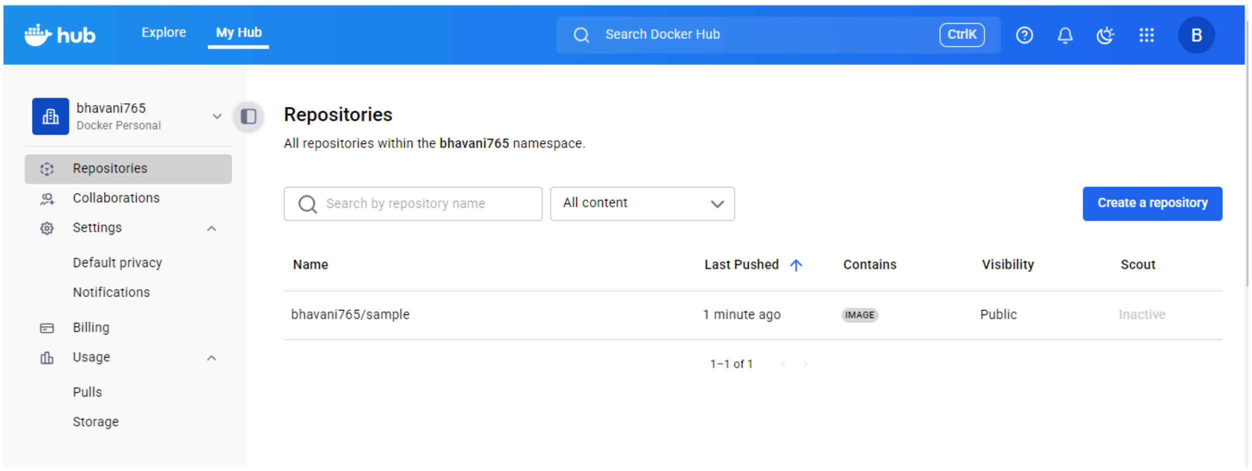
## Week-8: Integrate Kubernetes and Docker.

### Step 1: Push the local docker image into docker hub registry

1. Create a login in docker hub (https://hub.docker.com/)
2. After the email verification, Login into the docker hub using credentials
3. Click on Repositories → Click on create repo→ Give repo name and click on create
4. Check the Repositories



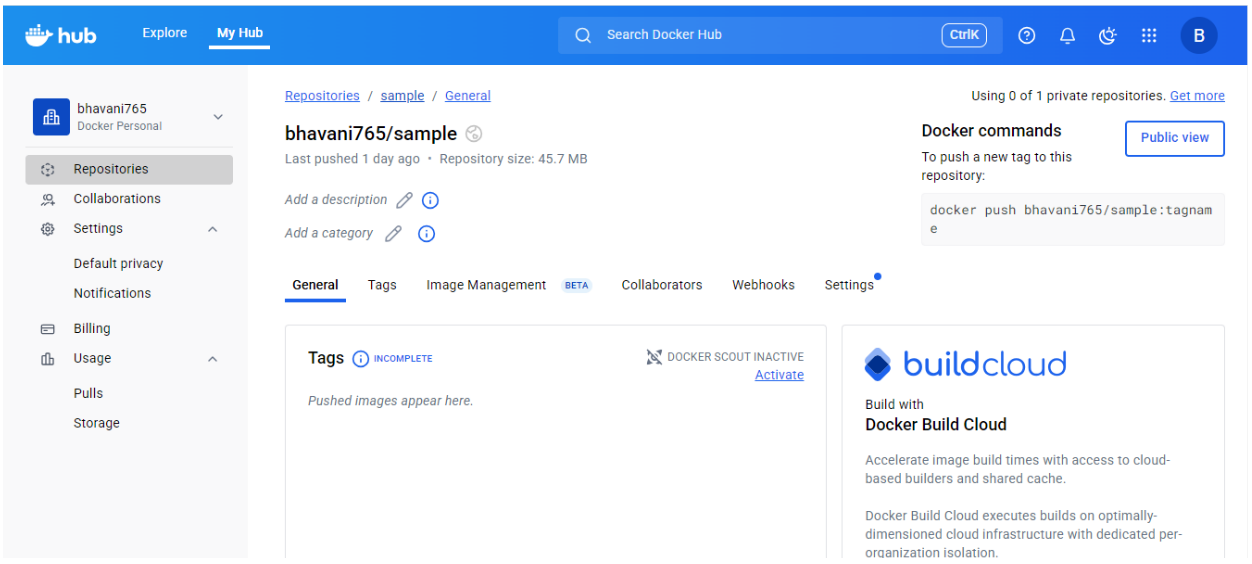
Repositories

All repositories within the **bhavani765** namespace.

| Name | Last Pushed | Contains | Visibility | Scout |
|---|---|---|---|---|
| bhavani765/sample | 1 minute ago | IMAGE | Public | Inactive |

1–1 of 1

Click on repository name, there are no images initially.



Repositories / sample / General

Using 0 of 1 private repositories. Get more

bhavani765/sample

Last pushed 1 day ago • Repository size: 45.7 MB

*Add a description*

*Add a category*

Docker commands

To push a new tag to this repository:

```
docker push bhavani765/sample:tagname
```

Tags INCOMPLETE

DOCKER SCOUT INACTIVE

*Pushed images appear here.*

Build with

Docker Build Cloud

Accelerate image build times with access to cloud-based builders and shared cache.

Docker Build Cloud executes builds on optimally-dimensioned cloud infrastructure with dedicated per-organization isolation.

