Adv. Devops Experiment no. 4

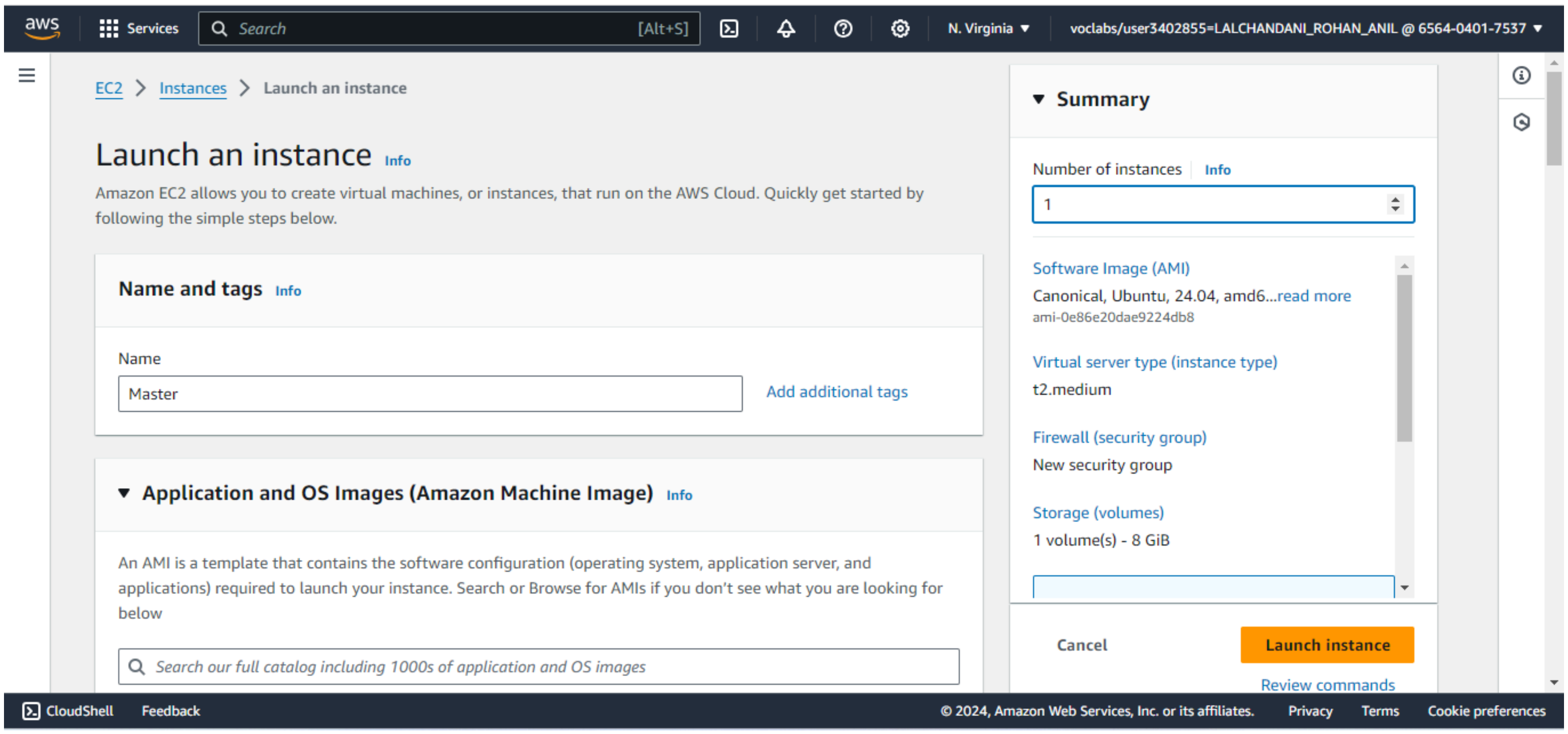
Name: Rohan Lalchandani
Class: D15A   Roll no.: 25

Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Theory:

Implementation:

1. Create an EC2 Ubuntu Instance - Master.



aws
Services
Search [Alt+S]
N. Virginia
voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537
EC2 > Instances > Launch an instance
Summary
Cancel
Launch instance
Review commands
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: Master
Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Number of instances: 1

