**PROJET FINAL DATA**

**B2 G1 GROUPE 4 :**

**DIALLO Sogona Yasmine**

**DJEDJE ELOHIM**

**BAKARI ZIDANE**

**NIANGADOU MAMADOU**

## ****1. Scraping des données →**** scraper.py

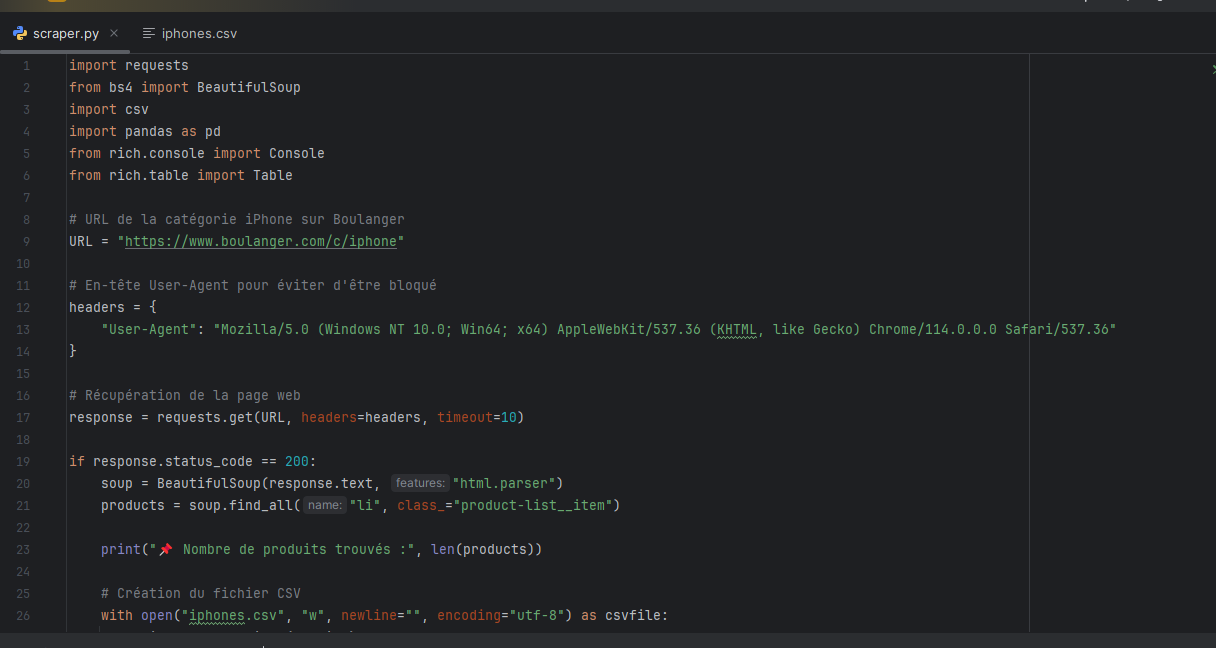
**Objectif :**  
Ce fichier est responsable de **l'extraction des informations** sur les iPhones depuis le site web **Boulanger**.

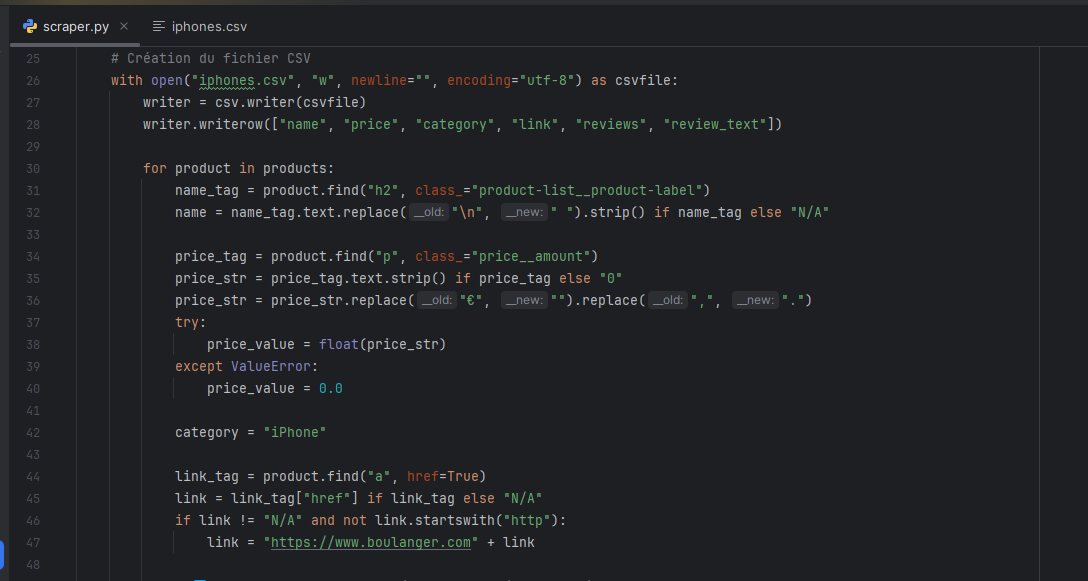
**Fonctionnalités :**

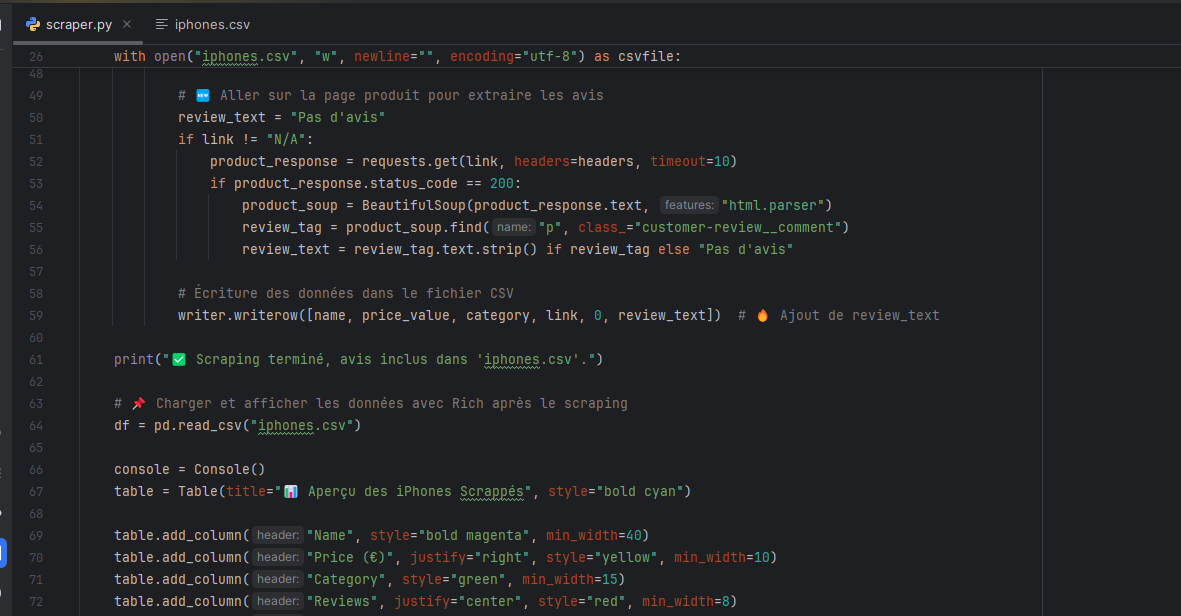
* Utilisation de requests pour récupérer le contenu HTML du site.
* Analyse du HTML avec BeautifulSoup pour extraire les informations.
* Récupération des données principales :
  + **Nom du produit** (h2.product-list\_\_product-label)
  + **Prix** (p.price\_\_amount)
  + **Lien vers la fiche produit** (a[href])
  + **Nombre d’avis** (span.rating-count)
* Stockage des données dans un fichier CSV (iphones.csv).

