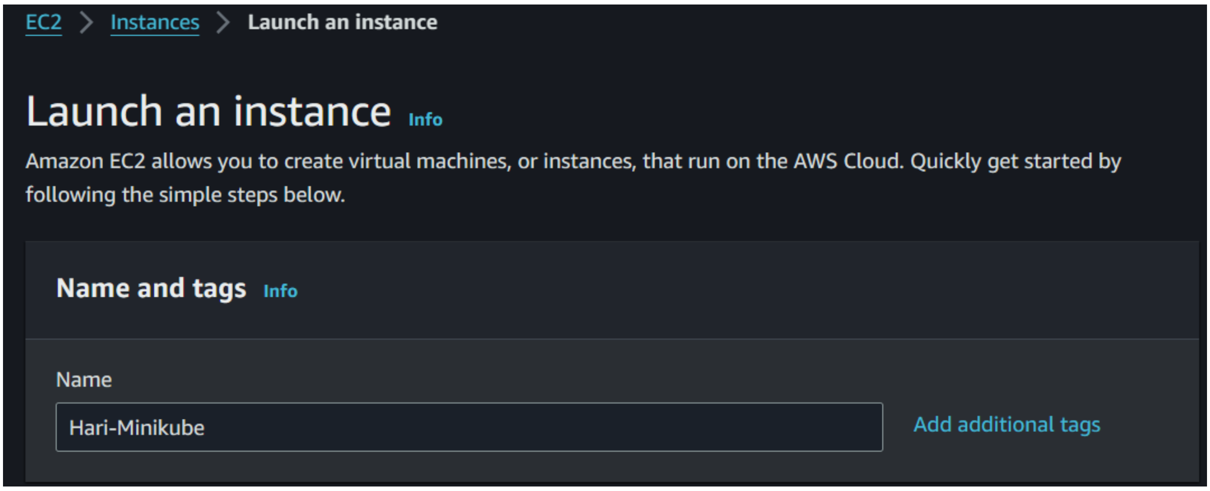
INDEX

Tasks **Page No**

You have been asked to:

- Use the previous deployment
- Deploy an nginx deployment of 3 replicas
- Create an nginx service of type clusterip
- Create an ingress service /apache to apache service /nginx to nginx service

Task1 - Launched instance for minikube installation.

