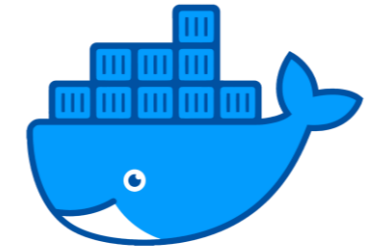
Outlines

Client-Server (ZMQ)

Networks

Minikube

kubernetes



docker

Ubuntu

Client-Server (ZMQ)

ZeroMQ

https://zeromq.org/

server: * : 5555

client: myzmqserver-service : 5555

zmqclient.py ~/Desktop/Minikube1/dock...

```python
#
#   Hello World client in Python
#   Connects REQ socket to tcp://myzmqserver-service:5555
#   Sends "Hello" to server, expects "World" back
#

import zmq


print('\n\nZMQ client version, use "myzmqserver-service" as server ip, tcp protocol, and 5555 port')
server_ip = 'myzmqserver-service' # default
#server_ip = 'localhost'
#server_ip = '0.0.0.0'


context = zmq.Context()

#  Socket to talk to server
print("Connecting to server…")
socket = context.socket(zmq.REQ)
socket.connect("tcp://"+server_ip+":5555")

#  Do 10 requests, waiting each time for a response
for request in range(10):
    print(f"Sending request {request} …")
    socket.send(b"Hello")

    #  Get the reply.
    message = socket.recv()
    print(f"Received reply {request} [ {message} ]")
```

zmqserver.py ~/Desktop/Minikube1/dock...

```python
#
#   Hello World server in Python
#   Binds REP socket to tcp://*:5555
#   Expects b"Hello" from client, replies with b"World"
#

import time
import zmq


print('The server app is running.')

context = zmq.Context()
socket = context.socket(zmq.REP)

# Server
socket.bind("tcp://*:5555")

counter = 0
while True:
    #  Wait for next request from client
    if counter == 0:
        print('Server is ready to recieve msg from client: \n')
        counter += 1
    message = socket.recv()
    print(f"Received request: {message}")

    #  Do some 'work'
    time.sleep(1)

    #  Send reply back to client
    socket.send(b"World")
```

Client-Server (ZMQ)

Dockerfile
~/Desktop/Minikube1/dock...

```
1  # Stage 1: Build stage
2  FROM python:3.11.4-alpine3.18 AS builder
3
4  # Install build dependencies
5  RUN apk add --no-cache build-base
6
7  # Install and compile pyzmq
8  RUN pip install --no-cache-dir pyzmq
9
10 # Stage 2: Final stage
11 FROM python:3.11.4-alpine3.18
12
13 ENV PATH /usr/local/bin:$PATH
14 ENV LANG C.UTF-8
15 # its important to show print outputs as logs in kubernetes
16 ENV PYTHONUNBUFFERED=1
17 ENV PYTHONIOENCODING=UTF-8
18
19 # Create a non-root user
20 RUN addgroup -S user && adduser -S -G user user
21
22 # Copy compiled pyzmq from the builder stage
23 COPY --from=builder --chown=user:user /usr/local/lib/
   python3.11/site-packages/ /usr/local/lib/python3.11/site-
   packages/
24
25 # Copy zmqclient.py
26 COPY --chown=user:user zmqclient.py /tmp/zmqclient.py
27
28 # Set the working directory
29 WORKDIR /tmp
30
31 # Zmq Sub Server
32 EXPOSE 5555
33
34 # Switch to the non-root user
35 USER user
36
37 CMD ["python3", "zmqclient.py"]
```

