Step1: <https://docs.docker.com/desktop/install/linux/>

1. modprobe kvm
2. modprobe kvm\_intel # Intel processors
3. modprobe kvm\_amd # AMD processors
4. Kvm-ok
5. lsmod | grep kvm

kvm\_amd 167936 0

ccp 126976 1 kvm\_amd

kvm 1089536 1 kvm\_amd

irqbypass 16384 1 kvm

1. ls -al /dev/kvm
2. sudo usermod -aG kvm $USER

To know about OS: cat /etc/os-release

Step 2: <https://docs.docker.com/desktop/install/linux/ubuntu/>

1. sudo apt install gnome-terminal
2. <https://docs.docker.com/engine/install/ubuntu/#install-using-the-repository>
   1. Run below command# Add Docker's official GPG key:

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

* 1. sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
  2. docker run -it ubuntu bash (optional to run ubuntu)
  3. To uninstall: sudo apt-get purge docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-ce-rootless-extras

1. Install docker destop : [https://desktop.docker.com/linux/main/amd64/docker-desktop-amd64.deb?utm\_source=docker&utm\_medium=webreferral&utm\_campaign=docs-driven-download-linux-amd64&\_gl=1\*4oduzy\*\_gcl\_au\*MTY0NTA2MTQ2NC4xNzMwOTU3NzQx\*\_ga\*MTQ1NTUxNDEzOS4xNzMwOTU3NzQy\*\_ga\_XJWPQMJYHQ\*MTczMDk1Nzc0MS4xLjEuMTczMDk1ODQyMy4yMC4wLjA](https://desktop.docker.com/linux/main/amd64/docker-desktop-amd64.deb?utm_source=docker&utm_medium=webreferral&utm_campaign=docs-driven-download-linux-amd64&_gl=1*4oduzy*_gcl_au*MTY0NTA2MTQ2NC4xNzMwOTU3NzQx*_ga*MTQ1NTUxNDEzOS4xNzMwOTU3NzQy*_ga_XJWPQMJYHQ*MTczMDk1Nzc0MS4xLjEuMTczMDk1ODQyMy4yMC4wLjA).
2. Install the package:

