Architecture for Real-Time Air Quality Monitoring and Personalized Recommendations

Johan Esteban Castaño Martínez, Stivel Pinilla Puerta, Jose Alejando Cortazar
Systems Engineering Program — Faculty of Engineering
Universidad Distrital Francisco José de Caldas, Bogotá, Colombia
jecastanom@udistrital.edu.co, spinillap@udistrital.edu.co, jacortazarl@udistrital.edu.co

Introduction

- Air pollution kills ~7–8 million people annually and ranks as the second leading risk factor for mortality, ahead of high blood pressure or smoking.
- In Bogotá, $PM_{2.5}$ levels declined from 15.7 µg/m^3 (2017) to 13.1 µg/m^3 (2019), yet spatially heterogeneous hot spots persist despite Air Plan 2030.
- Public APIs (AQICN, Google Air Quality, IQAir) provide real-time data but lack personalized recommendations and unified integration for citizen-oriented decision support.

Research Goal

- **Research Question:** How can we deliver highly-available, low-latency air-quality insights that adapt to user context (location, activity, health risk) in real time?
- **Objective:** Design and validate a PostgreSQL+TimescaleDB architecture that ingests multi-source AQI streams every 10 min, normalizes heterogeneous payloads, and produces personalized health recommendations within 2 s (p95) under 1000 concurrent users.

Proposed Architecture

- **Ingestor:** Python service polls AQICN, Google Air Quality, and IQAir APIs every 10 min; raw JSON archived in MinIO (`raw-airquality`) for audit and replay.
- **Normalizer:** Stateless mapping layer aligns field names, units, and AQI scales before inserting into TimescaleDB hypertable partitioned by *city* $\times$ *month*.
- **Query Acceleration:** Concurrently-refreshed materialized views prevent blocking reads during updates; dashboards access consistent snapshots.
- **Recommendation Engine:** Classifies AQI into EPA bands (Good 0–50, Moderate 51–100, Unhealthy $\geq$151, Hazardous >300) and combines with user metadata (location, activity, risk profile) to generate personalized health advice aligned with WHO guidelines. Suggests certified protective products (N95 masks, air purifiers) when AQI$\geq$151.
- **API Layer:** REST + GraphQL endpoints for citizens, researchers, and administrators; Grafana dashboards monitor ingest lag, query latency, and view-refresh duration.

