Homework Sesi 15 Bootcamp DevOps Engineer Digitalskola

1. Buatlah 1 main.tf untuk membuat 1 instance di gcp menggunakan terraform

2. Jelaskan step by step untuk membuat 1 instance di gcp menggunakan terraform

Untuk memulai membuat instance di gcp menggunakan terraform, package terraform perlu diinstall terlebih dahulu. Berikut adalah langkah-langkahnya secara detail:

1. **Perbarui Daftar Paket (Update Package Lists):**

Pembaruan daftar paket di sistem Ubuntu adalah langkah pertama. Hal ini memastikan versi paket terbaru dan informasi repositori yang akurat didapatkan. Perintah **sudo apt update** akan mengunduh informasi paket terbaru dari semua repositori yang terkonfigurasi di sistem. Proses ini perlu ditunggu hingga selesai.

2. **Instal Paket yang Dibutuhkan (Install Required Packages):**

Sebelum instalasi Terraform, beberapa paket utilitas mungkin dibutuhkan untuk proses instalasi dan verifikasi kunci GPG. Perintah berikut dijalankan untuk instalasi paket-paket tersebut:

sudo apt install gnupg software-properties-common curl

- **gnupg**: Berguna untuk manajemen kunci GPG (GNU Privacy Guard). Ini penting dalam verifikasi tanda tangan paket dari HashiCorp.
- **software-properties-common**: Menyediakan utilitas untuk pengelolaan repositori perangkat lunak dan konfigurasi distribusi.
- **curl**: Utilitas baris perintah untuk transfer data dengan URL. Digunakan untuk mengunduh kunci GPG HashiCorp.

3. **Tambahkan Kunci GPG HashiCorp (Add HashiCorp GPG Key):**

Terraform didistribusikan oleh HashiCorp. Untuk memastikan paket Terraform yang diunduh asli dan tidak termodifikasi, kunci GPG HashiCorp perlu ditambahkan ke sistem. Perintah berikut dijalankan untuk mengunduh dan menambahkan kunci GPG:

curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg

Perintah ini melakukan beberapa hal:

- **curl -fsSL https://apt.releases.hashicorp.com/gpg**: Mengunduh kunci GPG dari server HashiCorp.

- **-f**: Berguna agar proses gagal secara diam-diam (tanpa output error) jika server melaporkan kesalahan.
- **-s**: Mode senyap. Progress meter atau pesan kesalahan tidak akan ditampilkan.
- **-S**: Saat kesalahan ditampilkan, kesalahan server akan diperlihatkan.
- **-L**: Jika server melaporkan URL yang diminta telah dipindahkan ke lokasi berbeda (ditunjukkan dalam header Lokasi), curl akan mengulangi permintaan di URL baru tersebut.

- **| sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg**: Mengirim output dari perintah curl ke perintah gpg.
  - **sudo gpg --dearmor**: Menjalankan perintah gpg dengan hak administrator untuk operasi. --dearmor mengubah kunci GPG ASCII menjadi format biner.
  - **-o /usr/share/keyrings/hashicorp-archive-keyring.gpg**: Menentukan file output untuk kunci GPG biner, yaitu /usr/share/keyrings/hashicorp-archive-keyring.gpg. Direktori /usr/share/keyrings/ adalah lokasi standar penyimpanan kunci GPG sistem.

4. Tambahkan Repositori HashiCorp APT (Add HashiCorp APT Repository):

Setelah penambahan kunci GPG, sistem APT Ubuntu perlu diberi tahu lokasi paket Terraform. Hal ini dilakukan dengan menambahkan repositori APT HashiCorp ke daftar sumber paket. Perintah berikut dijalankan:

echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

Perintah ini melakukan beberapa hal:

- **echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main"**: Mencetak baris konfigurasi repositori ke output standar.
  - **deb**: Menandakan baris ini adalah baris konfigurasi untuk repositori paket Debian (yang digunakan Ubuntu).
  - **[signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]**: Menentukan bahwa paket dari repositori ini harus ditandatangani oleh kunci GPG yang terletak di /usr/share/keyrings/hashicorp-archive-keyring.gpg. Ini adalah kunci yang baru saja ditambahkan.
  - **https://apt.releases.hashicorp.com**: URL dasar dari repositori APT HashiCorp.

