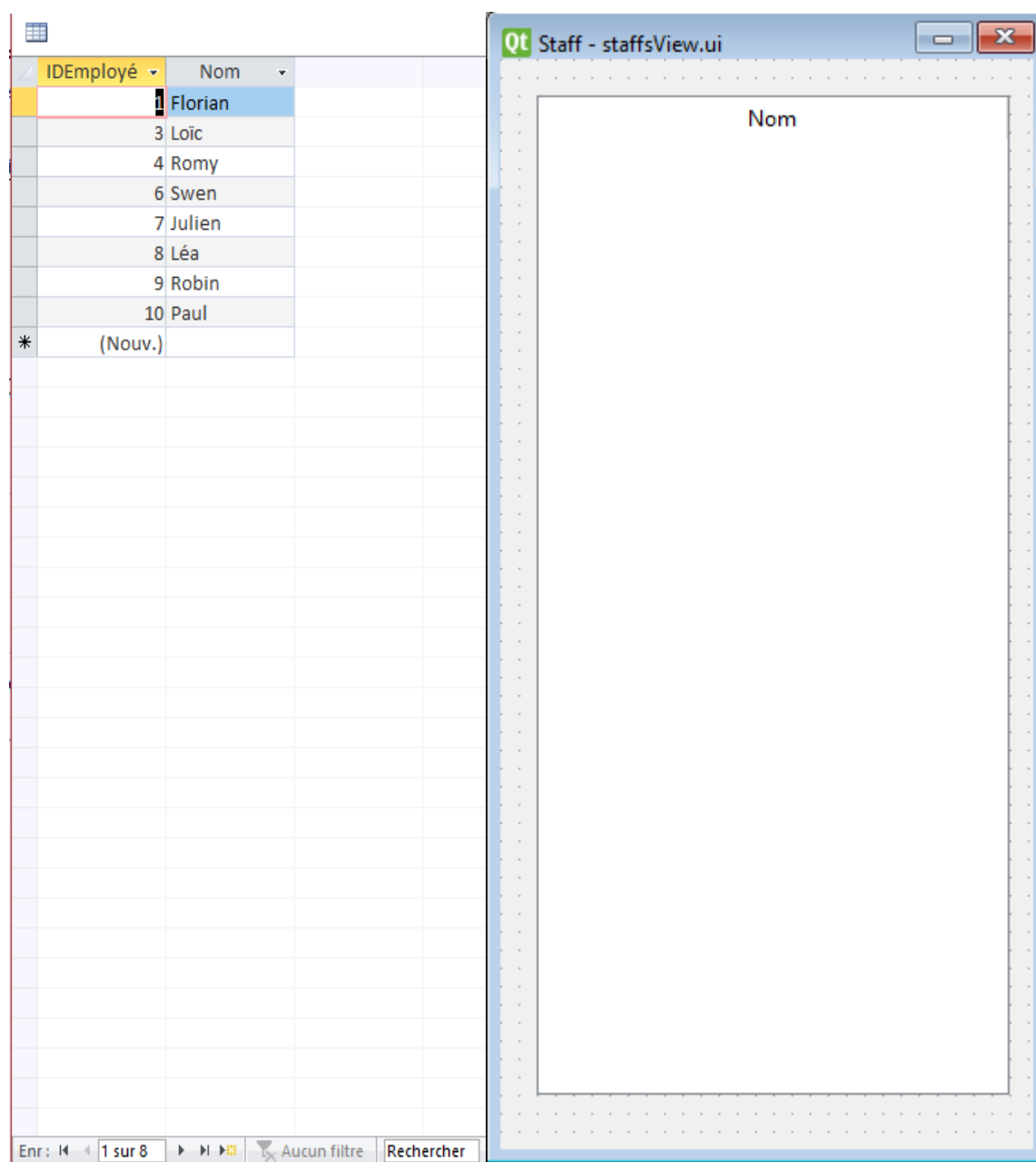
Code	Marque	Modèle	Numéro série	Taille	Cat	Cout	Rev	Type	Stock
00003	Hammer	Scamn	1669 - 240 AF	121	1		1222	Snow2	1
00004	Osin	Shavers	155539 - 130 /	127	2		375	Snow3	1
00005	O Sin	Shavers	20372 -	127	2		660	Snow2	1
00006	Nitro	Shadow	14900672 - 22	128	2			Snow2	1
00007	Head	Coaster	7713369 -	130	2		977	Snow2	2
00008	Head	Coaster	8600096	130	2		622	Snow2	1
00009	Osin	Oz Peak	019051	143	2	0	440	Snow2	1
00010	Nidecker	Plasma 81016106	a81016106	145	2	150	1265	Snow1	1
00012	Head	Matrix	4100848	148	2	150	975	Snow2	1
00013	Head	Matrix	4113097	151	2	150	822	Snow2	1
00015	Nidecker	Matrix	8819381	160	1	150	0	Snow1	1
00016	Nitro		17400704	118	2		580	Snow1	2
00017	Lamer	120	065235	120	2	0	762	Snow2	1
00018	Head	Coaster	7722120 - 200	120	2		910	Snow3	1
00019	Original Sins	Numeric	625 - 299/156/	122	2		450	Snow3	1
00020	Osin	Shavers	135551 - 130 /	127	1		610	Snow3	0
00021	Osin	Shavers	12127 - 130 / s	127	2		640	Snow2	1
00022	Osin	Team Y	43204 - 176 / s	136	2		870	Snow1	1
00023	Head	Coaster	8600098	130	2		678	Snow2	1
00024	Osin	Team Y	43206 - 176 / s	136	2		530	Snow2	1
00025	Osin	Team Y	43208 - 176 / s	136	2		475	Snow1	1
00026	Nitro	Gothic	3490320 - 220	148	2		691	Snow1	1
00027	Wild Duck	Groove	83835 - 310 / s	138	2		725	Snow1	1
00028	Osin	Team Y	28513 - 176 / s	141	2		595	Snow1	1
00029	Osin	Shaver	107596 - 132 /	137	2		1147	Snow2	1
00030	Head	Rush	13300382	145	2		1240	Snow2	1
00031	Head	Matrix	4111459	148	2	150	760	Snow2	1
00057	Dynastar	Kid	5987	100	3	49	239	Ski100	2
00058	Dynastar	Kid	5995	100	3	99	297	Ski100	1

