Laporan Tugas 1 Cloud Computing

Nama : Unedo Viery Kristenzky Tampubolon
Nrp : 5025221116
Link Github : https://github.com/unedtamps/CloudComputing-Task1

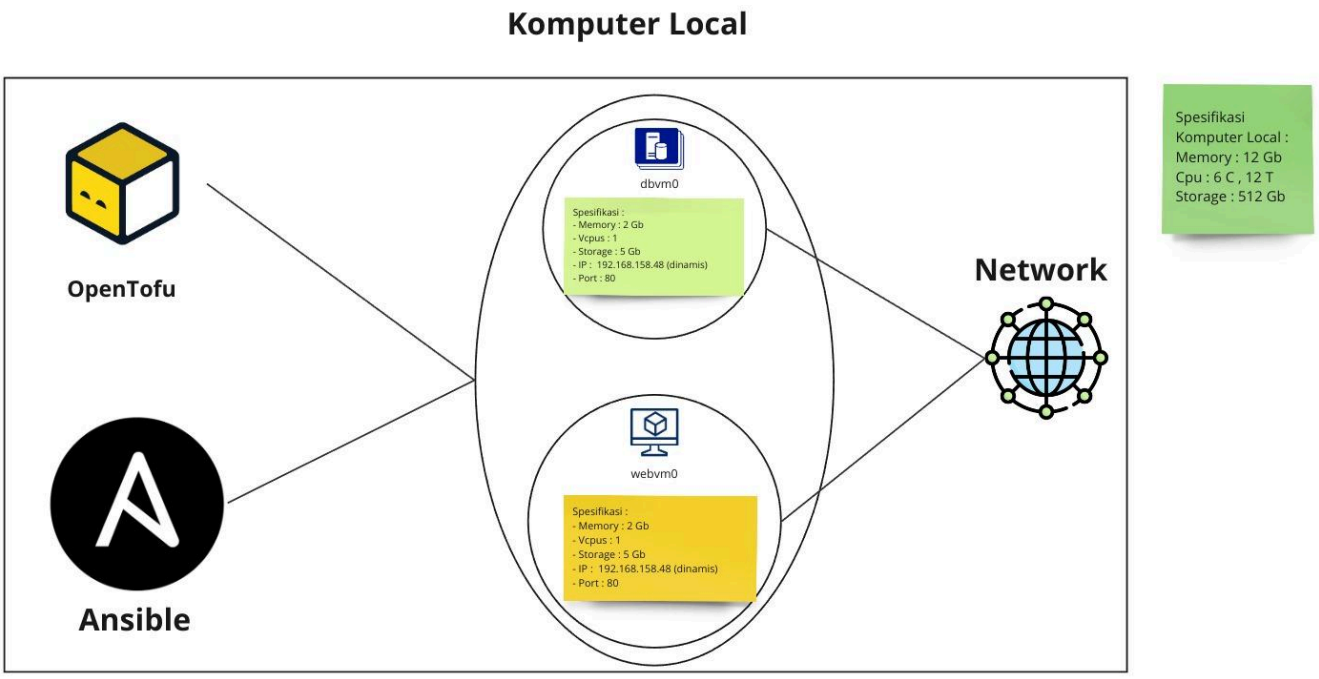
Pendahuluan

Laporan ini akan melakukan implementasi pembuatan virtual mechine di lingkungan local dengan menggunakan tools seperti open tofu dan ansible. Implementasi ini memanfaatkan kedua tools tersebut untuk mengotomatisasi provisioning dan konfigurasi virtual mechine serta menyederhanakan proses provisioning.

Tugas:

Berikut ini adalah detail tugasnya secara lengkap

1. Menggunakan open source tool cloud orchestration yang berbasis QEMU/KVM yang berparadigma infrastructure as code (Open Tofu dan Ansible)
2. Menggunakan tools tersebut untuk melakukan untuk melakukan provisioning pada
    - Website berbasis apache web server dengan isi website dari repo (https://github.com/rm77/web-sample-6.git)
    - Mysql database server beserta aplikasi pengelolaannya berbasis web (phpmyadmin)



Gambar 1 Arsitektur Percobaan

Penjelasan Arsitektur

Diagram ini menggambarkan sistem virtualisasi yang berjalan di dalam komputer lokan. Arsitektur percobaan ini akan melakukan integrasi antara opentofu dan ansible untuk mengatur virtual machine dan VMs. Spesifikasi dari komputer lokal yang digunakan adalah 12GB memory , CPU 6 core 12 thread dan storage 512GB yang akan berperan sebagai host untuk dua VM dengan nama ***dbvm0*** dan ***webvm0*** **.** Kedua VM akan di provision dengan 2 GB memory , 1 Virtual CPU dan 5 GB storage. Keduanya menggunakan IP address dinamis yang didapatkan setelah selesai membuat VM dan nantinya aplikasi yang berjalan akan berada di port 80.