**Explication du code :**

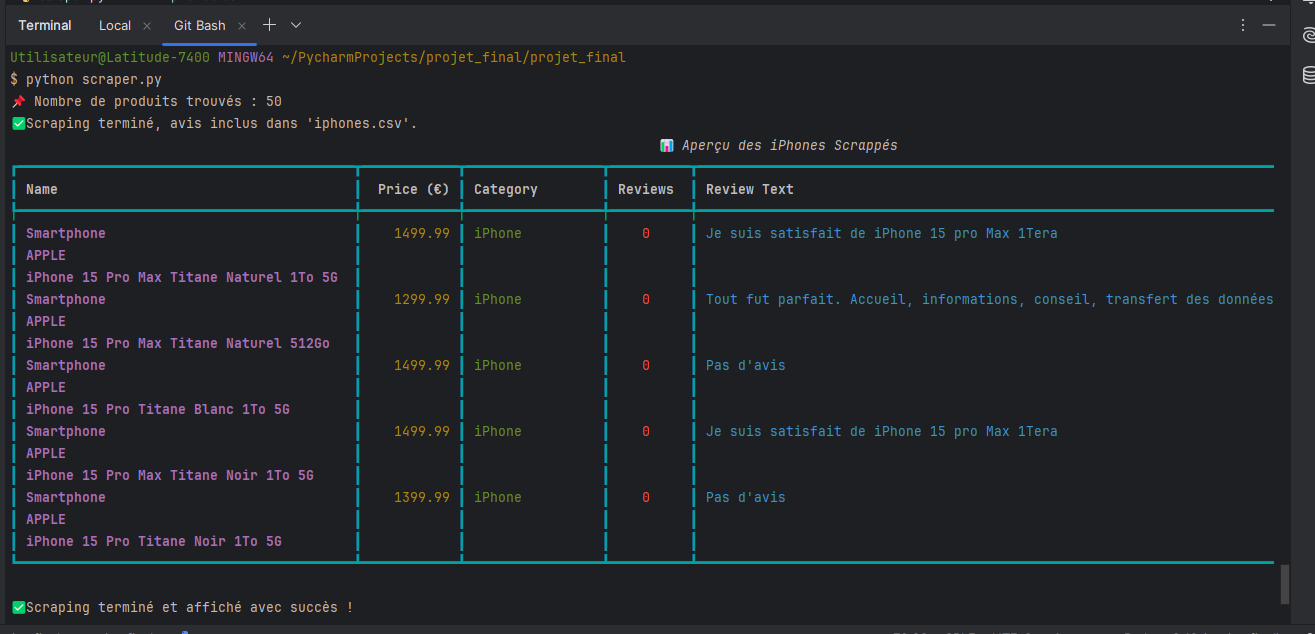
1. **Récupération de la page HTML** avec requests.get(URL, headers=headers).
2. **Parsing du HTML** avec BeautifulSoup.
3. **Boucle sur chaque produit** pour extraire les informations.
4. **Écriture des données** dans iphones.csv.