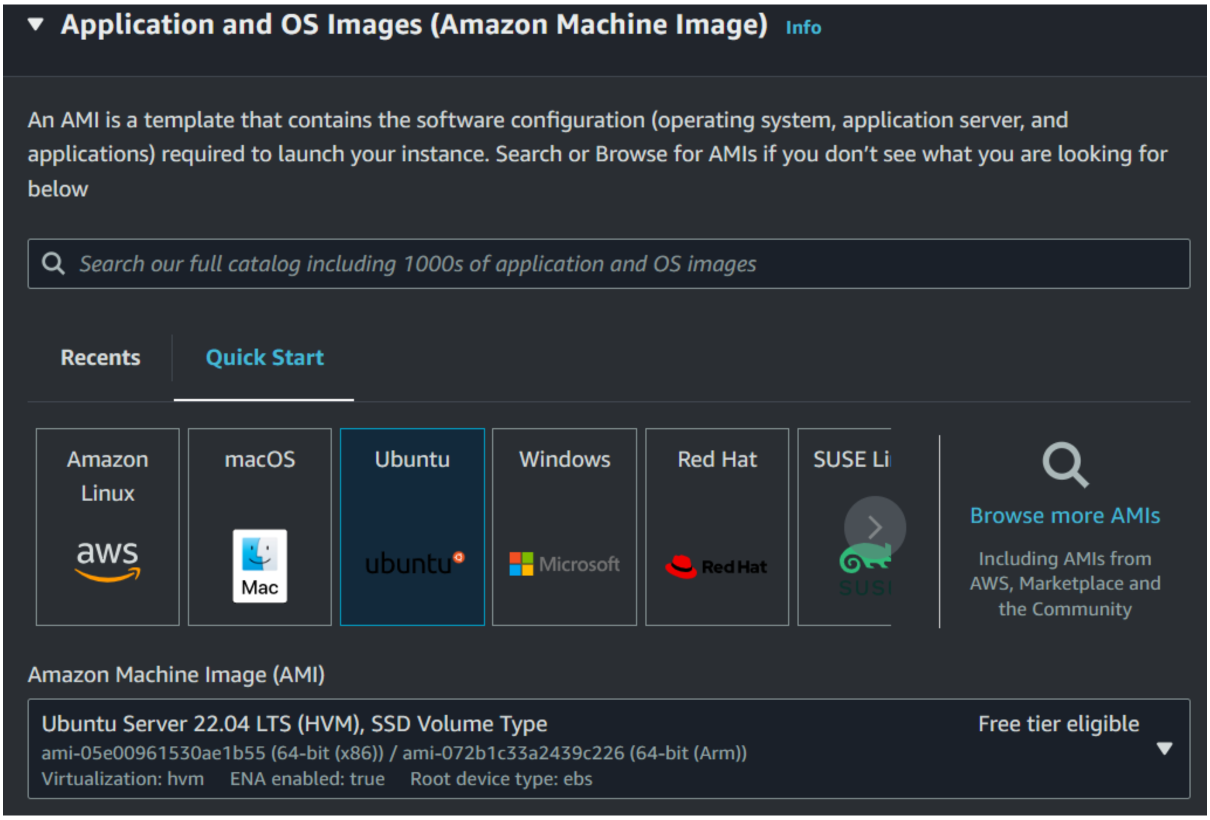


Launch an instance

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

Name: Hari-Minikube



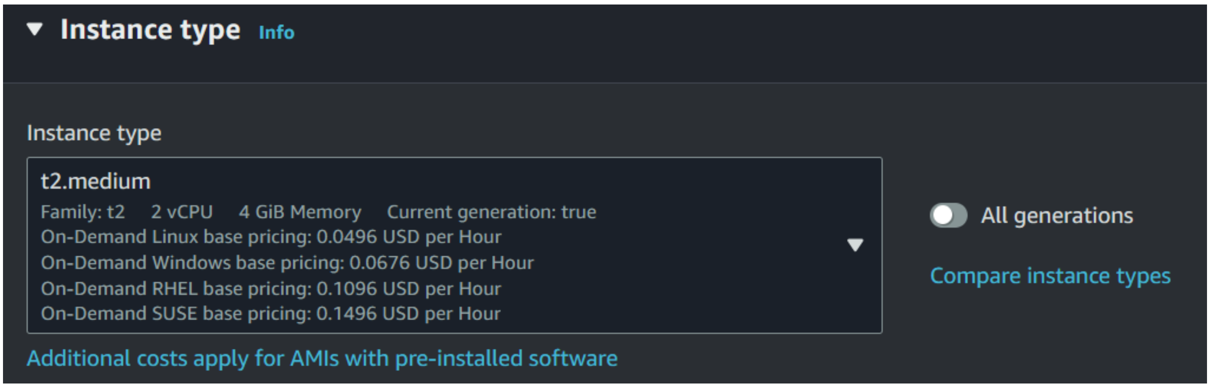
▼ **Application and OS Images (Amazon Machine Image)** Info

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

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-05e00961530ae1b55 (64-bit (x86)) / ami-072b1c33a2439c226 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

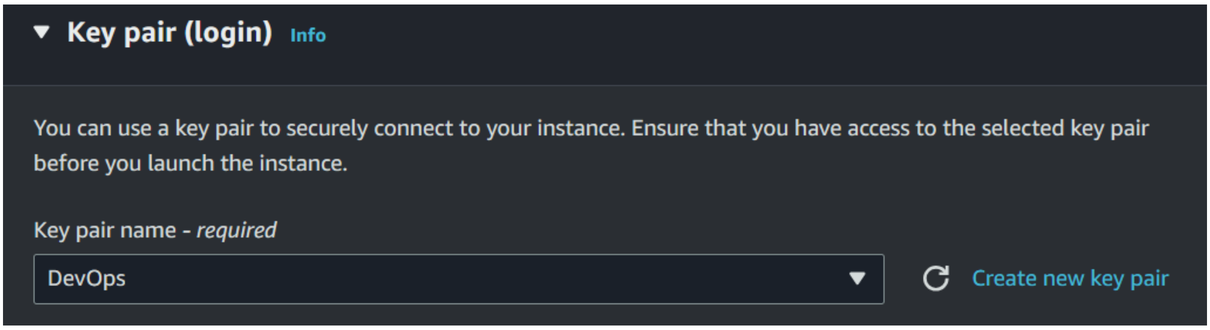


▼ **Instance type** Info

Instance type

t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0496 USD per Hour
On-Demand Windows base pricing: 0.0676 USD per Hour
On-Demand RHEL base pricing: 0.1096 USD per Hour
On-Demand SUSE base pricing: 0.1496 USD per Hour

