Name: Agpaoa, Ma.Diane J.	**Date Performed:** 17/11/2022
Course/Section: CPE232-CPE31S22	**Date Submitted:** 19/11/2022
Instructor: Dr. Jonathan V. Taylar	**Semester and SY:** 1st sem 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

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.

Source: https://docs.docker.com/get-started/overview/

You may also check the difference between containers and virtual machines. Click the link given below.

Source: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm

3. Tasks

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)

Tasks

1. Create a new repository for this activity.

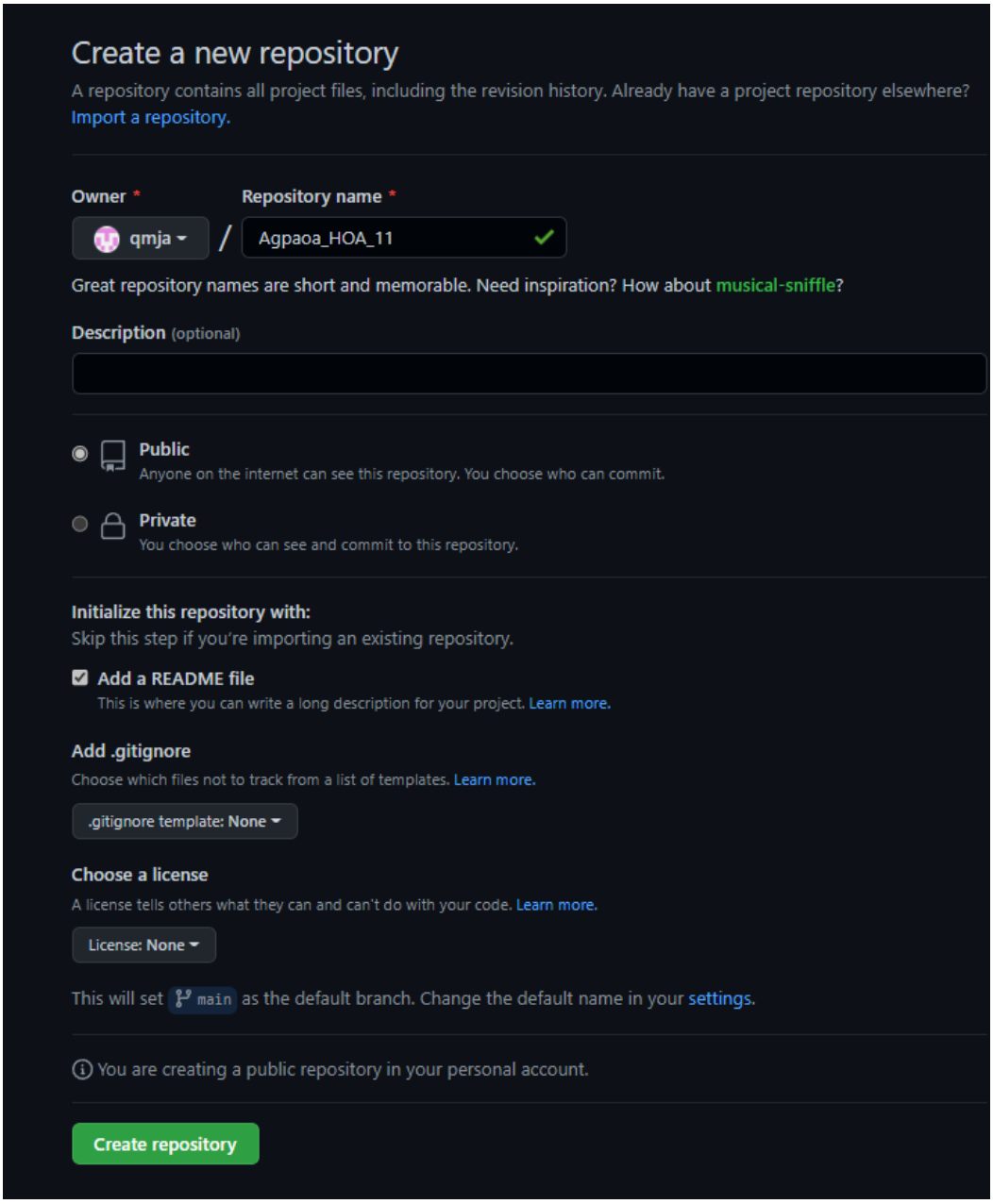

Figure 1.1 Creating new repository for the activity

I created a new repository for this activity and named it Agpaoa_HOA_11.