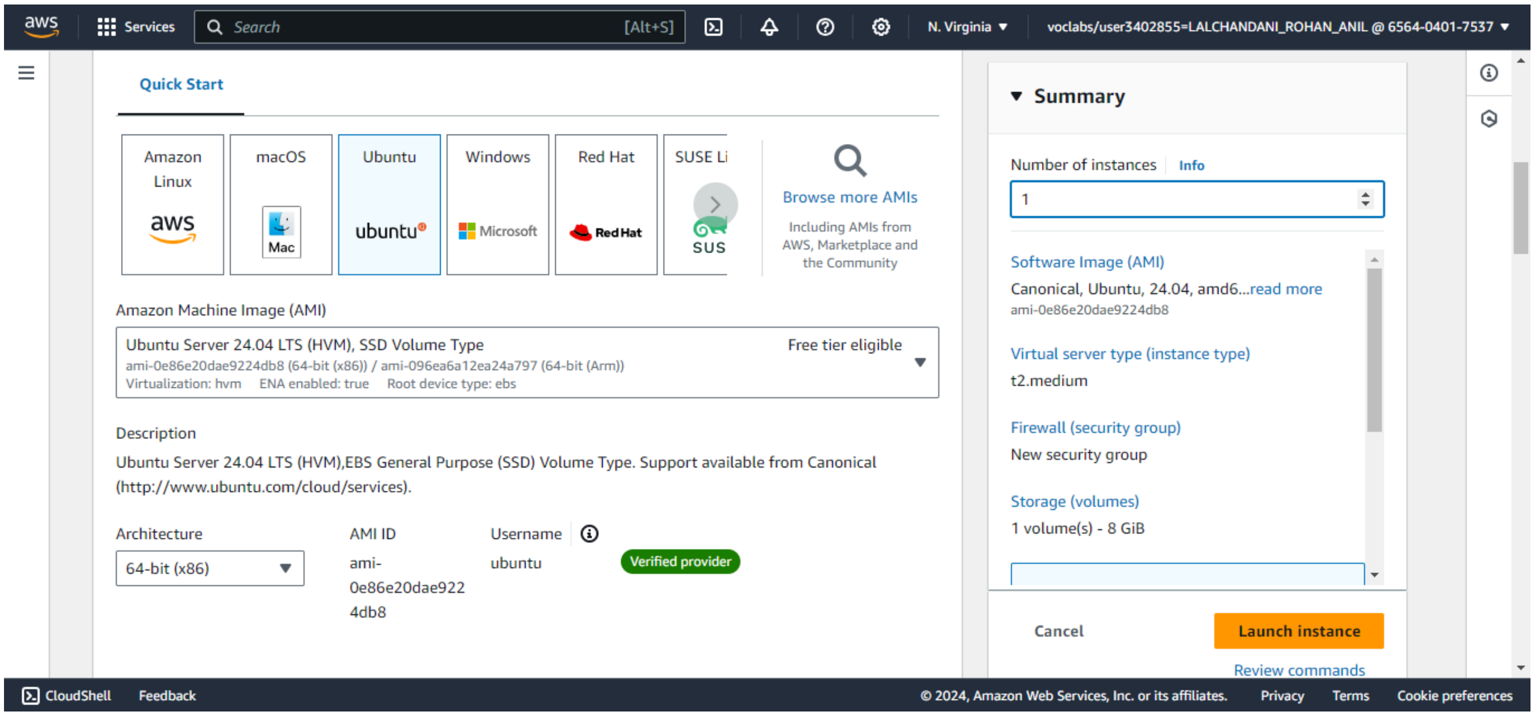
Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type)
t2.medium

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

2. Select the following AMI image.



Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0e86e20dae9224db8 (64-bit (x86)) / ami-096ea6a12ea24a797 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Architecture: 64-bit (x86)
AMI ID: ami-0e86e20dae9224db8
Username: ubuntu
Verified provider

