WEATHER DATA ETL PIPELINE (WITH DOCKER & MYSQL)

Building a Scalable ETL Pipeline to Load Weather Data into a Database

Aparna Surya

INTRODUCTION

- This project aims to automate the process of extracting, transforming, and loading weather data into a MySQL database.
- The solution uses Python for the ETL logic, Docker for containerization, and MySQL for storing the processed data.
- Key Goal: Build a scalable, automated system for weather data processing and storage.

DATA SOURCE

- The weather data is sourced from the OpenWeatherMap API in CSV format.
- It includes data such as city name, temperature (in Celsius), humidity, weather description, and timestamp.
- The pipeline extracts this weather data and loads it into a MySQL database.

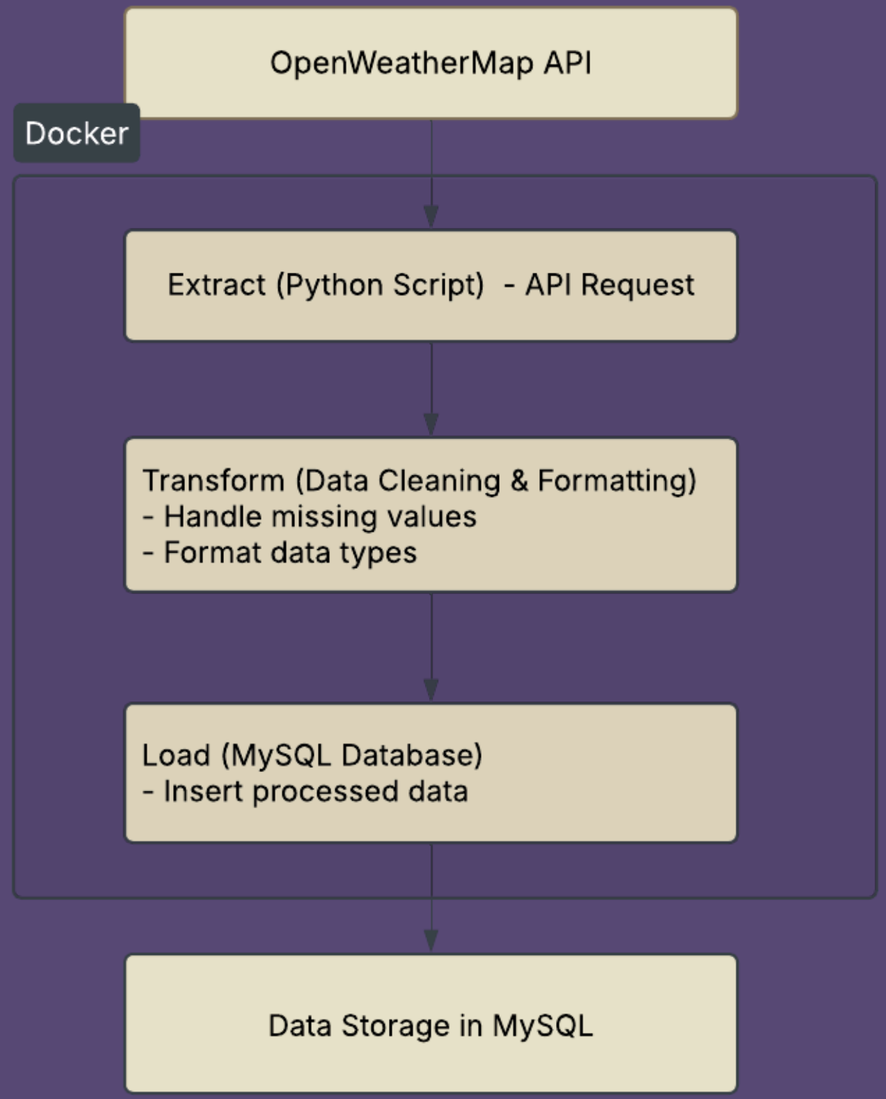
ARCHITECTURE

ETL Process:

- Extract: Weather data is extracted using the OpenWeatherMap API.
- Transform: Data is cleaned, formatted, and prepared for storage.
- Load: The cleaned data is loaded into a MySQL database.

Technologies:

- Python: ETL logic and transformation.
- Docker: Containerization of the pipeline for scalability and portability.
- MySQL: Relational database for storing the weather data.

