Project2

Following are the specifications of the lifecycle:

1. Git workflow should be implemented. Since the company follows a monolithic architecture of development, you need to take care of version control. The release should happen only on the 25th of every month.
2. CodeBuild should be triggered once the commits are made in the master branch.
3. The code should be containerized with the help of the Dockerfile. The Dockerfile should be built every time if there is a push to GitHub. Create a custom Docker image using a Dockerfile.
4. As per the requirement in the production server, you need to use the Kubernetes cluster and the containerized code from Docker Hub should be deployed with 2 replicas. Create a NodePort service and configure the same for port 30008.
5. Create a Jenkins Pipeline script to accomplish the above task.
6. For configuration management of the infrastructure, you need to deploy the configuration on the servers to install necessary software and configurations.
7. Using Terraform, accomplish the task of infrastructure creation in the AWS cloud provider.

Architectural Advice

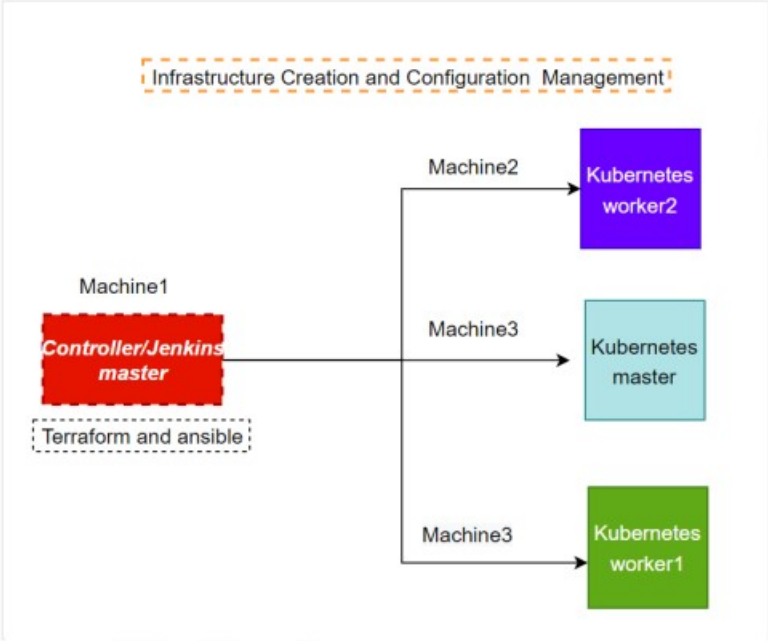
Softwares to be installed on the respective machines using configuration management.