Number of instances: 1

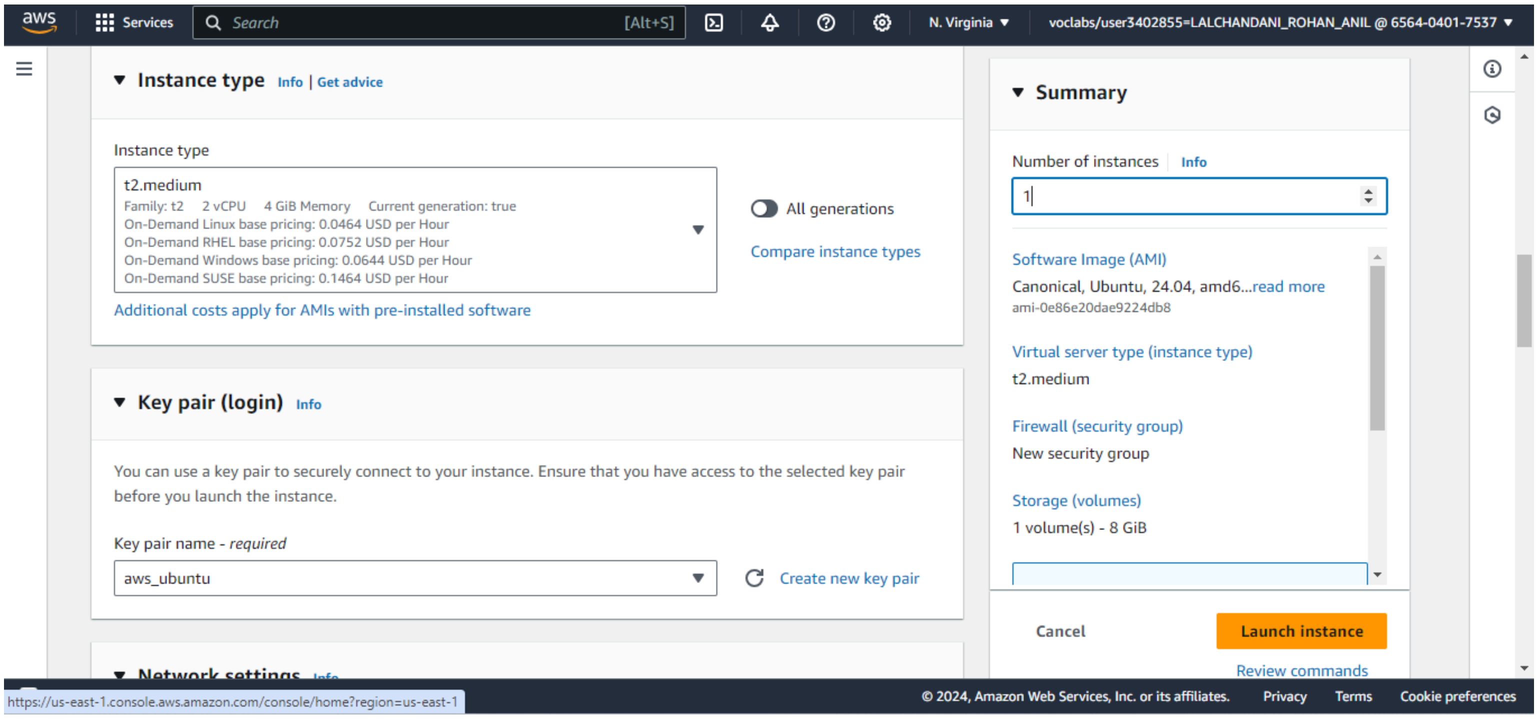
Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type)
t2.medium

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

3. Select t2.medium in instance type.
Why?
t2.medium has 2 vCPUs and 4 GB of RAM, which allows it to run more pods and handle larger or more resource-intensive containers compared to **t2.micro** (1 vCPU, 1 GB of RAM).



Instance type

Instance type: t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0464 USD per Hour
On-Demand RHEL base pricing: 0.0752 USD per Hour
On-Demand Windows base pricing: 0.0644 USD per Hour
On-Demand SUSE base pricing: 0.1464 USD per Hour

All generations

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*: aws_ubuntu

Summary

Number of instances: 1

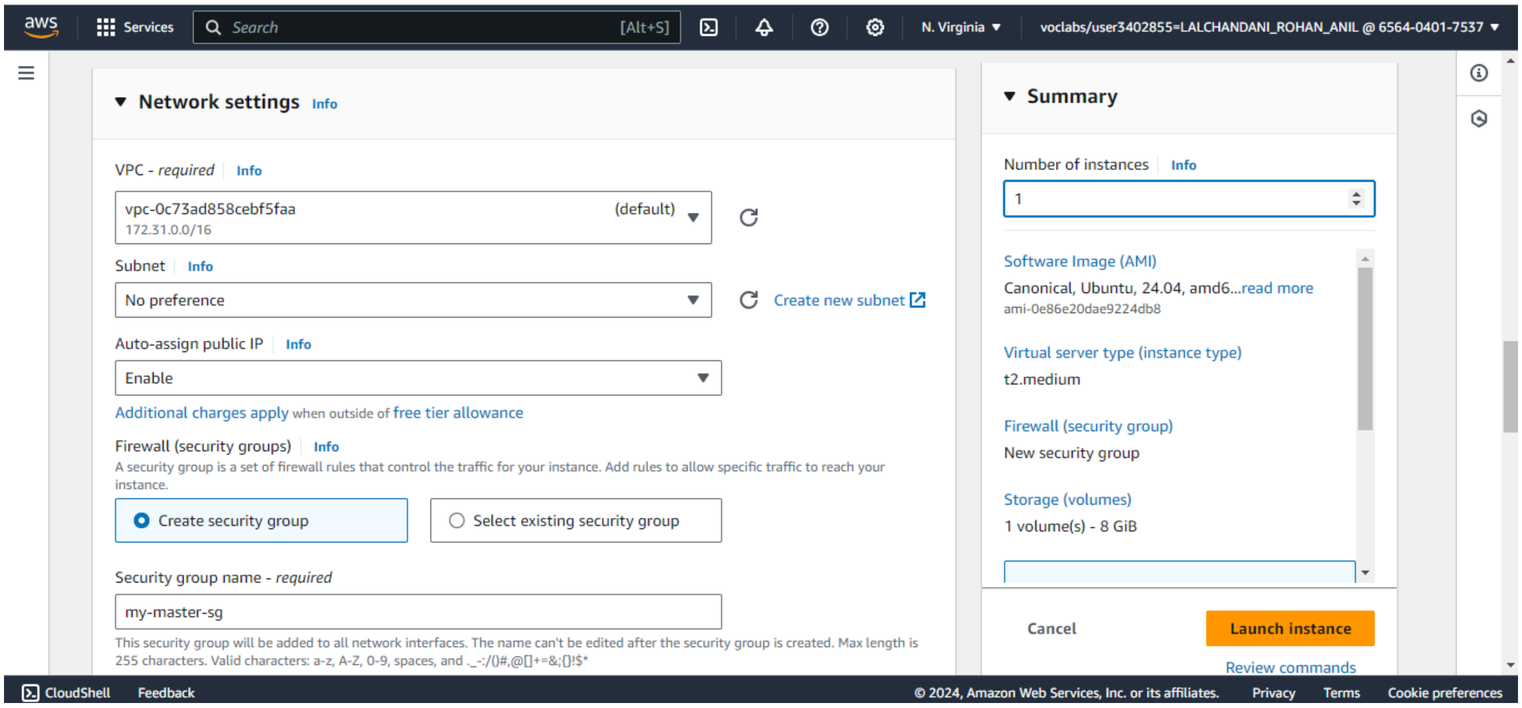
Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type): t2.medium