Additional costs apply for AMIs with pre-installed software

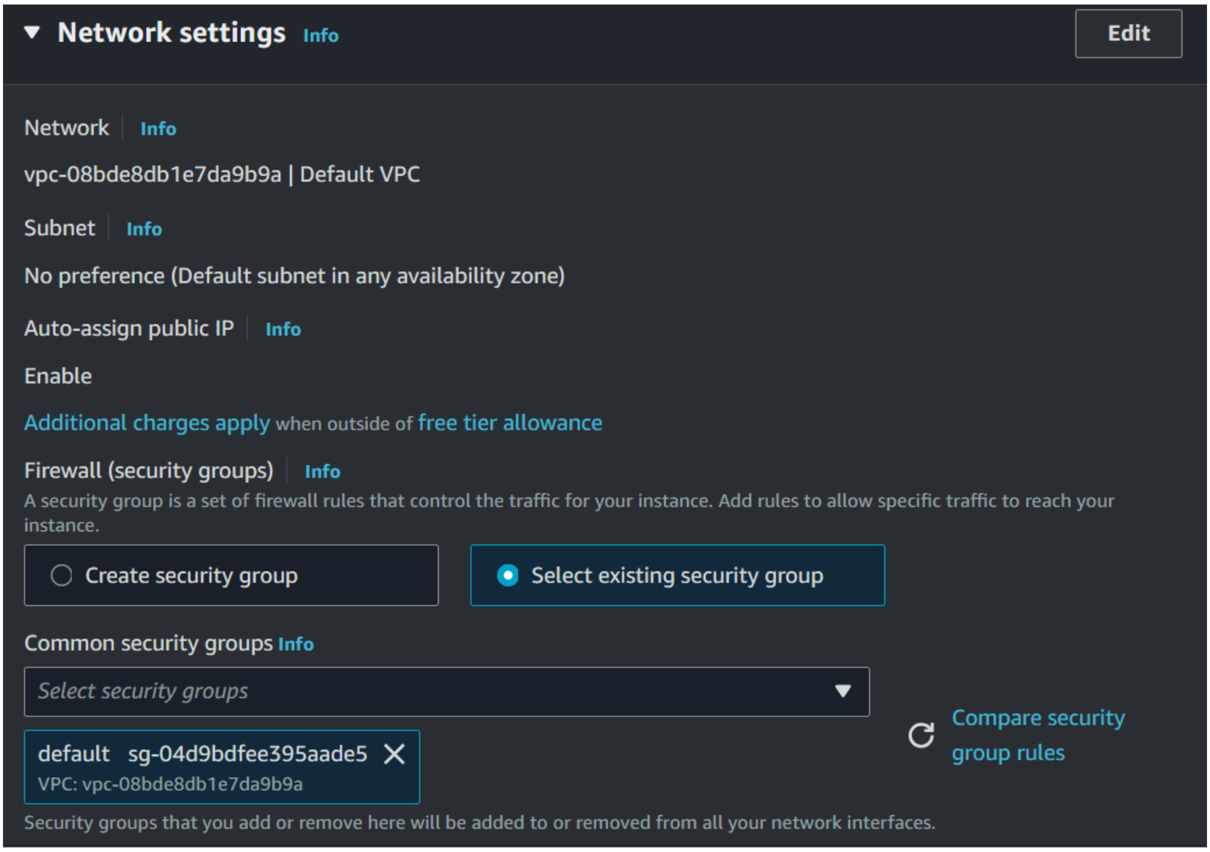


▼ **Key pair (login)** Info

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*

DevOps



Network Info

vpc-08bde8db1e7da9b9a | Default VPC

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

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

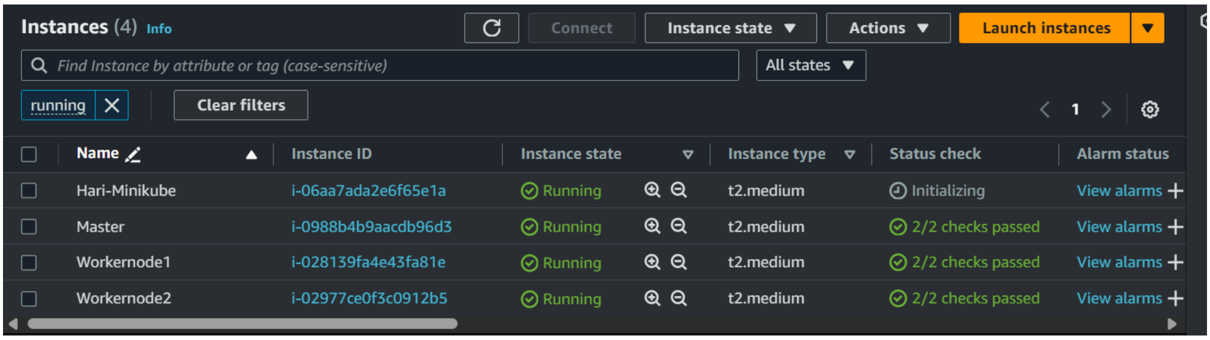
Common security groups Info

Select security groups ▼

default sg-04d9bdfee395aade5 ✕
VPC: vpc-08bde8db1e7da9b9a

Security groups that you add or remove here will be added to or removed from all your network interfaces.