2. Install Docker and enable the docker socket.

```
madiane@workstation:~$ sudo apt install docker.io
[sudo] password for madiane:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools btrfs-progs cgroupfs-mount | cgroup-lite debootstrap
  docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd docker.io pigz runc ubuntu-fan
0 upgraded, 6 newly installed, 0 to remove and 3 not upgraded.
Need to get 65.3 MB of archives.
After this operation, 282 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1
 [63.6 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1
.7-1ubuntu3 [34.4 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 runc amd64 1.1.0-0ub
untu1 [4,087 kB]
Get:4 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 containerd amd64 1.5
.9-0ubuntu3 [27.0 MB]
Get:5 http://ph.archive.ubuntu.com/ubuntu jammy/universe amd64 docker.io amd64
20.10.12-0ubuntu4 [34.0 MB]
Get:6 http://ph.archive.ubuntu.com/ubuntu jammy/universe amd64 ubuntu-fan all 0
.12.16 [35.2 kB]
Fetched 65.3 MB in 4min 18s (253 kB/s)
```

Figure 2.1 Installing Docker

I installed Docker on the workstation server to be able to do the tasks that will follow.

```
madiane@workstation:~$ systemctl enable docker
madiane@workstation:~$ systemctl start docker
```

Figure 2.2 Enabling the docker socket

I enable the docker socket by executing the command “systemctl enable docker” and “systemctl start docker”.

```
madiane@workstation:~$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor prese>
     Active: active (running) since Wed 2022-11-16 18:44:23 PST; 7min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2490 (dockerd)
      Tasks: 7
     Memory: 96.3M
        CPU: 462ms
     CGroup: /system.slice/docker.service
             └─2490 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.086967051>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.087076499>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.087143345>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.200232242>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.371821452>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.714614238>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.828486335>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.828792074>
Nov 16 18:44:23 workstation systemd[1]: Started Docker Application Container E>
Nov 16 18:44:23 workstation dockerd[2490]: time="2022-11-16T18:44:23.906121211>
lines 1-22/22 (END)
```

Figure 2.3 Verifying the status of Docker service

I verified the status of Docker's service by executing the command "systemctl status docker".

3. Add to Docker group to your current user.

```
madiane@workstation:~$ grep docker /etc/group
docker:x:137:
```

Figure 3.1 Checking if the docker group belongs to a user

By using the command grep docker /etc/group, I checked if the docker group already belonged to a user or if it already belonged to my current user. Based on the result, the docker group does not belong to any particular user.

Group

```
madiane@workstation:~$ sudo usermod -aG docker ${USER}
[sudo] password for madiane:
```

Figure 3.2 Adding the docker group to my current user

From the previous command, it's verified that the docker group does not belong to the current user or any user in particular. In order to add the docker group to my current user, I used the command sudo usermod -aG docker ${USER}.

```
madiane@workstation:~$ grep docker /etc/group
docker:x:137:madiane
```

Figure 3.3 Verifying that the docker group belongs my current user

In order to verify that the docker group belongs to my current user, I used the command “grep docker /etc/group”.

```
madiane@workstation:~$ su - ${USER}
Password:
```

```
madiane@workstation:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Figure 3.4 Executing a command of docker without using a sudo command

As a result of adding the docker group to my current user, I didn’t need to use the command sudo every time I used docker.

4. Create a Dockerfile to install web and DB server.

```
madiane@workstation:~/Agpaoa_HOA_11$ mkdir Docker_file
madiane@workstation:~/Agpaoa_HOA_11$ cd Docker_file
madiane@workstation:~/Agpaoa_HOA_11/Docker_file$ nano Dockerfile
```

Figure 4.1 Creating directory for Dockerfile

I created a new directory for Dockerfile and named it Docker_file by executing the command “mkdir Docker_file”. Lastly, I created the Dockerfile by using nano.

```
madiane@workstation: ~/Agpaoa_HOA_11/Docker_file
GNU nano 6.2                                    Dockerfile
FROM ubuntu
MAINTAINER madiane <qmjagpaoa@tip.edu.ph>

# Skip prompts
ARG DEBIAN_FRONTEND=noninteractive

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

# Install packages
RUN apt install -y apache2 vim-nox

# Set entrypoint
ENTRYPOINT apache2ctl -D FOREGROUND
```

Figure 4.2 Contents of Dockerfile

The contents of Dockerfile consists of codes that will skip prompts, update packages, install apache2 and vim-nox and setting an entrypoint in ubuntu.

5. Install and build the Dockerfile using Ansible.

```
madiane@workstation:~/Agpaoa_HOA_11$ ls
ansible.cfg  Docker_file  insbuildockf.yml  inventory  README.md  roles
```

Figure 5.1 Contents of Agpaoa_HOA_11 directory

The directory Agpaoa_HOA_11 consists of ansible.cfg, inventory, Dockerfile directory, .yml file and roles directory in order to use ansible. The process of creating the roles and the contents of insbuildockf.yml will be seen afterwards.

