**Guide d'utilisation : DataVisualizer**

**1. Introduction**

Ce programme permet de visualiser des données environnementales stockées dans un fichier **Parquet** (fichier .csv compressé). Il génère des graphiques représentant l'évolution de plusieurs paramètres (niveau sonore, luminosité, température, humidité, vent, pression, précipitations) en fonction du temps.

L'utilisateur peut tracer des graphiques :

* Pour un nombre de jours voulu
* Par mois
* Pour des plages horaires spécifiques

**2. Prérequis**

Avant d'exécuter ce script, assurez-vous d'avoir installé les bibliothèques nécessaires en exécutant la commande suivante :

*pip install pandas matplotlib pyarrow*

**3. Comment utiliser le programme ?**

**3.1 Chargement des données**

Tout d'abord, il faut créer un objet DataVisualizer en fournissant le chemin du fichier contenant les données :

*visualizer = DataVisualizer("data.parquet")*

**3.2 Tracer des graphiques**

**a) Afficher les données sur plusieurs jours**

Si vous souhaitez afficher les données sur plusieurs jours à partir d'une date donnée :

*visualizer.plot\_data\_by\_days(start\_date=pd.Timestamp("2024-01-02"), days=4, elements=["Leq", "Lux"])*

* **start\_date** : Date de départ (format pd.Timestamp("YYYY-MM-DD"))
* **days** : Nombre de jours à afficher
* **elements** : Liste des éléments à afficher parmi Leq, Lux, Temp, Humidity, Wind, Pressure, Precipitations

**b) Afficher les données sur plusieurs mois**

Pour afficher les données sur une période mensuelle :

*visualizer.plot\_data\_by\_months(start\_date=pd.Timestamp("2023-12-01"), months=1, elements=["Leq", "Lux"])*

* **start\_date** : Date de départ
* **months** : Nombre de mois à afficher

**c) Afficher les données pour certaines heures de la journée**

Pour analyser les données uniquement sur une plage horaire spécifique (ex : 23h à 3h du matin) :

*visualizer.plot\_data\_by\_hour\_range(*

*start\_date=pd.Timestamp("2024-01-01"),*

*end\_date=None,*

*hours=[23, 0, 1, 2, 3],*

*elements=["Leq", "Lux", "Temp", "Humidity", "Wind", "Pressure", "Precipitations"])*

* **start\_date** : Date de départ
* **end\_date** : (Optionnel) Date de fin. Si None, une seule journée est affichée.
* **hours** : Liste des heures à inclure dans l'analyse

Si vous souhaitez analyser plusieurs jours en gardant une plage horaire fixe (ex : 5h à 7h sur 2 jours) :

*visualizer.plot\_data\_by\_hour\_range(*

*start\_date=pd.Timestamp("2024-02-01"),*

*end\_date=pd.Timestamp("2024-02-02"),*

*hours=[5, 6, 7],*

*elements=["Leq", "Lux"])*

**4. Explication du Code**

**4.1 Chargement et préparation des données**

Le script lit le fichier **Parquet** contenant les données, puis effectue une agrégation des mesures :

* **Données horaires** (df\_hourly) : Moyenne, minimum et maximum par heure
* **Données par minute** (df\_minutes) : Moyenne, minimum et maximum par minute

**4.2 Fonction de tracé des données**

La fonction \_plot\_time\_series génère les graphiques et :

* Met en évidence les week-ends en grisé
* Ajoute une bande d'incertitude (min-max) pour les paramètres Leq et Lux
* Applique une échelle logarithmique pour Lux
* Permet de zoomer sur des plages horaires spécifiques

**5. Remarque**

* Si le fichier Parquet ne contient pas les colonnes attendues (Leq\_dBA, lux\_S1, etc.), des erreurs peuvent survenir.
* Vérifiez que start\_date et end\_date sont correctement définis pour éviter les erreurs d'affichage.