Open Tofu disini sebagai “arsitek” yang memiliki tanggung jawab untuk:

1. Melakukan provisioning dan manajemen pembuatan virtual machine ***dbvm0*** dan ***webvm0***
2. Mengatur konfigurasi dari virtual machine vCPU, memory dan storage yang dibutuhkan oleh vm
3. Memastikan bahwa sumber daya komputer lokal didistribusikan secara efisien untuk masing masing VM.

Ansible disini berperan sebagai “pengelola” dari sistem ini. Ansible akan digunakan setelah VMs selesai di provision. Konfigurasi , deployment software dan ongoing update akan di handle oleh ansible . Berikut ini adalah beberapa hal yang dilakukan oleh ansible:

1. Melakukan konfigurasi pada VM, setelah VM selesai dibuat oleh opentofu maka selanjutnya ansible akan melakukan instalasi kebutuhan - kebutuhan untuk perangkat lunak yang ingin di deploy
2. Mengotomasi perangkat lunak , melakukan update secara berkala dan sebagainya
3. Mengelola perubahan , ansible dapat melacak adanya perubahan pada sistem dan melakukan perbaikan jika ada kesalahan

Kedua VM ini dapat konek ke internet karena network sudah di konfigurasi menggunakan NAT. Kedua VM juga dapat berkomunikasi satu sama lain karena kedua VM sudah berada di network yang sama. Contohnya ***dbvm0*** dapat diguanakan sebagia database server dan ***webvm0*** dapat berperan sebagai web server. Arsitektur ini umumnya digunakan untuk testing local dan di lingkungan testing saja, kalau sudah di lingkungan produksi biasanya akan menggunakan provider cloud computing pihak ketiga.

Penjelasan Kode dan Script

1. Opentofu

Pada penjelasan sebelumnya telah dijelaskan bahwa kegunaan opentofu pada tugas kali ini. Penjelasan berikutnya adalah mengenai setiap kode yang digunakan dalam provisioning vm.

a. Konfigurasi Kode Terraform

- **Terraform Block**

```
terraform {
  required_providers {
    libvirt = {
      source = "dmacvicar/libvirt"
    }
  }
}
```

Terraform block ini berfungsi untuk menggunakan provider yang dibutuhkan untuk menjalankan script terraform. required providers mendefinisikan provider libvirt dari sumber “dmacvicar/libvirt”. Provider ini membuat terraform dapat melola sumber daya basis QEMU/KVM

- **Template File Data Source**

```
data "template_file" "db_user_data" {
  template = file("config/db/user-data")
}

data "template_file" "db_network-config" {
  template = file("config/db/network-config")
}

data "template_file" "web_user_data" {
  template = file("config/web/user-data")
}

data "template_file" "web_network-config" {
  template = file("config/web/network-config")
}
```

Template file berfungsi untuk membaca file template berupa user data dan network-config yang akan digunakan di dalam vm. Di dalam repo kita bisa mengubah ini sesuai dengan keinginan kita.

- **Konfigurasi Provider**

```
provider "libvirt" {
  uri = "qemu:///system"
}
```

Provider libvirt berfungsi untuk melakukan koneksi ke hypervisor berbasis kvm dengan uri “qemu://system

- **Disk Cloud-init**

```
resource "libvirt_cloudinit_disk" "db_cloud_init" {
  name      = "db_cloudinit.iso"
  user_data = data.template_file.db_user_data.rendered
  network_config = data.template_file.db_network-config.rendered
}

resource "libvirt_cloudinit_disk" "web_cloud_init" {
  name      = "web_cloudinit.iso"
  user_data = data.template_file.web_user_data.rendered
  network_config = data.template_file.web_network-config.rendered
}
```

Kode ini akan membuat sebuah file iso cloud init yang digunakan untuk melakukan konfigurasi user data , dan network konfigurasi seperti IP address , MAC address dan lain lain

