**Build\_Docker\_Jenkins\_Pipeline\_to\_Implement\_CI-CD\_Workflow\_steps**

**SECTION A**

**PART 1:-**

#Ubuntu machine setup with docker

Steps:-

1:-Setup Ubuntu-20.04 linux machine on physical machine or vm (i have choosed physical machine)

2:-Launch terminal and run below commands with sudo permission for installing required packages

such as docker , docker-compose with thier dependency

sudo apt update

sudo apt install docker.io

Output :-

Reading package lists... Done

Building dependency tree

Reading state information... Done

The following packages were automatically installed and are no longer required:

chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libllvm11 linux-image-5.8.0-63-generic linux-modules-5.8.0-63-generic linux-modules-extra-5.8.0-63-generic shim

Use 'sudo apt autoremove' to remove them.

The following additional packages will be installed:

bridge-utils containerd git git-man liberror-perl pigz runc ubuntu-fan

Suggested packages:

ifupdown aufs-tools btrfs-progs cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-cvs

git-mediawiki git-svn

The following NEW packages will be installed:

bridge-utils containerd docker.io git git-man liberror-perl pigz runc ubuntu-fan

0 upgraded, 9 newly installed, 0 to remove and 33 not upgraded.

Need to get 79.5 MB of archives.

After this operation, 397 MB of additional disk space will be used.

sudo apt install docker-compose

Output :-

Reading package lists... Done

Building dependency tree

Reading state information... Done

The following packages were automatically installed and are no longer required:

chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libllvm11 linux-image-5.8.0-63-generic linux-modules-5.8.0-63-generic linux-modules-extra-5.8.0-63-generic shim

Use 'sudo apt autoremove' to remove them.

The following additional packages will be installed:

python3-attr python3-cached-property python3-docker python3-dockerpty python3-docopt python3-importlib-metadata python3-jsonschema python3-more-itertools python3-pyrsistent python3-texttable

python3-websocket python3-zipp

Suggested packages:

python-attr-doc python-jsonschema-doc

The following NEW packages will be installed:

docker-compose python3-attr python3-cached-property python3-docker python3-dockerpty python3-docopt python3-importlib-metadata python3-jsonschema python3-more-itertools python3-pyrsistent

python3-texttable python3-websocket python3-zipp

0 upgraded, 13 newly installed, 0 to remove and 33 not upgraded.

Need to get 445 kB of archives.

After this operation, 2,574 kB of additional disk space will be used.

--------------------------------------------------------------------

3 :- Add user to docker group so that user can do docker operation without sudo

sudo usermod -aG docker gk

Outout :-

gk@gk-ThinkPad-E15-Gen-2:~$ id gk

uid=1000(gk) gid=1000(gk) groups=1000(gk),4(adm),24(cdrom),27(sudo),29(audio),30(dip),46(plugdev),120(lpadmin),131(lxd),132(sambashare),998(eveusb),135(docker)

4:- Reload docker group so that we dont require os restart

newgrp docker

5:-Create new folder for our project workspace and enter into it

mkdir /home/gk/Documents/Simplilearn\_Devops\_PG\_2021/DevOps\_Course2/projects/Deploy\_Angular\_Application\_in\_Docker\_Container\_Project/app\_service\_compose

cd /home/gk/Documents/Simplilearn\_Devops\_PG\_2021/DevOps\_Course2/projects/Deploy\_Angular\_Application\_in\_Docker\_Container\_Project/app\_service\_compose

**PART 2 (Jenkins in docker):-**

#Run Jenkins in docker with jenkins data folder persistent using docker volumes

#We will use prebuilt jenkins image created by myself

**vi docker-compose.yml**

version: '3.3'

services:

jenkins:

image: ganeshkale/jenkins\_docker:1.0

ports:

- '8080:8080'

- '50000:50000'

networks:

- jenkins

volumes:

- jenkins-data:/var/jenkins\_home

- jenkins-docker-certs:/certs/client:ro

restart: always

networks:

jenkins:

volumes:

jenkins-data:

external: true

jenkins-docker-certs:

external: true

save file with Esc:wq

**Note :-**

1:-port number “50000” is used for communication with windows slave agents

#Now spin up jenkins service

**docker-compose up -d**

#Goto firefox browser and enter url : localhost:8080

**PART 2 :-**

#Now pull node latest docker image for creating and testing node sample application

#pull node latest image from docker hub

docker pull node

#Run node image

docker run -itd --name nodejs –network host node

#enter into container shell

docker exec -it nodejs /bin/bash

#Create sample app

ng new sample-angular-app

cd sample-angular-app

#To test application

ng serve

#check in browser

localhost:4200

#Now take out sample angular code from container to current path

docker cp 534f198e7d42:sample-angular-app .

**#Additional steps for saving angular development docker image**

docker commit 534f198e7d42 ganeshkale/angulardev:1.0

**Output :-**

gk@gk-ThinkPad-E15-Gen-2:~/Documents/Simplilearn\_Devops\_PG\_2021/DevOps\_Course2/projects/Deploy\_Angular\_Application\_in\_Docker\_Container\_Project/app\_service\_compose$ docker commit 534f198e7d42 ganeshkale/angulardev:1.0

sha256:41babbce7ad03569f22d74df82e5b7585095787bfce21b5022491a386e2b31a5

gk@gk-ThinkPad-E15-Gen-2:~/Documents/Simplilearn\_Devops\_PG\_2021/DevOps\_Course2/projects/Deploy\_Angular\_Application\_in\_Docker\_Container\_Project/app\_service\_compose$ docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

ganeshkale/angulardev 1.0 41babbce7ad0 6 seconds ago 1.38GB

<none> <none> 59db36ddf89b About an hour ago 1.24GB

node latest c66552d59c4b 35 hours ago 907MB

nginx latest dd34e67e3371 2 days ago 133MB

-------------------------------------------------------------------------------------------------------------------

**PART 3**

Steps :-

1:- Create folder app\_service\_compose

mkdir app\_service\_compose

cd app\_service\_compose

#Create dockerfile for angular app image

**vi Dockerfile**

#Stage 1 compile source code

FROM node:latest as build

MAINTAINER "GANESH KALE"

LABEL VERSION="1.0"

WORKDIR /usr/local/webapp

#copy source code folder

COPY sample-angular-app /usr/local/webapp/

#will install package dependency mentioned in .json files

RUN npm install

#will compile app

RUN npm run build

#Stage 2 run compile source code from stage1

FROM nginx:latest

#using stage 1 compiled app and copy to nginx default html

COPY --from=build /usr/local/webapp/dist/sample-angular-app /usr/share/nginx/html

#eposing nginx port 80 to docker host machine

EXPOSE 80

save above file with esc:wq

**vi docker-compose.yml**

version: '3.3'

services:

web:

#will build Dockerfile from current directory

build: .

labels:

- "Angular app webhosting"

ports:

#Mapping exposed port to host network port

- "80:80"

networks:

#Separate bridge network

- webhost

#will restart always on os boottime or when docker service restarted

restart: always

networks:

webhost:

save above file with esc:wq

**PART 4:-**

#Push the source code to github

git init

git add -f docker-compose.yml Dockerfile sample-angular-app

git branch -M main

git remote add origin [git@github.com](mailto:git@github.com):ganeshjkale/angular\_docker.git

git push -u origin main

Note:-

1:-laptop ssh public key was added in my github account

#Project Github URL

**https://github.com/ganeshjkale/angular\_docker**

**PART 5:-**

#Run docker compose image

docker-compose up -d

localhost:80

#For stopping application run below command from same path where compose file of application are located

docker-compose down

**SECTION B**

**PART 1:-**