- **$(lsb_release -cs)**: Perintah ini akan menjalankan lsb_release -cs dan menggantinya dengan nama kode rilis Ubuntu (misalnya, focal untuk Ubuntu 20.04, jammy untuk Ubuntu 22.04). Hal ini memastikan repositori yang benar untuk versi Ubuntu ditambahkan.
- **main**: Komponen repositori yang akan digunakan (biasanya main, universe, restricted, multiverse). Dalam kasus ini, hanya komponen main yang digunakan.

- **| sudo tee /etc/apt/sources.list.d/hashicorp.list**: Mengirim output dari perintah echo ke perintah sudo tee.
  - **sudo tee /etc/apt/sources.list.d/hashicorp.list**: Menulis output ke file /etc/apt/sources.list.d/hashicorp.list. tee memungkinkan penulisan ke file dengan hak administrator (sudo) dan juga menampilkan output ke terminal (walaupun dalam kasus ini output dari echo tidak ditampilkan di terminal karena dialihkan ke tee). File /etc/apt/sources.list.d/ adalah direktori tempat APT membaca file konfigurasi repositori tambahan. Pembuatan file hashicorp.list di direktori ini akan menambahkan repositori HashiCorp ke daftar sumber paket APT.

5. **Perbarui Daftar Paket Sekali Lagi (Update Package Lists Again):**

Setelah penambahan repositori HashiCorp, daftar paket APT perlu diperbarui sekali lagi agar sistem mengetahui paket Terraform yang tersedia dari repositori baru tersebut. Jalankan perintah:

sudo apt update

Tunggu hingga proses pembaruan daftar paket selesai. Informasi tentang repositori HashiCorp akan terlihat dalam output.

6. **Instal Terraform (Install Terraform):**

Instalasi Terraform siap dilakukan. Perintah berikut dijalankan untuk menginstal paket Terraform dari repositori HashiCorp:

sudo apt install terraform

Tekan Y jika konfirmasi instalasi diminta dan tunggu hingga proses selesai. Proses instalasi akan mengunduh dan menginstal Terraform beserta dependensinya.

7. **Verifikasi Instalasi Terraform (Verify Terraform Installation):**

Setelah instalasi selesai, verifikasi keberhasilan instalasi Terraform perlu dilakukan. Perintah berikut dijalankan untuk memeriksa versi Terraform yang terinstal:

terraform -v

Jika instalasi berhasil, perintah ini akan menampilkan versi Terraform yang terinstal.

Pembuatan instance GCP menggunakan Terraform

Langkah-langkahnya sebagai berikut:

