UNIVERSITE PARIS 8
Vincennes - Saint-Denis

RAPPORT DU PROJET : GESTION DU STOCK D'UN MAGASIN

Matière : outils libres pour le développement logiciel

Membres :

Kolman Jean-Marie DANTODJI

Sarah KADI

Jessica HENAO HENAO

Massinissa CHETTI

UNIVERSITE PARIS 8 Vincennes - Saint-Denis

RAPPORT DU PROJET : GESTION DU STOCK D'UN MAGASIN

Description de l'application :

L'application a pour but de gérer le stock d'un « magasin » et de modifier sa base de données après avoir entré un « reçu d'achats » des articles.

Chaque article dans la base de données est identifié par un nom, la quantité existante et le prix à l'unité. L'application contient 10 produits différents ; des cahiers, stylos, gommes, crayons, ciseaux, compas, surligneurs, équerres et rapporteurs.

Langages utilisés :

- Base de données : SQL
- Gestion de la base de données et de l'interface : python

Logiciels utilisés :

- PyCharm
- Sqlite
- Github
- Visual Studio Code
- Tkinter

Structure du programme :

Explication du code :

- **Fichier main.py** : le fichier permet d'effectuer la liaison entre la base de données, le fichier lu en entrée et l'interface.

Fonctions	Objectif
openingFile()	Ouvre le fichier « products.txt » le lis, et conserve son contenu.
displayBDD()	Affiche la base de données qui contient les produits du magasin.
updateBDD()	Modifie la base de données suite à la lecture des produits achetés.
notifyEmptyStock()	Envoie d'une notification si le stock d'un produit est bientôt épuisé.
fillStock()	Recharge le stock des produits dont la quantité est épuisée.
searchArticle()	Recherche un article dans la base de données.

- **Fichier controller.py** : le fichier gère les fonctions liées à l'interface.

Funtion	Objectif
charge_data_from_file()	Affiche les informations du fichiers « products.txt » sur l'interface.
update_data_by_receipt()	Affiche les produits dans l'interface.
search_product()	Cherche un produit dans la base de données.
add_or_update()	Modifie la base de données en ajoutant un produit ou en mettant à jour les modifications des quantités de produits.
display_datas()	Affiche le résultat du bouton « add or update »
display_rupture()	Affiche la notification sur l'interface en cas de rupture de stock.

Création de la base de données et de la table :

Les fichiers « **initDB.py** » et « **schema.sql** » se chargent de créer la table « magasin » dans la base de données, avec les champs, types de données et restrictions respectifs.

```
create table "magasin" (
cle INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
produit NVARCHAR(160)  NOT NULL,
quantite INTEGER  NOT NULL,
prix FLOAT  NOT NULL
)
```

Mode opératoire de l'application :

Pour exécuter l'application, la base de données doit être chargée. Cela peut être fait directement à partir du terminal en utilisant la commande suivante :

```
~ / < > ProjetGestionDeStock $ python initDB.py
```

Ensuite, pour lancer l'application, la commande a entré est la suivante :

```
~ / < > ProjetGestionDeStock $ python frontend.py
```

Lorsque l'application est exécutée, le menu principal apparait, offrant à l'utilisateur cinq options comme montré ci-dessous dans la figure 1 :

