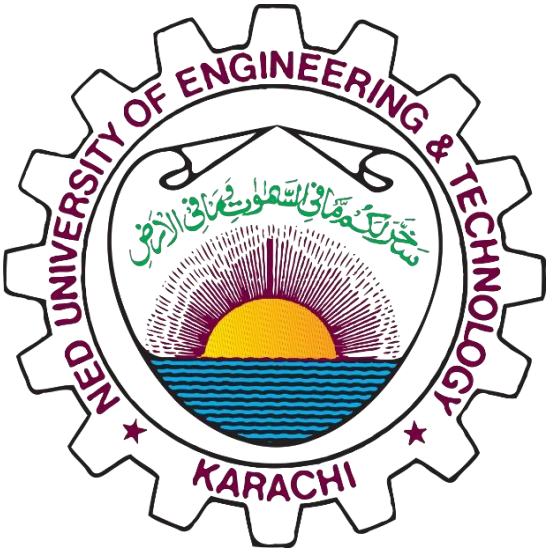

Name : Rabat Shahid

Roll Number : DS-W-14

Subject : BDA

Submitted to : Sir Umer

Mid Term Report:
ETL Pipeline for Weather Analytics

1. Introduction
This report describes the implementation of an **ETL (Extract, Transform, Load) pipeline** for weather analytics. The pipeline collects weather data from multiple sources, processes it for consistency and completeness, and stores the final dataset in **MongoDB** for further analysis.

2. Data Extraction (Sources of Data)
The pipeline gathers weather data from **five different sources**:

1. Real-time Weather Data (WeatherAPI)
- **Source:** WeatherAPI (https://www.weatherapi.com/)
- **Format:** JSON
- **Method:** API request using Python's requests library
- **Data Extracted:** Current temperature, weather condition, location details

2. Historical Weather Data (Open-Meteo API)
- **Source:** Open-Meteo API (https://archive-api.open-meteo.com/)
- **Format:** JSON
- **Method:** API request
- **Data Extracted:** Daily maximum temperature for January 2024

3. NOAA Climate Data
- **Source:** National Centers for Environmental Information (NOAA)
- **Format:** CSV
- **Method:** Direct CSV download from NOAA's website
- **Data Extracted:** Hourly dry bulb temperature

4. Google Drive CSV File
- **Source:** Google Drive (Simulated dataset)
- **Format:** CSV
- **Method:** File downloaded using gdown
- **Data Extracted:** Historical temperature records

5. NoSQL Database (MongoDB Atlas)
- **Source:** MongoDB Atlas (Cloud Database)
- **Format:** JSON (retrieved as a Pandas DataFrame)
- **Method:** Connection established using pymongo
- **Data Extracted:** Previously stored weather records

3. Data Transformation

Once data is extracted from multiple sources, it undergoes the following transformations:

1. Handling Missing Data

- Missing values are **forward-filled** (ffill) and **back-filled** (bfill) to ensure continuity.

2. Unit Conversion

- Temperatures recorded in **Celsius** are converted to **Fahrenheit** using the formula:

$$\textbf{Temperature (°F)=(Temperature (°C)×59)+32}$$

3. Standardizing Date Formats

- Dates from different sources are converted into a **common format** using pandas.to_datetime().

4. Removing Duplicates

- Duplicate entries are **identified and removed** from all datasets.

5. Aggregation for Consistency

- Data is grouped by **date**, ensuring only one record per day for analysis.

6. Weather Condition Mapping

- Weather descriptions are categorized into standard groups like **Clear, Cloudy, Precipitation, and Severe Weather** for uniformity.

4. Data Loading (Storage in MongoDB)

After transformation, the cleaned data is stored in **MongoDB** using the following process:

1. **Previous records** in MongoDB are deleted to avoid duplication.
2. **New transformed records** are inserted into the MongoDB collection.
3. A final **merged dataset** is printed for verification.

5. Conclusion

This ETL pipeline efficiently collects, processes, and stores weather data from multiple sources. The transformed dataset ensures **consistency, completeness, and accuracy**, making it suitable for further **trend analysis and predictions**.

Final dataset is successfully stored in MongoDB as shown in output image.