5. To push an image to Docker Hub, Open Docker desktop, click on sign in→ click on proceed to Docker Desktop
6. Open command prompt or the terminal in Docker Desktop, and execute the following commands:
   1. **docker login:** authenticate your Docker client to Docker Hub or another Docker registry (like AWS ECR, GitLab, etc.)

```
C:\Users\Admin>docker login
Authenticating with existing credentials... [Username: bhavani765]

i Info → To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded
```

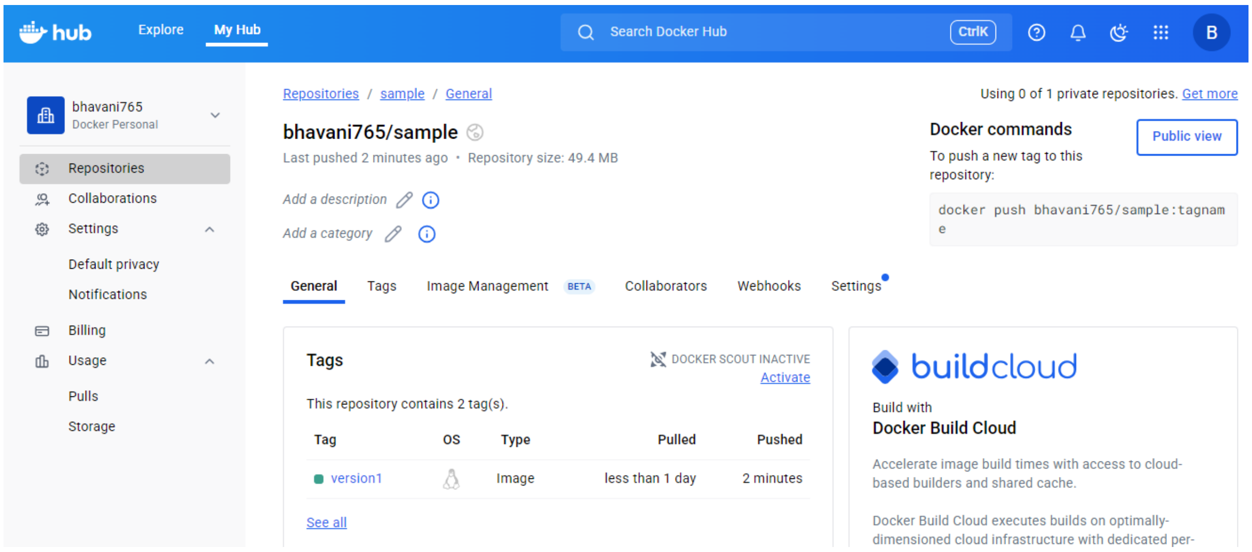
2. **docker tag:** tag the local docker image to the remote repository

```
C:\Users\Admin>docker tag registrationapplication:version1 bhavani765/sample:version1
```

3. **docker push:** Push the local docker image to the remote repository using docker push command

```
C:\Users\Admin>docker push bhavani765/sample:version1
The push refers to repository [docker.io/bhavani765/sample]
be1bdd00985c: Pushed
2e1c130fa3ec: Layer already exists
8d53da260408: Layer already exists
6ff643c4af9d: Pushed
58640652b40d: Pushed
8b91b88d5577: Layer already exists
824416e23423: Layer already exists
4128ba76256a: Layer already exists
84c8c79126f6: Layer already exists
version1: digest: sha256:1c2147a2bacb90ac226bf3f12c61c1f962b566da0c8e8ce2208efcd6e59094ac size: 856
```

7. Now go to Docker Hub → Repositories → click on repository name, we will find the pushed docker image stored in the registry.



Using 0 of 1 private repositories. Get more

bhavani765/sample

Last pushed 2 minutes ago • Repository size: 49.4 MB

Add a description

Add a category

Docker commands

To push a new tag to this repository:

```
docker push bhavani765/sample:tagname
```

General | Tags | Image Management BETA | Collaborators | Webhooks | Settings

Tags

DOCKER SCOUT INACTIVE Activate

This repository contains 2 tag(s).

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| version1 | | Image | less than 1 day | 2 minutes |

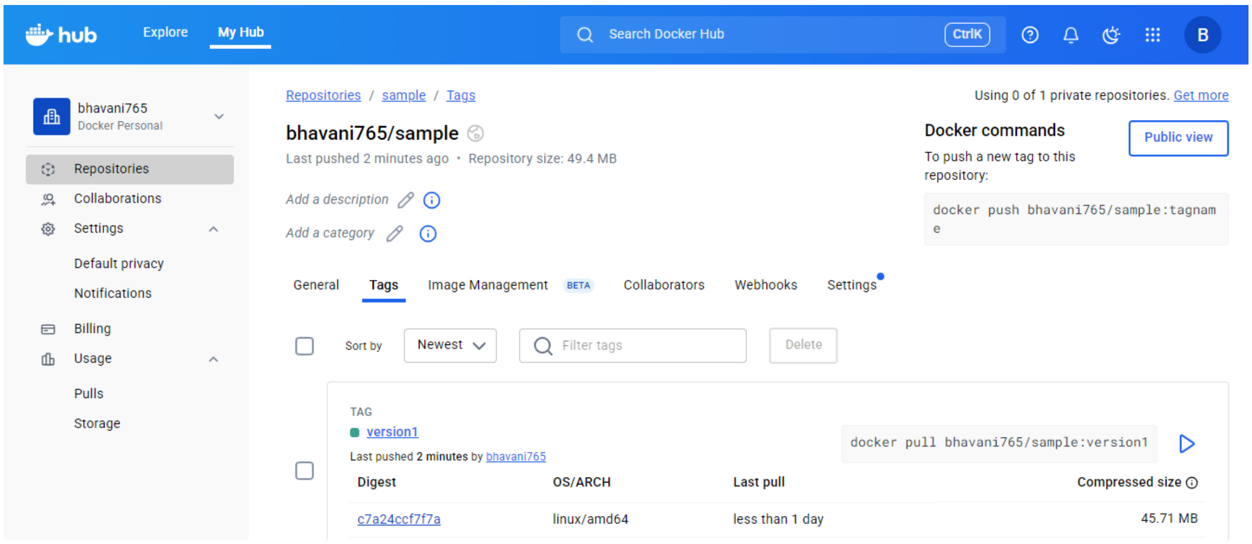
See all

buildcloud

Build with Docker Build Cloud

Accelerate image build times with access to cloud-based builders and shared cache.