Encore une fois l'inventaire est assez identique, c'est à dire que ces interfaces contiendront pratiquement les mêmes informations. Je dis pratiquement car dans l'application Coliks il y a par exemple le coût qui ne sera pas présent dans Sploks.

ID	Nom	Prénom	Adresse	NPA	Localité	Téléphone	Email	Mobile
----	-----	--------	---------	-----	----------	-----------	-------	--------

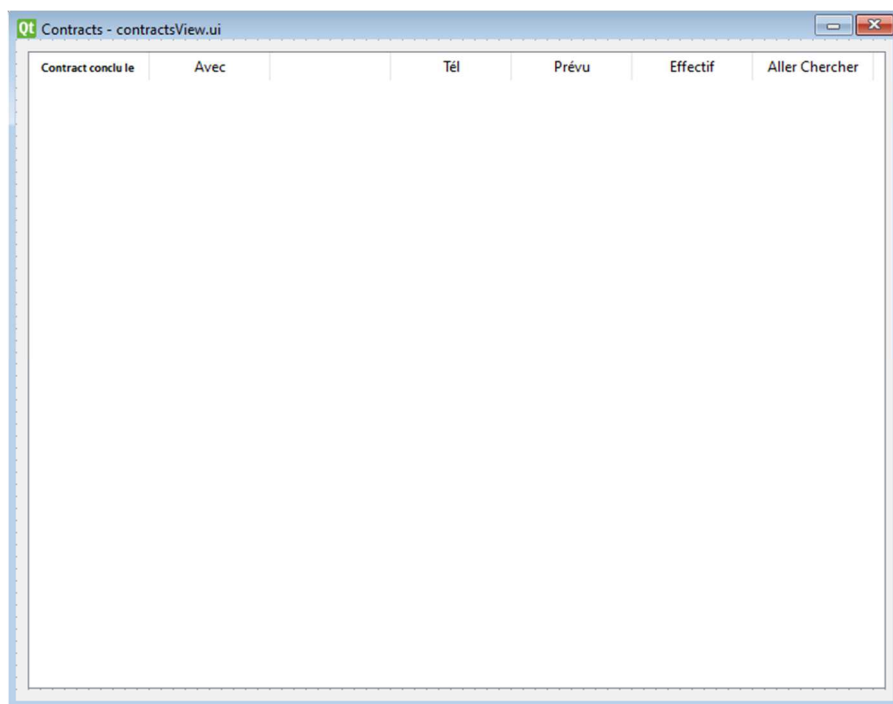
Dernière modif :

Staff :



Nous avons décidé d'enlever le « IDEmployé » et de simplifier un maximum dans Sploks.

