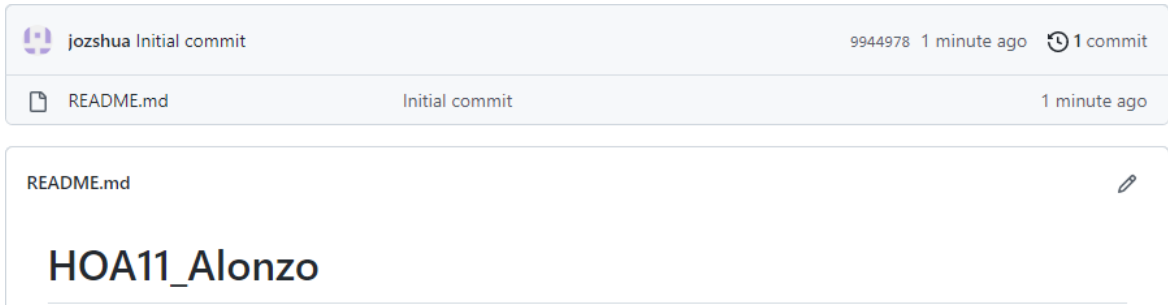


Name: Jozshua Amiel Alonzo	Date Performed: November 21, 2022
Course/Section: CPE31S2	Date Submitted:
Instructor: Dr. Jonathan Taylar	Semester and SY: 2022-2023
Activity 11: Containerization	
1. Objectives	
Create a Dockerfile and form a workflow using Ansible as Infrastructure as Code (IaC) to enable Continuous Delivery process	
2. Discussion	
<p>Github Link: https://github.com/jozshua/HOA11_Alonzo.git</p> <p>Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.</p> <p>Source: https://docs.docker.com/get-started/overview/</p> <p>You may also check the difference between containers and virtual machines. Click the link given below.</p> <p>Source: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm</p>	
3. Tasks	
<ol style="list-style-type: none"> 1. Create a new repository for this activity. 2. Install Docker and enable the docker socket. 3. Add to Docker group to your current user. 4. Create a Dockerfile to install web and DB server. 5. Install and build the Dockerfile using Ansible. 6. Add, commit and push it to your repository. 	
4. Output (screenshots and explanations)	
	

Created the new repository for this activity.

```
jozshua@workstation-VirtualBox:~$ systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor prese
   Active: active (running) since Sat 2022-11-19 17:38:12 PST; 1 day 22h ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 6723 (dockerd)
      Tasks: 9
     Memory: 23.9M
        CPU: 5.691s
    CGroup: /system.slice/docker.service
            └─6723 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

jozshua@workstation-VirtualBox:~$ sudo systemctl enable docker
[sudo] password for jozshua:
jozshua@workstation-VirtualBox:~$ sudo systemctl start docker
```

Already installed the docker from using the `sudo apt install docker.io`. Then use the following commands to enable the docker socket.

```
- name: add docker group to user current user
  command: sudo usermod -a -G docker jozshua
```

Adding the docker group to the current user by putting these commands inside the file of the installing and building the docker.

```
jozshua@server2-VirtualBox:~$ groups jozshua
jozshua : jozshua adm cdrom sudo dip plugdev lpadmin lxd sambashare docker
jozshua@server2-VirtualBox:~$
```

For checking if the group was already added type the `groups <username>` command. So as you can see, the docker group was already in the list when we issued the command.

```
GNU nano 6.2                                dockerfile
FROM ubuntu
MAINTAINER jozshua<qjaaalonzo@tip.edu.ph>

# skip interactions
ARG DEBIAN_FRONTEND=noneinteractive

# update packages
RUN apt update; apt dist-upgrade -y

# install packages
RUN apt install -y apache2 mariadb-server

# set entrypoint
ENTRYPOINT apache2ctl -D FOREGROUND
```