1. Display articles.
2. Charge receipt.
3. Validate.
4. Search.
5. Add or update.

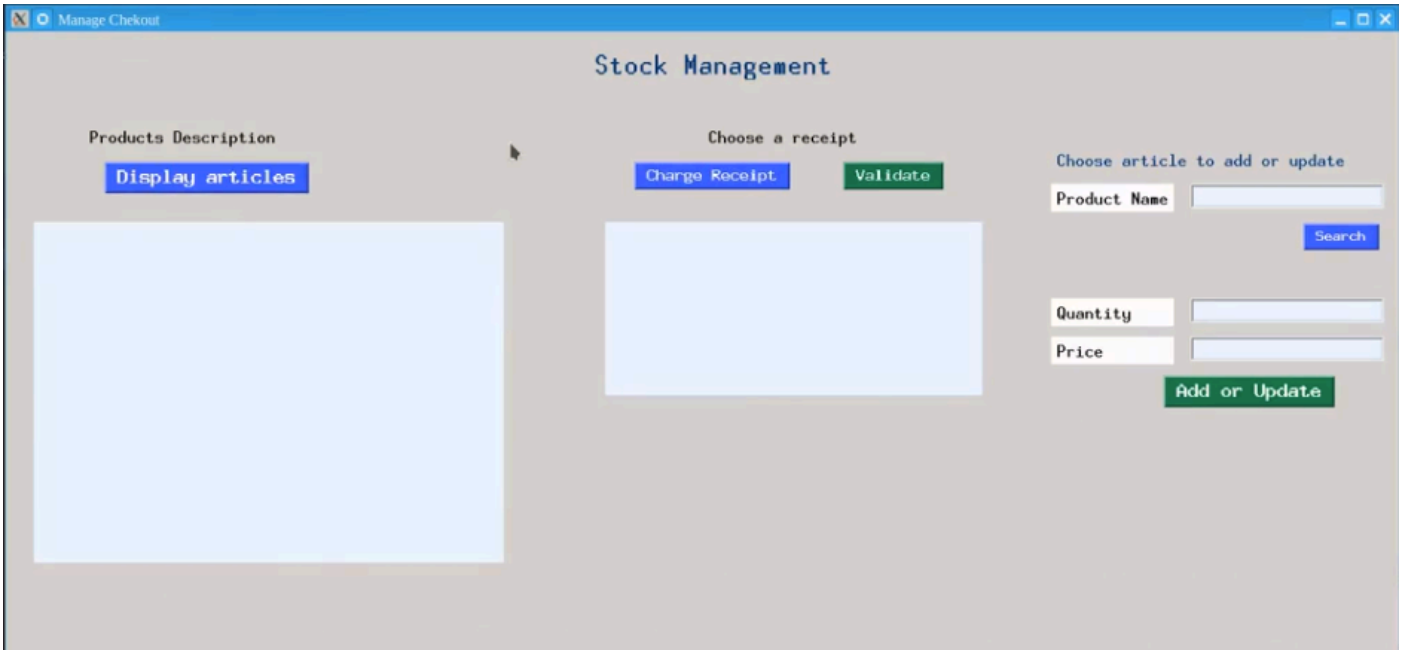


Figure 1 : présentation de l'écran d'accueil de l'application

En cliquant sur le bouton « Display articles », la base de données s'affiche et nous présente tous les produits disponibles, leur quantité et le prix à l'unité comme montré sur la figure 2.

Number	Product Name	Quantity	Price
1	CAHIER	100	1.9€
2	STYLO	100	1.2€
3	GOMME	10	1.3€
4	CRAYON	100	1.5€
5	CISEAUX	100	1.6€
6	COMPAS	100	1.7€
7	SURLIGNEURS	100	2.2€
8	REGLE	100	1.1€
9	EQUERRE	100	1.2€
10	RAPPORTEUR	100	1.5€

Figure 2 : présentation des articles de la BDD

Le bouton « Charge Receipt » offre la possibilité de sélectionner le reçu d'achat comme monté ci-dessous :

