Pour mettre en place une plateforme capable d'extraire des données de plusieurs documents Excel, de les analyser et d'afficher des diagrammes et histogrammes, vous pouvez suivre une approche structurée. Voici un guide détaillé pour y parvenir :

---

### \*\*Étapes pour créer la plateforme\*\*

#### \*\*1. Définir les besoins et les outils\*\*

Avant de commencer, clarifiez vos besoins :

- Quels types de données souhaitez-vous extraire (nombres, textes, dates, etc.) ?

- Où sont stockés les fichiers Excel (local, cloud, base de données) ?

- Quels types de graphiques souhaitez-vous générer (histogrammes, diagrammes circulaires, courbes, etc.) ?

\*\*Outils recommandés :\*\*

- \*\*Langages de programmation\*\* : Python est idéal pour ce type de projet grâce à ses bibliothèques puissantes.

- \*\*Bibliothèques Python\*\* :

- `pandas` : Pour manipuler et analyser les données.

- `openpyxl` ou `xlrd` : Pour lire les fichiers Excel.

- `matplotlib` ou `seaborn` : Pour créer des graphiques.

- `plotly` : Pour des visualisations interactives.

- `Flask` ou `Django` : Pour créer une interface web si nécessaire.

- \*\*Interface utilisateur\*\* : Vous pouvez utiliser une application web ou une interface graphique (GUI) avec des bibliothèques comme `Dash`, `Streamlit` ou `Tkinter`.

---

#### \*\*2. Extraire les données des fichiers Excel\*\*

Utilisez Python pour lire les données de plusieurs fichiers Excel. Voici un exemple de code :

```python

import pandas as pd

import os

# Chemin vers le dossier contenant les fichiers Excel

dossier\_excel = "chemin/vers/dossier"

# Liste pour stocker les données de tous les fichiers

donnees\_combinees = []

# Parcourir tous les fichiers Excel dans le dossier

for fichier in os.listdir(dossier\_excel):

if fichier.endswith(".xlsx"): # Vérifier que le fichier est un Excel

chemin\_fichier = os.path.join(dossier\_excel, fichier)

# Lire chaque fichier Excel

df = pd.read\_excel(chemin\_fichier)

donnees\_combinees.append(df)

# Combiner toutes les données en un seul DataFrame

donnees\_totales = pd.concat(donnees\_combinees, ignore\_index=True)

# Afficher un aperçu des données

print(donnees\_totales.head())

```

---

#### \*\*3. Nettoyer et préparer les données\*\*

Les données brutes peuvent nécessiter un nettoyage avant d'être utilisées pour les graphiques :

- Supprimer les doublons.

- Gérer les valeurs manquantes.

- Convertir les types de données.

Exemple de nettoyage :

```python

# Supprimer les lignes avec des valeurs manquantes

donnees\_totales.dropna(inplace=True)

# Convertir une colonne en type numérique

donnees\_totales['Colonne\_Numerique'] = pd.to\_numeric(donnees\_totales['Colonne\_Numerique'], errors='coerce')

```

---

#### \*\*4. Créer des diagrammes et histogrammes\*\*

Utilisez `matplotlib` ou `seaborn` pour générer des graphiques. Voici quelques exemples :

\*\*Histogramme :\*\*

```python

import matplotlib.pyplot as plt

# Créer un histogramme pour une colonne spécifique

plt.figure(figsize=(10, 6))

plt.hist(donnees\_totales['Colonne\_Numerique'], bins=20, color='blue', edgecolor='black')

plt.title('Histogramme des Données')

plt.xlabel('Valeurs')

plt.ylabel('Fréquence')

plt.show()

```

\*\*Diagramme en barres :\*\*

```python

# Calculer les moyennes par catégorie

moyennes\_par\_categorie = donnees\_totales.groupby('Categorie')['Valeur'].mean()

# Créer un diagramme en barres

plt.figure(figsize=(10, 6))

moyennes\_par\_categorie.plot(kind='bar', color='green')

plt.title('Moyennes par Catégorie')

plt.xlabel('Catégories')

plt.ylabel('Moyenne')

plt.show()

```

---

#### \*\*5. Rendre la plateforme interactive (optionnel)\*\*

Si vous souhaitez rendre la plateforme plus conviviale, vous pouvez créer une interface web ou une application graphique.

\*\*Option 1 : Utiliser Streamlit\*\*

Streamlit est un framework Python simple et rapide pour créer des applications web interactives.

```python

import streamlit as st

import pandas as pd

import matplotlib.pyplot as plt

# Charger les données

donnees = pd.read\_excel("chemin/vers/fichier.xlsx")

# Interface utilisateur

st.title("Analyse de Données Excel")

st.write("Voici un aperçu des données :")

st.dataframe(donnees.head())

# Sélectionner une colonne pour l'histogramme

colonne\_selectionnee = st.selectbox("Choisissez une colonne pour l'histogramme", donnees.columns)

# Afficher l'histogramme

plt.figure(figsize=(10, 6))

plt.hist(donnees[colonne\_selectionnee], bins=20, color='blue', edgecolor='black')

plt.title(f'Histogramme de {colonne\_selectionnee}')

plt.xlabel('Valeurs')

plt.ylabel('Fréquence')

st.pyplot(plt)

```

Lancez l'application avec la commande :

```bash

streamlit run app.py

```

\*\*Option 2 : Utiliser Dash\*\*

Dash est un framework Python pour créer des tableaux de bord interactifs.

```python

from dash import Dash, dcc, html

import plotly.express as px

import pandas as pd

# Charger les données

donnees = pd.read\_excel("chemin/vers/fichier.xlsx")

# Initialiser l'application Dash

app = Dash(\_\_name\_\_)

# Créer un graphique Plotly

fig = px.histogram(donnees, x="Colonne\_Numerique", title="Histogramme des Données")

# Layout de l'application

app.layout = html.Div([

html.H1("Analyse de Données Excel"),

dcc.Graph(figure=fig)

])

# Lancer l'application

if \_\_name\_\_ == '\_\_main\_\_':

app.run\_server(debug=True)

```

---

#### \*\*6. Automatiser et déployer la plateforme\*\*

- \*\*Automatisation\*\* : Si vous avez besoin d'extraire régulièrement des données, utilisez des tâches planifiées (par exemple, avec `cron` sur Linux ou `Task Scheduler` sur Windows).

- \*\*Déploiement\*\* :

- Pour une application web, hébergez-la sur des services comme \*\*Heroku\*\*, \*\*AWS\*\*, ou \*\*Azure\*\*.

- Pour une application locale, créez un exécutable avec `PyInstaller`.

---

### \*\*Conclusion\*\*

En suivant ces étapes, vous pouvez créer une plateforme complète pour extraire, analyser et visualiser des données à partir de plusieurs fichiers Excel. Voici un résumé des outils clés :

- \*\*Extraction\*\* : `pandas`, `openpyxl`.

- \*\*Visualisation\*\* : `matplotlib`, `seaborn`, `plotly`.

- \*\*Interface utilisateur\*\* : `Streamlit`, `Dash`, `Flask`.

Si vous avez besoin d'aide pour implémenter une partie spécifique ou adapter le projet à vos besoins, n'hésitez pas à demander ! 😊