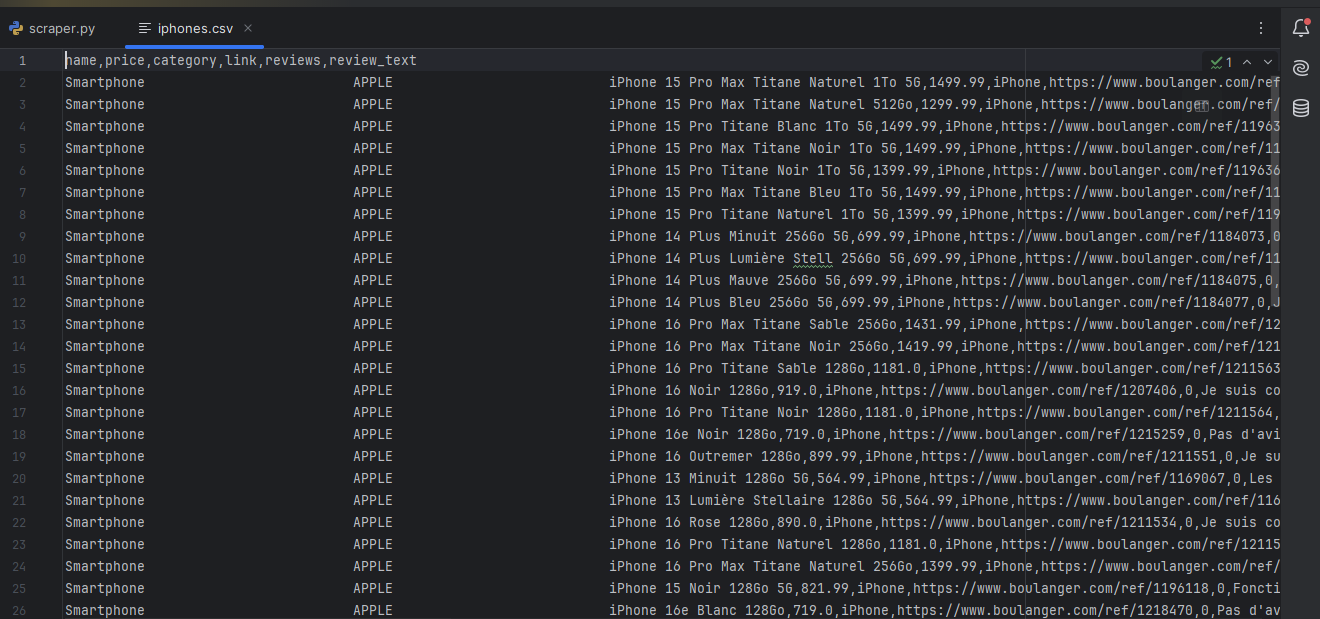








**Iphone.csv**

****

## ****2. Nettoyage des données →**** clean\_iphones.py(Premier nettoyage simple (suppression d’erreurs basiques)

**Objectif :**  
Ce script nettoie et prépare les données brutes issues du scraping pour une utilisation plus efficace.

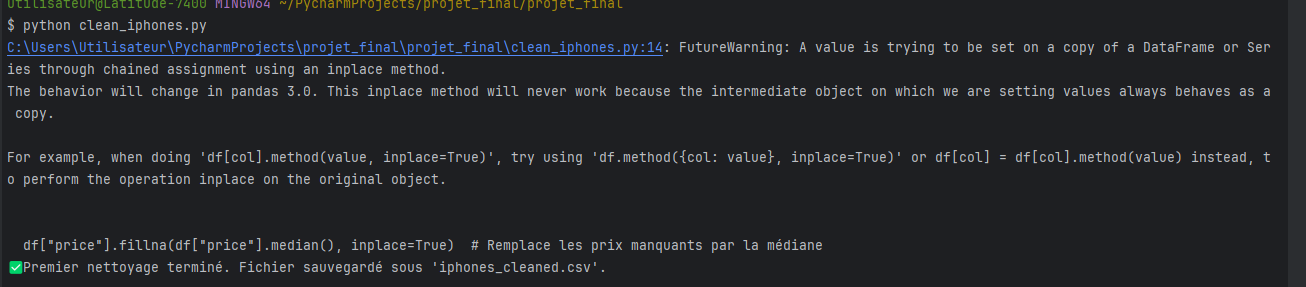
**Fonctionnalités :**

* Suppression des espaces et caractères inutiles (str.strip()).
* Conversion des **prix** en format numérique (float).
* Vérification des valeurs manquantes et gestion des erreurs.
* Ajout de la colonne **"price\_category"** pour segmenter les iPhones :
  + **Entrée de gamme** : Prix < 700€
  + **Moyenne gamme** : 700€ ≤ Prix < 1000€
  + **Haut de gamme** : Prix ≥ 1000€
* Sauvegarde des données propres dans iphones\_cleaned.csv.

**Explication du code :**

1. Chargement des données brutes (pd.read\_csv("iphones.csv")).
2. Nettoyage et transformation des colonnes (df["price"] = pd.to\_numeric(df["price"], errors="coerce")).
3. Ajout de la segmentation des prix (df["price\_category"] = ...).
4. Enregistrement dans iphones\_cleaned.csv.





**Iphone-cleaned.csv**



#### **Nettoyage des données → cleaning.py (**Amélioration avancée, transformation des données**)**

### **Objectif :**

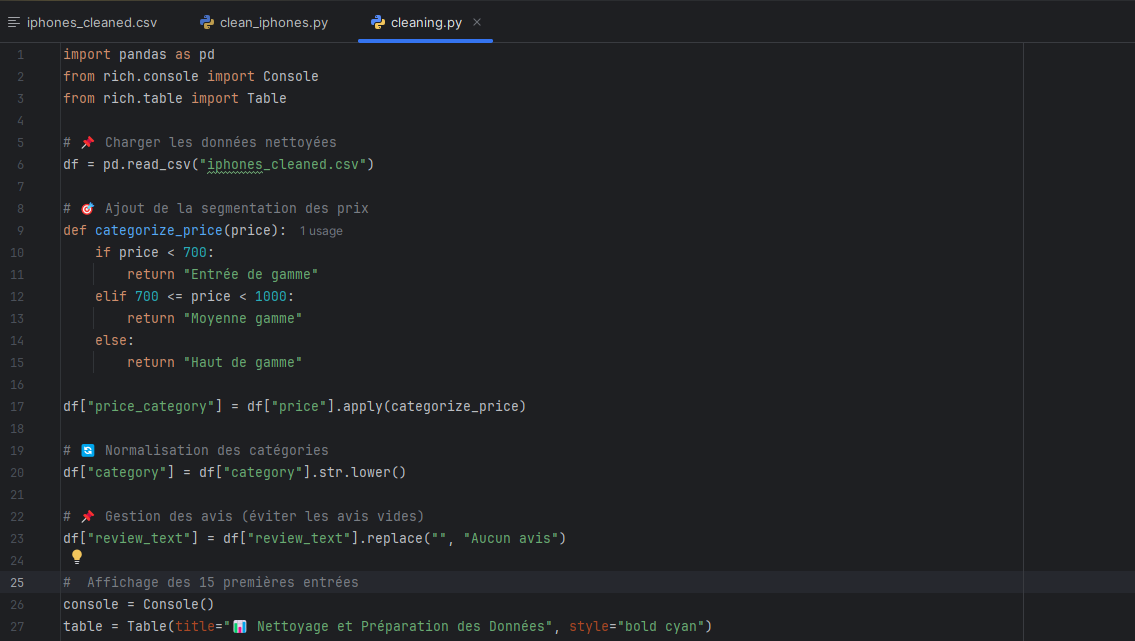
Ce script assure le **nettoyage et la préparation** des données scrappées afin de les rendre exploitables pour l'analyse et le stockage.

#### **Fonctionnalités :**

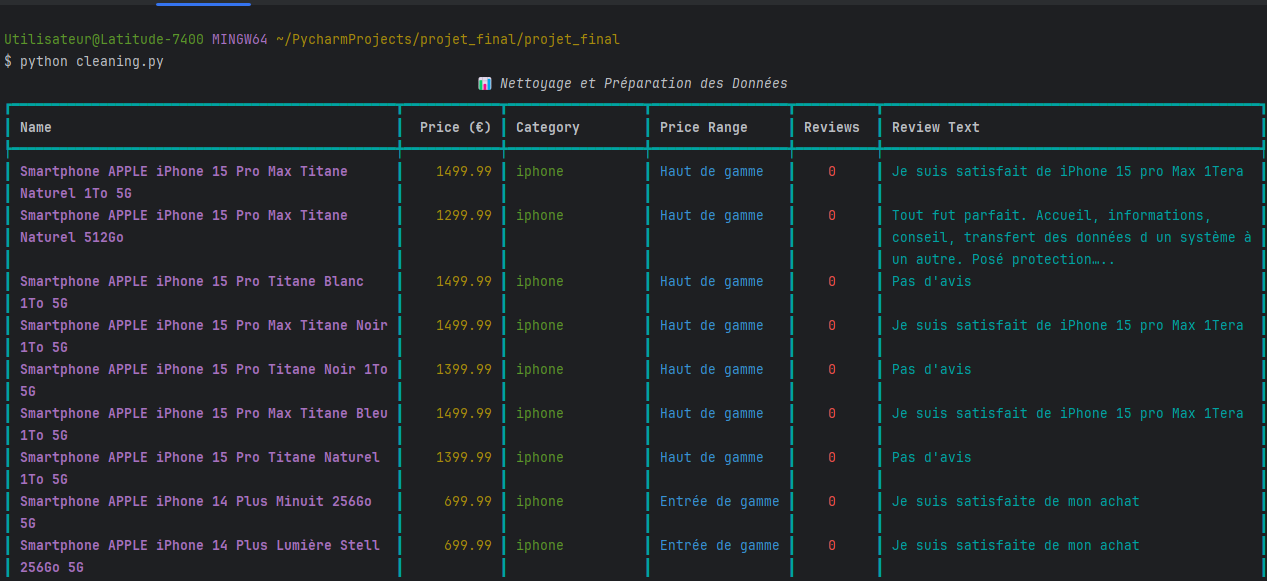
* Chargement des données brutes depuis iphones\_cleaned.csv.
* Suppression des espaces inutiles dans les noms.
* Conversion des prix et des avis en format numérique.
* Gestion des valeurs manquantes (remplacement par la médiane).
* Ajout d'une **catégorie de prix** (entrée de gamme, moyenne gamme, haut de gamme).
* Normalisation des catégories pour uniformiser les données.

