

Kubernetes-DAY1

Module Agenda:

1. Introduction to Kubernetes
2. Docker Swarm vs Kubernetes
3. Kubernetes Architecture
4. Kubernetes Installation
5. Working of Kubernetes
6. Deployments in Kubernetes
7. Services in Kubernetes
8. Ingress in Kubernetes
9. Kubernetes Dashboard

Note: Launch t2.Medium for Kubernetes, don't go with t.2 micro, t2.small, t2.nano,

```
root@Kubernetes-Master:~  X  root@Kubernetes-Worker:~  X  +  -  
root@Kubernetes-Master:~# kubeadm init --apiserver-advertise-address=172.31.24.209 --pod-network-cidr=10.244.0.0/16  
I0528 10:19:00.222771    4075 version.go:256] remote version is much newer: v1.30.1; falling back to: stable-1.28  
[init] Using Kubernetes version: v1.28.10  
[preflight] Running pre-flight checks  
error execution phase preflight: [preflight] Some fatal errors occurred:  
    [ERROR NumCPU]: the number of available CPUs 1 is less than the required 2  
    [ERROR Mem]: the system RAM (957 MB) is less than the minimum 1700 MB  
[preflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=...'  
To see the stack trace of this error execute with --v=5 or higher  
root@Kubernetes-Master:~#
```

And After Launching t2.medium Login and change hostname as your wish

the user switching from a regular user account to the root account with sudo su.

```
ubuntu@Kubernetes-Master:~  X  ubuntu@Kubernetes-worker:~  X  +  
ubuntu@Kubernetes-Master:~$ sudo su -
```

the user switching from a regular user account to the root account with sudo su.

```
ubuntu@Kubernetes-Master: ~ X
ubuntu@Kubernetes-worker: ~ X

ubuntu@Kubernetes-worker:~$ sudo su -
```

commands to install Kubernetes, a container orchestration platform, on the system

```
ubuntu@Kubernetes-Master:~$ sudo su -
root@Kubernetes-Master:~# ls
kubernetes_preq_install.sh snap
root@Kubernetes-Master:~# cat kubernetes_preq_install.sh
sudo apt update -y
sudo apt install curl apt-transport-https -y
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt install -y kubeadm=1.28.1-1.1 kubelet=1.28.1-1.1 kubectl=1.28.1-1.1
sudo apt update -y
sudo apt install kubelet kubeadm kubectl -y
sudo apt-mark hold kubelet kubeadm kubectl
sudo swapoff -a
sudo sed -i '/ swap / s/^(.*)$/#\1/g' /etc/fstab
free -m
sudo tee /etc/modules-load.d/k8s.conf <<EOF
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
sudo tee /etc/sysctl.d/kubernetes.conf<<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system
sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/docker-archive-keyring.gpg
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update -y
sudo apt install -y containerd.io
sudo mkdir -p /etc/containerd
sudo containerd config default|sudo tee /etc/containerd/config.toml
sudo sed -i 's/SystemdCgroup *= false/SystemdCgroup *= true/g' /etc/containerd/config.toml
sudo systemctl restart containerd
sudo systemctl enable containerd
systemctl status containerd
root@Kubernetes-Master:~# kubernetes_preq_install.sh
```

The script name is `kubernetes_preq_install.sh`. `chmod` is used to change the permissions of files and directories in Linux/Unix systems. IN MASTER MACHINE

```
root@Kubernetes-Master: ~ X
root@Kubernetes-worker: ~ X + ^

root@Kubernetes-Master:~# chmod +x kubernetes_preq_install.sh
```

commands to install Kubernetes, a container orchestration platform, on the system IN WORKERS MACHINE

```

ubuntu@Kubernetes-worker:~$ sudo su -
root@Kubernetes-worker:~# ls
kubernetes_preq_install.sh  snap
root@Kubernetes-worker:~# cat kubernetes_preq_install.sh
sudo apt update -y
sudo apt install curl apt-transport-https -y
curl -fsSL https://pkgs.k8s.io/core:/stable/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt install -y kubeadm=1.28.1-1.1 kubelet=1.28.1-1.1 kubectl=1.28.1-1.1
sudo apt update -y
sudo apt install kubelet kubeadm kubectl -y
sudo apt-mark hold kubelet kubeadm kubectl
sudo swapoff -a
sudo sed -i '/ swap / s/^.*$/#\!1/g' /etc/fstab
free -m
sudo tee /etc/modules-load.d/k8s.conf <<EOF
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
sudo tee /etc/sysctl.d/k8s.conf <<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system
sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/docker-archive-keyring.gpg
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update -y
sudo apt install -y containerd.io
sudo mkdir -p /etc/containerd
sudo containerd config default|sudo tee /etc/containerd/config.toml
sudo sed -i 's/SystemdCgroup *= false/SystemdCgroup *= true/g' /etc/containerd/config.toml
sudo systemctl restart containerd
sudo systemctl enable containerd
systemctl status containerd
root@Kubernetes-worker:~#

```

The script name is `kubernetes_preq_install.sh`. `chmod` is used to change the permissions of files and directories in Linux/Unix systems. IN WORKERS MACHINE

```

root@Kubernetes-Master: ~      X  root@Kubernetes-worker: ~      X  +  ▾
root@Kubernetes-worker:~# chmod +x kubernetes_preq_install.sh

```

trying to install Kubernetes, a container orchestration platform, IN MASTER MACHINE

```

root@Kubernetes-Master: ~      X  root@Kubernetes-worker: ~      X  +  ▾
root@Kubernetes-Master:~# bash kubernetes_preq_install.sh

```

trying to install Kubernetes, a container orchestration platform, IN WORKERS MACHINE

```

root@Kubernetes-Master: ~      X  root@Kubernetes-worker: ~      X  +  ▾
root@Kubernetes-worker:~# bash kubernetes_preq_install.sh

```

initialization of a Kubernetes cluster on a machine with IP address 172.31.23.111.

Kubernetes is a system for automating deployment, scaling, and management of containerized applications.

```

root@Kubernetes-Master: ~      X  root@Kubernetes-worker: ~      X  +  ▾
root@Kubernetes-Master:~# kubeadm init --apiserver-advertise-address=172.31.23.111 --pod-network-cidr=10.244.0.0/16

```

for joining a Kubernetes worker node to a cluster with the following details:

- API server address: 172.31.18.124:6443
- Discovery token: 610kcm.5emj9zn875dtw1v5
- Discovery token CA certificate hash:
sha256:289b550db765d4f8bb4279c693a28b6a4feebc887b027a23c5fef64c4ed8af6

AND COPY THE TOKEN IN Worker node which I didn't captured

```
kubeadm join 172.31.18.124:6443 --token 610kcm.5emj9zn875dtw1v5 \
--discovery-token-ca-cert-hash sha256:289b550db765d4f8bb4279c693a28b6a
4feebc887b027a23c5fef64c4ed8af6d
```

The three commands in the image set up a local Kubernetes cluster on your machine.

1. **Makes a directory:** Creates a directory named .kube in your home directory to store Kubernetes configuration files.
2. **Copies configuration file:** Copies the admin configuration file from the system directory to your .kube directory.
3. **Sets ownership:** Sets the ownership and permissions of the configuration file so only your user can access it.

To start using your cluster, you need to run the following as a regular user:

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

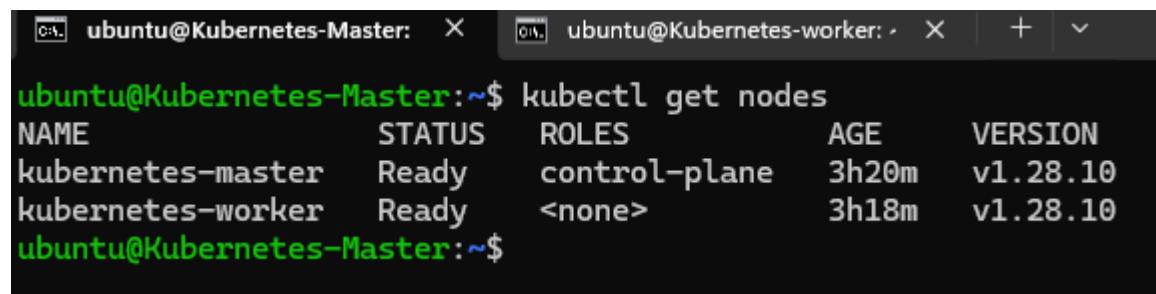
Alternatively, if you are the root user, you can run:

command applies a configuration file from GitHub to set up Flannel for your Kubernetes cluster in one line:

Flannel is a network fabric implementation for Kubernetes that provides pod networking without a dedicated overlay network.

```
ubuntu@kubernetesmaster:~$ mkdir -p $HOME/.kube
ubuntu@kubernetesmaster:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@kubernetesmaster:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@kubernetesmaster:~$ kubectl apply -f https://raw.githubusercontent.com/
flannel-io/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
ubuntu@kubernetesmaster:~$
```

kubectl get nodes to list the nodes (computers) in a Kubernetes cluster and their current status.



```
ubuntu@Kubernetes-Master:~$ kubectl get nodes
NAME           STATUS   ROLES      AGE    VERSION
kubernetes-master   Ready    control-plane   3h20m   v1.28.10
kubernetes-worker   Ready    <none>        3h18m   v1.28.10
ubuntu@Kubernetes-Master:~$
```

Kubernetes-DAY2

- `kubectl`: This is the command-line tool for interacting with Kubernetes clusters.
- `apply`: This subcommand tells `kubectl` to create or update resources in your cluster.
- `-f`: This flag specifies that the following argument (`pod.yml`) is a file containing the resource definition.
- `pod.yml`: This is the YAML file that defines the pod configuration. It includes details like container image, ports, storage, etc.

The command `kubectl get pods` is used to retrieve information about the pods in a Kubernetes cluster

```
ubuntu@Kubernetes-Master: ~$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80

ubuntu@Kubernetes-Master: ~$ kubectl apply -f pod.yml
pod/nginx-pod unchanged
ubuntu@Kubernetes-Master: ~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
nginx-pod  1/1     Running   0          21m
ubuntu@Kubernetes-Master: ~$
```

`kubectl describe pod nginx-pod` provides detailed information about a specific pod in your Kubernetes cluster named `nginx-pod`.

By running this command, you'll get a comprehensive view of the `nginx-pod`, including its:

- Status (Running, Pending, etc.)
- IP address
- Container details (image, ports, etc.)
- Events related to the pod
- Configuration details

- And more

```
ubuntu@Kubernetes-Master:~$ kubectl describe pod nginx-pod
Name:           nginx-pod
Namespace:      default
Priority:      0
Service Account: default
Node:          kubernetes-worker/172.31.53.219
Start Time:    Wed, 29 May 2024 09:36:37 +0000
Labels:         <none>
Annotations:   <none>
Status:        Running
IP:            10.244.1.2
IPs:
  IP:  10.244.1.2
```

Events:				
Type	Reason	Age	From	Message
Normal	Scheduled	27m	default-scheduler	Successfully assigned default/nginx-pod to kubernetes-worker
Normal	Pulling	27m	kubelet	Pulling image "nginx"
Normal	Pulled	27m	kubelet	Successfully pulled image "nginx" in 4.027s (4.027s including waiting)
Normal	Created	27m	kubelet	Created container nginx-container
Normal	Started	27m	kubelet	Started container nginx-container

The command `kubectl logs nginx-pod` displays the logs generated by the containers running within the pod named `nginx-pod`

```
ubuntu@Kubernetes-Master:~$ kubectl logs nginx-pod
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/05/29 09:36:41 [notice] 1#1: using the "epoll" event method
2024/05/29 09:36:41 [notice] 1#1: nginx/1.25.5
2024/05/29 09:36:41 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/05/29 09:36:41 [notice] 1#1: OS: Linux 6.8.0-1008-aws
2024/05/29 09:36:41 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/05/29 09:36:41 [notice] 1#1: start worker processes
2024/05/29 09:36:41 [notice] 1#1: start worker process 30
2024/05/29 09:36:41 [notice] 1#1: start worker process 31
ubuntu@Kubernetes-Master:~$
```

`kubectl exec -it pod/nginx-pod -- /bin/bash`. This command allows you to establish an interactive shell session directly with a specific container inside a pod in your Kubernetes cluster

```
ubuntu@Kubernetes-Master:~$ kubectl exec -it pod/nginx-pod -- /bin/bash
root@nginx-pod:/# ls
bin  dev  docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot  docker-entrypoint.d  etc  lib  media  opt  root  sbin  sys  usr
root@nginx-pod:/#
```

Kubectl delete pod/nginx-pod this command delete the pod

```
ubuntu@Kubernetes-Master:~$ kubectl get pod
NAME      READY   STATUS    RESTARTS   AGE
nginx-pod  1/1     Running   0          30m
ubuntu@Kubernetes-Master:~$ kubectl delete pod/nginx-pod
pod "nginx-pod" deleted
ubuntu@Kubernetes-Master:~$ kubectl apply -f pod.yml
pod/nginx-pod created
ubuntu@Kubernetes-Master:~$ kubectl get pod
NAME      READY   STATUS    RESTARTS   AGE
nginx-pod  1/1     Running   0          9s
```

`kubectl get pod -o wide` retrieves information about pods in your Kubernetes cluster and displays it in a wider format with additional details.

- **Node:** The name of the Kubernetes node where the pod is running.
- **NOMINATED NODE:** The node the pod is nominated to run on (if scheduling is not yet complete).
- **READINESS GATES:** The readiness gates of the pod (if any).
- **AGE:** The age of the pod.
- **IP:** The IP address of the pod.

```
ubuntu@Kubernetes-Master:~$ kubectl get pod -o wide
NAME      READY   STATUS    RESTARTS   AGE   IP           NODE        NOMINATED-NODE   R
EADINESS-GATES
nginx-pod  1/1     Running   0          52s  10.244.1.3  kubernetes-worker  <none>       <
none>
ubuntu@Kubernetes-Master:~$ |
```

```
ubuntu@Kubernetes-Master:~$ kubectl get nodes
NAME      STATUS   ROLES      AGE   VERSION
kubernetes-master  Ready    control-plane  3h34m  v1.28.10
kubernetes-worker  Ready    <none>      3h32m  v1.28.10
ubuntu@Kubernetes-Master:~$ kubectl get nodes -o wide
NAME      STATUS   ROLES      AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE
KERNEL-VERSION CONTAINER-RUNTIME
kubernetes-master  Ready    control-plane  3h34m  v1.28.10  172.31.62.130  <none>       Ubuntu 24.04 LTS
6.8.0-1008-aws   containerd://1.6.32
kubernetes-worker  Ready    <none>      3h33m  v1.28.10  172.31.53.219  <none>       Ubuntu 24.04 LTS
6.8.0-1008-aws   containerd://1.6.32
ubuntu@Kubernetes-Master:~$
```

A ReplicaSet in Kubernetes ensures a specified number of identical pods are running at all times.

```
ubuntu@Kubernetes-Master: ~$ nano replicaset.yml
```

• **replicas: 3:** This sets the number of replicas (identical pods) that the ReplicaSet will manage.

• **app: nginx:** This defines a label selector that will match pods with the label `app: nginx`. The ReplicaSet will manage any pods that match this selector.

• **nginx-container:** This defines the container template that will be used to create the pods. The container image is set to `nginx` and the container port is set to `80`.

```
GNU nano 7.2                                                 replicaset.y
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: example-replicaset
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx
          ports:
            - containerPort: 80
```

We Given 3 Replicaset so it will run 3 Pods

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f replicaset.yml
replicaset.apps/example-replicaset created
ubuntu@Kubernetes-Master:~$ kubectl get pod
NAME                  READY   STATUS    RESTARTS   AGE
example-replicaset-jvskb 1/1     Running   0          19s
example-replicaset-m9q8n  1/1     Running   0          19s
example-replicaset-njt76  1/1     Running   0          19s
nginx-pod              1/1     Running   0          23m
```

We Delete one Pod and automatically Kubernetes will its created another one

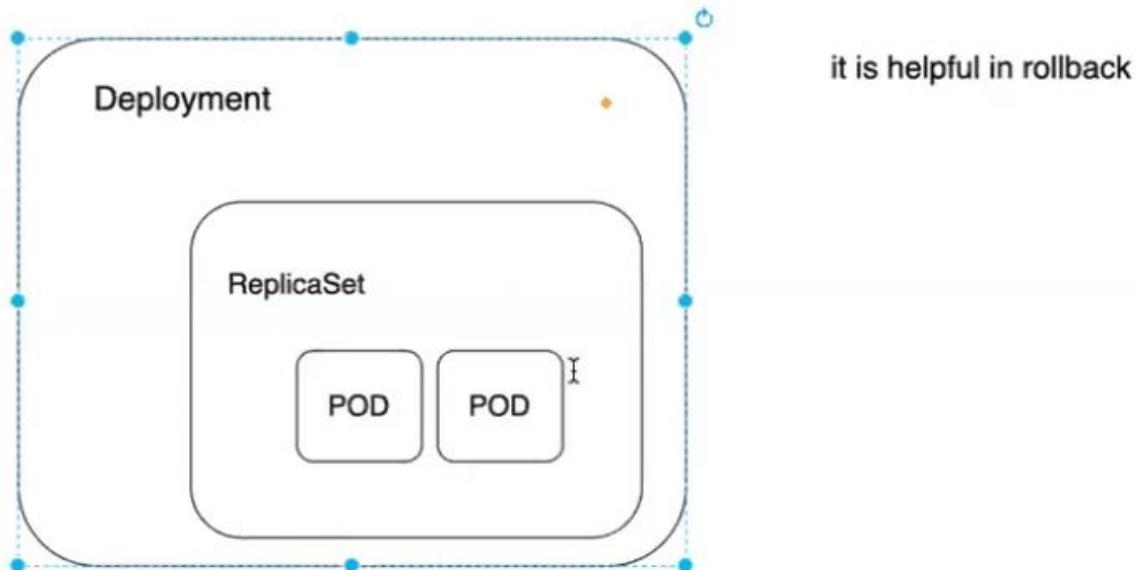
```
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
example-replicaset-jvskb 1/1     Running   0          3m19s
example-replicaset-m9q8n  1/1     Running   0          3m19s
example-replicaset-njt76  1/1     Running   0          3m19s
nginx-pod              1/1     Running   0          26m
ubuntu@Kubernetes-Master:~$ kubectl delete pod/example-replicaset-jvskb
pod "example-replicaset-jvskb" deleted
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
example-replicaset-6md99 1/1     Running   0          12s
example-replicaset-m9q8n  1/1     Running   0          3m50s
example-replicaset-njt76  1/1     Running   0          3m50s
nginx-pod              1/1     Running   0          27m
ubuntu@Kubernetes-Master:~$
```

REPLICASET

SINGLE POD

```
ubuntu@Kubernetes-Master: ~ X  ubuntu@Kubernetes-worker: ~ X  ubuntu@Kubernetes-Master: ~ X  ubuntu@Kubernetes-Master: ~ X
GNU nano 7.2                               replicaset.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: example-replicaset
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx
          ports:
            - containerPort: 80
ubuntu@Kubernetes-Master:~$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
```

Kubernetes-DAY3



1. Launch Kubernetes- Master and Kubernetes- Slave

```
ubuntu@Kubernetes-Master: ~$
```

2. Now We Copied Replicaset script to deployment.yml

```
ubuntu@Kubernetes-Master: ~$ ls
pod.yml replicaset.yml
ubuntu@Kubernetes-Master: ~$ cp replicaset.yml deployment.yml
ubuntu@Kubernetes-Master: ~$ nano deployment.yml
```

3. wrote the script

```
GNU nano 7.2                                                 deployment.yml *
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx
          ports:
            - containerPort: 80
```

4. kubectl apply -f deployment.yml and its created. And 3 nginx deployment has created

```
ubuntu@Kubernetes-Master:~$ ls
pod.yml  replicaset.yml
ubuntu@Kubernetes-Master:~$ cp replicaset.yml deployment.yml
ubuntu@Kubernetes-Master:~$ nano deployment.yml
ubuntu@Kubernetes-Master:~$ nano deployment.yml
ubuntu@Kubernetes-Master:~$ kubectl apply -f deployment.yml
deployment.apps/nginx-deployment created
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME                           READY   STATUS    RESTARTS   AGE
example-replicaset-6md99       1/1     Running   1 (6m28s ago) 23h
example-replicaset-m9q8n       1/1     Running   1 (6m28s ago) 23h
example-replicaset-njt76       1/1     Running   1 (6m28s ago) 23h
nginx-deployment-75f5f7957b-8b69f  1/1     Running   0          9s
nginx-deployment-75f5f7957b-bs2c5  1/1     Running   0          9s
nginx-deployment-75f5f7957b-nmt66  1/1     Running   0          9s
nginx-pod                       1/1     Running   1 (6m28s ago) 23h
ubuntu@Kubernetes-Master:~$ kubectl get deployment
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment  3/3     3           3           35s
ubuntu@Kubernetes-Master:~$
```

5. Deleting all rs/example-replicaset and deleting one pod/nginx-pod and only deployment is available and I am deleting one deployment and after that check pods replica created another deployment server because replicas is inside the deployment.

```
ubuntu@Kubernetes-Master:~$ kubectl delete rs/example-replicaset
replicaset.apps "example-replicaset" deleted
ubuntu@Kubernetes-Master:~$ kubectl delete pod/nginx-pod
pod "nginx-pod" deleted
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
nginx-deployment-75f5f7957b-8b69f 1/1 Running 0 3m45s
nginx-deployment-75f5f7957b-bs2c5 1/1 Running 0 3m45s
nginx-deployment-75f5f7957b-nmt66 1/1 Running 0 3m45s
ubuntu@Kubernetes-Master:~$ kubectl delete pod/nginx-deployment-75f5f7957b-8b69f
pod "nginx-deployment-75f5f7957b-8b69f" deleted
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
nginx-deployment-75f5f7957b-bs2c5 1/1 Running 0 4m35s
nginx-deployment-75f5f7957b-nmt66 1/1 Running 0 4m35s
nginx-deployment-75f5f7957b-rqbnh 1/1 Running 0 9s
ubuntu@Kubernetes-Master:~$
```

6. Here we scaled the only 1 replicas. We can see only one deployment server is available

```
ubuntu@Kubernetes-Master:~$ kubectl scale deploy/nginx-deployment --replicas=1
deployment.apps/nginx-deployment scaled
ubuntu@Kubernetes-Master:~$ kubectl gets pods
error: unknown command "gets" for "kubectl"

Did you mean this?
  set
  get
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
nginx-deployment-75f5f7957b-nmt66 1/1 Running 0 7m13s
ubuntu@Kubernetes-Master:~$
```

7. We can see inside the deployment.yml there is 3 replicas.

