# LAPORAN

**INSTALASI DAN KONFIGURASI TERRAFORM**

**CLOUD COMPUTING**



**OLEH :**

**ILMI FAIZAN**

**E1E1 20 011**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS HALU OLEO**

**KENDARI**

**2023**

# KATA PENGANTAR

Dengan menyebut nama Allah SWT yang maha pengasih lagi maha penyayang, penulis panjatkan puja dan puji syukur atas kehadiran-Nya, yang telah melimpahkan rahmat serta hidayah-Nya kepada kami, sehingga penulis dapat menyelesaikan laporan tentang “Instalasi dan konfigurasi Terraform”.

Laporan ini telah disusun dengan maksimal dan mendapatkan bantuan dari beberapa pihak sehingga dapat memperlancar pembuatan laporan ini. Untuk itu kami menyampakan banyak terima kasih kepada semua pihak yang telah berkontribusi dalam pembuatan laporan ini.

Terlepas dari semua itu, penulis menyadari sepenuhnya bahwa masih ada kekurangan baik dari segi susunan kalimat maupun tata bahasanya. Oleh karena itu dengan tangan terbuka kami menerima segala saran dan kritik agar kami dapat memperbaiki laporan ini.

Akhir kata penulis berharap semoga laporan tentang “Instalasi dan Konfigruasi Terraform” ini dapat memberikan manfaat maupun inspirasi terhadap pembaca.

Kendari, 29 Desember

Penyusun

# DAFTAR ISI

[HALAMAN SAMPUL i](#_Toc154785548)

[KATA PENGANTAR ii](#_Toc154785549)

[DAFTAR ISI iii](#_Toc154785550)

[DAFTAR GAMBAR iv](#_Toc154785551)

[BAB I PENDAHULUAN 1](#_Toc154785552)

[1.1 Latar Belakang 1](#_Toc154785553)

[1.2 Rumusan Masalah 1](#_Toc154785554)

[1.3 Tujuan 2](#_Toc154785555)

[1.4 Manfaat 2](#_Toc154785556)

[BAB II LANDASAN TEORI 3](#_Toc154785557)

[2.1 Terraform 3](#_Toc154785558)

[2.2 Perbedaan Terraform dan Ansible 5](#_Toc154785559)

[BAB III PEMBAHASAN 7](#_Toc154785560)

[3.1 Alat dan Bahan 7](#_Toc154785561)

[3.2 Prosedur Instalasi 7](#_Toc154785562)

[3.3 Langkah – Langkah Instalasi Terraform di Windows 7](#_Toc154785563)

[3.4 Contoh Konfigurasi Menggunakan Terraform 9](#_Toc154785564)

[BAB IV PENUTUP 13](#_Toc154785565)

[4.1 Kesimpulan 13](#_Toc154785566)

[4.2 Saran 13](#_Toc154785567)

[DAFTAR PUSTAKA 14](#_Toc154785568)

# DAFTAR GAMBAR

[Gambar 3. 1 Bahan Instalasi Terraform 7](#_Toc154785569)

[Gambar 3. 2 Menyalin alamat folder installer terraform 8](#_Toc154785570)

[Gambar 3. 3 Konfigrasi path terraform di windows 8](#_Toc154785571)

[Gambar 3. 4 Mengecek Hasil Instalasi Terraform 8](#_Toc154785572)

[Gambar 3. 5 Inisialisasi Provider Terraform 9](#_Toc154785573)

[Gambar 3. 6 Mendefinisikan sumber daya instance EC2 10](#_Toc154785574)

[Gambar 3. 7 Inisialisasi variabel instance\_name 10](#_Toc154785575)

[Gambar 3. 8 Menampilkan Output 11](#_Toc154785576)

[Gambar 3. 9 Tampilan seluruh konfigurasi 12](#_Toc154785577)

# BAB I PENDAHULUAN

## Latar Belakang

Infrastuktur as Code adalah sebuah pengelolaan infrastruktur TI yang menggunakan file konfigurasi. Tujuan dari infrastruktur sebagai kode adalah agar dapat secara otomatis untuk mengelola, memantau, menyediakan sumber daya tanpa konfigurasi manual. Bisa kita katakan infrastruktur ini sebagai kode mirip dengan script pemrograman. Namun, ia menggunakan bahasa tingkat tinggi atau deskriptif untuk membuat kode. Keberadaan pengembang IaC dan tim secara otomatis mengelola tumpukan teknologi untuk aplikasi melalui perangkat lunak, membantu mengkonfigurasi dan menerapkan komponen ini dengan cepat dan konsisten.