- **Volume Disk VM**

```
resource "libvirt_volume" "db_vm_disk" {
  name   = "db_disk_vm.qcow2"
  pool   = "default"
  source = "./image/base-image.qcow2" # Path to your base image
  format = "qcow2"
}

resource "libvirt_volume" "web_vm_disk" {
  name   = "web_disk_vm.qcow2"
  pool   = "default"
  source = "./image/base-image.qcow2" # Path to your base image
  format = "qcow2"
}
```

Didalam README.md repository github salah satu prerequisite adalah mendownload ubuntu image dan mengconvertnya ke dalam format qcow2. Setelah semua itu dilakukan baru dapat menjalankan program ini. Libvirt volume disini berfungsi untuk membuat dan mengelola disk virtual untuk VM yang nanti digunakan untuk penyimpanan utama dan tambahan untuk VM. Sesuaikan konfigurasinya dengan yang anda gunakan.

- **Domain VM**

```
resource "libvirt_domain" "db_vm" {
  name   = var.db_vm_name
  memory = var.db_vm_memory
  vcpu   = var.db_vm_vcpus

  cloudinit = libvirt_cloudinit_disk.db_cloud_init.id

  disk {
    volume_id = libvirt_volume.db_vm_disk.id
  }

  network_interface {
    network_name = "custom-net"
    wait_for_lease = true
  }

  console {
    type = "pty"
    target_type = "serial"
    target_port = "0"
  }
}

resource "libvirt_domain" "web_vm" {
  name   = var.web_vm_name
  memory = var.web_vm_memory
  vcpu   = var.web_vm_vcpus # Number of CPUs

  cloudinit = libvirt_cloudinit_disk.web_cloud_init.id

  disk {
    volume_id = libvirt_volume.web_vm_disk.id
  }

  network_interface {
    network_name = "custom-net"
    wait_for_lease = true
  }

  console {
    type = "pty"
    target_type = "serial"
    target_port = "0"
  }
} You, 2024-11-25 01:15:51 - init tofu
```

Libvirt domain disini berfungsi untuk mengelola domain vm di hypervisor yang berbasis KVM. Domain itu adalah unit eksekusi untuk membuat vm, Termasuk melakukan setting , nama vm , memory , vcpu , konfigurasi cloud-int , volume dan network interface serta console dari vm. Disini kita membuat dua vm yaitu dbvm0 dan webvm0

b. Konfigurasi Network

```
<network>
  <name>custom-net</name>
  <forward mode="nat"/>
  <ip address="192.168.158.1" netmask="255.255.255.0">
    <dhcp>
      <range start="192.168.158.10" end="192.168.158.200"/>
    </dhcp>
  </ip>
</network>
```

Berikut ini adalah konfigurasi custom network yang kita gunakan. Dimana IP yang akan diberikan kepada vm ada pada range seperti diatas. Kita bisa menjalankan network ini menggunakan ***virsh*** . virsh adalah program cli untuk memanajemen vm.

c. VM Network Konfigurasi

Berikut ini adalah network configuration yang akan digunakan VM.

- **Network Config**

```
version: 2 You, 2024-11-25 01:15:51 - init tofu
ethernets:
  ens3:
    dhcp4: true
```

Menggunakan dhcp sama seperti network yang digunakan dan ip yang di dapat juga akan berada di range tersebut

- **User Config**

```
#cloud-config
ssh_pwauth: true
chpasswd:
  list: |
    root:password
  expire: false
users:
  - name: ubuntu
    ssh_authorized_keys:
      - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC2zDKRBgNuHU1lAR0SqW3kUpOWjC3X3FDD9OoLIA1fSaQ8aXMsLYC7VtD1MNczV+IFSSrxHSLcdhk66FpLWwPUrFni3+qjfq9E8C4QzKsIOhodXVl+JVY8i4oRdRQPkcNVMQ78U4p6zTnK7VK6zShUnCK8CCiYxX73iS4MWm6YxGtbaaWUpycMva/SLBhK3Twb8rdIOGbJzG6Q+Dixj7OpbZkuZHbgib6aTUxri/wypNxXReirGNR8Zr/9yFQYdJavOJeC5ja8QnTtDMI7cUCUgR8N6GiLb2H+at7H1tmaqqQ7XY/Na4fMCOZSAMww0+0F9u+hPO5EpQmmXNyE9z2YC99jq6xTHPazgbnOQty7cLW75eJT/4LWAGkZf3Y5Wu8cWD3uld2JPzCOtDLhJfq/L1UVHpt0luT1YLwvPU2fXXkFpf7rG1aSg3SjsxEfMtDKf5GrglW/dJbb07Af3Er7BipvdZXb9mD4Hnce7CDdEdipyreOKkxlxi0h0Jzllxc= unedotamps@mylaptop
    sudo: ALL=(ALL) NOPASSWD:ALL
    groups: sudo
    shell: /bin/bash
    lock_passwd: false
```

