Real-Time Air Quality Monitoring System

Big Data Pipeline with Kafka, Spark, and Multi-Database Architecture

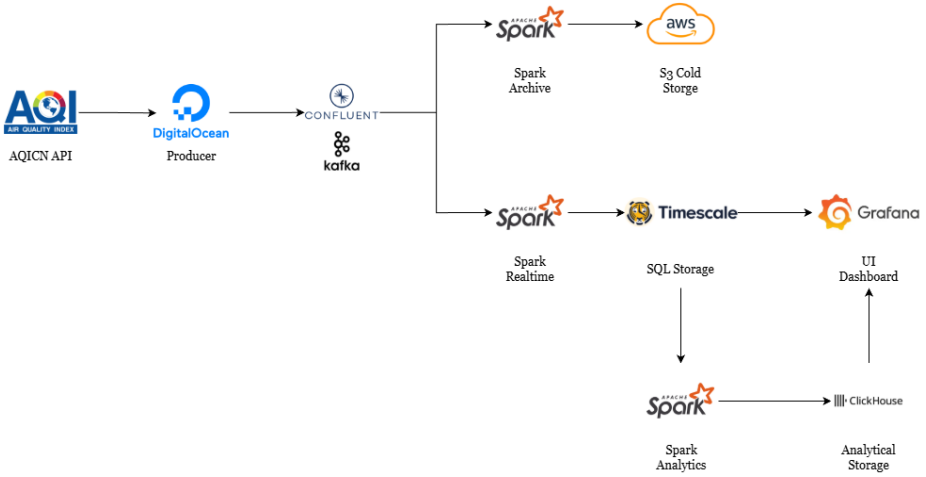
IT4931 - Big Data

December 10, 2025

Project Overview

- **Objective**: Build a scalable real-time air quality monitoring system for 7 Vietnamese cities
- **Data Source**: AQICN API (Air Quality Index Network)
- **Cities**: Hanoi, Ho Chi Minh City, Da Nang, Hai Phong, Can Tho, Nha Trang, Vung Tau
- **Update Frequency**: Every 5 minutes from cloud producers
- **Architecture**: Kappa

System Architecture



Technology Stack

Data Ingestion:

- Kafka (Confluent Cloud)
- Python Producers

Stream Processing:

- Apache Spark 3.5.0
- PySpark Structured Streaming
- Micro-batch (10 min trigger)

Storage:

- TimescaleDB (Time-series)
- ClickHouse (Analytics)
- MinIO (Object Storage)

Visualization:

- Grafana
- PostgreSQL & ClickHouse plugins

Data Pipeline Components

1. Data Producers (DigitalOcean VM)

- 7 independent Python scripts (one per city)
- Fetch data from AQICN API every 5 minutes
- Publish to dedicated Kafka topics: `raw.airquality.<city>`

2. Realtime Processing (Spark Streaming)

- Consumes from Kafka topics
- Enriches data with city metadata (population, region, thresholds)
- Calculates health status and pollution level
- Writes to TimescaleDB hypertables (partitioned by time)

Data Pipeline Components (cont.)

3. Archive Processing (Spark Streaming)

- Archives raw Kafka messages to MinIO
- Parquet format with Snappy compression
- Partitioned by date for compliance and replay

4. Batch Analytics (Spark)

- Hourly aggregations: Reads from TimescaleDB, writes to ClickHouse
- Daily aggregations: Summarizes hourly data
- AQI prediction: ML model using Linear Regression

Database Design - TimescaleDB

Schema: `<city>_measurements`

- **Purpose**: Store real-time sensor measurements
- **Partitioning**: Hypertable partitioned by `processed_time`
- **Columns**: AQI, PM2.5, PM10, O3, NO2, temperature, humidity, location
- **Enriched Fields**: health_status, pollution_level, city_metadata

Key Features:

- Automatic time-based partitioning
- Optimized for time-series queries
- PostgreSQL compatibility

Database Design - ClickHouse

Schema: `hourly_aggregates`, `daily_aggregates`

- **Purpose**: Fast analytical queries for dashboards
- **Engine**: MergeTree
- **Ordering**: (city, hour_start)