#### **Explication du code :**

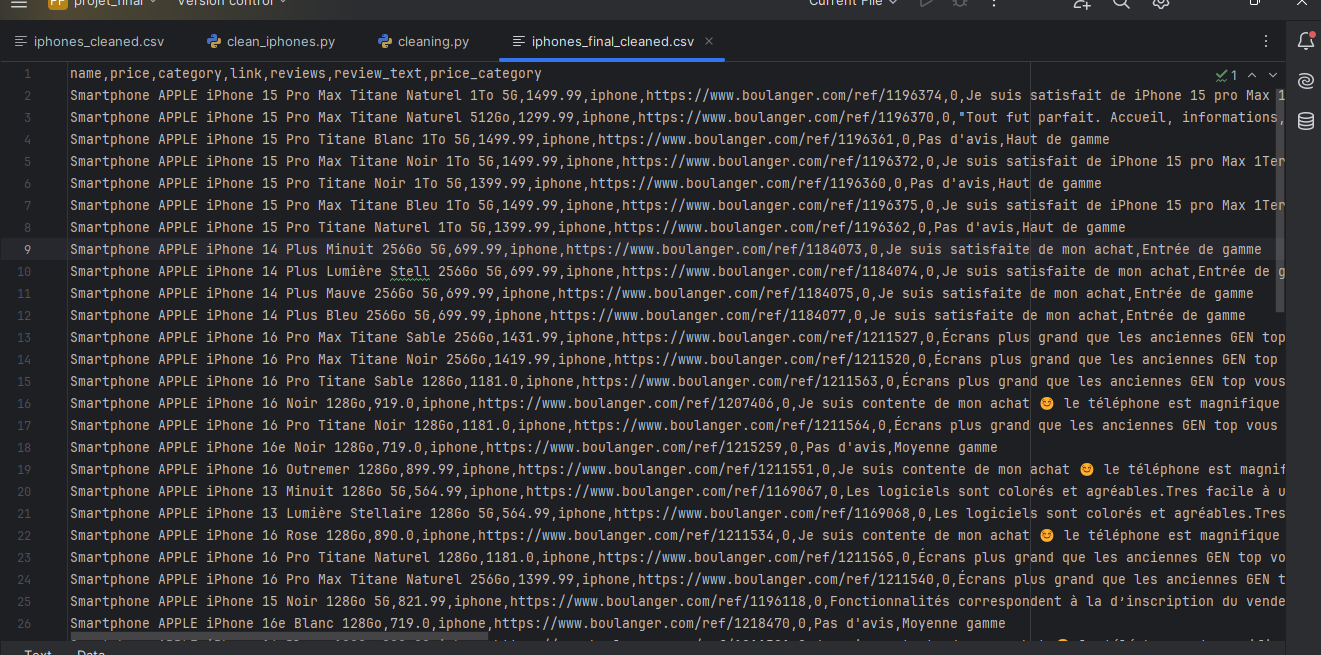
1. **Chargement des données brutes** (pd.read\_csv("iphones\_cleaned.csv")).
2. **Nettoyage des noms** (suppression des espaces inutiles).
3. **Conversion des colonnes numériques** (price et reviews).
4. **Traitement des valeurs manquantes** (remplacement des prix manquants par la médiane).
5. **Ajout d'une segmentation des prix** (price\_category).
6. **Enregistrement des données propres** dans iphones\_final\_cleaned.csv.







iphones\_final\_cleaned.csv



Haut du formulaire

Bas du formulaire

## ****3. Stockage dans MySQL →**** store\_mysql.py

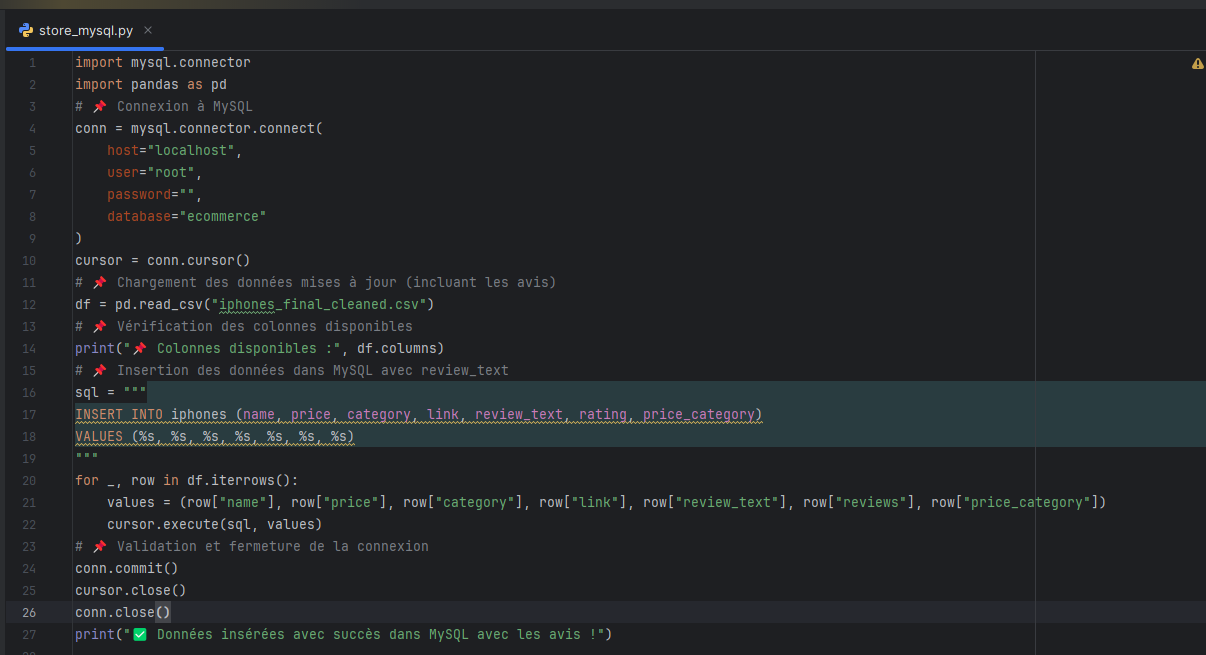
**Objectif :**  
Importer les données nettoyées dans une **base de données MySQL** pour une exploitation plus avancée.

