Objective:
This take-home test is designed to assess your ability to design, implement, and deploy a basic cloud infrastructure on Google Cloud Platform (GCP) using Terraform. It will involve creating a Cloud Function, exposing it through a Load Balancer , and securing the environment, showcasing your understanding of DevOps principles and GCP services.

Scenario:
You are tasked with creating a simple "Hello World" application deployed as a GCP Cloud Function, exposed through a GCP Load Balancer , and secured appropriately in a GCP Project. This test should showcase your skills in infrastructure-as-code, serverless computing, load balancing, and security within the GCP ecosystem.

Requirements:
Infrastructure as Code (Terraform):
- Use Terraform to define and provision all GCP infrastructure components.
- Ensure your Terraform code is well-structured, documented, and follows best practices (e.g., modularity, reusability, and proper naming conventions).
- Include a README file with clear instructions on how to deploy and destroy the infrastructure using Terraform.
- Ensure your implementation allows for multiple environments/projects {dev, tst, prd} to be created and managed independently

Project:
Create a new gcp project named: “SMT Take Home Exercise” with a project id that matches: smt-the-{env}-{yourname}-{random4char}

Cloud Function:
Create a simple GCP Cloud Function using (e.g., Node.js, Python, Go) that returns "Hello World" upon invocation.
The function should be stateless and idempotent.
Define the function's runtime, memory allocation, and any necessary environment variables.

Load Balancer:
- Configure a GCP Load Balancer (e.g., HTTP(S) Load Balancer) to distribute incoming traffic to the Cloud Function.
- Ensure the load balancer is properly configured for health checks and failover.
  Consider using a Serverless NEG (Network Endpoint Group) to connect the load balancer to the Cloud Function.

Security:
Implement appropriate security measures to protect your GCP infrastructure and application. This could include:
- Blocking known OWASP Exploits
- Network security (e.g., VPC Service Controls, firewall rules).
- Identity and Access Management (IAM) to control access to GCP resources.
- Secure configuration of the Cloud Function and Load Balancer .
- Consider using GCP Secret Manager for sensitive data if applicable.

Creativity and Flexibility:
Demonstrate creativity in your solution, going beyond the basic requirements. **This could include:**
- Implementing automated testing for your infrastructure using tools like Terratest.
- Using advanced Terraform features (e.g., modules, workspaces).
- Incorporating monitoring and logging with GCP Cloud Monitoring and Logging to track the health and performance of your application.
- Designing for scalability and high availability.
- Providing options for different environments (e.g., development, staging, production) using Terraform workspaces or variables.
-

Deliverables:

A Git repository containing your Terraform code, Cloud Function code, and any supporting files.
A README file with clear instructions on how to deploy and test your solution on GCP.

Acceptance/Validation

Your work will be validated by us running this, to create a new project in GCP.
Your Terraform should execute to create:

- Project
- Cloud Function (with hello world code deployed)
- Load Balancer
- Any other infra, configuration you create as part of this

Once your terraform is executed, if successful we should be able to:

- HTTP GET http://{IPAddressOfLoadBalancer}/helloWorld and get a response
- HTTP POST http://{IPAddressOfLoadBalancer}/helloWorld and get a failure
- HTTP GET {direct path to cloud function} get a 401 or 403 error

Follow Up Interview

If the Terraform Passes, in the follow interview you should be able to;

1. explain your design choices, benefits and negatives of your decisions
2. make changes for new infrastructure