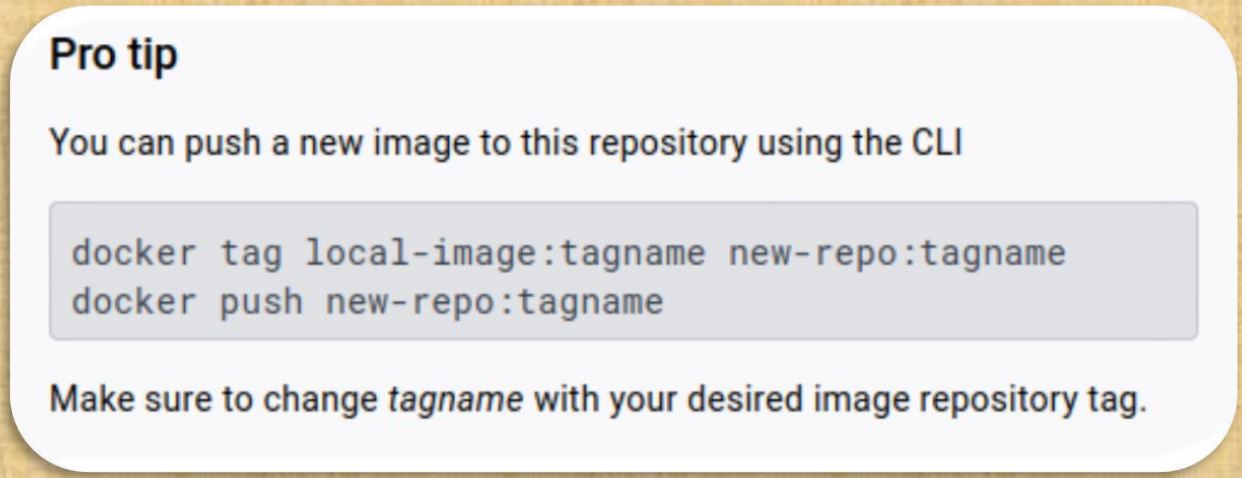
Dockerfile
~/Desktop/Minikube1/dock...

```
1  # Stage 1: Build stage
2  FROM python:3.11.4-alpine3.18 AS builder
3
4  # Install build dependencies
5  RUN apk add --no-cache build-base
6
7  # Install and compile pyzmq
8  RUN pip install --no-cache-dir pyzmq
9
10 # Stage 2: Final stage
11 FROM python:3.11.4-alpine3.18
12
13 ENV PATH /usr/local/bin:$PATH
14 ENV LANG C.UTF-8
15 # its important to show print outputs as logs in kubernetes
16 ENV PYTHONUNBUFFERED=1
17 ENV PYTHONIOENCODING=UTF-8
18
19 # Create a non-root user
20 RUN addgroup -S user && adduser -S -G user user
21
22 # Copy compiled pyzmq from the builder stage
23 COPY --from=builder --chown=user:user /usr/local/lib/
   python3.11/site-packages/ /usr/local/lib/python3.11/site-
   packages/
24
25 # Copy zmqclient.py
26 COPY --chown=user:user zmqserver.py /tmp/zmqserver.py
27
28 # Set the working directory
29 WORKDIR /tmp
30
31 # Zmq Sub Server
32 EXPOSE 5555
33
34 # Switch to the non-root user
35 USER user
36
37 CMD ["python3", "zmqserver.py"]
```

Client-Server (ZMQ)

Build, Tag, Push

server: * : 5555
client: myzmqserver-service : 5555



Build

```
sudo docker build -t myzmqserver:5.0 .
sudo docker build -t myzmqclient:5.0 .
```

Tag

```
sudo docker image tag myzmqserver:5.0 hfarkhari/client_server_zmq:server_5.0
sudo docker image tag myzmqclient:5.0 hfarkhari/client_server_zmq:client_5.0
```

Push

```
sudo docker login
sudo docker image push hfarkhari/client_server_zmq:server_5.0
sudo docker image push hfarkhari/client_server_zmq:client_5.0
```

Server: * : 5555
Client: myzmqserver-service : 5555

Run with Docker

Networks

Create Network

```
sudo docker network create mynet
```

```
sudo docker run -it --rm --net=mynet --name=myzmqserver-service myzmqserver:5.0
sudo docker run -it --rm --net=mynet myzmqclient:5.0
```