Docker Build Cloud executes builds on optimally-dimensioned cloud infrastructure with dedicated per-



Using 0 of 1 private repositories. Get more

bhavani765/sample

Last pushed 2 minutes ago • Repository size: 49.4 MB

Add a description

Add a category

Docker commands

To push a new tag to this repository:

```
docker push bhavani765/sample:tagname
```

General | Tags | Image Management BETA | Collaborators | Webhooks | Settings

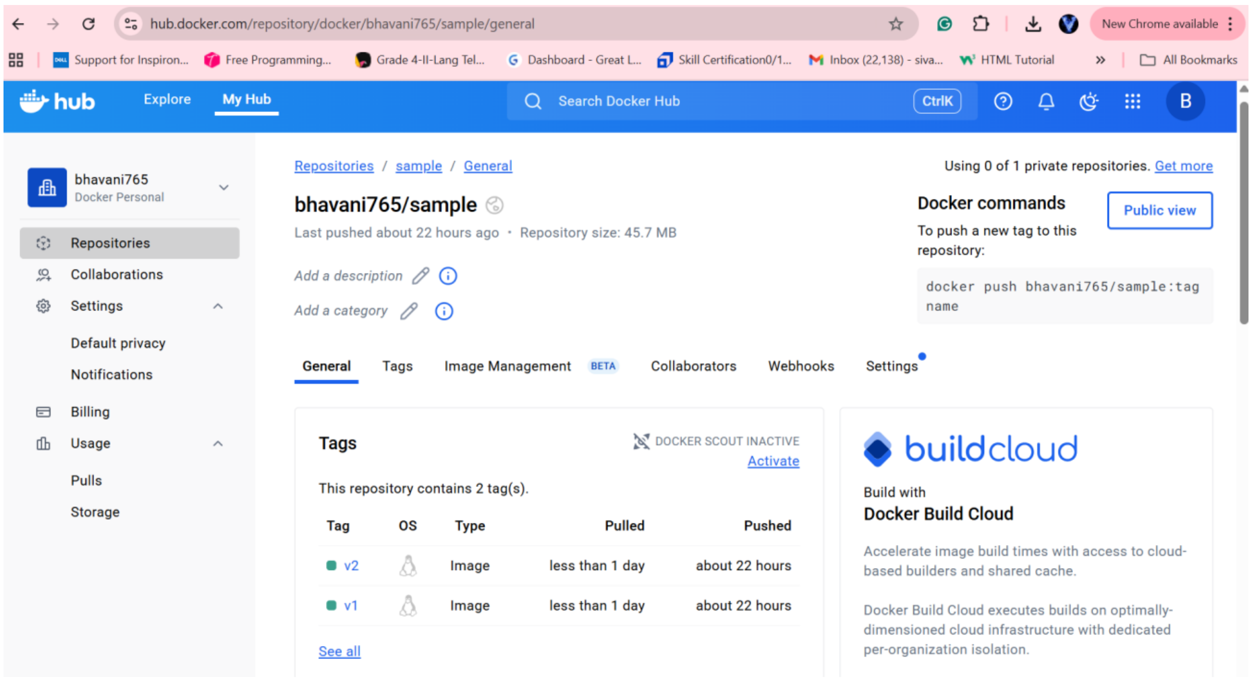
TAG

version1

Last pushed 2 minutes by bhavani765

```
docker pull bhavani765/sample:version1
```

| Digest | OS/ARCH | Last pull | Compressed size |
|---|---|---|---|
| c7a24ccf7f7a | linux/amd64 | less than 1 day | 45.71 MB |


hub.docker.com/repository/docker/bhavani765/sample/general

Repositories / sample / General

bhavani765/sample

Last pushed about 22 hours ago • Repository size: 45.7 MB

Using 0 of 1 private repositories. Get more

Docker commands

To push a new tag to this repository:

```
docker push bhavani765/sample:tagname
```

Tags

This repository contains 2 tag(s).

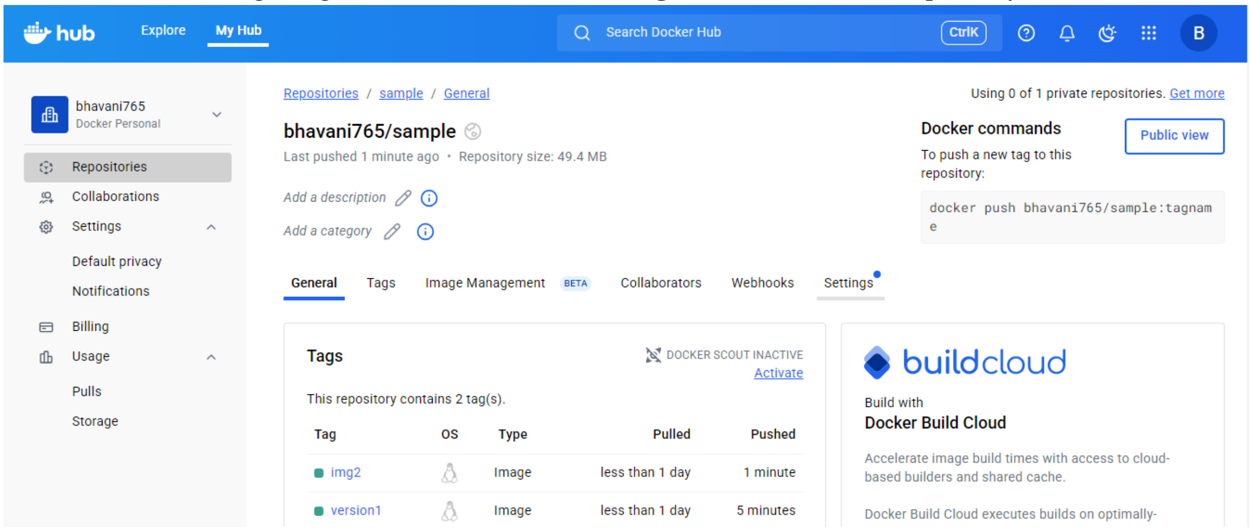
| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| v2 | | Image | less than 1 day | about 22 hours |
| v1 | | Image | less than 1 day | about 22 hours |

See all

8. Multiple images can be pushed to the repository

```
C:\Users\Admin>docker tag pyflskapp:v1 bhavani765/sample:img2

C:\Users\Admin>docker push bhavani765/sample:img2
The push refers to repository [docker.io/bhavani765/sample]
008beda60caa: Layer already exists
2e1c130fa3ec: Layer already exists
84c8c79126f6: Layer already exists
8b91b88d5577: Layer already exists
824416e23423: Layer already exists
ccaa32836067: Already exists
8d53da260408: Layer already exists
4db27a2f1aab: Layer already exists
4128ba76256a: Layer already exists
img2: digest: sha256:fecac4c7b67c734f79886714024f84f4d246965dde7434d06cbb0905cde27478 size: 856
```

