Name: Bonifacio, Redj Guillian F.	**Date Performed:** October 24, 2025
Course/Section: CPE31S4	**Date Submitted:** October 24, 2025
Instructor: Engr. Valenzuela, Robin	**Semester and SY:** 2nd Sem, 25-26

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.

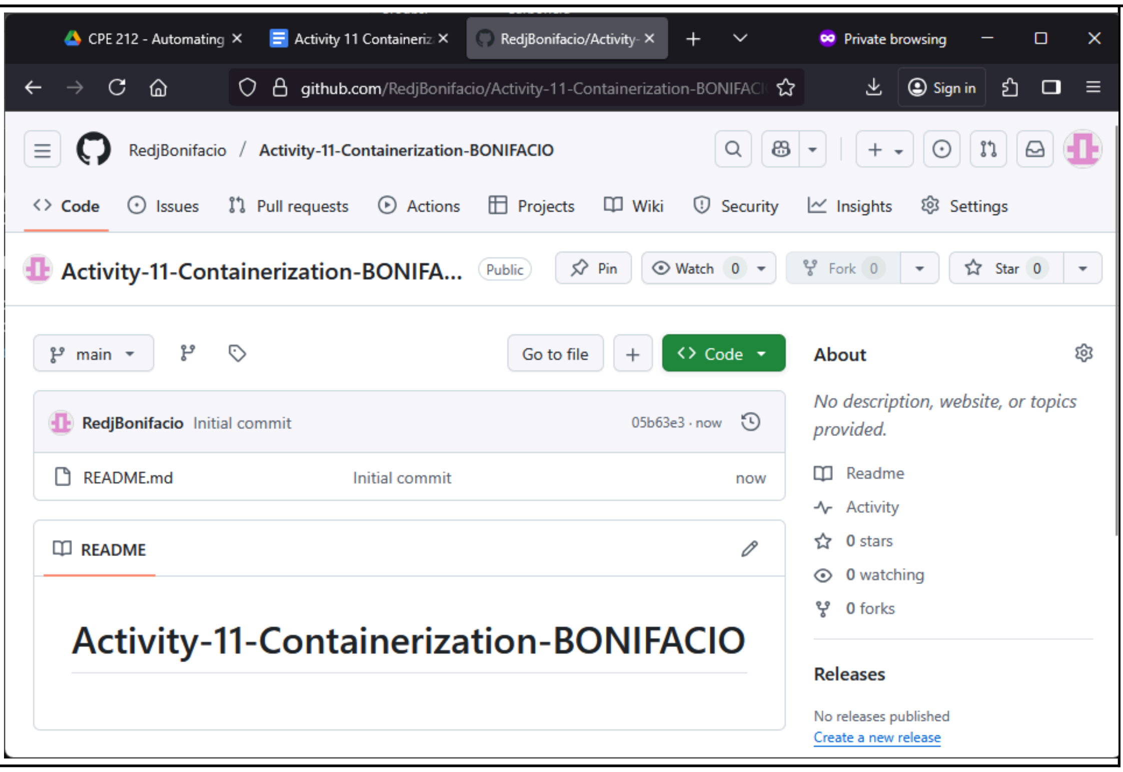
Source: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm

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)

1. Create a new repository for this activity.


CPE 212 - Automating | Activity 11 Containeriz | RedjBonifacio/Activity- | Private browsing

