Bibliothèque de gestion de DB pour le projet Goods

```
#Pour la gestion de produits
from database.products import Products, Product
#Pour la gestion d'utilisateurs
from database.users import User

product = Product.create(name="Figurine Mario",
                         description="Une petite figurine... etc",
                         price=10.67,
                         quantity=34)
# Tous les autres champs que `name` sont facultatifs. De plus, deux produits ne
peuvent pas avoir le même nom.

print(product.name)
#Sortie : Figurine Mario

product.price = 23.45
#Tous les champs peuvent être modifiés librement sauf le nom qui devra ne pas
déjà exister

product.delete()
#Suppression du produit




search_results = Products.search("mario")
#Renvoie tous les produits dont le nom ou la description contiennent "mario",
ou toutes les combinaisons de majuscules et de minuscules possibles ("Mario",
"mArIo", "MARIO", etc...)

search_results = Products.search("mario", sort_by="price")
#Idem, mais trié dans l'ordre de prix croissant
#Les valeurs possibles sont "name", "description", "price", "quantity"

search_results = Products.search("mario", sort_by="price", desc=True)
#Idem, mais dans l'odre décroissant

search_results = Products.search("mario", min_price=3.45, max_price=34.02)
#Bornes de prix

search_results = Products.search("", sort_by="price")
#On peut aussi ne pas chercher de mot en particulier, dans ce cas là, le
résultat sera tous les produits triés.

search_results = Products.search_by_name("mario")
search_results = Products.search_by_description("mario")
#Mais ces deux fonctions n'implémentent pas les bornes de prix
```

```
search_results.filter_by_price(min_price=3.89, max_price=15.43)
#Une fois les résultats obtenus, on peut encore utiliser les bornes de prix

for product in search_results:
    product.price = 6.65
#Chaque produit dans les résultats a les propriétés décrites plus haut.

first_product = search_results[0]
#On peut aussi utiliser les résultats comme une liste




user = User.create(pseudo="Guiraan23",
                   password="dcdd32996ea69946e3fe50c6dfe1cd7d", #en hash MD5 !
Attention à la sécurité, le mdp ne devra jamais être stocké en clair.
                   description="etc...")
#La description est facultative, et deux utilisateurs ne peuvent pas avoir le
même pseudo.

user = User.verify_password(pseudo="Guiraan23",
password="dcdd32996ea69946e3fe50c6dfe1cd7d")
#Si les données sont incorrectes, rien ne sera renvoyé (None).
if user is not None:
    print(user.pseudo)
#Sortie : Guiraan23

user.password = "08ff3d1cae789621a0edce358ec8fb95"
#Idem que pour les produits, et le pseudo ne peut pas être changé si un autre
utilisateur le possède déjà.

user.delete()
#Suppression de l'utilisateur.
```