DevOps Engineer Technical Test

This test aims to assess your skills in infrastructure as code (Terraform), CI/CD pipeline development (GitHub Actions), and Bash scripting within the context of a Cloud SQL, Cloud Run, static file bucket, Nginx, and PHP-FPM application.

We understand that the given time frame may be a constraint, but it is important that you manage your time efficiently and prioritize the challenges accordingly. We value quality over quantity, so we encourage you to take your time and provide well-documented and well-commented solutions.

We wish you the best of luck in completing these challenges and look forward to reviewing your work.

Deliverable: A public GitHub repository containing all code, configurations and documentation.

Time Limit: 4 hours

Test Structure:

1. Infrastructure as Code (Terraform) (1 hour)
   - Create a Terraform configuration for the following resources:
     - Google Cloud project (optional, only if not existing)
     - Cloud SQL instance (MySQL)
     - Cloud Storage bucket for static files
     - Cloud Run service with:
       - Container image built from a Dockerfile for your PHP-FPM application (**provide a simple PHP script as a starting point**)
       - Nginx configuration for serving static files and proxying requests to the PHP-FPM container
     - Cloud Load Balancing (HTTP(S)) to route traffic to the Cloud Run service
   - Refactor the Terraform configuration to follow best practices:
     - Use modules for reusable infrastructure components (e.g., Cloud SQL instance)
     - Leverage variables for environment-specific configurations (e.g., database name, Cloud Storage bucket name)
     - Implement state management (Terraform Cloud, remote backend, etc.)
   - Document your Terraform code with clear comments and references to relevant GCP documentation.
   - Commit your changes with descriptive messages reflecting your thought process.
2. CI/CD Pipeline with GitHub Actions (1.5 hours)
   - Create a GitHub Actions workflow file (.yml) to automate the following tasks upon code push:
     - Check out the code repository
     - Build the Docker image using Dockerfile
     - Push the image to a container registry (e.g., Google Container Registry)
     - Deploy the application to Cloud Run using gcloud CLI or Terraform
   - Configure environment variables for sensitive information (e.g., database credentials) using

GitHub secrets.
    - Utilize appropriate triggers for the workflow (e.g., push to main branch).
    - Include unit/integration tests for your PHP application (if applicable) and integrate them into the CI pipeline.
    - Document your GitHub Actions workflow with comments explaining each step.
    - Commit your changes with messages reflecting your approach to automation.
3. Bash Scripting (1 hour)
    - Write a Bash script that retrieves the public IP address of the deployed Cloud Run service.
    - Implement error handling for potential issues during script execution.
    - Extend the script to take an argument (e.g., environment name) and adapt the configurations accordingly (if applicable).
    - Consider logging script execution details for future reference.
    - Document the script functionality and usage with inline comments.
    - Commit your script with a descriptive message.
4. Readme File (0.5 hour)
    - Create a comprehensive Readme file in your GitHub repository explaining the project setup and functionalities.
    - Include instructions on:
        - Setting up Terraform environment variables (if needed)
        - Running the Terraform configuration to deploy infrastructure
        - Accessing the deployed application
        - Running the Bash script to retrieve the public IP address
        - Troubleshooting potential issues
    - Describe any problems encountered during the test and how you addressed them.
    - Mention any additional features you would consider implementing for a production-ready environment (e.g., monitoring, security configurations).

Additional challenges (for experienced candidates):

- Integrate Cloud Build into the CI/CD pipeline for container image building.
- Implement infrastructure as code for continuous delivery using Terraform Cloud or other tools.
- Utilize security best practices like IAM roles for authentication and authorization.
- Design a disaster recovery strategy for the application (Explain in a separate dr.md file in the repository.
- Add anything you think is interesting to showcase in this project and document it in a extra.md file in the repository