Minikube

Networks

Client-Server
(ZMQ)

server: * : 5555
client: myzmqserver-service : 5555

Run with Docker

```
usr@usr:~$ sudo docker run -it --rm --net=mynet --name=myzmqserver-service myzmqserver:5.0
The server app is running.
Server is ready to recieve msg from client:

Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
Received request: b'Hello'
```

```
usr@usr:~$ sudo docker run -it --rm --net=mynet myzmqclient:5.0

ZMQ client version, use "myzmqserver-service" as server ip, tcp protocol, and 5555 port
Connecting to server…
Sending request 0 …
Received reply 0 [ b'World' ]
Sending request 1 …
Received reply 1 [ b'World' ]
Sending request 2 …
Received reply 2 [ b'World' ]
Sending request 3 …
Received reply 3 [ b'World' ]
Sending request 4 …
Received reply 4 [ b'World' ]
Sending request 5 …
Received reply 5 [ b'World' ]
Sending request 6 …
Received reply 6 [ b'World' ]
Sending request 7 …
Received reply 7 [ b'World' ]
Sending request 8 …
Received reply 8 [ b'World' ]
Sending request 9 …
Received reply 9 [ b'World' ]
usr@usr:~$
```

Networks

Docker Network

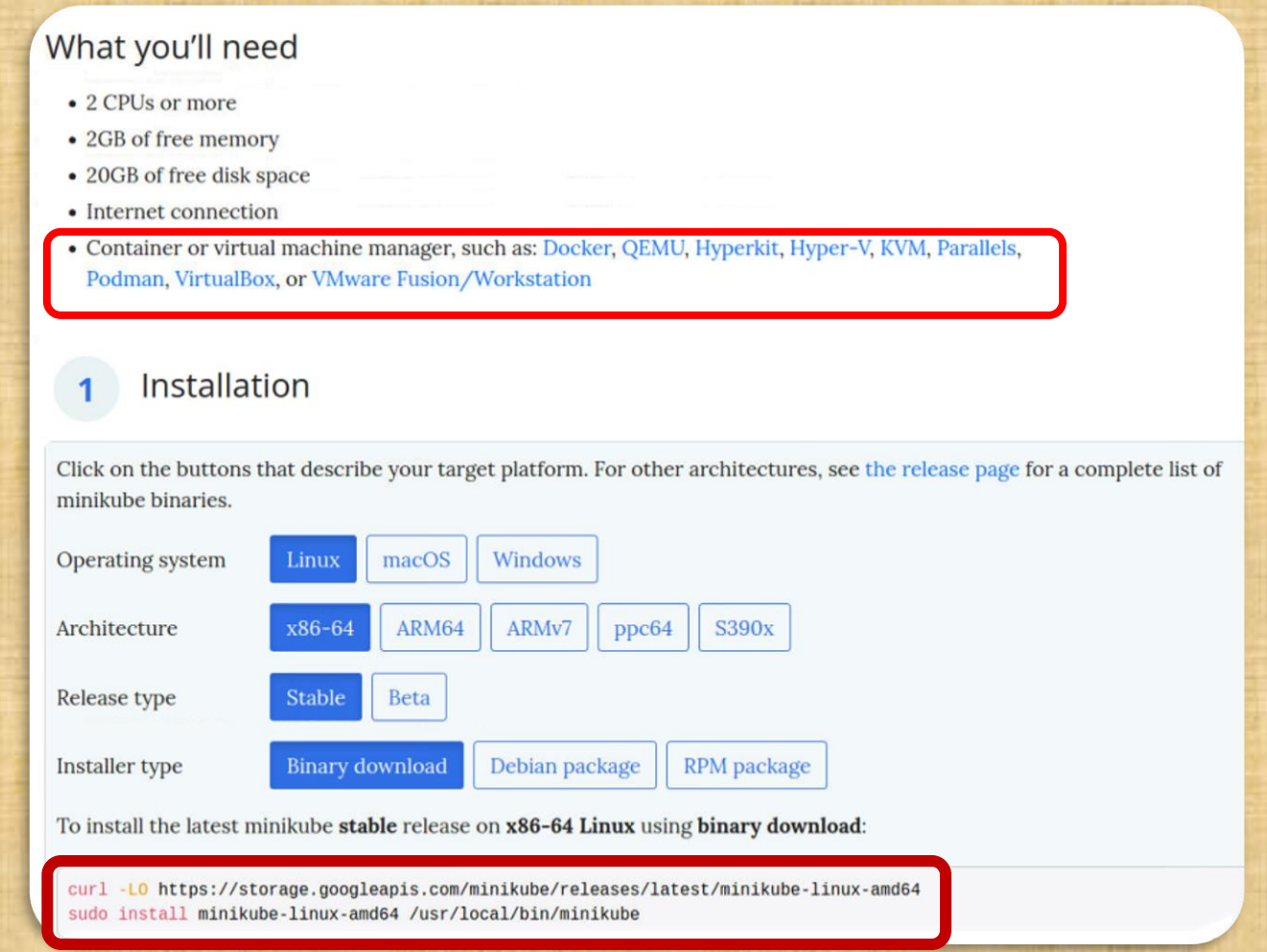
Run with Docker

```
usr@usr:~$ sudo docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
[redacted]     bridge    bridge    local
               host      host      local
               mynet     bridge    local
               none      null      local
```