github.com/RedjBonifacio/Activity-11-Containerization-BONIFACI

RedjBonifacio / Activity-11-Containerization-BONIFACIO

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

Activity-11-Containerization-BONIFA... Public | Pin | Watch 0 | Fork 0 | Star 0

main | Go to file | + | Code

RedjBonifacio Initial commit 05b63e3 · now

README.md	Initial commit	now

README

Activity-11-Containerization-BONIFACIO

About

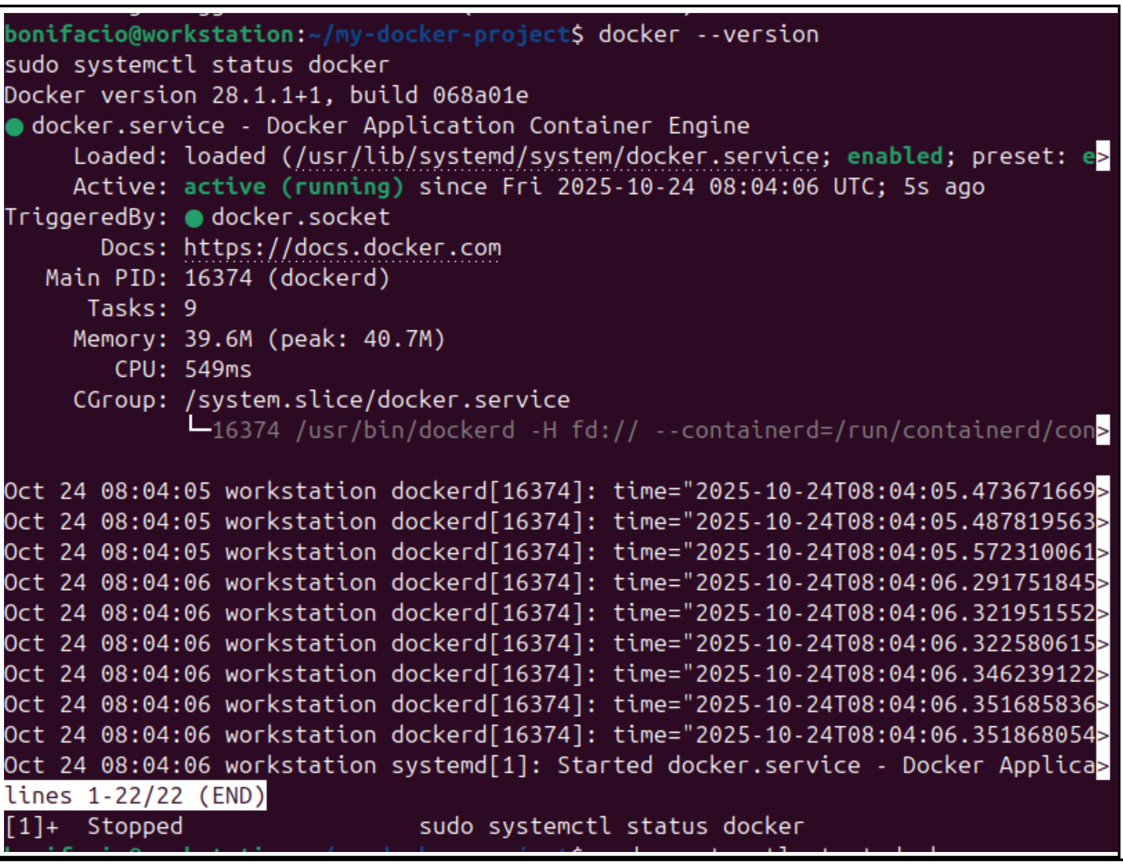
No description, website, or topics provided.