Worker1: Jenkins, Java

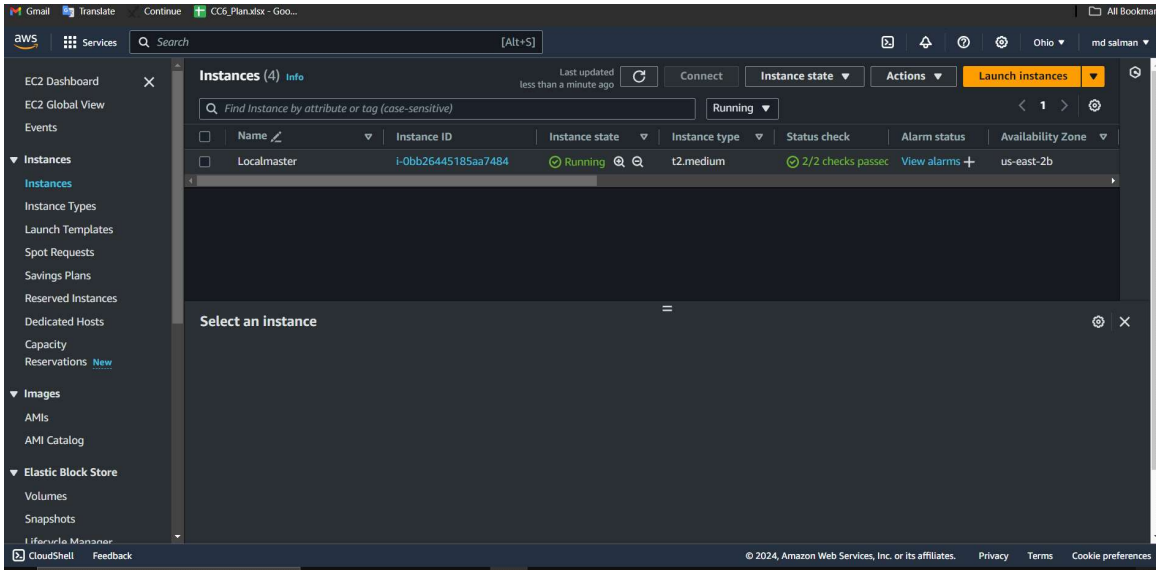
Worker2: Docker, Kubernetes

Worker3: Java, Docker, Kubernetes

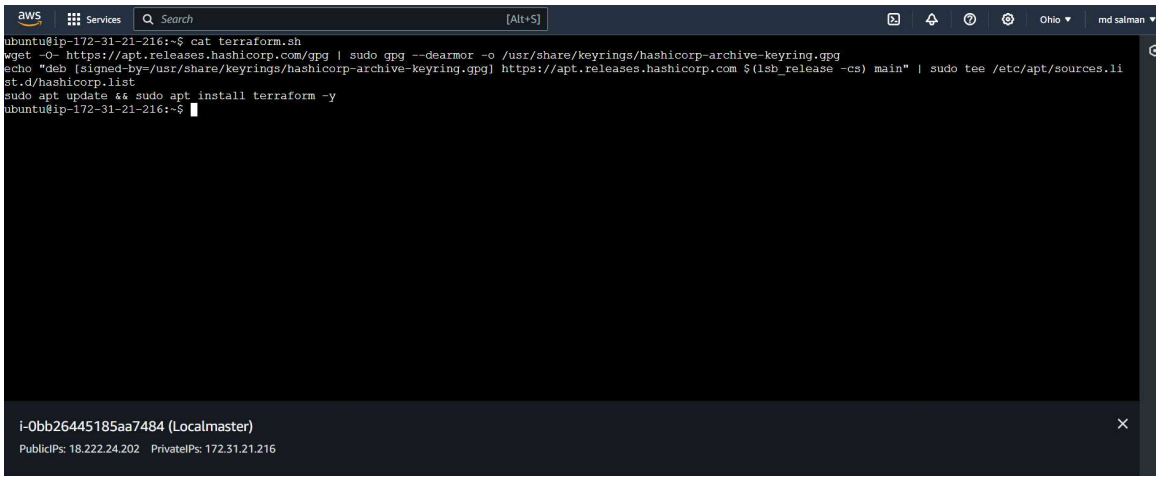
Worker4: Docker, Kubernetes



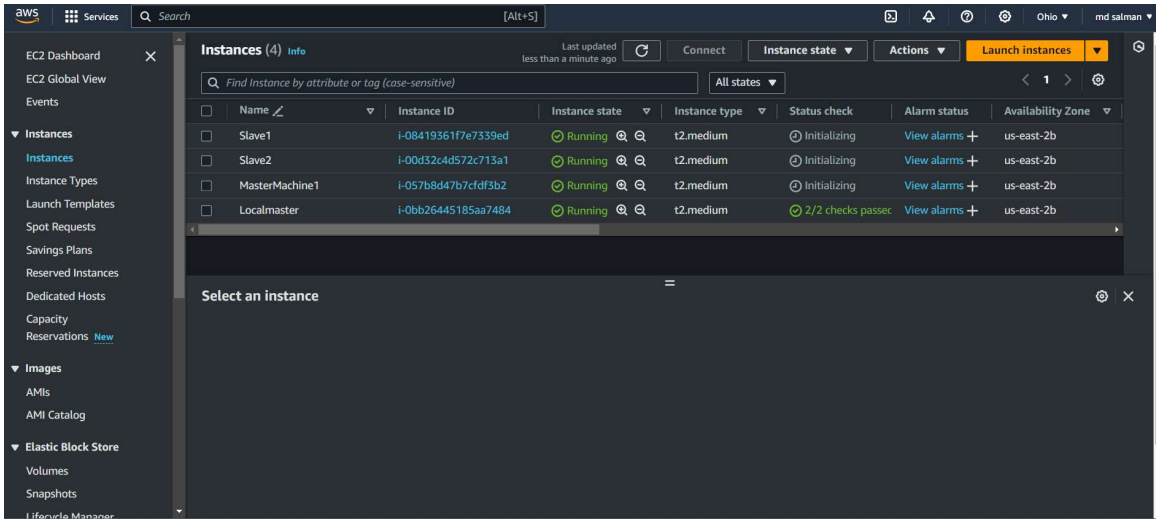
Step1: I have launced the ubuntu (OS) as local master which is my worker1



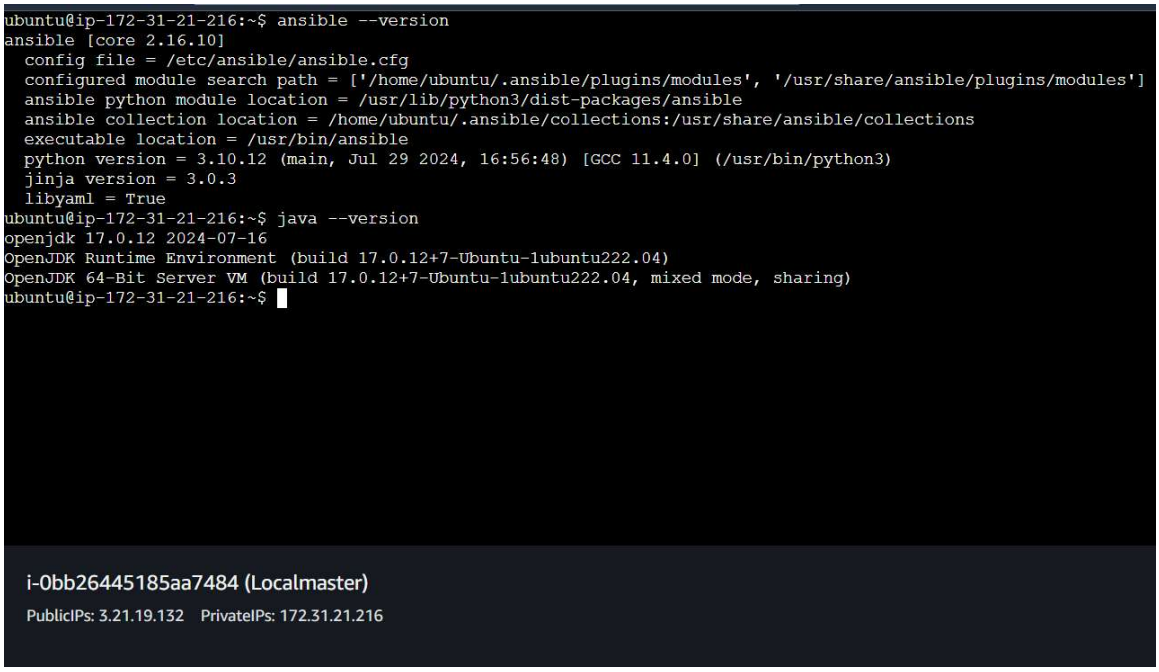
Step2:I have installed terrform and launched other 3 worker machines with the help of terraform.