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

4. Create security group
In it type the group name and description as my-master-sg



Network settings

VPC - *required*: vpc-0c73ad858cebf5faa (default)
172.31.0.0/16

Subnet: No preference

Auto-assign public IP: Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[✓ Create security group] Select existing security group

Security group name - *required*: my-master-sg

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Summary

Number of instances: 1

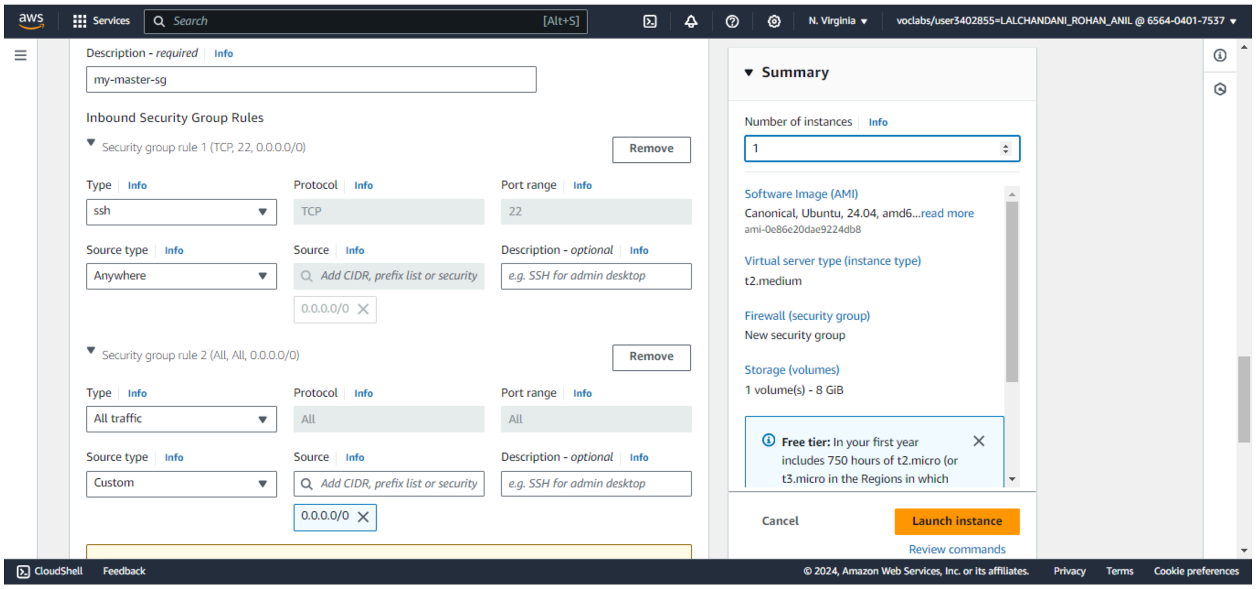
Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type): t2.medium

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

5. Edit the inbound rules and add a new rule to accept “All traffic” as shown below.



Description - *required* Info
my-master-sg

Inbound Security Group Rules

Security group rule 1 (TCP, 22, 0.0.0.0/0) — Remove

Type	Protocol	Port range
ssh	TCP	22

Source type	Source	Description - *optional*
Anywhere	Add CIDR, prefix list or security — 0.0.0.0/0	e.g. SSH for admin desktop

