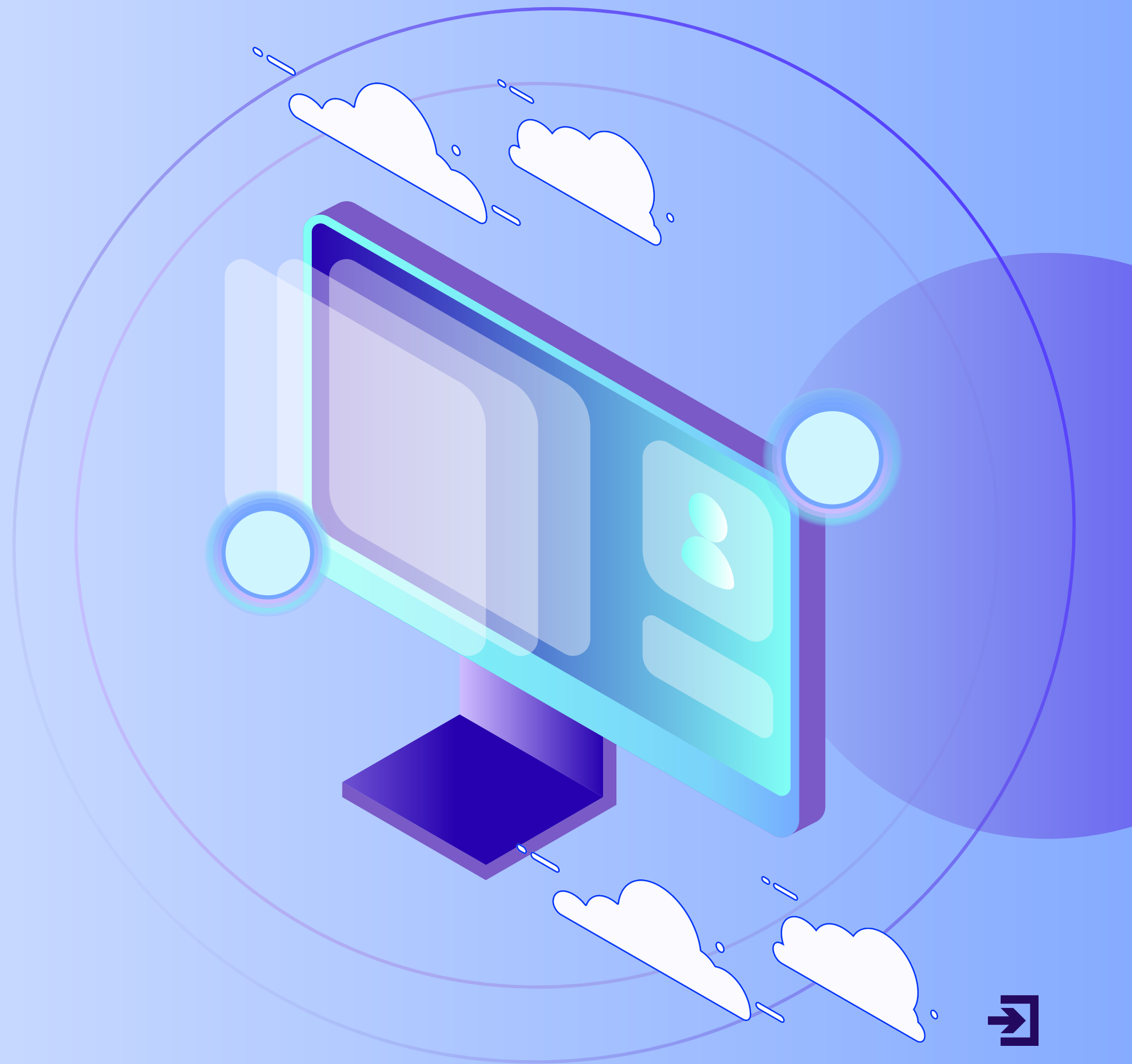
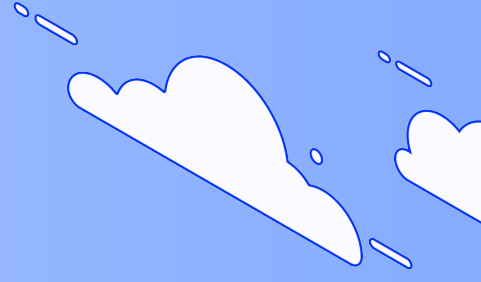