We could see in below snip that the instance has been launched.

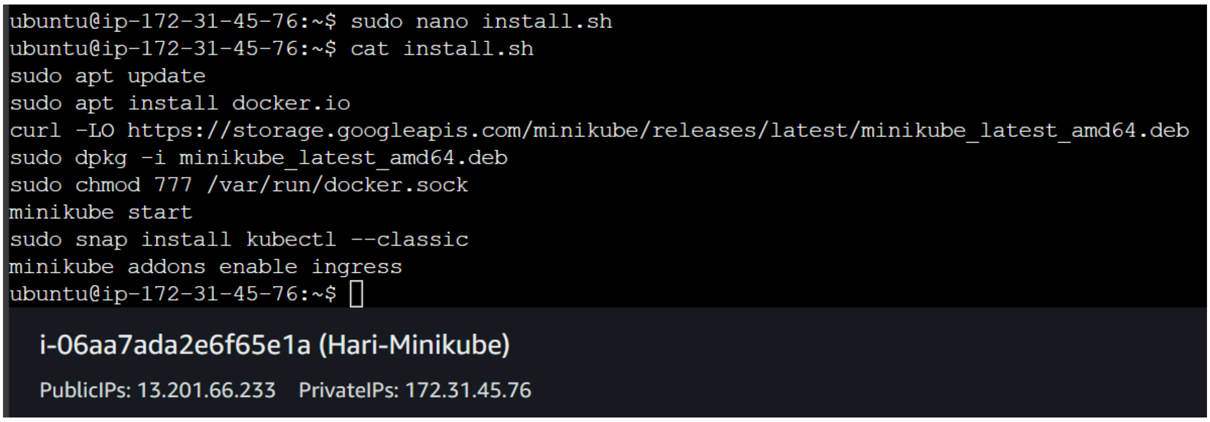


□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
□	Hari-Minikube	i-06aa7ada2e6f65e1a	✓ Running	t2.medium	Initializing	View alarms +
□	Master	i-0988b4b9aacdb96d3	✓ Running	t2.medium	✓ 2/2 checks passed	View alarms +
□	Workernode1	i-028139fa4e43fa81e	✓ Running	t2.medium	✓ 2/2 checks passed	View alarms +
□	Workernode2	i-02977ce0f3c0912b5	✓ Running	t2.medium	✓ 2/2 checks passed	View alarms +

Task2 – Install minikube.

```
sudo apt update

sudo apt install docker.io

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb

sudo dpkg -i minikube_latest_amd64.deb

sudo chmod 777 /var/run/docker.sock

minikube start

sudo snap install kubectl --classic
```

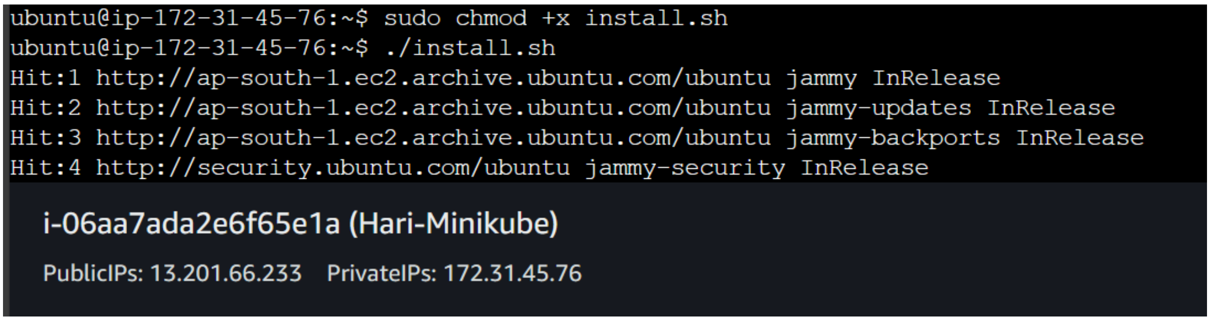
minikube addons enable ingress



```
ubuntu@ip-172-31-45-76:~$ sudo nano install.sh
ubuntu@ip-172-31-45-76:~$ cat install.sh
sudo apt update
sudo apt install docker.io
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb
sudo dpkg -i minikube_latest_amd64.deb
sudo chmod 777 /var/run/docker.sock
minikube start
sudo snap install kubectl --classic
minikube addons enable ingress
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

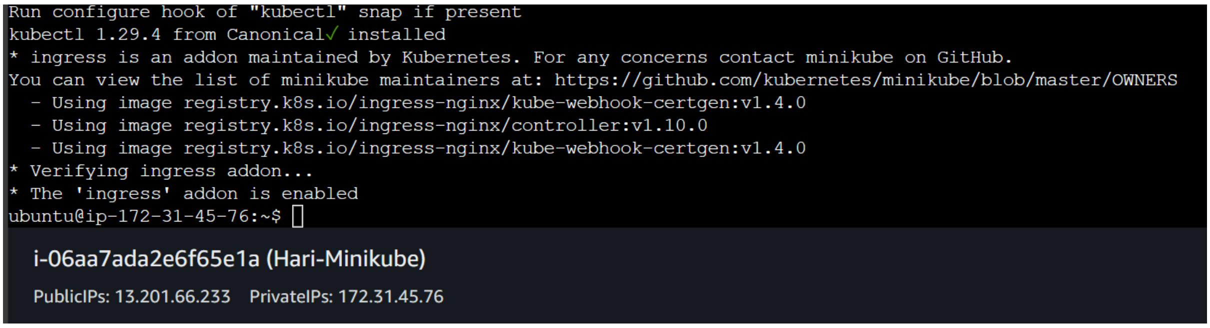
PublicIPs: 13.201.66.233   PrivateIPs: 172.31.45.76