```
ubuntu@Kubernetes-Master:~$ cat deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
  spec:
    containers:
      - name: nginx-container
        image: nginx
        ports:
          - containerPort: 80
ubuntu@Kubernetes-Master:~$
```

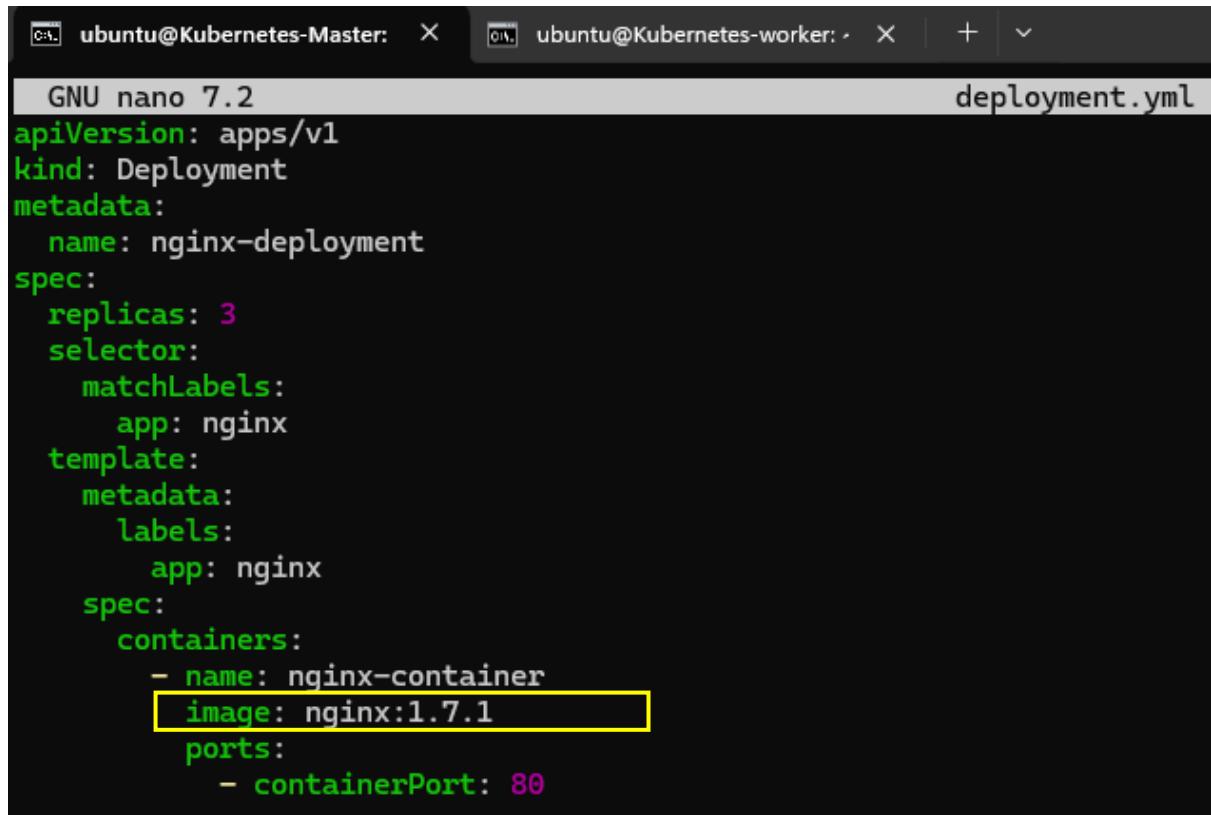
8. Now we scaled 2 replicas

```
ubuntu@Kubernetes-Master:~$ kubectl scale deploy/nginx-deployment --replicas=2
deployment.apps/nginx-deployment scaled
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
nginx-deployment-75f5f7957b-5gn59   1/1     Running   0          7s
nginx-deployment-75f5f7957b-nmt66   1/1     Running   0          9m27s
ubuntu@Kubernetes-Master:~$
```

9. Now Again Edit the deployment.yml

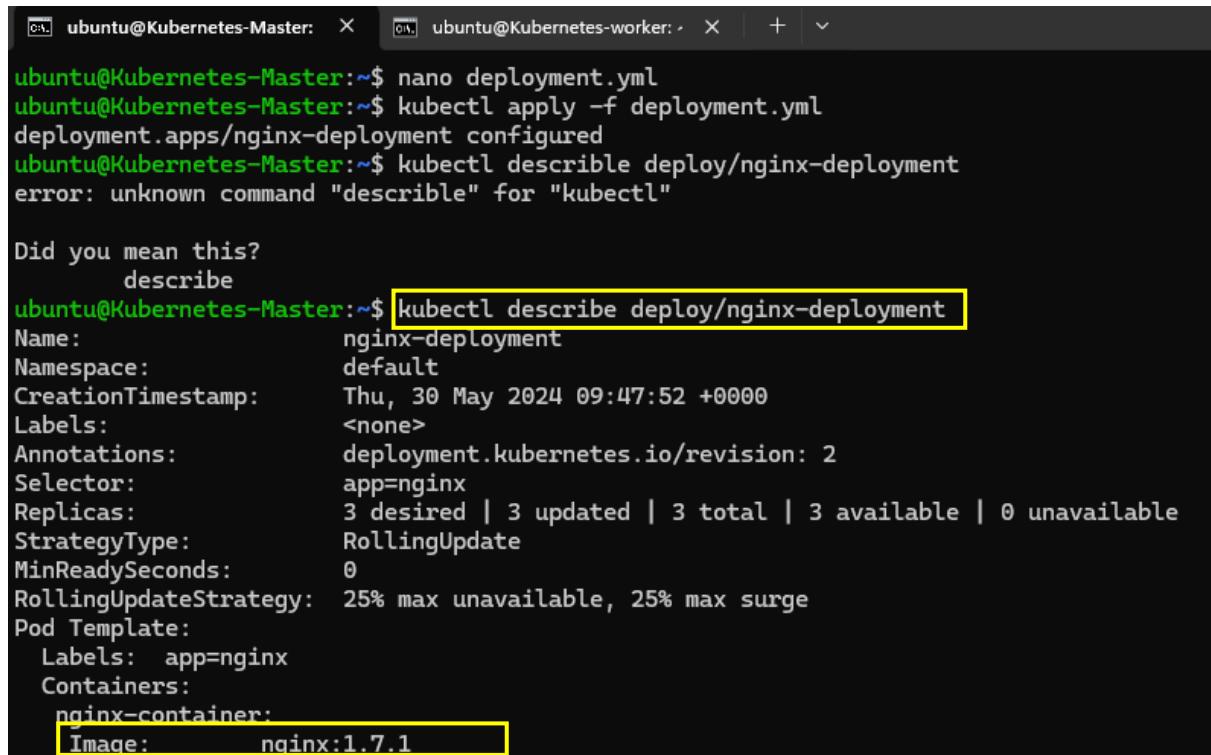
```
ubuntu@Kubernetes-Master:~$ nano deployment.yml
```

10. This time we are Modifying image nginx with version 1.7.1



```
GNU nano 7.2 deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx:1.7.1
        ports:
          - containerPort: 80
```

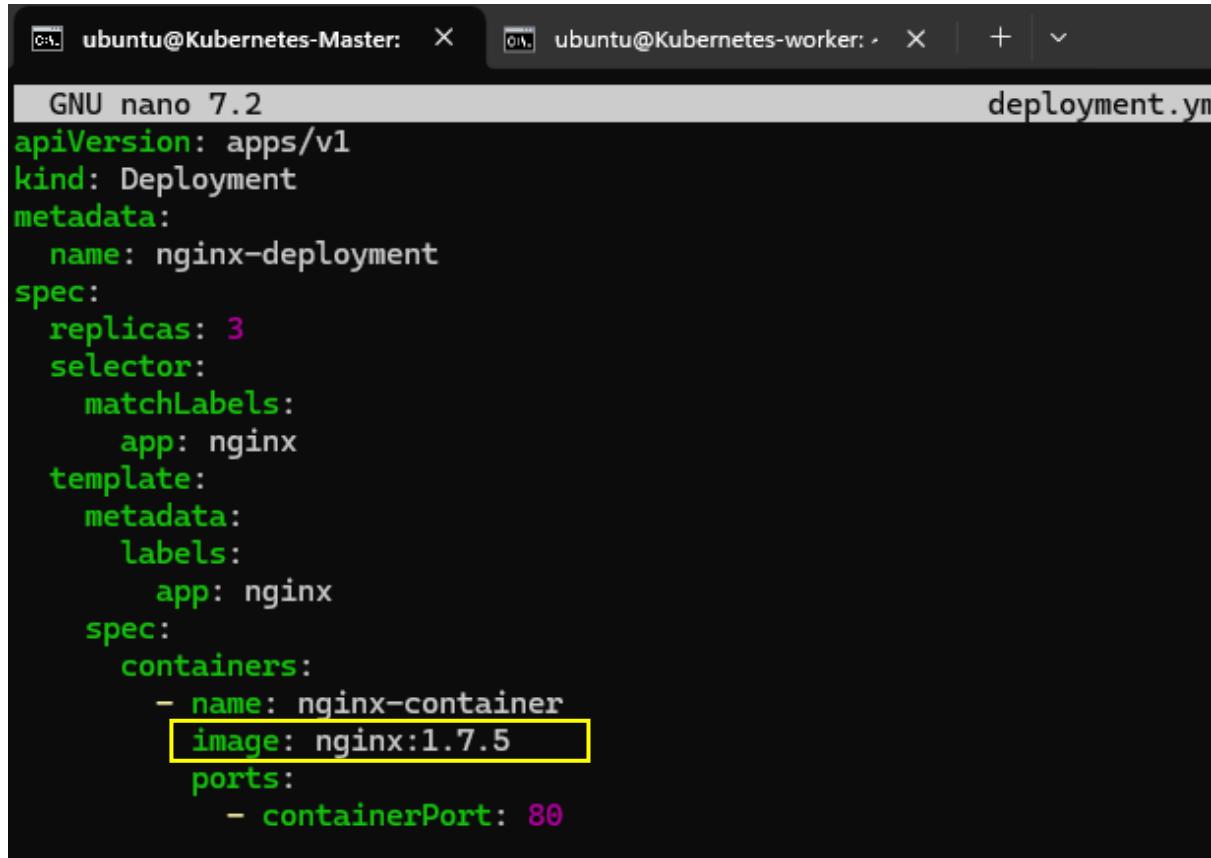
11. Apply and Describe it and we can see the image version is nginx:1.7.1



```
ubuntu@Kubernetes-Master:~$ nano deployment.yml
ubuntu@Kubernetes-Master:~$ kubectl apply -f deployment.yml
deployment.apps/nginx-deployment configured
ubuntu@Kubernetes-Master:~$ kubectl describle deploy/nginx-deployment
error: unknown command "describle" for "kubectl"

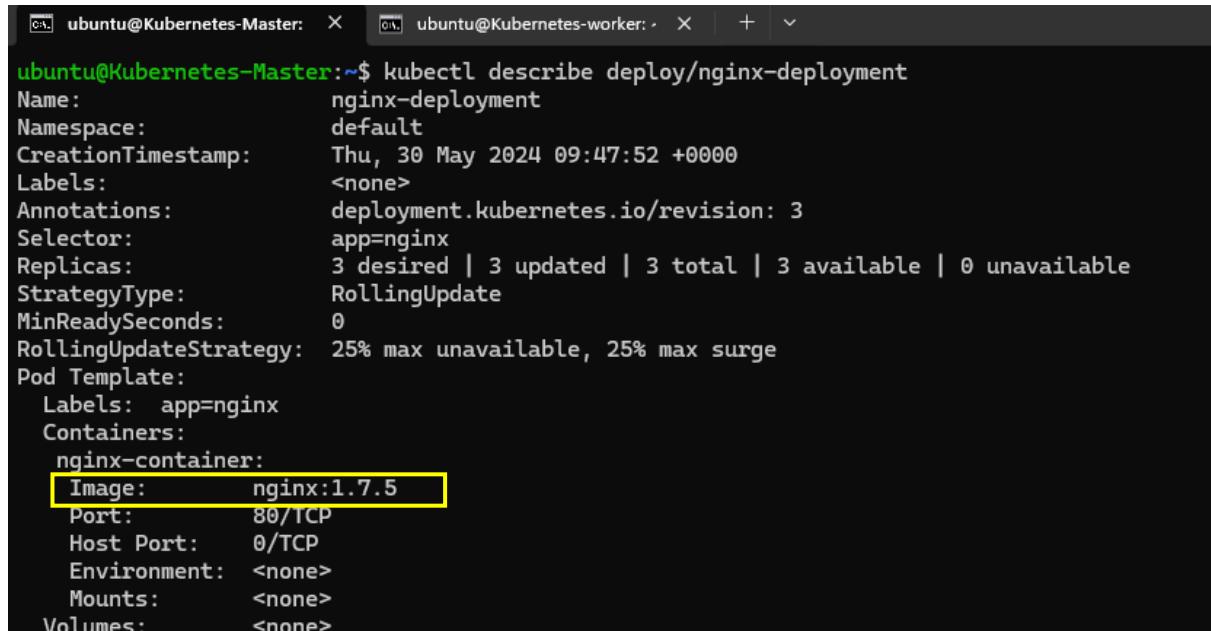
Did you mean this?
  describe
ubuntu@Kubernetes-Master:~$ kubectl describe deploy/nginx-deployment
Name:           nginx-deployment
Namespace:      default
CreationTimestamp: Thu, 30 May 2024 09:47:52 +0000
Labels:          <none>
Annotations:    deployment.kubernetes.io/revision: 2
Selector:        app=nginx
Replicas:       3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:   RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
    nginx-container:
      Image:      nginx:1.7.1
```

12. Again Modified



```
GNU nano 7.2 deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx:1.7.5
          ports:
            - containerPort: 80
```

13. Now we can see the updated one



```
ubuntu@Kubernetes-Master:~$ kubectl describe deploy/nginx-deployment
Name:           nginx-deployment
Namespace:      default
CreationTimestamp: Thu, 30 May 2024 09:47:52 +0000
Labels:          <none>
Annotations:    deployment.kubernetes.io/revision: 3
Selector:        app=nginx
Replicas:       3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:   RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
    nginx-container:
      Image:      nginx:1.7.5
      Port:       80/TCP
      Host Port:  0/TCP
      Environment: <none>
      Mounts:     <none>
      Volumes:    <none>
```

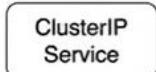
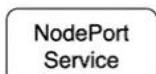
14. Rollout means it will go back and fetch the previous version

```
ubuntu@Kubernetes-Master:~$ kubectl rollout undo deployment/nginx-deployment
deployment.apps/nginx-deployment rolled back
ubuntu@Kubernetes-Master:~$ kubectl describe deploy/nginx-deployment
Name:           nginx-deployment
Namespace:      default
CreationTimestamp: Thu, 30 May 2024 09:47:52 +0000
Labels:          <none>
Annotations:    deployment.kubernetes.io/revision: 4
Selector:        app=nginx
Replicas:       3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:   RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
    nginx-container:
      Image:  nginx:1.7.1
```

NodePort: Makes the service accessible on all nodes through a single port, exposing it externally via firewall configuration on the cluster's nodes.

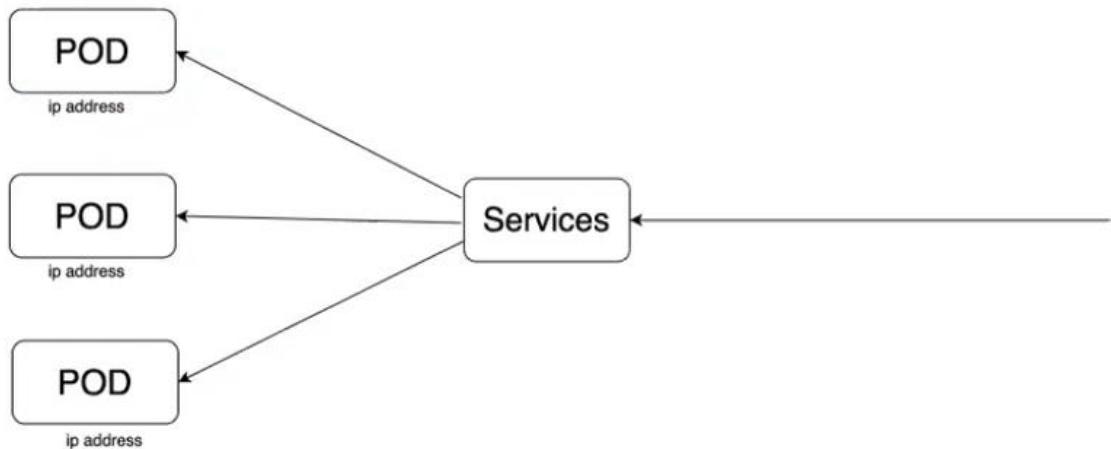
ClusterIP: Creates a service accessible only from within the Kubernetes cluster using a cluster-internal IP address.

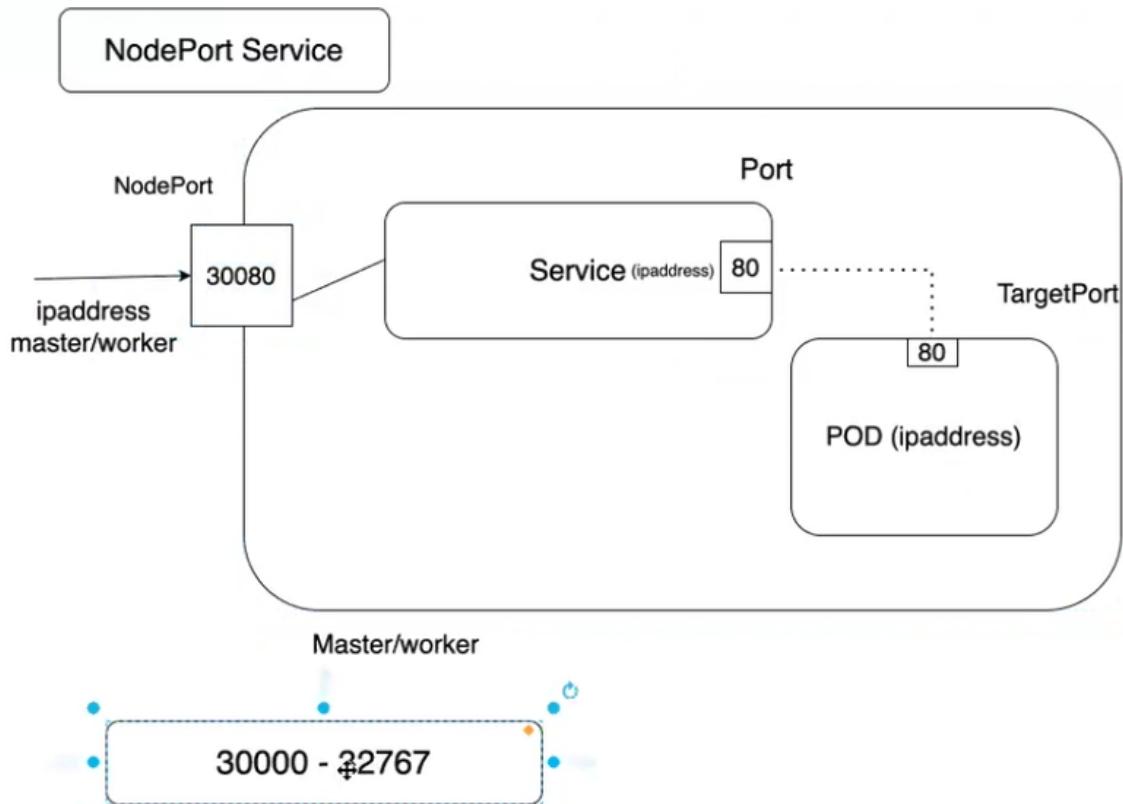
LoadBalancer: Exposes the service externally using a cloud provider's load balancer, typically with a dynamic IP address.



15. `kubectl get pods -o wide`(Displays detailed information about all pods in the current Kubernetes namespace.)

```
ubuntu@Kubernetes-Master:~$ kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE     IP           NODE      NOMINATED NODE   READINESS GAGE
nginx-deployment-64dc447c-dzv6z  1/1    Running   0          6m22s  10.244.1.26  kubernetes-worker  <none>        <none>
nginx-deployment-64dc447c-hslnf  1/1    Running   0          6m26s  10.244.1.24  kubernetes-worker  <none>        <none>
nginx-deployment-64dc447c-tlxr4  1/1    Running   0          6m24s  10.244.1.25  kubernetes-worker  <none>        <none>
ubuntu@Kubernetes-Master:~$ curl 10.244.1.26:80
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
ubuntu@Kubernetes-Master:~$
```





16. Now Creating nodeport.yml

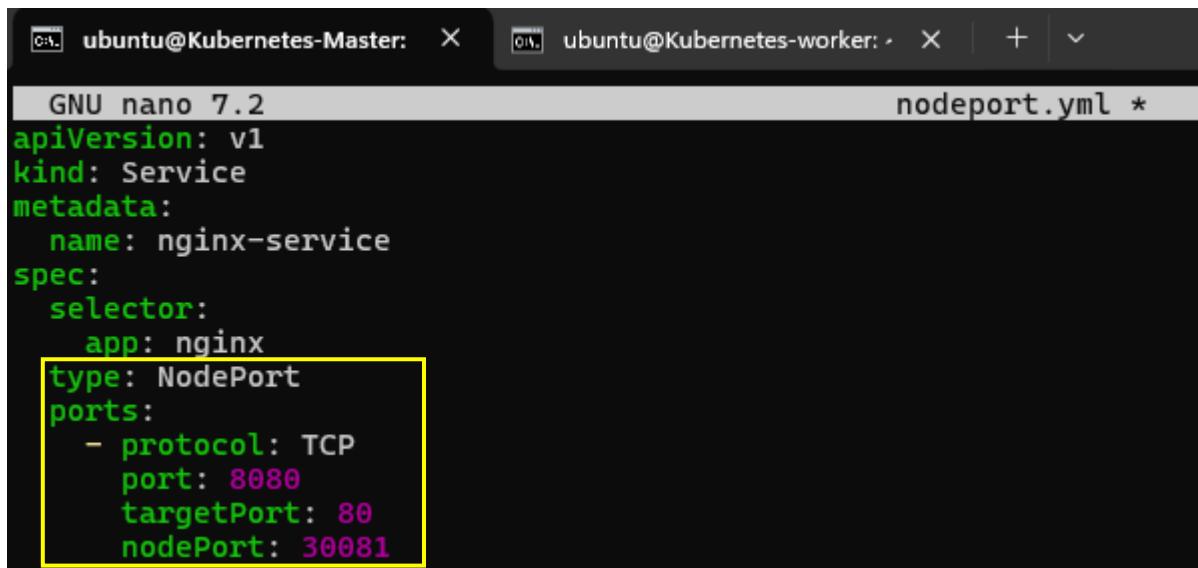
NodePort: Makes the service accessible on all nodes through a single port, exposing it externally via firewall configuration on the cluster's nodes.

```
ubuntu@Kubernetes-Master:~$ nano nodeport.yml
ubuntu@Kubernetes-Master:~$
```

17. Copy the data from deployment.yml and edit it

```
ubuntu@Kubernetes-Master:~$ cat deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-container
          image: nginx:1.7.5
          ports:
            - containerPort: 80
```

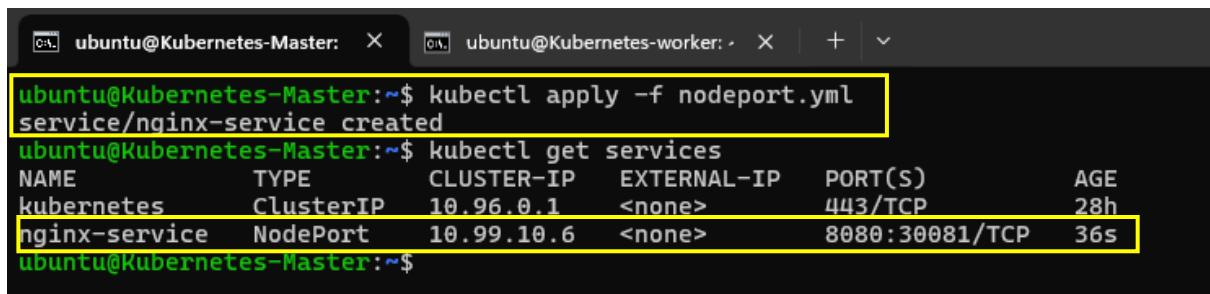
18. observe in the ports part



```
GNU nano 7.2                                     nodeport.yml *
```

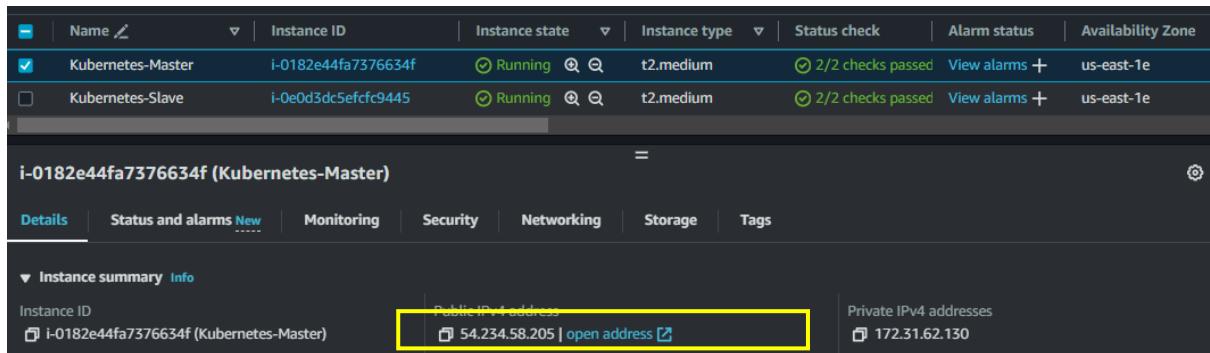
```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app: nginx
  type: NodePort
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
      nodePort: 30081
```

19. Apply the nodeport.yml and its created and launched the nginx server in the browser and port no 8080 and nodeport 30081



```
ubuntu@Kubernetes-Master:~$ kubectl apply -f nodeport.yml
service/nginx-service created
ubuntu@Kubernetes-Master:~$ kubectl get services
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes     ClusterIP  10.96.0.1   <none>        443/TCP         28h
nginx-service   NodePort   10.99.10.6  <none>        8080:30081/TCP  36s
ubuntu@Kubernetes-Master:~$
```

20. Copy the Kubernetes-Master Public-IP



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Kubernetes-Master	i-0182e44fa7376634f	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1e
Kubernetes-Slave	i-0e0d3dc5efcf9445	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1e

i-0182e44fa7376634f (Kubernetes-Master)

Details Status and alarms New Monitoring Security Networking Storage Tags

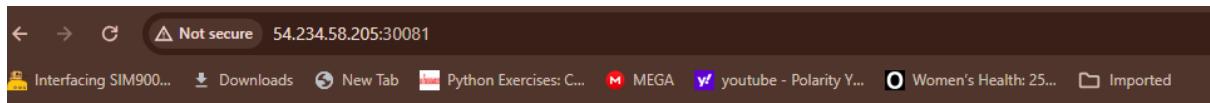
Instance summary Info

Instance ID: i-0182e44fa7376634f (Kubernetes-Master)

Public IPv4 addresses: 54.234.58.205 | open address

Private IPv4 addresses: 172.31.62.130

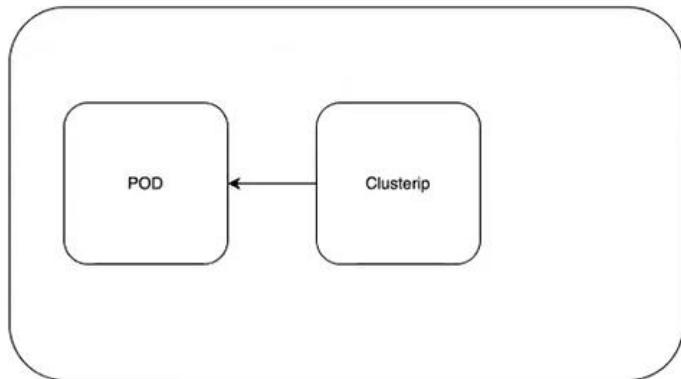
21. We can see the default nginx page



ClusterIP: Creates a service accessible only from within the Kubernetes cluster using a cluster-internal IP address.

ClusterIP Service

If communication is needed inside kubernetes cluster



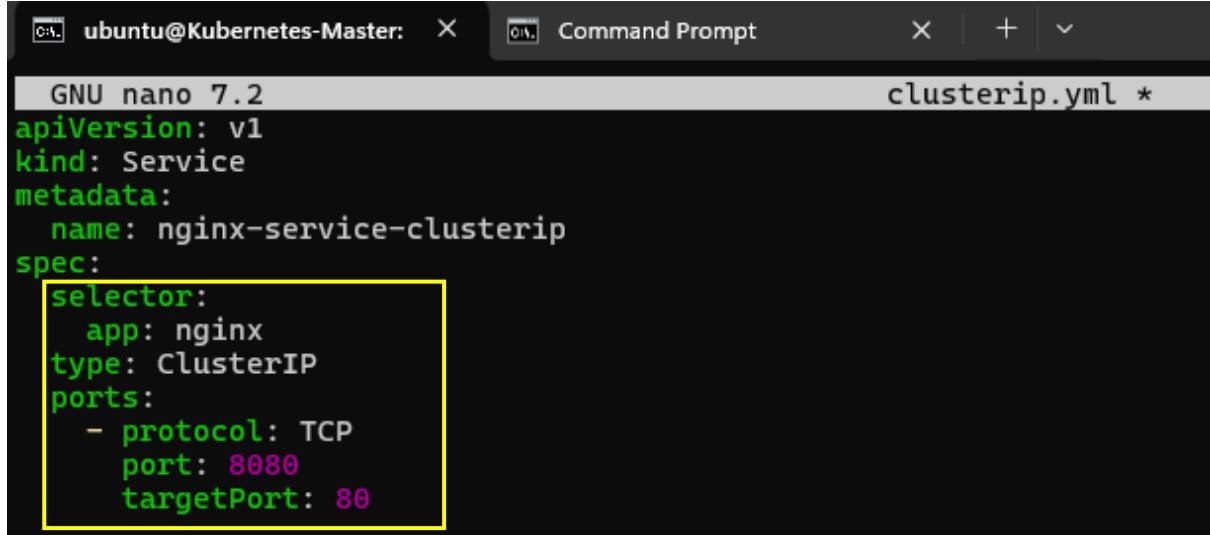
22. Creating Clusterip.yml