9. The following image versions **version1** and **img2** are in docker hub repository


Repositories / sample / General

bhavani765/sample

Last pushed 1 minute ago • Repository size: 49.4 MB

Using 0 of 1 private repositories. Get more

Docker commands

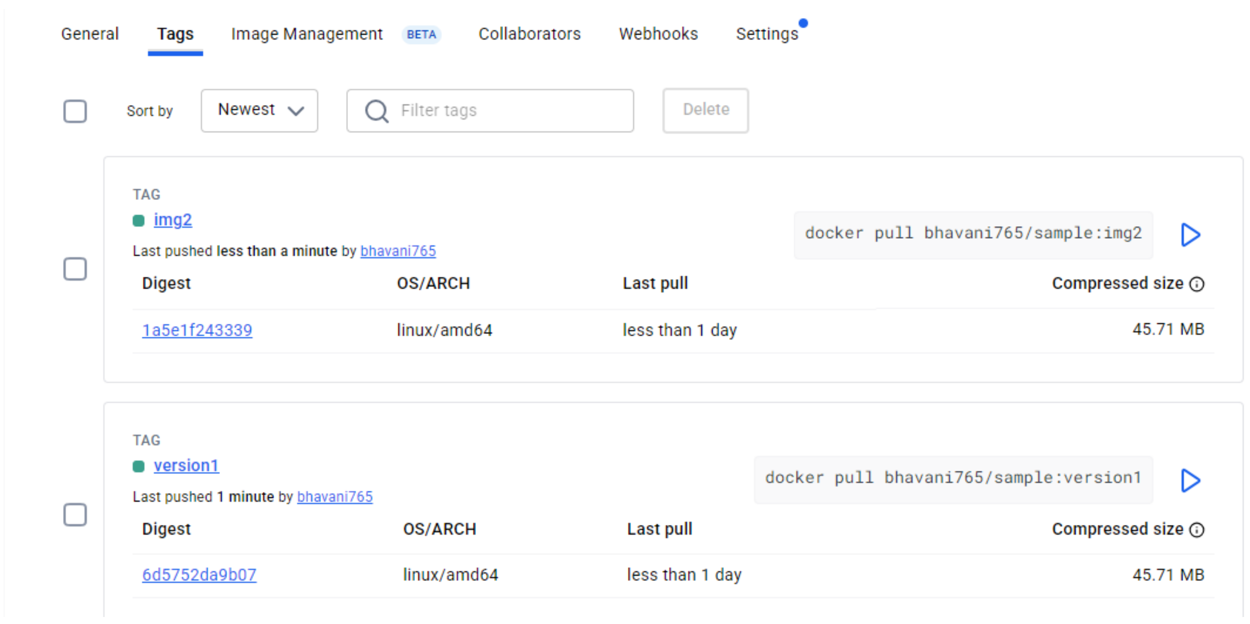
To push a new tag to this repository:

```
docker push bhavani765/sample:tagname
```

Tags

This repository contains 2 tag(s).

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| img2 | | Image | less than 1 day | 1 minute |
| version1 | | Image | less than 1 day | 5 minutes |



TAG
img2
Last pushed less than a minute by bhavani765

```
docker pull bhavani765/sample:img2
```

| Digest | OS/ARCH | Last pull | Compressed size |
|---|---|---|---|
| 1a5e1f243339 | linux/amd64 | less than 1 day | 45.71 MB |

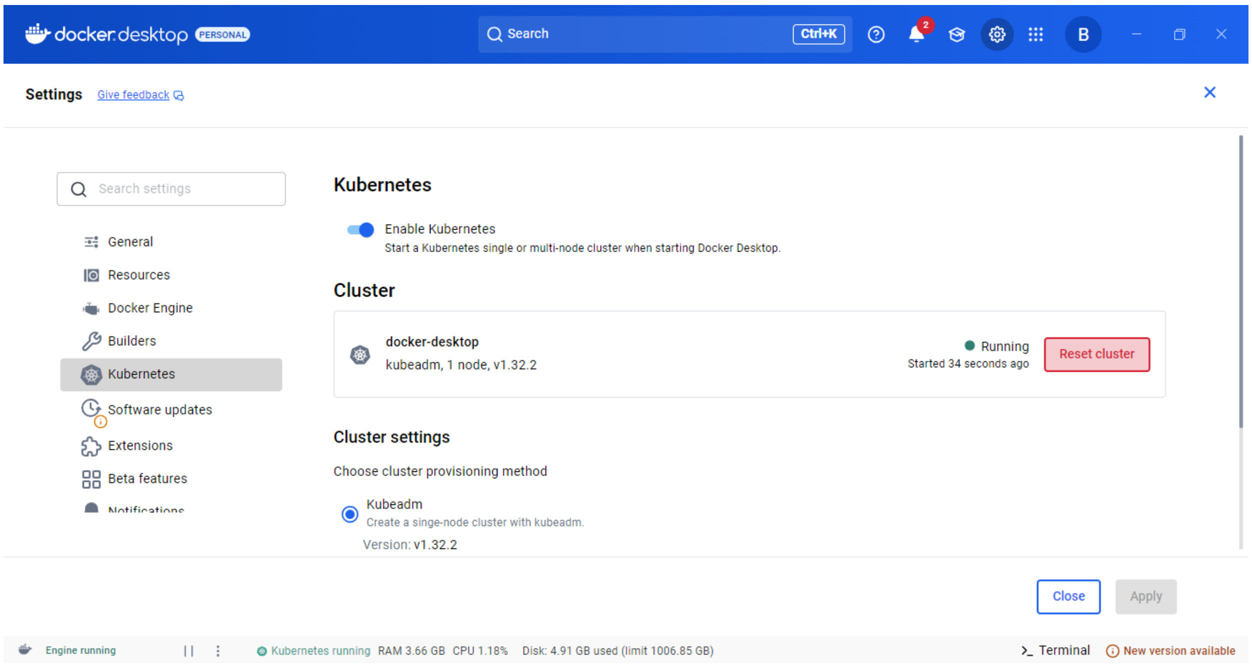
TAG
version1
Last pushed 1 minute by bhavani765

```
docker pull bhavani765/sample:version1
```

| Digest | OS/ARCH | Last pull | Compressed size |
|---|---|---|---|
| 6d5752da9b07 | linux/amd64 | less than 1 day | 45.71 MB |

**Step 2: Integrate Kubernetes in Docker Desktop**

1. Open Docker Desktop
2. Go to Settings → Kubernetes
3. Check "Enable Kubernetes"
4. Installs the kubectl CLI



Kubernetes

Enable Kubernetes [✓]
Start a Kubernetes single or multi-node cluster when starting Docker Desktop.

