Name: Ian Carlo T. Bello	**Date Performed:** November 15, 2022
Course/Section: CPE31S24 – CPE232	**Date Submitted:** November 16, 2022
Instructor: Dr. Jonathan V. Taylar	**Semester and SY:** 1st sem – 3rd year

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.

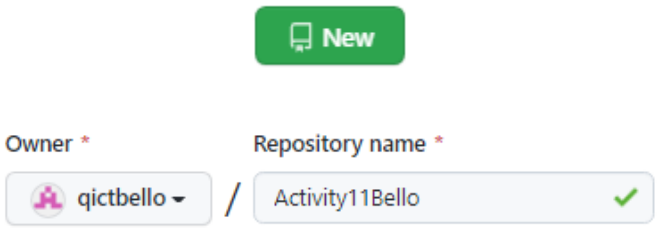
Source: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm

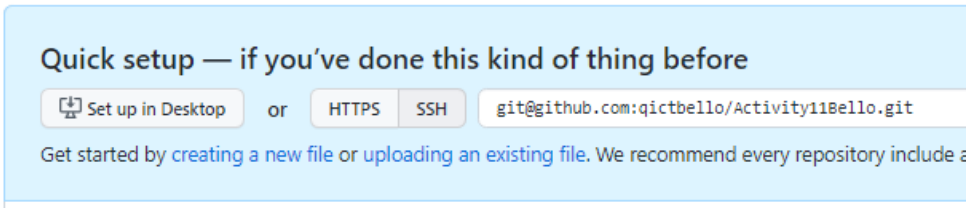
3. Tasks

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)

First we need to create a repository for this activity





Next, we will git clone it and work in this repository

```
ubuntuhost@workstation:~$ git clone git@github.com:qictbello/Activity11Bello.git
Cloning into 'Activity11Bello'...
warning: You appear to have cloned an empty repository.
ubuntuhost@workstation:~$ cd Activity11Bello/
ubuntuhost@workstation:~/Activity11Bello$
```

Next we will be installing docker and enabling the docker

```
ubuntuhost@workstation:~/Activity11Bello$ sudo apt install docker.io
[sudo] password for ubuntuhost:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (20.10.12-0ubuntu4).
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libflashrom1 libftdi1-2
  libgstreamer-plugins-bad1.0-0 libva-wayland2
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntuhost@workstation:~/Activity11Bello$ systemctl enable docker
ubuntuhost@workstation:~/Activity11Bello$ systemctl start docker
```

We will check the status if its running

```
ubuntuhost@workstation:~/Activity11Bello$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset>
     Active: active (running) since Tue 2022-11-15 10:46:28 PST; 52min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 1272 (dockerd)
      Tasks: 19
     Memory: 51.9M
        CPU: 1.726s
     CGroup: /system.slice/docker.service
             └─1272 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/cont>

Nov 15 10:46:26 workstation dockerd[1272]: time="2022-11-15T10:46:26.604097570+>
Nov 15 10:46:26 workstation dockerd[1272]: time="2022-11-15T10:46:26.657649594+>
Nov 15 10:46:26 workstation dockerd[1272]: time="2022-11-15T10:46:26.811825312+>
Nov 15 10:46:26 workstation dockerd[1272]: time="2022-11-15T10:46:26.864589950+>
Nov 15 10:46:27 workstation dockerd[1272]: time="2022-11-15T10:46:27.027389608+>
Nov 15 10:46:27 workstation dockerd[1272]: time="2022-11-15T10:46:27.136630959+>
Nov 15 10:46:27 workstation dockerd[1272]: time="2022-11-15T10:46:27.943458942+>
Nov 15 10:46:27 workstation dockerd[1272]: time="2022-11-15T10:46:27.981196121+>
Nov 15 10:46:28 workstation systemd[1]: Started Docker Application Container En>
Nov 15 10:46:28 workstation dockerd[1272]: time="2022-11-15T10:46:28.170722326+>
lines 1-22/22 (END)
```

Next we will create and add docker group to our current user

```
ubuntuhost@workstation:~/Activity11Bello$ sudo groupadd docker
groupadd: group 'docker' already exists
ubuntuhost@workstation:~/Activity11Bello$ sudo usermod -aG docker $USER
ubuntuhost@workstation:~/Activity11Bello$
```