Security group rule 2 (All, All, 0.0.0.0/0) — Remove

Type	Protocol	Port range
All traffic	All	All

Source type	Source	Description - *optional*
Custom	Add CIDR, prefix list or security — 0.0.0.0/0	e.g. SSH for admin desktop

Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

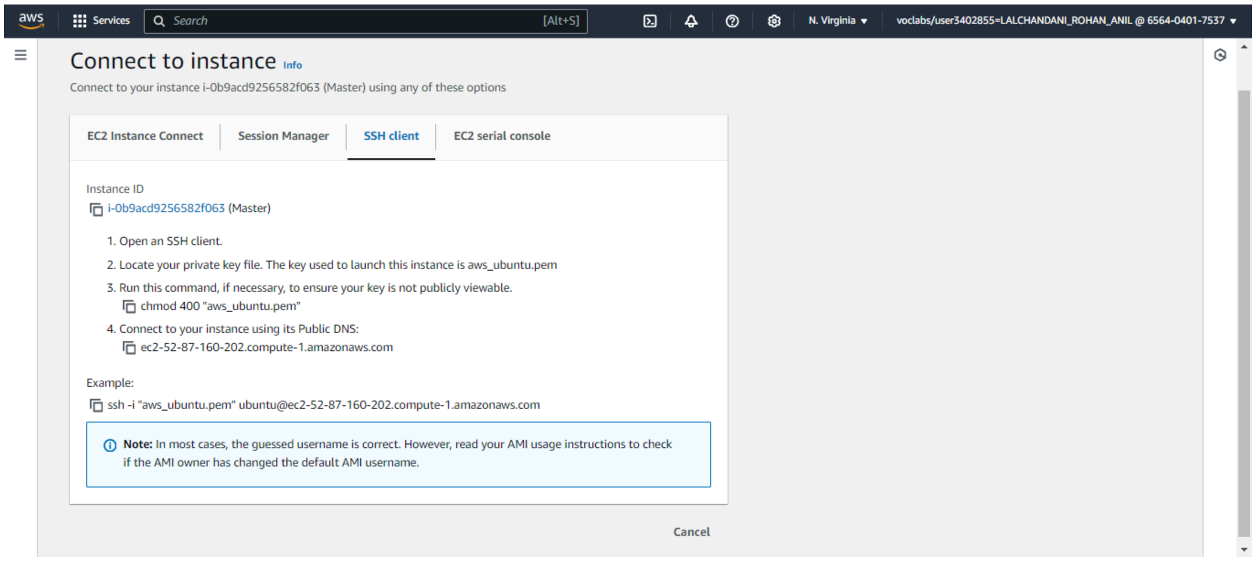
Virtual server type (instance type): t2.medium

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which

6. Now connect to the instance by copying the “ssh” command and executing in Git Bash.



Connect to instance Info

Connect to your instance i-0b9acd9256582f063 (Master) using any of these options

EC2 Instance Connect | Session Manager | **SSH client** | EC2 serial console

Instance ID
i-0b9acd9256582f063 (Master)

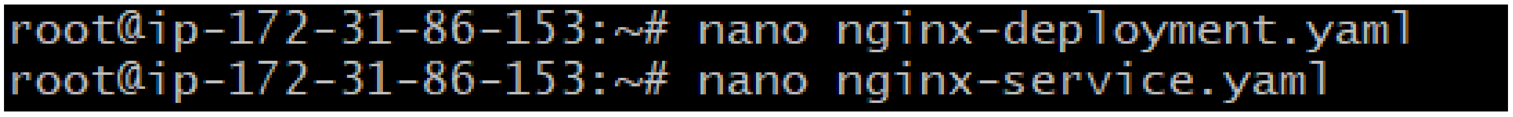
1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is aws_ubuntu.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
```
chmod 400 "aws_ubuntu.pem"
```
4. Connect to your instance using its Public DNS:
```
ec2-52-87-160-202.compute-1.amazonaws.com
```

Example:
```
ssh -i "aws_ubuntu.pem" ubuntu@ec2-52-87-160-202.compute-1.amazonaws.com
```