As you can see my 3 workers machine up and running.I named them worker2:Master1 and worker3:slave1 and worker4:slave2.



Step3: I have installed ansible and java in local master machine which is worker1. In order to make connection with other worker nodes.



Step4: ssh key-gen in Local master worker1 and nodes ip address attachment in ansible hosts and also

```
ubuntu@ip-172-31-21-216:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
/home/ubuntu/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:+4km5K6YgS6bQnpwqs61xn19J2bt1Zb1+WEPsDAmgnE ubuntu@ip-172-31-21-216
The key's randomart image is:
+---[RSA 3072]----+
|                 |
|                 |
|     . E         |
|      +          |
|    . . S + .   .|
|.o.   .. + o o  *|
|++o..o  o   o .*+|
|B.o*..o..+ * ooo+|
|O*+..ooo. * +.  o|
+----[SHA256]-----+
ubuntu@ip-172-31-21-216:~$
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.222.24.202 PrivateIPs: 172.31.21.216

```
  GNU nano 6.2                                                    hosts
[master]
172.31.17.247
[slaves]
172.31.22.177
172.31.30.92

# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character
                                                  [ Read 72 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut      ^T Execute    ^C Location   M-U Undo      M-A Se
^X Exit      ^R Read File  ^\ Replace    ^U Paste    ^J Justify    ^/ Go To Line M-E Redo      M-6 Co
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.222.24.202 PrivateIPs: 172.31.21.216

Step5: And pasted the public key of local master in other worker nodes.

```
ubuntu@ip-172-31-21-216:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub  known_hosts  known_hosts.old
ubuntu@ip-172-31-21-216:~/.ssh$ id_rsa.pub
id_rsa.pub: command not found
ubuntu@ip-172-31-21-216:~/.ssh$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCVKzIDHxmTw8UWf3OG5tmWMybIzpo6/8iFz8qhwoLKgHaqPWRHCw9rgd5l2Ky/Q+QWbAKSpJNmvA6FKa3Y4OLgeIREsLfA
bzBFYPD4mf+ZMpPmMF306dAQk0mKF8nDFH2CIEd/5qciLUOUFMEBKP8zCsGGmfzbeKDlSnu5Wa8YYNlBkeVvOEvDvJQJFMuZrKLelXtEYaL9F6d0CaV6Lp/HCVrHHZlq7sue
LF1C9hjvglZxE9CwNoR0YTy6k0D4YMxnjS/EKsdCBGj+cUNhCWhX7oGQrwnxLmG5MdyTTWFwSE3RRqwt61Y/fEbOCczLR4GYCCdvKQ3AkmsnE5RE41ED63rhewH5acgVK/tS
3wBcBG4+chUM8GGpbtfEi6Cp+uaZqE1ThJBQfF/o2GSMllpz3DM7VcCV8Bc= ubuntu@ip-172-31-21-216
ubuntu@ip-172-31-21-216:~/.ssh$
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.222.24.202 PrivateIPs: 172.31.21.216

And as you can see we sucessfully able to make the connections with other Machines.

```
ubuntu@ip-172-31-21-216:/etc/ansible$ ansible -m ping all
172.31.22.177 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
172.31.30.92 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
172.31.17.247 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@ip-172-31-21-216:/etc/ansible$
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.222.24.202 PrivateIPs: 172.31.21.216

Step6: And now Iam gonna write the playbook and install the software packages in ther servers.

```
ubuntu@ip-172-31-21-216:/etc/ansible$ cat playbook.yml
---

- name: installing jenkins,java,kubernetes in master1
  become: true
  hosts: localhost
  tasks:
   - name: run script1
     script: script1.sh
- name: installing java,kubernetes in master1
  become: true
  hosts: master
  tasks:
   - name: run script2
     script: script2.sh
- name: installing kubernetes in slaves1 and 2
  become: true
  hosts: slaves
  tasks:
   - name: run script3
     script: script3.sh
ubuntu@ip-172-31-21-216:/etc/ansible$
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.222.24.202 PrivateIPs: 172.31.21.216

Step7:In playbook I am gonna write 3 scripts In script1 we have jenkins and java and kubernetes. Script 1 is for local master which is my worker1.