Aggregated Metrics:

- AQI: avg, max, min, stddev
- PM2.5 & PM10: avg, max, min
- Temperature & Humidity: avg
- Record count per hour/day

Deployment Configuration

Cloud Services:

- **Kafka**: Confluent Cloud (pkc-817wq.ap-east-1.aws.confluent.cloud)
- **Producers**: DigitalOcean VM (178.128.223.93)

Local Services (Docker):

- TimescaleDB: Port 5432
- ClickHouse: Port 8123 (HTTP), 9003 (Native)
- MinIO: 4-node cluster, Port 9000 (API), 9001 (Console)
- Grafana: Port 3000

Processing Jobs:

- 7 realtime streaming jobs (one per city)
- 7 archive streaming jobs
- 7 hourly analytics jobs

Data Flow Example

Realtime Processing Pipeline:

1. Producer fetches AQI data from AQICN API (every 5 min)
2. Data published to Kafka topic: `raw.airquality.hanoi`
3. Spark Streaming consumes JSON messages
4. Parse and validate data schema
5. Join with city metadata (broadcast)
6. Calculate derived fields:
   - health_status: Good/Moderate/Unhealthy/Hazardous
   - pollution_level: Low/Medium/High/Severe
7. Write to TimescaleDB: `hanoi_measurements`
8. Checkpoint state every 10 minutes

Code Structure

Source Code:

- `src/common/config.py`
- `src/ingestion/producer.py`
- `src/processing/realtime.py`
- `src/processing/archive.py`
- `src/processing/analytics_hourly.py`
- `src/processing/analytics_daily.py`
- `src/processing/aqi_prediction.py`

Run Scripts (per city):

- `scripts/produce_<city>.py`
- `scripts/run_realtime_<city>.py`
- `scripts/run_archive_<city>.py`
- `scripts/run_analytics_hourly_<city>.`
- `scripts/run_aqi_prediction_<city>.py`

Configuration:

- `docker-compose.yaml`: Infrastructure services
- `.env`: Credentials and endpoints
- `data/city_metadata.json`: Static reference data

Current Status

Completed:

- ✓ Kafka producers (7 cities)
- ✓ Docker infrastructure
- ✓ Realtime streaming pipeline
- ✓ TimescaleDB integration
- ✓ City metadata management
- ✓ Per-city run scripts

In Progress:

- Archive processing (MinIO)
- Hourly analytics aggregation
- ClickHouse integration
- Grafana dashboards
- AQI prediction models

Planned Visualizations (Grafana)

5 Dashboard Views:

1. **Real-time City Dashboard**
   - Current AQI, PM2.5, PM10 levels
   - Health status indicators
   - 24-hour trend charts
2. **Historical Trends**
   - Weekly/monthly AQI patterns
   - Pollution level comparisons
3. **City Comparison**
   - Side-by-side metrics for all 7 cities
   - Regional analysis (North/Central/South)

Planned Visualizations

4. **AQI Predictions**
   - ML-based forecasts (next 24 hours)
   - Confidence intervals
   - Feature importance
5. **Summary Statistics**
   - Daily/weekly aggregates
   - Best/worst air quality rankings
   - Health risk alerts

Future Enhancements

Short-term:

- Complete Grafana dashboard implementation
- Deploy ML prediction models
- Set up automated alerting

Long-term:

- Add more cities and sensors
- Implement Delta Lake for data lake
- Build REST API for external access
- Mobile app for public access
- Integrate weather data for correlation analysis

Conclusion

Project Achievements:

- Built end-to-end real-time data pipeline
- Integrated 5 different technologies (Kafka, Spark, TimescaleDB, ClickHouse, MinIO)
- Implemented Lambda architecture
- Handled 7 cities with scalable per-city processing
- Cloud-native design with local development environment

Technical Skills Demonstrated:

- Distributed stream processing
- Time-series database optimization
- Docker containerization
- Cloud service integration
- Data pipeline orchestration

Thank You!