Cluster

docker-desktop
kubeadm, 1 node, v1.32.2
Running
Started 34 seconds ago
Reset cluster

Cluster settings

Choose cluster provisioning method
[✓ Kubeadm]
Create a singe-node cluster with kubeadm.
Version: v1.32.2

Kubernetes running RAM 3.66 GB CPU 1.18% Disk: 4.91 GB used (limit 1006.85 GB)

5. **Single-node Kubernetes cluster** is created inside your machine. It's good for local testing and learning. It doesn't support multiple nodes. For multiple nodes, we need to use Kind/Minikube/kubeadm
    a. Docker Desktop spins up a Kubernetes control-plane + worker node inside Docker.
    b. The node is called docker-desktop. This node is both **control-plane + worker node.** We **cannot directly add worker nodes** in Docker Desktop — it's a simplified, single-node setup just for local development.

c. This node runs:
   i. API server (port 6443),
   ii. Controller manager,
   iii. Scheduler,
   iv. etcd (the database),
   v. Kubelet (agent on the node).

d. Once Kubernetes is running, default system pods are deployed in kube-system namespace
   i. coredns (cluster DNS)
   ii. kube-proxy (networking)
   iii. metrics-server (for resource metrics)
   iv. storage-provisioner (for PVCs)

e. Docker Desktop updates ~/.kube/config with a new context called docker-desktop and then kubectl commands start working right away without extra setup

f. A local DNS (kubernetes.docker.internal) is set up and the Services you expose via NodePort can be accessed through **localhost:<nodePort>**

g. Docker Desktop provides a default storage class.

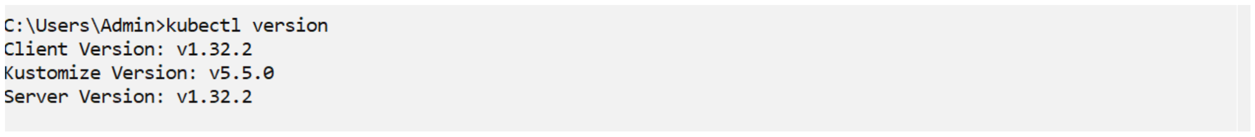
h. If you create a PersistentVolumeClaim (PVC), Kubernetes automatically provisions storage inside Docker Desktop.

6. Ensure both Docker Engine and Kubernetes are running in the Docker Desktop

**Step 3: Execute Kubernetes commands using kubectl**

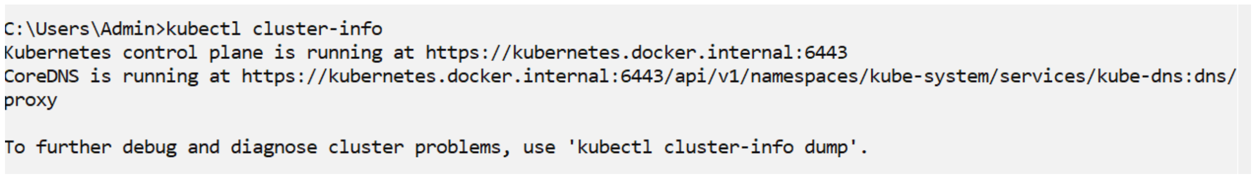
Open the command prompt or the terminal in the Docker Desktop and execute the following kubectl commands:

1. kubectl version → shows the version



```
C:\Users\Admin>kubectl version
Client Version: v1.32.2
Kustomize Version: v5.5.0
Server Version: v1.32.2
```