```
ubuntu@Kubernetes-Master: ~$ nano clusterip.yml
```

The screenshot shows a terminal window with the following details:

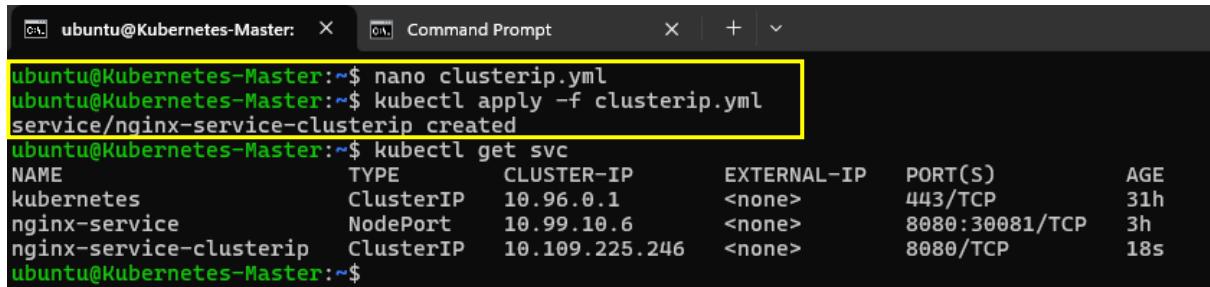
- Title bar: ubuntu@Kubernetes-Master: ~\$ Command Prompt
- Content area:
 - Text: "ubuntu@Kubernetes-Master: ~\$ nano clusterip.yml"

23. in the selector this time clusterIP and there is no clusteripport like nodeport this is only for internal connection.



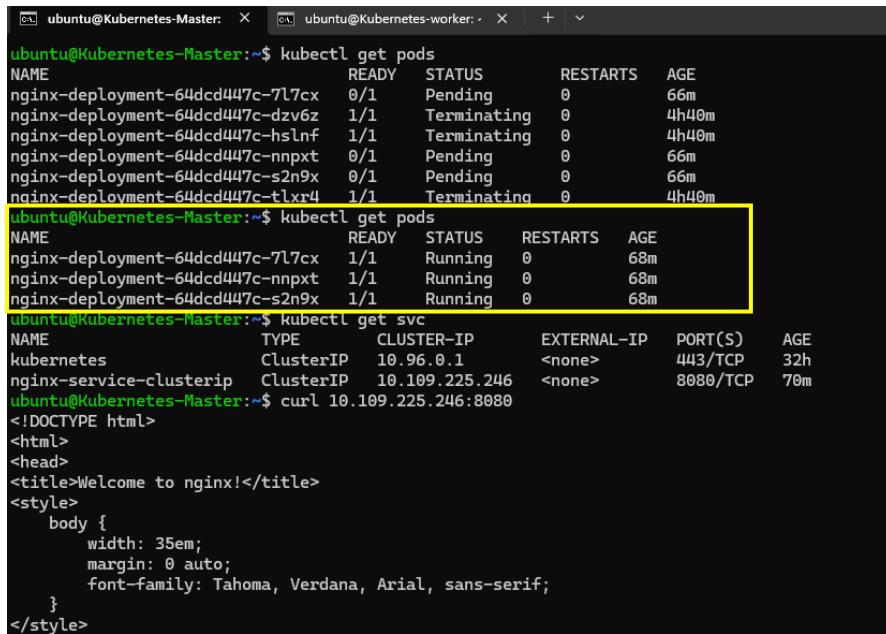
```
GNU nano 7.2                                         clusterip.yml *
apiVersion: v1
kind: Service
metadata:
  name: nginx-service-clusterip
spec:
  selector:
    app: nginx
  type: ClusterIP
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
```

24. We can see the nginx-service-clusterip



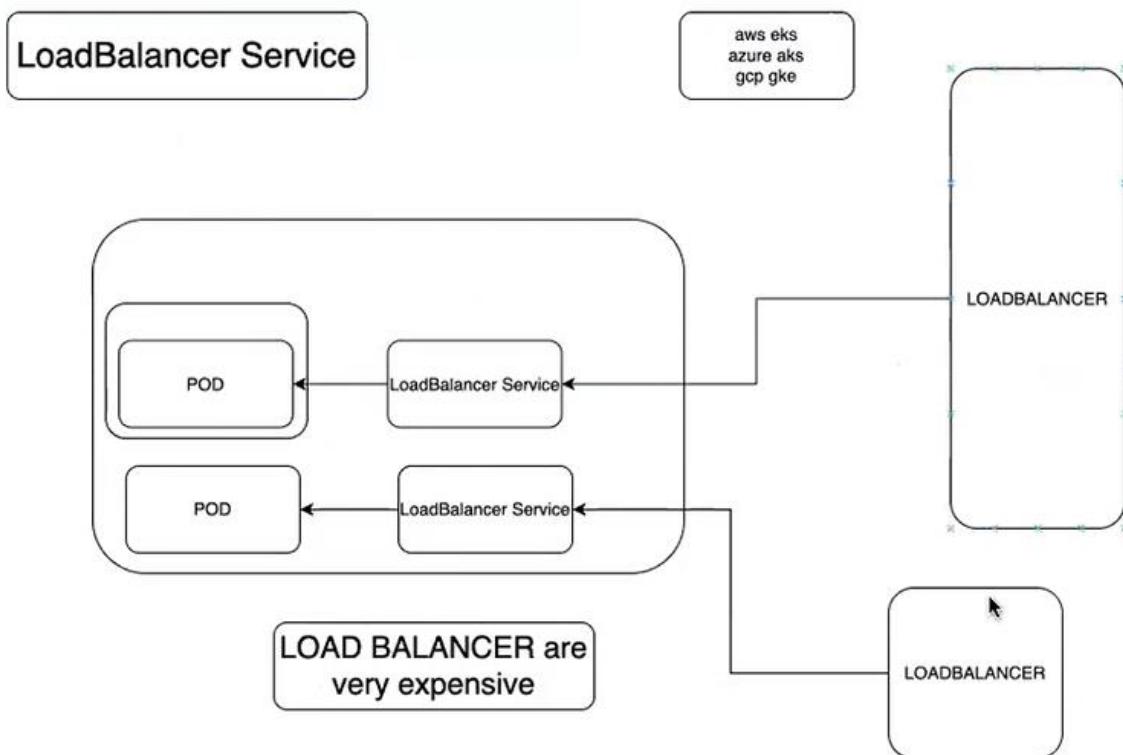
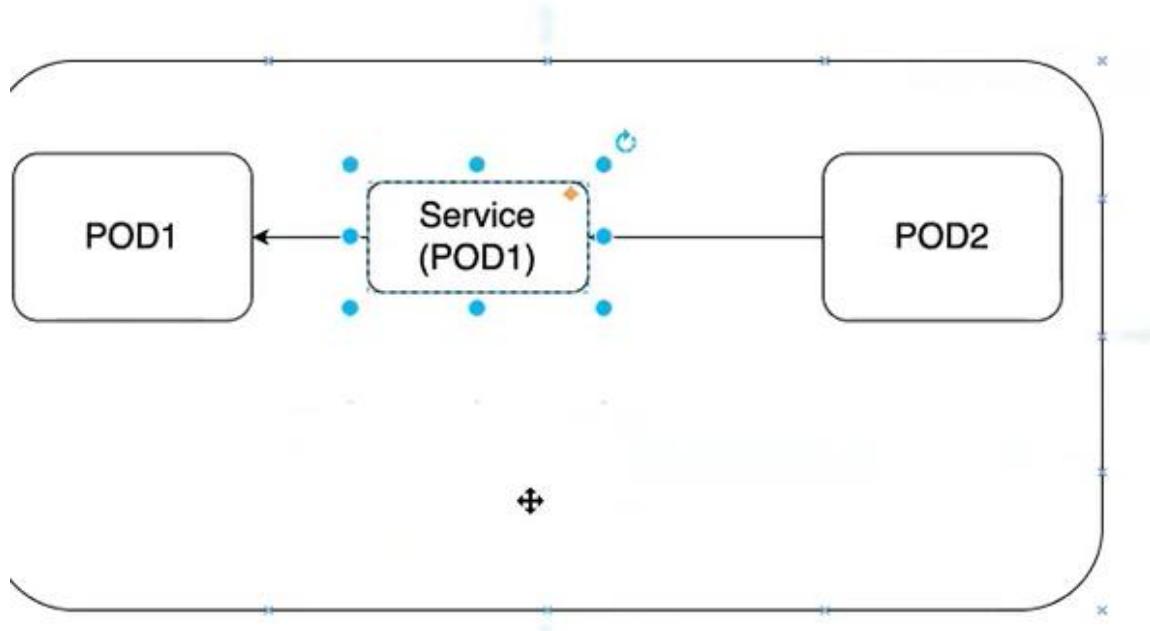
```
ubuntu@Kubernetes-Master:~$ nano clusterip.yml
ubuntu@Kubernetes-Master:~$ kubectl apply -f clusterip.yml
service/nginx-service-clusterip created
ubuntu@Kubernetes-Master:~$ kubectl get svc
NAME           TYPE      CLUSTER-IP      EXTERNAL-IP     PORT(S)        AGE
kubernetes     ClusterIP  10.96.0.1      <none>        443/TCP       31h
nginx-service  NodePort   10.99.10.6     <none>        8080:30081/TCP 3h
nginx-service-clusterip  ClusterIP  10.109.225.246  <none>        8080/TCP     18s
ubuntu@Kubernetes-Master:~$
```

25. its pending and terminating because I didn't launch Kubernetes-worker node and launch its terminated and running and we can see the curl 10.109.225.246:8080.



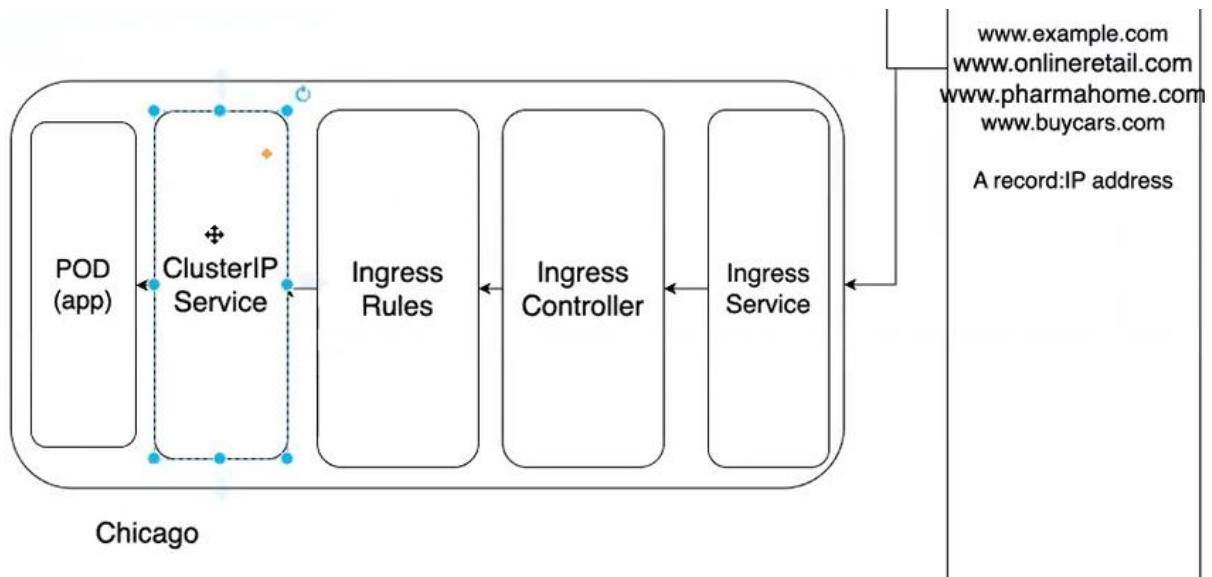
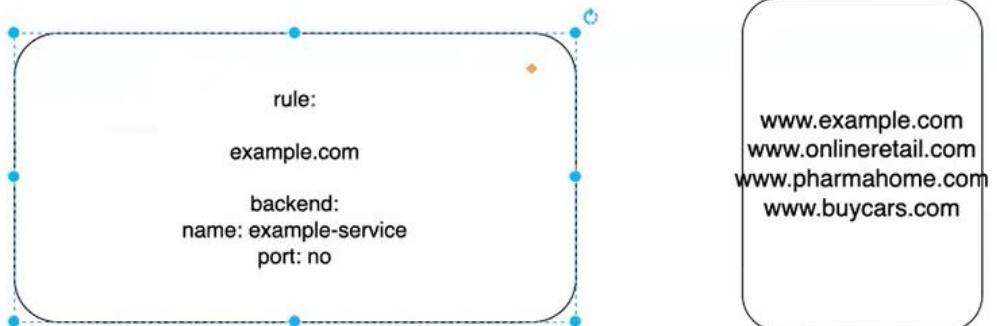
```
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-deployment-64dc447c-7l7cx  0/1     Pending   0          66m
nginx-deployment-64dc447c-dzv6z  1/1     Terminating   0          4h40m
nginx-deployment-64dc447c-hslnf  1/1     Terminating   0          4h40m
nginx-deployment-64dc447c-nnpxt  0/1     Pending   0          66m
nginx-deployment-64dc447c-s2n9x  0/1     Pending   0          66m
nginx-deployment-64dc447c-t1xr4  1/1     Terminating   0          4h40m
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-deployment-64dc447c-7l7cx  1/1     Running   0          68m
nginx-deployment-64dc447c-nnpxt  1/1     Running   0          68m
nginx-deployment-64dc447c-s2n9x  1/1     Running   0          68m
ubuntu@Kubernetes-Master:~$ kubectl get svc
NAME           TYPE      CLUSTER-IP      EXTERNAL-IP     PORT(S)        AGE
kubernetes     ClusterIP  10.96.0.1      <none>        443/TCP       32h
nginx-service-clusterip  ClusterIP  10.109.225.246  <none>        8080/TCP     70m
ubuntu@Kubernetes-Master:~$ curl 10.109.225.246:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
```

kubernetes cluster



Ingress

routing rules -> on how the outside traffic will be routed to inside



Kubernetes Ingress

Configuration

```
kubectl apply -f  
https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.8.2/deploy/static/provider/baremetal/deploy.yaml
```

Ingress: In Kubernetes, ingress acts as an **entry point for external traffic, routing it to internal services based on defined rules.**

26. Deploying the Ingress

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller/v1.8.2/deploy/static/provider/baremetal/deploy.yaml and
```

```
kubectl get pods -n ingress-nginx
```

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.8.2/deploy/static/provider/baremetal/deploy.yaml
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
serviceaccount/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
configmap/ingress-nginx-controller created
service/ingress-nginx-controller created
service/ingress-nginx-controller-admission created
deployment.apps/ingress-nginx-controller created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ingressClass.networking.k8s.io/nginx created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created
ubuntu@Kubernetes-Master:~$ kubectl get pods -n ingress-nginx
NAME           READY   STATUS    RESTARTS   AGE
ingress-nginx-admission-create-x762l   0/1     Completed   0          22s
ingress-nginx-admission-patch-lj9jw    0/1     Completed   0          22s
ingress-nginx-controller-6dc9c5fb7c-9gndj 0/1     Running    0          22s
ubuntu@Kubernetes-Master:~$
```

27. kubectl get svc -n ingress-nginx

```
ubuntu@Kubernetes-Master:~$ kubectl get svc -n ingress-nginx
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)
AGE
ingress-nginx-controller   NodePort   10.105.75.98 <none>        80:30909/TCP,443:30927/TCP
72s
ingress-nginx-controller-admission ClusterIP  10.101.221.162 <none>        443/TCP
72s
ubuntu@Kubernetes-Master:~$
```

28. writing ingressrule.yml

```
ubuntu@Kubernetes-Master:~$ nano ingressrule.yml
```

29. Observer the things

The screenshot shows two terminal windows side-by-side. Both windows have a title bar with the text "ubuntu@Kubernetes-Master" and "ubuntu@Kubernetes-worker". The left window displays the command "nano ingressrule.yml" and the contents of the file. The right window shows the file name "ingressrule.yml *". The file itself contains the following YAML configuration:

```
GNU nano 7.2                                     ingressrule.yml *
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  ingressClassName: nginx
  rules:
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: nginx-service-clusterip
                port:
                  number: 8080
```

30. deploy ingressrule.yml

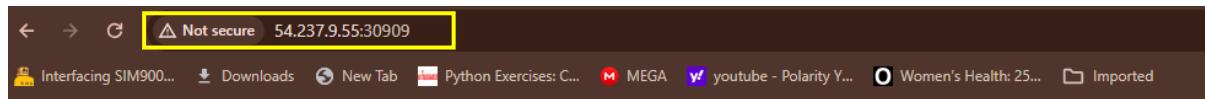
The screenshot shows two terminal windows. The left window shows the command "nano ingressrule.yml" and the right window shows the command "kubectl apply -f ingressrule.yml". The output indicates that an "ingress.networking.k8s.io/my-ingress" was created. The right window then shows the command "kubectl get svc -n ingress-nginx" which lists two services:

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
ingress-nginx-controller	NodePort	10.105.75.98	<none>	80:30909/TCP,443:30927/TCP
ingress-nginx-controller-admission	ClusterIP	10.101.221.162	<none>	443/TCP

31. copy the public ip

The screenshot shows the AWS Lambda console interface. At the top, there are four tabs: "EC2 Instance Connect", "Session Manager", "SSH client", and "EC2 serial console". Below these tabs, the "Instance ID" is listed as "i-0182e44fa7376634f (Kubernetes-Master)". Under the "Connection Type" section, there are two options: "Connect using EC2 Instance Connect" (selected) and "Connect using EC2 Instance Connect Endpoint". A tooltip for the first option states: "Connect using the EC2 Instance Connect browser-based client with a public IP address." A success message "Public IP address copied" is displayed in a blue box. At the bottom, the public IP address "54.237.9.55" is shown in a yellow box.

32. now we can see the nginx server with the port no of 30909



33. observe the path of ingress and compare with nodeport

```
ubuntu@Kubernetes-Master: ~$ cat my-ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  ingressClassName: nginx
  rules:
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: nginx-service-clusterip
                port:
                  number: 8080
ubuntu@Kubernetes-Master: ~$ cat nodeport.yml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app: nginx
  type: NodePort
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
      nodePort: 30081
```

34. Check the ingress pods and checking the logs

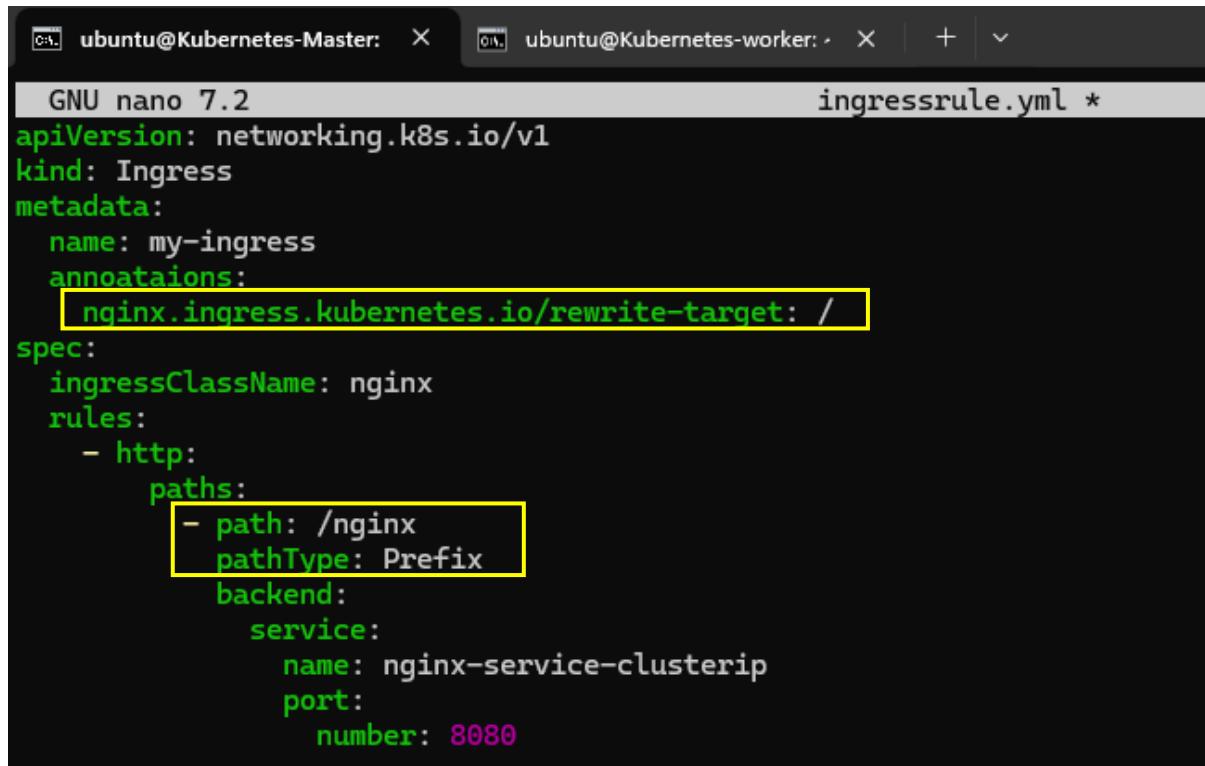
```
ubuntu@Kubernetes-Master:~$ kubectl get pods -n ingress-nginx
NAME                               READY   STATUS    RESTARTS   AGE
ingress-nginx-admission-create-x762l   0/1     Completed   0          17m
ingress-nginx-admission-patch-lj9jw    0/1     Completed   0          17m
ingress-nginx-controller-6dc9c5fb7c-9gndj  1/1     Running   0          17m
ubuntu@Kubernetes-Master:~$ kubectl logs ingress-nginx-controller-6dc9c5fb7c-9gndj -n ingress-nginx

NGINX Ingress controller
  Release:        v1.8.1
  Build:         dc88dce9ea5e700f3301d16f971fa17c6cfe757d
  Repository:    https://github.com/kubernetes/ingress-nginx
  nginx version: nginx/1.21.6

-----
[0530 15:52:34.726477] W0530 15:52:34.726477 [client_config.go:618] Neither --kubeconfig nor --master was specified. Using the inClusterConfig. This might not work.
[0530 15:52:34.726793] I0530 15:52:34.734964 [main.go:209] "Creating API client" host="https://10.96.0.1:443"
[0530 15:52:34.734964] I0530 15:52:34.734964 [main.go:253] "Running in Kubernetes cluster" major="1" minor="28" git="v1.28.10" state="clean" commit="21be1d76a90bc00e2b0f6676a664bd0f97224155" platform="linux/amd64"
[0530 15:52:35.182147] I0530 15:52:35.182147 [main.go:104] "SSL fake certificate created" file="/etc/ingress-controller/ssl/default-fake-certificate.pem"
[0530 15:52:35.200802] I0530 15:52:35.200802 [ssl.go:533] "loading tls certificate" path="/usr/local/certificates/cert" key="/usr/local/certificates/key"
[0530 15:52:35.211110] I0530 15:52:35.211110 [nginx.go:261] "Starting NGINX Ingress controller"
[0530 15:52:35.217923] I0530 15:52:35.217923 [event.go:285] Event{v1.ObjectReference{Kind:"ConfigMap", Namespace:"ingress-nginx", Name:"ingress-nginx-controller", UID:"fe675d76-25b9-4214-bd29-07437a1123a0", APIVersion:"v1", ResourceVersion:"24791", FieldPath:""}: type: 'Normal' reason: 'CREATE' ConfigMap ingress-nginx/ingress-nginx-controller
[0530 15:52:36.413126] I0530 15:52:36.413126 [nginx.go:30U] "Starting NGINX process"
[0530 15:52:36.413364] I0530 15:52:36.413364 [leaderelection.go:248] attempting to acquire leader lease ingress-nginx/ingress-nginx-leader...
[0530 15:52:36.413630] I0530 15:52:36.413630 [nginx.go:324] "Starting validation webhook" address=:8443 certPath=/usr/local/certificates/cert keyPath=/usr/local/certificates/key"
[0530 15:52:36.414108] I0530 15:52:36.414108 [controller.go:190] "Configuration changes detected, backend reload required"
[0530 15:52:36.422143] I0530 15:52:36.422143 [leaderelection.go:258] successfully acquired lease ingress-nginx/ingress-nginx-leader
[0530 15:52:36.422356] I0530 15:52:36.422356 [status.go:84] "New leader elected" identity="ingress-nginx-controller-6dc9c5fb7c-9gndj"
[0530 15:52:36.429184] I0530 15:52:36.429184 [status.go:215] "POD is not ready" pod="ingress-nginx/ingress-nginx-controller-6dc9c5fb7c-9gndj" node="kubernetes-worker"