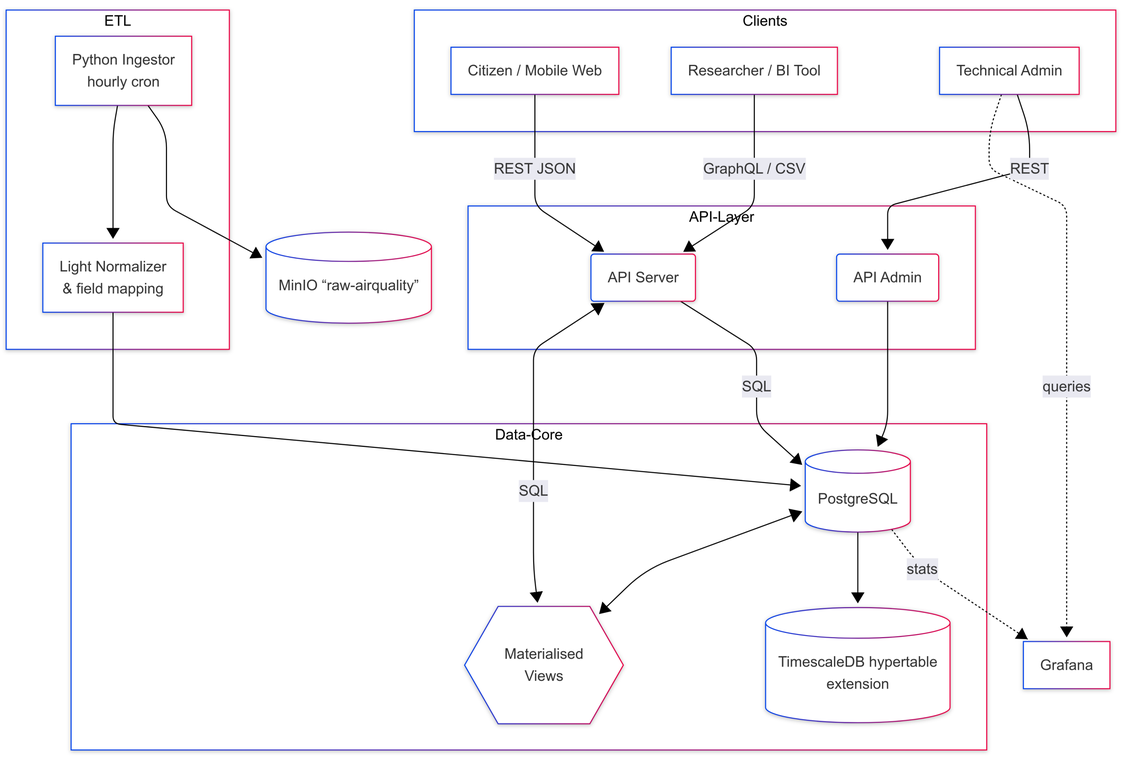

Figure 1: End-to-end architecture with three-layer data model (Geospatial, Customer, Recommendation Engine).

Experimental Setup

- **Dataset:** AQICN historical CSV (2015–2024). Phase 1 loads 2022–2024 Bogotá data (~2.5M rows/month under current ingestion rates).
- **Hardware:** Primary DB node: 4 vCPU, 16 GB RAM, NVMe storage; MinIO object storage for raw JSON archives.
- **Software:** PostgreSQL 17.1, TimescaleDB 2.14, Python 3.12, Grafana 11.
- **Load Test:** Apache JMeter simulates 1000 concurrent users, each issuing ~5 REST/GraphQL requests per second over a 10-minute test window.
- **Metrics:** Query latency (NFR1), dashboard load time (NFR6), report generation (NFR4), recommendation update frequency (NFR5), materialized-view refresh, CPU utilization, system uptime (NFR7).

Target Performance Metrics

Metric	Target (p95)	Requirement
Query latency ($\geq$1M rows)	$\leq$2 s	NFR1, NFR3
Dashboard load time	$\leq$2 s	NFR6, US12
Report generation	$\leq$10 s	NFR4
Recommendation update	10 min	NFR5
Materialized-view refresh	$\leq$5 s	Near real-time
Concurrent users	1000 users	US13, NFR8
System uptime	$\geq$99.9%	NFR7, US14
Peak CPU utilization	$<$70%	Headroom

Targets aligned with functional (FR1–FR14) and non-functional (NFR1–NFR10) requirements defined in project specification.

Conclusions and Future Work

Key Contributions:

- End-to-end PostgreSQL-based architecture unifying three public APIs with 10-min refresh cycles, MinIO raw storage, and monthly city-partitioned TimescaleDB hypertables.
- Lightweight recommendation engine translating AQI thresholds + user context into personalized health advice and certified product suggestions (FR8–FR11).
- Target performance: sub-2 s latency (p95) for 1000 concurrent users without distributed streaming frameworks (Kafka/Flink).

Future Directions:

- Predictive analytics: ARIMAX/LSTM forecasts for 6–24h pollution peaks.
- Performance optimization: read replicas, Redis caching, message queue (RabbitMQ/Kafka) for asynchronous ingestion.
- Geographic expansion: multi-region deployment across Latin American cities (Medellín, Cali, Santiago, Lima) using PostgreSQL 17 logical replication.

References

References

[1] WHO, "Global Air Quality Guidelines", 2021.
[2] State of Global Air, "Report 2024", 2024.
[3] AQICN, "Real-Time Air Quality Index API", 2024.
[4] Google, "Air Quality API Overview", 2024.
[5] IQAir, "AirVisual API Documentation", 2024.
[6] Timescale Inc., "Tables and Hypertables", 2025.
[7] PostgreSQL, "REFRESH MATERIALIZED VIEW", Docs.
[8] MinIO Inc., "Object Storage for Linux", 2024.
[9] Grafana Labs, "Dashboards Overview", 2024.