HyphaMetrics

Hyphametrics Technical Challenge: DevOps & Cloud Architect

Role: DevOps & Cloud Architect **Estimated Time:** 2–4 hours **Goal:** Demonstrate your ability to provision infrastructure, containerize a Python application, and design a production-ready deployment pipeline on Google Cloud Platform (GCP).

The Scenario
You are building a "Log Archiver" microservice. The goal of this service is to ingest message data from a Google Cloud Pub/Sub subscription and archive the content as JSON files into a Google Cloud Storage (GCS) bucket.

Part 1: Infrastructure as Code (Terraform)
Create a Terraform configuration that provisions the necessary GCP resources. **Requirements:**

1. **Pub/Sub:** One Topic and one Subscription.
2. **Storage:** One GCS Bucket (Standard class).
3. **IAM:** A Service Account with the *minimum necessary permissions* to read from the subscription and write to the bucket.
4. **Security:** Ensure the bucket is not publicly accessible.

Note: Please assume the state file is stored remotely (you can mock the backend configuration).

Part 2: The Application (Python & Docker)
Write a simplified Python script and a Dockerfile. **Requirements:**

1. **Python Script:**
    - It should accept configuration (Project ID, Subscription Name, Bucket Name) via **Environment Variables**.
    - It should mock the behavior of listening to a Pub/Sub message (you don't need to implement the full client if you prefer to pseudo-code the logic, but the structure must be valid).
    - Upon receiving a "message", it should upload a dummy JSON object to the GCS bucket.
    - Include basic error handling (what happens if the bucket doesn't exist?).
2. **Dockerfile:**
    - Create a production-ready Dockerfile for this script.
    - Focus on image size optimization and security (e.g., not running as root).

Part 3: CI/CD Pipeline (GitHub Actions)
Write a .github/workflows/deploy.yaml file that defines how this application would be deployed to **Google Kubernetes Engine (GKE)** or **Cloud Run**. **Requirements:**

1. The pipeline should trigger on a release tag specifying versioning (eg. v1.0.1).

2. It should build the Docker image and push it to Google Artifact Registry (GAR).
3. It should apply the Terraform changes.
4. **Important:** Explain in comments how you would authenticate GitHub Actions to GCP securely (avoiding long-lived JSON keys if possible).

Part 4: Architectural Questions (Short Answer)
Please answer the following briefly (2-3 sentences each):

1. **Secret Management:** How would you inject sensitive database credentials into the container at runtime in a GKE environment?
2. **Networking:** If this service needed to access a private MongoDB instance in another VPC, how would you architect the connectivity?
3. **Observability:** The service is failing silently in production. What tools in the GCP stack would you set up to alert you immediately?

Submission Guidelines

- Provide a link to a private or public GitHub repository to https://github.com/paulo guevarac OR a zipped folder containing the files.
- Include a README.md explaining how to run your solution and any assumptions you made.