```
madiane@workstation: ~/Agpaoa_HOA_11
GNU nano 6.2                                    inventory

[web_servers]
192.168.56.105

[db_servers]
192.168.56.106
```

Figure 5.2 Contents of inventory

The inventory consists of the remote servers and it was grouped between web_servers and db_servers.

```
madiane@workstation: ~/Agpaoa_HOA_11
GNU nano 6.2                                insbuildockf.yml
---
- hosts: all
  become: true
  pre_tasks:

  - name: install updates (Ubuntu)
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

- hosts: db_servers
  become: true
  roles:
    - db_server

- hosts: web_servers
  become: true
  roles:
    - web_server
```

Figure 5.3 Contents of insbuildockf.yml

Inside the playbook of insbuildockf.yml, it consists of pre-tasks for all the remote servers and plays for db_servers and web_servers.

```
madiane@workstation:~/Agpaoa_HOA_11$ mkdir roles
madiane@workstation:~/Agpaoa_HOA_11$ cd roles
madiane@workstation:~/Agpaoa_HOA_11/roles$ mkdir web_server db_server
madiane@workstation:~/Agpaoa_HOA_11/roles$ ls
db_server  web_server
madiane@workstation:~/Agpaoa_HOA_11/roles$ cd web_server
madiane@workstation:~/Agpaoa_HOA_11/roles/web_server$ mkdir tasks
madiane@workstation:~/Agpaoa_HOA_11/roles/web_server$ cd tasks
madiane@workstation:~/Agpaoa_HOA_11/roles/web_server/tasks$ sudo nano main.yml
madiane@workstation:~/Agpaoa_HOA_11/roles/web_server/tasks$ cd ../..
madiane@workstation:~/Agpaoa_HOA_11/roles$ cd db_server
madiane@workstation:~/Agpaoa_HOA_11/roles/db_server$ mkdir tasks
madiane@workstation:~/Agpaoa_HOA_11/roles/db_server$ cd tasks
madiane@workstation:~/Agpaoa_HOA_11/roles/db_server/tasks$ sudo nano main.yml
```

Figure 5.4 Creating roles for web_server and db_server

I created a directory for roles, then within the roles directory I created new directories and named them as web_server and db_server. These directories will contain the tasks directory and within the tasks directory it will contain the playbook main.yml.

```
madiane@workstation: ~/Agpaoa_HOA_11/roles/db_server/tasks
GNU nano 6.2                                   main.yml
- name: intstall docker
  tags: docker
  apt:
    name: docker.io

- name: Adding Docker group to the current user
  shell: |
    usermod -aG docker madiane

- name: Getting the docker image of Ubuntu
  shell: |
    docker run ubuntu

- name: Copying Docker_file in servers
  tags: copy_dockerfile
  copy:
    src: ~/Agpaoa_HOA_11/Docker_file
    dest: /home/madiane
    owner: root
    group: root
    mode: 0644

- name: Building Dockerfile
  shell: |
    cd Docker_file
    docker build -t cpe232:1.2 .

- name: Checking Docker Images
  shell: 'docker images'
  register: command_output

- debug:
        var: command_output.stdout_lines
```

```
madiane@workstation: ~/Agpaoa_HOA_11/roles/web_server/tasks
GNU nano 6.2                                   main.yml
- name: intstall docker
  tags: docker
  apt:
    name: docker.io

- name: Adding Docker group to the current user
  shell: |
    usermod -aG docker madiane

- name: Getting the docker image of Ubuntu
  shell: |
    docker run ubuntu

- name: Copying Docker_file in servers
  tags: copy_dockerfile
  copy:
    src: ~/Agpaoa_HOA_11/Docker_file
    dest: /home/madiane
    owner: root
    group: root
    mode: 0644

- name: Building Dockerfile
  shell: |
    cd Docker_file
    docker build -t cpe232:1.2 .

- name: Checking Docker Images
  shell: 'docker images'
  register: command_output

- debug:
        var: command_output.stdout_lines
```

Figure 5.5 Contents of main.yml from db_server and web_server directory

Inside the playbook are the flow of tasks that will install and build the Dockerfile in the db_servers and web_servers. To specify, the flow of task accordingly is to install the docker, adding the docker group to the current user of remote servers, getting the docker image of ubuntu, copying the Docker_file directory to web_servers and db_servers, building the Dockerfile and checking the docker images to confirm that the installation and building of Dockerfile is successful.