Terraform awal berkembang bersama HashiCorp yang merupakan infrastruktur sumber terbuka yang populer sebagai alat perangkat lunak kode yang memungkinkan membuat, mengubah, dan meningkatkan infrastuktur dengan aman dan dapat diprediksi. Terraform memungkinkan otomatisasi dalam manajemen infrastruktur sebagai kode (IaC). Dengan mendefinisikan infrastruktur dalam berkas konfigurasi, pengguna dapat memulai, menghentikan, dan memperbarui sumber daya dengan mudah tanpa campur tangan manual.

Dengan menyajikan cara yang terstruktur, konsisten, dan otomatis untuk mengelola infrastruktur, Terraform membantu organisasi untuk lebih responsif terhadap kebutuhan bisnis, meningkatkan keamanan, dan mengoptimalkan penggunaan sumber daya IT. Keberlanjutan dan pertumbuhan infrastruktur dapat dicapai dengan lebih efisien melalui pendekatan IaC yang diusung oleh Terraform.

## Rumusan Masalah

1. Bagaimana cara instalasi Terraform di Windows 10?
2. Bagaimana cara konfigurasi Terraform di Windows 10?

## Tujuan

1. Mengetahui proses instalasi Terraform pada sistem operasi Windows.
2. Mengetahui proses konfigurasi Terraform pada sistem operasi Windows.

## Manfaat

1. Dapat mengetahui proses instalasi Terraform pada sistem operasi Windows.
2. Dapat mengetahui proses konfigurasi Terraform pada sistem operasi Windows.

# BAB II LANDASAN TEORI

## Terraform

Terraform adalah perangkat lunak open-source yang diluncurkan oleh HashiCorp. Dengan keberadaan Terraform pengguna dapat dengan mudah dan menentukan pusat data menggunakan bahasa konfigurasi deklaratif yang kita kenal sebagai HashiCorp Configuration Language (HCL) atau optional JSON. File Konfigurasi menjelaskan kepada Terraform komponen yang kita perlukan untuk menjalankan satu aplikasi atau seluruh pusat data Anda. Terraform menghasilkan rencana eksekusi yang menjelaskan apa yang akan kita lakukan untuk mencapai keadaan yang kita inginkan selanjutnya mengeksekusinya untuk membangun infrastuktur yang dijelaskan. Ketika konfigurasi berubah, Terraform dapat menentukan apa yang berubah dan membuat rencana eksekusi tambahan yang dapat kita terapkan.

Seiring dengan perkembangan teknologi, infrastruktur IT semakin kompleks. Organisasi seringkali mengelola infrastruktur di berbagai penyedia cloud dan lingkungan on-premises, serta berbagai layanan seperti mesin virtual, penyimpanan, jaringan, dan banyak lagi. Manajemen manual infrastruktur yang semakin rumit ini menjadi tantangan. Infrastruktur sebagai Kode (IaC) menjadi semakin populer. Dalam paradigma ini, infrastruktur didefinisikan dan dikelola menggunakan skrip atau kode, mirip dengan cara kita menulis kode untuk aplikasi perangkat lunak. Pendekatan ini memungkinkan otomatisasi, konsistensi, dan skalabilitas yang lebih baik. Meskipun sudah ada alat manajemen konfigurasi seperti Chef dan Puppet, masih ada kebutuhan untuk alat yang dapat mengelola infrastruktur di tingkat yang lebih tinggi dan bekerja secara lintas-platform tanpa terlalu berkutat dengan konfigurasi detail.