Untuk dapat mengakses vm dapat menggunakan dua cara yaitu melalui console dan ssh. Karena kita menggunakan ansible untuk mendeploy aplikasi maka dari itu ssh lah nanti yang akan digunakan. Jangan lupa ganti ssh key sesuai dengan env anda

d. Environment Variable

```
export TF_VAR_db_vm_name="db_vm0"
export TF_VAR_db_vm_memory=2048
export TF_VAR_db_vm_vcpus=1
export TF_VAR_web_vm_name="web_vm0"
export TF_VAR_web_vm_memory=2048 You, 2024-11-30 19:47:54 - add docs
export TF_VAR_web_vm_vcpus=1
```

Berikut ini adalah environment variable yang akan kita gunakan di terraform

e. Script

Terdapat tiga script yang akan digunakan agar dapat dengan cepat menjalankan program ini

- Startup Script

```
#!/bin/bash You, 2024-11-30 19:47:54 - add docs

tofu init
sudo virsh net-define custom-network.xml
sudo virsh net-start custom-net
sudo virsh net-autostart custom-net
```

script ini hanya dijalankan sekali saja saat pertama kali menjalankan program. karena ini hanya untuk mendefine network dari vm dan init tofu

- Up Script

```
#!/usr/bin/bash You, 2024-11-30 19:47:54 - add docs

source '.vm-settings'     ▪ Not following: .vm-settings was not specif

tofu apply
```

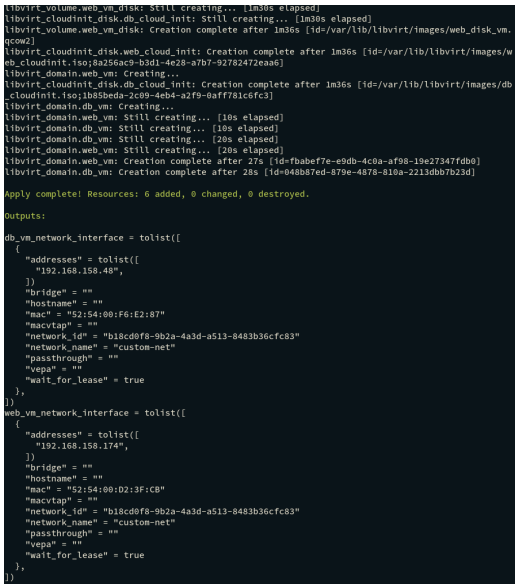
Script ini hanya berfungsi untuk melakukan source dari environment sehingga ketika meng apply tofu dapat membaca environmentnya

- Down Script

```
#!/bin/bash You, 2024-11-30 19:47:54 - add docs

source '.vm-settings'     ▪ Not following: .vm-settings was not specif
tofu destroy
```

Sama seperti Up script cuman disini akan men destroy vm dan membalikan tofu ke state semua

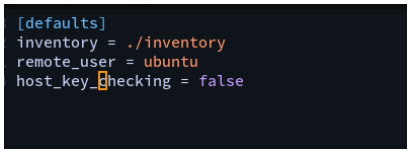
Setelah semua config selesai maka jalankan scriptnya dan hasilnya:



2. Ansible

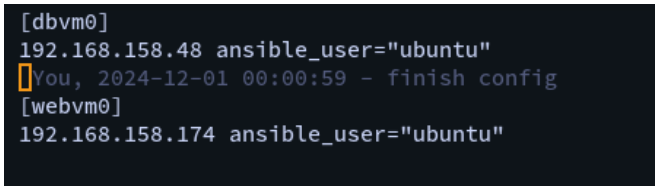
Seperti yang sudah dijelaskan sebelumnya ansible disini akan digunakan untuk otomatisasi instalasi dependensi, dan deploy aplikasi yang akan digunakan di VM. Akan terdapat dua VM yaitu ***dbvm0*** dan ***webvm0*** **.** Berikut ini adalah penjelasan dari kode - kode yang digunakan