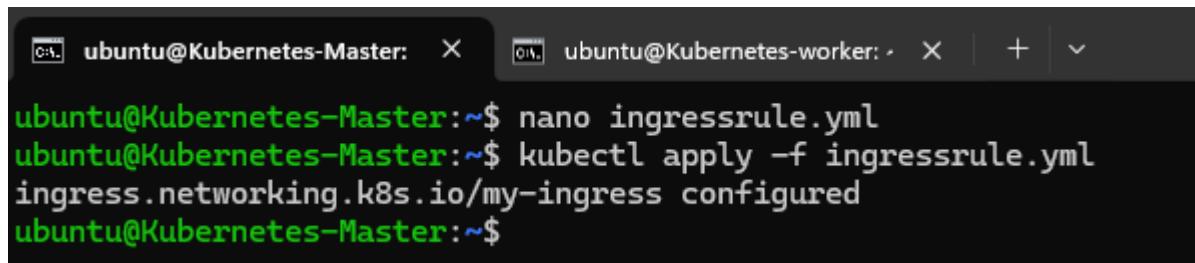
[0530 15:52:36.429184] I0530 15:52:36.429184 [status.go:215] "POD is not ready" pod="ingress-nginx/ingress-nginx-controller-6dc9c5fb7c-9gndj" node="kubernetes-worker"
[0530 15:52:36.498817] I0530 15:52:36.498817 [controller.go:207] "Backend successfully reloaded"
[0530 15:52:36.499460] I0530 15:52:36.499460 [controller.go:218] "Initial sync, sleeping for 1 second"
[0530 15:52:36.499627] I0530 15:52:36.499627 [event.go:285] Event{v1.ObjectReference{Kind:"Pod", Namespace:"ingress-nginx", Name:"ingress-nginx-controller-6dc9c5fb7c-9gndj", UID:"la3758f9-718d-411b-bc40-7b25f18b2d87", APIVersion:"v1", ResourceVersion:"24826", FieldPath:""}: type: 'Normal' reason: 'RELOAD' NGINX reload triggered due to a change in configuration
[0530 16:03:21.686521] I0530 16:03:21.686521 [admission.go:149] processed ingress via admission controller {testedIngressLength:1 testedIngressTime:0.044s renderingIngressLength:1 renderingIngressTime:0s admissionTime:14.3kBps testedConfigurationSize:0.044}
[0530 16:03:21.686563] I0530 16:03:21.686563 [main.go:110] "successfully validated configuration, accepting" ingress="default/my-ingress"
[0530 16:03:21.695557] I0530 16:03:21.695557 [store.go:432] "Found valid IngressClass" ingress="default/my-ingress" ingressClass="inginx"
[0530 16:03:21.696032] I0530 16:03:21.696032 [controller.go:190] "Configuration changes detected, backend reload required"
[0530 16:03:21.696052] I0530 16:03:21.696052 [event.go:285] Event{v1.ObjectReference{Kind:"Ingress", Namespace:"default", Name:"my-ingress", UID:"62a17fb0-3b89-490e-820c-857b6952bafe", APIVersion:"networking.k8s.io/v1", ResourceVersion:"25883", FieldPath:""}: type: 'Normal' reason: 'Sync' Scheduled for sync
[0530 16:03:21.778693] I0530 16:03:21.778693 [controller.go:207] "Backend successfully reloaded"
[0530 16:03:21.779080] I0530 16:03:21.779080 [event.go:285] Event{v1.ObjectReference{Kind:"Pod", Namespace:"ingress-nginx", Name:"ingress-nginx-controller-6dc9c5fb7c-9gndj", UID:"la3758f9-718d-411b-bc40-7b25f18b2d87", APIVersion:"v1", ResourceVersion:"24826", FieldPath:""}: type: 'Normal' reason: 'RELOAD' NGINX reload triggered due to a change in configuration
[0530 16:03:36.432655] I0530 16:03:36.432655 [status.go:300] "Updating Ingress status" namespace="default" ingress="my-ingress" currentValue=null newValue=[{"ip":"172.31.53.219"}]
[0530 16:03:36.438917] I0530 16:03:36.438917 [event.go:285] Event{v1.ObjectReference{Kind:"Ingress", Namespace:"default", Name:"my-ingress", UID:"62a17fb0-3b89-490e-820c-857b6952bafe", APIVersion:"networking.k8s.io/v1", ResourceVersion:"25909", FieldPath:""}: type: 'Normal' reason: 'Sync' Scheduled for sync
[0530 16:03:36.440000] I0530 16:03:36.440000 ["\x16\x03\x01\x07\x00\x01\x00\x06\xFC\x03\x03<\x03<\x0B\xA\x07\x01\x00\x06\xDC\x03\x03\x18<\x87D-\x4ej\xA6\xF2_\xA50z\x12\xA8:\xCF\xW\xF8\xBD\xAC\x04_\x12\x93\x04_\xP\xB7\xEC\xA7\x7A\x05\x97\x07]\x1E15\x9B\x7F\xF3\xB1\x11\x9B\xA\xC\x80\xA\x01\xDB\x00\xBA\xBA\x13\x01\x13\x02\x13\x03\xC0+\x04\xC9\x00\xC0\xA\xC\x01\x13\x00\x9C\x00\xD\x00\x00\x05\x01\x00\x93\xA\x8A\x00\x00\x00\xFE" 400 150 "-" "0 0.492 []" [] -- ff008d938ced739b0485d2f30d0c29a4
[0530 16:03:36.440000] I0530 16:03:36.440000 [10.244.0.0 -- [30/May/2024:16:05:32 +0000] "\x16\x03\x01\x07\x00\x01\x00\x06\xFC\x03\x03<\x03<\x0B\xA\x07\x01\x00\x06\xDC\x03\x03\x18<\x87D-\x4ej\xA6\xF2_\xA50z\x12\xA8:\xCF\xW\xF8\xBD\xAC\x04_\x12\x93\x04_\xP\xB7\xEC\xA7\x7A\x05\x97\x07]\x1E15\x9B\x7F\xF3\xB1\x11\x9B\xA\xC\x80\xA\x01\xDB\x00\xBA\xBA\x13\x01\x13\x02\x13\x03\xC0+\x04\xC9\x00\xC0\xA\xC\x01\x13\x00\x9C\x00\xD\x00\x00\x05\x01\x00\x93\xA\x8A\x00\x00\x00\xFE" 400 150 "-" "0 0.233 []" [] -- e33fab4545216ale968d49e0453d2d6
[0530 16:03:36.440000] I0530 16:03:36.440000 [10.244.0.0 -- [30/May/2024:16:05:33 +0000] "GET / HTTP/1.1" 200 612 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" 432 0.002 [default-ingress-service-clusterip-8080] [] 10.244.1.30:80 612 0.002 200 b644f54c7fd807d223358853d0292583
[0530 16:03:36.440000] I0530 16:03:36.440000 [10.244.0.0 -- [30/May/2024:16:05:33 +0000] "\x16\x03\x01\x06\xA0\x01\x00\x06\x9C\x03\x03\x03\x08\x8\x8\x8\x8\x9\xA\x0\xE\x6\xC\x4\x7\xF" 400 150 "-"]
```

35. Now in the Ingress in the Path : /nginx we thought we are routing but its not.



```
GNU nano 7.2                                     ingressrule.yml *
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
      - path: /nginx
        pathType: Prefix
      backend:
        service:
          name: nginx-service-clusterip
          port:
            number: 8080
```

36. Apply it it will show configure

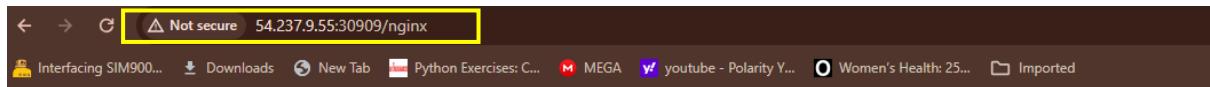


```
ubuntu@Kubernetes-Master:~$ nano ingressrule.yml
ubuntu@Kubernetes-Master:~$ kubectl apply -f ingressrule.yml
ingress.networking.k8s.io/my-ingress configured
ubuntu@Kubernetes-Master:~$
```

37. After ingress port no we enter /hello its show 404 error



38. but type /nginx it will show the welcome page



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

39. cat and get current pods

```
ubuntu@Kubernetes-Master:~$ cat ingressrule.yml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
      - path: /nginx
        pathType: Prefix
        backend:
          service:
            name: nginx-service-clusterip
            port:
              number: 8080
ubuntu@Kubernetes-Master:~$ kubectl get pod
NAME                      READY   STATUS    RESTARTS   AGE
nginx-deployment-64dcd447c-7l7cx   1/1     Running   1 (34m ago)  161m
nginx-deployment-64dcd447c-nnpxt   1/1     Running   1 (34m ago)  161m
nginx-deployment-64dcd447c-s2n9x   1/1     Running   1 (34m ago)  161m
ubuntu@Kubernetes-Master:~$
```

A screenshot of a terminal window showing two command-line sessions. The first session shows the contents of the file 'ingressrule.yml'. The second session shows the output of the command 'kubectl get pod', which lists three pods: 'nginx-deployment-64dcd447c-7l7cx', 'nginx-deployment-64dcd447c-nnpxt', and 'nginx-deployment-64dcd447c-s2n9x', all in a 'Running' state with an age of 161m.

40. default html actually thing is what ever you /typeinfo it will show the default page its not routing

```
ubuntu@Kubernetes-Master: ~  X  ubuntu@Kubernetes-worker: ~  X  +  v
ubuntu@Kubernetes-Master:~$ kubectl exec -it pod/nginx-deployment-64dc447c-7l7cx -- /bin/bash
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx/html# ls
50x.html index.html
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx/html# cat index.html
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx/html#
```

```
ubuntu@Kubernetes-Master: ~  X  ubuntu@Kubernetes-worker: ~  X  +  v
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx/html#
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx/html# cd ..
root@nginx-deployment-64dc447c-7l7cx:/usr/local/nginx# cd ..
root@nginx-deployment-64dc447c-7l7cx:/usr/local# ls
bin etc games include lib man nginx sbin share src
root@nginx-deployment-64dc447c-7l7cx:/usr/local# cd /etc
root@nginx-deployment-64dc447c-7l7cx:/etc# ls
X11           hostname      network      rpc
alternatives   hosts        networks     scgi_params
apt           init         nginx.conf  scgi_params.default
bash.bashrc    init.d       nginx.conf.default  securityty
bash_completion.d  inittab     nsswitch.conf  security
bindresvport.blacklist  inputrc      opt          selinux
cron.daily     isserv       os-release   services
debconf.conf   isserv.conf  pam.conf    sgml
debian_version  isserv.conf.d pam.d       shadow
default        iproute2     passwd      shadow-
dpkg          issue        passwd-    shells
environment    issue.net    perl        skel
fastcgi.conf   kernel       profile     staff-group-for-usr-local
fastcgi.conf.default  koi-utf     profile.d   subgid
fastcgi_params  koi-win     protocols   subgid-
fastcgi_params.default ld.so.cache  rc.local    subuid
fonts          ld.so.conf   rc0.d      subuid-
fonts.conf.d   ld.so.conf.d  rc1.d      terminfo
fstab          libaudit.conf rc2.d      timezone
fstab.d        localtime   rc3.d      ucf.conf
gai.conf       login.defs   rc4.d      uwsgi_params
group          logrotate.d  rc5.d      uwsgi_params.default
group-         mime.types   rc6.d      win-utf
gshadow        mime.types.default  rcS.d      xml
gshadow-       mke2fs.conf  resolv.conf
host.conf      motd        rmt
```

```
ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + | ^

root@nginx-deployment-64dc447c-7l7cx:/etc# cat nginx.conf

location / {
    root    html;
    index  index.html index.htm;
}

#error_page  404          /404.html;

# redirect server error pages to the static page /50x.html
#
error_page   500 502 503 504  /50x.html;
location = /50x.html {
    root    html;
}
```

EXAMPLE OF INGRESS

```
ubuntu@Kubernetes-Master: ~$ www.something.com/nginx = it will consider as www.something.com

ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + | ^

GNU nano 7.2                                     ingressrule.yml *

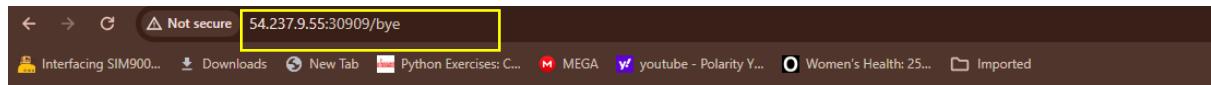
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
      - path: /bye
        pathType: Prefix
        backend:
          service:
            name: nginx-service-clusterip
            port:
              number: 8080
```

```
ubuntu@Kubernetes-Master:~$ nano ingressrule.yml
ubuntu@Kubernetes-Master:~$ kubectl apply -f ingressrule.yml
ingress.networking.k8s.io/my-ingress configured
ubuntu@Kubernetes-Master:~$ |
```



404 Not Found

nginx



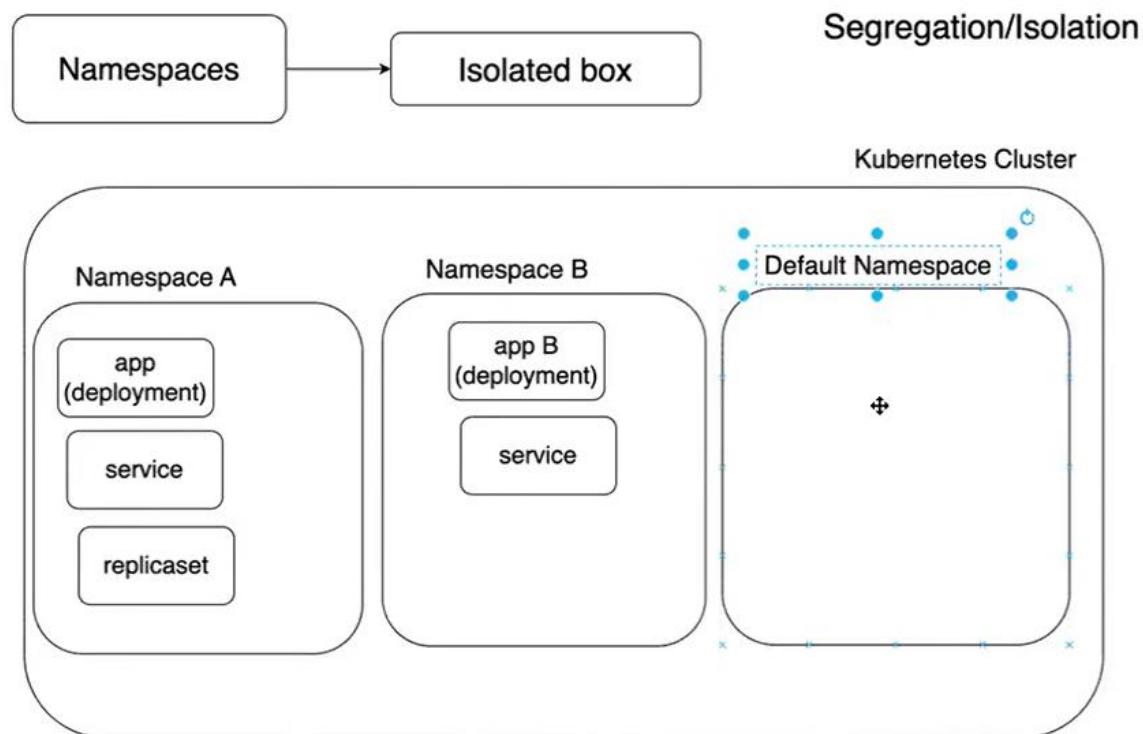
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Kubernetes-DAY4



NAME SPACE:

In Kubernetes, a namespace provides a mechanism for **isolating groups of resources within a single cluster**

```
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME                           READY   STATUS    RESTARTS   AGE
nginx-deployment-64dcd447c-7l7cx 1/1     Running  2 (84s ago)  20h
nginx-deployment-64dcd447c-nnpxt 1/1     Running  2 (84s ago)  20h
nginx-deployment-64dcd447c-s2n9x 1/1     Running  2 (84s ago)  20h
ubuntu@Kubernetes-Master:~$ kubectl get pod -n default
NAME                           READY   STATUS    RESTARTS   AGE
nginx-deployment-64dcd447c-7l7cx 1/1     Running  2 (2m13s ago)  20h
nginx-deployment-64dcd447c-nnpxt 1/1     Running  2 (2m13s ago)  20h
nginx-deployment-64dcd447c-s2n9x 1/1     Running  2 (2m13s ago)  20h
ubuntu@Kubernetes-Master:~$
```

Creating namespace named as web

```
ubuntu@Kubernetes-Master:~$ kubectl create ns web
namespace/web created
ubuntu@Kubernetes-Master:~$ ls
clusterip.yml deployment.yml ingressrule.yml nodeport.yml pod.yml replicaset.yml
ubuntu@Kubernetes-Master:~$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
ubuntu@Kubernetes-Master:~$
```

Editing pod.yml

```
ubuntu@Kubernetes-Master:~$ nano pod.yml
```

In metadata namespace: web

```
GNU nano 7.2                                     pod.yml *
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: web
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
```

```
ubuntu@Kubernetes-Master:~$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: web
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
ubuntu@Kubernetes-Master:~$
```

Created the pod

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f pod.yml
pod/nginx-pod created
ubuntu@Kubernetes-Master:~$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
nginx-deployment-64dcd447c-7l7cx  1/1     Running   2 (5m59s ago)  20h
nginx-deployment-64dcd447c-nnpxt  1/1     Running   2 (5m59s ago)  20h
nginx-deployment-64dcd447c-s2n9x  1/1     Running   2 (5m59s ago)  20h
ubuntu@Kubernetes-Master:~$
```

This is nginx-pod which in this we given namespace as web

```
ubuntu@Kubernetes-Master:~$ kubectl get pods -n web
NAME      READY   STATUS    RESTARTS   AGE
nginx-pod  1/1     Running   0          55s
ubuntu@Kubernetes-Master:~$
```

Deleted nginx-pod and showing no web namespace

```
ubuntu@Kubernetes-Master:~$ kubectl delete pod/nginx-pod -n web
pod "nginx-pod" deleted
ubuntu@Kubernetes-Master:~$ kubectl get pod -n web
No resources found in web namespace.
ubuntu@Kubernetes-Master:~$
```

Go to edit pod

```
ubuntu@Kubernetes-Master:~$ nano pod.yml
```

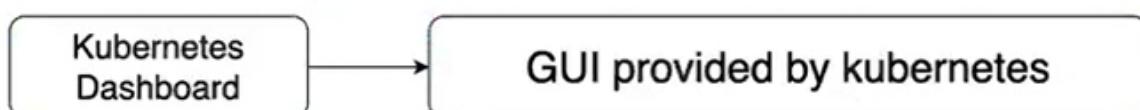
Remove namespace

```
GNU nano 7.2                                     pod.yml *
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
```

Now apply -f pod.yml -n web and it will creates and kubectl get pod -n web and this command tell with the name of web, pod is running

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f pod.yml -n web
pod/nginx-pod created
ubuntu@Kubernetes-Master:~$ kubectl get pod -n web
NAME      READY   STATUS    RESTARTS   AGE
nginx-pod  1/1     Running   0          17s
ubuntu@Kubernetes-Master:~$ ls
clusterip.yml deployment.yml ingressrule.yml nodeport.yml pod.yml replicaset.yml
ubuntu@Kubernetes-Master:~$
```

KUBERNETES Dashboard:



Step1: Deploy the `kubernetes-dashboard` manifest
Step2: made `svc` to `nodeport` (practice)
Step3: create serviceaccount
Step4: give permission to serviceaccount
Step5: create token for serviceaccount
Step6: paste the token in `kubernetes-dashboard`

Need to deploy

The screenshot shows the official Kubernetes documentation website at <https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/>. The page title is "Deploy and Access the Kubernetes Dashboard". The main content area includes a search bar, navigation links for Documentation, Getting started, Concepts, and Tasks, and a link to the Kubernetes Blog and Training. The URL in the browser's address bar is https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/.

Kubernetes Dashboard

Configuration

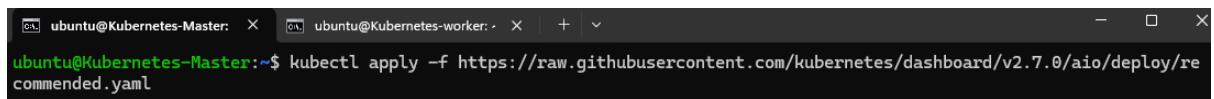
Step 1: Apply manifest

```
kubectl apply -f
```

```
https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
```

```
https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
```

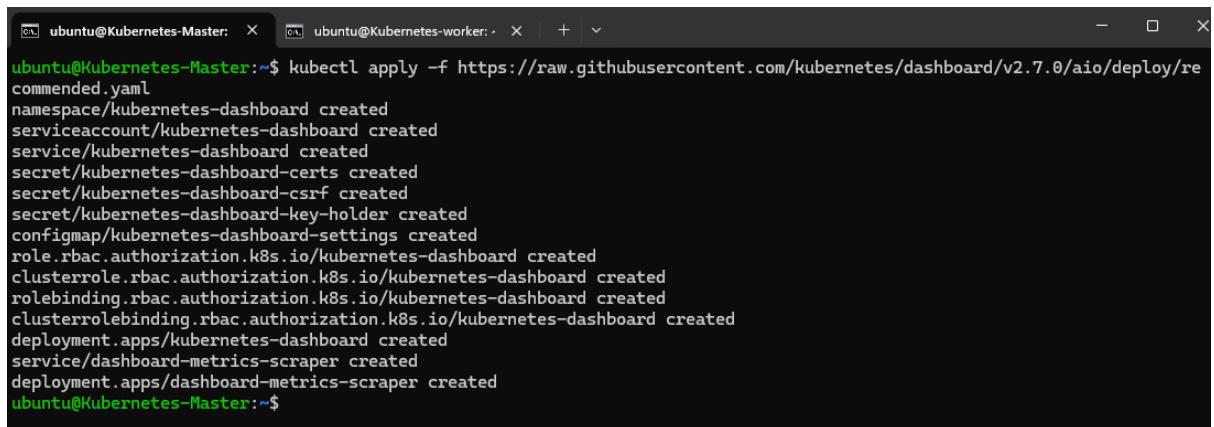
if you want to see the link, u can it will show script



```
ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + -
```

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
```

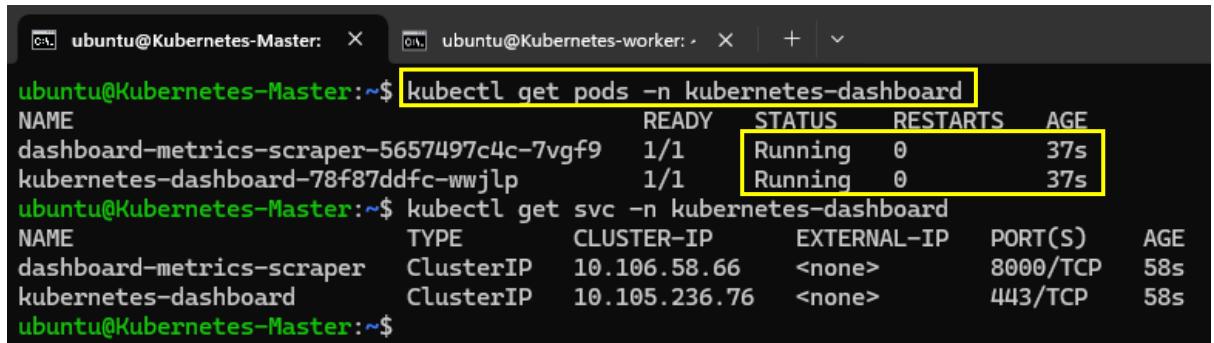
Follow it



```
ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + -
```

```
ubuntu@Kubernetes-Master:~$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
ubuntu@Kubernetes-Master:~$
```

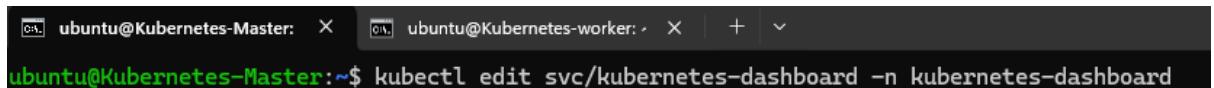
Now we can see the pod with name of **Kubernetes-dashboard** we can see



```
ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + -
```

```
ubuntu@Kubernetes-Master:~$ kubectl get pods -n kubernetes-dashboard
NAME                               READY   STATUS    RESTARTS   AGE
dashboard-metrics-scraper-5657497c4c-7vgf9   1/1     Running   0          37s
kubernetes-dashboard-78f87ddfc-wwjlp        1/1     Running   0          37s
ubuntu@Kubernetes-Master:~$ kubectl get svc -n kubernetes-dashboard
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
dashboard-metrics-scraper   ClusterIP  10.106.58.66 <none>       8000/TCP  58s
kubernetes-dashboard   ClusterIP  10.105.236.76  <none>       443/TCP   58s
ubuntu@Kubernetes-Master:~$
```

Go edit this Kubernetes dashboard because it in clusterip, need to change nodeport



```
ubuntu@Kubernetes-Master: ~  ubuntu@Kubernetes-worker: ~ + -
```

```
ubuntu@Kubernetes-Master:~$ kubectl edit svc/kubernetes-dashboard -n kubernetes-dashboard
```

Scroll and edit

```
apiVersion: v1
kind: Service
selector:
  k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: ClusterIP
status:
  loadBalancer: {}
<-- INSERT -->

apiVersion: v1
kind: Service
selector:
  k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
<-- INSERT -->
```