We can verify it if we can run docker without sudo

```
ubuntuhost@workstation:~/Activity11Bello$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

ubuntuhost@workstation:~/Activity11Bello$
```

Next, we will create a dockerfile to install apache as a webserver and mariadb as a dbserver

```
  GNU nano 6.2                                  dockerfile *
FROM ubuntu
MAINTAINER ubuntuhost <qictbello@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 mariadb-server

# Set entrypoint
ENTRYPOINT apache2ctl -D FOREGROUND
```

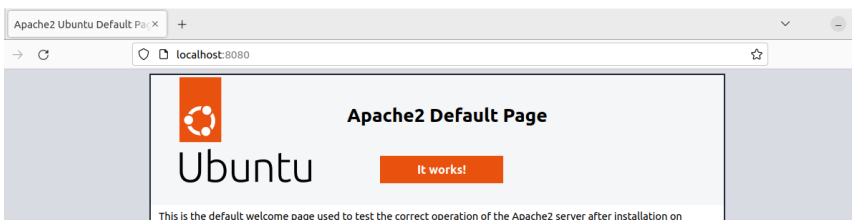
We will test it first in our workstation

```
ubuntuhost@workstation:~/Activity11Bello$ docker build -t container:1.0 .
Sending build context to Docker daemon  44.54kB
Step 1/6 : FROM ubuntu
 ---> a8780b506fa4
Step 2/6 : MAINTAINER ubuntuhost <qictbello@tip.edu.ph>
 ---> Using cache
 ---> 113ae7ff1e90
Step 3/6 : ARG DEBIAN_FRONTEND=noninteractive
 ---> Using cache
 ---> 473cab95303e
Step 4/6 : RUN apt update; apt dist-upgrade -y
 ---> Using cache
 ---> 3bb77baa7965
Step 5/6 : RUN apt install -y apache2 mariadb-server
 ---> Running in cffc28e7723a

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists...
Building dependency tree...
Reading state information...
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 ca-certificates file galera-4
```

```
ubuntuhost@workstation:~/Activity11Bello$ docker run -d -it -p 8080:80 container
:1.0
9db17a1d281602c59656d98bda2292952f5d44924ac3469c68c82ce094d5a770
ubuntuhost@workstation:~/Activity11Bello$ docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED          STATUS
        PORTS                                       NAMES
9db17a1d2816   container:1.0   "/bin/sh -c 'apache2…"   13 seconds ago   Up 2 se
conds   0.0.0.0:8080->80/tcp, :::8080->80/tcp   distracted_ellis
ubuntuhost@workstation:~/Activity11Bello$
```



Next we will install and build dockerfile using ansible

```
ubuntuhost@workstation:~/Activity11Bello$ nano inventory
ubuntuhost@workstation:~/Activity11Bello$ nano ansible.cfg
```

```
  GNU nano 6.2                                  inventory *
server1
servercent
```

```
  GNU nano 6.2                                  ansible.cfg
[defaults]
command_warnings=False
deprecation_warnings=False
inventory=inventory
private_key_file = ~/.ssh/ansible
```

```yaml
---
- hosts: all
  become: true
  pre_tasks:

  -  name: update repository index CentOS
     dnf:
       update_cache: yes
     changed_when: false
     when: ansible_distribution == "CentOS"

  -  name: update repository index Ubuntu
     apt:
       upgrade: dist
       update_cache: yes
     changed_when: false
     when: ansible_distribution == "Ubuntu"
```

```yaml
- hosts: all
  become: true
  tasks:

  -  name: install docker ubuntu
     apt:
       name: docker.io
       state: latest
     when: ansible_distribution == "Ubuntu"

  -  name: install docker centos
     shell: 'curl -fsSL https://get.docker.com/ | sh'
     when: ansible_distribution == "CentOS"

  -  name: install docker sdk ubuntu
     apt:
       name: python3-docker
       update_cache: yes
       cache_valid_time: 3600
     when: ansible_distribution == "Ubuntu"

  -  name: docker permission ubuntu
     shell: 'sudo usermod -aG docker $USER'
     when: ansible_distribution == "Ubuntu"
```

```yaml
- name: install docker sdk centos
  yum:
    name: python-docker-py
    update_cache: yes
  when: ansible_distribution == "CentOS"

- name: docker permission centos
  shell: 'sudo usermod -aG docker $(whoami)'
  when: ansible_distribution == "CentOS"


- name: start and enable docker
  service:
    name: docker
    state: started

- name: cpy dockerfile
  copy: src=dockerfile dest=/tmp/path/

- name: docker build
  docker_image:
    name: containeransible
    build:
      path: /tmp/path/
      args:
        listen_port: 8080
    source: build
```