```
ubuntu@ip-172-31-21-216:/etc/ansible$ cat script1.sh
sudo apt update
sudo apt install openjdk-17-jdk -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins -y

sudo swapoff -a

# Create the .conf file to load the modules at bootup
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# sysctl params required by setup, params persist across reboots
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system

## Install CRIO Runtime
sudo apt-get update -y
sudo apt-get install -y software-properties-common curl apt-transport-https ca-certificates gpg

sudo curl -fsSL https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/cri-o-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg] https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /" | sudo tee /etc/apt/sources.list.d/cri-o.list

sudo apt-get update -y
sudo apt-get install -y cri-o

sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl start crio.service
```

Script2 is for master1 which is my worker2. java, and kubernetes

```
ubuntu@ip-172-31-21-216:/etc/ansible$ cat script2.sh
sudo apt update
sudo apt install openjdk-17-jdk -y
sudo swapoff -a

# Create the .conf file to load the modules at bootup
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# sysctl params required by setup, params persist across reboots
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system

## Install CRIO Runtime
sudo apt-get update -y
sudo apt-get install -y software-properties-common curl apt-transport-https ca-certificates gpg

sudo curl -fsSL https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/cri-o-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg] https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /" | sudo tee /etc/apt/sources.list.d/cri-o.list

sudo apt-get update -y
sudo apt-get install -y cri-o

sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl start crio.service
```

Script3 is for slaves1 and 2 (kubernetes.)

```
ubuntu@ip-172-31-21-216:/etc/ansible$ cat script3.sh
sudo apt-get update

sudo swapoff -a

# Create the .conf file to load the modules at bootup
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# sysctl params required by setup, params persist across reboots
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system

## Install CRIO Runtime
sudo apt-get update -y
sudo apt-get install -y software-properties-common curl apt-transport-https ca-certificates gpg

i-0bb26445185aa7484 (Localmaster)
PublicIPs: 18.222.24.202  PrivateIPs: 172.31.21.216
```

As you can see we successfully installed the required software packages and dependencies.

```
ubuntu@ip-172-31-21-216:~$ jenkins --version
2.475
ubuntu@ip-172-31-21-216:~$ java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu222.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu222.04, mixed mode, sharing)
ubuntu@ip-172-31-21-216:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-21-216:~$
```

i-0bb26445185aa7484 (Localmaster)

PublicIPs: 18.117.117.58 PrivateIPs: 172.31.21.216

```
ubuntu@ip-172-31-17-247:~$ java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu222.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu222.04, mixed mode, sharing)
ubuntu@ip-172-31-17-247:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-17-247:~$
```

i-057b8d47b7cfdf3b2 (MasterMachine1)

PublicIPs: 18.224.151.236 PrivateIPs: 172.31.17.247