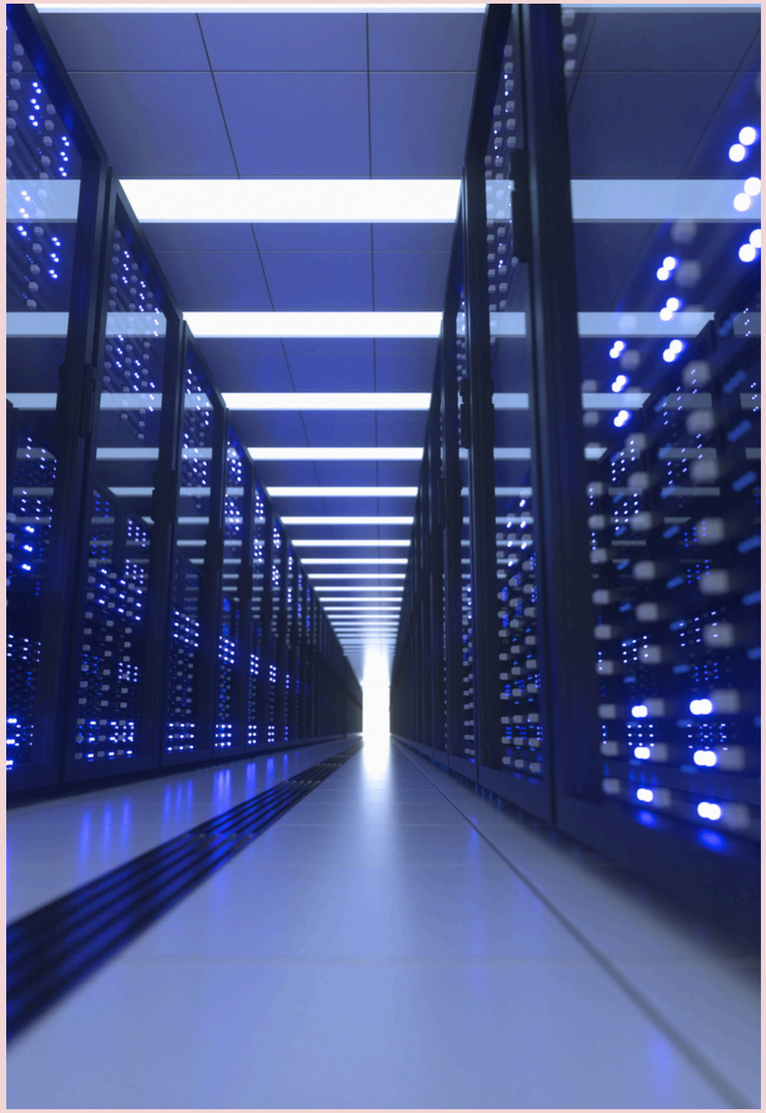


ETL PIPELINE FLOW

- Step 1: Extract weather data from OpenWeatherMap API.
- Step 2: Transform the data (clean and format the timestamp, handle missing values, etc.).
- Step 3: Load the transformed data into the MySQL database.
- Step 4: Schedule the pipeline to run periodically for automated data updates.

DOCKER CONTAINERIZATION

- Why Docker: Used to ensure that the ETL pipeline runs consistently across different environments.
- Steps:
- Dockerfile is created to define the container for the project.
- Docker Compose is used to manage the containerized MySQL database and the ETL pipeline.
- Benefits: Easy deployment and scalability, isolation of dependencies, and smooth handling of updates.

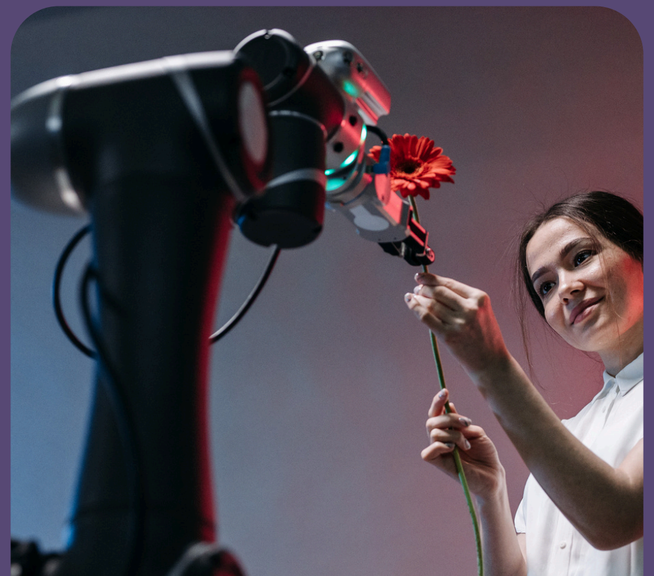


MYSQL DATA STORAGE

- Schema: Data is stored in the weather_data database, with a table named weather.
- Columns:
  - city, temperature_celsius, humidity, weather_description, timestamp.
- Key Features:
  - Timestamp is used to track when the weather data was recorded.
  - Multiple records for the same city and timestamp are allowed (but duplicates are managed).

AUTOMATING THE PIPELINE

Automation: The pipeline is automated using a Python script that is scheduled to run at regular intervals.



Key Benefits:

- Reduces the need for manual intervention.
- Ensures that the weather data in the database is continuously updated.

DATA ANALYSIS

- The extracted weather data was cleaned and transformed for analysis, focusing on key metrics like temperature, humidity, and wind speed.
- Statistical methods were applied to identify trends and patterns in the dataset.
- The analysis helped uncover relationships between weather parameters, providing insights into daily and seasonal variations.

Average Temperature per City:

Berlin: 1.51°C, London: 4.37°C, New York: 3.32°C, Paris: 4.54°C, Tokyo: 2.16°C

Average Humidity per City:

Berlin: 84.5%, London: 86%, New York: 92%, Paris: 75%, Tokyo: 39%

Highest Temperature Record:

Paris: 4.54°C at 2025-02-06 21:51:11, Overcast clouds

Lowest Temperature Record:

Berlin: -0.58°C at 2025-02-03 18:56:03, Clear sky

RESULTS & INSIGHTS

The ETL pipeline successfully loads data into MySQL with the following insights:

- A scalable system for weather data processing.
- Consistent and accurate data storage in MySQL.

The data sample represents weather information extracted from OpenWeatherMap API and loaded into the MySQL database.

City	Temperature_celsius	Humidity	Weather_description	Timestamp
Berlin	3.59	86	overcast clouds	2025-02-06 21:55:42
London	4.37	86	clear sky	2025-02-06 21:52:25
New York	3.32	92	mist	2025-02-06 21:51:14
Paris	4.54	75	overcast clouds	2025-02-06 21:51:11
Berlin	-0.58	83	clear sky	2025-02-03 18:56:03
Tokyo	2.16	39	few clouds	2025-02-06 21:55:59

THANK YOU