1. Ansible Config



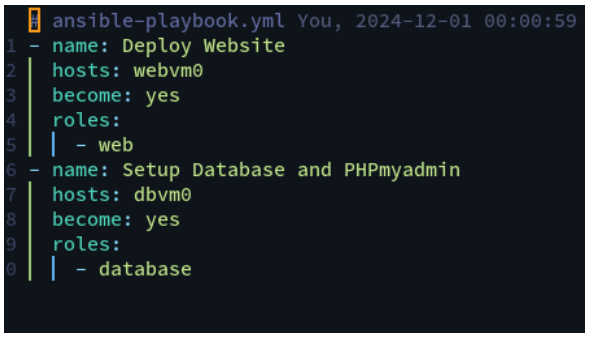
Disini kita mendefinisikan konfigurasi inventory serta nama remote user. Host key checking kita setting false karena ini hanya dijalankan di local saja

2. Inventory



Berisi kumpulan host - host yang akan dikelola oleh ansible. Disini ada dua yaitu **dbvm0** dan **webvm0** . Jika kita punya banyak vm untuk satu host kita bisa langsung menambahkannya disini

3. Playbook

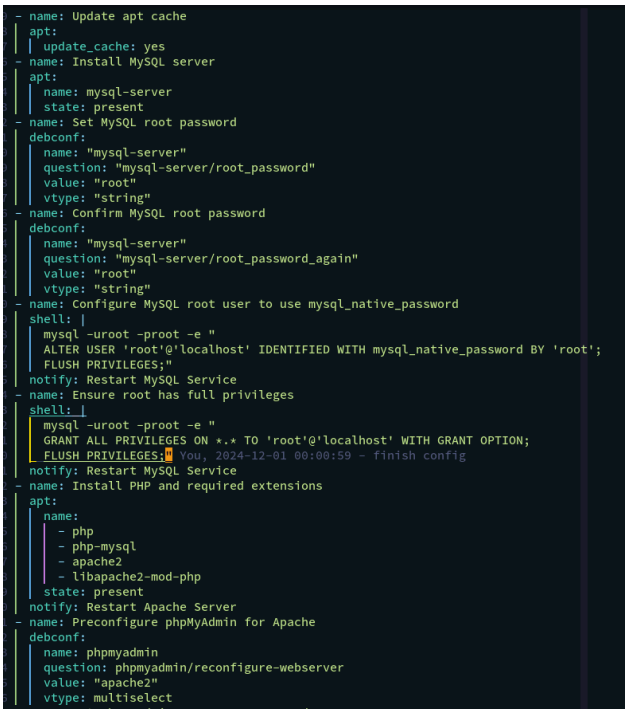


Akan ada dua play yang akan dijalankan, yang pertama adalah deploy website dan setup database phpmyadmin. hosts sesuaikan dengan yang ada di inventory, became yes ini agar ansible mendapat sudo privilege. roles adalah folder yang berisi task, handler , variabel yang akan dijalankan pada play tersebut. Berikut nya adalah penjelasan dari masing masing Role

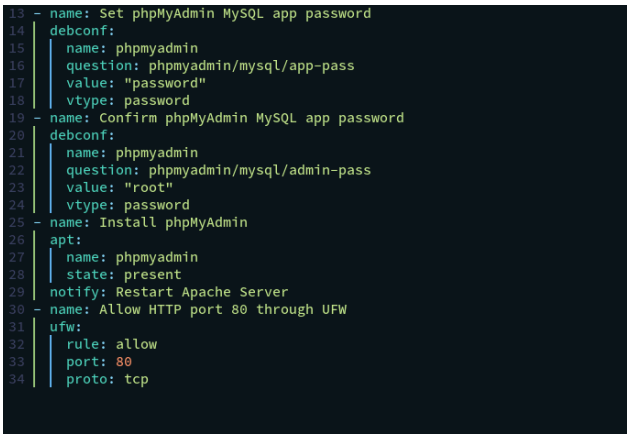
4. Roles
    a. Database

Akan terdapat 2 program yang akan dijalankan yaitu task dan handler. Task adalah program yang akan dijalankan pertama kali dan handler akan dijalankan ketika di notify oleh di task