It will show edited and remember the 32473

```
ubuntu@Kubernetes-Master:~$ kubectl edit svc/kubernetes-dashboard -n kubernetes-dashboard
service/kubernetes-dashboard edited 35L, 1145B written
ubuntu@Kubernetes-Master:~$ kubectl get svc -n kubernetes-dashboard
NAME           TYPE      CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
dashboard-metrics-scraper   ClusterIP  10.106.58.66 <none>       8000/TCP     6m32s
kubernetes-dashboard   NodePort   10.105.236.76 <none>       443:32473/TCP 6m32s
ubuntu@Kubernetes-Master:~$
```

And copy the Kubernetes-master public ip

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Available
<input checked="" type="checkbox"/>	Kubernetes-Master	i-0182e44fa7376634f	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1
<input type="checkbox"/>	Kubernetes-Slave	i-0e0d3dc5efcf9445	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1

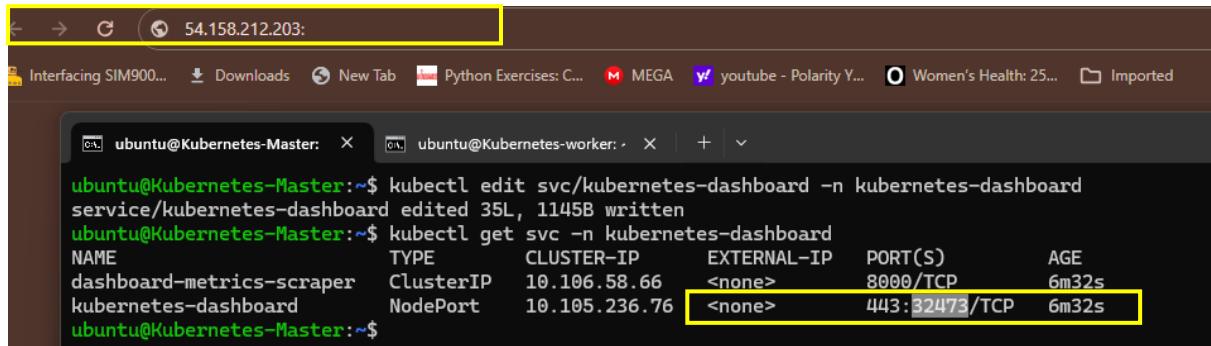
=

i-0182e44fa7376634f (Kubernetes-Master)

▼ Instance summary Info

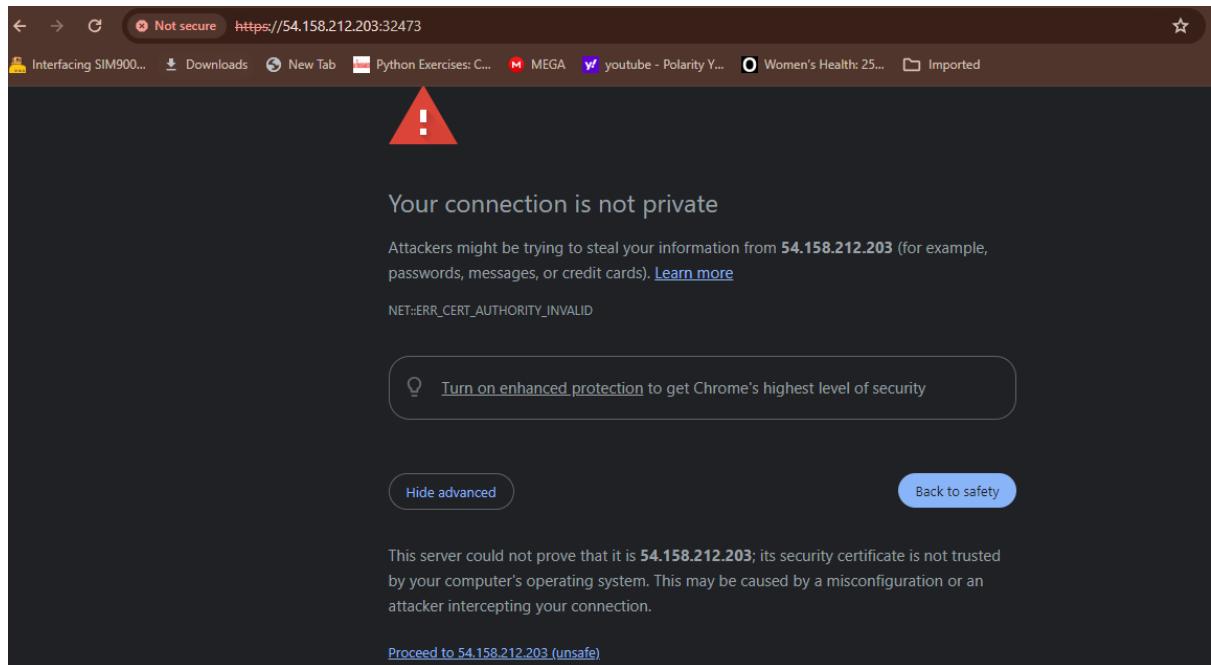
Instance ID i-0182e44fa7376634f (Kubernetes-Master)	Public IPv4 address 54.158.212.203 open address	Private IPv4 addresses 172.31.62.130
--	--	---

Copy it

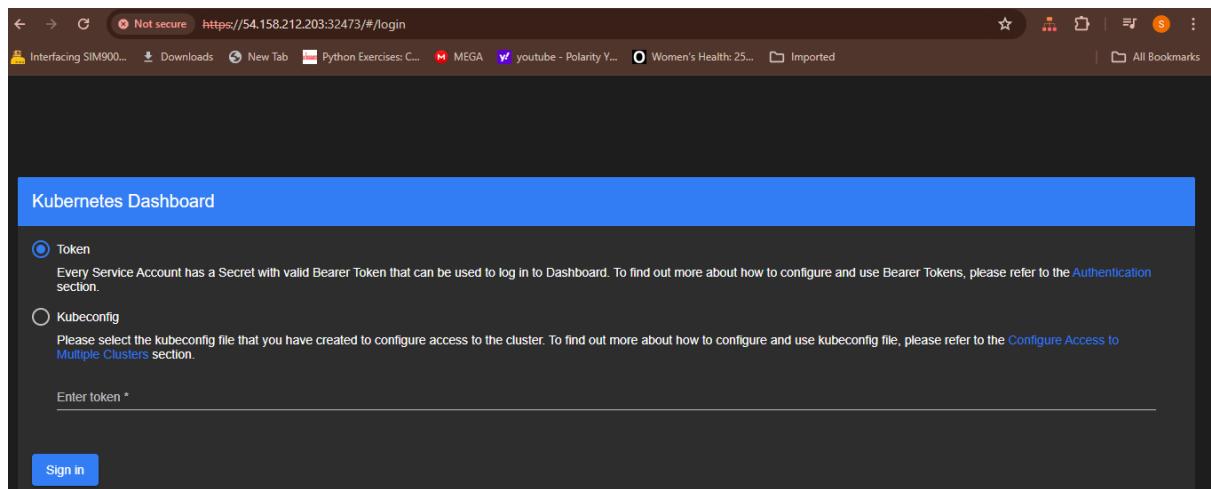


```
ubuntu@Kubernetes-Master:~$ kubectl edit svc/kubernetes-dashboard -n kubernetes-dashboard
service/kubernetes-dashboard edited 35L, 1145B written
ubuntu@Kubernetes-Master:~$ kubectl get svc -n kubernetes-dashboard
NAME                      TYPE      CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
dashboard-metrics-scraper  ClusterIP  10.106.58.66  <none>       8000/TCP      6m32s
kubernetes-dashboard        NodePort   10.105.236.76  <none>       443:32473/TCP  6m32s
ubuntu@Kubernetes-Master:~$
```

Paste it and to advance and hide advanced.



How to create a token?



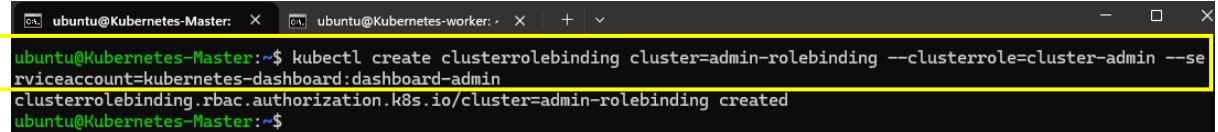
Kubernetes Dashboard

Token
Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

Kubeconfig
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.

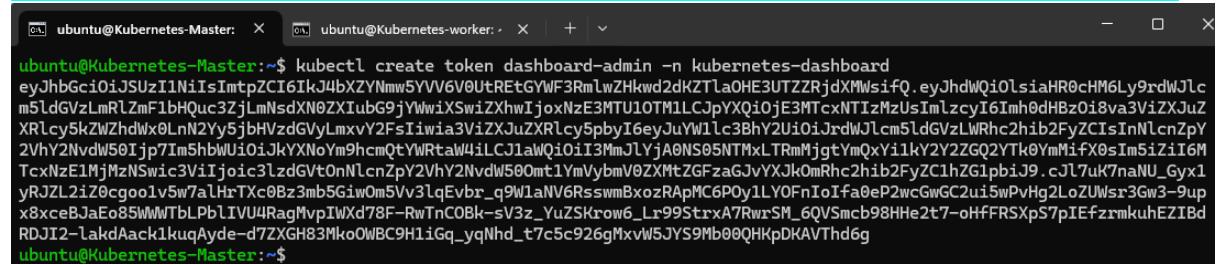
Enter token *

```
kubectl create clusterrolebinding cluster-admin-rolebinding --clusterrole=cluster-admin --serviceaccount=kubernetes-dashboard:dashboard-admin
```



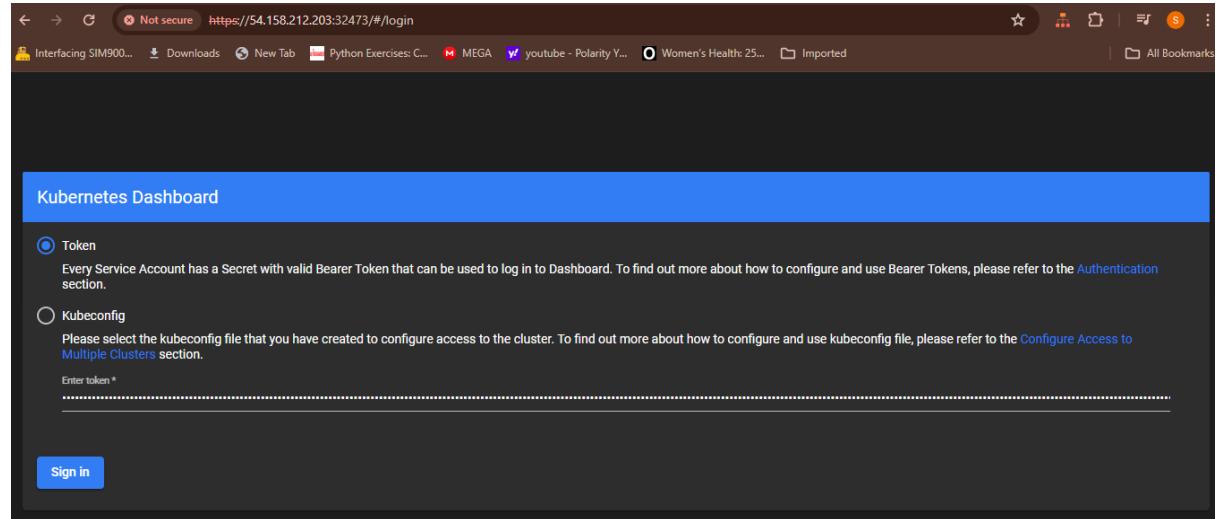
```
ubuntu@Kubernetes-Master:~$ kubectl create clusterrolebinding cluster-admin-rolebinding --clusterrole=cluster-admin --serviceaccount=kubernetes-dashboard:dashboard-admin
clusterrolebinding.rbac.authorization.k8s.io/cluster-admin-rolebinding created
ubuntu@Kubernetes-Master:~$
```

kubectl create serviceaccount dashboard-admin -n kubernetes-dashboard Copy the token



```
ubuntu@Kubernetes-Master:~$ kubectl create token dashboard-admin -n kubernetes-dashboard
eyJhbGciOiJSUzI1NiIsImtpZC16IkJ4bXZYNm5YVV6V0UTERetGYWF3RmLwZhKwd2dKZTlaHE3UTZzRjdXMWsifQ.eyJhdWQiOlsiaHR0cHM6Ly9rdWJlc
m5ldGVzLmRLzmF1bHQuc3ZjlmNsdxN6ZXIubG9jYWhiXSwiZXhwIjoxNzE3MTU0TM1LCJpYXQiOjE3MTcxNTIzMzUsImIzcyI61mh0dHBzOi8va3ViZXJuZ
XRlcyc5kZWzhdxWx0LnN2Yy5jbHvzdGvyLmxvY2fsIiwha3VizXJuZXRlcyc5pbyI6eyJuYW1lc3Bhy2UioiJrdWJlcm5ldGVzLWRhc2hib2FyZCisInNlcnZpY
2VhY2NvdW50Ijp7ImShbwUi0iJkYXNoYm9hcmQtYWRtaul4iLCJ1aWQioiI3mJLyjA0NS05NTMxLTrnMjgtYmQxYikY2ZGQ2YTk0YmMifX0sIm5iZii6M
TcxNzE1MjMzNSwic3ViIjoic3lzdGVtOnNlcnZpY2vhY2NvdW50mt1YmVybmv0ZXMtZGFzaGJvYXJkOmRhcz2hib2FyZC1hZG1pbij9.cJL7uK7naNU_Gyx1
yRJZL2iZ0cgoo1v5w7a1hXTc0Bz3mb5Giw0m5Vv3lqEvbr_q9W1aN6RsswmBxozRApMC6POy1LY0FnIoIfa0eP2wcGwGC2ui5wPvHg2LoZUWsr3Gw3-9up
x8xceBjaEo85WWTbLPb1IVU4RaqMvpIWxd78F-RwTnCOBk-sV3z_YuZSKrow6_Lx99StrxA7RwxSM_6QVSmcb98HHe2t7-oHffFRSxpS7pIEfzrmkuhEZIBd
RDJI2-LakdAAck1kuqAyde-d7ZXGH83Mko0WBC9H1iGq_yqNhdt7c5c926gMxvW5JYS9Mb00QHkpDKAVThd6g
ubuntu@Kubernetes-Master:~$
```

and paste it here



Successfully logged into Kubernetes dashboard

The screenshot shows the Kubernetes Dashboard's Workloads section. On the left, a sidebar lists options like Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, and Config & Storage. The main area is titled "Workload Status" and displays three large green circles, each representing a deployment. Below this, tabs for "Deployments", "Pods", and "Replica Sets" are visible. A table below the tabs shows one deployment named "nginx-deployment" with the image "nginx:1.7.1". The status of the deployment is "Running: 1".

Edit the deployment and we creates 5 deployment

The screenshot shows a modal dialog titled "Scale a resource" for the deployment "nginx-deployment". The dialog contains a text field labeled "Desired replicas" with the value "5" highlighted by a yellow box. To the right, it shows the "Actual replicas" as "3". Below the dialog, a note says "This action is equivalent to: kubectl scale -n default deployment nginx-deployment --replicas=5". At the bottom of the dialog are "Scale" and "Cancel" buttons.

And its created

The screenshot shows the Kubernetes Dashboard's Workloads section again. The sidebar is identical to the first screenshot. The main area now shows a table for "Pods". Five rows are listed, all belonging to the "nginx-deployment" pod. Each row has a yellow box around it. The columns include Name, Images, Labels, Node, Status, Restarts, CPU Usage (cores), Memory Usage (bytes), and Created. All pods are in a "Running" state, with two having 2 restarts and three having 0. The "Created" column shows times ranging from "9 seconds ago" to "21 hours ago".

Workloads

Deployments

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx-deployment	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	0	-	-	a day ago

Edit the deployment

Pods

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx-deployment-64ddcd447c-4x96q	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	0	-	-	a minute ago
nginx-deployment-64ddcd447c-rxxq2	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	0	-	-	8s ago
nginx-deployment-64ddcd447c-7l7cx	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	2	-	-	2s ago
nginx-deployment-64ddcd447c-npxt	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	2	-	-	21 hours ago
nginx-deployment-64ddcd447c-s2n9x	nginx:1.7.1	app: nginx pod-template-hash: 64ddcd447c	kubernetes-worker	Running	2	-	-	21 hours ago

We can Edit the deployment here also

Create

Workloads

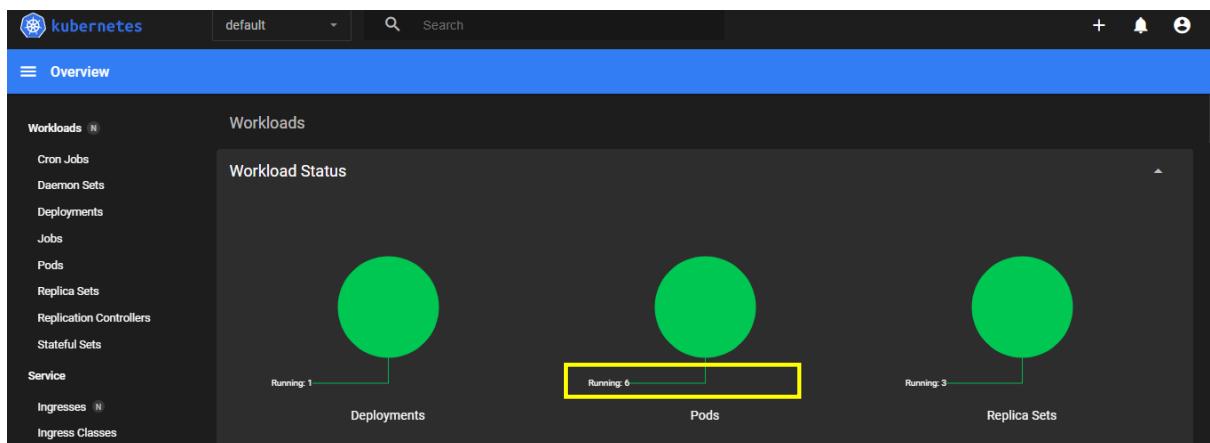
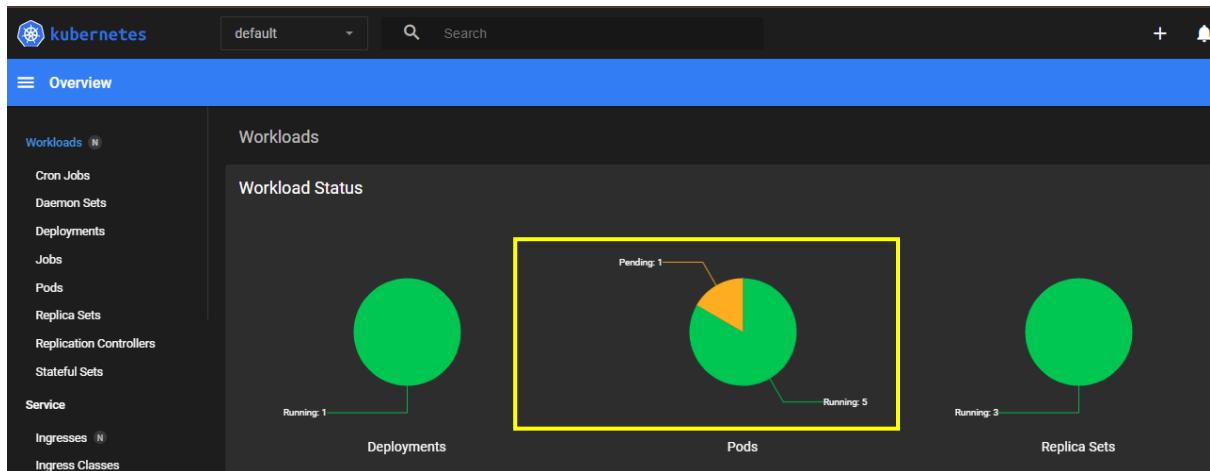
Create from input Create from file Create from form

```

apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
  
```

Upload Cancel

Here we can delete and create the pods, deployment



The screenshot shows the "Pods" section of the Overview page. The table lists six pods, all of which are currently running. The columns include Name, Images, Labels, Node, Status, Restarts, CPU Usage (cores), Memory Usage (bytes), and Created time. The pods are: nginx-pod, nginx-deployment-64ddc447c-4x9eq, nginx-deployment-64ddc447c-rxxq2, nginx-deployment-64ddc447c-77cx, nginx-deployment-64ddc447c-npxt, and another nginx-deployment entry. All pods are running on the "kubernetes-worker" node and were created within the last 21 hours.

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx-pod	nginx	-	kubernetes-worker	Running	0	-	-	a minute ago
nginx-deployment-64ddc447c-4x9eq	nginx:1.7.1	app: nginx pod-template-hash: 64ddc447c	kubernetes-worker	Running	0	-	-	7 minutes ago
nginx-deployment-64ddc447c-rxxq2	nginx:1.7.1	app: nginx pod-template-hash: 64ddc447c	kubernetes-worker	Running	0	-	-	7 minutes ago
nginx-deployment-64ddc447c-77cx	nginx:1.7.1	app: nginx pod-template-hash: 64ddc447c	kubernetes-worker	Running	2	-	-	21 hours ago
nginx-deployment-64ddc447c-npxt	nginx:1.7.1	app: nginx pod-template-hash: 64ddc447c	kubernetes-worker	Running	2	-	-	21 hours ago

Inside the pod

The screenshot shows the Kubernetes UI for a pod named `nginx-deployment-64ddcd447c-4x96q`. The sidebar on the left has a section for `Pods` which is highlighted with a yellow box. The main content area is divided into three sections: `Metadata`, `Resource information`, and `Conditions`. In the `Metadata` section, it shows the pod's name, namespace (default), creation date (May 31, 2024), and age (7 minutes ago). It also lists the pod's UID and labels (app: nginx, pod-template-hash: 64ddcd447c). The `Resource information` section shows the pod is running on a node named `kubernetes-worker` with IP `10.244.1.45`, QoS class `BestEffort`, and 0 restarts. The `Conditions` section is currently empty.

Inside the pods > Conditions, controlled, replicaset

This screenshot shows the Kubernetes UI for the same pod. The sidebar includes a `Events` section which lists several log entries. The `Containers` section is expanded, showing one container named `nginx-container` with the image `nginx:1.7.1`. It has a status table with columns `Ready`, `Started`, and `Started At`, showing values true, true, and `2024-05-31T10:52:12Z` respectively. Below the container details is a `Mounts` section with a table.

More details about pods

This screenshot provides more detailed views of the pod's history and file system. The `Events` section shows the same log entries as the previous screenshot. The `Mounts` section is expanded, showing a table with a single entry for a secret named `kube-api-access-kf6pq` mounted at `/var/run/secrets/kubernetes.io/serviceaccount`.

