

## Task 19: Setup minikube at your local and explore creating namespaces (Go through official documentation)

### MINIKUBE

#### 1. Launch an instance:

## 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

[Add additional tags](#)

### ▼ Instance type [Info](#) | [Get advice](#)

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: 0.0784 USD per Hour

On-Demand SUSE base pricing: 0.1496 USD per Hour

▼

☒ All generations

[Compare instance types](#)

[Additional costs apply for AMIs with pre-installed software](#)

#### 1. Install docker:

```
ubuntu@ip-172-31-40-193:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

```
API version:      1.43
Go version:       go1.22.2
Git commit:       24.0.7-0ubuntu4
Built:           Wed Apr 17 20:08:25 2024
OS/Arch:         linux/amd64
Context:         default
```

Server:

Engine:

```
Version:         24.0.7
API version:     1.43 (minimum version 1.12)
Go version:      go1.22.2
Git commit:      24.0.7-0ubuntu4
Built:           Wed Apr 17 20:08:25 2024
OS/Arch:         linux/amd64
Experimental:    false
```

containerd:

```
Version:         1.7.12
GitCommit:
```

runc:

```
Version:         1.1.12-0ubuntu3
GitCommit:
```

docker-init:

```
Version:         0.19.0
GitCommit:
```

```
ubuntu@ip-172-31-40-193:~$ sudo usermod -aG docker ubuntu
```

## 2. Install kubectl

```
ubuntu@ip-172-31-40-193:~$ curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 57.4M  100 57.4M    0     0  6263k      0  0:00:09  0:00:09 --:--:-- 8059k
ubuntu@ip-172-31-40-193:~$ chmod +x ./kubectl
ubuntu@ip-172-31-40-193:~$ sudo mv ./kubectl /usr/local/bin
ubuntu@ip-172-31-40-193:~$ kubectl version --short -client
Client Version: v1.19.6-eks-49a6c0
```

## 3. Install eksctl:

```
ubuntu@ip-172-31-40-193:~$ curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
ubuntu@ip-172-31-40-193:~$ sudo mv /tmp/eksctl /usr/local/bin
ubuntu@ip-172-31-40-193:~$ eksctl version
0.180.0
```

## 4. Install minikube:

```
ubuntu@ip-172-31-40-193:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 91.1M  100 91.1M    0     0  11.3M      0  0:00:08  0:00:08 --:--:-- 15.9M
ubuntu@ip-172-31-40-193:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
ubuntu@ip-172-31-40-193:~$
```

## 5. Start minikube:

```
ubuntu@ip-172-31-40-193:~$ sudo usermod -aG docker ubuntu
newgrp docker
ubuntu@ip-172-31-40-193:~$ minikube start
* minikube v1.33.1 on Ubuntu 24.04 (xen/amd64)
* Automatically selected the docker driver. Other choices: ssh, none
* Using Docker driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.44 ...
* Downloading Kubernetes v1.30.0 preload ...
  > preloaded-images-k8s-v18-v1...: 342.90 MiB / 342.90 MiB 100.00% 14.69 M
  > gcr.io/k8s-minikube/kicbase...: 481.58 MiB / 481.58 MiB 100.00% 14.58 M
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass

! /usr/local/bin/kubectl is version 1.19.6-eks-49a6c0, which may have incompatibilities with Kubernetes 1.30.0.
  - Want kubectl v1.30.0? Try 'minikube kubectl -- get pods -A'
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
ubuntu@ip-172-31-40-193:~$
```

## 6. Minikube status:

```
ubuntu@ip-172-31-40-193:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

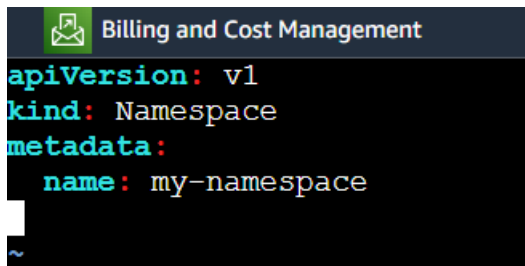
## NAMESPACE

## 7. Namespace:

### 1. Create custom namespace:

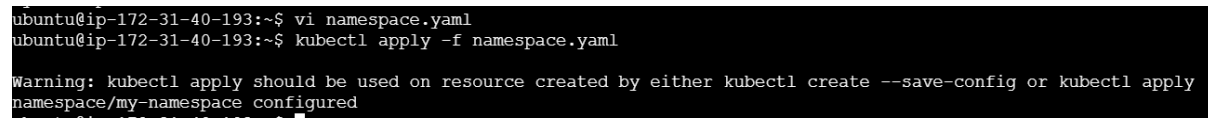
```
ubuntu@ip-172-31-40-193:~$ kubectl create namespace my-namespace
namespace/my-namespace created
ubuntu@ip-172-31-40-193:~$ kubectl get namespaces
NAME                STATUS    AGE
default             Active   2m43s
kube-node-lease     Active   2m43s
kube-public         Active   2m43s
kube-system         Active   2m43s
my-namespace        Active   12s
```

## 2. Namespace.yaml file: use kubectl / namespace.yaml for creating namespace



A terminal window titled "Billing and Cost Management" displays the contents of a file named namespace.yaml. The file contains the following YAML configuration:

```
apiVersion: v1
kind: Namespace
metadata:
  name: my-namespace
```



A terminal window shows the execution of the kubectl apply command. The user runs 'vi namespace.yaml' and then 'kubectl apply -f namespace.yaml'. The output shows a warning message and the successful configuration of the namespace.

```
ubuntu@ip-172-31-40-193:~$ vi namespace.yaml
ubuntu@ip-172-31-40-193:~$ kubectl apply -f namespace.yaml

Warning: kubectl apply should be used on resource created by either kubectl create --save-config or kubectl apply
namespace/my-namespace configured
```