Network	
Bridge	By default Connect containers with IP. Should set server IP inside the client sudo docker inspect myzmqserver-service | grep IPAddress
Host	Connect containers with ports only. Client should use localhost (0.0.0.0) to connect server. If client use server name, it can not connect to the server.
mynet	User-defined bridge (Similar in Minikube) When running server, should assign a name for it. (DNS name) --name=myzmqserver-service Client should use server name to connect server.

Install Minikube

https://minikube.sigs.k8s.io/docs/start/



What you'll need

- 2 CPUs or more
- 2GB of free memory
- 20GB of free disk space
- Internet connection
- Container or virtual machine manager, such as: Docker, QEMU, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMware Fusion/Workstation

1 Installation

Click on the buttons that describe your target platform. For other architectures, see the release page for a complete list of minikube binaries.

Operating system: Linux | macOS | Windows

Architecture: x86-64 | ARM64 | ARMv7 | ppc64 | S390x

Release type: Stable | Beta

Installer type: Binary download | Debian package | RPM package

To install the latest minikube **stable** release on **x86-64 Linux** using **binary download**:

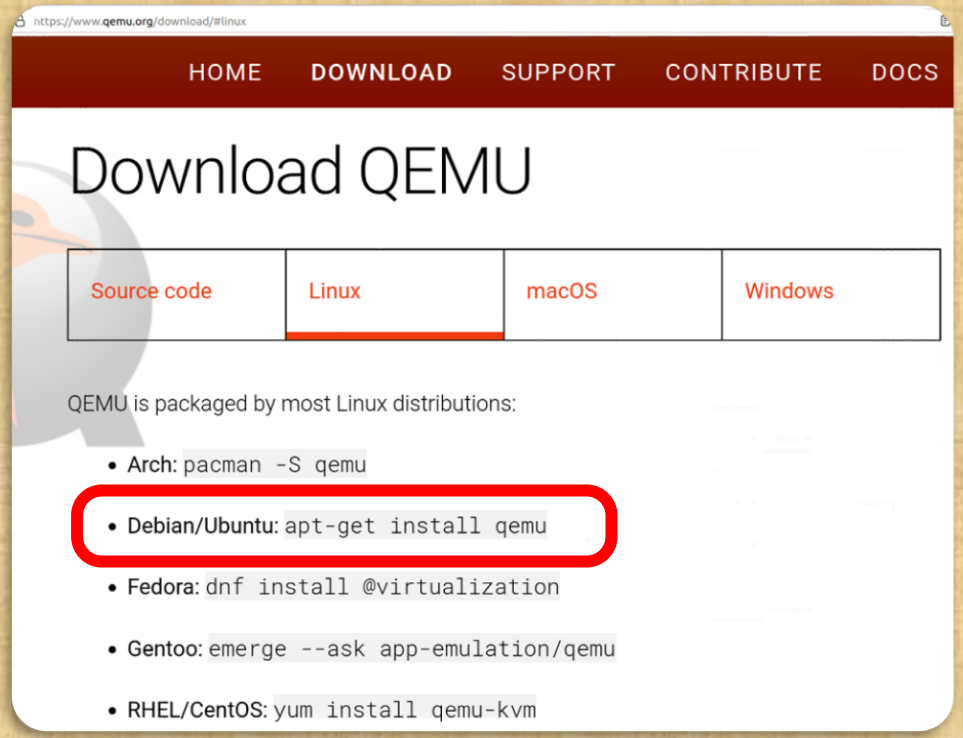
```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

Minikube

Install driver

Docker, QEMU, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMware Fusion/Workstation

https://www.qemu.org/download/#linux



QEMU-KVM Installation (recommend)

https://help.ubuntu.com/community/KVM/Installation

sudo apt-get install qemu-kvm libvirt-daemon-system libvirt-clients bridge-utils

Minikube Installation (recommend)

https://www.fosstechnix.com/how-to-install-minikube-on-ubuntu-22-04-lts/

Motherboard (Bios)

Check Virtualization be activated.

Intel CPU (Virtualization)

AMD CPU (SVM Mode → Enable)

Networks

Yaml File

Minikube

Networks

myzmqclient.yaml
~/Desktop/Minikube1/docker_files

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myzmqclient
spec:
  selector:
    matchLabels:
      run: myzmqclient
  replicas: 1
  template:
    metadata:
      labels:
        run: myzmqclient
    spec:
      containers:
      - name: myzmqclient
        image: hfarkhari/client_server_zmq:client_5.0
        ports:
        - containerPort: 5555
```