Contrats :



Il n'y a pas de liste avec tous les contrats dans l'application Coliks, mais cette interface nous a été imposée ici dans Sploks. L'utilité de cette page sera de pouvoir trier tous les contrats par rapport à les colonnes que vous voyez ci-dessus.

Détail d'un contrat :

The screenshot shows a window titled 'Locations' with a 'Contrat' tab. It features search buttons 'Chercher Contrat Par Numéro' and 'Chercher Par Num Série'. Below are input fields for 'Nom', 'Prénom', 'Adresse', 'Localité', 'Tél (fixe)', 'Natel', and 'Email'. A table with columns 'Pos', 'Objet', 'Cat', and 'Prix' is present but empty. At the bottom, there are date fields for 'A retourner le', 'Pris le', and 'Payé le', a 'Total:' label, a checkbox for 'Rabais fin de saison (20%)', and 'Servi par' and 'Réglage par' dropdowns.

The screenshot shows a Qt application window titled 'MainWindow - contractDetailsView.ui'. It has a 'Contrat' tab with a contract number '2222'. Fields for 'Établi le' and 'Retourné le' are marked as 'TextLabel'. Below are buttons 'Vider' and 'Chercher Contrats de ce client', and a '(poubelle)' button. Input fields for 'Nom', 'Prénom', 'Tél (fixe)', 'Adresse', 'Localité', 'Natel', and 'Email' are arranged in two columns. A large empty text area is at the bottom.

Ces interfaces ont pour but de rentrer dans le détail d'un contrat, donc qui est titulaire de ce contrat etc... Grace à la nouvelle interface plus moderne on pourra obtenir les même informations qu'avant.

Détail d'un client :

Qt Detail Customer - customerDetailsView.ui

Prénom :	<input type="text"/>	NPA/Localité :	<input type="text"/>
Nom :	<input type="text"/>	Mobile :	<input type="text"/>
Adresse :	<input type="text"/>	Téléphone :	<input type="text"/>
		Email :	<input type="text"/>

Liste des contrats **Nb Contrats:** Annuler Supprimer Valider Editer

Coliks ne comporte pas d'interface détaillées d'un client. Il y a un bouton « Editer » avant de pouvoir effectuer des changements, afin d'éviter un maximum les erreurs humaines.

Détail d'un objet :