```
ubuntu@ip-172-31-22-177:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-22-177:~$
```

i-08419361f7e7339ed (Slave1)

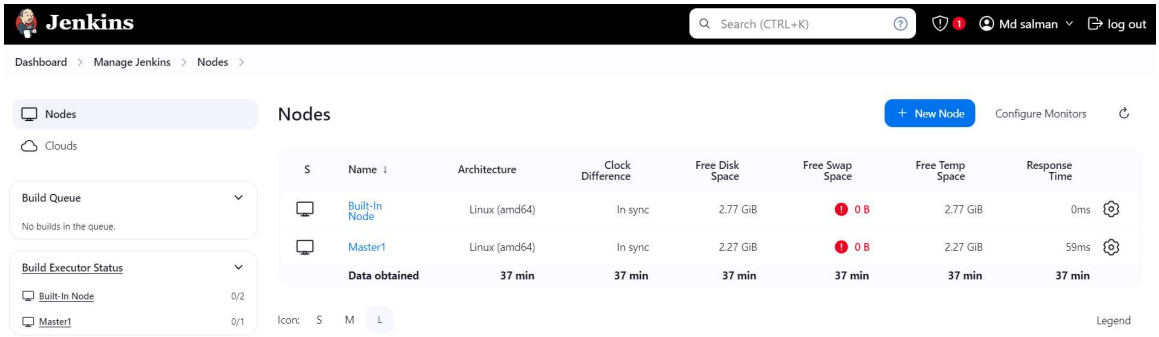
PublicIPs: 52.14.217.240 PrivateIPs: 172.31.22.177

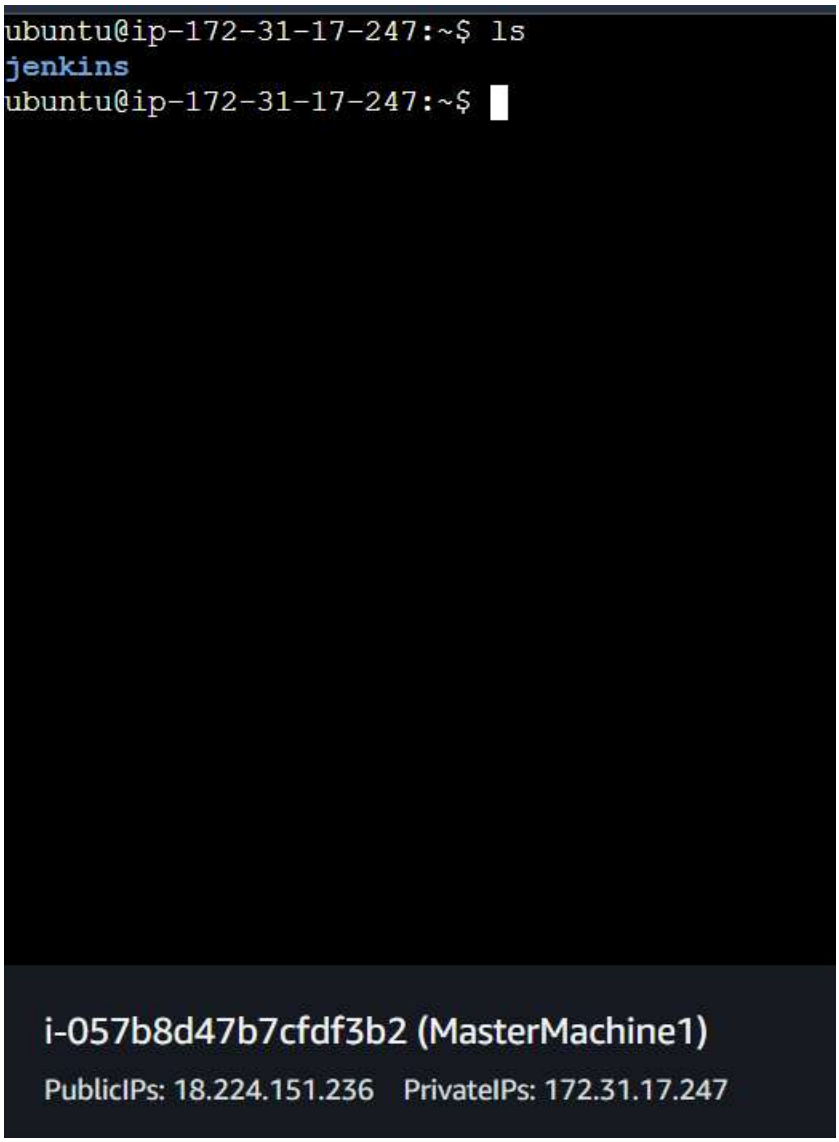
```
ubuntu@ip-172-31-30-92:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-30-92:~$
```

i-00d32c4d572c713a1 (Slave2)

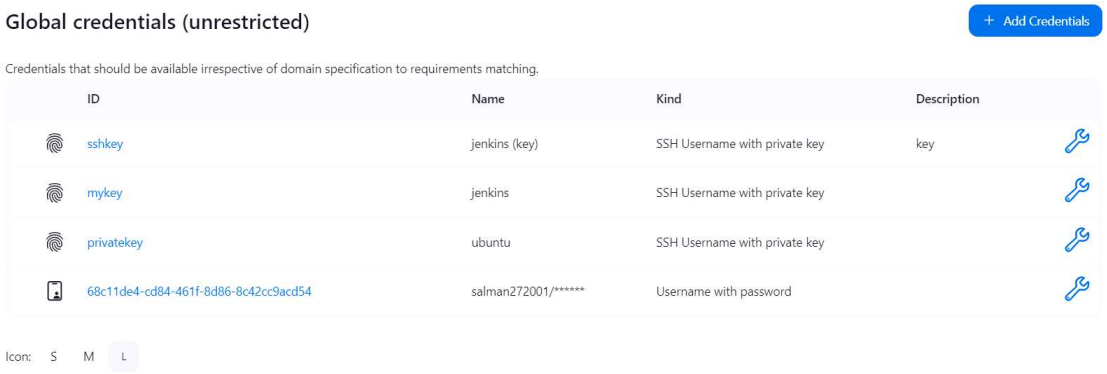
PublicIPs: 3.15.216.60 PrivateIPs: 172.31.30.92

Step8: I created node in jenkins and connected with master1 which is my worker2.

