Governance & Operations Strategy

Project: Real-Time Weather Anomaly Detection Pipeline

1. Project Assumptions

To limit scope and ensure feasibility within the term project timeline, the following architectural and operational assumptions were made:

- **Source Truth:** We assume the **Open-Meteo API** provides accurate, calibrated meteorological data. The pipeline does not implement secondary validation logic to cross-reference temperatures against other providers (e.g., NOAA or AccuWeather).
- **Stationarity:** The ARIMA_PLUS model assumes that historical weather patterns (seasonality) are relatively stable. It does not account for massive, permanent climate shifts that might render 2024 training data irrelevant for 2030 predictions without retraining.
- **Infrastructure Availability:** We assume Google Cloud Platform (GCP) services (specifically Cloud Scheduler and Dataflow) operate with 99.9% uptime. The pipeline relies on "at-least-once" delivery from Pub/Sub; minor duplicate data points are considered acceptable for anomaly detection purposes.
- **Latency:** A latency of up to 5 minutes between data generation (API call) and dashboard visualization is acceptable for this specific use case (infrastructure risk monitoring), unlike High-Frequency Trading (HFT) which requires microsecond latency.

2. Data Ethics

While weather data is publicly available, ethical considerations regarding its usage and interpretation remain critical:

- **Open Data Compliance:** This project strictly adheres to the **Open-Meteo Terms of Service**. We utilize the free tier for educational purposes and respect rate limits (simulated via the 15-minute interval) to avoid burdening the public infrastructure.
- **Algorithmic Transparency:** The "Anomaly" designation is a statistical probability, not a ground truth. To prevent alarmism, the dashboard explicitly visualizes the *confidence bounds* (Min/Max Expected) alongside the alert, allowing human operators to interpret the severity rather than relying solely on a "Red/Green" black box.
- **Bias Mitigation:** The model is currently trained solely on New York City data. We explicitly acknowledge that this model cannot be ethically deployed for other

regions (e.g., Miami or Alaska) without retraining, as doing so would generate false positives due to geographical bias.

3. Privacy & Security Protocol

Since this project ingests public meteorological data, **no Personally Identifiable Information (PII)** is processed. However, security best practices were implemented to protect the infrastructure:

- **Secret Management:** The architecture was deliberately designed to use the **Open-Meteo API** (non-authenticated) to avoid the risk of hardcoding API keys in source code. If the project were to scale to a paid provider (e.g., OpenWeatherMap), API keys would be migrated to **Google Secret Manager** and accessed programmatically via IAM roles.
- **Least Privilege Access (IAM):**
    - **Cloud Functions:** Deployed with a dedicated Service Account limited to Pub/Sub Publisher permissions.
    - **Dataflow:** Workers operate with a minimal scope Service Account, restricted to reading Pub/Sub and writing to BigQuery.
- **Public Access Controls:** The Cloud Function allows unauthenticated invocations *only* for the purpose of this demo. In a production environment, this endpoint would be secured behind OIDC authentication, requiring the Cloud Scheduler to present a valid Google identity token.

4. Failure Playbook (Incident Response)

This playbook defines the standard operating procedures (SOPs) for common failure scenarios detected in the pipeline.

Scenario A: API Outage (HTTP 429/500)

- **Symptom:** Cloud Function logs show 429 Too Many Requests or 500 Server Error.
- **Automated Response:** The Python script includes a try/except block to catch these exceptions. It logs a "Warning" payload to Cloud Logging but returns a generic 500 status to prevent the Dataflow pipeline from crashing.
- **Manual Action:**
    1. Pause the **Cloud Scheduler** job (weather-poller) to stop hammering the API.
    2. Wait 30 minutes.
    3. Resume the Scheduler and verify data flow in the Looker Dashboard.

Scenario B: Data Pipeline Stall

- **Symptom:** Dashboard "Current Temp" has not updated in >30 minutes, but API status is green.
- **Root Cause:** Likely a jammed Dataflow worker or a Pub/Sub quota issue.

- **Manual Action:**
  1. Navigate to the **Dataflow Console**.
  2. Select the active job (weather-stream-job) and click **Drain** (stops accepting new data but finishes current items).
  3. Once drained, clone the job and restart it. This forces the provisioning of fresh worker instances.

Scenario C: Model Drift (False Positives)

- **Symptom:** The dashboard flags >50% of data points as anomalies despite normal weather.
- **Root Cause:** The historical training data (2024) no longer reflects current seasonal realities (e.g., an unseasonably warm winter).
- **Manual Action:**
  1. Run the **Phase 1 (Batch Ingest)** script to fetch the most recent 3 months of data.
  2. Run the **Phase 4 (Model Training)** SQL script to retrain ARIMA_PLUS on the updated dataset.
  3. The View will automatically use the new model for future scoring.