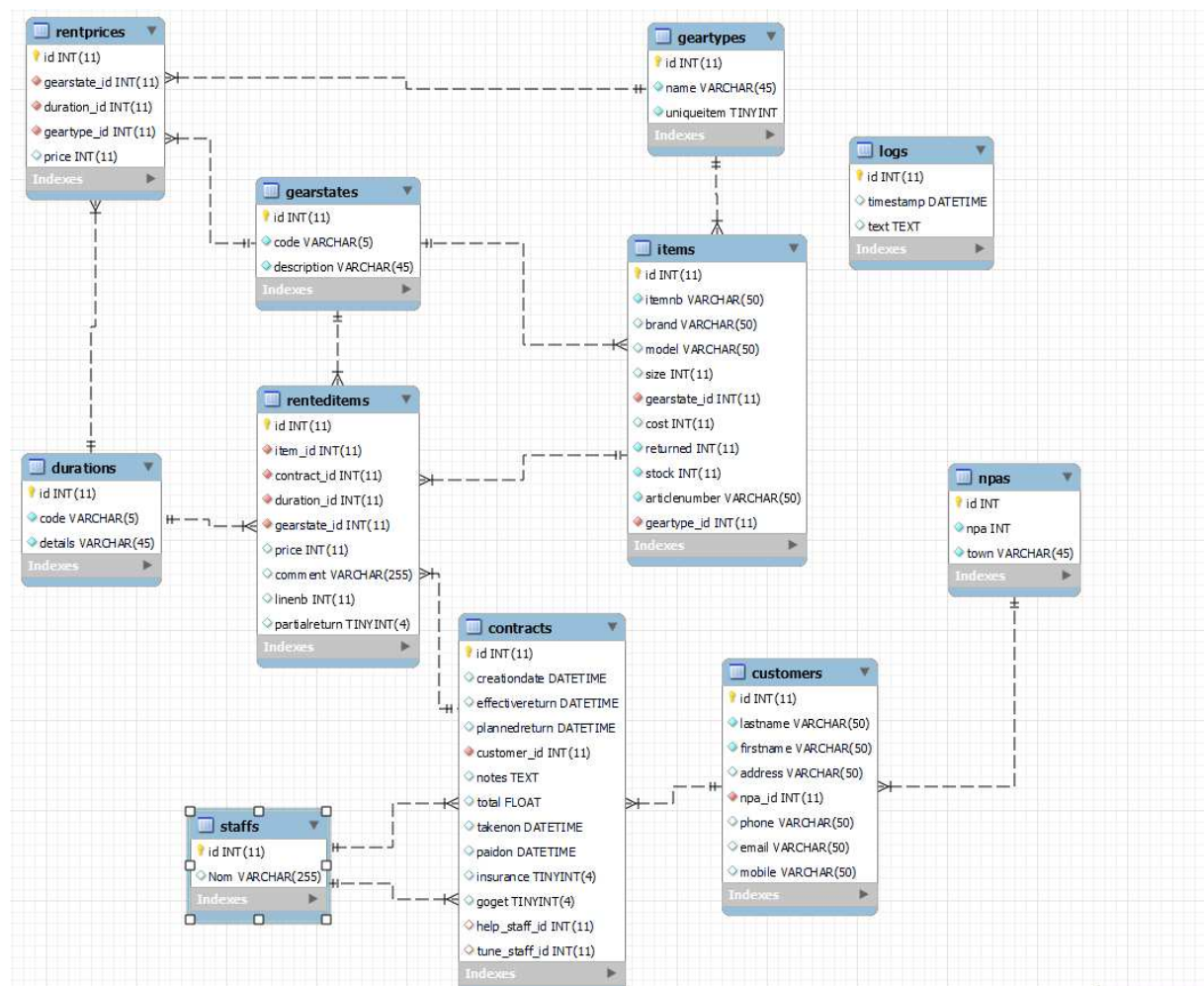
Qt Détail Article - itemDetailsView.ui*

Code d'article:	<input type="text"/>	Marque:	<input type="text"/>
Numéro de série:	<input type="text"/>	Modèle:	<input type="text"/>
Prix d'achat:	<input type="text"/>	Type:	<input type="text"/>
Revenues générés:	<input type="text"/>	Stock:	<input type="text"/>
		Taille:	<input type="text"/>

Liste des Contrats **Nb Contrats:** TextLabel

La dernière maquette existera aussi que sur Sploks. Il s'agit de la vue détaillé d'un objet. En bas il y a un bouton qui permet de lister les contrats par rapport à cet objet.

Monsieur Carrel nous à fournit une base de données fonctionnel avec laquelle nous devons exécuter notre projet. Il nous a ainsi fournit le modèle conceptuel de cette Base de données.



Il est important de noter la table « logs » qui n'est pas reliée aux autres. Elle permettra de noter quel utilisateur à fait quel modifications etc...

1.4 Stratégie de test

1. Lancer l'application
 - . Un menu devrait apparaître
2. Cliquer sur « clients »
 - . La liste de tous les clients devrait apparaître
3. Double cliquer sur un client
 - . L'inspecteur de tous les clients devrait apparaître
 - . Les champs devraient pas pouvoir être éditables tant que l'on a pas cliqué sur le bouton « Editer »
4. Cliquer sur le bouton éditer
 - . Les boutons « Annuler », « Supprimer » et « Valider » Devraient être à présent cliquable.
 - . Les champs devraient aussi être éditables
5. Changer le champ prénom avec « Fleur » (sans valider)
6. Cliquer sur annuler
 - . Le prénom qui était présent avant devrait réapparaître
7. Changer le champ prénom avec « Fleur » et cliquer sur valider
 - . La fenêtre devrait se fermer
 - . La fenêtre qui comporte tous les clients devrait réapparaître devant.
8. Fermer cette fenêtre et la rouvrir afin qu'elle se mette à jour.
9. L'ancien client que vous avez modifié devrait avoir comme prénom « Fleur »

1.5 Risques techniques

Ce projet demande de maîtriser pyqt5, une librairie python qui permet de faire des interfaces visuelles. Chaque librairie à sa manière de marcher, donc c'était un peu complexe au début de la maîtriser. Mais à présent je me sens beaucoup plus à l'aise avec ce dernier.