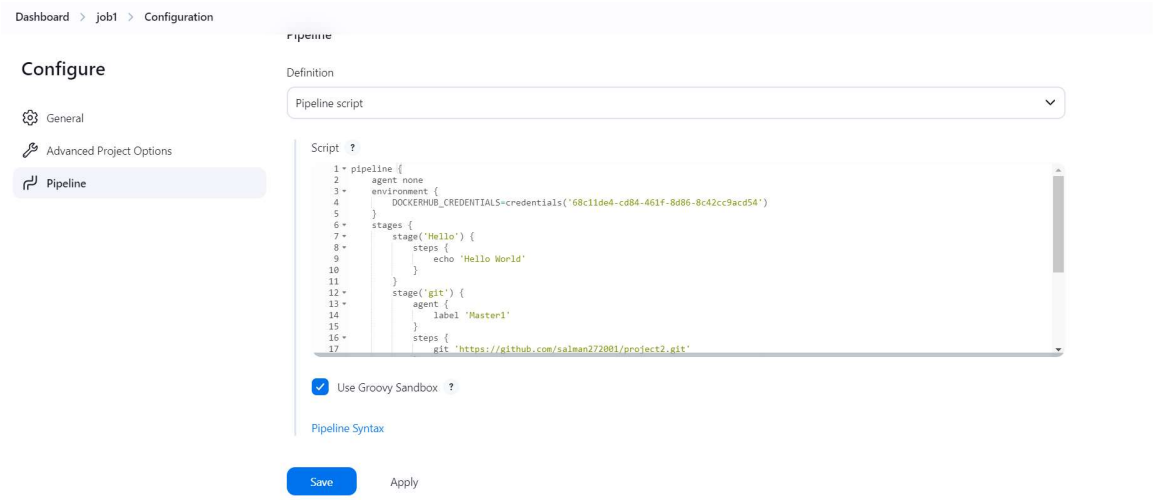

Jenkins
Search (CTRL+K)
Md salman
log out
Dashboard > Manage Jenkins > Nodes >
Nodes
Clouds
Build Queue
No builds in the queue.
Build Executor Status
Built-In Node 0/2
Master1 0/1
Nodes
+ New Node
Configure Monitors
S
Name ↓
Architecture
Clock Difference
Free Disk Space
Free Swap Space
Free Temp Space
Response Time
Built-In Node
Linux (amd64)
In sync
2.77 GiB
0 B
2.77 GiB
0ms
Master1
Linux (amd64)
In sync
2.27 GiB
0 B
2.27 GiB
59ms
Data obtained
37 min
37 min
37 min
37 min
37 min
37 min
Icon: S M L
Legend



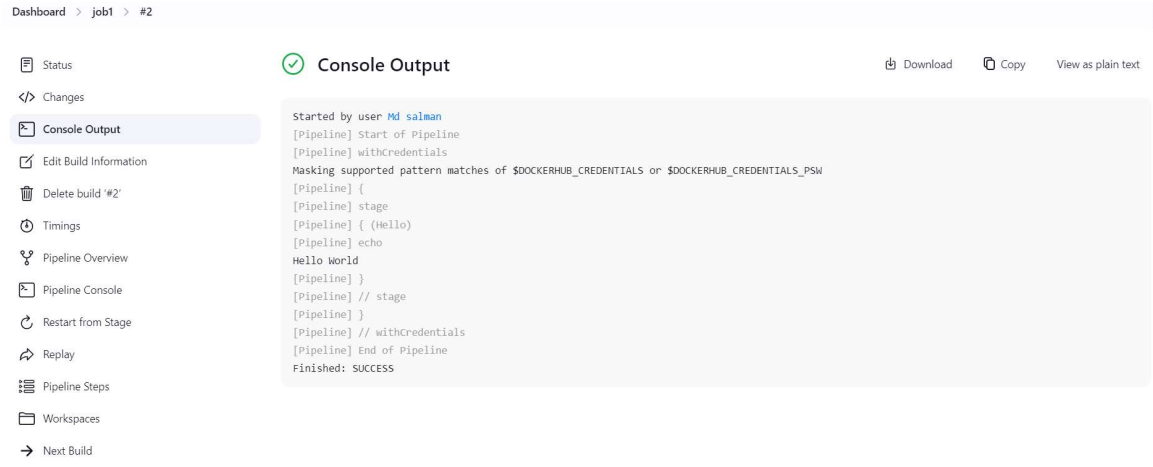