Terraform yang dikembangkan oleh HashiCorp, muncul sebagai solusi untuk mengatasi tantangan tersebut. Beberapa hal dibalik munculnya dan prinsip utama lahirnya Terraform adalah:

1. **Infrastruktur sebagai Kode (IaC) :** Terraform didesain dengan pendekatan IaC, memungkinkan pengguna mendefinisikan infrastruktur menggunakan konfigurasi berbasis kode.
2. **Multi-Cloud dan Multi-Environment:** Terraform dirancang untuk mendukung multi-cloud dan multi-environment. Ini berarti Anda dapat menggunakan Terraform untuk mengelola infrastruktur di berbagai penyedia cloud dan lingkungan infrastruktur.
3. **Deklaratif dan Berkas Kode**: Terraform menyediakan bahasa konfigurasi yang deklaratif dan mudah dipahami. Konfigurasi Terraform didefinisikan dalam berkas kode (seperti HCL - HashiCorp Configuration Language) yang memungkinkan pembacaan, pemahaman, dan kerja sama tim.
4. **Perencanaan dan Aplikasi yang Aman:** Terraform memberikan perencanaan sebelum menerapkan perubahan, memungkinkan pengguna melihat dampak perubahan sebelum diimplementasikan. Terraform menciptakan dan mengelola sumber daya dengan cara yang aman dan idempoten, memastikan bahwa keadaan dijaga sesuai dengan konfigurasi yang diinginkan.
5. **Ekosistem Modul dan Penyedia:** Terraform memiliki ekosistem modul dan penyedia yang kaya, memungkinkan pengguna untuk menggunakan dan berbagi konfigurasi infrastruktur yang telah dibuat dan teruji.
6. **Komunitas dan Dukungan yang Kuat:** Terraform mendapatkan dukungan dan kontribusi yang kuat dari komunitas open-source, yang berarti perkembangan dan pemeliharaannya aktif dan responsif terhadap kebutuhan pengguna.

Terraform telah menjadi salah satu alat utama untuk manajemen infrastruktur sebagai kode, memberikan fleksibilitas dan kontrol yang dibutuhkan oleh organisasi modern dalam mengelola lingkungan IT.

## Perbedaan Terraform dan Ansible

Ansible dan Terraform adalah dua *tools* yang sering digunakan sebagai platform infrastruktur sebagai kode (IaC). Meskipun tujuannya berbeda, kedua *tools* tersebut memiliki beberapa perbedaan. Terraform adalah platform IaC open source. Sedangkan, Ansible adalah alat manajemen konfigurasi sumber terbuka. Pengembang dapat menggunakan Ansible dan Terraform secara bersamaan tanpa mengganggu atau menggantikan kebutuhan antara satu dengan yang lain.

Terraform, dibuat oleh HashiCorp, menyediakan sintaks untuk menentukan infrastruktur dan layanan yang dapat dihosting di lokasi atau di cloud. Terraform dapat menggunakan layanan cloud untuk mengonfigurasi mulai dari layanan Azure hingga mesin virtual yang berjalan di VMware ESXi hingga entri DNS di Cloudflare.

Melihat perbedaan antara dua tools IaC yaitu Terraform dan Ansible, sangat penting untuk memahami kekuatan dan kelemahan masing-masing kedua platform. Terraform lebih berfokus pada manajemen infrastruktur sebagai kode dan pembuatan serta pengelolaan sumber daya, sementara Ansible lebih berorientasi pada otomatisasi konfigurasi dan penyebaran aplikasi. Berikut adalah tabel perbandingan antara Terraform dan Ansible, dalam manajemen konfigurasi dan otomatisasi:

|  |  |  |
| --- | --- | --- |
| **Kriteria** | **Terraform** | **Ansible** |
| Fungsi utama | Manajemen Infrastruktur dan Layanan Cloud | Otomatisasi Konfigurasi dan Penyebaran |
| Konfigurasi | HCL (HashiCorp Configuration Language) | YAML |
| Mode Operasi | Deklaratif | Imperatif |
| Pendekatan | Penyedia Berbasis dan Sumber Daya | Node-based (Agentless) |
| Fleksibilitas Infrastruktur | Memungkinkan untuk mengelola sumber daya secara terpusat | Lebih berfokus pada konfigurasi aplikasi dan sistem perangkat lunak |
| Pengelolaan Konfigurasi | Tidak dirancang untuk manajemen konfigurasi tingkat perangkat (configuration drift) | Dirancang untuk manajemen konfigurasi tingkat perangkat |

# BAB III PEMBAHASAN

## Alat dan Bahan

Alat dan bahan yang digunakan dalam proses penginstalan Sistem Operasi Proxmox pada Virtua Box adalah sebagai berikut :

1. Laptop / PC sebagai tempat berlangsungnya proses instalasi.
2. File installer Terraform versi Windows.
3. Visual Studio Code sebagai tempat penulisan kode dan konfigurasi Terraform.

## Prosedur Instalasi

Prosedur instalasi virtual box dan Linux Ubuntu Server 22.04 LTS yaitu sebgai berikut :

1. Menyiapkan alat dan bahan terlebih dahulu.
2. Menginstal Terraform dan pengaturan path instalasi.
3. Membuat konfigurasi sederhana menggunakan Terraform

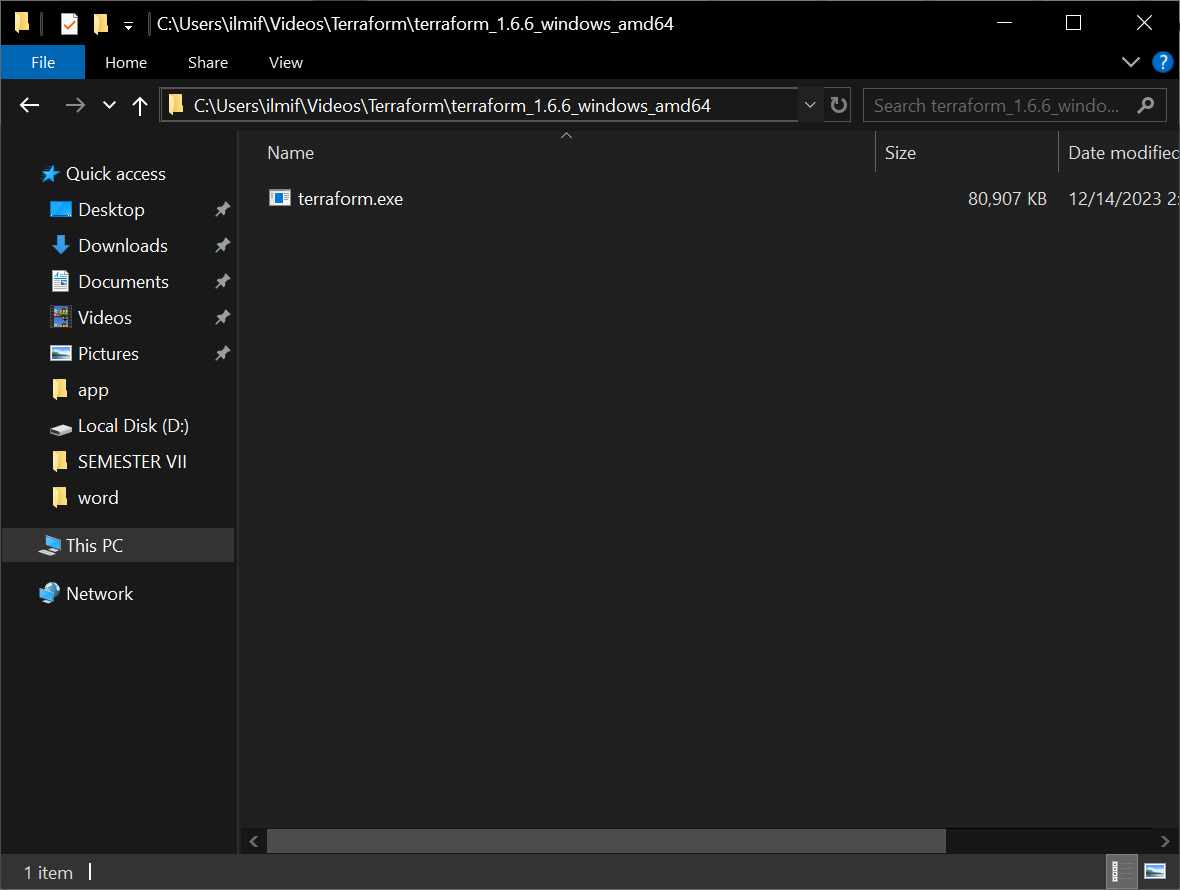
## Langkah – Langkah Instalasi Terraform di Windows