Concernant les priorités, j'ai essayé de prioriser la communication avec la base de donnée, donc en perdant de l'effort sur l'interface utilisateur.

1.6 Planification

Début de projet 07/02/2022 -> Fin du Projet 01/04/2022

Phase 1 : 07/02/2022 -> 14/02/2022

Phase 2 : 15/02/2022 -> 06/03/2022

Phase 3 : 07/03/2022 -> 20/03/2022

Phase 4 : 21/03/2022 -> 01/04/2022

Aucun partage de tâche a été effectué car c'est un projet individuel

1.7 Dossier de conception

Contenu détaillé de la base de données :

Table items

Cette table contient tout le stock des objet (item).

Elle comporte 2 clé étrangères, celle de l'état du matériel et celle du type de matériel.

Table gearstate

Cette table contient les différents états des matériaux.

1 = Terminé

2 = Neuf

3 = Usagé

4 = Vieux

Table geartypes

Cette table contient les différent type d'équipement disponibles

Table rentprices

Cette table contient les prix des locations

Elle contient 3 clés étrangères

- L'état du matériel
- La durée de la location
- Le type de matériel

Table durations

Cette table contient les différentes durées

1 = 1 semaine

2 = 2jours

3 = 2semaines

...

Table rented items

Cette table contient le matériel loué

Elle contient 4 clés étrangères

- Le matériel loué
- Le contrat attaché
- La durée de la location

- L'état du matériel

Table contracts

Cette table contient les contrats des clients

Elle contient 1 clé étrangère

- Le client auquel appartient le contrat

Table customers

Cette table contient tous les clients

Elle contient 1 clé étrangère

- Le code postal des clients

Table npas

Cette table contient les différents codes postaux existant

Table staffs

Cette table contient les différentes personnes du staff

Table logs

Cette table contient les différentes manipulations qui ont été effectuées dans l'application.

Exemple de requêtes dans l'application :

Quand l'utilisateur va cliquer sur un client, ce qui va permettre d'ouvrir l'inspecteur, il faut obtenir toutes les informations du client en question.

C'est là qu'on va exécuter une requête qui va prendre toutes les données d'un client par son ID.

```
(SELECT sploks.customers.*, sploks.npas.npa FROM sploks.customers LEFT JOIN  
sploks.npas ON sploks.customers.id = sploks.npas.id WHERE sploks.customers.id  
= {id};)
```

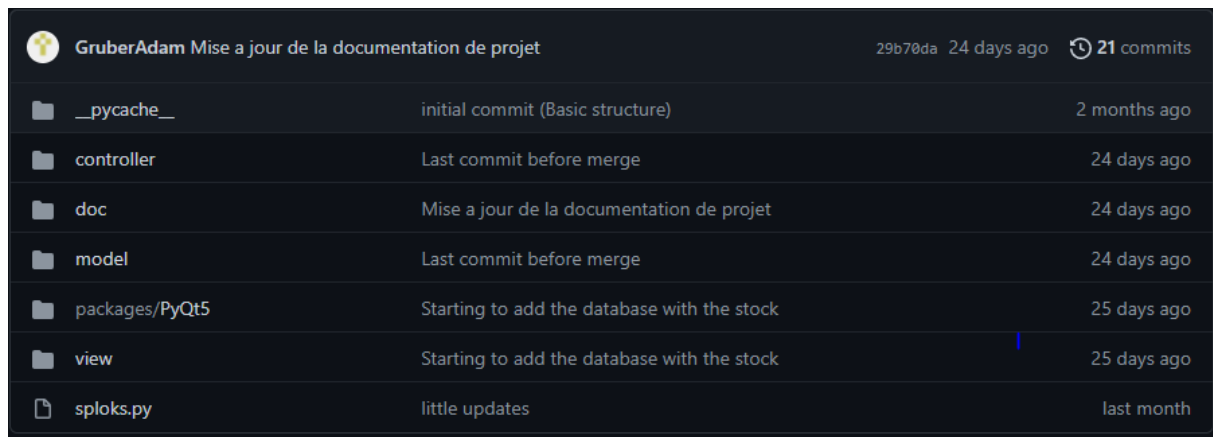
Bien évidemment le « {id} » est remplacé par le bon id du client.