We can edit from the terminal

```
ubuntu@Kubernetes-Master:~$ ls
clusterip.yml deployment.yml ingressrule.yml nodeport.yml pod.yml replicaset.yml
ubuntu@Kubernetes-Master:~$ cat clusterip.yml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service-clusterip
spec:
  selector:
    app: nginx
  type: ClusterIP
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
ubuntu@Kubernetes-Master:~$
```

And also we can edit from the Kubernetes dashboard

The screenshot shows the Kubernetes Dashboard's 'Create' interface for a 'Service'. The left sidebar lists various workload types: Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, and Services. The 'Service' item is selected. The main area has tabs for 'Create from input', 'Create from file', and 'Create from form'. A text input field contains the following YAML code:

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: nginx-service-clusterip
5 spec:
6   selector:
7     app: nginx
8   type: ClusterIP
9   ports:
10    - protocol: TCP
11      port: 8080
12      targetPort: 80
```

At the bottom of the input field are 'Upload' and 'Cancel' buttons.

We can upload from into our local machine

The screenshot shows the Kubernetes Dashboard's 'Create' interface for a 'Service', similar to the previous one but with a different UI for file uploads. The left sidebar and tabs are identical. The main area has a 'Choose YAML or JSON file' input field with a 'Choose...' button. Below it are 'Upload' and 'Cancel' buttons. A small '...' icon is located at the top right of the input field.

We can create app name, container image, number of pods through GUI(Kubernetes-dashboard)

The screenshot shows the 'Create' dialog for a 'Deployment' in the 'default' namespace. The 'Create from form' tab is selected. The fields are as follows:

- App name ***: nginx-salman
- Container image ***: nginx
- Number of pods ***: 2
- Service ***: None
- Namespace ***: default

Below the form are buttons for **Deploy**, **Preview**, **Cancel**, and **Show advanced options**.

Creating sample

The screenshot shows the 'Create' dialog for a 'Deployment' in the 'default' namespace. The 'Create from form' tab is selected. The fields are as follows:

- App name ***: nginx-salman
- Container image ***: nginx
- Number of pods ***: 2
- Service ***: None
- Namespace ***: default

Below the form are buttons for **Deploy**, **Preview**, **Cancel**, and **Show advanced options**.

Now we have 8 pods we can see just now we created.

The screenshot shows the 'Overview' page of the Kubernetes Dashboard. On the left, a sidebar lists various resources: Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, Config and Storage, Config Maps, Persistent Volume Claims, Secrets, and Storage Classes. The main area displays the 'Workload Status' section, which includes three circular progress indicators for Deployments, Pods, and Replica Sets, each showing a count of 'Running' pods. Below this is the 'Deployments' section, which lists two entries:

Name	Images	Labels	Pods	Created	⋮
nginx-salman	nginx	k8s-app: nginx-salman	2 / 2	17 seconds ago	⋮
nginx-deployment	nginx:1.7.1		5 / 5	a day ago	⋮

Now we are going to launch Jenkins-master. To deploy Kubernetes on jenkins

The screenshot shows the AWS CloudWatch Metrics interface. It lists several services and their current status:

Service	Instance ID	Status	Check Status	Region
Jenkins-Master	i-08500dd7c7a10a2bd	Running	2/2 checks passed	us-east-1
Kubernetes-Master	i-0182e44fa7376634f	Running	2/2 checks passed	us-east-1
Kubernetes-Slave	i-0e0d3dc5efcf9445	Running	2/2 checks passed	us-east-1
Target Service	i-0b37a2e181e7d1b0e	Stopped	2/2 checks passed	us-east-1

Below this, a detailed view for the Jenkins-Master instance (i-08500dd7c7a10a2bd) is shown. The "Status and alarms" tab is selected. The instance summary shows it has a Public IPv4 address (54.85.11.8) and a Private IPv4 address (172.31.31.51).

Sign in

The screenshot shows a web browser window with the URL 54.85.11.8:8080/login?from=%2F. The page title is "Sign in to Jenkins". It features a cartoon Jenkins head logo on the left. The form fields are as follows:

- Username: SalmanFarsi123
- Password: (redacted)
- Keep me signed in:
- Sign in button

This is Jenkins demo, previously have practiced

The screenshot shows the Jenkins dashboard. On the left, there are navigation links: New Item, Build History, Manage Jenkins, and My Views. The Build Queue section indicates "No builds in the queue". The Build Executor Status section shows:

Icon	S	M	L
Built-In Node	1	Idle	1
clawful	2	Idle	0

The main area displays build history for two jobs:

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀️	DownstreamJob	17 hr #2	N/A	21 ms
✓	☀️	UpstreamJob	17 hr #12	N/A	0.64 sec

Jenkins

Dashboard > UpstreamJob >

UpstreamJob

The Purpose of this Job Knowing About Upstream

Status Changes Workspace Build Now Configure Delete Project Rename

Downstream Projects

DownstreamJob

Permalinks

Build History trend Filter... /

#12 Jun 6, 2024, 1:02 PM

#11

Build in slave1

Dashboard > UpstreamJob > Configuration

Configure

General

Source Code Management Build Triggers Build Environment Build Steps Post-build Actions

Discard old builds GitHub project This project is parameterised Throttle builds Execute concurrent builds if necessary

Restrict where this project can be run Label Expression slave1

Advanced

Save Apply 53.02

Cloning the repo in the slave1 from the main branch

The screenshot shows the Jenkins configuration page for an upstream job. The left sidebar lists configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The 'Source Code Management' section is highlighted with a yellow box. The main panel is titled 'Git' and contains a 'Repositories' section. It shows a single repository with the URL <https://github.com/SalmanFarsi123/JenkinsMavenCode.git>. Below it is a 'Credentials' section showing '- none -'. A 'Branches to build' section specifies the branch specifier `*/main`. At the bottom are 'Save' and 'Apply' buttons.

Enabling github hook trigger, if any changes and commit happens then it will going to Build the Job

The screenshot shows the Jenkins configuration page for build triggers. The left sidebar lists configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The 'Build Triggers' section is highlighted with a yellow box. The main panel shows several trigger options: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', and 'GitHub hook trigger for GITScm polling', which is checked. At the bottom are 'Save' and 'Apply' buttons.

Use secret texts or files and using username and password(separated)

The screenshot shows the Jenkins configuration page for an upstream job. The 'Build Environment' section is highlighted with a yellow box. Inside, the 'Use secret text(s) or file(s)' checkbox is checked and highlighted with a yellow box. Below it, the 'Username and password (separated)' option is also highlighted with a yellow box. Other options like 'Certificate', 'Git Username and Password', 'SSH User Private Key', and 'Secret ZIP file' are listed but not selected.

Click Add > Jenkins (Remember) U need to have dockerhub account and using variables hiding username and password purpose.

The screenshot shows the Jenkins configuration page for an upstream job. The 'Build Environment' section is highlighted with a yellow box. A modal window titled 'Username and password (separated)' is open, showing 'Username Variable' set to 'DOCKER_USERNAME' and 'Password Variable' set to 'DOCKER_PASSWORD'. Below the modal, the 'Jenkins Credentials Provider' section shows a Jenkins icon and an 'Add' button. At the bottom of the configuration page, the 'Add timestamps to the Console Output' checkbox is checked and highlighted with a yellow box. Other options like 'Inspect build log for published build scans' and 'Terminate a build if it's stuck' are listed but not selected.

Kind: username and password and enter Your Dockerhub user name and password, ID and description, type anything which you want.

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

Treat username as secret ?

Password ?



I add my credentials

Jenkins Credentials Provider: Jenkins

Username ?

salmanfarsi

Treat username as secret ?

Password ?

ID ?

salman_cred

Description ?

docker hub credentials



We set up it.

The screenshot shows the Jenkins configuration interface for an 'UpstreamJob'. The top navigation bar indicates the path: Dashboard > UpstreamJob > Configuration. On the left, a sidebar lists several configuration sections: General, Source Code Management, Build Triggers, Build Environment (which is selected and highlighted with a yellow box), Build Steps, and Post-build Actions. The main content area is titled 'Configure' and contains several configuration options. One option is 'Use secret text(s) or file(s)', which is checked. Below this is a 'Bindings' section with fields for 'Username Variable' (set to 'DOCKER_USERNAME') and 'Password Variable' (set to 'DOCKER_PASSWORD'). Under 'Credentials', a radio button is selected for 'Specific credentials', and a dropdown menu shows an entry: 'salmanfarsi/***** (docker hub credentials)'. At the bottom of the configuration panel are 'Save' and 'Apply' buttons.

According to Jenkins task, u have keep this invoke top-level maven targets because anything developers add in the git hub, maven build going to change into executable format so keep it.

Now After come to Build steps u have to select execute shell and need to write script according to your project.

The screenshot shows the Jenkins configuration interface for an 'UpstreamJob'. The top navigation bar indicates the path: Dashboard > UpstreamJob > Configuration. On the left, a sidebar lists several configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps (which is selected and highlighted with a yellow box), and Post-build Actions. The main content area is titled 'Configure' and contains two build step configurations. The first step is 'Invoke top-level Maven targets' with 'Maven Version' set to 'maven' and 'Goals' set to 'clean install'. The second step is 'Execute shell' with the command 'pwd' entered into the text area. At the bottom of the configuration panel are 'Save' and 'Apply' buttons.

I am Launching my slave account, which is stopped, all build going to save and execute in this slave because I selected slave1

The screenshot shows the AWS CloudWatch Instances console. At the top, there are buttons for 'C' (Create), 'Connect', 'Instance state ▾', 'Actions ▾', 'Launch instances ▾', and a dropdown for 'All states ▾'. Below this is a search bar with placeholder text 'Find Instance by attribute or tag (case-sensitive)' and a 'Clear filters' button. A table lists one instance: 'Jenkins-Slave' (Instance ID: i-01a0d9d57466e4ca6, Status: Running, Instance type: t2.medium). The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability Zone (us-east-1e). Below the table, the instance details are shown: Instance ID i-01a0d9d57466e4ca6 (Jenkins-Slave), Public IPv4 address 54.236.12.43 (with a 'View address' link), and Private IPv4 addresses 172.31.50.152.

This was previous execution which I made in Jenkins demo

A terminal session with two tabs. The left tab is 'ubuntu@Jenkins-Master: ~' and the right tab is 'ubuntu@Jenkins-Slave: ~'. The Jenkins-Slave tab shows the following command output:

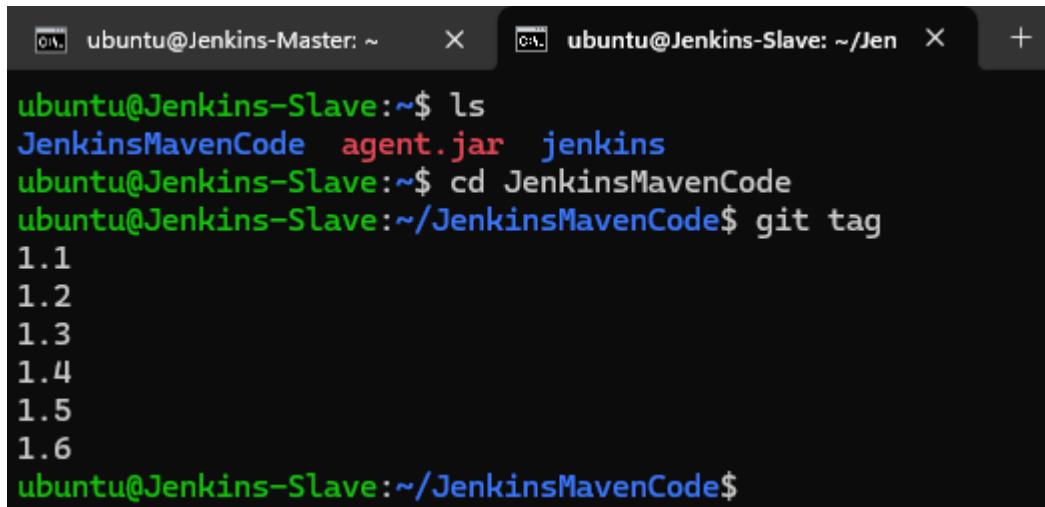
```
ubuntu@Jenkins-Slave:~$ ls
agent.jar jenkins
ubuntu@Jenkins-Slave:~$ cd jenkins
ubuntu@Jenkins-Slave:~/jenkins$ ls
remoting tools workspace
ubuntu@Jenkins-Slave:~/jenkins$ cd tools
ubuntu@Jenkins-Slave:~/jenkins/tools$ ls
hudson.tasks.Maven_MavenInstallation
ubuntu@Jenkins-Slave:~/jenkins/tools$ cd ~
ubuntu@Jenkins-Slave:~$
```

Copy Suhails Maven repository and Do Git Clone it will pull that all stuff of that repository.

A terminal session with two tabs. The left tab is 'ubuntu@Jenkins-Master: ~' and the right tab is 'ubuntu@Jenkins-Slave: ~'. The Jenkins-Slave tab shows the following command output:

```
ubuntu@Jenkins-Slave:~$ git clone https://github.com/suhailasad/JenkinsMavenCode.git
Cloning into 'JenkinsMavenCode'...
remote: Enumerating objects: 120, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 120 (delta 31), reused 21 (delta 20), pack-reused 83
Receiving objects: 100% (120/120), 166.11 KiB | 18.46 MiB/s, done.
Resolving deltas: 100% (36/36), done.
```

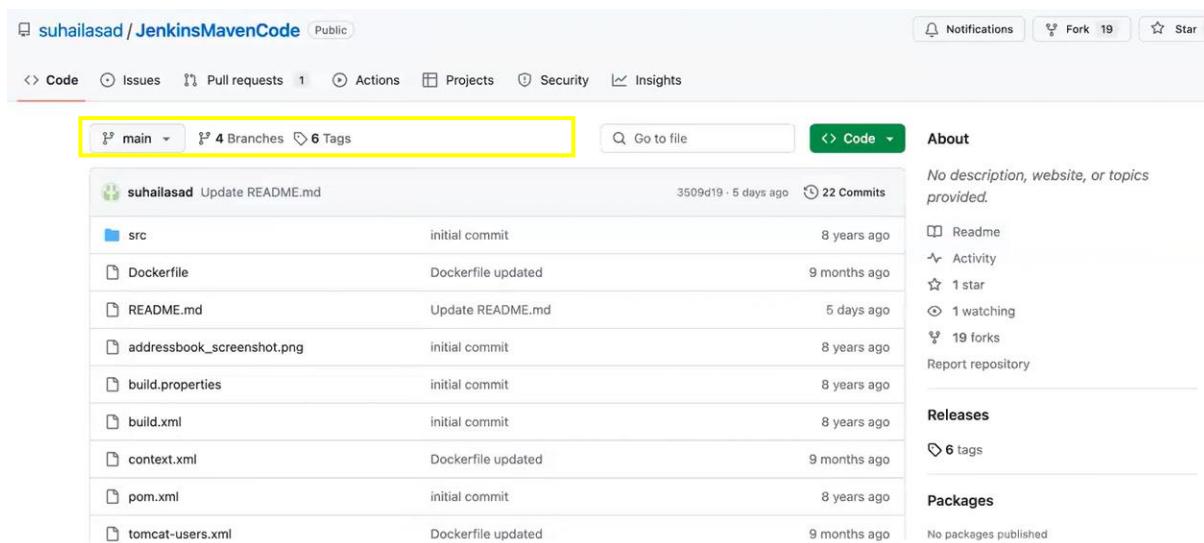
We can see the JenkinsMavenCode which is cloned from Suhail repo



A terminal session showing two windows. The left window shows the command `ls` outputting tags: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6. The right window shows the command `git tag` outputting the same tags.

```
ubuntu@Jenkins-Master: ~      X  ubuntu@Jenkins-Slave: ~/Jen  X  +
ubuntu@Jenkins-Slave:~$ ls
JenkinsMavenCode agent.jar jenkins
ubuntu@Jenkins-Slave:~$ cd JenkinsMavenCode
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git tag
1.1
1.2
1.3
1.4
1.5
1.6
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

We can see in Suhail repo has 4 branches and 6 tags



A GitHub repository page for `suhailasad/JenkinsMavenCode`. The page shows 4 branches and 6 tags. The main branch is selected. The repository has 22 commits, 1 star, 19 forks, and 6 tags. It includes sections for About, Releases, and Packages.

Code | **Issues** | **Pull requests** | **Actions** | **Projects** | **Security** | **Insights**

main | **4 Branches** | **6 Tags**

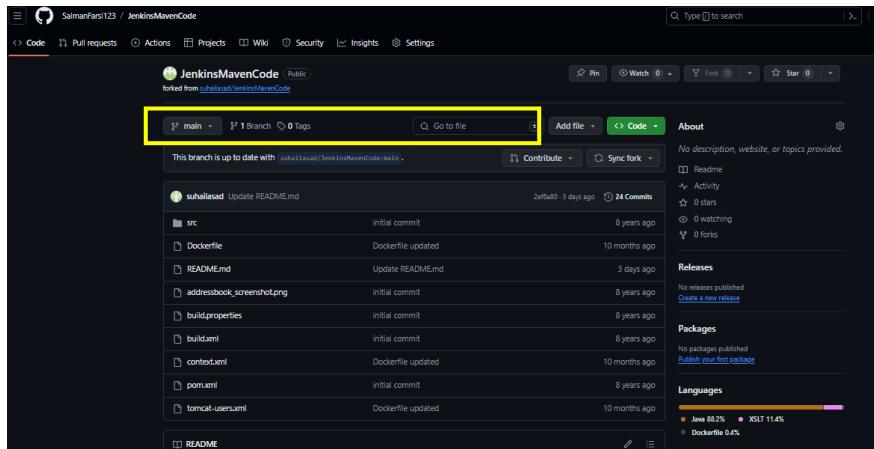
File	Commit Message	Time
src	initial commit	8 years ago
Dockerfile	Dockerfile updated	9 months ago
README.md	Update README.md	5 days ago
addressbook_screenshot.png	initial commit	8 years ago
build.properties	initial commit	8 years ago
build.xml	initial commit	8 years ago
context.xml	Dockerfile updated	9 months ago
pom.xml	initial commit	8 years ago
tomcat-users.xml	Dockerfile updated	9 months ago

About
No description, website, or topics provided.
Readme | Activity | 1 star | 1 watching | 19 forks | Report repository

Releases
6 tags

Packages
No packages published

But our repo have only 1 branch, we did this in Jenkins demo



The screenshot shows a GitHub repository named 'JenkinsMavenCode'. At the top, there's a navigation bar with options like 'Code', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. Below the bar, it says '1 Branch' and '0 Tags'. A yellow box highlights this area. To the right, there's an 'About' section with details like 'No description, website, or topics provided.', 'Activity' (0 stars, 0 watching, 0 forks), 'Releases' (none), 'Packages' (none), and 'Languages' (Java 88.2%, XSLT 11.4%, Dockerfile 0.4%).

Go to Inside the Maven and Follow all the commands

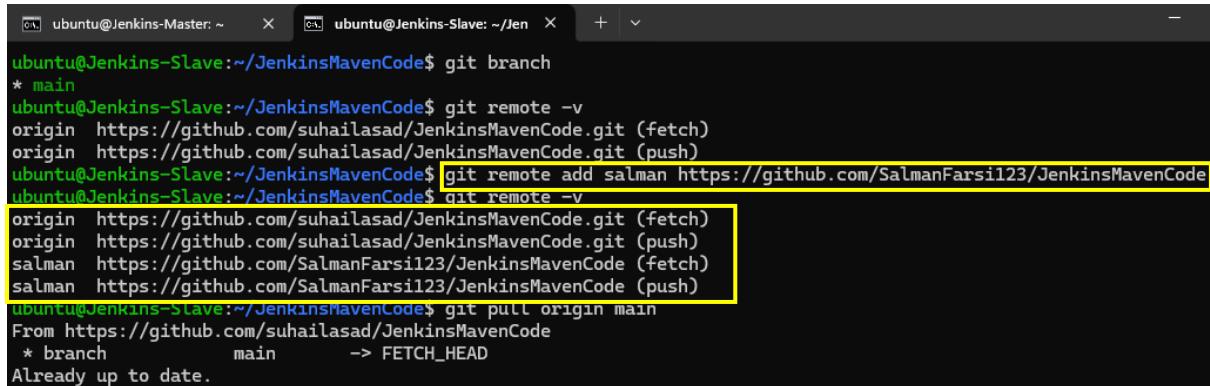
Check git remote -v which is we cloned suhailasad account it will shown

Now I am Adding My Github

Git remote add salman/replace your name Path of your github mavencode

Check git remote -v

Git pull origin main it show already up to date(Because we already clone)



```
ubuntu@Jenkins-Master: ~      x  ubuntu@Jenkins-Slave: ~/Jen  x  +  v
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git branch
* main
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git remote -v
origin https://github.com/suhailasad/JenkinsMavenCode.git (fetch)
origin https://github.com/suhailasad/JenkinsMavenCode.git (push)
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git remote add salman https://github.com/SalmanFarsi123/JenkinsMavenCode
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git remote -v
origin https://github.com/suhailasad/JenkinsMavenCode.git (fetch)
origin https://github.com/suhailasad/JenkinsMavenCode.git (push)
salman https://github.com/SalmanFarsi123/JenkinsMavenCode (fetch)
salman https://github.com/SalmanFarsi123/JenkinsMavenCode (push)
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git pull origin main
From https://github.com/suhailasad/JenkinsMavenCode
 * branch           main      -> FETCH_HEAD
Already up to date.
```

```
git branch git checkout feature git checkout featurebranch git checkout kubernetesdeployment
```

git branch will show all branches

```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git branch
* featurex
  main
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git checkout featurebranch
branch 'featurebranch' set up to track 'origin/featurebranch'.
Switched to a new branch 'featurebranch'
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git checkout kubernetesdeployment
branch 'kubernetesdeployment' set up to track 'origin/kubernetesdeployment'.
Switched to a new branch 'kubernetesdeployment'
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git branch
  featurebranch
  featurex
* kubernetesdeployment
  main
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

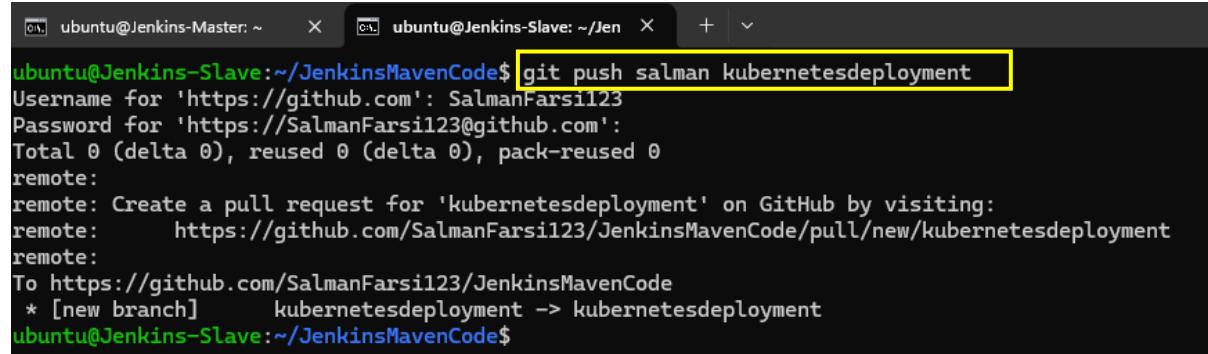
git push salman featurex and it will ask username and token of github

```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git push salman featurex
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'featurex' on GitHub by visiting:
remote:     https://github.com/SalmanFarsi123/JenkinsMavenCode/pull/new/featurex
remote:
To https://github.com/SalmanFarsi123/JenkinsMavenCode
 * [new branch]      featurex -> featurex
```

git push salman featurebranch and it will ask username and token of github

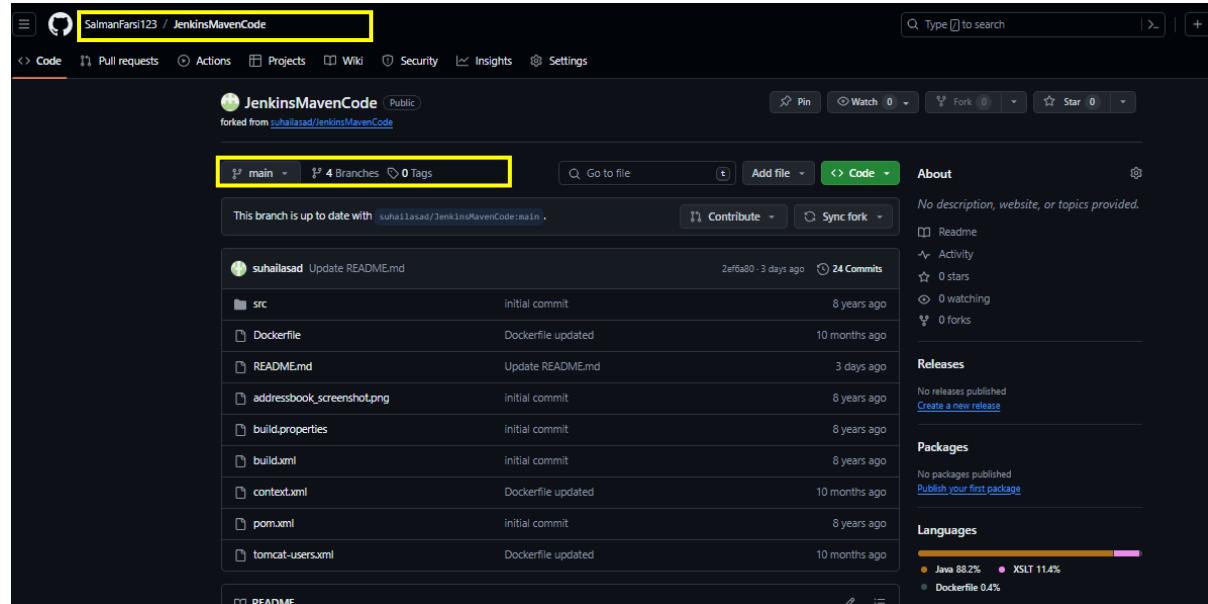
```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git push salman featurebranch
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'featurebranch' on GitHub by visiting:
remote:     https://github.com/SalmanFarsi123/JenkinsMavenCode/pull/new/featurebranch
remote:
To https://github.com/SalmanFarsi123/JenkinsMavenCode
 * [new branch]      featurebranch -> featurebranch
```

git push salman kubernetesdeployment and it will ask username and token of github and No need to Push Main branch because we already fork that branch.

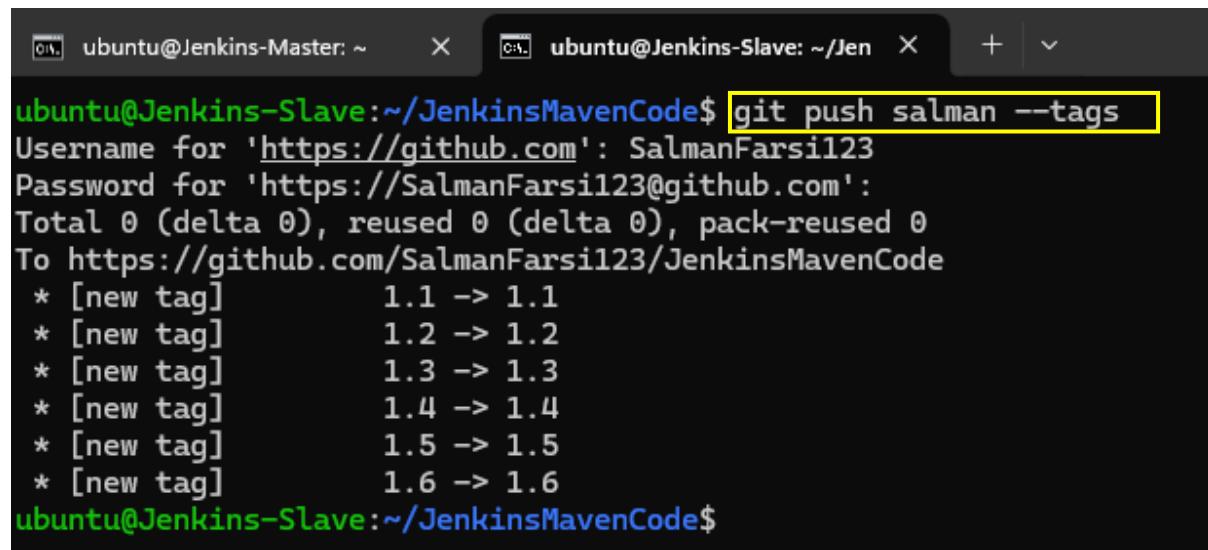


```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git push salman kubernetesdeployment
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'kubernetesdeployment' on GitHub by visiting:
remote:     https://github.com/SalmanFarsi123/JenkinsMavenCode/pull/new/kubernetesdeployment
remote:
To https://github.com/SalmanFarsi123/JenkinsMavenCode
 * [new branch]      kubernetesdeployment -> kubernetesdeployment
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

Now we can see in my account all 4 branches, but tags are missing



git push salman --tags and it will ask username and token of github and it will push all the tags.



```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git push salman --tags
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/SalmanFarsi123/JenkinsMavenCode
 * [new tag]      1.1 -> 1.1
 * [new tag]      1.2 -> 1.2
 * [new tag]      1.3 -> 1.3
 * [new tag]      1.4 -> 1.4
 * [new tag]      1.5 -> 1.5
 * [new tag]      1.6 -> 1.6
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

Now we are able to see all those stuff

The screenshot shows a GitHub repository page for 'JenkinsMavenCode'. The repository is public and forked from 'suhailasad/JenkinsMavenCode'. The main branch is 'main', which is up-to-date with 'suhailasad/JenkinsMavenCode:main'. There are 4 branches and 6 tags. A recent commit by 'suhailasad' updated 'README.md' on '2ef6a80' (3 days ago) with 24 commits.

Git rev-list -tags

```
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git rev-list --tags
76a1eac10cf9b5361e008c89c6b07ea496d2ad8
3285fcfeff276be573db016e4a48df141e7bcb789
00b8b6ae1c5a72979543aa305cf788bc8c25d3bb
15b162b80c344a1b176ed77874c998c3af0d8f04
9454d1c748f5ef82db5d3801e3a7182ac7beef87
c6cd4c895e057cbeddb4f9f612de44eb8e6e77e6
b3a6ec6fdffb5729dc36821497bf70785d45e5c9
428f77e5a97b09cbdf3d950b0178a8b8d6356f4a
1ee965fe055fa690002e46e2664c56872dfc200a
3f02c28107027befff14c79016925d53e62bd941f
7bda4815a5138519782c571012c89571fdbdff2
9559a109dcf6edb78396a84c29f86da54444207a
75de347665f6d29f1bd3198b96dc00728b361800
8acbb433e12aa0c7f486f31c7168821d2dff8913
55264047046480a073d5196732d7d0bd719c70a6
6af50251bcfe9274f21b508d8cde268e368cae7e
028a0532393520b14e12db4ccf638e0fdf7ed966
1621ce8fa0f8cdfc2cf78f97ba4a7b080b09775
725908313de10c0c4051be942625923df951c87f
c986b9eb275ae50c5a7b9b1d757be5c58e38c0da
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

Now this git tag 1.6 and 76a1eac10cf9b531e008c89c6b07ea496d2ad8 given name as TAG and the tag data stored in TAG with using linux commands

```
ubuntu@Jenkins-Master: ~      X  ubuntu@Jenkins-Slave: ~/Jen  X  +  v
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git rev-list --tags --max-count=1
76a1eac10cf9b531e008c89c6b07ea496d2ad8
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git describe --tags $(git rev-list --tags --max-count=1)
1.6
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ git describe --tags 76a1eac10cf9b531e008c89c6b07ea496d2ad8
1.6
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ TAG=$(git describe --tags $(git rev-list --tags --max-count=1))
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$ echo $TAG
1.6
ubuntu@Jenkins-Slave:~/JenkinsMavenCode$
```

Go to **Upstream > Configuration > Build Steps** Paste the Commands first the TAG data will and then Tag data will be print and going to build docker and salmanfarsi/addressbook repository make sure u need to have dockerhub account with have salmanfarsi/addressbook this name repository and paste docker login and paste dockhub username and password Below mentioned how to cover that password instead of giving directly.



