Lab06

Managing Google Cloud Storage using Terraform

Contents

Scenario

You are working for a startup that runs a lightweight media-sharing platform. The platform team needs a secure, scalable way to:

- Store static media files (e.g. images, text)
- Prevent public access to objects over unsecured connections
- Automatically transition unused content to cold storage

You have been asked to use Terraform to provision the required infrastructure on Google Cloud Platform (GCP).

Business Requirements

Bucket Setup

Create a Google Cloud Storage (GCS) bucket in `us-central1` with a globally unique name

Enable **versioning** on the bucket

Secure Access Controls

Implement a **bucket policy** to **deny access to any user using HTTP** (unencrypted transport)

Static File Upload

Upload all files from a local directory called static_files

Ensure each object has the correct **MIME type** assigned, based on file extension

Storage Management

Apply a lifecycle policy to: - Transition objects to Nearline or Coldline storage after 30 days. - Delete them after 90 days.

Rules

Use Terraform only — no manual actions in the GCP Console unless debugging.

You may use the GCP Console to verify deployments, but not to modify resources.

The bucket name must include your initials or student ID to ensure global uniqueness (e.g. `googlelabs-mcg-static-5678`).

The lab environment is pre-authenticated — no need to handle GCP credentials in your code.

Success Criteria

GCS bucket appears in the GCP Console with versioning enabled.

Public access is blocked by IAM policy.

All files in `static_files/` are uploaded with correct content type metadata.

Lifecycle rules are visible and correctly configured.

Solution

A proposed solution to this challenge can be found in the solutions folder. Only use this as a last resort.

Student Reference Guide – GCS Terraform Challenge

GCS Bucket Basics
Access Control
Upload Multiple Files with MIME Type
Loading and Using JSON in Terraform