1. Download bahan instalasi melalui situs resmi Terraform pada link berikut <https://developer.hashicorp.com/terraform/install#Windows>. Kemudian extract bahan menggunakan Winrar atau aplikasi serupa. Hasil download dapat dilihat pada gambar berikut.



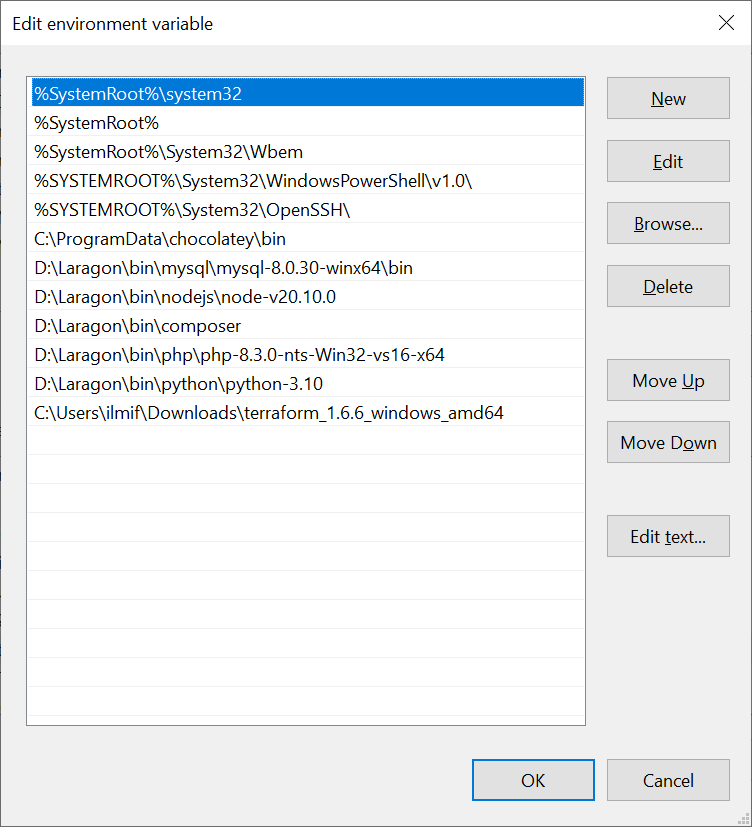
Gambar 3. 1 Bahan Instalasi Terraform

1. Setelah itu masuk ke dalam folder yang telah diextract dan copy alamat path folder. Pada bagian yang ditandai merupakan alamat path dari folder.