```
madiane@workstation:~/Agpaoa_HOA_11$ ansible-playbook --ask-become-pass insbuildockf.yml
BECOME password:

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.106]
ok: [192.168.56.105]

TASK [install updates (Ubuntu)] ************************************************************************************
ok: [192.168.56.106]
ok: [192.168.56.105]

PLAY [db_servers] **************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.106]

TASK [db_server : intstall docker] *********************************************************************************
ok: [192.168.56.106]

TASK [db_server : Adding Docker group to the current user] *********************************************************
changed: [192.168.56.106]

TASK [db_server : Getting the docker image of Ubuntu] **************************************************************
changed: [192.168.56.106]

TASK [db_server : Copying Docker_file in servers] ******************************************************************
ok: [192.168.56.106]

TASK [db_server : Building Dockerfile] *****************************************************************************
changed: [192.168.56.106]

TASK [db_server : Checking Docker Images] **************************************************************************
changed: [192.168.56.106]

TASK [db_server : debug] *******************************************************************************************
ok: [192.168.56.106] => {
    "command_output.stdout_lines": [
        "REPOSITORY   TAG       IMAGE ID       CREATED             SIZE",
        "cpe232       1.2       84fcad3ed2ab   About an hour ago   327MB",
        "ubuntu       latest    a8780b506fa4   2 weeks ago         77.8MB"
    ]
}

PLAY [web_servers] *************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.105]

TASK [web_server : intstall docker] ********************************************************************************
ok: [192.168.56.105]

TASK [web_server : Adding Docker group to the current user] ********************************************************
changed: [192.168.56.105]

TASK [web_server : Getting the docker image of Ubuntu] *************************************************************
changed: [192.168.56.105]

TASK [web_server : Copying Docker_file in servers] *****************************************************************
```

```
changed: [192.168.56.105]

TASK [web_server : Copying Docker_file in servers] ***********************************************************
ok: [192.168.56.105]

TASK [web_server : Building Dockerfile] ******************************************************************
changed: [192.168.56.105]

TASK [web_server : Checking Docker Images] ***************************************************************
changed: [192.168.56.105]

TASK [web_server : debug] ********************************************************************************
ok: [192.168.56.105] => {
    "command_output.stdout_lines": [
        "REPOSITORY   TAG       IMAGE ID       CREATED         SIZE",
        "cpe232       1.2       19645aa96cf2   5 minutes ago   327MB",
        "ubuntu       latest    a8780b506fa4   2 weeks ago     77.8MB"
    ]
}

PLAY RECAP ***********************************************************************************************
192.168.56.105             : ok=10   changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.106             : ok=10   changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

madiane@workstation:~/Agpaoa_HOA_11$
```

Figure 5.6 Running the insbuildockf.yml playbook

I run the insbuildockf.yml playbook and as the figure shown above, the execution of tasks is all successful. The first executed task is the pre-tasks, then the tasks for db_servers and then the tasks for web_servers. The docker images for each remote server are also shown to verify that the Dockerfile was successfully built and installed. In addition, based on the play recap, there are no errors that occur for running the playbook.

6. Add, commit and push it to your repository.

```
madiane@workstation:~/Agpaoa_HOA_11$ git add *
madiane@workstation:~/Agpaoa_HOA_11$ git commit -m "HOA_11"
[main 4129933] HOA_11
 4 files changed, 45 insertions(+), 20 deletions(-)
 rewrite roles/web_server/tasks/main.yml (100%)
madiane@workstation:~/Agpaoa_HOA_11$ git push
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Compressing objects: 100% (14/14), done.
Writing objects: 100% (22/22), 2.30 KiB | 168.00 KiB/s, done.
Total 22 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To github.com:qmja/Agpaoa_HOA_11.git
   ec5795b..4129933  main -> main
```

Figure 6.1 Saving the files to the Agpaoa_HOA_11 repository in GitHub

In order to save the files in the Agpaoa_HOA_11 repository in GitHub, I executed the commands "*git add **", "*git commit -m "HOA_11""*, and "*git push*".

Reflections:

Answer the following:

1. What are the benefits of implementing containerizations?

   Implementing containerizations is beneficial to the users, since containers utilize less resources than virtual machines, they can run anywhere like platforms such as Linode, Digital Ocean, Google Cloud and others, can easily be copied and deployed, it separates softwares from the host system, and cheaper than virtual machines.

Conclusions:

In conclusion, this activity helped me learn about docker and its benefits. In addition, I learned how to create docker images, dockerfile and to form a workflow in installing and building Dockerfile in remote servers using Ansible. In this activity, I was able to apply my previous knowledge and improve my skills in creating and consolidating playbooks such as the implementation of roles. This activity was also essential to help me improve in forming workflows according to the given tasks. I've encountered some errors but I managed to solve it and through these process I learned and realized new things about docker and ansible.

Honor Pledge:

"I affirm that I will not give or receive unauthorized help on this activity and that all work will be my own."