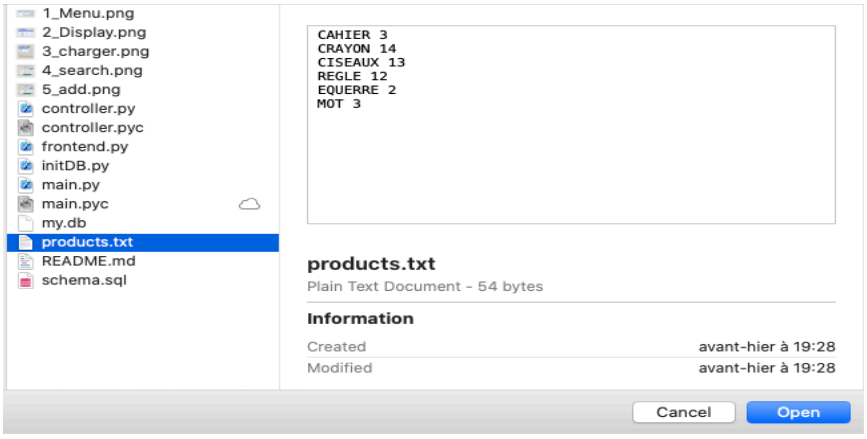


Figure 3 : présentation du bouton « Charge Receipt »

Le bouton « Validate » permet de mettre à jour la base de données suite à la lecture du reçu et décrémente donc, la quantité de chaque produit qui a été acheté.

Si la quantité d'un produit est inférieur à 10, l'application affichera la notification suivante :

« Attention, le produit « Règle » est bientôt épuisé » et s'il n'y a pas de produit en stock la notification sera : « Attention, le produit « Règle » est épuisé », comme montré ci-dessous :

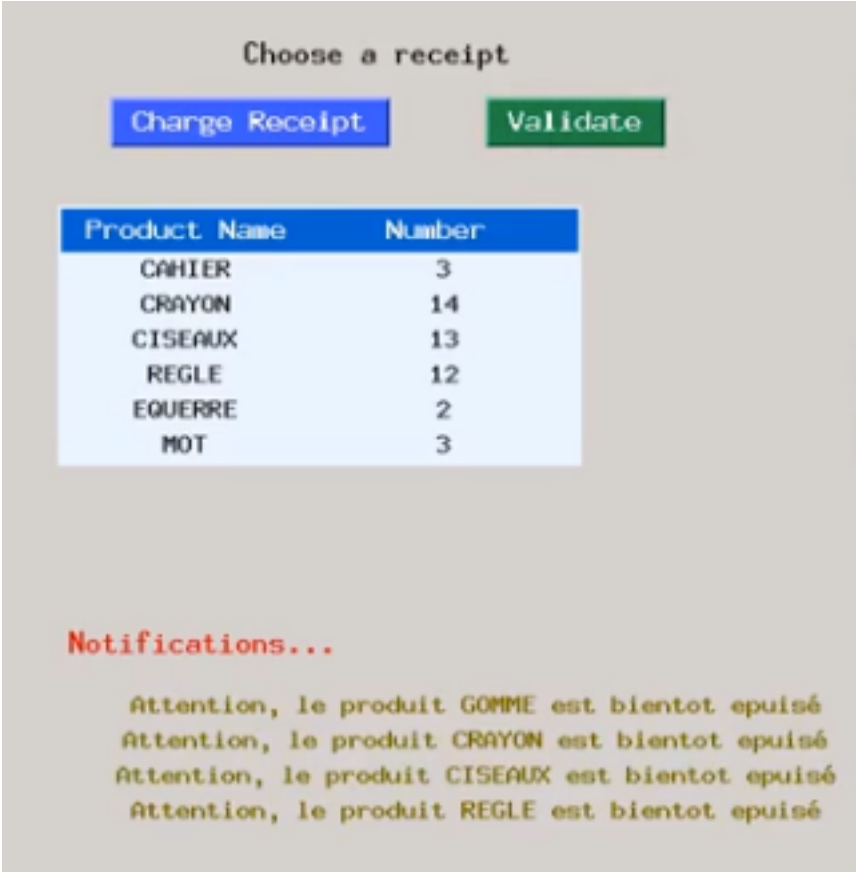
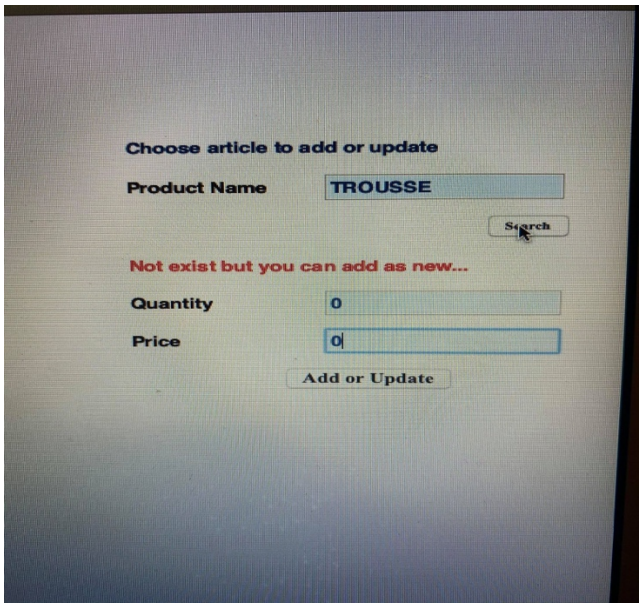