The screenshot shows the Docker Hub interface. At the top, there's a search bar with the text "salmanfarsi / addressbook". Below the search bar, it says "Contains: Image" and "Last pushed: about 2 hours ago". The Docker Hub logo is prominently displayed in yellow. To the right, there are buttons for "Public" and "Scout inactive". Below the header, the URL "Dashboard > UpstreamJob > Configuration" is visible. The main area is titled "Configure" and "Build Steps". On the left, a sidebar lists "General", "Source Code Management", "Build Triggers", "Build Environment", "Build Steps" (which is selected and highlighted in blue), and "Post-build Actions". The "Build Steps" section contains a single step named "Execute shell". The "Command" field contains the following script:

```
pwd
TAG=$(git describe --tags $(git rev-list --tags --max-count=1))
echo $TAG
echo $TAG > TAGFILE.txt
docker build -t salmanfarsi/addressbook:$TAG .
```

At the bottom of the "Build Steps" section, there are "Save" and "Apply" buttons.

Now we can see username and password was covered. And pushing to dockerhub and docker logout.

The screenshot shows the Jenkins configuration interface for an 'UpstreamJob'. The left sidebar lists several configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps (which is selected and highlighted with a yellow box), and Post-build Actions. The main panel is titled 'Execute shell' and contains a command block. The command is:

```
echo $TAG > TAGFILE.txt  
docker build -t salmanfarsi/addressbook:$TAG .  
docker login -u$DOCKER_USERNAME -P$DOCKER_PASSWORD  
docker push salmanfarsi/addressbook:$TAG  
docker logout
```

Below the command block is an 'Advanced' dropdown menu.

Copy the Public Ip of Jenkins Master and 8080 port

The screenshot shows a browser window with the URL `18.207.1.111:8080/job/UpstreamJob/configure`. The address bar also shows other tabs like 'Interfacing SIM900...', 'Downloads', 'New Tab', 'Python Exercises: C...', and 'MEGA'.

Go to Your GitHub Account > Settings > Webhooks

The screenshot shows a GitHub repository page for 'SalmanFarsi123/JenkinsMavenCode'. The repository name is highlighted with a yellow box. The top navigation bar includes tabs for Code, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings (which is selected and highlighted with a yellow box). Below the repository name, it says 'JenkinsMavenCode (Public)' and 'forked from suhailasad/JenkinsMavenCode'. The main content area shows a branch selector with 'main' selected, 4 branches, and 6 tags. A message indicates the branch is up-to-date with 'suhailasad/JenkinsMavenCode:main'. There are also 'Pin', 'Watch', and 'Code' buttons.

Go to Your GitHub Account > Settings > Webhooks

The screenshot shows the GitHub repository settings for 'SalmanFarsi123 / JenkinsMavenCode'. The 'General' tab is selected. On the left sidebar, the 'Webhooks' option is highlighted with a yellow box. On the right, under the 'General' heading, there is a 'Repository name' field containing 'JenkinsMavenCode' with a 'Rename' button next to it. Below it are two checkboxes: 'Template repository' (unchecked) and 'Require contributors to sign off on web-based commits' (unchecked). A link to 'more about signing off on commits' is provided. The 'Default branch' section follows.

click Add Webhook in the right corner

The screenshot shows the GitHub repository settings for 'SalmanFarsi123 / JenkinsMavenCode'. The 'Webhooks' tab is selected. On the left sidebar, the 'Webhooks' option is highlighted with a yellow box. On the right, the 'Webhooks' section is displayed with a descriptive text about what webhooks do and a link to the 'Webhooks Guide'. At the top right of this section, there is a 'Add webhook' button with a yellow box around it.

Paste the Jenkins Public-IP and port No of Jenkins 8080 and give github-webhook/

The screenshot shows the GitHub settings interface for a repository named 'SalmanFarsi123 / JenkinsMavenCode'. The 'Webhooks' section is highlighted with a yellow box. In the 'Payload URL' field, the URL 'http://18.207.1.111:8080/github-webhook/' is entered. The 'Content type' dropdown is set to 'application/x-www-form-urlencoded'. A 'Secret' field is present but empty. Below the form, a question asks 'Which events would you like to trigger this webhook?' with a 'Default' link.

Add Successfully

The screenshot shows the GitHub settings interface for the same repository. The 'Webhooks' section now displays a single entry with a green checkmark and the URL 'http://18.207.1.111:8080/github-we... (push)'. There are 'Edit' and 'Delete' buttons next to the entry. The rest of the interface remains consistent with the previous screenshot.

Now see the 12 Builds Made already

The screenshot shows the Jenkins interface for the 'UpstreamJob' configuration. On the left, there's a sidebar with options: Status (highlighted), Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main area has a green checkmark icon and the title 'UpstreamJob'. Below it is a description: 'The Purpose of this Job Knowing About Upstream'. A section titled 'Downstream Projects' contains a green checkmark icon and the link 'DownstreamJob'. Under 'Permalinks', there's a bulleted list of recent builds: 'Last build (#12), 21 hr ago', 'Last stable build (#12), 21 hr ago', 'Last successful build (#12), 21 hr ago', and 'Last completed build (#12), 21 hr ago'. To the left, there's a 'Build History' card showing '#12 Jun 6, 2024, 1:02 PM'.

Now U can Make changes and I selected README.md

The screenshot shows the GitHub code editor for the file 'README.md' in the 'JenkinsMavenCode' repository. The file content is as follows:

```
1 ## README
2
3 ## README -webhook demo
4 Addressbook Tutorial
5 =====
```

Now I removed webhook demo and click commit changes

The screenshot shows the GitHub code editor after the '## README -webhook demo' line has been removed. The file content is now:

```
1 ## README
2
3 Addressbook Tutorial
4 =====
```

A yellow box highlights the removed line '## README -webhook demo'. A green box highlights the 'Commit changes...' button at the top right of the editor.

Its Build but I Got an Error

Dashboard > UpstreamJob >

>Status

</> Changes

Workspace

Build Now

Configure

Delete Project

GitHub Hook Log

Rename

UpstreamJob

The Purpose of this Job Knowing About Upstream

Permalinks

- Last build (#13), 11 min ago
- Last stable build (#12), 22 hr ago
- Last successful build (#12), 22 hr ago
- Last failed build (#13), 11 min ago
- Last unsuccessful build (#13), 11 min ago
- Last completed build (#13), 11 min ago

Build History trend ▾

Filter... /

#13

Dashboard > UpstreamJob > #15 > Console Output

Status

</> Changes

Console Output

View as plain text

Edit Build Information

Delete build '#15'

Polling Log

Timings

Git Build Data

← Previous Build

Console Output

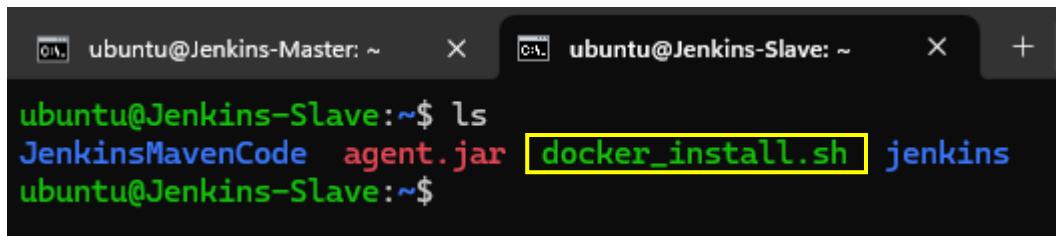
```
Started by GitHub push by SalmanFarsi123
Running as SYSTEM
Building remotely on slave1 in workspace /home/ubuntu/jenkins/workspace/UpstreamJob
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /home/ubuntu/jenkins/workspace/UpstreamJob/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/SalmanFarsi123/JenkinsMavenCode.git # timeout=10
Fetching upstream changes from https://github.com/SalmanFarsi123/JenkinsMavenCode.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/SalmanFarsi123/JenkinsMavenCode.git
+refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision b90f0dd2f0050a149a9b06b330932379c676c92 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f b90f0dd2f0050a149a9b06b330932379c676c92 # timeout=10
Commit message: "Update README.md"
```

[WARNING] Some problems were encountered while building the effective model for com.edurekademo.tutorial:addressbook:war:2.0
[WARNING] Reporting configuration should be done in <reporting> section, not in maven-site-plugin <configuration> as reportPlugins parameter. @ line 298, column 40
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[INFO] -----< com.edurekademo.tutorial:addressbook >-----
[INFO] Building Vaadin Addressbook example 2.0
[INFO] from pom.xml
[INFO] -----[war]-----
[INFO]
[INFO] --- clean:3.2.0:clean (default-clean) @ addressbook ---
[INFO] Deleting /home/ubuntu/jenkins/workspace/UpstreamJob/target
[INFO]
[INFO] --- enforcer:1.0:enforce (enforce-versions) @ addressbook ---
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ addressbook ---
[INFO] skip non existing resourceDirectory /home/ubuntu/jenkins/workspace/UpstreamJob/src/main/resources
[INFO]
[INFO] --- compiler:3.2:compile (default-compile) @ addressbook ---

Error I got is docker not found

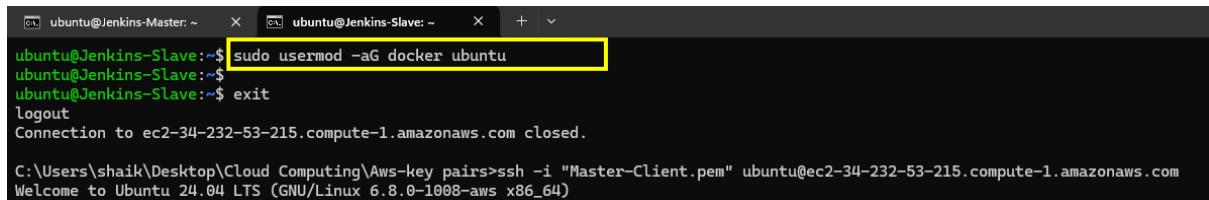
```
[UpstreamJob] $ /bin/sh -xe /tmp/jenkins12659865534336232270.sh
+ pwd
/home/ubuntu/jenkins/workspace/UpstreamJob
+ git rev-list --tags --max-count=1
+ git describe --tags 76a1eac10cf9b5361e008c89c6b07ea496d2ad8
+ TAG=1.6
+ echo 1.6
1.6
+ echo 1.6
+ docker build -t salmanfarsi/addressbook:1.6 .
/tmp/jenkins12659865534336232270.sh: 6: docker: not found
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

I installed dockers in Jenkins slave, gave permission and ran



```
ubuntu@Jenkins-Master: ~      X  ubuntu@Jenkins-Slave: ~      X  +
ubuntu@Jenkins-Slave:~$ ls
JenkinsMavenCode  agent.jar  docker_install.sh  jenkins
ubuntu@Jenkins-Slave:~$
```

Giving docker permission to ubuntu user



```
ubuntu@Jenkins-Master: ~      X  ubuntu@Jenkins-Slave: ~      X  +
ubuntu@Jenkins-Slave:~$ sudo usermod -aG docker ubuntu
ubuntu@Jenkins-Slave:~$ exit
Logout
Connection to ec2-34-232-53-215.compute-1.amazonaws.com closed.

C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Master-Client.pem" ubuntu@ec2-34-232-53-215.compute-1.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1008-aws x86_64)
```

I Updated once again, I thought maybe password is incorrect too.

Dashboard > UpstreamJob > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Execute shell

Command
See [the list of available environment variables](#)

```
pwd
TAG=$(git describe --tags $(git rev-list --tags --max-count=1))
echo $TAG
echo $TAG > TAGFILE.txt
docker build -t salmanfarsi/addressbook:$TAG .
docker login -u$DOCKER_USERNAME -P$DOCKER_PASSWORD
docker push salmanfarsi/addressbook:$TAG
docker logout
```

Again I updated because, I stopped the instances and again I launch then the public has been changed so, make sure on this

SalmanFarsi123 / JenkinsMavenCode

Code Pull requests Actions Projects Wiki Security Insights Settings

General

Access

- Collaborators
- Moderation options

Code and automation

- Branches
- Tags
- Rules
- Actions
- Webhooks

Repository name: JenkinsMavenCode [Rename](#)

Template repository
Template repositories let users generate new repositories with the same settings.

Require contributors to sign off on web-based commits
Enabling this setting will require contributors to sign off on commits to affirm that their commit complies with the repository's [more about signing off on commits](#).

Default branch

Old Jenkins master public IP

The screenshot shows the GitHub repository settings for 'SalmanFarsi123 / JenkinsMavenCode'. The 'Webhooks' tab is selected. A single webhook is listed with the URL `http://35.175.63.208:8080/github-webhook/`. There are 'Edit' and 'Delete' buttons next to it.

This is new public of Jenkins-Master

The screenshot shows a browser window with the URL `107.22.146.62:8080/job/UpstreamJob/configure`. The page title is 'UpstreamJob > Configuration'. The address bar shows other tabs like 'Interfacing SIM900...', 'Downloads', 'New Tab', etc.

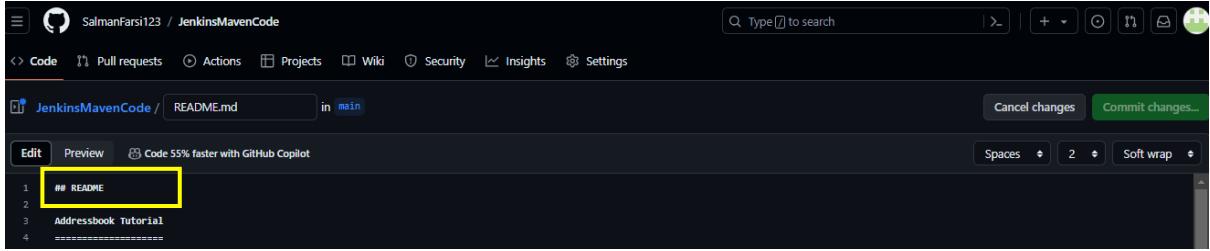
Updated Public Ip of Jenkins Master

The screenshot shows the GitHub repository settings for 'SalmanFarsi123 / JenkinsMavenCode'. The 'Webhooks' tab is selected. A new webhook is being added with the URL `http://107.22.146.62:8080/github-webhook/`. The 'Payload URL' field is highlighted with a yellow box.

Updated

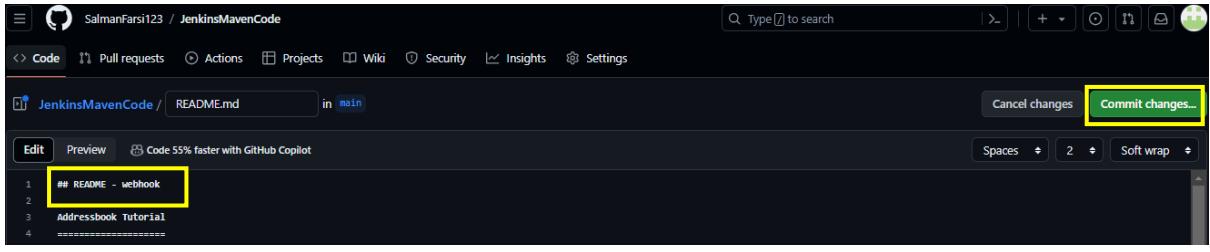
The screenshot shows the GitHub repository settings for 'SalmanFarsi123 / JenkinsMavenCode'. The 'Webhooks' tab is selected. Two webhooks are listed: one for the old IP (`http://35.175.63.208:8080/github-webhook/`) and one for the new IP (`http://107.22.146.62:8080/github-webhook/`). Both entries have 'Edit' and 'Delete' buttons.

Like this to 



The screenshot shows the GitHub code editor interface for a repository named 'JenkinsMavenCode'. A yellow box highlights the first line of the README.md file, which contains the text '# README'. The commit message at the bottom of the editor says 'Code 55% faster with GitHub Copilot'.

Like this I Make changes and commit changes



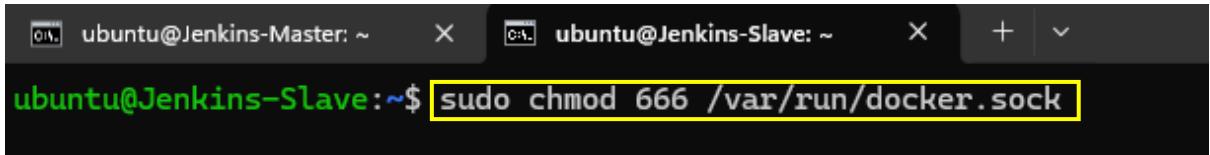
The screenshot shows the GitHub code editor interface for the same repository. A yellow box highlights the first line of the README.md file, which now includes the text '# README - webhook'. The 'Commit changes...' button in the top right corner is highlighted with a yellow box.

Again I got an error because of docker sock Permission



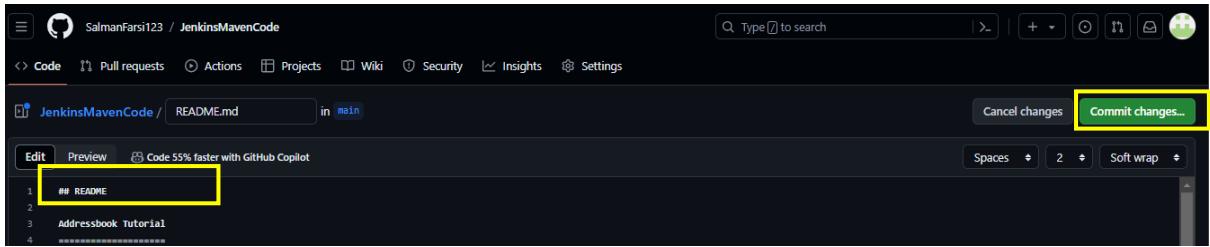
The screenshot shows the Jenkins console output for a build step. A yellow box highlights the error message: 'ERROR: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://127.0.0.1:4243/_ping": dial unix /var/run/docker.sock: connect: permission denied'. This indicates that the Jenkins slave process does not have the necessary permissions to access the Docker socket.

Go to Jenkins-slave and provide permissions



The screenshot shows a terminal window on a Jenkins slave machine. The command 'sudo chmod 666 /var/run/docker.sock' is being run to grant the necessary permissions to the Docker socket.

I Make changes and commit



The screenshot shows the GitHub code editor interface for the repository. A yellow box highlights the first line of the README.md file, which now includes the text '# README - webhook'. The 'Commit changes...' button in the top right corner is highlighted with a yellow box.

Again got error because of capital 'P' in password part

Dashboard > UpstreamJob > #17 > Console Output

```
#8 DONE 0.0s

#9 [5/6] ADD tomcat-users.xml /usr/local/tomcat/conf/tomcat-users.xml
#9 DONE 0.0s

#10 [6/6] ADD target/addressbook.war /usr/local/tomcat/webapps/
#10 DONE 0.1s

#11 exporting to image
#11 exporting layers
#11 exporting layers 0.2s done
#11 writing image sha256:6764785665d267188b17b112a9ee10908cbe2fe03a17893dc7bfb9155a3383ae done
#11 naming to docker.io/salmanfarsi/addressbook:1.6 done
#11 DONE 0.2s
+ docker login -usalmansfarsi -P*****
unknown shorthand flag: 'P' in '-P*****'
See 'docker login --help'.
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

Change into small p

Dashboard > UpstreamJob > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Command

See [the list of available environment variables](#)

```
pwd
TAG=$(git describe --tags $(git rev-list --tags --max-count=1))
echo $TAG
echo $TAG > TAGFILE.txt
docker build -t salmanfarsi/addressbook:$TAG .
docker login -u$DOCKER_USERNAME -p$DOCKER_PASSWORD
docker push salmanfarsi/addressbook:$TAG
docker logout
```

Once again I make changes

SalmanFarsi123 / JenkinsMavenCode

Code Pull requests Actions Projects Wiki Security Insights Settings

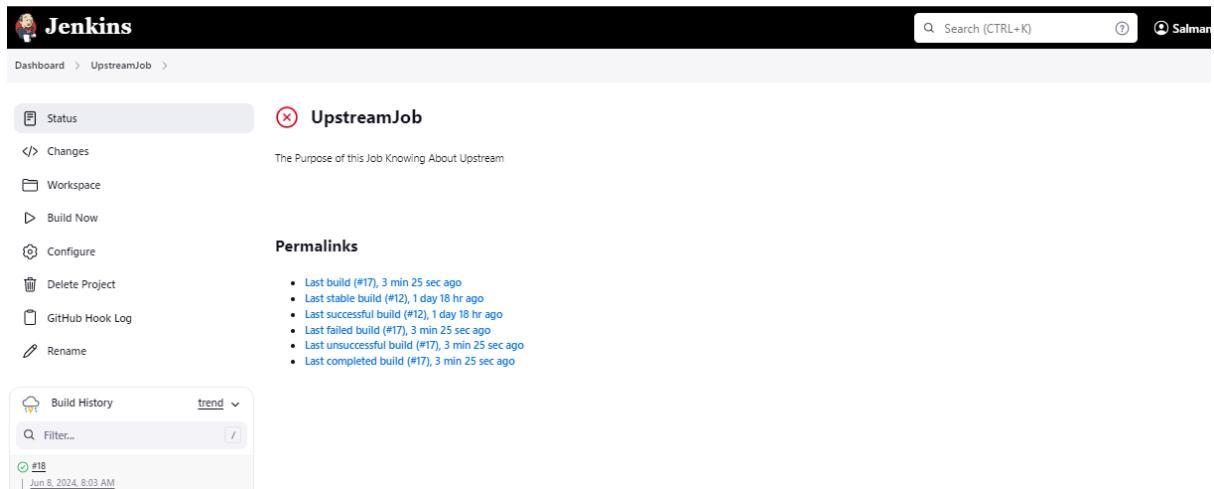
JenkinsMavenCode / README.md in main

Commit changes...

