**Data Visualization Dashboard**

**1. Project Overview**

**Objective**

The purpose of this project is to create a dashboard based on key performance indicators (KPIs) calculated from a given dataset and visualize it. The data is organized as csv files in a zip file, in which we can find 4 columns :

* The **Artist** name
* The **Album**
* The **Title** of the song
* The **Date** of listening.

Each csv file is named according to the name of the **User**.

**2. Workflow**

Here are the 3 main steps the application follows.

1. **Data Retrieval**
   * The dataset is obtained by downloading the zip file on Teams. There are 63 csv files.
   * The user can add any csv if it respects the same structure than the others such as : “artist name”, “Album”, “Title”, “Date”. The name of the file must be the username of the listener.
2. **Data Processing**
   * Execute “app.py” to import new csv files in the dataset. Une image contenant texte, capture d’écran, Police, logiciel

     Description générée automatiquement
   * Verify the presence of a specific file by entering its nameUne image contenant texte, capture d’écran, Police, logiciel

     Description générée automatiquement
   * Clean the Data and add the name of the columns
   * Upload the dataset to the Relational Database MySQL
3. **Data Visualization (vérifier s’il manque pas un truc)**
   * Use **Power BI** to create interactive visualizations by importing the SQL Database

**3. Tools and Technologies (verifier s’il manque pas un truc)**

**Backend**

* **Python Flask**: For developing the ETL pipeline and web application.
* **MySQL**: For storing and querying the dataset.

**Visualization**

* **Power BI**: To create dynamic and interactive dashboards for data visualization.

**4. KPIs**

The following KPIs were calculated:

* **Most listened-to track of all time**.
* **Most listened-to track for each week**.
* **Most listened-to album of all time**.
* **Most listened-to album for each week**.
* **Cross-tabulation**: Number of listened tracks by listener and artist.
* **Top 10 biggest listeners**:
  + **All time**.
  + **Each week**.

**5. Implementation Details**

**1. Data Extraction and Loading**

* The dataset is fetched and uploaded using a Python Flask-based web interface.
* The data is cleaned and loaded into a **MySQL database**.

**2. KPI Calculations**

We calculated all KPIs with the Power Query Editor

* The most listened song of all time :

We copy the dataset from the original one so that we don’t modify it.  
We applicate an aggregation to get the count of all songs.  
We sort the resulting table by descending order.  
We only keep the first row which is the most listened song.

Ex: Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

* The most listened songs per week :

We create a new column “week” that indicates the week number of the year.  
We group by song title and by weeks to get the number of listening of each track per week.  
We sort the resulting table in ascending order according to “Week” column.

Ex: Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

* Most listened-to album of all time.

We applicate an aggregation to get the count of all albums.  
We sort the resulting table by descending order.  
We only keep the first row which is the most listened album.

Ex: Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

* Most listened-to album for each week.

We create a new column “week” that indicates the week number of the year.  
We group by song title and by weeks to get the number of listening of each track per week.  
We sort the resulting table in ascending order according to “Week” column.

Ex: Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

* Cross-tabulation: Number of listened tracks by listener and artist.

We calculate the total of listening per artists  
We sort the artist per Total\_Count descending order and we keep the N most listened artists  
Filter the initial data to keep only the selected artists  
Group the data per user and artist  
Group the total of listening per user  
We add the column “UserTotal” to “GroupedData” and column for normalization

Ex: Une image contenant texte, logiciel, ligne, nombre

Description générée automatiquement

* Top 10 biggest listeners All time :

We group by username  
We sort the resulting table by descending order and we keep the 10 first rows

Ex: Une image contenant texte, capture d’écran, Caractère coloré, diagramme

Description générée automatiquement

* Top 10 biggest Each Week :

We add the column “Week”  
We group by Week and we sort by week ascending order  
We group every listener per week and only keep the top 10 so that we get a new column “Top10” that contains the 10 biggest listeners per week  
We expand the table to print the name and the count of the biggest listeners  
Ex: Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

**3. Visualization with Power BI**

* Data is imported into Power BI from the MySQL database
* Dashboards include:
  + Bar charts for top listeners and albums.
  + Cross-tabulated tables for listeners vs. artists.
  + Pie charts for 10 biggest listeners of all time

**6. Folder Structure**

**Application\_of\_big\_data/**

├── **Vizualisations**/

│ ├── Viz1.pbix

│ ├── Viz2.pbix

├── **data**/

│ ├── dataset1.csv

│ ├── dataset2.csv

│ ├── ...

├── **template**/

│ ├── index.html

├── Add\_file\_to\_sql.py

├── add\_username.py

├── app.py

├── fonctions.py

├── requirements.txt

├── .gitignore

└── README.md

**7. How to Run (à completer)**

**Step 1: Run the Flask Application**

**Step 2: View Dashboards**