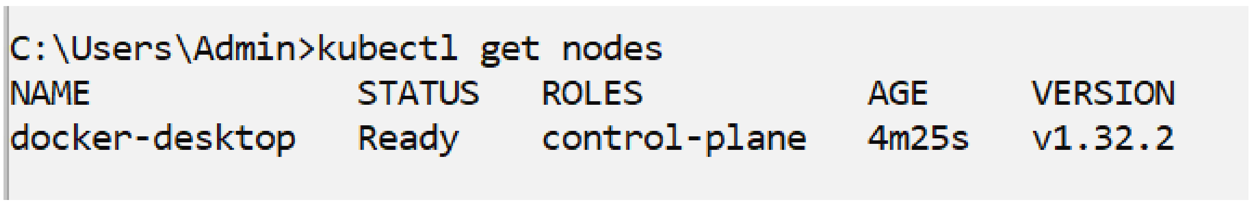
2. kubectl cluster-info →shows the info about the cluster.



```
C:\Users\Admin>kubectl cluster-info
Kubernetes control plane is running at https://kubernetes.docker.internal:6443
CoreDNS is running at https://kubernetes.docker.internal:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/
proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

Port 6443 is the default port of the Kubernetes API server. The API server is the “brain” of Kubernetes — all kubectl commands talk to it.
6443 is the port where the Kubernetes API server listens for requests from kubectl (or any client like Helm, CI/CD tools, etc.)

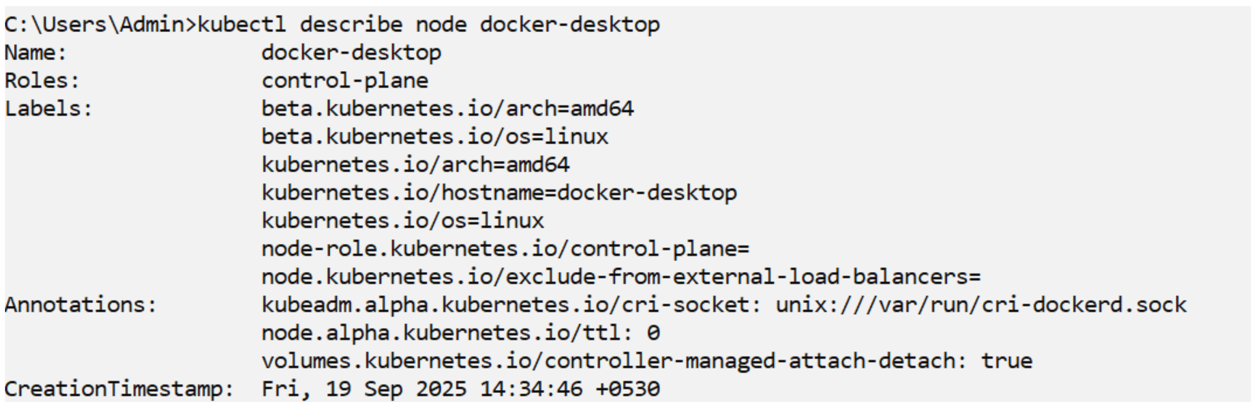
3. kubectl get nodes → List all nodes ( master node + worker nodes )



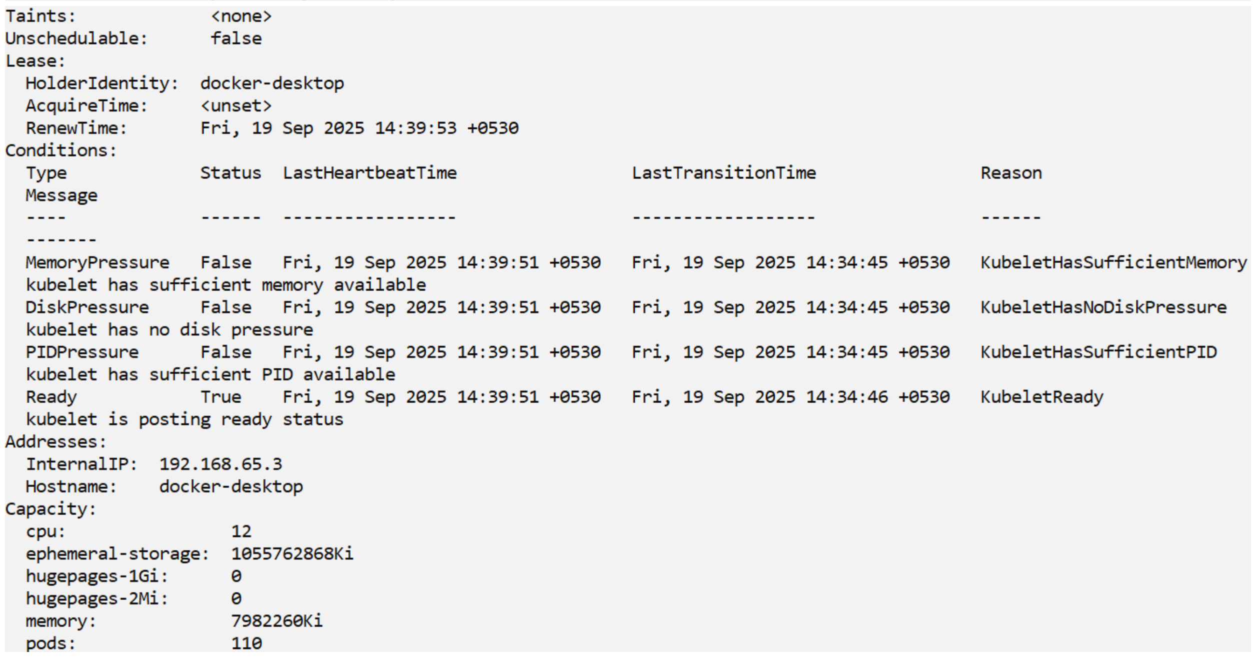
```
C:\Users\Admin>kubectl get nodes
NAME             STATUS   ROLES           AGE     VERSION
docker-desktop   Ready    control-plane   4m25s   v1.32.2
```

docker-desktop acts as both master node and worker node.

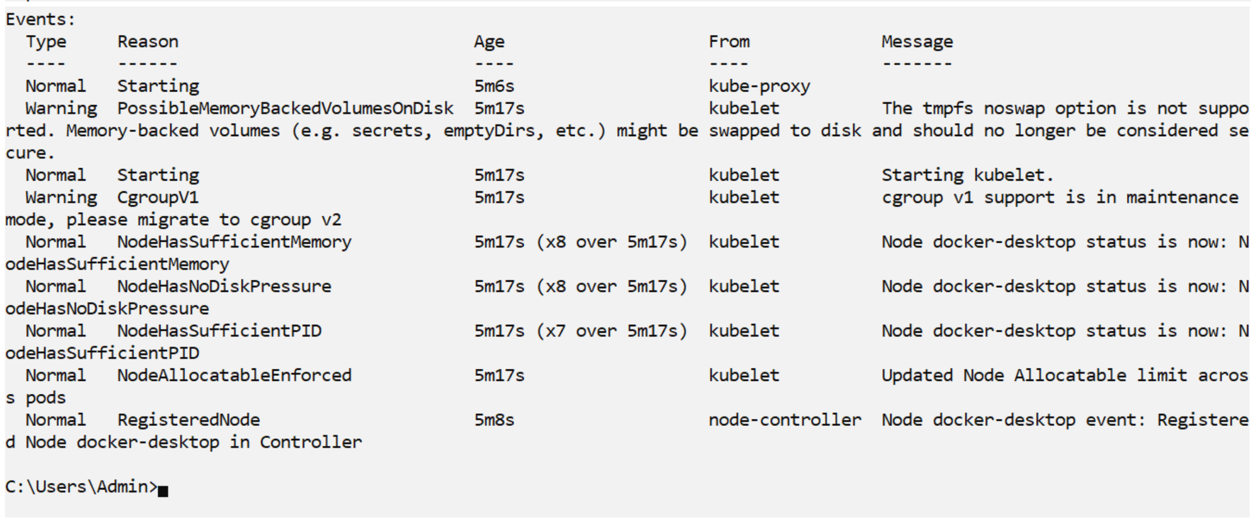
4. kubectl describe node <nodename> → gets the detailed information of a specific node



