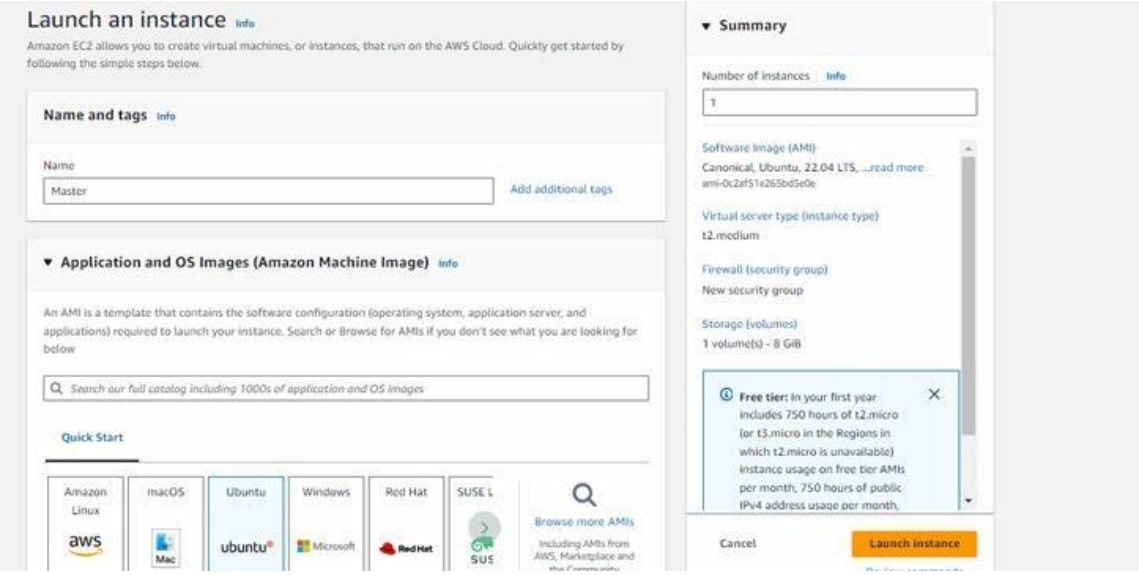
ADVANCE DEVOPS EXP 3

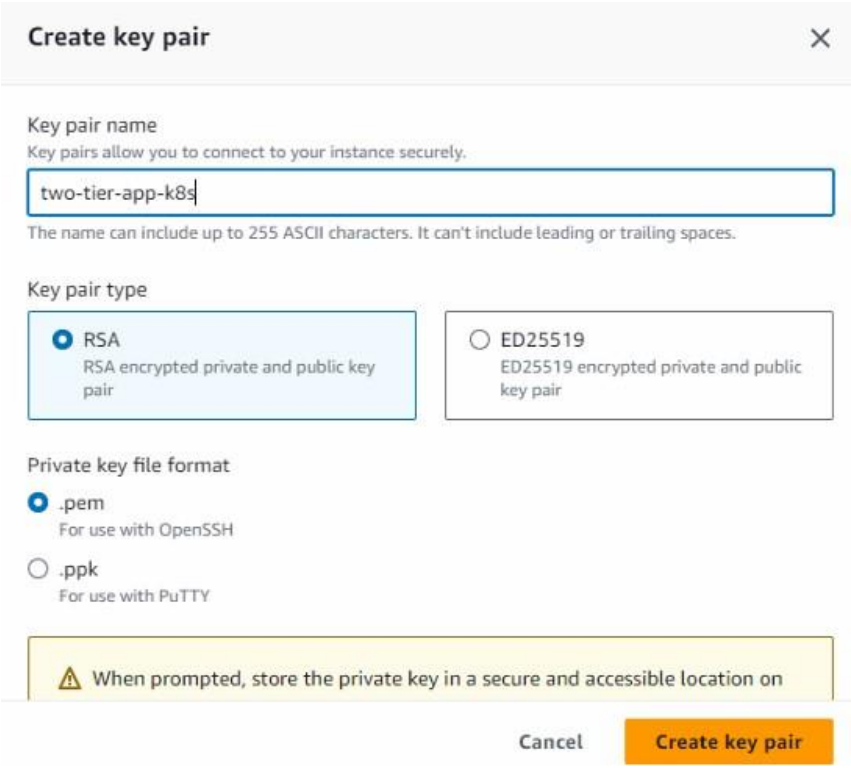
Name :- Ronit Santwani
Roll no :- 49

Aim:-
To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

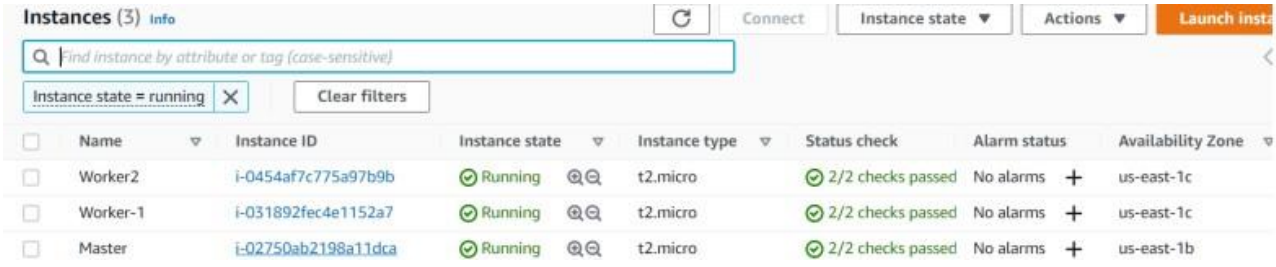
Step 1: Pre-requisites

1.1 Create 3 EC2 instances,one for the master node and two for the worker nodes.

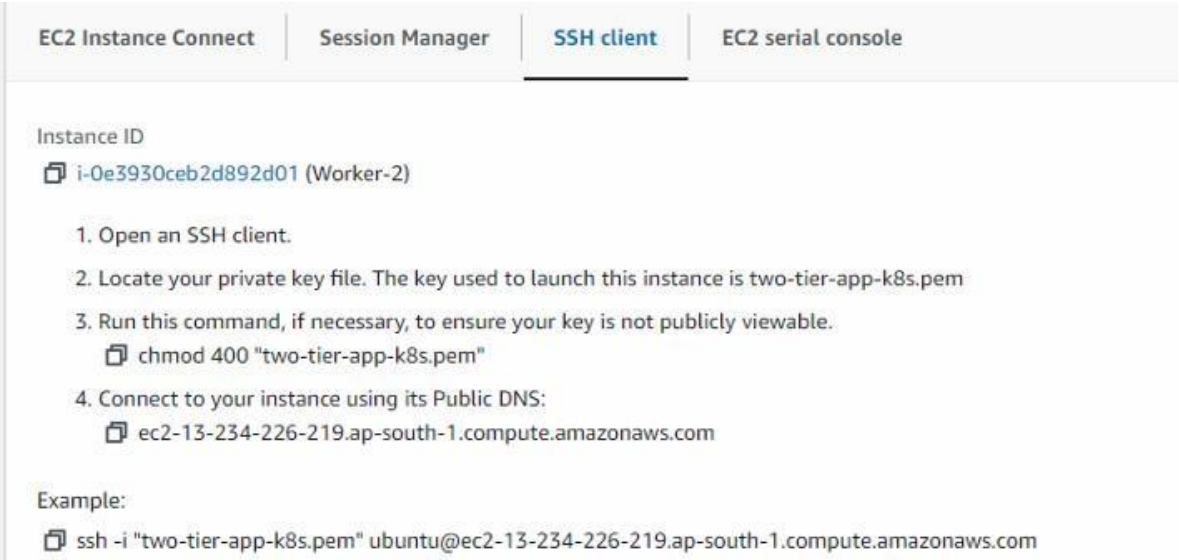




Create 3 EC2 Ubuntu Instances of Ubuntu version 20.04 and keep all the instances in the same security group on AWS. (Name 1 as Master, the other 2 as worker-1 and worker-2)



Now the ssh created, copy the text given in the example



```
acer@TMP214-53 MINGW64 ~/Downloads
$ ssh -i "two-tier-app-k8s.pem" ubuntu@ec2-13-232-36-34.ap-south-1.compute.amazo
naws.com
The authenticity of host 'ec2-13-232-36-34.ap-south-1.compute.amazonaws.com (13.
232.36.34)' can't be established.
ED25519 key fingerprint is SHA256:uVGEO+FWYefj60j0ft70Sralv8NrzEi/IwxAtBY+EPE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-232-36-34.ap-south-1.compute.amazonaws.com' (
ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-1022-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Sep 11 14:07:10 UTC 2024

  System load:  0.0               Processes:             106
  Usage of /:   20.7% of 7.57GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for eth0: 172.31.45.227
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
```