2 Réalisation

2.1 Dossier de réalisation

Tout le fichier / dossier se trouvent sur GitHub sur la branche « develop »

(<https://github.com/GruberAdam/Sploks/tree/develop>)



GruberAdam Mise a jour de la documentation de projet		29b70da 24 days ago 21 commits
__pycache__	initial commit (Basic structure)	2 months ago
controller	Last commit before merge	24 days ago
doc	Mise a jour de la documentation de projet	24 days ago
model	Last commit before merge	24 days ago
packages/PyQt5	Starting to add the database with the stock	25 days ago
view	Starting to add the database with the stock	25 days ago
sploks.py	little updates	last month

Maintenant passons aux commentaires sur chaque fichier / dossier :

A la racine

__pycache__ : Est dossier qui contient des caches que python crée pour lancer le programme un peu plus rapidement.

controller : Dossier qui va contenir tous les fichiers qui effectuent les redirections de l'application.

Doc : Dossier qui contient toute la documentation de projet.

Model : Dossier qui va contenir tous les fichiers qui communiquent avec la base de donnée.

Packages/PyQt5 : Contient tous les packages qui concernent l'application.

View : Dossier qui contient toutes les interfaces visuelles de l'application.

Sploks.py : Point d'entrée de l'application Sploks.

Dans controller :

GruberAdam Last commit before merge			20d5967 24 days ago	History
..				
__pycache__	Last commit before merge		24 days ago	
contractsController.py	Every button opens the right ui		last month	
customersController.py	Customer data is now loaded in the UI		27 days ago	
keyPressController.py	little updates		last month	
mainMenuController.py	Connection with the database working, and data of the customers is in...		28 days ago	
staffController.py	Every button opens the right ui		last month	
stockController.py	Starting to add the database with the stock		25 days ago	

contractsController.py : Effectue toutes les redirections concernant les contrats.

customerController.py : Effectue toutes les redirections concernant les clients.

keyPressController.py : Va faire toutes les redirections selon touches pressées.

mainMenuController.py : Effectue toutes les redirections sur le menu principal.

staffController.py : Effectue toutes les redirections concernant le staff.

stockController.py : Effectue toutes les redirections concernant le stock.

Dans Doc :

GruberAdam Mise a jour de la documentation de projet			29b70da 24 days ago	History
..				
IceScrub-master	initial commit (Basic structure)		2 months ago	
CdC Adam.pdf	initial commit (Basic structure)		2 months ago	
Documentation de projet.pdf	Mise a jour de la documentation de projet		24 days ago	
Journal de travail.pdf	Last commit before merge		24 days ago	

IceScrub-master : Application qui permet de créer le journal de travail.

Cdc Adam.pdf : Le cahier des charges de cette application

Documentation de projet.pdf : Le fichier que vous êtes en train de lire.

Journal de travail.pdf : Le journal de travail

Dans Model :

GruberAdam Last commit before merge			20d5967 24 days ago	History
..				
__pycache__	Last commit before merge		24 days ago	
customersModel.py	Customer data is now loaded in the UI		27 days ago	
database.py	Customer data is now loaded in the UI		27 days ago	
stockModel.py	Starting to add the database with the stock		25 days ago	

customersModel.py : Effectue toutes les requêtes sur la base de donnée concernant les clients

database.py : Effectue la connexion avec la base de donnée

stockModel.py : Effectue toutes les requêtes sur la base de donnée concernant le stock

Dans Packages/PyQt5 :

- PyQt5 (Librairie python qui permet de faire des interfaces utilisateurs)
- MySQL connector (Librairie python qui permet de communiquer avec la base de donnée)

Dans View :

