

TASK 3-Minikube Deployment Task

Step 1: Start Minikube

Start the Minikube cluster using the following command:

```
minikube start
```

```
vboxuser@Ubuntu:~$ minikube start
🐹 minikube v1.35.0 on Ubuntu 24.04 (vbox/amd64)
🌟 Automatically selected the docker driver. Other choices: none, ssh
🔗 Using Docker driver with root privileges
👍 Starting "minikube" primary control-plane node in "minikube" cluster
📦 Pulling base image v0.0.46 ...
📦 Downloading Kubernetes v1.32.0 preload ...
> preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 3.10 Mi
> gcr.io/k8s-minikube/kicbase...: 498.81 MiB / 500.31 MiB 99.70% 3.89 MiB
🔥 Creating docker container (CPUs=2, Memory=2200MB) ...
🐳 Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
  ▪ 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
🏡 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
vboxuser@Ubuntu:~$
```

This initializes the Minikube cluster using Docker as the driver.

Step 2: Install Kubectl

Since Kubectl is not found, install it with the following command:

```
sudo snap install kubectl --classic
```

Alternatively, you can download it using curl:

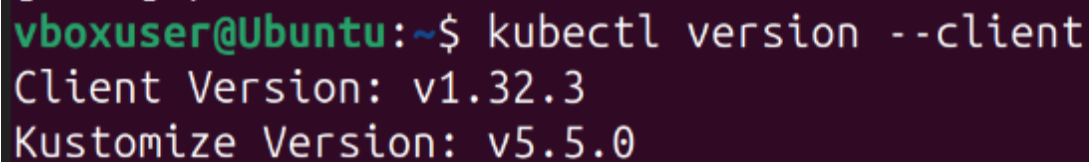
```
vboxuser@Ubuntu:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 138    100 138    0    0    235      0 --:--:-- --:--:-- --:--:-- 235
100 64    100 64    0    0    80      0 --:--:-- --:--:-- --:--:-- 80
```

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
```

Step 3: Verify Kubectl Installation

Check the client version to confirm successful installation:

```
kubectl version --client
```

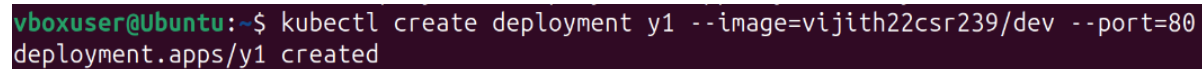
A terminal window with a dark purple background. The prompt is 'vboxuser@Ubuntu:~\$'. The command 'kubectl version --client' has been executed. The output shows 'Client Version: v1.32.3' and 'Kustomize Version: v5.5.0'.

```
vboxuser@Ubuntu:~$ kubectl version --client
Client Version: v1.32.3
Kustomize Version: v5.5.0
```

Step 4: Create a Deployment

Create a deployment named `pod1` with the image `shankar4112/devops-training`:

```
kubectl create deployment y1 --image=vijith22csr239/dev --port=80
```

A terminal window with a dark purple background. The prompt is 'vboxuser@Ubuntu:~\$'. The command 'kubectl create deployment y1 --image=vijith22csr239/dev --port=80' has been executed. The output is 'deployment.apps/y1 created'.

```
vboxuser@Ubuntu:~$ kubectl create deployment y1 --image=vijith22csr239/dev --port=80
deployment.apps/y1 created
```

Step 5: Expose the Deployment

Expose the deployment as a NodePort service:

```
kubectl expose deployment y1 --port=80 --type=NodePort
```

```
vboxuser@Ubuntu:~$ kubectl expose deployment y1 --port=80 --type=NodePort
service/y1 exposed
```

Step 6: Verify the Pod

Check the running pods:

```
kubectl get pods
```

```
vboxuser@Ubuntu:~$ kubectl get pods
```

| NAME | READY | STATUS | RESTARTS | AGE |
|---------------------|-------|---------|----------|-------|
| y-7c59f4b849-gg67g | 1/1 | Running | 0 | 3m39s |
| y1-5db6df44bc-c7cfc | 1/1 | Running | 0 | 112s |

Step 7: Access the Service

Expose the service using Minikube and get the URL:

```
minikube service y1
```

```
vboxuser@Ubuntu:~$ minikube service y1
```

| NAMESPACE | NAME | TARGET PORT | URL |
|-----------|------|-------------|---------------------------|
| default | y1 | 80 | http://192.168.49.2:30814 |

```
Opening service default/y1 in default browser...
vboxuser@Ubuntu:~$ update.go:85: cannot change mount namespace according to change mount (/run/user/1000/doc/by-app/snap.firefox /run/user/1000/doc none bind,rw,x-snapd.ignore-missing 0 0): cannot inspect "/run/user/1000/doc": lstat /run/user/1000/doc: permission denied
Gtk-Message: 06:00:47.331: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
[25255, Main Thread] WARNING: Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/25255/root: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:201
(firefox_firefox:25255): Gdk-WARNING **: 06:00:47.417: Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/25255/root
vboxuser@Ubuntu:~$
```

Step 8: Output in the Web Browser