Readme
Activity
0 stars
0 watching
0 forks

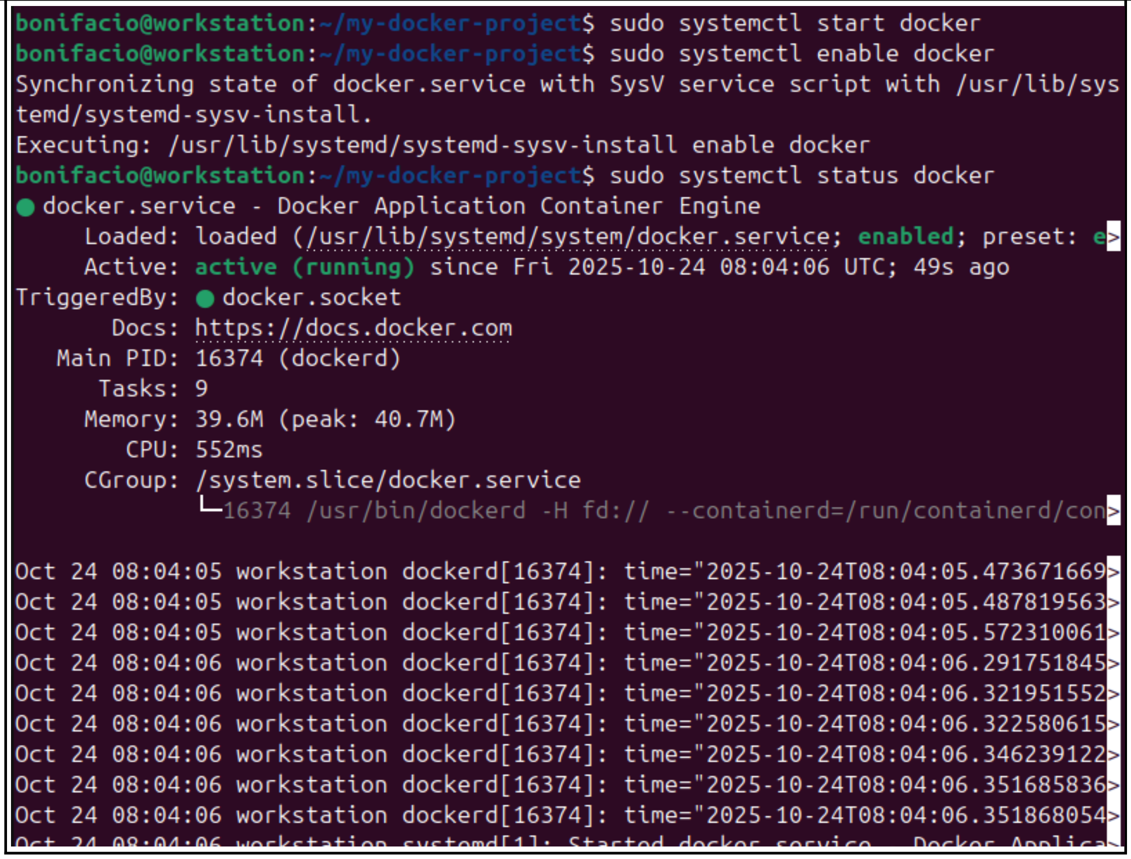
Releases

No releases published
Create a new release

2. Install Docker and enable the docker socket.

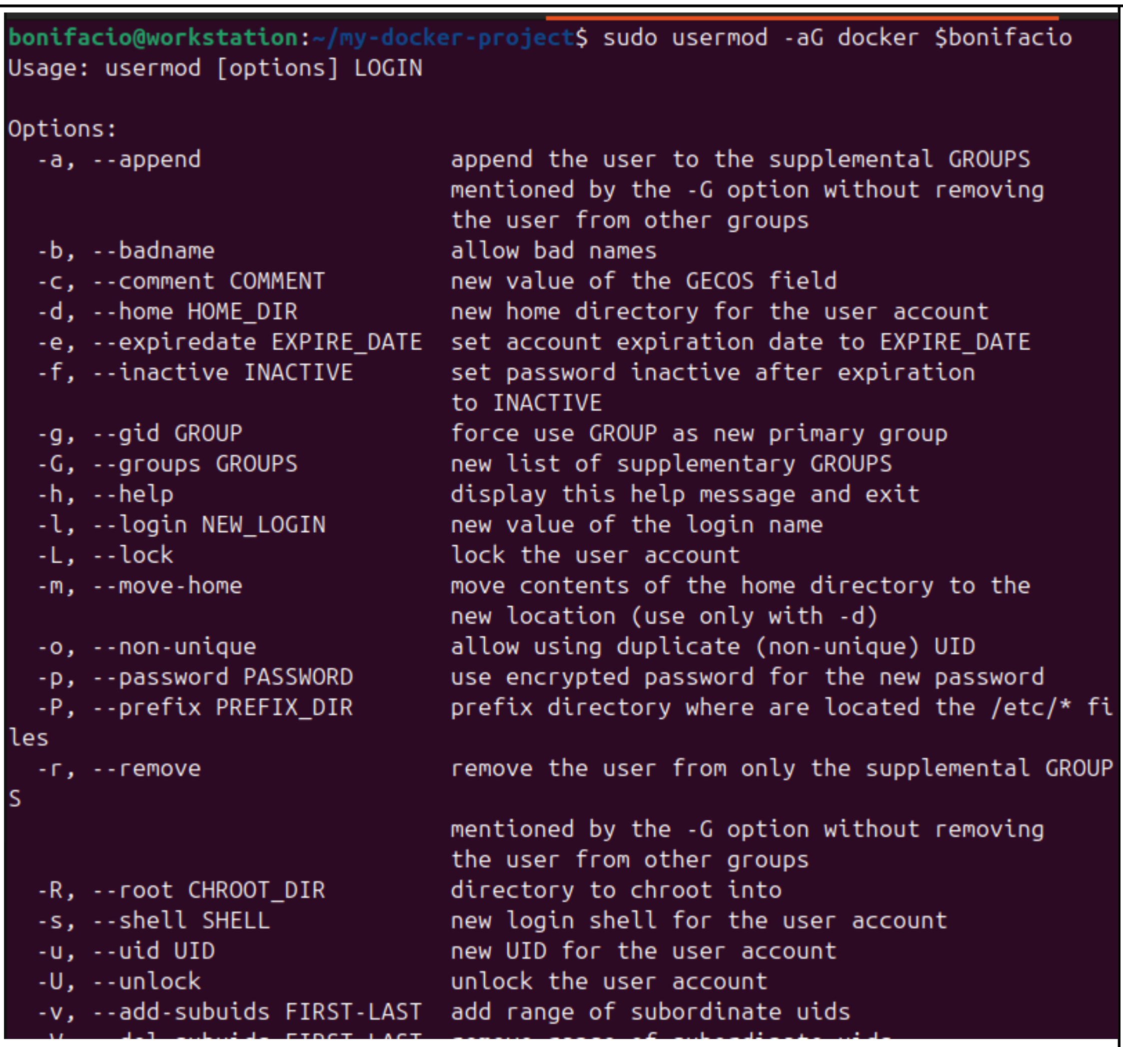


```
bonifacio@workstation:~/my-docker-project$ docker --version
sudo systemctl status docker
Docker version 28.1.1+1, build 068a01e
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: e>
     Active: active (running) since Fri 2025-10-24 08:04:06 UTC; 5s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 16374 (dockerd)
      Tasks: 9
     Memory: 39.6M (peak: 40.7M)
        CPU: 549ms
     CGroup: /system.slice/docker.service
             └─16374 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.473671669>
Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.487819563>
Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.572310061>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.291751845>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.321951552>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.322580615>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.346239122>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.351685836>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.351868054>
Oct 24 08:04:06 workstation systemd[1]: Started docker.service - Docker Applica>
lines 1-22/22 (END)