```
C:\Users\Admin>kubectl describe node docker-desktop
Name:               docker-desktop
Roles:              control-plane
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=docker-desktop
                    kubernetes.io/os=linux
                    node-role.kubernetes.io/control-plane=
                    node.kubernetes.io/exclude-from-external-load-balancers=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: unix:///var/run/cri-dockerd.sock
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Fri, 19 Sep 2025 14:34:46 +0530
```

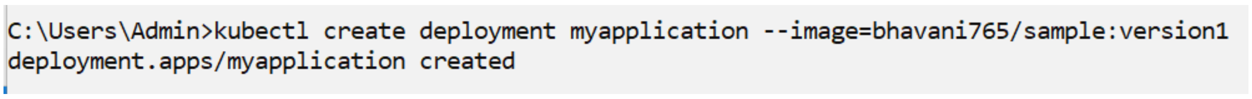


```
Taints:             <none>
Unschedulable:      false
Lease:
  HolderIdentity:  docker-desktop
  AcquireTime:     <unset>
  RenewTime:       Fri, 19 Sep 2025 14:39:53 +0530
Conditions:
  Type             Status  LastHeartbeatTime                 LastTransitionTime                Reason
  Message
  ----             ------  -----------------                 ------------------                ------
  -------
  MemoryPressure   False   Fri, 19 Sep 2025 14:39:51 +0530   Fri, 19 Sep 2025 14:34:45 +0530   KubeletHasSufficientMemory
  kubelet has sufficient memory available
  DiskPressure     False   Fri, 19 Sep 2025 14:39:51 +0530   Fri, 19 Sep 2025 14:34:45 +0530   KubeletHasNoDiskPressure
  kubelet has no disk pressure
  PIDPressure      False   Fri, 19 Sep 2025 14:39:51 +0530   Fri, 19 Sep 2025 14:34:45 +0530   KubeletHasSufficientPID
  kubelet has sufficient PID available
  Ready            True    Fri, 19 Sep 2025 14:39:51 +0530   Fri, 19 Sep 2025 14:34:46 +0530   KubeletReady
  kubelet is posting ready status
Addresses:
  InternalIP:  192.168.65.3
  Hostname:    docker-desktop
Capacity:
  cpu:                12
  ephemeral-storage:  1055762868Ki
  hugepages-1Gi:      0
  hugepages-2Mi:      0
  memory:             7982260Ki
  pods:               110
```



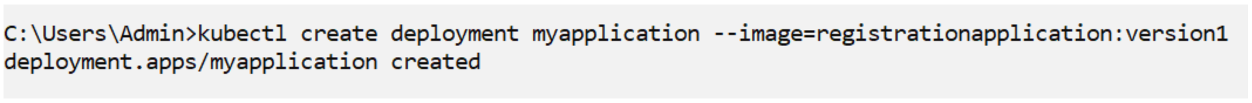
```
Events:
  Type     Reason                             Age                   From             Message
  ----     ------                             ----                  ----             -------
  Normal   Starting                           5m6s                  kube-proxy
  Warning  PossibleMemoryBackedVolumesOnDisk  5m17s                 kubelet          The tmpfs noswap option is not suppo
rted. Memory-backed volumes (e.g. secrets, emptyDirs, etc.) might be swapped to disk and should no longer be considered se
cure.
  Normal   Starting                           5m17s                 kubelet          Starting kubelet.
  Warning  CgroupV1                           5m17s                 kubelet          cgroup v1 support is in maintenance
mode, please migrate to cgroup v2
  Normal   NodeHasSufficientMemory            5m17s (x8 over 5m17s)  kubelet          Node docker-desktop status is now: N
odeHasSufficientMemory
  Normal   NodeHasNoDiskPressure              5m17s (x8 over 5m17s)  kubelet          Node docker-desktop status is now: N
odeHasNoDiskPressure
  Normal   NodeHasSufficientPID               5m17s (x7 over 5m17s)  kubelet          Node docker-desktop status is now: N
odeHasSufficientPID
  Normal   NodeAllocatableEnforced            5m17s                 kubelet          Updated Node Allocatable limit acros
s pods
  Normal   RegisteredNode                     5m8s                  node-controller  Node docker-desktop event: Registere
d Node docker-desktop in Controller

C:\Users\Admin>
```

5. kubectl create deployment → to create deployment
   **Syntax:** kubectl create deployment <deployment-name> --image=<image-name[:tag]>



```
C:\Users\Admin>kubectl create deployment myapplication --image=bhavani765/sample:version1
deployment.apps/myapplication created
```

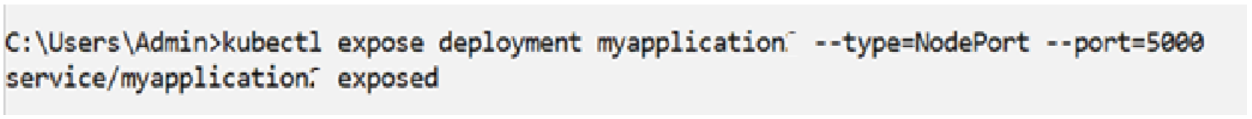
(OR)



```
C:\Users\Admin>kubectl create deployment myapplication --image=registrationapplication:version1
deployment.apps/myapplication created
```

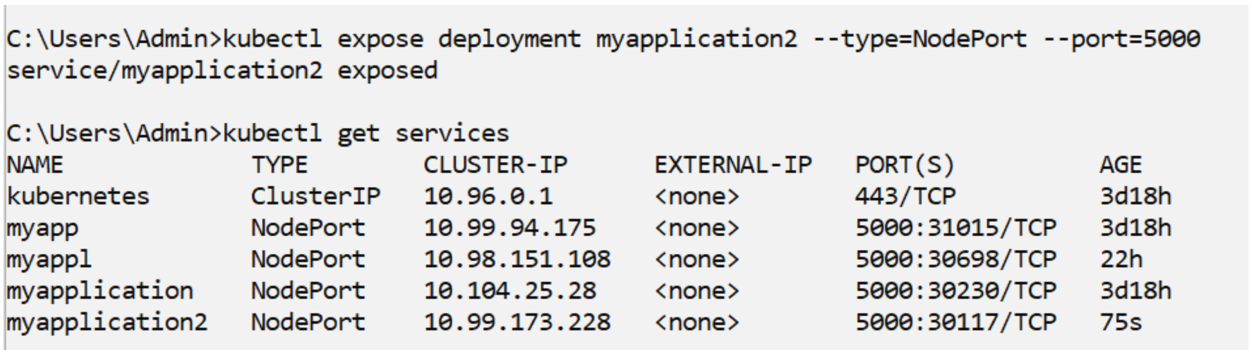
6. kubectl expose deployment → to expose the deployment as a service, so that it can be accessed from outside.

**Syntax:** kubectl expose deployment <deployment-name> --type=NodePort --port=<port-number>