```
ubuntu@ip-172-31-45-76:~$ sudo chmod +x install.sh
ubuntu@ip-172-31-45-76:~$ ./install.sh
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233   PrivateIPs: 172.31.45.76

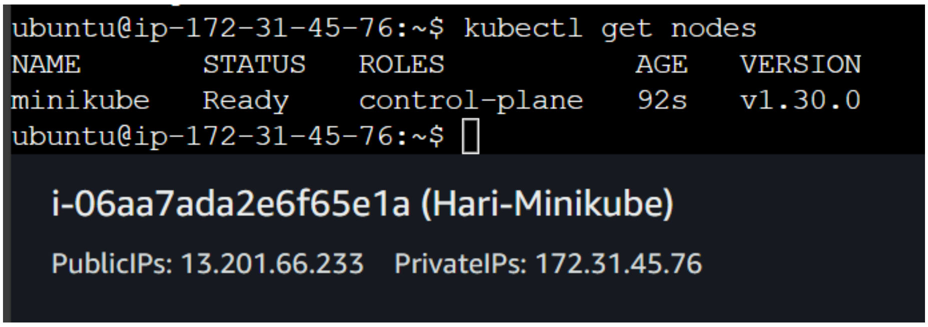


```
Run configure hook of "kubectl" snap if present
kubectl 1.29.4 from Canonical✓ installed
* ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.0
  - Using image registry.k8s.io/ingress-nginx/controller:v1.10.0
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.0
* Verifying ingress addon...
* The 'ingress' addon is enabled
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233   PrivateIPs: 172.31.45.76

Task3 – Create deployment.

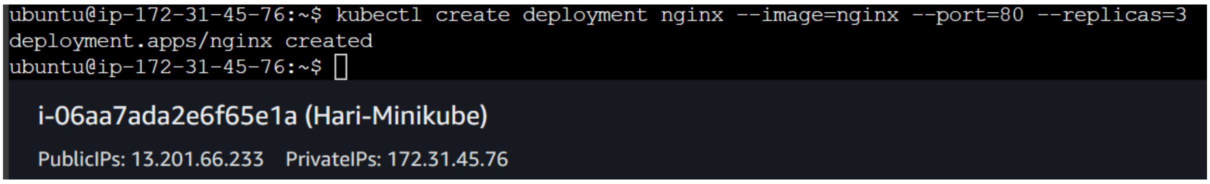


```
ubuntu@ip-172-31-45-76:~$ kubectl get nodes
NAME       STATUS   ROLES           AGE   VERSION
minikube   Ready    control-plane   92s   v1.30.0
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233   PrivateIPs: 172.31.45.76