1. **Buat Service Account di GCP:**
   Service Account akan digunakan sebagai identitas Terraform untuk berinteraksi dengan GCP.
   - **Buka Konsol Google Cloud Platform**: Akses konsol GCP melalui browser.
   - **Navigasi ke IAM & Admin > Service Accounts**: Pada menu navigasi kiri, cari dan klik IAM & Admin, kemudian pilih Service Accounts.
   - **Buat Service Account**: Klik tombol + CREATE SERVICE ACCOUNT di bagian atas halaman.
   - **Isi Detail Service Account**:
     - **Service account name**: Berikan nama yang deskriptif, contoh: terraform-service-account.
     - **Service account ID**: ID akan terisi otomatis, bisa disesuaikan jika perlu.
     - **Service account description**: Deskripsi opsional, contoh: Service account for Terraform to manage GCP resources.
     - Klik CREATE AND CONTINUE.
   - **Berikan Roles (Hak Akses)**: Pada bagian "Grant this service account access to project", berikan *role* yang sesuai dengan kebutuhan Terraform. Untuk membuat instance Compute Engine, *role* Compute Admin cukup. Pilih *role* dari dropdown "Select a role". Ketik "Compute Admin" pada kolom filter untuk mempercepat pencarian, kemudian pilih *role* Compute Admin. Klik CONTINUE.
   - **Grant Users Access to this service account (Opsional):** Langkah ini bisa dilewati jika tidak ada pengguna lain yang perlu diberikan akses ke Service Account ini. Klik **DONE**.
2. **Buat dan Unduh Service Account Key (JSON):**
   File kunci JSON akan digunakan oleh Terraform untuk autentikasi.
   - **Kembali ke Daftar Service Accounts**: Pada halaman Service Accounts, cari Service Account yang baru dibuat (terraform-service-account) dan klik pada nama Service Account tersebut.
   - **Navigasi ke Keys**: Klik tab **KEYS**.
   - **Tambahkan Key**: Klik **ADD KEY** dan pilih **Create new key**.
   - **Pilih JSON**: Pada "Key type", pilih **JSON**. Kemudian klik **CREATE**.
   - **Simpan File Kunci JSON:** File kunci JSON akan otomatis terunduh ke komputer. Simpan file ini di lokasi yang aman dan **catat lokasi file tersebut**, karena path file ini akan dibutuhkan dalam konfigurasi Terraform.
3. **Buat Direktori Proyek Terraform dan File main.tf:**
   - **Buat Direktori Proyek**: Buka terminal WSL Ubuntu dan buat direktori proyek (misalnya, terraform-gcp-instance-sa) dan masuk ke direktori tersebut:
     mkdir terraform-gcp-instance-sa
     cd terraform-gcp-instance-sa

- **Buat File main.tf:** Buat file main.tf dengan editor teks dan isi dengan konfigurasi berikut.

```
terraform {
  required_providers {
    google = {
      source  = "hashicorp/google"
      version = "~> 4.0"
    }
  }
}

provider "google" {
  project     = "data-rookery-453406-i3" # ID Proyek GCP
  region      = "asia-southeast2"    # Region Jakarta
  zone        = "asia-southeast2-a"  # Zone Jakarta
  credentials = file("/home/kaisenberg/terraform-gcp-instance-
sa/data-rookery-453406-i3-55f6fa9f2a31.json") # Path ke file kunci
JSON
}

resource "google_compute_instance" "default" {
  name         = "terraform-instance" # Nama instance
  machine_type = "e2-medium"
  zone         = "asia-southeast2-a"  # Zone harus konsisten dengan
zone provider (Jakarta)

  boot_disk {
    initialize_params {
      image = "ubuntu-os-cloud/ubuntu-2204-lts"
    }
  }

  network_interface {
    network = "default"
    access_config {
      # Konfigurasi IP publik
    }
  }
}
```

Penjelasan Konfigurasi main.tf:

- **terraform { ... }**: Blok ini mendefinisikan konfigurasi Terraform secara umum.
  - **required_providers { ... }**: Blok ini menentukan provider-provider yang dibutuhkan oleh konfigurasi ini.
    - **google = { ... }**: Mendefinisikan provider google dari HashiCorp.
      - **source = "hashicorp/google"**: Menentukan sumber provider Google, yaitu dari HashiCorp Registry.