**Fonctionnalités :**

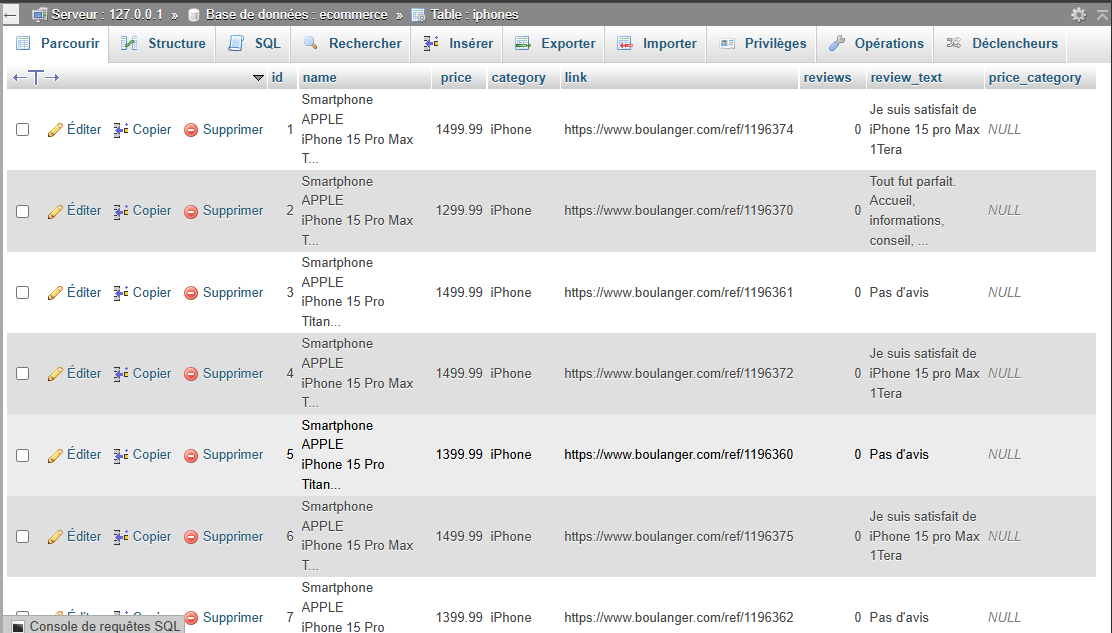
* Connexion à MySQL (mysql.connector.connect(...)).
* Création de la table iphones.
* Insertion des données de iphones\_cleaned.csv dans la base.
* Vérification et gestion des erreurs SQL.

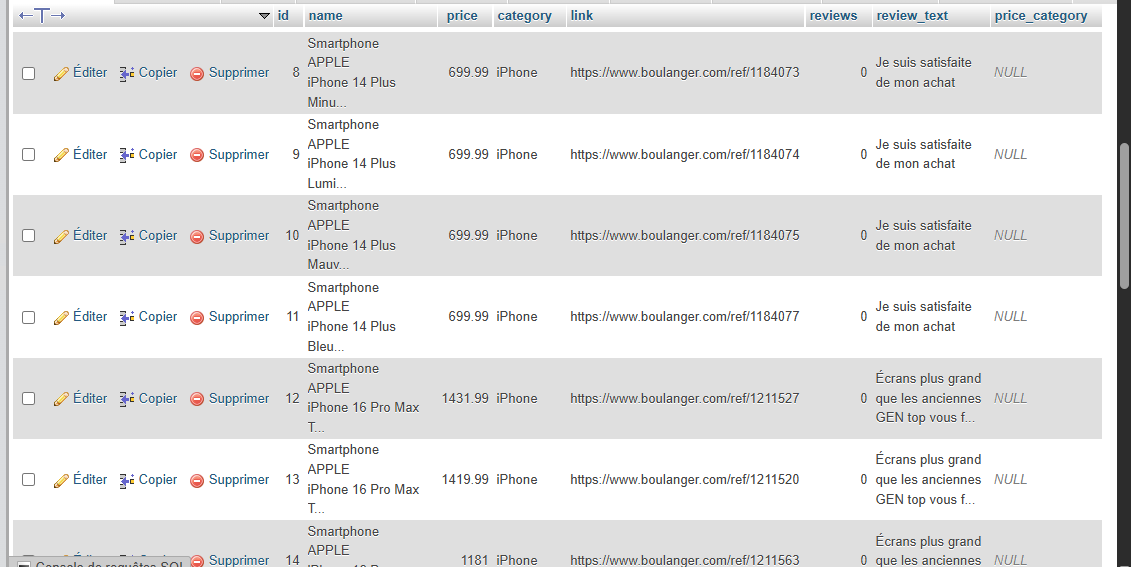
**Explication du code :**

1. Connexion à la base de données MySQL.
2. Lecture des données nettoyées (pd.read\_csv("iphones\_cleaned.csv")).
3. Boucle sur chaque ligne pour insérer dans MySQL (cursor.execute(...)).
4. Validation avec conn.commit() et fermeture de la connexion.









## ****4. Visualisation des données →**** visualization.py

**Objectif :**  
Créer des graphiques pour analyser les tendances des prix et des avis des iPhones.

**Fonctionnalités :**

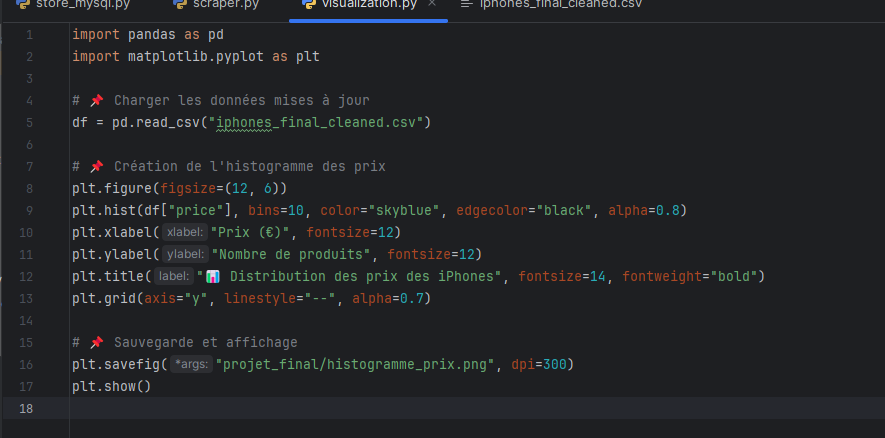
* **Histogramme** : Distribution des prix des iPhones.
* **Barplot** : Moyenne des prix par modèle.
* **Scatter Plot** : Corrélation entre le prix et le nombre d’avis.
* **Pie Chart** : Répartition des iPhones selon leur gamme de prix.
* Enregistrement des graphiques en PNG (plt.savefig(...)).