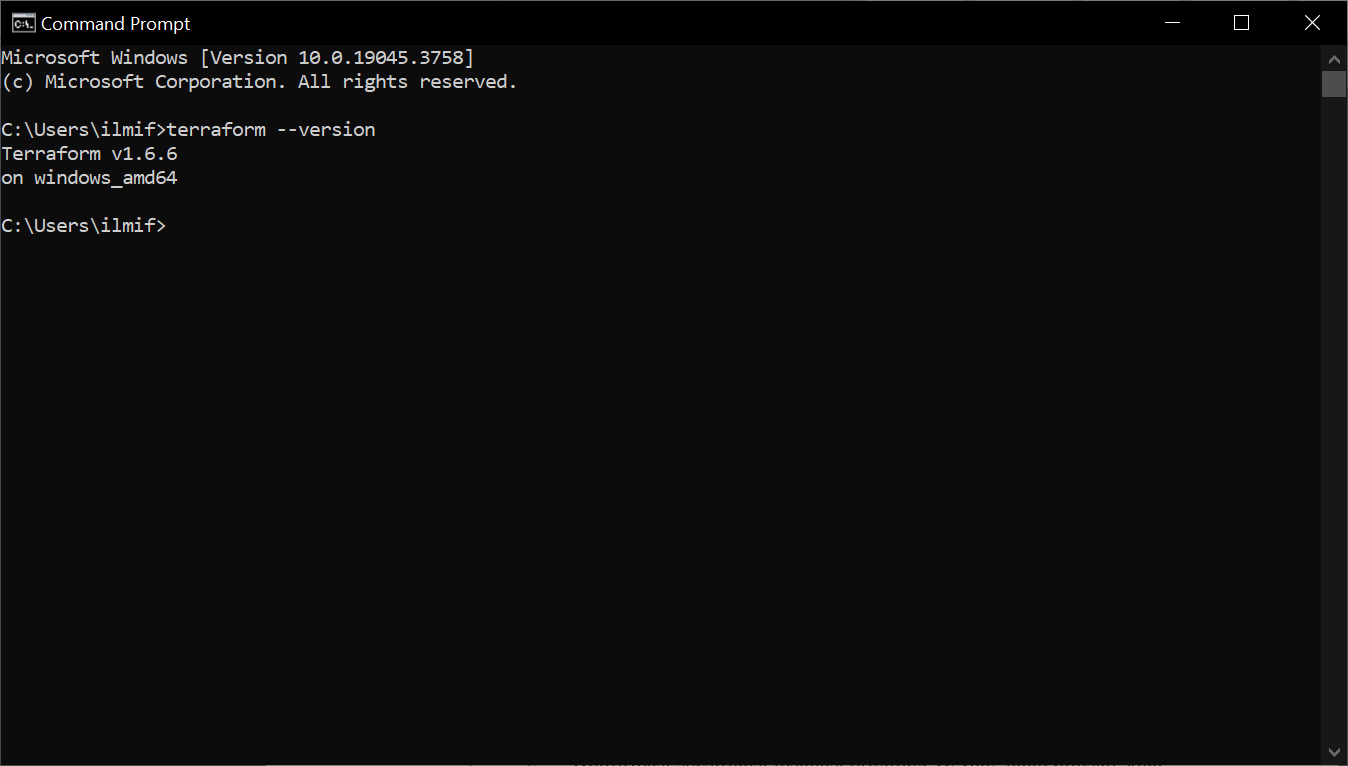
Gambar 3. 2 Menyalin alamat folder installer terraform

1. Pastekan alamat path folder installer terraform pada Environment Variable – Path Windows.



Gambar 3. 3 Konfigrasi path terraform di windows

1. Untuk mengecek apakah Terraform sudah terinstal dengan baik silakan buka *command prompt* kemudian ketikan *terraform –version*. Jika muncul tampilan seperti gambar maka instalasi telah berhasil.



Gambar 3. 4 Mengecek Hasil Instalasi Terraform

## Contoh Konfigurasi Menggunakan Terraform

Pada contoh kali ini kita akan membuat dan mengkonfigurasi EC2 instance pada Terraform. Amazon Elastic Compute Cloud atau Amazon EC2 adalah layanan komputasi awan yang disediakan oleh Amazon Web Services (AWS). Dalam konteks Terraform, "EC2 instance" merujuk pada mesin virtual yang di-host di lingkungan komputasi EC2 di AWS. EC2 instances adalah unit komputasi utama yang memungkinkan pengguna untuk menjalankan aplikasi dan beban kerja di cloud AWS.