- Task



```yaml
- name: Update apt cache
  apt:
    update_cache: yes
- name: Install MySQL server
  apt:
    name: mysql-server
    state: present
- name: Set MySQL root password
  debconf:
    name: "mysql-server"
    question: "mysql-server/root_password"
    value: "root"
    vtype: "string"
- name: Confirm MySQL root password
  debconf:
    name: "mysql-server"
    question: "mysql-server/root_password_again"
    value: "root"
    vtype: "string"
- name: Configure MySQL root user to use mysql_native_password
  shell: |
    mysql -uroot -proot -e "
    ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'root';
    FLUSH PRIVILEGES;"
  notify: Restart MySQL Service
- name: Ensure root has full privileges
  shell: |
    mysql -uroot -proot -e "
    GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' WITH GRANT OPTION;
    FLUSH PRIVILEGES;" You, 2024-12-01 00:00:59 - finish config
  notify: Restart MySQL Service
- name: Install PHP and required extensions
  apt:
    name:
      - php
      - php-mysql
      - apache2
      - libapache2-mod-php
    state: present
  notify: Restart Apache Server
- name: Preconfigure phpMyAdmin for Apache
  debconf:
    name: phpmyadmin
    question: phpmyadmin/reconfigure-webserver
    value: "apache2"
    vtype: multiselect
```



```yaml
- name: Set phpMyAdmin MySQL app password
  debconf:
    name: phpmyadmin
    question: phpmyadmin/mysql/app-pass
    value: "password"
    vtype: password
- name: Confirm phpMyAdmin MySQL app password
  debconf:
    name: phpmyadmin
    question: phpmyadmin/mysql/admin-pass
    value: "root"
    vtype: password
- name: Install phpMyAdmin
  apt:
    name: phpmyadmin
    state: present
  notify: Restart Apache Server
- name: Allow HTTP port 80 through UFW
  ufw:
    rule: allow
    port: 80
    proto: tcp
```

Berikut ini adalah task - task yang dijalankan dari gambar diatas

1. Melakukan apt-update: ini biasa dilakukan saat ingin melakukan instalasi dari program
2. Melakukan install mysql server dan selanjutnya adalah settings password , konfirmasi password untuk database
3. Selanjutnya adalah melakukan setting supaya mysql dapat menggunakan native password agar kita dapat login ke dalam phpmyadmin nantinya

4. Selanjutnya adalah melakukan instalasi PHP, php mysql, apache dan phpmyadmin dan set app password phpmyadmin
5. Setelah semua dijalankan kita juga harus mengizinkan port 80 untuk di akses.

- Handler

```yaml
- name: Restart MySQL Service
  service:
    name: mysql
    state: restarted
- name: Restart Apache Server
  service:
    name: apache2
    state: restarted
```

Handler ini hanya akan dijalankan jika di notify di task. Bisa kita lihat program ini di notify untuk restart apache dan mysql

b. Website

```yaml
- name: Update Apt Cache You, 2024-12-01 00:00:59 - fir
  apt:
    update_cache: yes
- name: Install Apache2
  apt:
    name: apache2
    state: present
- name: Apache is running and enabled
  service:
    name: apache2
    state: started
    enabled: yes
- name: Install Git
  apt:
    name: git
    state: present
- name: Remove /var/www/html and all contents
  command: rm -rf /var/www/html/
- name: Clone web-sample-6 repository
  git:
    repo: https://github.com/rm77/web-sample-6.git
    dest: /var/www/html
    update: yes
- name: permissions for /var/www/html
  file:
    path: /var/www/html
    state: directory
    owner: www-data
    group: www-data
    mode: '0755'
    recurse: yes
- name: Allow HTTP traffic on port 80 (UFW)
  ufw:
    rule: allow
    port: 80
    proto: tcp
```

Berikut ini adalah task - task yang akan dijalankan pada VM website

1. Melakukan instalasi apache dan start apache
2. Install GIT yang akan kita gunakan untuk meng clone website
3. Kemudian kita menghapus file yang ada di /var/www/html karena kita ingin menempatkan file kita disini file kita disini
4. Setelah itu clone repo ke /var/www/html dan update permission nya
5. Jangan lupa untuk update dengan mengizinkan port 80