Next, we will be running the docker.yml to both servers this will install and setup docker. We will use the dockerfile we made and test recently and this will be deployed and build as image file into the servers

```
ubuntuhost@workstation:~/Activity11Bello$ ansible-playbook --ask-become-pass docker.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [servercent]
ok: [server1]

TASK [update repository index CentOS] ******************************************
skipping: [server1]
ok: [servercent]

TASK [update repository index Ubuntu] ******************************************
skipping: [servercent]
ok: [server1]

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

TASK [Gathering Facts] *********************************************************
ok: [servercent]
ok: [server1]

TASK [install docker ubuntu] ***************************************************
skipping: [servercent]
ok: [server1]

TASK [install docker centos] ***************************************************
skipping: [server1]
changed: [servercent]
```

```
TASK [install docker sdk ubuntu] ***********************************************************************
skipping: [servercent]
ok: [server1]

TASK [docker permission ubuntu] ************************************************************************
skipping: [servercent]
changed: [server1]

TASK [install docker sdk centos] ***********************************************************************
skipping: [server1]
ok: [servercent]

TASK [docker permission centos] ************************************************************************
skipping: [server1]
changed: [servercent]

TASK [start and enable docker] *************************************************************************
ok: [servercent]
ok: [server1]

TASK [cpy dockerfile] **********************************************************************************
changed: [server1]
ok: [servercent]

TASK [docker build] ************************************************************************************
ok: [servercent]
ok: [server1]

PLAY RECAP *********************************************************************************************
server1                    : ok=9    changed=2    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0
servercent                 : ok=9    changed=2    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0
```

Next, we will check both servers if we created the image using docker images using ssh in our workstation

```
ubuntuhost@server1:~$ docker images
REPOSITORY          TAG       IMAGE ID       CREATED       SIZE
containeransible    latest    d291c28423cb   5 hours ago   512MB
ubuntu              latest    a8780b506fa4   12 days ago   77.8MB
ubuntuhost@server1:~$
```

```
ubuntuhost@workstation:~/Activity11Bello$ ssh servercent
Last login: Tue Nov 15 22:00:46 2022 from 192.168.56.102
Ansible Managed node by Bello
[ubuntuhost@localhost ~]$ docker images
REPOSITORY          TAG       IMAGE ID       CREATED       SIZE
containeransible    latest    5d6ffbc94260   2 hours ago   512MB
ubuntu              latest    a8780b506fa4   12 days ago   77.8MB
[ubuntuhost@localhost ~]$
```

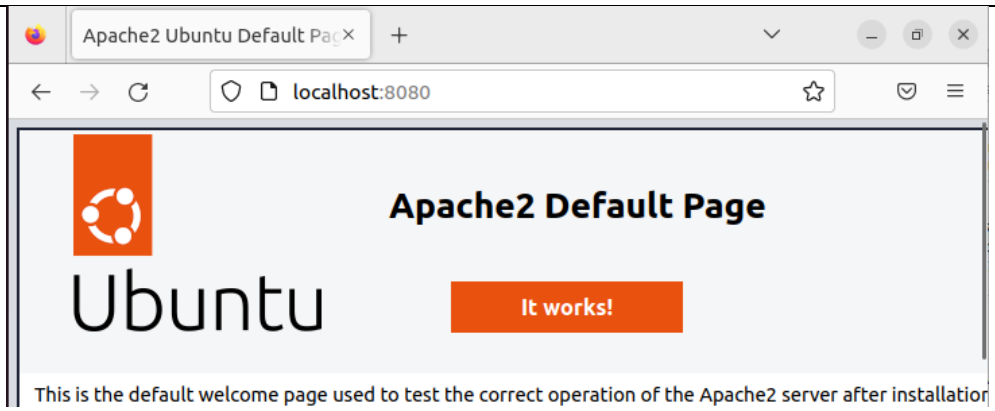
So, both servers do have the created images we can run it in ssh and check if the services are working