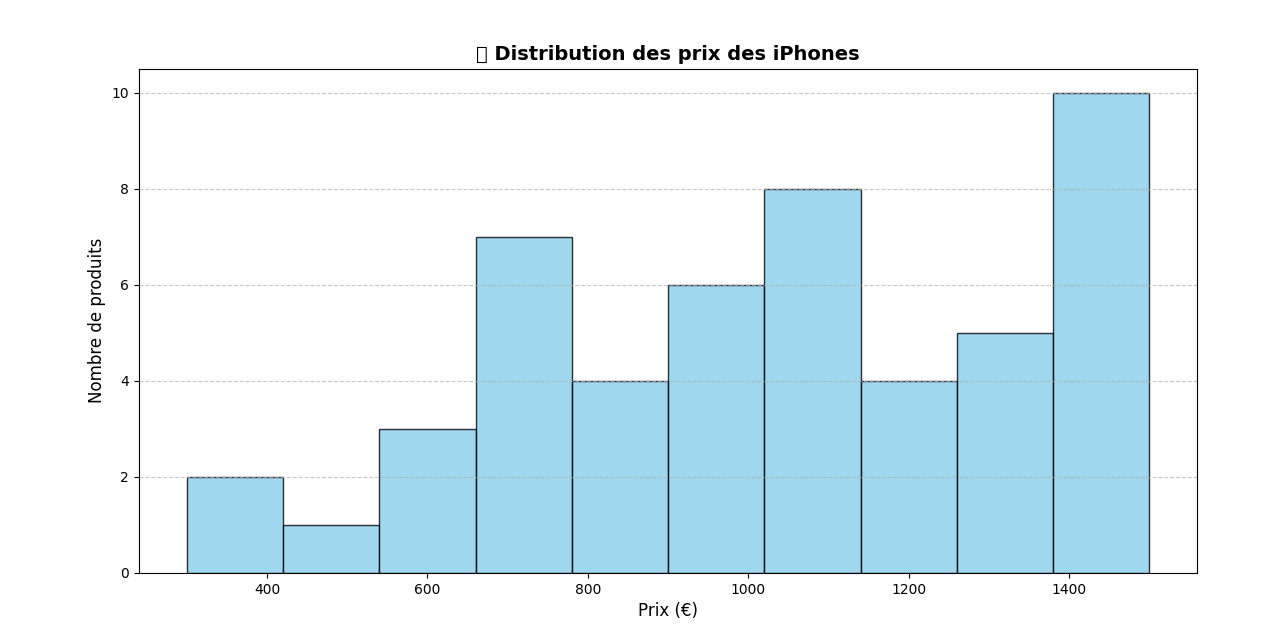
**Explication du code :**

1. Chargement des données nettoyées (pd.read\_csv("iphones\_cleaned.csv")).
2. Création de chaque type de graphique avec matplotlib.pyplot.
3. Ajout des légendes et styles pour une meilleure lisibilité.
4. Sauvegarde dans projet\_final/histogramme\_prix.png, etc.

* **Histogramme des prix des iPhones**

Permet de voir la distribution des prix des iPhones disponibles.





Interpretation

 Concentration des prix sur le haut de gamme :

* On observe une **forte concentration des iPhones dans la tranche de prix supérieure à 1000€**.
* Cela confirme que **les modèles premium dominent le marché**.

 Présence notable des modèles autour de 800-1000€ :

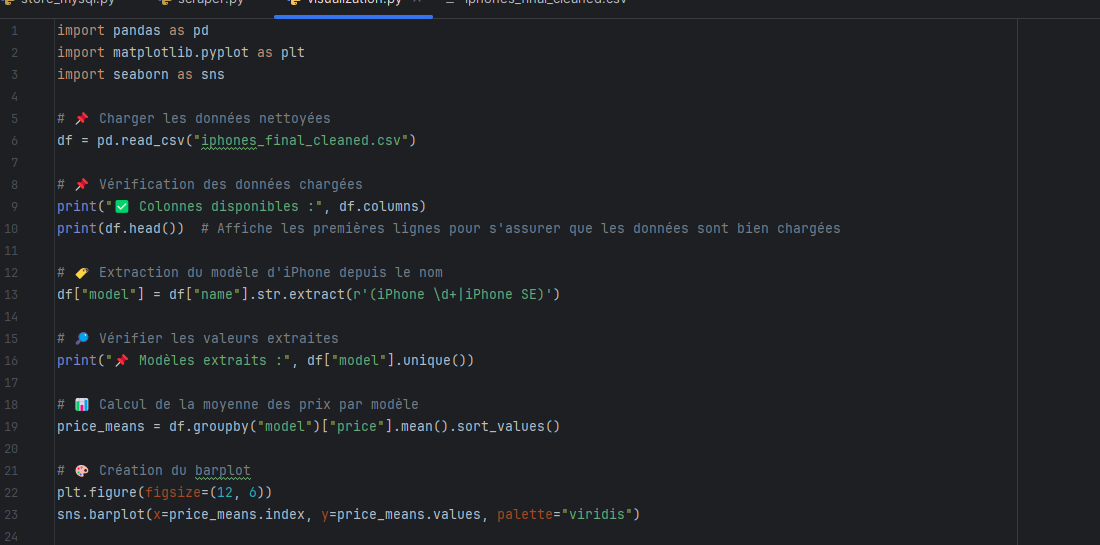
* Un pic significatif est visible **entre 800€ et 1000€**, montrant que plusieurs modèles sont disponibles dans cette gamme de prix.
* Cette tranche correspond aux versions **"standard" des iPhones récents**.

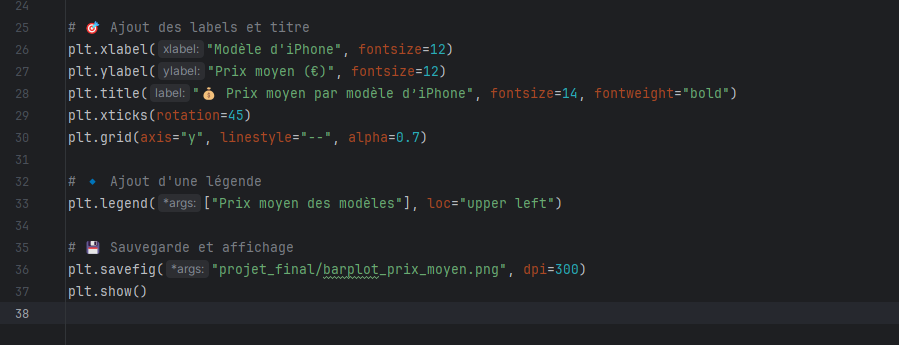
 Faible nombre de modèles abordables (<600€) :

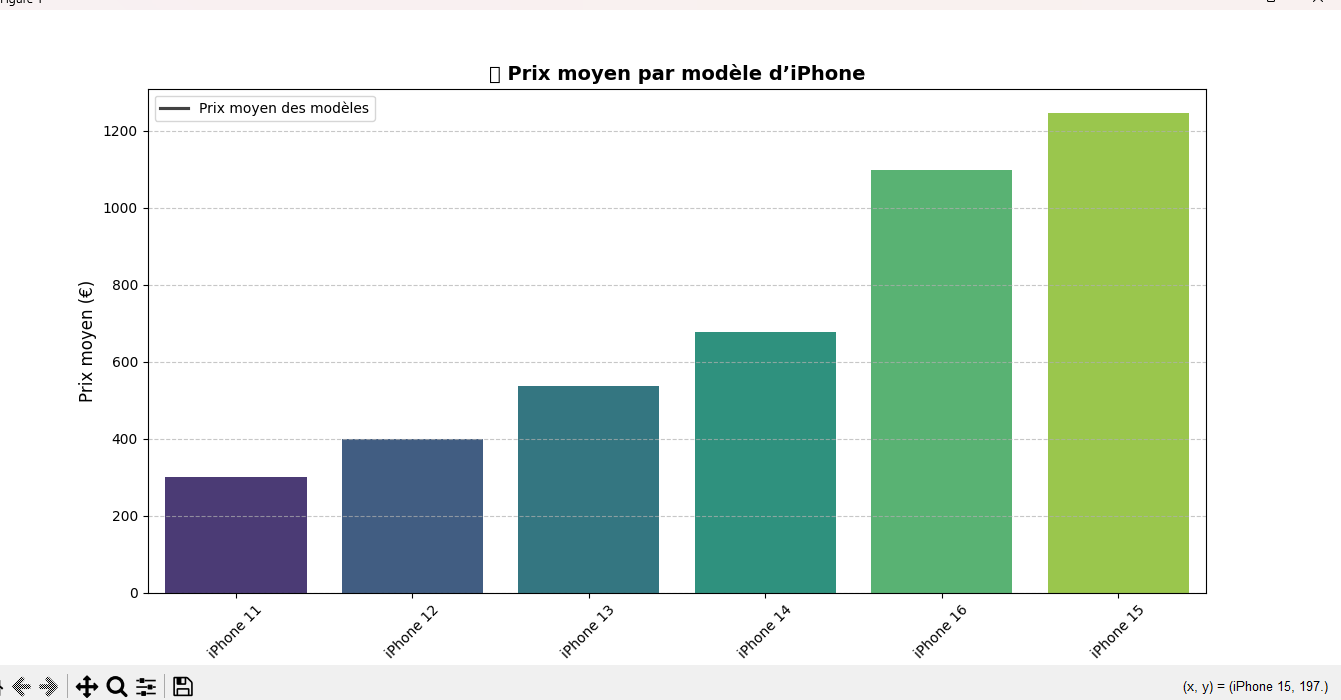
* Les modèles les moins chers sont peu nombreux.
* Il s’agit probablement d’**iPhones reconditionnés** ou d’anciennes générations.