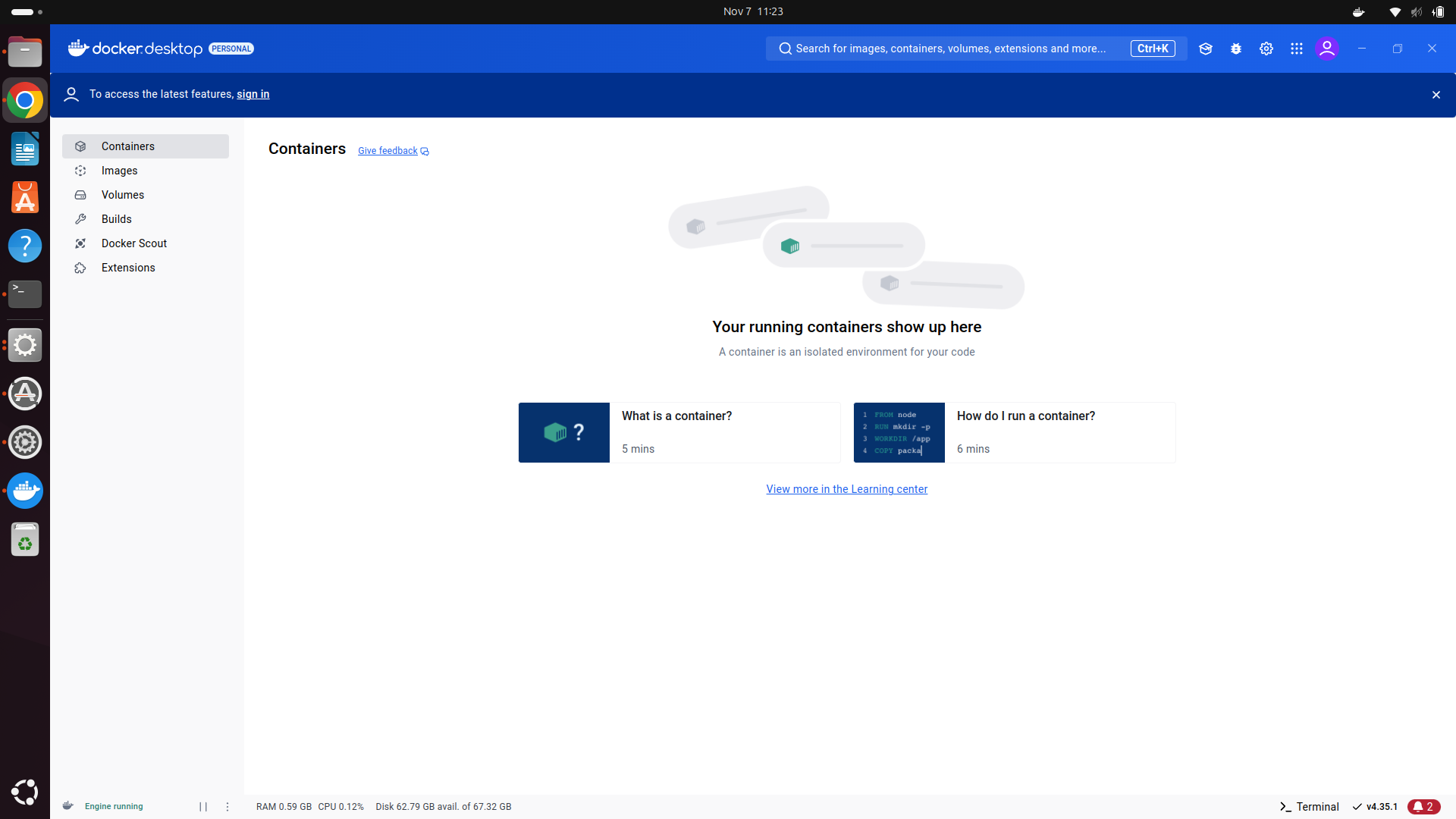
sudo apt-get update

sudo apt-get install ./docker-desktop-amd64.deb

1. Launch docker desktop by search or run below command:

systemctl --user start docker-desktop

1. To enable Docker Desktop to start on sign in: systemctl --user enable docker-desktop
2. To stop Docker Desktop: systemctl --user stop docker-desktop
3. Upgrade: sudo apt-get install ./docker-desktop-<arch>.deb



Done!!!

Install kubernetes:

curl -LO "https://dl.k8s.io/release/**$(**curl -L -s <https://dl.k8s.io/release/stable.txt>**)**/bin/linux/amd64/kubectl"

curl -LO "https://dl.k8s.io/release/**$(**curl -L -s https://dl.k8s.io/release/stable.txt**)**/bin/linux/amd64/kubectl.sha256"

echo "**$(**cat kubectl.sha256**)** kubectl" | sha256sum --checksudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

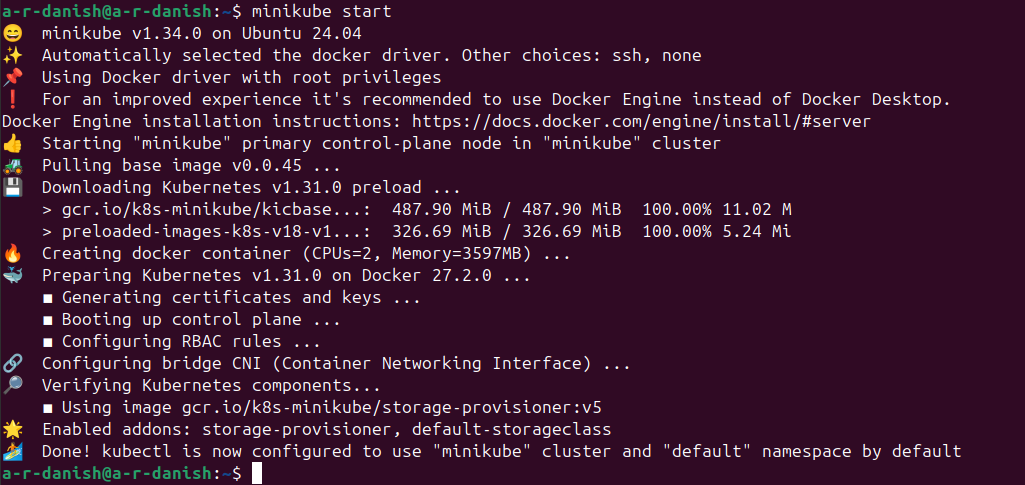
Done!!!

