ETL Pipeline Report

Project Title: Weather and Air Quality ETL Pipeline for Sindh, Pakistan
Name: Naeem Sharif
Roll Number: DS047
Date: April 5, 2025

1. Introduction

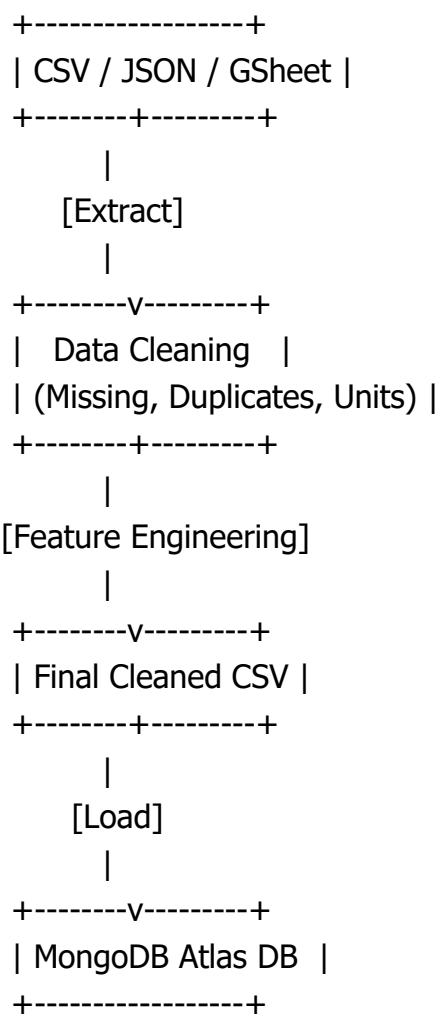
This project focuses on building an end-to-end ETL (Extract, Transform, Load) pipeline for collecting, processing, and storing weather and air quality data for the Sindh province of Pakistan. The goal is to automate the ingestion of data from diverse sources and transform it into a structured, unified format suitable for analysis and storage.

Data is extracted from:

- Local CSV files
- JSON files simulating API responses
- Google Sheets exported as CSV

The final cleaned and processed data is loaded into a **MongoDB Atlas** database hosted in the cloud.

2. **ETL Pipeline Architecture**



3. Technology Stack

Component	Technology
Language	Python
ETL Automation	schedule module
Version Control	Git + GitHub
CI/CD	GitHub Actions
Database	MongoDB Atlas (NoSQL)
Code Editor	Visual Studio Code (VS Code)

4. ETL Process

◇ Extraction

Three datasets were sourced:

- sample_data.csv — local CSV containing weather data
- sample_weather.json — JSON with weather + precipitation info
- google_sheet_sample.csv — CSV export from Google Sheets with air quality info

All datasets are stored in the data/ directory.

◇ Transformation

- **Data Cleaning**
  - Missing values handled using imputation or row drops
  - Duplicates removed
  - Invalid values filtered
- **Unit Conversion**
  - Temperatures converted from Fahrenheit to Celsius where applicable
- **Timestamp Formatting**
  - All dates formatted to ISO 8601
- **Feature Engineering**
  - A new weather_impact_score column was created using temperature, humidity, and wind speed

◇ Output

- Cleaned and transformed data is saved as output/final_cleaned_data.csv

5. Automation with Scheduler

- The pipeline is automated using Python's schedule module
- scheduler.py runs the etl_pipeline.py every day at 1:00 PM local time using:

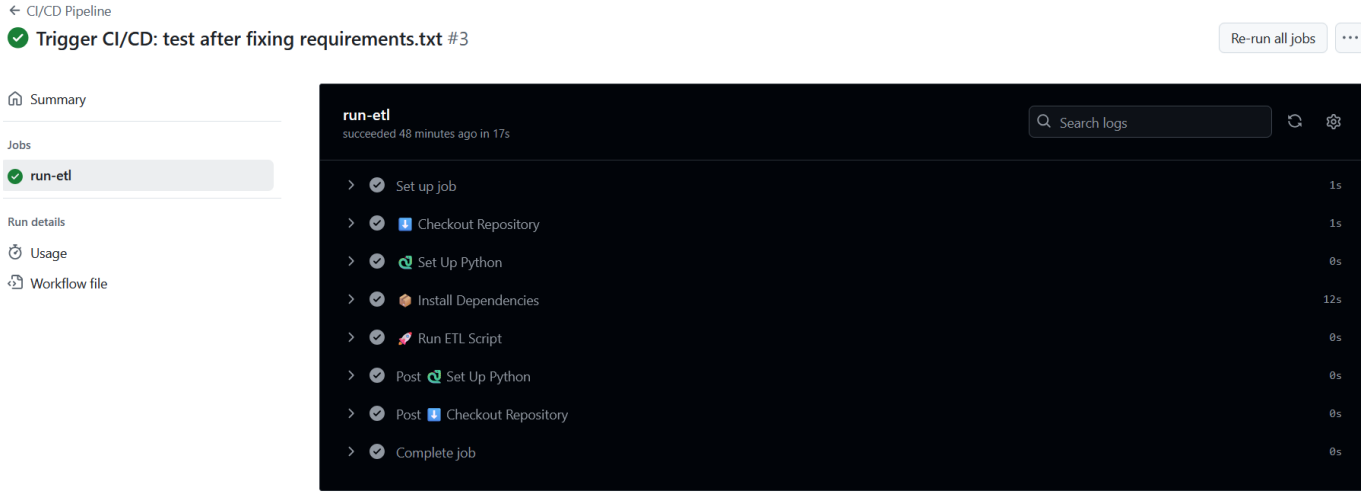
```
schedule.every().day.at("13:00").do(run_etl)
```

This ensures data is always up to date without manual intervention.