[1]+  Stopped                 sudo systemctl status docker
```

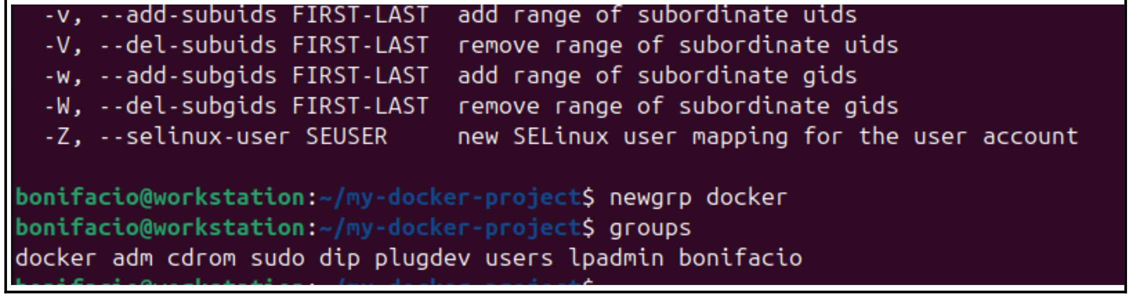


```
bonifacio@workstation:~/my-docker-project$ sudo systemctl start docker
bonifacio@workstation:~/my-docker-project$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /usr/lib/sys
temd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
bonifacio@workstation:~/my-docker-project$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: e>
     Active: active (running) since Fri 2025-10-24 08:04:06 UTC; 49s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 16374 (dockerd)
      Tasks: 9
     Memory: 39.6M (peak: 40.7M)
        CPU: 552ms
     CGroup: /system.slice/docker.service
             └─16374 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.473671669>
Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.487819563>
Oct 24 08:04:05 workstation dockerd[16374]: time="2025-10-24T08:04:05.572310061>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.291751845>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.321951552>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.322580615>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.346239122>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.351685836>
Oct 24 08:04:06 workstation dockerd[16374]: time="2025-10-24T08:04:06.351868054>
Oct 24 08:04:06 workstation systemd[1]: Started docker.service - Docker Applica>
```