## ****Moyenne des prix par modèle d’iPhone (Barplot)****

Affiche la moyenne des prix des différents modèles pour voir quels sont les plus chers ou les plus abordables.







### **Interprétation du Barplot : Prix moyen par modèle d’iPhone**

#### **1️ - Analyse globale**

* Le graphique montre **la variation du prix moyen** en fonction des modèles d’iPhone disponibles dans notre dataset.
* Les prix augmentent progressivement **d’un modèle plus ancien vers un modèle plus récent**.

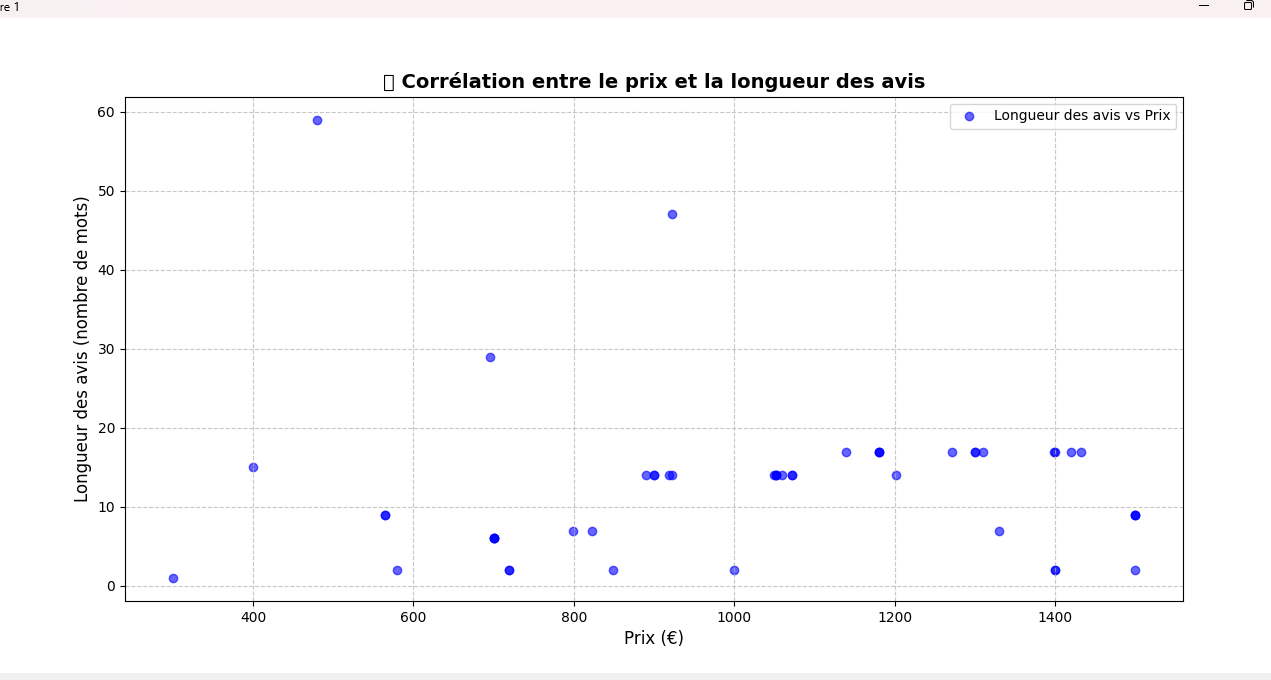
#### **2️- Observations spécifiques**

* **iPhone 11 et iPhone 12** : Ce sont les modèles les moins chers, avec un prix moyen inférieur à **500€**, ce qui est cohérent avec leur ancienneté et leur positionnement sur le marché du reconditionné.
* **iPhone 13 et iPhone 14** : Le prix moyen augmente progressivement, **se situant entre 500€ et 700€**. Ces modèles sont encore populaires mais leur prix est en baisse par rapport aux modèles les plus récents.
* **iPhone 16 et iPhone 15** : Ce sont les modèles **les plus récents et les plus chers**, avec un prix moyen dépassant **1000€**. Cela reflète leur positionnement haut de gamme et leurs nouvelles fonctionnalités.

## ****Corrélation entre prix et nombre d’avis (Scatter plot)****

Permet de voir si un iPhone avec un prix plus élevé reçoit plus d’avis ou non.





**Interprétation**

* Répartition des avis :
* La majorité des avis ont **moins de 20 mots**, quelle que soit la gamme de prix.
* Quelques avis exceptionnels dépassent **40-60 mots**, ce qui montre qu'un faible nombre de clients laissent des avis détaillés.

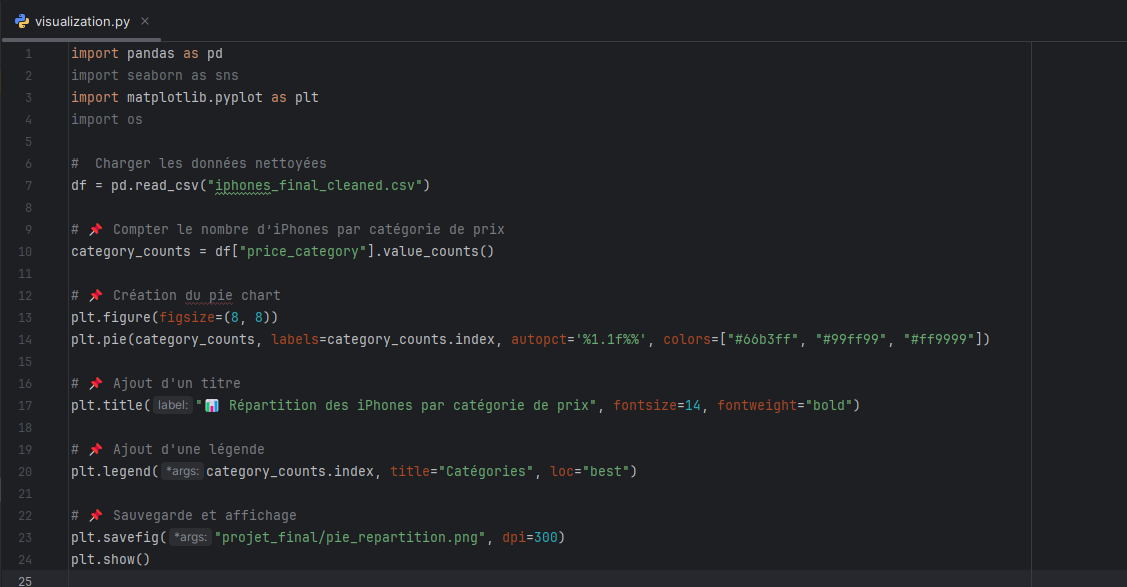
 Influence du prix sur la longueur des avis **:**