kubectl create deployment nginx --image=nginx --port=80 --replicas=3

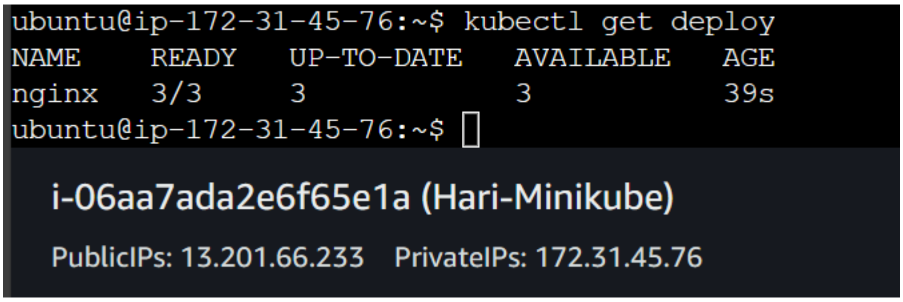


```
ubuntu@ip-172-31-45-76:~$ kubectl create deployment nginx --image=nginx --port=80 --replicas=3
deployment.apps/nginx created
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233   PrivateIPs: 172.31.45.76

kubectl get deploy

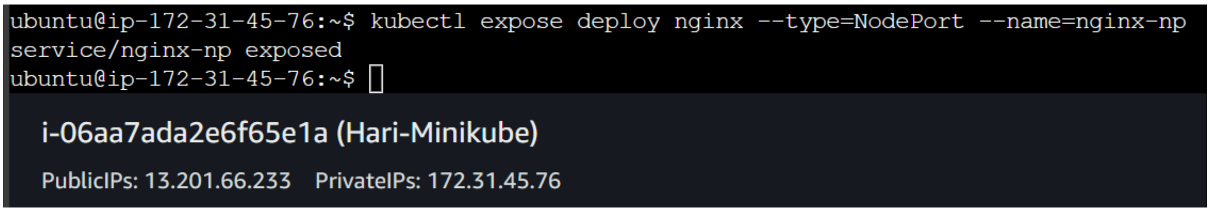


```
ubuntu@ip-172-31-45-76:~$ kubectl get deploy
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nginx   3/3     3            3           39s
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

kubectl expose deploy nginx --type=NodePort --name=nginx-np

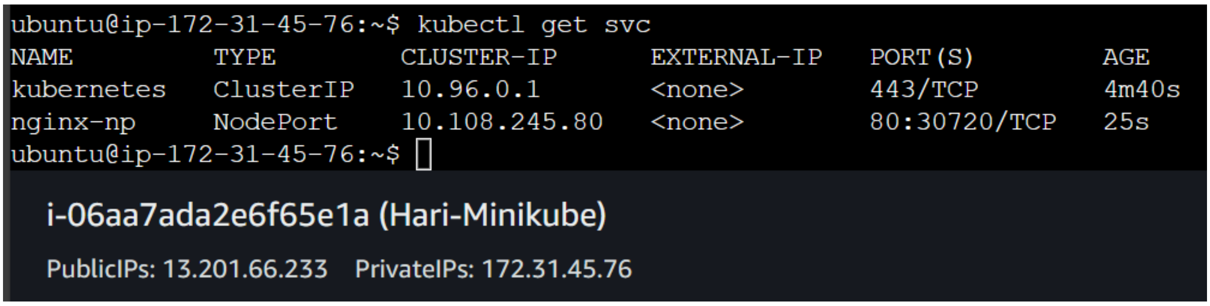


```
ubuntu@ip-172-31-45-76:~$ kubectl expose deploy nginx --type=NodePort --name=nginx-np
service/nginx-np exposed
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

Kubectl get svc



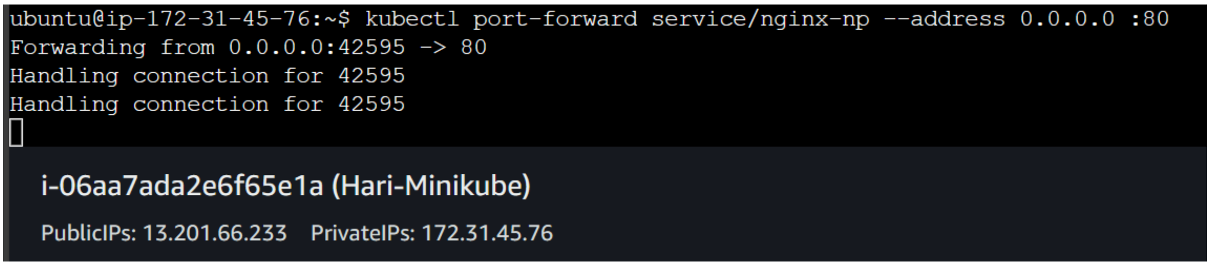
```
ubuntu@ip-172-31-45-76:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        4m40s
nginx-np     NodePort    10.108.245.80   <none>        80:30720/TCP   25s
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

Task4 – Forwarding Traffic.