myzmqserver.yaml
~/Desktop/Minikube1/docker_files

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myzmqserver
  labels:
    app: myzmqserver
spec:
  replicas: 1
  selector:
    matchLabels:
      app: myzmqserver
  template:
    metadata:
      labels:
        app: myzmqserver
    spec:
      containers:
      - name: myzmqserver
        image: hfarkhari/client_server_zmq:server_5.0
        ports:
        - containerPort: 5555
---
apiVersion: v1
kind: Service
metadata:
  name: myzmqserver-service
  labels:
    app: myzmqserver
spec:
  selector:
    app: myzmqserver
  type: NodePort
  ports:
  - port: 5555
#    nodePort: 31000
    targetPort: 5555
    protocol: TCP
    name: http
```

Minikube

Minikube Commands

minikube start
minikube start –driver=qemu2
minikube start --driver=virtualbox
sudo minikube start

minikube dashboard

kubectl apply -f <name.yaml>
kubectl apply -f myzmqserver.yaml
kubectl apply -f myzmqclient.yaml

kubectl get pods
watch kubectl get pod

kubectl logs <pod name>

Stop running pods/services/…

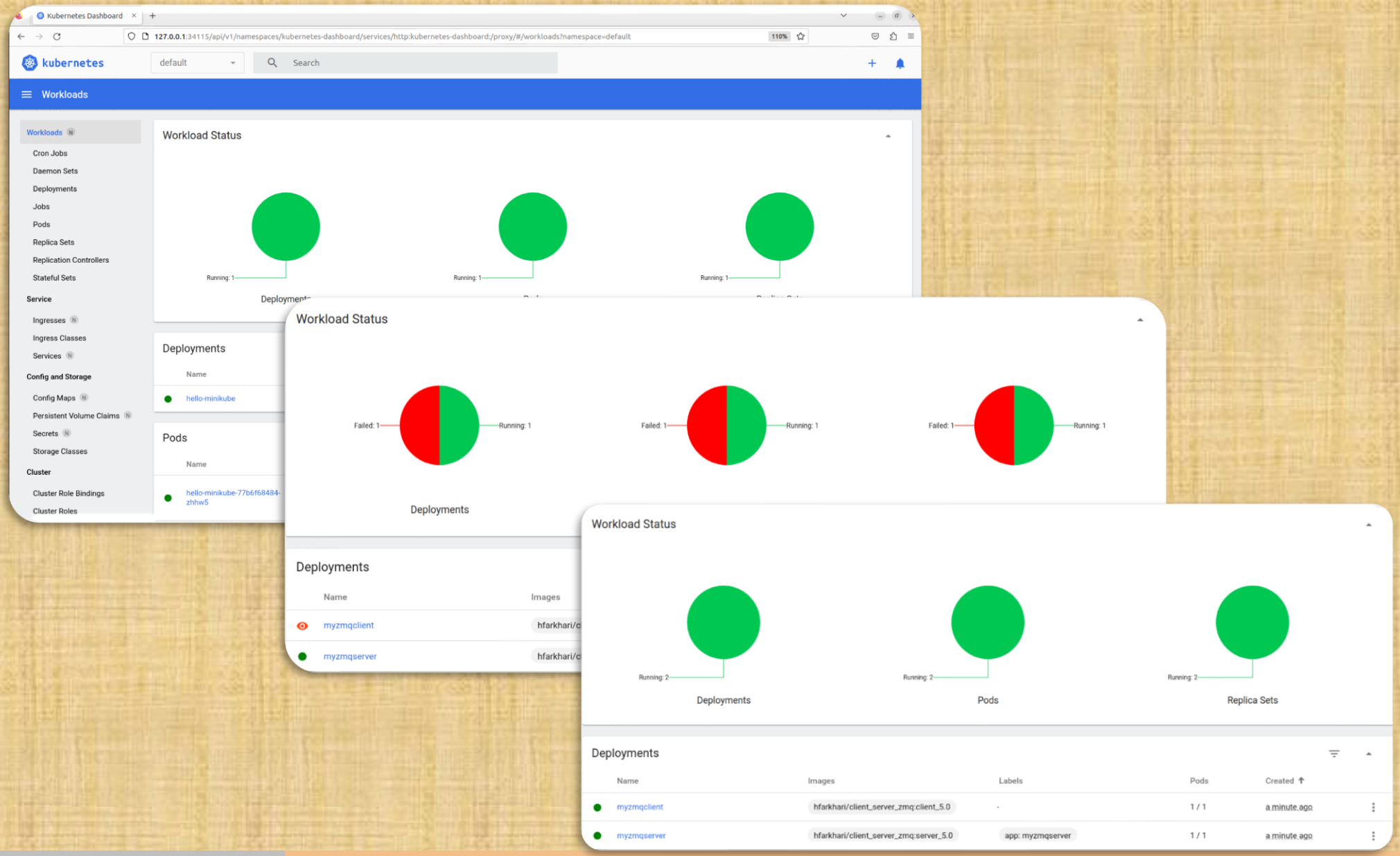
kubectl delete pod <pod name>
kubectl delete -f <name.yaml>

kubectl delete service –all
kubectl delete deployment –all
kubectl delete pod –all

minikube stop
minikube delete

Networks

Minikube dashboard



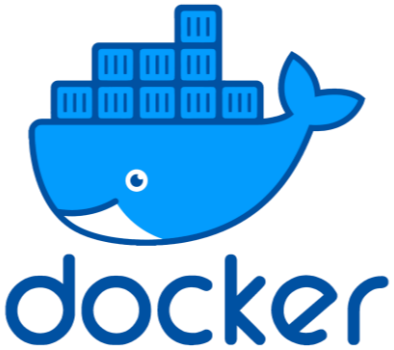
Minikube

Networks

Thanks

Any Question ?

kubernetes



Ubuntu