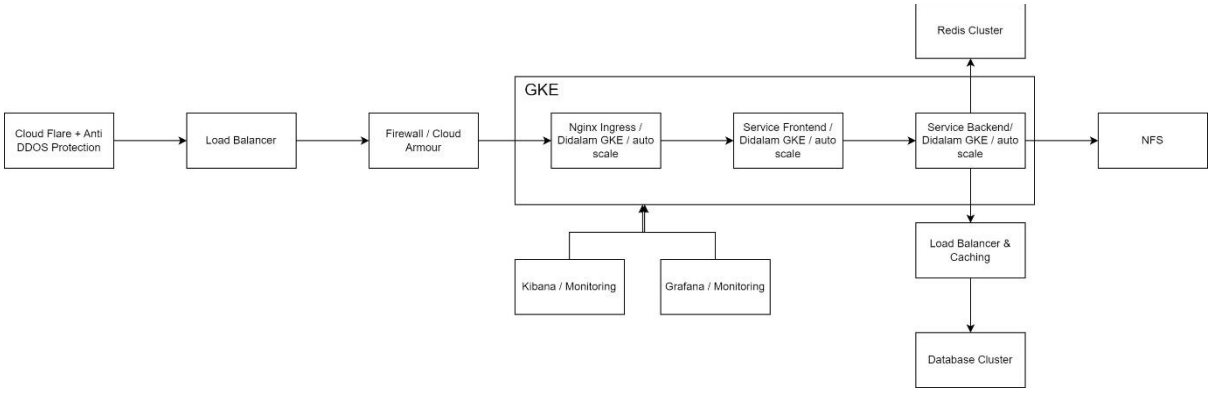
Nama / Job : Khaerul Anam / Devops Engineer

1. Buatlah arsitektur N-Tier server yang dapat menampung connection current sekitar 100 ribu user. Mohon untuk arsitektur yang dibuat berdasarkan item yang ada di Cloud Provider misal : Kubernetes, Droplet, Redis dll). Dapat Digambar menggunakan cloudcraft.co atau draw.io. Mohon untuk mencantumkan OS yang digunakan, tweak dan jenis Web Server (Apache,NginX) dan alasannya mengapa menggunakan feature itu di setiap item yang digunakan.

Jawaban :

Topologi :



Komponen :

Komponen : Load balancer, Firewall, Service Frontend, Service Backend, Database cluster, Cache Server (Redis), Monitoring Server

Keterangan :

1. Cload Flare : DNS Register + DDOS Protection
2. Load balancer : Load balancer GCP
3. Firewall : Pembatasan akses dan security policy
4. Nginx ingress : load balancer service. Keuntungan : memungkinkan untuk membagi menjadi domain internal dan eksternal dan memudahkan dalam management domain.
5. Service frontend : Nginx (autoscaling), os ubuntu 20.04 / Alphine
6. Service backend : os ubuntu 20.04 / Alphine
7. Load balancer database : Proxysql (mendukung caching dan pemisahan akses read dan write) sehingga beban tidak sepenuhnya di master
8. Cache server : Redis cluster untuk high availability cache server
9. Database : Mysql Cluster untuk high availability database
10. Storage : NFS untuk penyimpanan data

2. Implementasi CICD Deployment untuk case berikut (https://github.com/satrioolin/olintes)

- Berdasarkan dari repository berikut buatkan Dockerfile dengan menerapkan prinsip DevSecOps standar

Jawaban :

Isi Dockerfile :

```
# Use the official PHP 8.0 Apache base image
```

```
FROM php:8.0-apache

# Set the working directory to /var/www/html
WORKDIR /var/www/html

# Copy the current directory contents into the container at /var/www/html
COPY . /var/www/html

# Install any needed PHP extensions or packages
# RUN docker-php-ext-install pdo_mysql

# Expose port 80 for Apache
EXPOSE 80

# Define environment variables if necessary
# ENV MY_VARIABLE=my_value

# Start Apache when the container runs
CMD ["apache2-foreground"]
```

- Buatkan script terraform untuk melakukan automasi dalam pembuatan infrastruktur cloudnya

Jawaban :

Isi script terraform :

```
# Specify your provider and authentication settings
provider "google" {
 credentials = file("path/to/your/credentials.json")
 project     = "your-gcp-project"
 region      = "us-central1"
}

# Create a GKE cluster
resource "google_container_cluster" "my_cluster" {
 name               = "my-gke-cluster"
 location           = "us-central1"
 initial_node_count = 3
 node_locations     = ["us-central1-a", "us-central1-b", "us-central1-c"]

 master_auth {
   username = ""
   password = ""

   client_certificate_config {
     issue_client_certificate = false
   }
 }
}
```

```
# Create a Cloud SQL PostgreSQL database
resource "google_sql_database_instance" "my_database" {
  name             = "my-sql-instance"
  database_version = "POSTGRES_13"
  region           = "us-central1"
  project          = "your-gcp-project"

  settings {
    tier = "db-f1-micro"
  }
}

resource "google_sql_database" "my_app_db" {
  name     = "my-app-db"
  instance = google_sql_database_instance.my_database.name
}

# Create a Redis cluster (Cloud Memorystore)
resource "google_redis_instance" "my_redis" {
  name     = "my-redis-instance"
  tier     = "STANDARD_HA"
  location = "us-central1"
  project  = "your-gcp-project"
  memory_size_gb = 1
}

# Create a Google Cloud HTTP(S) Load Balancer
resource "google_compute_target_http_proxy" "lb_proxy" {
  name        = "lb-proxy"
  description = "My Load Balancer Proxy"
  url_map     = google_compute_url_map.lb_url_map.self_link
}

resource "google_compute_url_map" "lb_url_map" {
  name        = "lb-url-map"
  description = "My URL Map"
}

resource "google_compute_backend_service" "lb_backend_service" {
  name        = "lb-backend-service"
  description = "My Backend Service"
  protocol    = "HTTP"
  timeout_sec = 10
  port_name   = "http"

  backend {
    group = google_container_cluster.my_cluster.node_pool[0].instances_group_urls[0]
```

```
  }
}

resource "google_compute_global_forwarding_rule" "lb_forwarding_rule" {
  name       = "lb-forwarding-rule"
  description = "My Forwarding Rule"
  target     = google_compute_target_http_proxy.lb_proxy.self_link
  port_range  = "80"
}

# Define output variables if needed
output "gke_cluster_name" {
  value = google_container_cluster.my_cluster.name
}

# Output other resources as needed
```

Keterangan :
Fungsi script diatas digunakan untuk membuat server :

- A GKE cluster.
- A Cloud SQL PostgreSQL database.
- A Redis cluster (Cloud Memorystore).
- A Google Cloud HTTP(S) Load Balancer.

- Deploy menggunakan docker atau kubernetes ke dalam server (Linode atau Digitalocean)

  Jawaban :

  -

- Buatlah github action atau Jenkins file workflow untuk melakukan deployment ke server untuk di publikasi

  Jawaban :

  (di dalam repo)

- Daftarkan kedalam domain agar bisa diakses secara public

  Jawaban :

  -