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

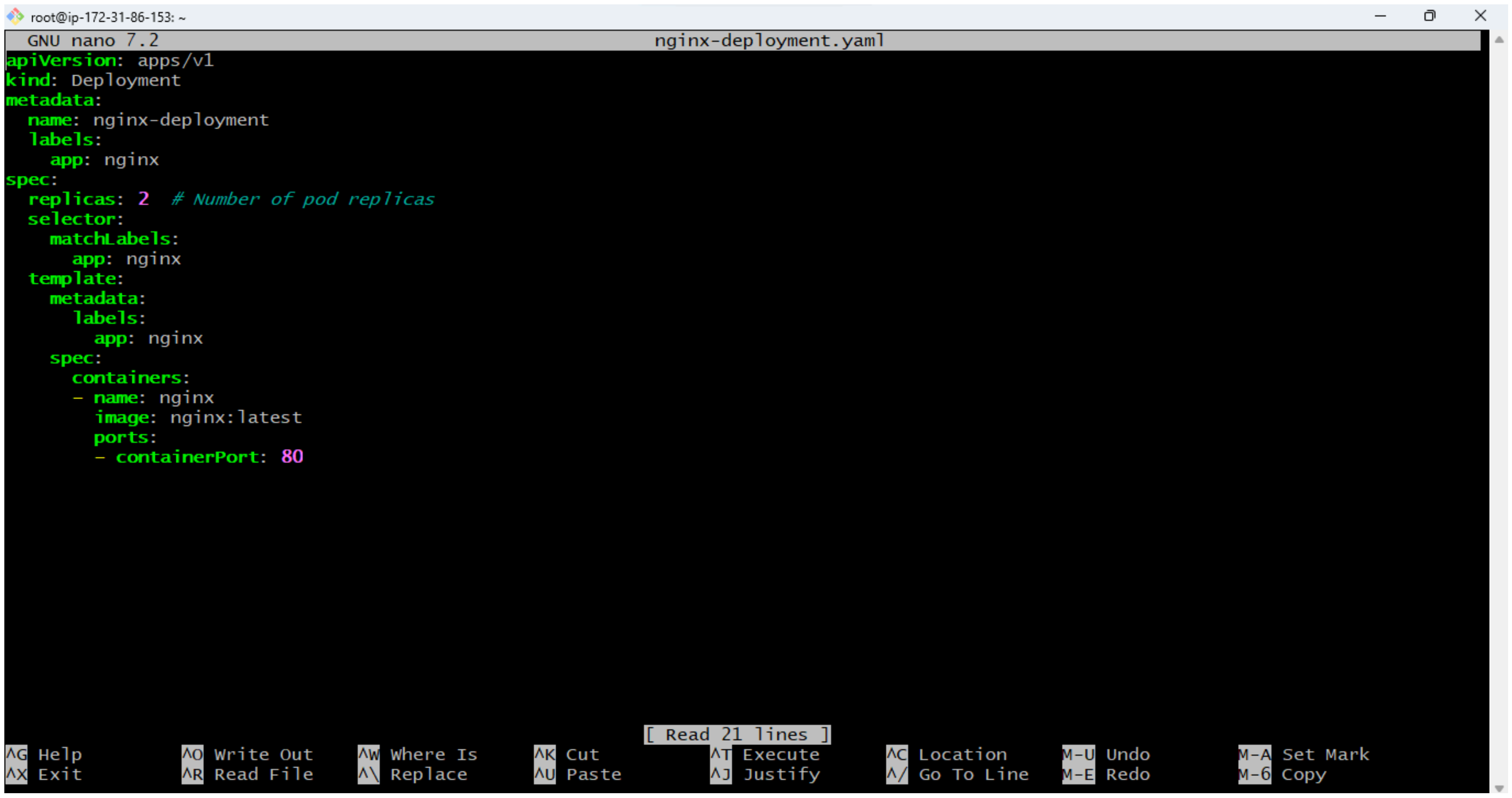
7. Install docker, kubernetes by steps did in 3rd experiment.

8. After that we have to create 2 files as shown below.



```
root@ip-172-31-86-153:~# nano nginx-deployment.yaml
root@ip-172-31-86-153:~# nano nginx-service.yaml
```