        - **version = "~> 4.0"**: Menentukan versi provider Google yang digunakan. ~> 4.0 berarti Terraform akan menggunakan versi 4.0 atau versi yang kompatibel di atasnya (misalnya 4.1, 4.5), tetapi tidak versi 5.0 atau lebih tinggi.
- **provider "google" { ... }**: Blok ini mengkonfigurasi provider Google. Provider ini bertanggung jawab untuk berinteraksi dengan GCP API.
    - **project = " data-rookery-453406-i3"**: ID Proyek GCP yang telah dibuat di konsol Google Cloud.
    - **region = "asia-southeast2"**: Menentukan region GCP default yang akan digunakan.
    - **zone = "asia-southeast2-a"**: Menentukan zone GCP default dalam region **asia-southeast2-a**
    - **credentials = file("./nama-file-kunci-service-account.json")**: Baris ini secara eksplisit memberitahu provider Google untuk menggunakan kredensial dari file JSON yang ditentukan.
- **resource "google_compute_instance" "default" { ... }**: Blok ini mendefinisikan sumber daya (resource) yang akan dibuat, yaitu instance komputasi GCP.
    - **"google_compute_instance"**: Jenis sumber daya yang didefinisikan, yaitu instance komputasi Google Compute Engine.
    - **"default"**: Nama lokal (local name) untuk sumber daya ini dalam konfigurasi Terraform. Nama ini digunakan untuk mereferensikan sumber daya ini dalam konfigurasi Terraform lainnya.
    - **name = "terraform-instance"**: Nama instance GCP yang akan dibuat. Nama ini akan terlihat di konsol GCP.
    - **machine_type = "e2-medium"**: Tipe mesin (machine type) untuk instance. e2-medium adalah tipe mesin dengan konfigurasi memori dan vCPU yang seimbang dan cukup umum untuk aplikasi web sederhana atau server percobaan.
    - **zone = " asia-southeast2-a "**: Zone tempat instance akan dibuat. **Pastikan zone instance ini sama dengan zone yang telah didefinisikan di blok provider "google" { ... }.** Inkonsistensi zone dapat menyebabkan kesalahan saat terraform plan atau terraform apply.
    - **boot_disk { ... }**: Blok ini mendefinisikan konfigurasi disk boot untuk instance.
        - **initialize_params { ... }**: Blok ini mendefinisikan parameter inisialisasi untuk disk boot.
            - **image = "ubuntu-os-cloud/ubuntu-2204-lts"**: Image sistem operasi yang akan digunakan untuk disk boot.
    - **network_interface { ... }**: Blok ini mendefinisikan konfigurasi antarmuka jaringan untuk instance.

- **network = "default"**: Menentukan jaringan VPC (Virtual Private Cloud) yang akan digunakan instance. default adalah jaringan VPC default yang dibuat secara otomatis di setiap proyek GCP.
- **access_config { ... }**: Blok ini mendefinisikan konfigurasi akses eksternal (IP publik) untuk instance.
    - Blok ini dikosongkan ({}) agar instance mendapatkan IP publik ephemeral (sementara).

4. **Pindahkan File Kunci JSON (Jika Perlu):**
   Pastikan file kunci JSON yang diunduh berada di lokasi yang tepat sesuai dengan path yang ditentukan di **main.tf**. Dalam **main.tf** di atas, file kunci berada di direktori yang sama dengan **main.tf**.
5. **Inisialisasi Terraform:**
   Pada direktori **terraform-gcp-instance-sa**, jalankan perintah **terraform init.**

```
kaisenberg@MENOMEN:~/terraform-gcp-instance-sa$ terraform init
```

   Perintah **terraform init** akan melakukan beberapa hal:
   - **Inisialisasi Backend:** Jika backend dikonfigurasi (dalam contoh ini tidak dikonfigurasi, sehingga menggunakan backend lokal default), Terraform akan menginisialisasi backend untuk menyimpan state Terraform.
   - **Unduh Provider:** Terraform akan membaca blok required_providers dalam main.tf dan mengunduh provider Google yang dibutuhkan dari Terraform Registry. Provider disimpan di direktori .terraform.
   - **Inisialisasi Modul:** Jika konfigurasi menggunakan modul, modul-modul tersebut juga akan diinisialisasi.

   Tunggu hingga proses inisialisasi selesai. Output akan menunjukkan bahwa provider Google telah berhasil diinisialisasi.

6. **Rencanakan Perubahan:**
   Selanjutnya, jalankan perintah **terraform plan** untuk melihat rencana perubahan infrastruktur yang akan dilakukan oleh Terraform berdasarkan konfigurasi di **main.tf**.

```
kaisenberg@MENOMEN:~/terraform-gcp-instance-sa$ terraform plan
```

   Perintah **terraform plan** akan:
   - **Baca Konfigurasi:** Membaca file konfigurasi Terraform (main.tf, dll.).
   - **Refresh State:** Membaca state infrastruktur saat ini (dalam kasus pertama ini, state masih kosong karena belum ada infrastruktur yang dibuat oleh Terraform).
   - **Buat Plan:** Membandingkan konfigurasi yang diinginkan dengan state saat ini, dan menghasilkan rencana perubahan yang perlu dilakukan untuk mencapai konfigurasi yang diinginkan.