```
C:\Users\Admin>kubectl expose deployment myapplication --type=NodePort --port=5000
service/myapplication exposed
```

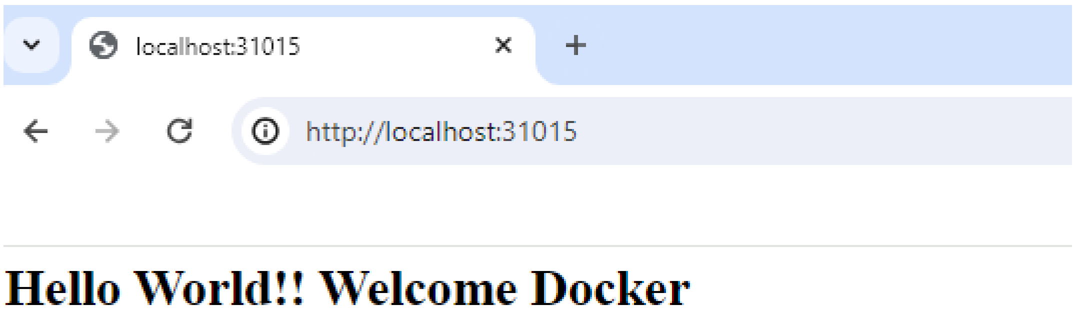
7. kubectl get svc → List all the services



```
C:\Users\Admin>kubectl expose deployment myapplication2 --type=NodePort --port=5000
service/myapplication2 exposed

C:\Users\Admin>kubectl get services
NAME             TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes       ClusterIP   10.96.0.1       <none>        443/TCP          3d18h
myapp            NodePort    10.99.94.175    <none>        5000:31015/TCP   3d18h
myappl           NodePort    10.98.151.108   <none>        5000:30698/TCP   22h
myapplication    NodePort    10.104.25.28    <none>        5000:30230/TCP   3d18h
myapplication2   NodePort    10.99.173.228   <none>        5000:30117/TCP   75s
```

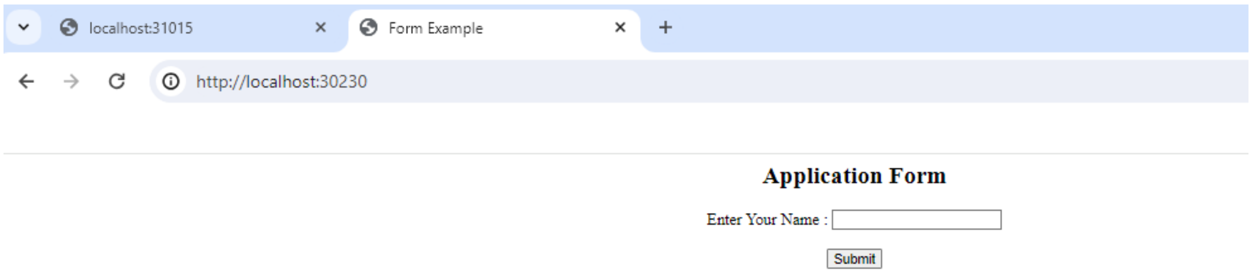
8. Now to access the containerized application from outside, open the web browser and type the url http://localhost:<NodePort>

a. Open web browser and type localhost:31015



# Hello World!! Welcome Docker

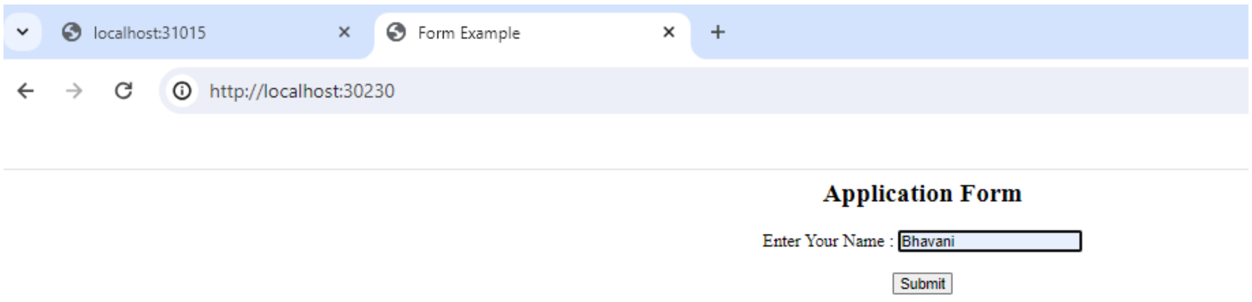
b. Open web browser and type localhost:30230



**Application Form**

Enter Your Name :

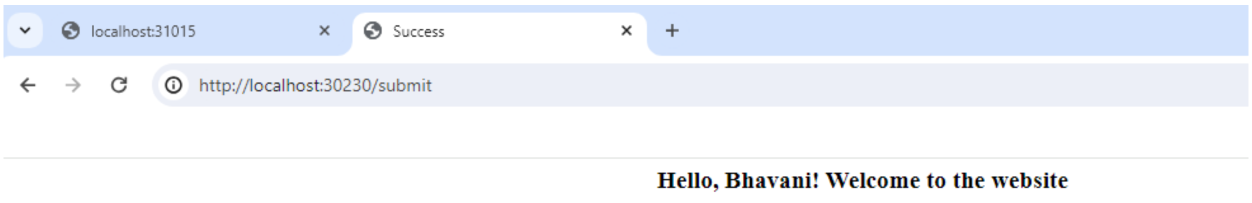
Submit



**Application Form**

Enter Your Name : Bhavani

Submit



Hello, Bhavani! Welcome to the website

9. kubectl get deployments → List all deployments
   [place ur outputs]

10. kubectl get pods → List all pods
    [place ur outputs]

11. kubectl scale deployment myapplication --replicas=4
    [place ur outputs]

12. kubectl get pods
    [place ur outputs]