3. Add to Docker group to your current user.

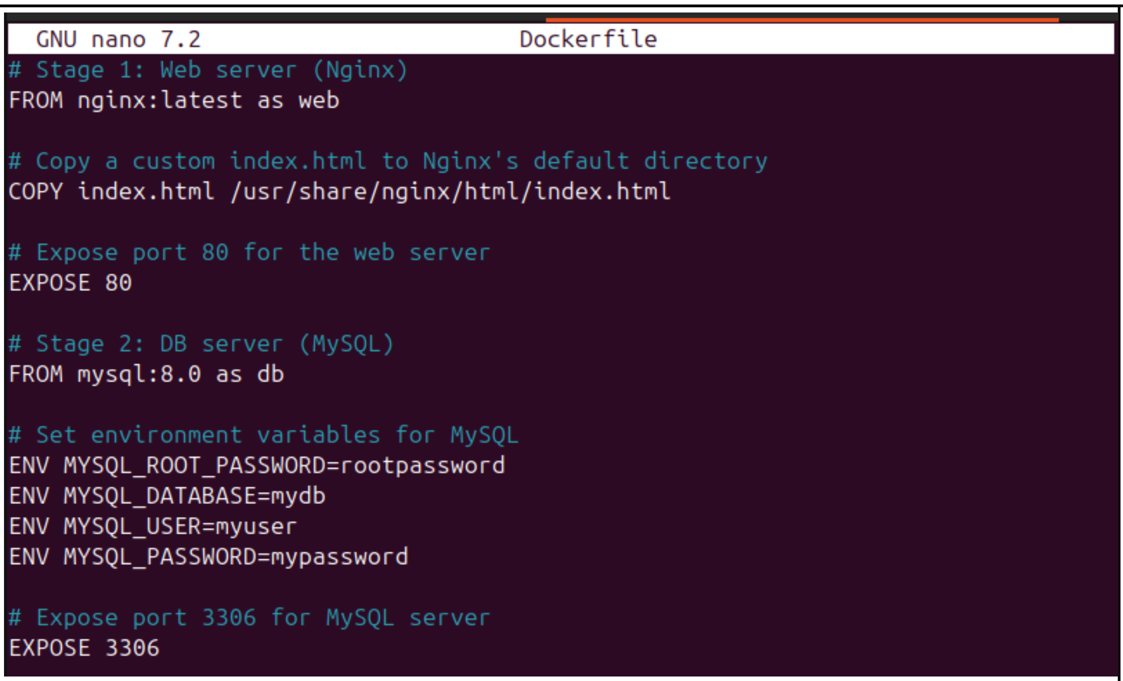


```
bonifacio@workstation:~/my-docker-project$ sudo usermod -aG docker $bonifacio
Usage: usermod [options] LOGIN

Options:
  -a, --append                  append the user to the supplemental GROUPS
                                mentioned by the -G option without removing
                                the user from other groups
  -b, --badname                 allow bad names
  -c, --comment COMMENT         new value of the GECOS field
  -d, --home HOME_DIR           new home directory for the user account
  -e, --expiredate EXPIRE_DATE  set account expiration date to EXPIRE_DATE
  -f, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -g, --gid GROUP               force use GROUP as new primary group
  -G, --groups GROUPS           new list of supplementary GROUPS
  -h, --help                    display this help message and exit
  -l, --login NEW_LOGIN         new value of the login name
  -L, --lock                    lock the user account
  -m, --move-home               move contents of the home directory to the
                                new location (use only with -d)
  -o, --non-unique              allow using duplicate (non-unique) UID
  -p, --password PASSWORD       use encrypted password for the new password
  -P, --prefix PREFIX_DIR       prefix directory where are located the /etc/* fi
les
  -r, --remove                  remove the user from only the supplemental GROUP
S
                                mentioned by the -G option without removing
                                the user from other groups
  -R, --root CHROOT_DIR         directory to chroot into
  -s, --shell SHELL             new login shell for the user account
  -u, --uid UID                 new UID for the user account
  -U, --unlock                  unlock the user account
  -v, --add-subuids FIRST-LAST  add range of subordinate uids
  -V, --del-subuids FIRST-LAST  remove range of subordinate uids
```