Step9: Added the docker credentials for pushing the image in Docker-Hub

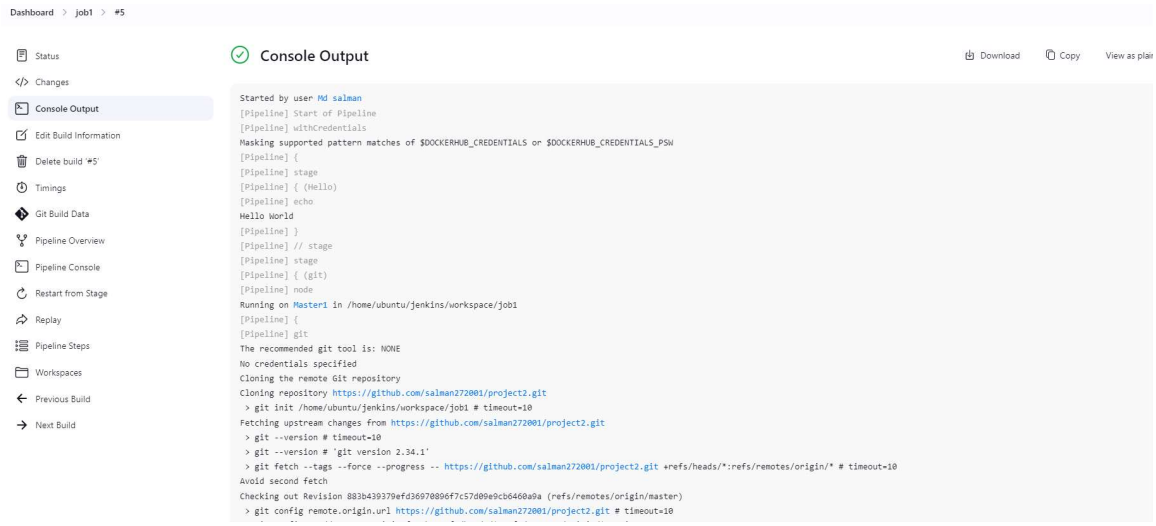


Step10:Pipeline creation builded the first stage (Hello world) in pipeline.

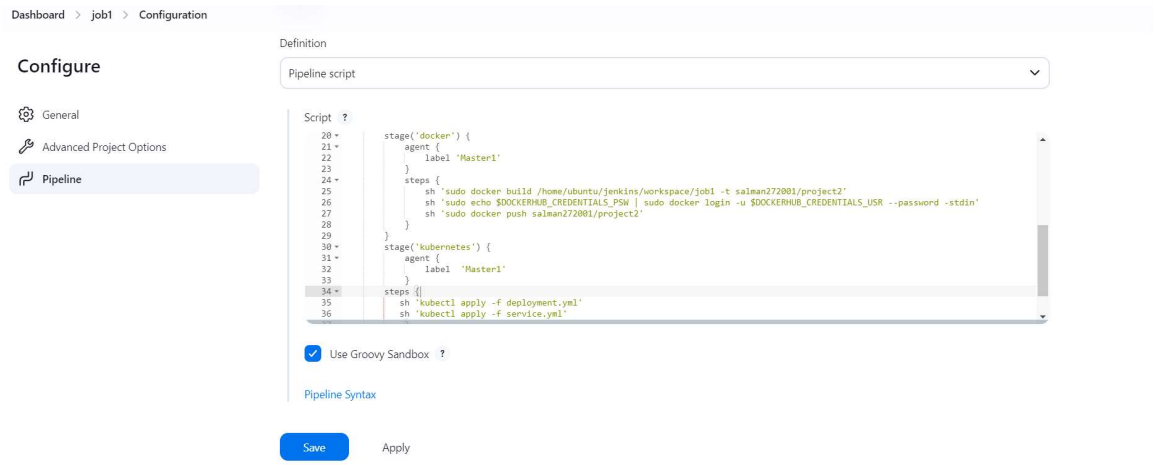


Created the Dockerfile in git-hub and it succesfully build.

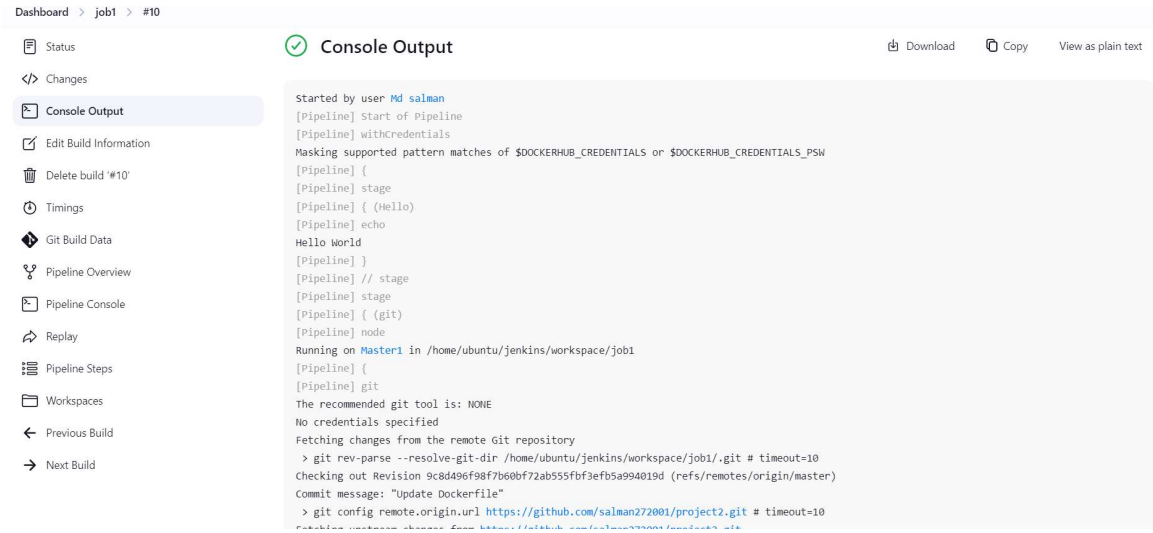




Step11:Now its time to Build and push the docker Image in Stage2 and we have validate the credentials in pipeline script check out below.

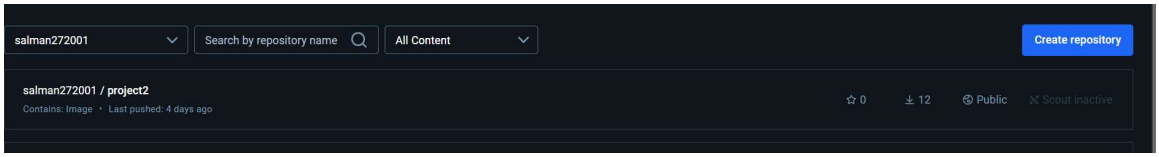


And it successfully build and pushed in Docker-hub.