Install minikube:

Documentation:

<https://minikube.sigs.k8s.io/docs/start/?arch=%2Flinux%2Fx86-64%2Fstable%2Fbinary+download>

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



Done!!!

To stop all the services:

1. Stop Minikube: minikube stop
2. To completely remove the Minikube cluster: minikube delete
3. Stop Docker: sudo systemctl stop docker
4. To remove all Docker containers, images, networks, and volumes: docker system prune -a --volumes
5. Check and Kill Any Remaining Processes (Optional): docker ps -q | xargs docker stop

Start minikube after restart:

1. Add Your User to the Docker Group (if you haven’t already): sudo usermod -aG docker $USER
2. Log Out and Log Back In: newgrp docker
3. Start Minikube Without sudo: minikube start

Start a fresh Minikube instance with additional resources (if needed): minikube start --memory=4096 --cpus=2

**Check Available Memory**:free -h

**Check Available CPU Cores**:nproc

minikube start --memory=8192 --cpus=4

sudo systemctl restart docker

minikube start --memory=8192 --cpus=12

kubectl apply -f testing\_service.yaml

kubectl port-forward deployment/testing-service 8080:8080

List our all the services: minikube service list

minikube ip

http://<minikube-ip>:30080

Delete the deployment: kubectl delete deployment testing-service

Delete the service: kubectl delete service testing-service

Delete all the resources for a label: kubectl delete all -l app=testing-service

kubectl get pods

kubectl get deployments

kubectl logs platform-mongo-bb8b7984f-2x2b7 -f

Check Logs for All Pods (Optional) > kubectl logs -l app=service-1 --all-containers=true

Stream logs (real-time) > kubectl logs -f <pod-name>

Open a Shell Inside the Pod: kubectl exec -it <pod-name> -- /bin/bash

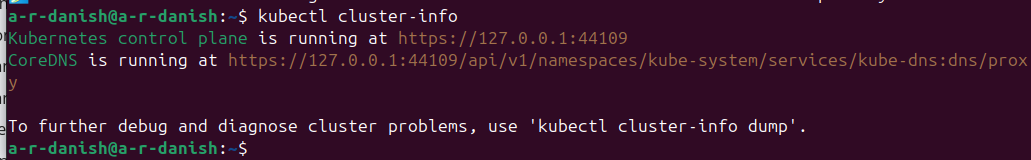
kubectl exec -it testing-service-75b5ff7bc5-cjhrr – /bin/bash

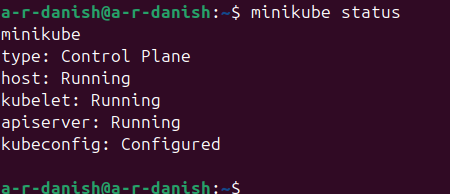
kubectl exec -it platform-mongo-bb8b7984f-2x2b7 – /bin/bash

kubectl exec -it platform-mongo-bb8b7984f-2x2b7 -- /bin/bash

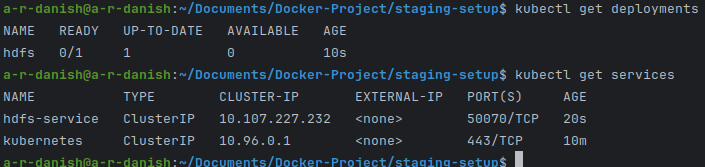
root@platform-mongo-bb8b7984f-2x2b7:/#

Testing:





1. To apply the YAML file to your Kubernetes cluster: **kubectl apply -f hdfs-deployment.yaml**
2. Check status of deployment: **kubectl get deployments**
3. Check the status of the service and its endpoints: **kubectl get services**



kubectl apply -f hdfs-deployment.yaml

kubectl port-forward service/hdfs-service 50070:50070

kubectl scale deployment hdfs --replicas=3

kubectl delete -f hdfs-deployment.yaml

Check Pod Events for Detailed Information

kubectl describe pod -l app=hdfs

kubectl describe nodes

Check for Storage or Persistent Volume Claims (if applicable):

kubectl get pvc

kubectl get networkpolicy --all-namespaces

Verify Kubernetes Networking Settings:

kubectl get networkpolicy --all-namespaces

minikube ssh

ping -c 4 google.com

If using Minikube, try enabling a CNI (Container Network Interface) plugin to manage networking