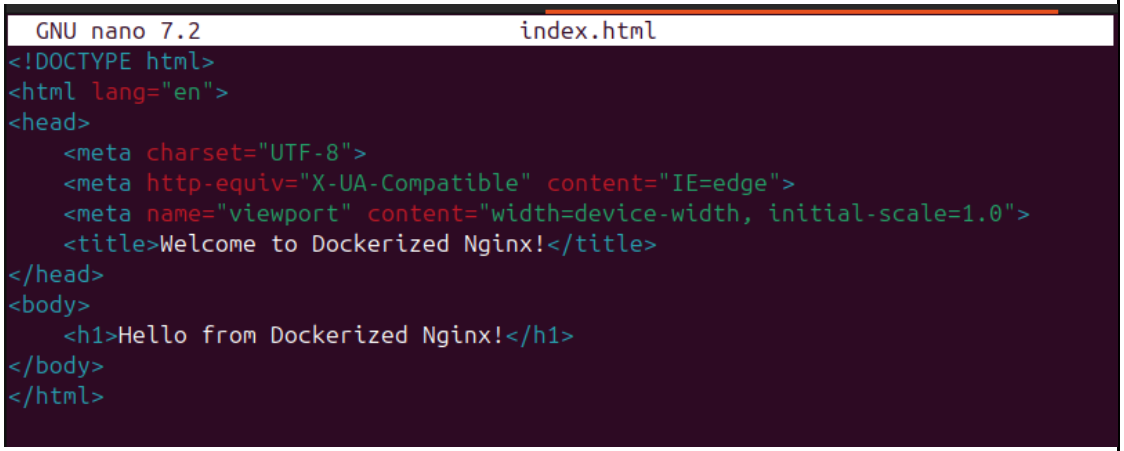


```
  -v, --add-subuids FIRST-LAST  add range of subordinate uids
  -V, --del-subuids FIRST-LAST  remove range of subordinate uids
  -w, --add-subgids FIRST-LAST  add range of subordinate gids
  -W, --del-subgids FIRST-LAST  remove range of subordinate gids
  -Z, --selinux-user SEUSER     new SELinux user mapping for the user account

bonifacio@workstation:~/my-docker-project$ newgrp docker
bonifacio@workstation:~/my-docker-project$ groups
docker adm cdrom sudo dip plugdev users lpadmin bonifacio
bonifacio@workstation:~/my-docker-project$
```