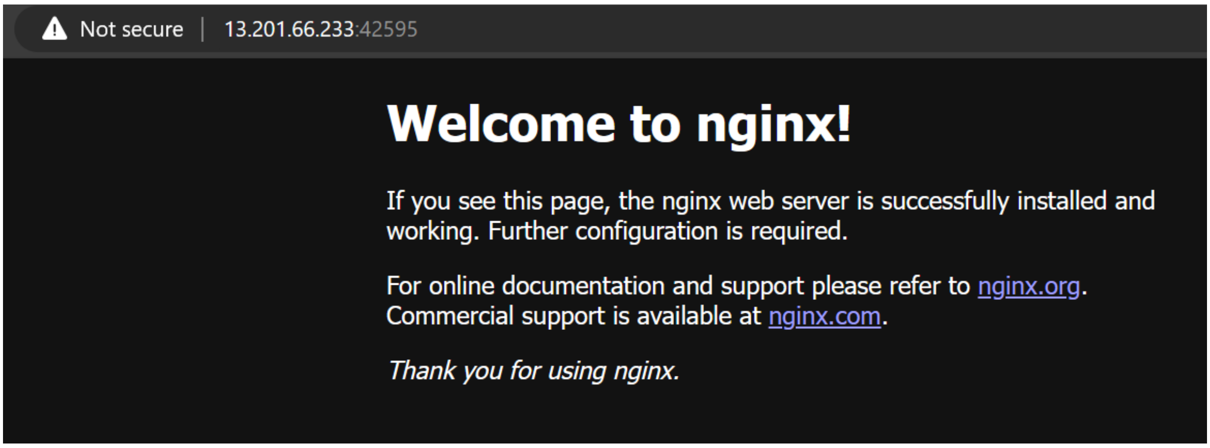
kubectl port-forward service/nginx-np --address 0.0.0.0 :80



```
ubuntu@ip-172-31-45-76:~$ kubectl port-forward service/nginx-np --address 0.0.0.0 :80
Forwarding from 0.0.0.0:42595 -> 80
Handling connection for 42595
Handling connection for 42595
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76



Welcome to nginx!

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

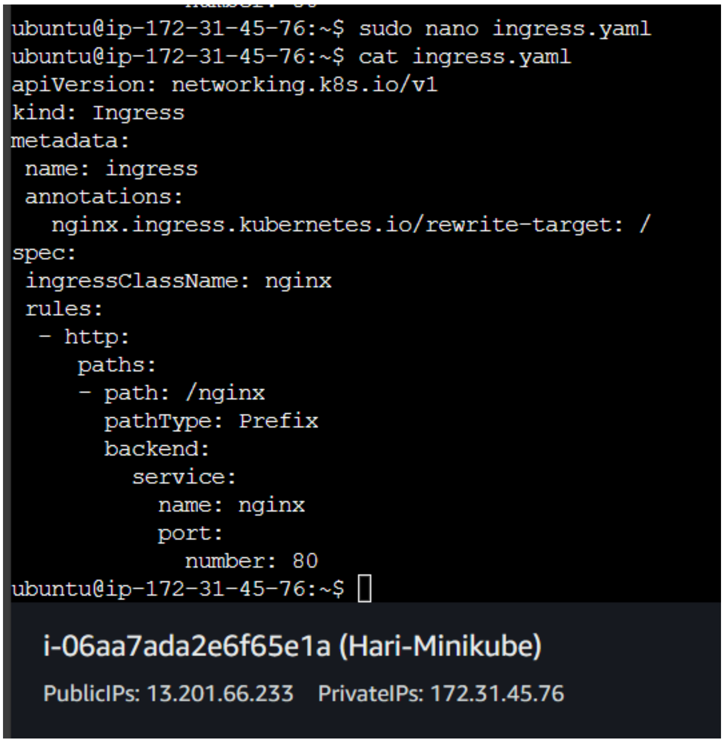
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Task5 – Creation of ingress service.

```
sudo nano ingress.yaml
```

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: ingress
 annotations:
   nginx.ingress.kubernetes.io/rewrite-target: /
spec:
 ingressClassName: nginx
 rules:
  - http:
     paths:
     - path: /nginx
       pathType: Prefix
       backend:
         service:
           name: nginx
           port:
             number: 80
```