GruberAdam Starting to add the database with the stock			eea82a6 25 days ago	History
..				
contractDetailsView.ui	Added different views from louis's project		last month	
contractsView.ui	Update of the views		last month	
customerDetailsView.ui	Added different views from louis's project		last month	
customersView.ui	Customer data is now loaded in the UI		27 days ago	
itemDetailsView.ui	Added different views from louis's project		last month	
menuView.ui	initial commit (Basic structure)		2 months ago	
staffsView.ui	Update of the views		last month	
stockView.ui	Starting to add the database with the stock		25 days ago	

contractDetailsView.ui : interface utilisateur des détails des contrats

contractsView.ui : interface utilisateur des contrats

customerDetailsView.ui : interface utilisateur de la vue détaillée des clients

customersView.ui : interface utilisateur des clients

itemDetailsView.ui : interface utilisateur du détail des objets

menuView.ui : interface utilisateur du menu principal

staffsView.ui : interface utilisateur du staff

stockView.ui : interface utilisateur du stock

Version du projet : 1.0

2.2 Erreurs restantes

Lorsqu'on met à jour un client, la fenêtre de la vue détaillée se ferme.

Sur l'interface des clients les données mise à jour ne sont pas à jour.

Ce qui est considéré comme un problème.

Pour remédier à ce problème il faut trouver la fonction qui met à jour l'affichage d'une fenêtre.

Les conséquences de ce problème ne sont pas très grand, car si on ferme la fenêtre des clients et on la rouvre, l'affichage sera à jour.

2.3 Liste des documents fournis

- La documentation de projet
- Cahier des charges
- Journal de travail
- Lien vers le repository GitHub

3 Conclusions

Objectifs atteints :

- Tous les clients de la base de donnée sont affichés.
- L'inspecteur d'un seul client marche.
- La modification d'un client marche (La logique).

Objectifs non atteints :

- Les interfaces ne sont pas très propre.
- La fenêtre des clients ne se met pas à jour toute seule.
- Il n'est pas possible de supprimer un client.

Points positifs :

Etant donné que mon TPI sera aussi sur PyQt5 je serais déjà prêt de comment marche cette librairie.

Points négatifs :

Je n'ai pas réussi à tenir à jour mes fichiers de documentations.

Difficultés particulières :

La communication régulière des différents fichiers qui ne concernent pas la programmation est très difficile pour moi.

Mais je pense la plus grosse difficulté pour moi c'est le fait d'être en maturité, ça n'aidera pas non plus pour mon TPI.

Suites possibles pour le projet :

Sur le long terme, ça serait de ne pas seulement rester sur l'interface des clients mais d'en faire une application complète. Bien évidemment avant il faut régler les bugs actuels.

4 Annexes

4.1 Résumé du rapport du TPI / version succincte de la documentation

4.2 Sources – Bibliographie

En tant que développeur je ne pourrais pas vous citer toutes les sources qui m'ont aidé mais je vais vous donner les principales.

- <https://stackoverflow.com/> (Pour les erreur en python)
- <https://www.riverbankcomputing.com/static/Docs/PyQt5/> (Pour des questions concernant la librairie pyqt5)
- <https://dev.mysql.com/doc/connector-python/en/> (Pour communiquer avec la base de donnée MySQL)

4.3 Journal de travail

Mon journal de travail se trouve sur mon GitHub (<https://github.com/GruberAdam/Sploks>) dans le dossier « doc » il y a un document appelé Journal de travail.pdf

4.4 Manuel d'Installation

Pour installer l'application elle se fera en 3 étapes.

1. La première sera d'installer python3
2. La deuxième sera de télécharger mon repository GitHub
3. La dernière étape sera de télécharger mysql et effectuer les configurations nécessaires.

INSTALLER PYTHON3

Pour cette étape je vous invite à vous rendre sur le site <https://www.python.org/downloads/> et de télécharger la dernière version qui n'est pas en test.

Pour tester si python a bien été installé il faut exécuter la commande pip dans le CMD.

Installer MySQL Workbench

Installez mysql workbench avec les configurations de base.

Lancez le script disponible sur Github (<https://github.com/GruberAdam/Sploks>) sql/script.sql

Une fois exécuté il vous reste une seule étape, veuillez changer le fichier Model/database.py avec les identifiants de votre base de donnée.

Pour lancer le program il suffit d'effectuer un py sploks.py à la racine.

4.5 Manuel d'Utilisation

L'utilisation de l'application est plus simple que vous le pensez. Grâce à votre souris vous pouvez utiliser le click gauche pour naviguer entre les différents menus de l'application. Si vous souhaitez changer des données n'hésitez pas à utiliser votre clavier afin de pouvoir les mettre à jour dans la base de donnée.