If login issue change configuration:

docker login -u helloardanish@gmail.com

Password:

Error saving credentials: error storing credentials - err: exit status 1, out: `error storing credentials - err: exit status 1, out: `pass not initialized: exit status 1: Error: password store is empty. Try "pass init".``

docker login -u helloardanish@gmail.com

Password:

WARNING! Your password will be stored unencrypted in /home/a-r-danish/.docker/config.json.

Configure a credential helper to remove this warning. See

https://docs.docker.com/engine/reference/commandline/login/#credential-stores

service docker stop rm ~/.docker/config.json service docker start

kubectl config view

apiVersion: v1

clusters: null

contexts: null

current-context: ""

kind: Config

preferences: {}

users: null

kubectl config view

apiVersion: v1

clusters:

- cluster:

certificate-authority: /home/a-r-danish/.minikube/ca.crt

extensions:

- extension:

last-update: Thu, 07 Nov 2024 12:39:22 IST

provider: minikube.sigs.k8s.io

version: v1.34.0

name: cluster\_info

server: https://127.0.0.1:44109

name: minikube

contexts:

- context:

cluster: minikube

extensions:

- extension:

last-update: Thu, 07 Nov 2024 12:39:22 IST

provider: minikube.sigs.k8s.io

version: v1.34.0

name: context\_info

namespace: default

user: minikube

name: minikube

current-context: minikube

kind: Config

preferences: {}

users:

- name: minikube

user:

client-certificate: /home/a-r-danish/.minikube/profiles/minikube/client.crt

client-key: /home/a-r-danish/.minikube/profiles/minikube/client.key

=====

Testing a service inside minikube using jdk 1.8

service docker stop rm ~/.docker/config.json service docker start

Testing\_service.yaml:

*#image: helloardanish/testing-build:latest*

apiVersion: apps/v1

kind: Deployment

metadata:

name: testing-service

spec:

replicas: 1

selector:

matchLabels:

app: testing-service

template:

metadata:

labels:

app: testing-service

spec:

containers:

- name: testing-service

image: helloardanish/testing-build:latest

ports:

- containerPort: 8080

*# kubectl apply -f testing\_service.yaml*

Testing\_service\_service.yaml >

apiVersion: v1

kind: Service

metadata:

name: testing-service

spec:

selector:

app: testing-service

ports:

- protocol: TCP

port: 80 *# Exposes this port on the service*

targetPort: 8080 *# Maps to the container port*

*# nodePort: 30080 # The port you can access on your local machine*

type: NodePort

*# > minikube ip*

*# 192.168.58.2*

*# http://192.168.58.2:30080/test*

*#*

*# > minikube service testing-service*

============

Test.yaml

*# YAML file with two deployments and two services*

*# Deployment for Service 1 (creates a pod for service 1)*

apiVersion: apps/v1

kind: Deployment

metadata:

name: service-1

spec:

replicas: 1

selector:

matchLabels:

app: service-1

template:

metadata:

labels:

app: service-1

spec:

containers:

- name: service-1

image: myservice1-image

ports:

- containerPort: 8080

---

*# Deployment for Service 2 (creates a pod for service 2)*

apiVersion: apps/v1

kind: Deployment

metadata:

name: service-2

spec:

replicas: 1

selector:

matchLabels:

app: service-2

template:

metadata:

labels:

app: service-2

spec:

containers:

- name: service-2

image: myservice2-image

ports:

- containerPort: 8081

---

*# Service for Service 1*

apiVersion: v1

kind: Service

metadata:

name: service-1

spec:

selector:

app: service-1

ports:

- protocol: TCP

port: 80

targetPort: 8080

type: NodePort

---

*# Service for Service 2*

apiVersion: v1

kind: Service

metadata:

name: service-2

spec:

selector:

app: service-2

ports:

- protocol: TCP

port: 80

targetPort: 8081

type: NodePort