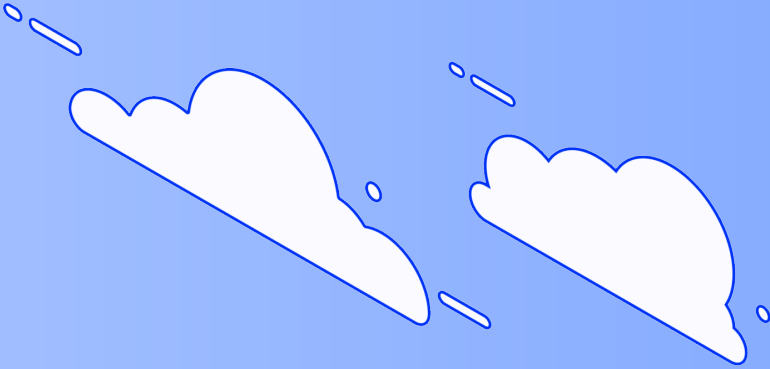
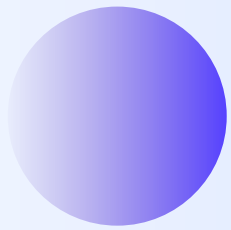





WEB SCRAPPING





ETAPE PROJET



- Étape 1 : Scraper les prix sur le site pap
- Étape 2 : Injecter ses données sur une bdd
- Étape 3 : Entraîner des modèles de prédictions afin de prédire les prix de vente



```
# Extraction en parallèle avec ThreadPoolExecutor pour le CSV
logging.info("Début de l'extraction des annonces.")
print("Début de l'extraction des annonces.")
with ThreadPoolExecutor(max_workers=10) as executor:
    futures = [executor.submit(extract_annonce, annonce, existing_annonces) for annonce in annonces]
    for future in as_completed(futures):
        result, reason = future.result()
        if result:
            data_to_save.append(result)
            nombre_enregistrees += 1
        else:
            nombre_ignored += 1
            ignored_details.append(reason)
```

ETAPE PROJET



```
import undetected_chromedriver as uc
from selenium.webdriver.common.by import By
from fake_useragent import UserAgent
import time
import random
import csv
import os
import sqlite3
import logging
from concurrent.futures import ThreadPoolExecutor, as_completed
```

- Analyse du site
- Contourner les sécurités
- Conversion des données en csv
- Multi-Threading

```
# Accepter les cookies si une bannière de consentement est présente
logging.info("Vérification de la bannière de cookies.")
print("Vérification de la bannière de cookies.")
try:
    cookie_button = driver.find_element(By.XPATH, "//span[contains(@class, 'sd-cmp-fuQAp') and contains(@class, 'sd-cmp-3_LLS')]")
    cookie_button.click()
    logging.info("Bannière de cookies acceptée.")
    print("Bannière de cookies acceptée.")
except Exception:
    logging.info("Pas de bannière de cookies détectée.")
    print("Pas de bannière de cookies détectée.")
```

```
# Fonction pour lire les annonces existantes à partir du fichier CSV
def load_existing_annonces(filename):
    if not os.path.exists(filename):
        return set() # Retourner un ensemble vide si le fichier n'existe pas
    with open(filename, mode="r", newline="", encoding="utf-8") as file:
        reader = csv.reader(file)
        next(reader) # Ignorer l'en-tête
        return {row[0] for row in reader} # Utiliser la location comme clé
```

```
# Fonction pour ajouter une annonce à la base de données SQLite
def add_annonce_to_db(db_file, annonce):
    try:
        connection = sqlite3.connect(db_file)
        cursor = connection.cursor()
        cursor.execute("INSERT INTO annonces (location, pieces, surfa
        ?, ?, ?, ?)", annonce)
        connection.commit()
        cursor.close()
        connection.close()
    return True
```

ETAPE PROJET

- Création de la bdd SQLite
- Vérification du csv
- Insertion des données en base
- Le fichier output ne doit pas être créé

```
# Fonction pour créer la table dans la base de données SQLite
def create_annonces_table(db_file):
    try:
        connection = sqlite3.connect(db_file)
        cursor = connection.cursor()
        cursor.execute('
        CREATE TABLE IF NOT EXISTS annonces (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        location INTEGER NOT NULL,
        pieces INTEGER NOT NULL,
```



```
Coefficient de détermination ( $R^2$ ): 0.9000840964273804
Erreur quadratique moyenne (MSE): 44649.215011910244
Racine carrée de l'erreur quadratique moyenne (RMSE): 211.30360861071503
Erreur absolue moyenne (MAE): 140.1500592276712
Erreur moyenne absolue en pourcentage (MAPE): 10.71%
```