The creation of dockerfile with the installation of the web and mariadb servers.

```
jozshua@workstation-VirtualBox: ~/HOA11_Alonzo/cpe_HO...
GNU nano 6.2                                site.yml
--
- hosts: all
  become: true

  tasks:
    - name: Install aptitude
      apt:
        name: aptitude
        state: latest
        update_cache: true

    - name: Install required system packages
      apt:
        pkg:
          - apt-transport-https
          - ca-certificates
          - curl
          - software-properties-common
          - python3-pip
          - virtualenv
          - python3-setuptools
        state: latest
        update_cache: true
```

```
- name: Add Docker GPG apt Key
  apt_key:
    url: https://download.docker.com/linux/ubuntu/gpg
    state: present

- name: Add Docker Repository
  apt_repository:
    repo: deb https://download.docker.com/linux/ubuntu focal stable
    state: present

- name: Update apt and install docker-ce
  apt:
    name: docker-ce
    state: latest
    update_cache: true

- name: Install Docker Module for Python
  pip:
    name: docker
```

```
- name: create build directory
  file:
    path: /root/demo-dockerfile
    state: directory
    owner: root
    group: root
    mode: '0755'

- name: copy Dockerfile
  copy:
    src: ./dockerfile
    dest: /root/demo-dockerfile/dockerfile
    owner: root
    group: root
    mode: '0644'

- name: build container image
  docker_image:
    name: democontainer:v1.0
    build:
      path: /root/demo-dockerfile
    source: build
    state: present
```

Using these codes for installing and building the dockerfile. Then run the ansible command to execute the playbook for the docker in Ubuntu remote server.


```

PLAY [all] *****
*

TASK [Gathering Facts] *****
*
ok: [192.168.56.102]

TASK [Install aptitude] *****
*
ok: [192.168.56.102]

TASK [Install required system packages] *****
*
ok: [192.168.56.102]

TASK [Add Docker GPG apt Key] *****
*
ok: [192.168.56.102]

TASK [Add Docker Repository] *****
*
ok: [192.168.56.102]

TASK [Update apt and install docker-ce] *****
*
ok: [192.168.56.102]

TASK [Install Docker Module for Python] *****
*
ok: [192.168.56.102]

TASK [create build directory] *****
*
ok: [192.168.56.102]

TASK [copy Dockerfile] *****
*
ok: [192.168.56.102]

TASK [build container image] *****
*
changed: [192.168.56.102]

TASK [add docker group to user current user] *****
*
changed: [192.168.56.102]

PLAY RECAP *****
*
192.168.56.102      : ok=11   changed=2    unreachable=0    failed=0
skipped=0      rescued=0    ignored=0

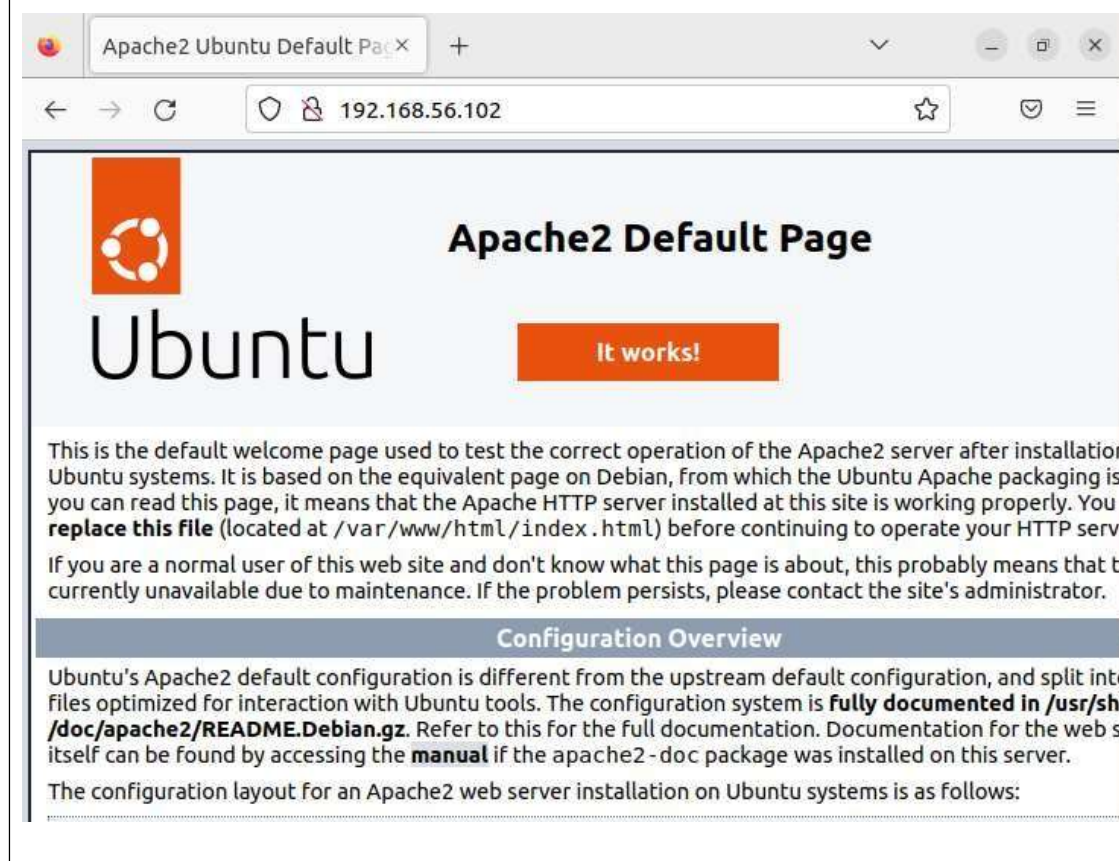
```