Langkah awal yang dilakukan adalah menginisiasi provider apa yang ingin digunakan. Dalam contoh ini kita AWS sebagai provider.



Gambar 3. 5 Inisialisasi Provider Terraform

* *provider "aws"* : Mendefinisikan penyedia (provider) AWS untuk konfigurasi. Penyedia ini akan digunakan untuk berinteraksi dengan layanan AWS.
* *region = "us-west-2"* : Menentukan wilayah AWS di mana sumber daya akan dibuat. Dalam contoh ini, wilayahnya adalah "us-west-2".

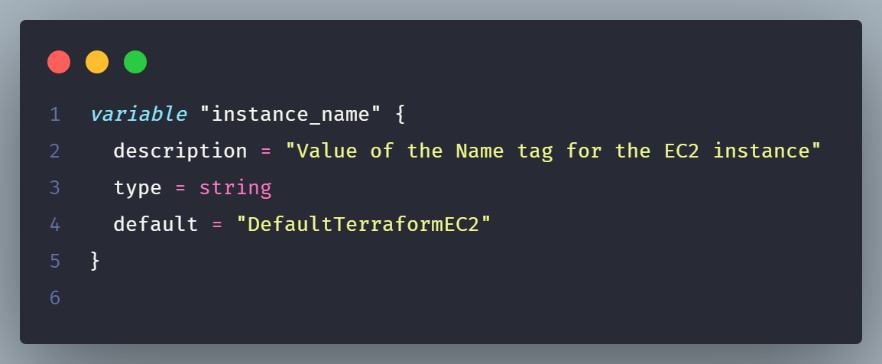
Tahap berikutnya adalah menginisialisasi atau mendefinisikan sumber daya instance EC2 dapat dilihat pada gambar berikut.



Gambar 3. 6 Mendefinisikan sumber daya instance EC2

* *aws\_instance "app\_server"* : Menentukan bahwa kita akan membuat instance EC2 dan memberikan alias "app\_server" untuk merujuk ke sumber daya dalam konfigurasi.
* *ami = "ami-0c55b159cbfafe1f0"* : Menentukan Amazon Machine Image (AMI) yang akan digunakan untuk instance EC2.
* *instance\_type = "t2.micro"* : Menentukan jenis instance, dalam hal ini *"t2.micro"* yang memiliki sumber daya komputasi yang terbatas.
* *key\_name = "personal-virginia"* : Menentukan nama key pair yang akan digunakan untuk mengakses instance melalui SSH.
* *tags = { Name = var.instance\_name }* : Menentukan tag untuk instance, dalam hal ini "Name" akan diambil dari variabel instance\_name.

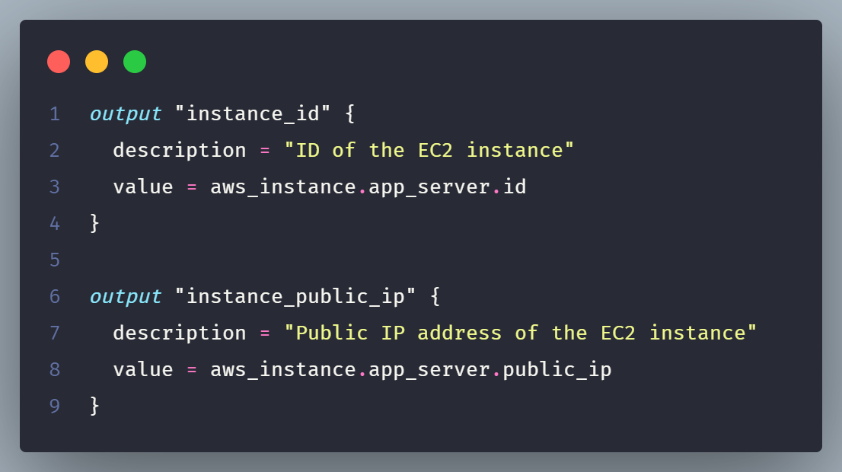
Pada potongan kode sebelumnya *var.instance\_name* belum didefinisikan. Oleh karena itu kita definisikan terlebih dulu.