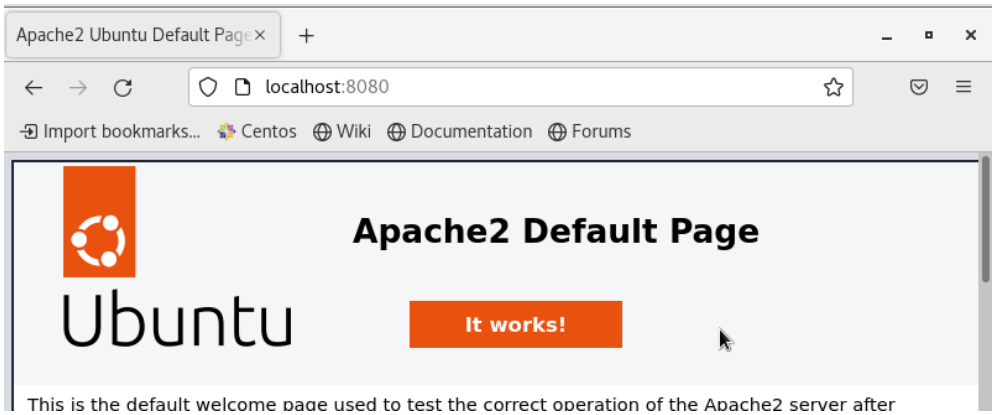
```
[ubuntuhost@localhost ~]$ docker run -d -it -p 8080:80 containeransible
7ccea1a2af93c28ff3b3f6a4764ce881f5a031bf0722c682e3c869c812078c99
[ubuntuhost@localhost ~]$
```

```
[ubuntuhost@localhost ~]$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                                       NAMES
7ccea1a2af93   containeransible   "/bin/sh -c 'apache2…"   29 seconds ago   Up 23 seconds   0.0.0.0:8080->80/tcp, :::8080->80/tcp       epic_mclar
en
[ubuntuhost@localhost ~]$
```

```
ubuntuhost@server1:~$ docker run -d -it -p 8080:80 containeransible
fde7dc6221c7a237d87e67716e06b5d12f36f577662c6d51710d24ce5de4bf0e
ubuntuhost@server1:~$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                                       NAMES
fde7dc6221c7   containeransible   "/bin/sh -c 'apache2…"   26 seconds ago   Up 4 seconds    0.0.0.0:8080->80/tcp, :::8080->80/tcp       lucid_heyro
vsky
ubuntuhost@server1:~$
```

So, both are running let's see if both servers have the services running in browser





Git add, commit, and push of repository

```
ubuntuhost@workstation:~/Activity11Bello$ git add -A
ubuntuhost@workstation:~/Activity11Bello$ git commit -m "containerization"
[main (root-commit) a0a033a] containerization
 4 files changed, 91 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 docker.yml
 create mode 100644 dockerfile
 create mode 100644 inventory
ubuntuhost@workstation:~/Activity11Bello$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.11 KiB | 1.11 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qictbello/Activity11Bello.git
 * [new branch]      main -> main
ubuntuhost@workstation:~/Activity11Bello$
```

Reflections:

Answer the following:

1. What are the benefits of implementing containerizations?

Images are portable and efficient, making them faster and easier to deliver. It improves the security even more since it is inside a container. Faster startup and easier management of the image or container.

Conclusions:

In conclusion, in this activity, we tackled and learned more about Docker and how to use it with Ansible by distributing it to the servers. Dockerfile becomes useful because it's like creating a prebuilt image, and it will be easier to run on the servers since it is also in a container that doesn’t change anything in the environment, making it efficient. We encountered errors such as prerequisites, but it is easier to debug and base on the Ansible and Docker manuals.