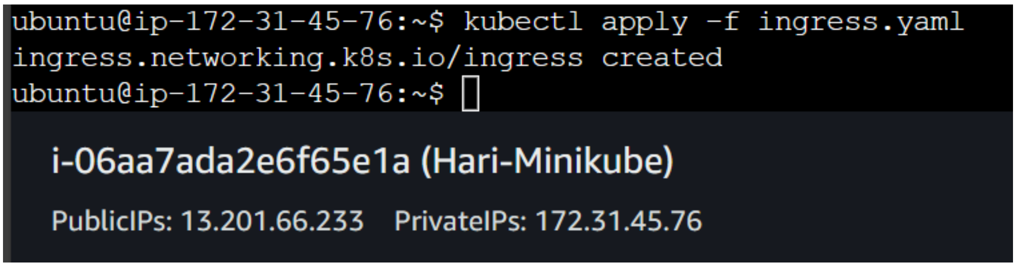


```
ubuntu@ip-172-31-45-76:~$ sudo nano ingress.yaml
ubuntu@ip-172-31-45-76:~$ cat ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: ingress
 annotations:
   nginx.ingress.kubernetes.io/rewrite-target: /
spec:
 ingressClassName: nginx
 rules:
  - http:
     paths:
     - path: /nginx
       pathType: Prefix
       backend:
         service:
           name: nginx
           port:
             number: 80
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

kubectl apply -f ingress.yaml

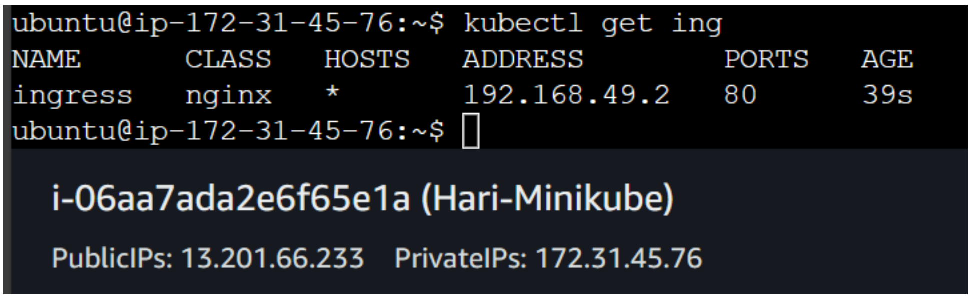


```
ubuntu@ip-172-31-45-76:~$ kubectl apply -f ingress.yaml
ingress.networking.k8s.io/ingress created
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

kubectl get ing



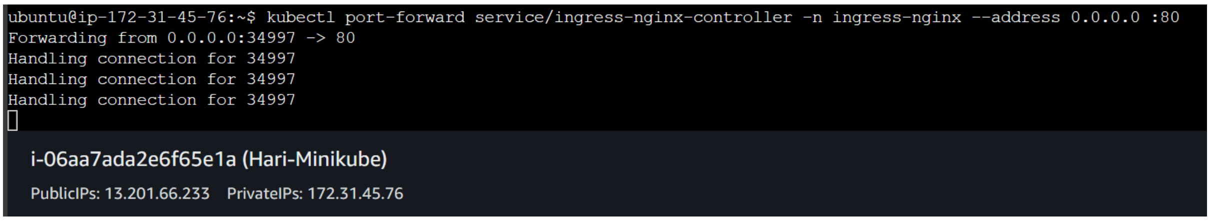
```
ubuntu@ip-172-31-45-76:~$ kubectl get ing
NAME      CLASS   HOSTS   ADDRESS        PORTS   AGE
ingress   nginx   *       192.168.49.2   80      39s
ubuntu@ip-172-31-45-76:~$ 
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76

Task6 – Establishing connection.

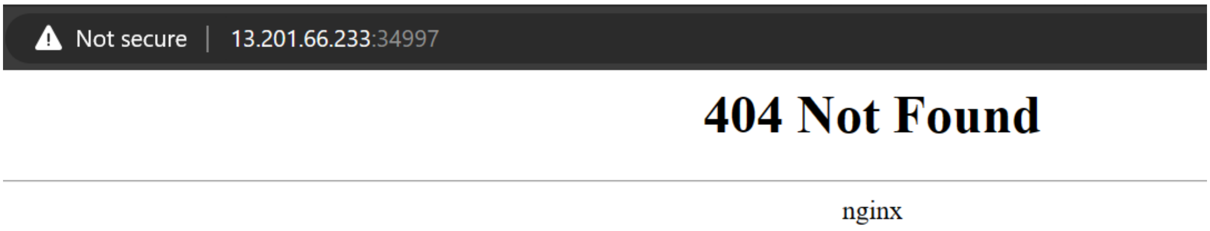
kubectl port-forward service/ingress-nginx-controller -n ingress-nginx --address 0.0.0.0 :80



```
ubuntu@ip-172-31-45-76:~$ kubectl port-forward service/ingress-nginx-controller -n ingress-nginx --address 0.0.0.0 :80
Forwarding from 0.0.0.0:34997 -> 80
Handling connection for 34997
Handling connection for 34997
Handling connection for 34997
```

i-06aa7ada2e6f65e1a (Hari-Minikube)

PublicIPs: 13.201.66.233 PrivateIPs: 172.31.45.76



Not secure | 13.201.66.233:34997

404 Not Found

nginx

********** THE END **********