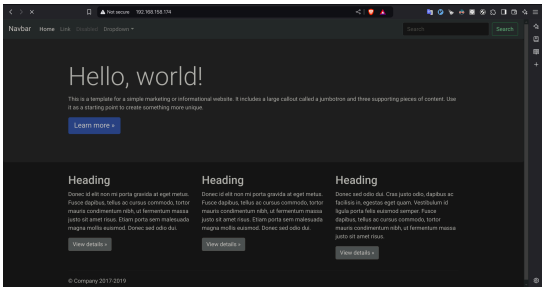
Setelah semua konfigurasi lengkap kita tinggal menjalankan command **ansible-playbook playbook.yml .** Berikut ini adalah hasil runnya

```
TASK [database : Confirm MySQL root password] ***********************************************************
changed: [192.168.158.48]

TASK [database : Configure MySQL root user to use mysql_native_password] ********************************
changed: [192.168.158.48]

TASK [database : Ensure root has full privileges] ********************************************************
changed: [192.168.158.48]

TASK [database : Install PHP and required extensions] ****************************************************
changed: [192.168.158.48]

TASK [database : Preconfigure phpMyAdmin for Apache] *****************************************************
changed: [192.168.158.48]

TASK [database : Set phpMyAdmin MySQL app password] ******************************************************
changed: [192.168.158.48]

TASK [database : Confirm phpMyAdmin MySQL app password] **************************************************
changed: [192.168.158.48]

TASK [database : Install phpMyAdmin] *********************************************************************
changed: [192.168.158.48]

TASK [database : Allow HTTP port 80 through UFW] *********************************************************
changed: [192.168.158.48]

RUNNING HANDLER [database : Restart MySQL Service] *******************************************************
[WARNING]: Skipping plugin (/usr/lib/python3/dist-packages/ansible/plugins/filter/core.py) as it seems to
be invalid: cannot import name 'environmentfilter' from 'jinja2.filters'
(/home/unedotamps/.local/lib/python3.10/site-packages/jinja2/filters.py)
[WARNING]: Skipping plugin (/usr/lib/python3/dist-packages/ansible/plugins/filter/mathstuff.py) as it
seems to be invalid: cannot import name 'environmentfilter' from 'jinja2.filters'
(/home/unedotamps/.local/lib/python3.10/site-packages/jinja2/filters.py)
changed: [192.168.158.48]

RUNNING HANDLER [database : Restart Apache Server] *******************************************************
[WARNING]: Skipping plugin (/usr/lib/python3/dist-packages/ansible/plugins/filter/core.py) as it seems to
be invalid: cannot import name 'environmentfilter' from 'jinja2.filters'
(/home/unedotamps/.local/lib/python3.10/site-packages/jinja2/filters.py)
[WARNING]: Skipping plugin (/usr/lib/python3/dist-packages/ansible/plugins/filter/mathstuff.py) as it
seems to be invalid: cannot import name 'environmentfilter' from 'jinja2.filters'
(/home/unedotamps/.local/lib/python3.10/site-packages/jinja2/filters.py)
changed: [192.168.158.48]

PLAY RECAP ***********************************************************************************************
192.168.158.174            : ok=9    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored
0
192.168.158.48             : ok=15   changed=14   unreachable=0    failed=0    skipped=0    rescued=0    ignored
0
```

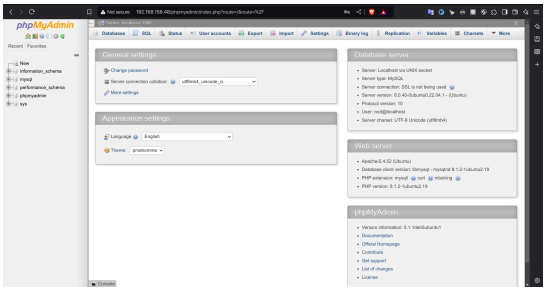
Kesimpulan

Kita telah berhasil melakukan konfigurasi opentofu dan ansible berikut ini adalah

Web



Phpmyadmin



Kedua VM telah berhasil menjalankan web pada webvm0 dan phpmyadmin pada dbvm0. Opentofu pada percobaan ini menggunakan libvirt untuk membuat dan provision vm berbasis kvm dan ansible untuk melakukan update dan deploy aplikasi secara otomatis. Dengan menggunakan OpenTofu dan Ansible, kita dapat dengan mudah membuat, mengkonfigurasi, dan mengelola VM, sehingga meningkatkan produktivitas dan efisiensi dalam pengembangan dan pengelolaan aplikasi. Arsitektur ini cocok digunakan untuk testing di local, jika sudah masuk produksi sebaiknya menggunakan provider cloud computing yang sudah banyak tersedia