ETAPE PROJET

```
import matplotlib.pyplot as plt
#je crée un boxplot pour voir si il y a des valeurs aberrantes
data.boxplot(column="price")
[4]
```

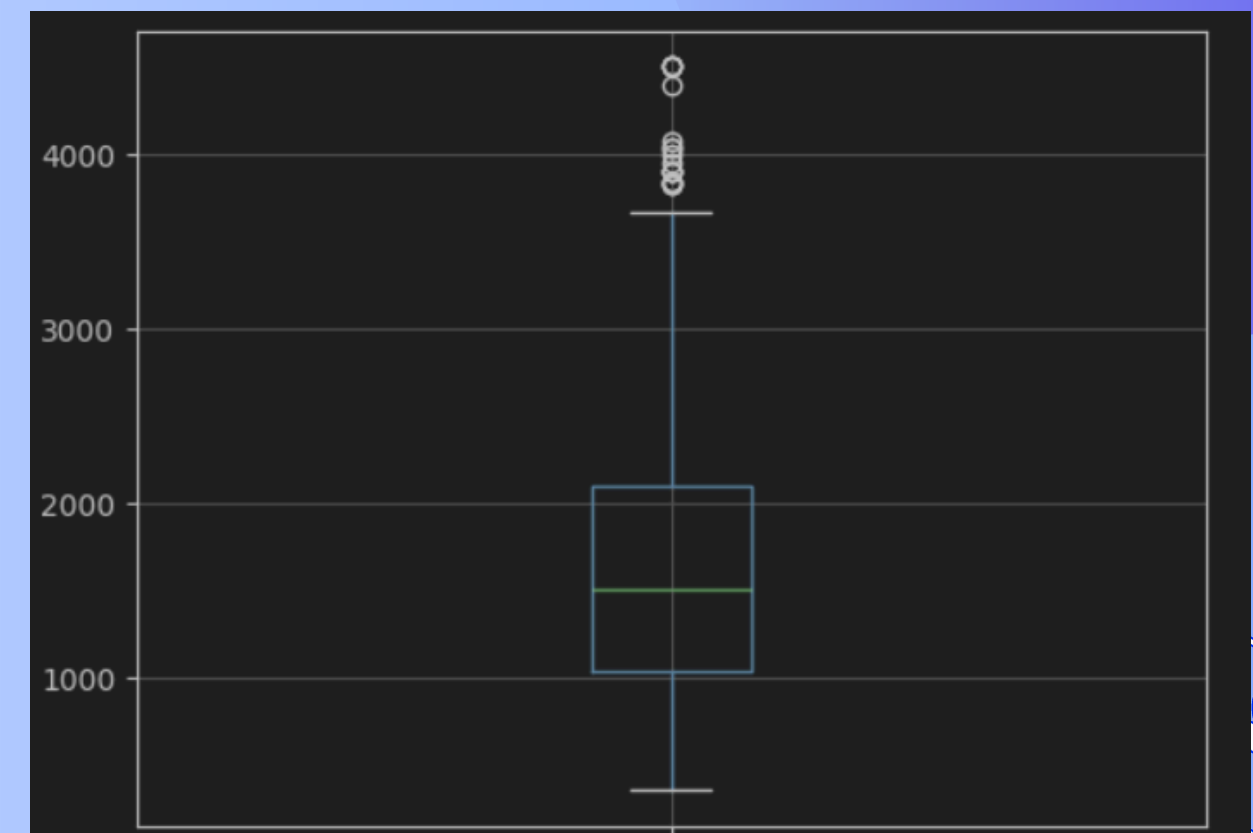
```
# Préddictions
y_pred = model.predict(X_test_scaled)

# Affichage des résultats
results = pd.DataFrame({'Location': X_test['location'],
                        'Prix Réel': y_test, 'Prix Prédit': y_pred})

print(results)

# Calcul des erreurs
mse = mean_squared_error(y_test, y_pred)
rmse = mse ** 0.5
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

- Vérification des données aberrantes
- Choix du modèle
- Test





THANK YOU!