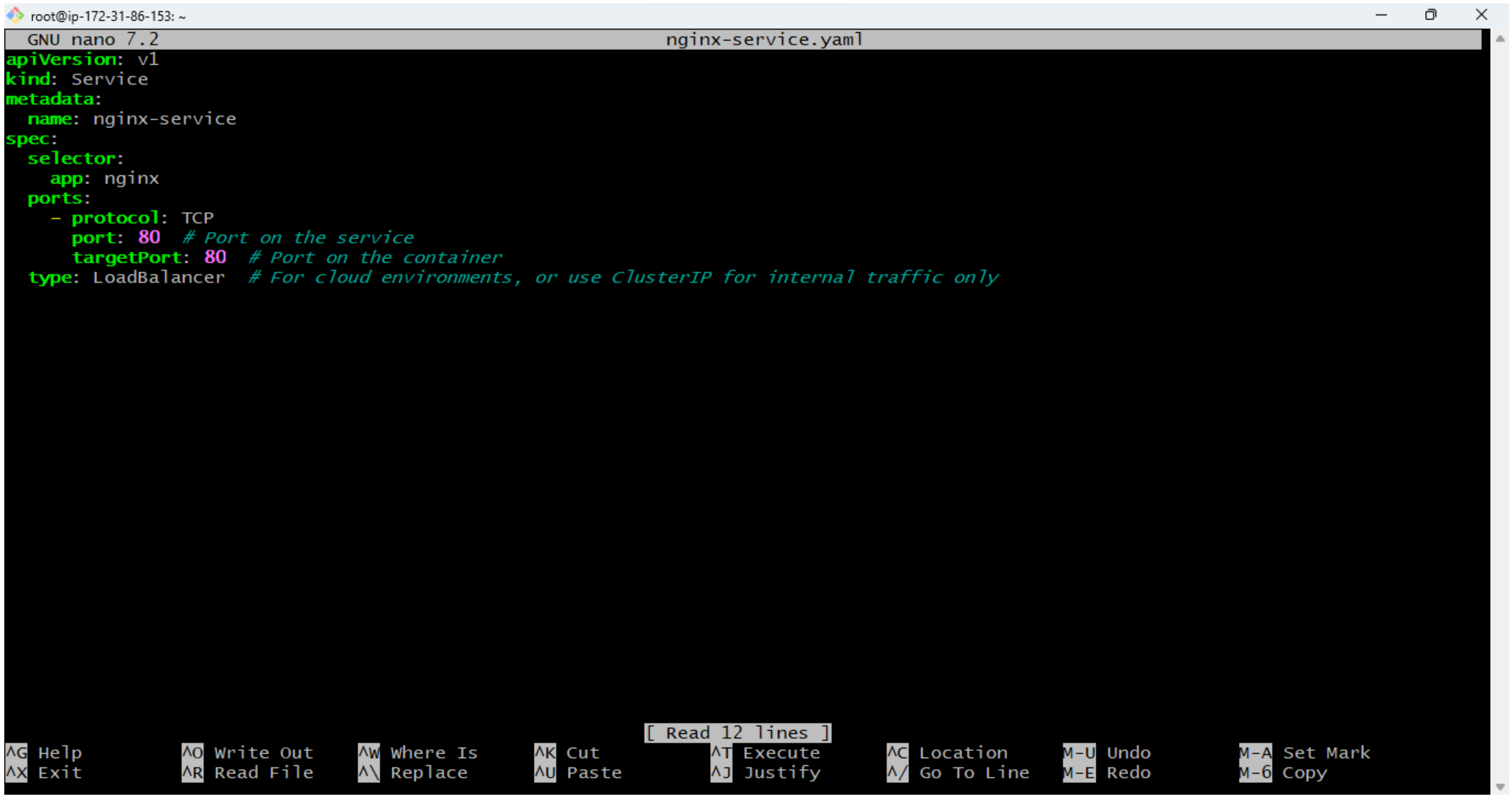
nginx-deployment.yaml file :



```
  GNU nano 7.2                         nginx-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2  # Number of pod replicas
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

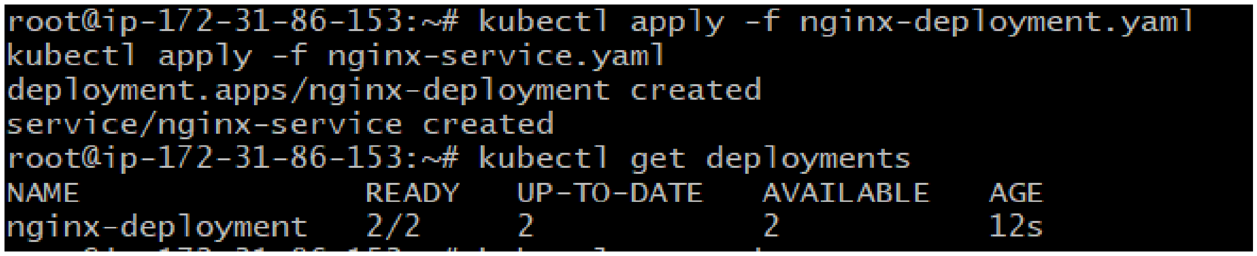
```
                                    [ Read 21 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

nginx-service.yaml file :



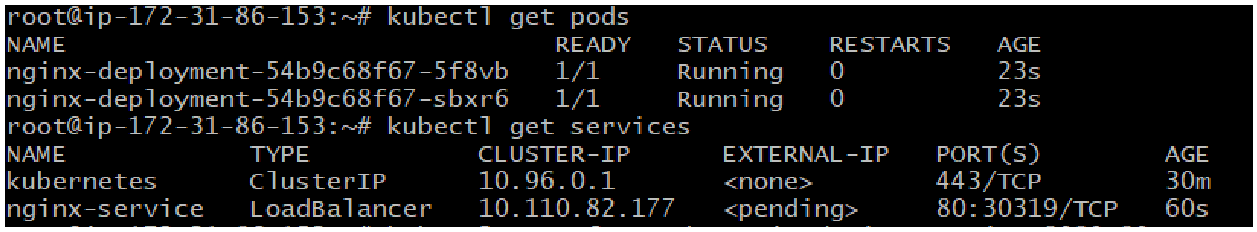
```
  GNU nano 7.2                          nginx-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80  # Port on the service
      targetPort: 80  # Port on the container
  type: LoadBalancer  # For cloud environments, or use ClusterIP for internal traffic only

                                [ Read 12 lines ]
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy
```