README - webhook

Addressbook Tutorial

The job 18 ran successfully



Jenkins

Dashboard > UpstreamJob >

Status

UpstreamJob

The Purpose of this Job Knowing About Upstream

Changes

Workspace

Build Now

Configure

Delete Project

Github Hook Log

Rename

Permalinks

- Last build (#17), 3 min 25 sec ago
- Last stable build (#12), 1 day 18 hr ago
- Last successful build (#12), 1 day 18 hr ago
- Last failed build (#17), 3 min 25 sec ago
- Last unsuccessful build (#17), 3 min 25 sec ago
- Last completed build (#17), 3 min 25 sec ago

Build History

Filter... trend ▾

#18 | Jun 8, 2024, 8:03 AM

Now its pushed to my docker hub account



Dashboard > UpstreamJob > #18 > Console Output

```
2f140462f3bc: Preparing
63c99163f472: Preparing
ccdbb80308cc: Preparing
225e381f3879: Waiting
7c845aef85c9: Waiting
d9210666eabd: Waiting
f67287db387a: Waiting
2f140462f3bc: Waiting
63c99163f472: Waiting
ccdbb80308cc: Waiting
1e85ed7274f1: Waiting
ff4127f9a834: Pushed
5a2a9c899bfa: Pushed
16144e7b9e0c: Pushed
225e381f3879: Layer already exists
d9210666eabd: Layer already exists
7c845aef85c9: Layer already exists
f67287db387a: Layer already exists
1e85ed7274f1: Layer already exists
2f140462f3bc: Layer already exists
63c99163f472: Layer already exists
e1034d8179a5: Pushed
ccdbb80308cc: Layer already exists
549e14c3b00a: Pushed
1.6: digest: sha256:1f8e7af853365870496e1bf9fd224837ee4b5c08db8722775a5d12ec5df5f8cf size: 3041
+ docker logout
Removing login credentials for https://index.docker.io/v1/
Finished: SUCCESS
```

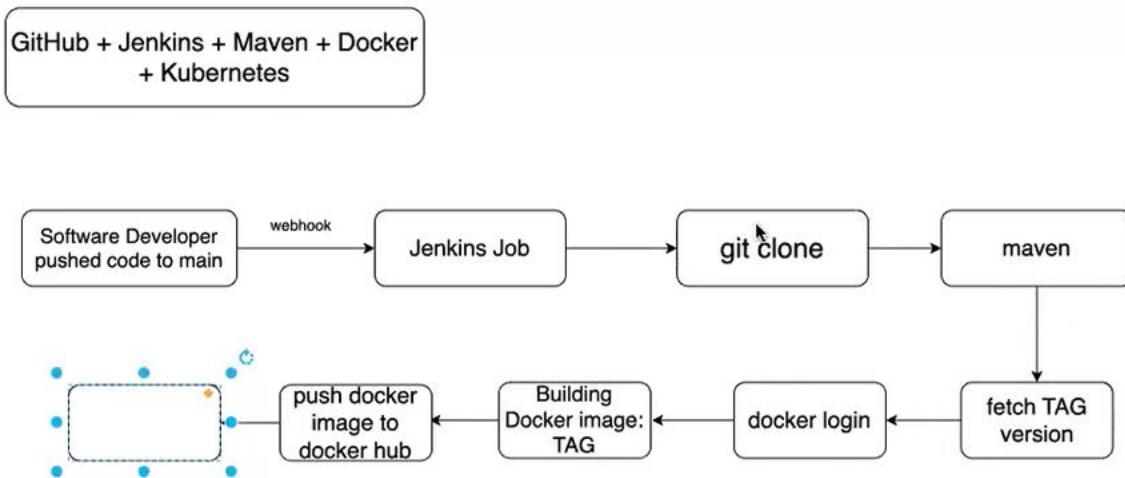
We can see the Image as been pushed into the docker Hub

The screenshot shows the Docker Hub interface for the repository `salmanfarsi/addressbook`. At the top, there are tabs for Explore, Repositories (which is selected), Organizations, and a search bar. Below the search bar, it says "Using 0 of 1 private repositories. [Get more](#)". The repository details show it was updated 1 minute ago, has no description, and no category. A "Docker commands" section includes a "Public View" button and a command line box with the text `docker push salmanfarsi/addressbook:tagname`. On the left, a "Tags" section lists one tag: `1.6`. On the right, an "Automated Builds" section explains how to automatically push images to the Hub.

FOLLOW THIS IN ALL INTEGRATION TOOLS PDF:

THIS IS INCOMPLETE TOPIC, FOLLOW THE NEXT CLASS

I Captured the Suhail practiced things but it was in complete

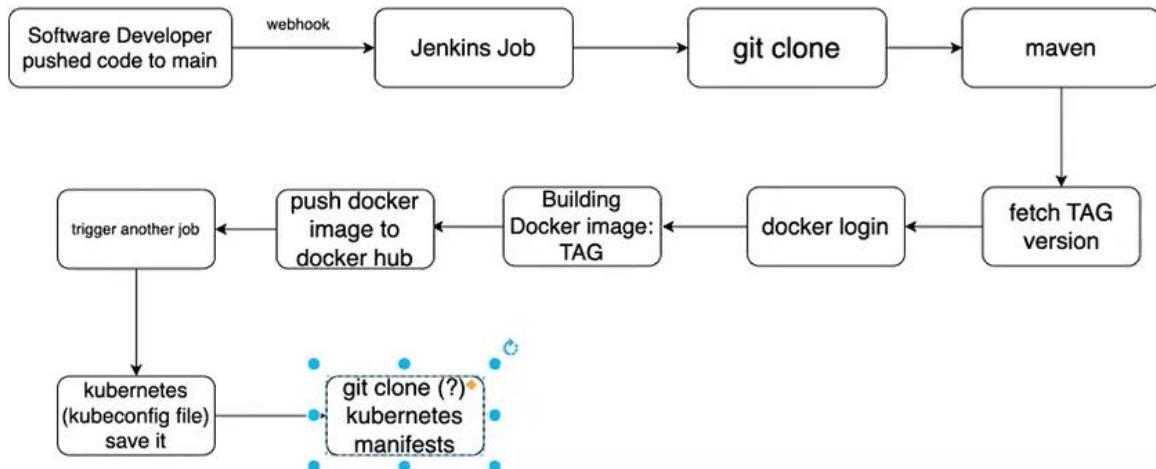


```

ubuntu@JenkinsMa:~/JenkinsMa X  ubuntu@Kubernetes-Master: ~ X + | v
ubuntu@Kubernetes-Master:~$ cd .kube/
ubuntu@Kubernetes-Master:~/ .kube$ ls
cache config
ubuntu@Kubernetes-Master:~/ .kube$ cat config
apiVersion: v1
clusters:
- cluster:
  certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0E
kIodmNOQVFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN
pFekFSQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0l
i

File Edit View
File.txt git token.txt suhai asad dockers_inst suhai asad git token.txt git token.txt git token.txt config X
apiVersion: v1
clusters:
- cluster:
  certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0E
kIodmNOQVFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN
pFekFSQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0
iI0MRRRUJBUVVBOTR0kR3QxdnZ0VLckFvSU8UURRa095M0kwBEl1VNINW1zVUtXSGE0WRjWnRaZDB2MFcrd3T2aXRJSTVxdwJPYmJRC8xV1JQUHQKcGROlmjtby9id3Q
0Mj1jQjBTNWZS05EWm1mb5tNwV1zNmVxaEdaQktPMG1zQXB25Ct0an1xTGUsxmJnYuhMVgpJdU1oRmdTY3V2UE1TWDA4VnhXTXAvTm43OX1oM1RuWZkd2FU0ndyb2NEVp0
QncycSjVMWFdMNGVCL3NKeftkC1k0ekQydw5ERkVzVjf2OS92R292a43aTVibHRjVfISRWFBR1puSGtDefJPK0ZGV1BqMwRMwlpqVUN2UIIvNTYKUmVodH1Tdk15eFnTqU0xa
GMM0VRmeHE1R1dCNzd1dzlsMhdZdEtnK31pT3RJSEtaK0xkcGhucDxe1JoaEJSZQpTUjNWSWJ2UDJNU1grWGJGWrXYE1hRFp4Wk1MQWdNQkFBR2pXVeJYTUE0R0ExVREd0
VCL3dRRUF3SUWnREFQCKjnTIZUK1CQWY4RUUQURBUUgvTUIwR0ExVREZ1FXQkJUQT1COUtSNnF3am1YdEw5YkUhU1A4TjErN3pqQVYKQmd0VkhSRUVEakFNZ2dwcmrXSmx
jbTVsZEdWek1BMeDU3FHU01iM0RRRUJDd1VBQTRJQkFRQS9GMmZU0F6UgpkYKTBDd11b3EvSkTV21QUJNlvoGdmMS90Q00vUExPSXR1cz1B0Xo3N2FDNInhw059QbmF35U9D
UUw3akNLMjV0C1A4RVYweGxGZjE4MU5hNGdDS3RcczRXOU3ja21FR3NyejVY2TFxaJVBV3BnTmp6SzR2VTrn0VNxs3kvSEt5QmcKvk5nNU1yZw84LzQ1akFRRH1UWdTSGJrd
EZBQ1Zebzg4MEF1QjdWcUF5QUFxMxEveE5kNVJ1UXYyeGdyUT1LcQo1cWQ2NnPNSDZuZlQ3UGsyWmZmTQG0S1Z5cEdxQnM2V2FKYVE0TQ5SFh0TmJGsdnjUFpQK1Q0duS0j
IwL1NGCnQzdnIvnln0ZnN4MX1QQ2JnVFnDtZuSEVSeE83TURQMs1QkpZOW1BRzFnWjZoUDZvSFJrVhPyNlbT1V0DcKcEzoMctmNzZERFZ0Ci0tLS0tRU5E1ENFU1RJRk1
DQVRFLS0tLS0K
  server: https://172.31.62.130:6443
  name: kubernetes
contexts:

```



SUHAILASAD PRACTISED STUFF

<https://github.com/suhailasad/JenkinsMavenCode>

suhailasad / JenkinsMavenCode

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

JenkinsMavenCode Public

main 4 Branches 6 Tags

Switch branches/tags

Find or create a branch...

Branches Tags

- main default
- featurebranch
- featurex
- kubernetesdeployment

initial commit 8 years ago

Dockerfile updated 9 months ago

Update README.md 14 minutes ago

initial commit 8 years ago

initial commit 8 years ago

<https://github.com/suhailasad/JenkinsMavenCode/tree/kubernetesdeployment>

JenkinsMavenCode Public

kubernetesdeployment 4 Branches 6 Tags

This branch is 4 commits ahead of, 23 commits behind main.

Contribute

suhailasad Update test 302040f · 2 months ago 4 Commits

README.md initial commit 4 months ago

deployment.yaml initial commit 4 months ago

nodeport.yaml initial commit 4 months ago

test Update test 2 months ago

README

<https://github.com/suhailasad/JenkinsMavenCode/blob/kubernetesdeployment/deployment.yaml>

The screenshot shows a GitHub repository page for 'JenkinsMavenCode'. The left sidebar lists files: README.md, deployment.yaml (selected), nodeport.yaml, and test. The main area displays the 'deployment.yaml' file content:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: customerinformation-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: addressbook
  template:
    metadata:
      labels:
        app: addressbook
    spec:
      containers:
        - name: addressbook-container
          image: suhailasad/addressbook:1.2
      ports:
        - containerPort: 8080
```

Below the code, a commit history is shown for the file:

JenkinsMavenCode / deployment.yaml

suhailasad initial commit

Code Blame 19 lines (19 loc) · 384 Bytes Code 55% faster with GitHub Copilot

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: customerinformation-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: addressbook
  template:
    metadata:
      labels:
        app: addressbook
    spec:
      containers:
        - name: addressbook-container
          image: suhailasad/addressbook:1.2
      ports:
        - containerPort: 8080
```

https://github.com/suhailasad/JenkinsMavenCode/blob/kubernetesdeployment/nodeport.yaml

Files

- README.md
- deployment.yaml
- nodeport.yaml**
- test

JenkinsMavenCode / nodeport.yaml

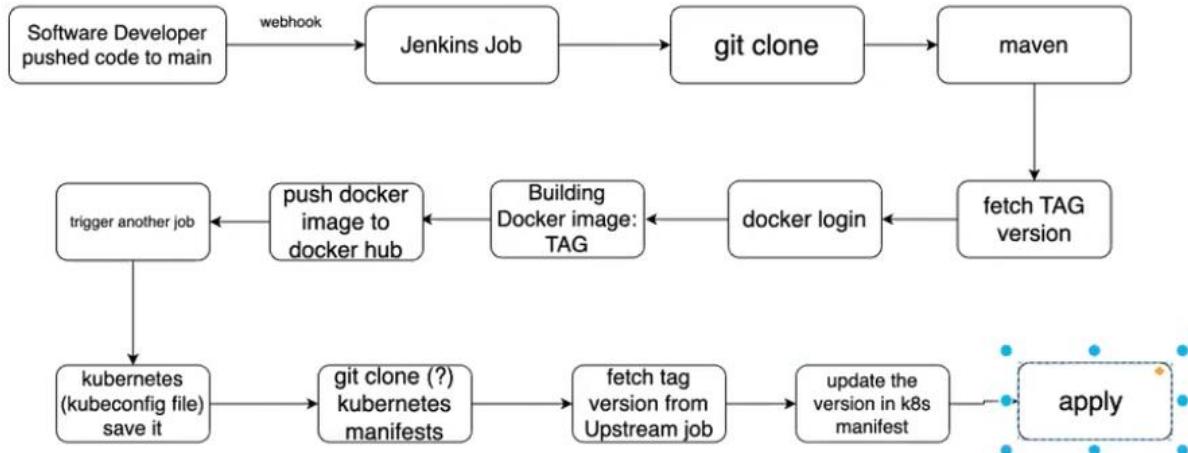
suhailasad initial commit

Code Blame 13 lines (13 loc) · 221 Bytes Code 55% faster with GitHub Copilot

```

1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: customerinformation-nodeport
5  spec:
6    selector:
7      app: addressbook
8    type: NodePort
9    ports:
10      - protocol: TCP
11        port: 80
12        targetPort: 8080
13        nodePort: 30089

```



Not Secure http://3.129.5.33:8080/job/UpstreamJob/8/console

Dashboard > UpstreamJob > #8 > Console Output

Timestamps View as plain text

System clock time Use browser timezone Elapsed time None

```

03:00:53 # [1/6] FROM docker.io/library/tomcat:9.0.45-jdk11-adoptopenjdk-hotspot@sha256:b76071b553b248ea9560636e75c444b6b28b21c4520f6801cccef9740633bf7
03:00:53 #4 resolve docker.io/library/tomcat:9.0.45-jdk11-adoptopenjdk-hotspot@sha256:b76071b553b248ea9560636e75c444b6b28b21c4520f6801cccef9740633bf7 0.0s done
03:00:53 #4 ...
03:00:53 #5 [internal] load build context
03:00:53 #5 transferring context: 16.54MB 0.2s done
03:00:53 #5 DONE 0.2s
03:00:53
03:00:53 #4 [1/6] FROM docker.io/library/tomcat:9.0.45-jdk11-adoptopenjdk-hotspot@sha256:b76071b553b248ea9560636e75c444b6b28b21c4520f6801cccef9740633bf7 1.21kB / 1.21kB done
03:00:53 #4 sha256:b76071b553b248ea9560636e75c444b6b28b21c4520f6801cccef9740633bf7 0.0s done
03:00:53 #4 sha256:e7281e9ea44afc5e4c2e5b44f64d1d7fd0e483d415c4e80dbfacc29b238e8d 2.00kB / 2.00kB done
03:00:53 #4 sha256:345e3491a907bb7c6f1bddcfa494284b8b6dd77eb7d93f09432b17b20f2bbe 9.44MB / 28.54MB 0.2s
03:00:53 #4 sha256:5e9250ddb7d0fa6d13302c7c3e6a0aa40390e42424caed1e5289077ee4054709 187B / 187B 0.2s done
03:00:53 #4 sha256:57671312f61fdecf340efed0fb0863350c886e92bffd7978add02afc5c3 851B / 851B 0.1s done
03:00:53 #4 sha256:592e2c2d7c1370aca6e4f48ee1d20fb3a4c148db400ad5cd575778c681f5dba0 0B / 16.03MB 0.2s
03:00:53 #4 sha256:ce0a2410aba2adb3fc36d0afe53fdb794de060dfa24ff52424735d165af5ee9 0B / 193.65MB 0.2s
03:00:53 #4 sha256:345e3491a907bb7c6f1bddcfa494284b8b6dd77eb7d93f09432b17b20f2bbe 24.12MB / 28.54MB 0.4s
03:00:53 #4 sha256:592e2c2d7c1370aca6e4f48ee1d20fb3a4c148db400ad5cd575778c681f5dba0 6.29MB / 16.03MB 0.4s
03:00:53 #4 sha256:345e3491a907bb7c6f1bddcfa494284b8b6dd77eb7d93f09432b17b20f2bbe 28.54MB / 28.54MB 0.5s done
03:00:53 #4 sha256:592e2c2d7c1370aca6e4f48ee1d20fb3a4c148db400ad5cd575778c681f5dba0 16.03MB / 16.03MB 0.6s
03:00:53 #4 sha256:ce0a2410aba2adb3fc36d0afe53fdb794de060dfa24ff52424735d165af5ee9 15.73MB / 193.65MB 0.6s
03:00:53 #4 sha256:7267948abfc47d400ccf2e9273d7a6ce5ecea0dbb0dz27d250e2200954db0 0B / 172B 0.6s
03:00:53 #4 sha256:592e2c2d7c1370aca6e4f48ee1d20fb3a4c148db400ad5cd575778c681f5dba0 16.03MB / 16.03MB 0.6s done

```

← → ⌂ Not Secure http://3.129.5.33:8080/job/UpstreamJob/8/console

Dashboard > UpstreamJob > #8 > Console Output

Timestamps [View as plain text](#)

- System clock time
- Use browser timezone
- Elapsed time
- None

```

03:01:07 1e85ed7274f1: Waiting
03:01:07 2f140462f3bc: Waiting
03:01:07 63c99163f472: Waiting
03:01:07 cddb803080cc: Waiting
03:01:07 225e381f3879: Waiting
03:01:07 7c845aef85c9: Waiting
03:01:09 86570429c0f6: Pushed
03:01:09 bla@e5cd63e9: Pushed
03:01:09 225e381f3879: Layer already exists
03:01:09 7c845aef85c9: Layer already exists
03:01:09 d2813231739a: Pushed
03:01:09 349643f57b88: Pushed
03:01:09 f67287db387a: Layer already exists
03:01:09 2f140462f3bc: Layer already exists
03:01:09 cddb803080cc: Layer already exists
03:01:09 63c99163f472: Layer already exists
03:01:09 63c99163f472: Layer already exists
03:01:10 1e85ed7274f1: Layer already exists
03:01:10 ade55b3e7343: Pushed
03:01:12 1.6: digest: sha256:aa583a13301e41f657e9ee8a85fd1022d72f3c6502bdf1acac592f0bfed16304 size: 3041
03:01:12 + docker logout
03:01:12 Removing login credentials for https://index.docker.io/v1/
03:01:12 Finished: SUCCESS

```

← → ⌂ Not Secure http://3.129.5.33:8080/job/DownstreamJob/configure

Jenkins

Dashboard > DownstreamJob >

Status [\(green\)](#) DownstreamJob

</> Changes

Workspace

Build Now

Configure

Delete Project

Rename

Permalinks

- Last build (#1), 5 days 11 hr ago
- Last stable build (#1), 5 days 11 hr ago
- Last successful build (#1), 5 days 11 hr ago
- Last completed build (#1), 5 days 11 hr ago

← → ⌂ Not Secure http://3.129.5.33:8080/job/DownstreamJob/configure

Dashboard > DownstreamJob > Configuration

Configure

Throttle builds ?

Execute concurrent builds if necessary ?

Restrict where this project can be run ?

Label Expression ?

slave1

Label slave1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Advanced

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Source Code Management

← → ⌂ Not Secure http://3.129.5.33:8080/job/DownstreamJob/configure

Dashboard > DownstreamJob > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

slave1

Label slave1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Advanced ▾

Source Code Management

None

Git ?

Repositories ?

Repository URL ?

https://github.com/suhailasad/JenkinsMavenCode.git

! Please enter Git repository.

Credentials ?

- none -

← → ⌂ Not Secure http://3.129.5.33:8080/job/DownstreamJob/configure

Dashboard > DownstreamJob > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

+ Add ▾

Advanced ▾

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

* /kubernetesdeployment

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add ▾

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Build Triggers

Trigger builds remotely (e.g., from scripts) ?

Build after other projects are built ?

Projects to watch

UpstreamJob,

No such project 'u': Did you mean 'UpstreamJob'?

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build periodically ?

Advanced Options

Configure

General

Source Code Management

Build Environment

Build Steps

Post-build Actions

Build Environment

Delete workspace before build starts

Use secret text(s) or file(s) ?

Bindings

Secret file ?

Variable ?

CONFIGFILE

Credentials ?

Specific credentials Parameter expression

+ Add

Jenkins

Add

Add timestamps to the Console Output

Inspect build log for published build scans

Terminate a build if it's stuck

Save Apply

1:56:30

Jenkins Credentials Provider: Jenkins

Domain

Global credentials (unrestricted)

Kind

- ✓ Username with password
- GitHub App
- SSH Username with private key
- Secret file**
- Secret text
- Certificate

Username ?

Treat username as secret ?

Password ?

Jenkins Credentials Provider: Jenkins

Secret file

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

File

Choose file config

ID ?

KUbeconfig

Description ?

kubernetes kubeconfig file

Cancel

Add

Not Secure http://3.129.5.33:8080/job/DownstreamJob/configure

Dashboard > DownstreamJob > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

+ Add ▾

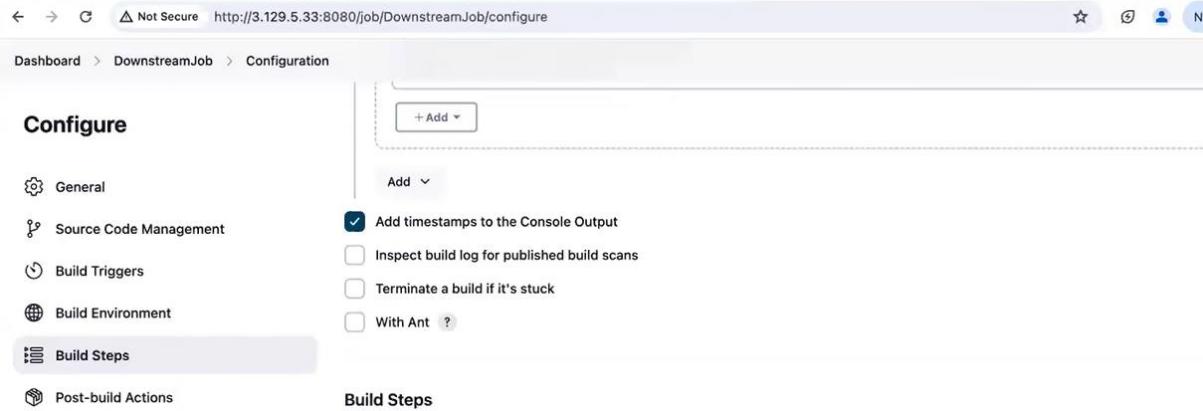
Add timestamp to the Console Output

Inspect build log for published build scans

Terminate a build if it's stuck

With Ant ?

Build Steps

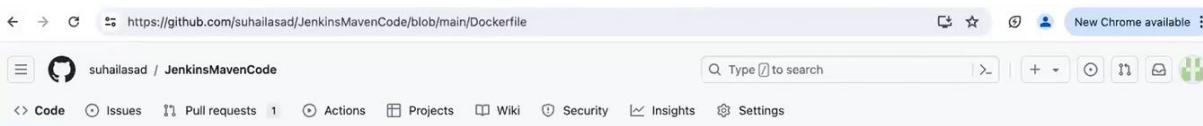


https://github.com/suhailasad/JenkinsMavenCode/blob/main/Dockerfile

suhailasad / JenkinsMavenCode

Type ⌘ to search

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings



JenkinsMavenCode Public

main 4 Branches 6 Tags Go to file Add file Code

suhailasad Update README.md f15bdae · 28 minutes ago 23 Commits

src initial commit 8 years ago

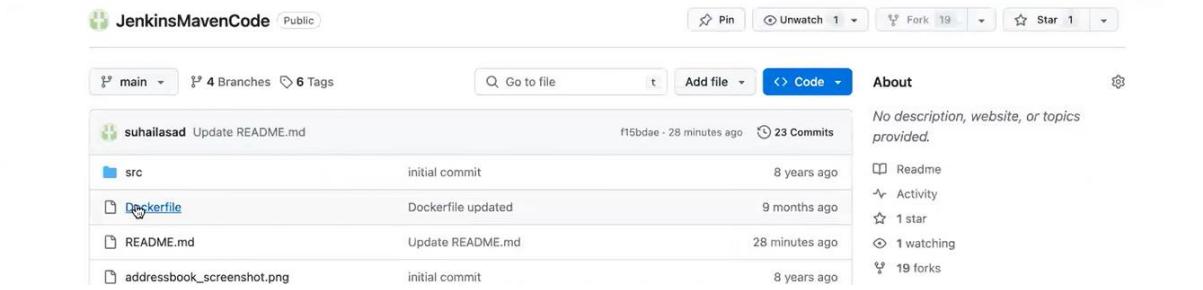
Dockerfile Dockerfile updated 9 months ago

README.md Update README.md 28 minutes ago

addressbook_screenshot.png initial commit 8 years ago

About No description, website, or topics provided.

Readme Activity 1 star 1 watching 19 forks



https://github.com/suhailasad/JenkinsMavenCode/blob/main/Dockerfile

suhailasad / JenkinsMavenCode

Type ⌘ to search

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

Files main + Go to file

src Dockerfile README.md addressbook_screenshot.png build.properties build.xml context.xml pom.xml tomcat-users.xml

JenkinsMavenCode / Dockerfile

suhailasad Dockerfile updated 7259083 · 9 months ago History

Code Blame 9 lines (7 loc) · 303 Bytes Code 55% faster with GitHub Copilot

```
1 FROM tomcat:9.0.45-jdk11-adoptopenjdk-hotspot
2 RUN mv webapps webapps2
3 RUN mv webapps.dist/ webapps
4 ADD context.xml /usr/local/tomcat/webapps/manager/META-INF/context.xml
5 ADD tomcat-users.xml /usr/local/tomcat/conf/tomcat-users.xml
6 ADD target/addressbook.war /usr/local/tomcat/webapps/
7 EXPOSE 8080
```

Raw ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