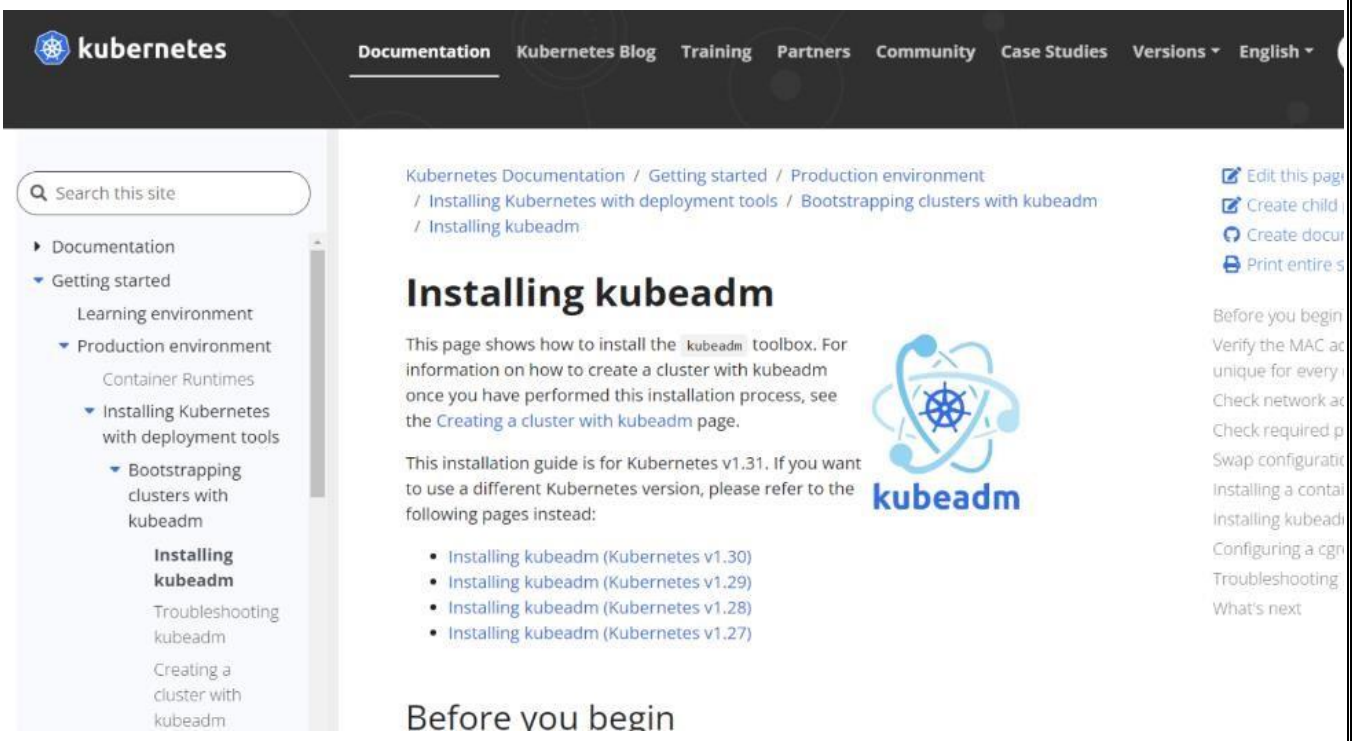
```
ubuntu@ip-172-31-81-188:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-81-188:~$
```

```
Services   Search for services, features, blogs, docs, and more
ubuntu@ip-172-31-23-53:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-23-53:~$
```