Gambar 3. 7 Inisialisasi variabel instance\_name

* *variable "instance\_name"* : Mendefinisikan variabel yang digunakan untuk memberikan nilai pada tag "Name" untuk instance EC2.
* *description = "Value of the Name tag for the EC2 instance"* : Memberikan deskripsi singkat tentang variabel.
* *type = string* : Menentukan tipe data variabel sebagai string.
* *default = "DefaultTerraformEC2"* : Memberikan nilai default untuk variabel.

Langkah terakhir membuat *output* untuk menguji instance yang telah dibuat. Berikut potongan baris kodenya.



Gambar 3. 8 Menampilkan Output

* *output "instance\_id"* : Menentukan output yang akan memberikan informasi ID instance EC2.
* *output "instance\_public\_ip"* : Menentukan output yang akan memberikan informasi alamat IP publik instance EC2.

Dengan menggunakan konfigurasi tersebut, Terraform dapat membuat dan mengelola instance EC2 di wilayah "us-west-2" dengan konfigurasi dan tag yang telah ditentukan. Variabel dan output memberikan fleksibilitas dan memberikan informasi setelah konfigurasi diterapkan. Berikut konfigurasi Terraform untuk seluruh potongan kode apabila disatukan.



Gambar 3. 9 Tampilan seluruh konfigurasi

# BAB IV PENUTUP

## Kesimpulan

Berdasarkan laporan yang telah disusun, dapat disimpulkan bahwa Terraform adalah alat yang kuat untuk manajemen infrastruktur sebagai kode, memberikan fleksibilitas dan kontrol yang dibutuhkan oleh organisasi modern. Instalasi dan konfigurasi Terraform pada Windows dijelaskan dengan rinci, dan contoh konfigurasi untuk EC2 instance memberikan pemahaman praktis tentang penggunaan Terraform.

## Saran

Adapun saran penulis yaitu agar selalu memperhatikan setiap tahapan dalam penginstalan terutama dalam hal konfigurasi saat instalasi untuk menghindari terjadinya kesalahan atau error.

# DAFTAR PUSTAKA

Ashtari, H. (2022, 09 22). *Terraform vs. Ansible: Memahami Perbedaan Utama*. Retrieved from spiceworks: https://www-spiceworks-com.translate.goog/tech/devops/articles/terraform-vs-ansible/?\_x\_tr\_sl=en&\_x\_tr\_tl=id&\_x\_tr\_hl=id&\_x\_tr\_pto=tc

Cloudhost. (2021, 11 23). *Mengenal Terraform : Pengertian dan Cara Kerja*. Retrieved from Cloudhost: https://idcloudhost.com/panduan/mengenal-terraform/

Kumar, S. (2023, 09 05). *Terraform vs. Ansible: Perbedaan Utama dan Perbandingan Tools*. Retrieved from K21Academy: https://k21academy-com.translate.goog/ansible/terraform-vs-ansible/?\_x\_tr\_sl=en&\_x\_tr\_tl=id&\_x\_tr\_hl=id&\_x\_tr\_pto=tc

Levan, M. (2021, 05 26). *Terraform vs. Ansible: Apa perbedaan alat DevOps ini?* Retrieved from theserverside: https://www-theserverside-com.translate.goog/answer/Terraform-vs-Ansible-How-do-these-DevOps-tools-differ?\_x\_tr\_sl=en&\_x\_tr\_tl=id&\_x\_tr\_hl=id&\_x\_tr\_pto=tc

Wijaya, K. (2017, 02 06). *mediun*. Retrieved from Mengenal Terraform untuk Automatisasi Pembuatan Infrastruktur Server: https://medium.com/@ikrisnaw/automatisasi-pembuatan-infrastruktur-server-menggunakan-terraform-763fc07838f