6. CI/CD with GitHub Actions

- GitHub Actions is configured via .github/workflows/ci_cd.yml
- Pipeline runs on every push or pull request
- Automates:
  - Python environment setup
  - Dependency installation

- Running ETL pipeline and checking for success
- Ensures reliability and faster feedback loops



7. Database Integration with MongoDB Atlas

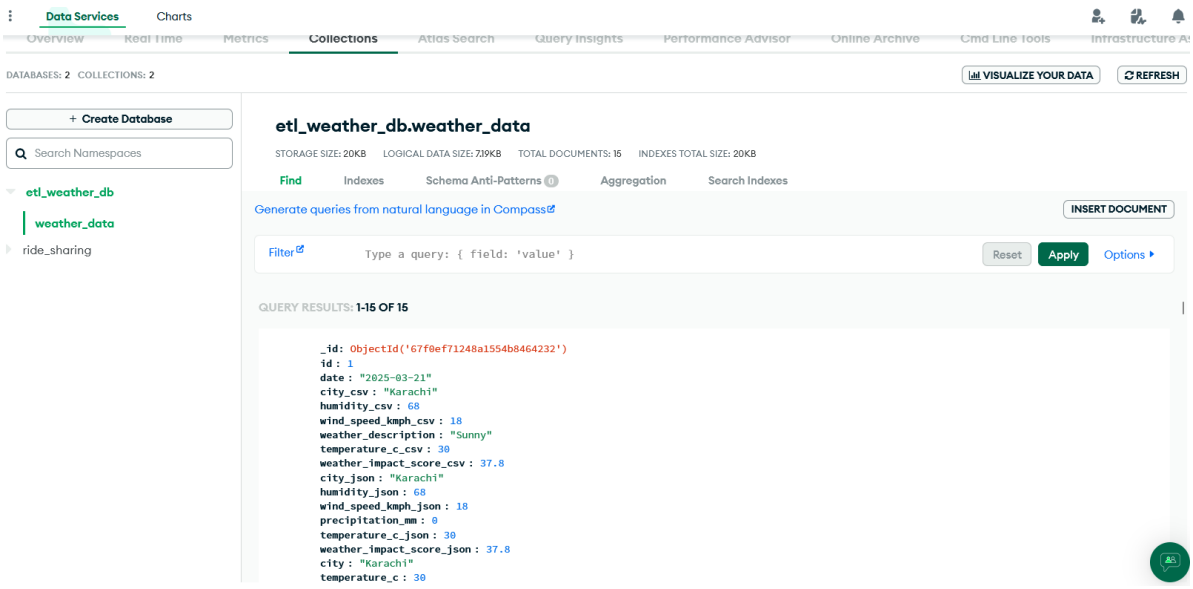
Instead of a traditional SQL database, the project uses **MongoDB Atlas**, a cloud-based NoSQL database.

Setup Steps:

- Created a MongoDB Atlas cluster
- Created a user with credentials
- Whitelisted local IP address
- Used pymongo to connect and insert data

```
{
  "uri": "mongodb+srv://<username>:<password>@cluster0.mongodb.net",
  "database": "etl_weather_db",
  "collection": "weather_data"
}
```

```
from pymongo import MongoClient
df = pd.read_csv("output/final_cleaned_data.csv")
collection.insert_many(df.to_dict(orient='records'))
```



8. Challenges Faced

- Initial setup of MongoDB Atlas URI and credentials
- Choosing the right DB (MongoDB over PostgreSQL)
- Dealing with inconsistent data formats across CSV/JSON/Sheet
- Creating a flexible but reusable cleaning pipeline
- CI/CD failures due to missing requirements (later fixed)

9. Conclusion

This ETL pipeline is robust, scalable, and well-automated. It supports:

- Multi-source ingestion
- Feature enrichment
- Automated loading into a cloud DB
- CI/CD-based reliability and easier deployment

This setup can easily be expanded with:

- Real-time API ingestion (e.g., OpenWeatherMap)
- Dashboards using Streamlit or Plotly
- Advanced analytics (e.g., anomaly detection, forecasts)

10. **Appendix**

- **GitHub Repo:** https://github.com/naeemsharifai/etl-pipeline-naeem.git
- **Folder Structure:**

```
ETL_Pipeline_NaeemSharif_DS047/
├── etl_pipeline.py
├── scheduler.py
├── load_to_db.py
├── config/db_config.json
├── data/
│   ├── sample_data.csv
│   ├── sample_weather.json
│   └── google_sheet_sample.csv
├── output/final_cleaned_data.csv
├── requirements.txt
├── README.md
├── .github/workflows/ci_cd.yml
├── report.pdf
```