```
+ sudo docker login -u salman272001 --password-stdin
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] sh
+ sudo docker push salman272001/project2
Using default tag: latest
The push refers to repository [docker.io/salman272001/project2]
5462c7bef063: Preparing
deb9dc140b41: Preparing
e5974be2b52b: Preparing
a30a5965a4f7: Preparing
a30a5965a4f7: Mounted from ubuntu/apache2
deb9dc140b41: Mounted from ubuntu/apache2
e5974be2b52b: Mounted from ubuntu/apache2
5462c7bef063: Pushed
latest: digest: sha256:44bae9f5aeee0b758b7267b221228be3bcd43a59465bcd804cec8221565834f4 size: 1158
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] End of Pipeline
Finished: SUCCESS
```



Step12:Now I am gonna create the deployment.yml with 2 replicas in order to display the image in other workers. And this our stage3 which is

kubernetes stage.

project2 / deployment.yml

salman272001 Create deployment.yml bd9687b

Code Blame 21 lines (21 loc) · 353 Bytes Code 55% faster with GitHub Copilot Raw

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: project2
        image: salman272001/project2
        ports:
        - containerPort: 80
```

And we,ll also create the service.yml for our node on port no.300008.

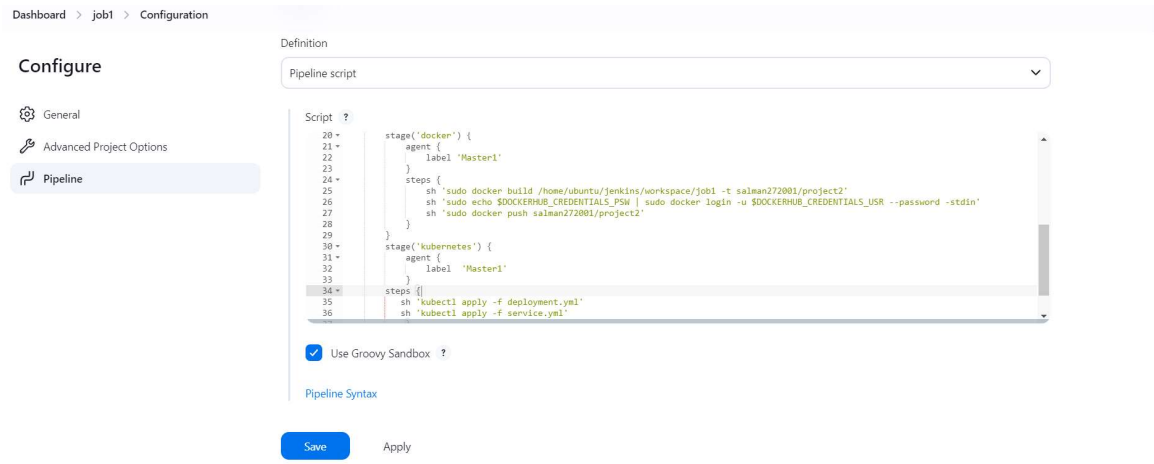
project2 / service.yml

salman272001 Update service.yml

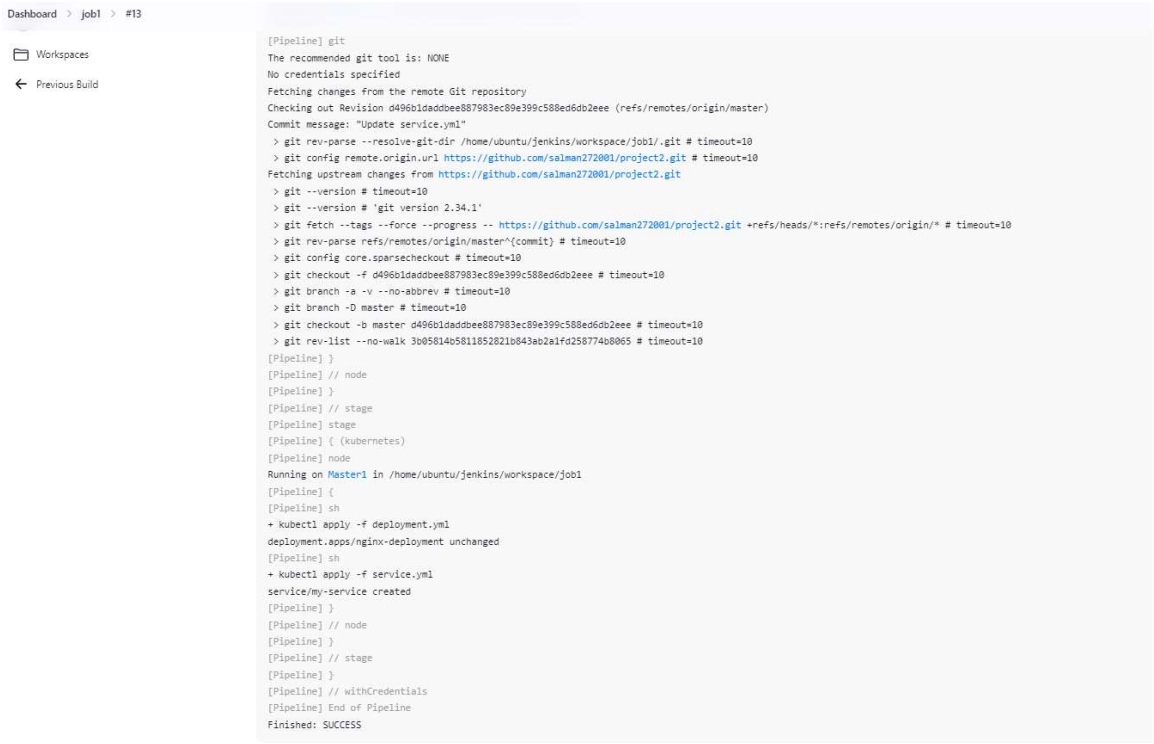
Code Blame 12 lines (12 loc) · 175 Bytes Code 55% faster with GitHub Copilot

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: nginx
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30008
```

Step13: Now its time to write kubernetes command in pipeline script. and click on build.



And as you can see our deployment and service.yml successfully ran.



And the image successfully displayed in all the workers except local master. Master1 which is Worker 2.

Slave1 which is worker3.

Slave2 which is worker4.


Not secure
3.143.68.233:30008
Gmail
Translate
Continue
CC6_Plan.xlsx - Goo...
All Bookmarks
Hello world!
GitHub