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

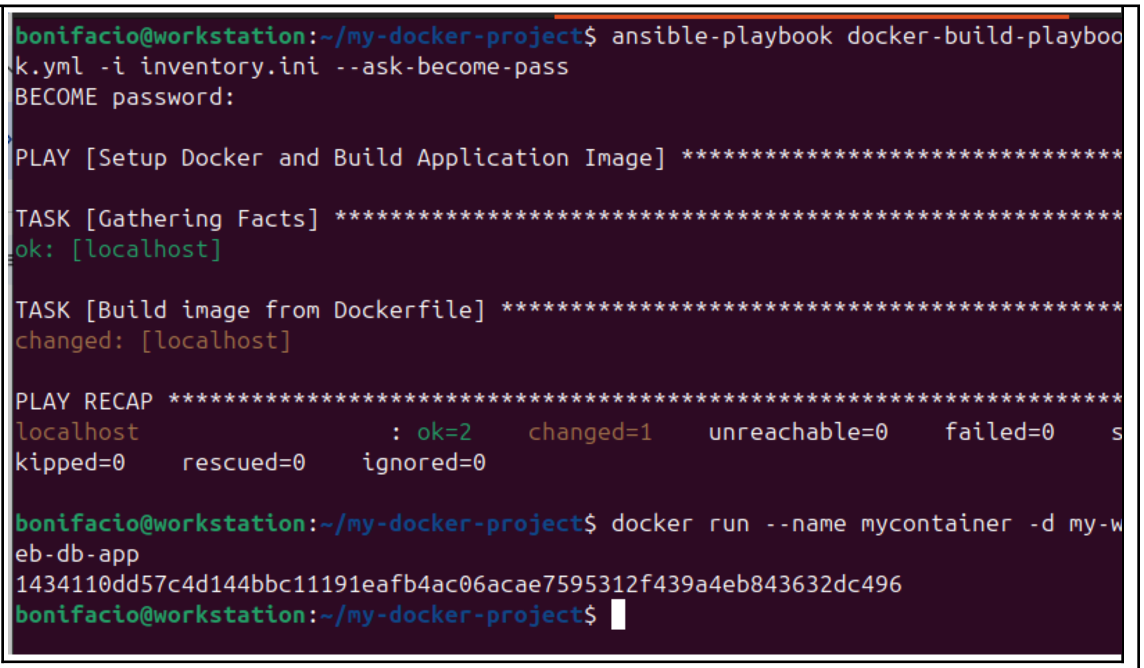


```
  GNU nano 7.2                                 Dockerfile
# Stage 1: Web server (Nginx)
FROM nginx:latest as web

# Copy a custom index.html to Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html

# Expose port 80 for the web server
EXPOSE 80

# Stage 2: DB server (MySQL)
FROM mysql:8.0 as db

# Set environment variables for MySQL
ENV MYSQL_ROOT_PASSWORD=rootpassword
ENV MYSQL_DATABASE=mydb
ENV MYSQL_USER=myuser
ENV MYSQL_PASSWORD=mypassword

# Expose port 3306 for MySQL server
EXPOSE 3306
```



```
  GNU nano 7.2                                 index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Welcome to Dockerized Nginx!</title>
</head>
<body>
    <h1>Hello from Dockerized Nginx!</h1>
</body>
</html>
```

5. Install and build the Dockerfile using Ansible.



```
bonifacio@workstation:~/my-docker-project$ ansible-playbook docker-build-playboo
k.yml -i inventory.ini --ask-become-pass
BECOME password:

PLAY [Setup Docker and Build Application Image] *********************************

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

TASK [Build image from Dockerfile] **********************************************
changed: [localhost]

PLAY RECAP **********************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

bonifacio@workstation:~/my-docker-project$ docker run --name mycontainer -d my-w
eb-db-app
1434110dd57c4d144bbc11191eafb4ac06acae7595312f439a4eb843632dc496
bonifacio@workstation:~/my-docker-project$ 
```

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

Reflections:

Answer the following:

1. What are the benefits of implementing containerizations?
   - The main benefit of implementing containerization is that it allows for faster, more reliable deployment by packaging an application and its dependencies together to run consistently across various computing environments.

Conclusions:

Containerization utilizing Docker and automated by Ansible effectively met the requirement of a continuous delivery workflow. The creation of a single Docker image containing both the web and DB servers did not only allow for an environment that could be deployed anywhere but also made the whole process faster. This is a clear indication of how IaC (Infrastructure as Code) together with container technology can greatly enhance the speed and reliability of software deployment.