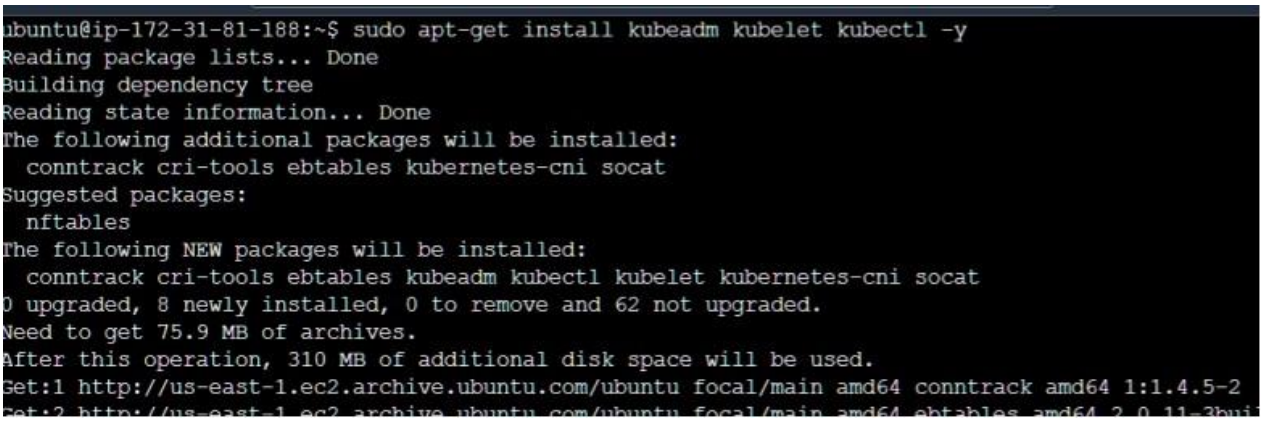
```
ubuntu@ip-172-31-21-143:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-21-143:~$
```

Kubernetes Installation

Go to official documentation off kubedam



1. $sudo apt-get install kubeadm kubelet kubectl -y



```
ubuntu@ip-172-31-81-188:~$ sudo apt-get install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 62 not upgraded.
Need to get 75.9 MB of archives.
After this operation, 310 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 conntrack amd64 1:1.4.5-2
```

2. Verify the installation with

```
ubuntu@ip-172-31-23-53:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.0", GitCommit:"a866cbe2e5bbaa01cfd5e969aa3e033f3282a8a2", GitTreeState:"clean"
, BuildDate:"2022-08-23T17:43:25Z", GoVersion:"go1.19", Compiler:"gc", Platform:"linux/amd64"}
ubuntu@ip-172-31-23-53:~$
```

3. $sudo swapoff –a

```
ubuntu@ip-172-31-22-29:~$ sudo swapoff -a
sudo sed -i '/ swap / s/^/#/' /etc/fstab
```

4. $sudo hostnamectl set-hostname

```
ubuntu@ip-172-31-23-53:~$ sudo hostnamectl set-hostname worker1
ubuntu@ip-172-31-23-53:~$
```

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1 EOF
sudo sysctl—system

```
ubuntu@ip-172-31-81-188:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.25.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (967 MB) is less than the minimum 1700 MB
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [kubernetes kubernetes.default kubernetes.default.svc kub
```

```
lternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

ou should now deploy a pod network to the cluster.
un "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

hen you can join any number of worker nodes by running the following on each as root:

ubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu \
        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bfcadac2d827699b0db3d87e13192873a1044f86e2
buntu@ip-172-31-81-188:~$
```

Deploy Pod Network to Cluster A Pod Network is a way to allow communication between different nodes in the cluster. This tutorial uses the flannel virtual network.

```
ubuntu@ip-172-31-81-188:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@ip-172-31-81-188:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/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@ip-172-31-81-188:~$
```

Join Worker Nodes to the Cluster On the worker nodes, run the command provided by the master node during initialization . It looks something like this: sudo kubeadm join :6443--token --discovery-token-ca-cert-hash sha256:

```
root@worker1:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu         --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bf
adac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
preflight] Running pre-flight checks
error execution phase preflight: couldn't validate the identity of the API Server: Get "https://172.31.81.188:6443/api/v1/namespaces/kube-public/confi
maps/cluster-info?timeout=10s": net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)
To see the stack trace of this error execute with --v=5 or higher
root@worker1:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu         --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bf
adac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
preflight] Running pre-flight checks
preflight] Reading configuration from the cluster...
preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
kubelet-start] Starting the kubelet
kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Verify the Cluster

```
Using cluster from kubectl context: workshop.k8s.local

Validating cluster workshop.k8s.local

INSTANCE GROUPS
NAME                    ROLE    MACHINETYPE     MIN     MAX     SUBNETS
master-us-west-2a       Master  t3.medium       1       1       us-west-2a
nodes-us-west-2a        Node    t3.medium       1       1       us-west-2a

NODE STATUS
NAME                                            ROLE    READY
ip-172-20-40-55.us-west-2.compute.internal      master  True
ip-172-20-58-174.us-west-2.compute.internal     node    True

Your cluster workshop.k8s.local is ready
```