Figure 4 : présentation des notifications en cas de rupture de stock

Le bouton « Search » permet de chercher un produit dans la base de données, une notification s'affiche si le produit est inexistant et nous indique qu'on peut le rajouter à la base de données grâce au bouton « Add or Update ».

La recherche d'un article permet de connaitre la quantité présente en stock et le prix de ce dernier, cela permet également de changer la quantité si jamais le stock est épuisé et la remettre donc à son nombre initial dans la base de données.

Le nom est unique, un article ne peut donc pas être ajouté deux fois à la base de données, seuls la quantité et le prix peuvent être modifiés.



Number	Product Name	Quantity	Price
1	CAHIER	82	1.9€
2	STYLO	100	1.2€
3	GOMME	110	1.3€
4	CRAYON	16	1.5€
5	CISEAUX	22	1.6€
6	COMPAS	100	1.7€
7	SURLIGNEURS	100	2.2€
8	REGLE	73	4.0€
9	EQUERRE	88	1.2€
10	RAPPORTEUR	100	1.5€
11	TROUSSE	100	5.0€

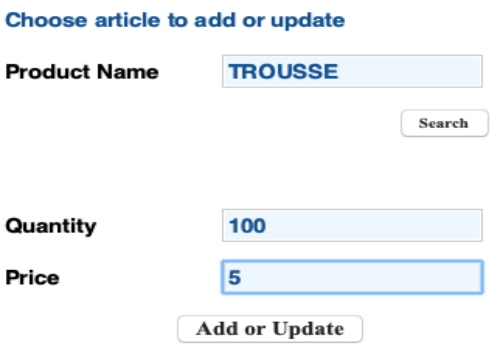


Figure 5 : présentation de l'ajout d'un produit dans la base de données

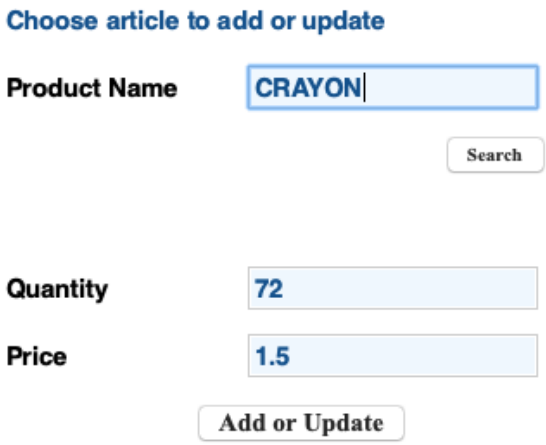


Figure 6 : présentation de la notification en cas de produit trouvé dans la base de données.

Le lien suivant : https://jmkd.fr/projects_demos/project_manage_stock.mp4 permet de visionner une brève vidéo présentant l'application et ses différentes fonctionnalités.