After using the ansible command there are eleven tasks that are successfully executed and 2 changed state.



The codes that we used for installing and building the docker is now committed and pushed to the github repository.

Proof of installation:



```

jozshua@server2-VirtualBox:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor prese>
   Active: active (running) since Tue 2022-11-22 21:18:43 PST; 23min ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 1131 (dockerd)
      Tasks: 7
     Memory: 18.2M
        CPU: 4.134s
    CGroup: /system.slice/docker.service
            └─1131 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Nov 22 21:18:34 server2-VirtualBox dockerd[1131]: time="2022-11-22T21:18:34.52>
Nov 22 21:18:34 server2-VirtualBox dockerd[1131]: time="2022-11-22T21:18:34.52>
Nov 22 21:18:37 server2-VirtualBox dockerd[1131]: time="2022-11-22T21:18:37.03>
Nov 22 21:18:38 server2-VirtualBox dockerd[1131]: time="2022-11-22T21:18:38.90>
Nov 22 21:18:40 server2-VirtualBox dockerd[1131]: time="2022-11-22T21:18:40.97>

```

```

jozshua@Server1-VirtualBox:~$ systemctl status mariadb
● mariadb.service - MariaDB 10.6.7 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor pres>
   Active: active (running) since Thu 2022-10-06 10:59:01 PST; 4min 13s ago
     Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/
   Process: 51852 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /v>
   Process: 51853 ExecStartPre=/bin/sh -c systemctl unset-environment _WSREP_>
   Process: 51855 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] &>
   Process: 51895 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP>
   Process: 51897 ExecStartPost=/etc/mysql/debian-start (code=exited, status=>
  Main PID: 51884 (mariabdb)
    Status: "Taking your SQL requests now..."
     Tasks: 8 (limit: 1080)
    Memory: 59.3M
        CPU: 490ms
    CGroup: /system.slice/mariadb.service
            └─51884 /usr/sbin/mariabdb

```

These are the proof that the installation of the docker, mariadb and web servers are successfully executed to the remote server of the Ubuntu.

Reflections:

Answer the following:

1. What are the benefits of implementing containerizations?

Containerization enables programmers to utilize resources more effectively. Virtually all of the computing resources that are available can be configured to use containers, and they can function with essentially no overhead.

Conclusions:

In this activity I used ansible.cfg, inventory from the github and I used the site.yml file for installing and building the docker. I also created the dockerfile for executing the installation of the web and mariadb servers from using the ansible command. As the

ansible command was executed the playbook were having a total of eleven successfully executed tasks and 2 changed state for the Ubuntu remote server. Above all I also installed the latest version of the docker from the local machine and enable its socket. I also add the docker group to the current user, and I was able to display it that it was already added. So after installation I was able to check if it is already installed using the ip address on the web browser and through the terminal. Lastly the codes were pushed and committed to the github with the new created repository for this activity. Therefore