   Output dari terraform plan menunjukkan daftar tindakan yang akan dilakukan Terraform. Dalam kasus ini, karena ini adalah pertama kali instance dibuat, output akan

menunjukkan tindakan + create untuk resource google_compute_instance.default. Periksa output terraform plan dengan seksama untuk memastikan rencana sesuai dengan yang diharapkan. Perhatikan sumber daya yang akan dibuat, diubah, atau dihapus.

7. **Terapkan Perubahan:**

   Jika rencana di terraform plan sudah sesuai dan yakin ingin membuat instance GCP, jalankan perintah **terraform apply** untuk menerapkan perubahan dan membuat instance di GCP.

```
kaisenberg@MENOMEN:~/terraform-gcp-instance-sa$ terraform apply
```

   Perintah terraform apply akan:

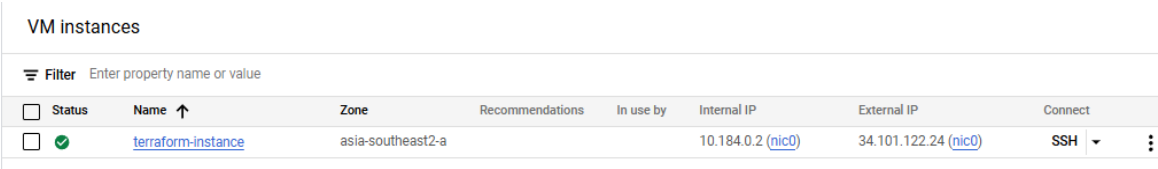
   - **Tampilkan Rencana (Sekali Lagi):** Menampilkan rencana perubahan yang sama seperti output terraform plan untuk konfirmasi akhir.
   - **Minta Konfirmasi:** Meminta konfirmasi untuk menerapkan perubahan dengan mengetik yes dan menekan Enter.
   - **Terapkan Perubahan:** Jika dikonfirmasi, Terraform akan memanggil GCP API untuk membuat sumber daya yang didefinisikan dalam konfigurasi (google_compute_instance.default dalam kasus ini).
   - **Simpan State:** Setelah berhasil membuat instance, Terraform akan menyimpan state infrastruktur yang baru dibuat ke dalam file state (secara default terraform.tfstate di direktori lokal). State ini penting untuk manajemen infrastruktur selanjutnya.

   Ketik yes ketika diminta konfirmasi, dan tunggu hingga proses **terraform apply** selesai. Output akan menunjukkan progress pembuatan instance dan akhirnya pesan **Apply complete! Resources: 1 added, 0 changed, 0 destroyed.** yang menandakan bahwa instance telah berhasil dibuat.

8. **Verifikasi di Konsol GCP:**

   Setelah terraform apply selesai, verifikasi bahwa instance GCP telah dibuat.

   - Klik ke Compute Engine -> Instance VM.
   - Pastikan berada di proyek GCP yang benar (pada HW ini adalah proyek dengan id `data-rookery-453406-i3`)
   - Instance dengan nama **terraform-instance** sudah ada. Periksa detail instance, seperti zone, tipe mesin, dan statusnya.



DETAILS | OBSERVABILITY | OS INFO | SCREENSHOT

Basic information

Name	terraform-instance
Instance Id	3455086530278516996
Description	None
Type	Instance
Status	Running
Creation time	Mar 12, 2025, 4:46:24 PM UTC+07:00
Location	asia-southeast2-a
Instance template	None
In use by	None
Physical host	None
Maintenance status	–
Reservations	Automatically choose (default)
Labels	None
Tags	–
Deletion protection	Disabled
Confidential VM service	Disabled
Preserved state size	0 GB

Machine configuration

Machine type	e2-medium (2 vCPUs, 4 GB Memory)
CPU platform	AMD Rome
Minimum CPU platform	None
Architecture	x86/64
vCPUs to core ratio	–
Custom visible cores	–
All-core turbo-only mode	–
Display device	Disabled