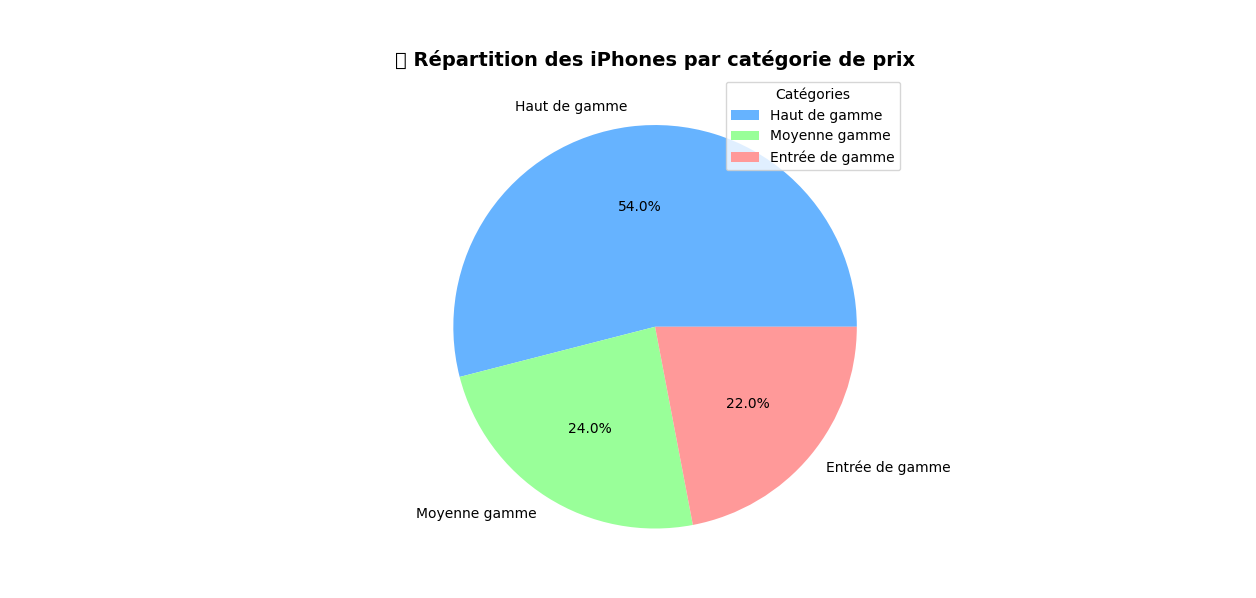
* On observe une certaine **dispersion** des points, ce qui signifie qu'il n'y a **pas de tendance claire** entre le prix et la longueur des avis.
* Les modèles **moins chers (~300-600€)** ont des avis **courts et rares**, ce qui peut indiquer un **engagement client plus faible**.
* Les modèles **plus chers (>1000€)** ont un éventail plus large d'avis, mais ils ne sont pas forcément plus longs.

 Cas particuliers :

* Quelques modèles à **prix moyen (~700-800€)** possèdent **des avis bien plus longs**, ce qui peut indiquer une satisfaction ou une insatisfaction plus détaillée des utilisateurs.
* **Répartition des iPhones selon les catégories (Pie chart)**

Visualisation de la proportion des modèles d’iPhones en fonction de leur catégorie (Entrée de gamme, Moyenne gamme, Haut de gamme).





**Interpretation**

 Dominance des modèles haut de gamme **:**

* **54%** des iPhones répertoriés appartiennent à la **catégorie haut de gamme** (>1000€).
* Cela montre une forte présence des modèles récents et premium dans le marché analysé.

 Part modérée des modèles moyenne gamme :

* **24%** des iPhones sont classés en **moyenne gamme** (entre 700€ et 1000€).
* Cette catégorie comprend probablement des iPhones récents avec moins de stockage ou des versions légèrement moins performantes.

 Faible proportion d’iPhones d’entrée de gamme :

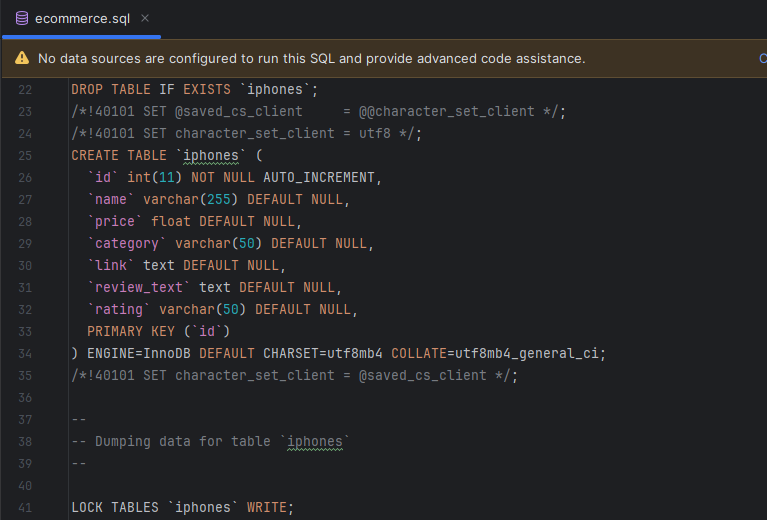
* **22%** des modèles appartiennent à l’**entrée de gamme** (<700€).
* Cela peut s’expliquer par la rareté des iPhones bon marché, souvent limités aux modèles reconditionnés ou plus anciens.

## ****5. Exportation de la base →**** ecommerce.sql

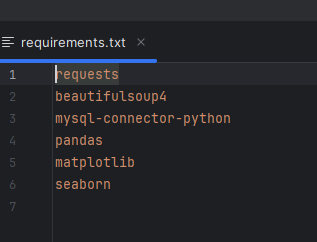
**Objectif :**  
Permet de sauvegarder la base de données sous format SQL pour **importation sur un autre système**.

**Fonctionnalités :**

* Contient **la structure de la base** (CREATE TABLE iphones (...)).
* Contient **les données des iPhones** (INSERT INTO iphones VALUES (...)).
* Permet de restaurer la base avec mysql -u root -p ecommerce < ecommerce.sql.



**6- Fichier Requirements**



Le fichier **requirements.txt** est un fichier texte qui liste toutes les **dépendances** (bibliothèques Python) nécessaires pour exécuter le projet.

Chaque ligne correspond à une bibliothèque spécifique et, éventuellement, à une version précise.