Apply the files



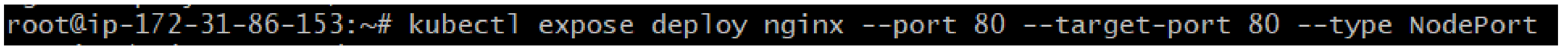
```
root@ip-172-31-86-153:~# kubectl apply -f nginx-deployment.yaml
kubectl apply -f nginx-service.yaml
deployment.apps/nginx-deployment created
service/nginx-service created
root@ip-172-31-86-153:~# kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           12s
```

Check the status of the services and pods if they are running properly



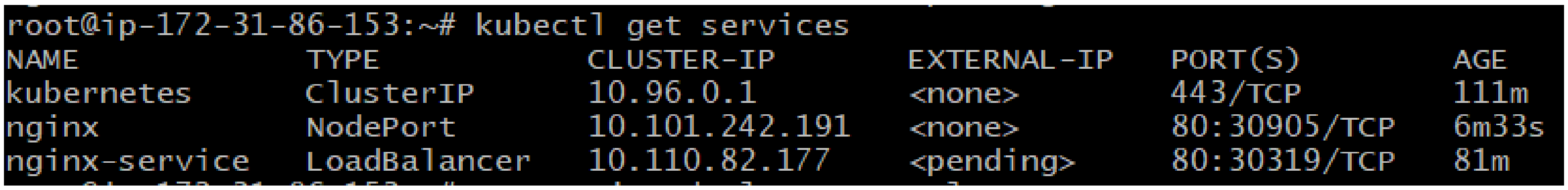
```
root@ip-172-31-86-153:~# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-54b9c68f67-5f8vb   1/1     Running   0          23s
nginx-deployment-54b9c68f67-sbxr6   1/1     Running   0          23s
root@ip-172-31-86-153:~# kubectl get services
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1       <none>        443/TCP        30m
nginx-service   LoadBalancer   10.110.82.177   <pending>     80:30319/TCP   60s
```

Expose the port by this command



```
root@ip-172-31-86-153:~# kubectl expose deploy nginx --port 80 --target-port 80 --type NodePort
```

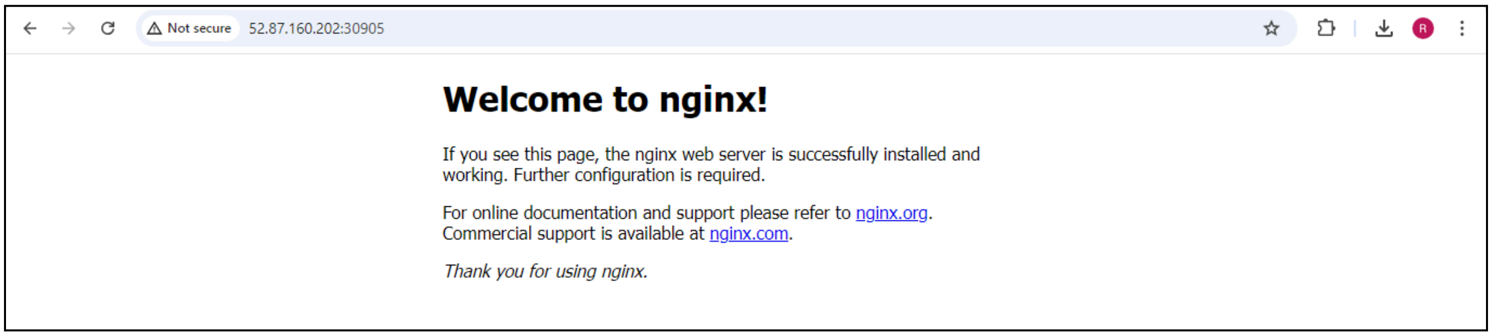
Check the status of the port



```
root@ip-172-31-86-153:~# kubectl get services
NAME             TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes       ClusterIP      10.96.0.1        <none>        443/TCP        111m
nginx            NodePort       10.101.242.191   <none>        80:30905/TCP   6m33s
nginx-service    LoadBalancer   10.110.82.177    <pending>     80:30319/TCP   81m
```

Now go to the browser and copy the “public dns” of the master instance created, paste it in the browser put a colon and then type the port number of the nginx service, in this case it is 30905

Example: http://52.87.160.202:30905/



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.