Lab 09

Laboratory Exercise

Part 1: Deploy Single Container POD in Kubernetes

LAB EXERCISE

This LAB exercise shows you how to deploy a sample application in kubernetes.

Time to Complete
Approximately 15 Minutes

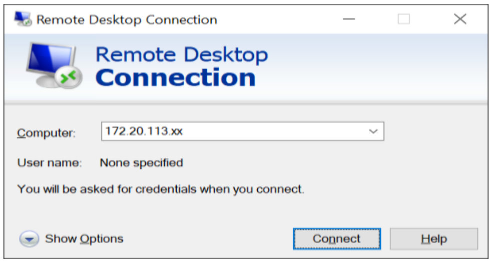
What You Need
Kubernetes Cluster should be running. If not start the cluster using the command “minikube start”

Question: What is the main difference between Kubernetes and Minikube?

For YAML Basic Syntax and Meaning, please read
https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html

From your machine logged-in to RP VPN, run Remote Desktop Connection to connect to the ubuntu Linux Virtual Machine (VM). Please login based on your assigned VM as shown below:

S/N	Name	VM	IP Address	User Name	Password
1	ABDUL SALIM BIN ABDUL RASHITH	LABC03 - 172.20.115.50	172.20.115.50	dockeradm	docker!2
2	CASPER LEOW YU HAN (LIAO YU HANG)	LABC03 - 172.20.115.51	172.20.115.51	dockeradm	docker!2
3	CHAN JUN ZHI, GLENN	LABC03 - 172.20.115.52	172.20.115.52	dockeradm	docker!2
4	CHIA WAI TAT	LABC03 - 172.20.115.53	172.20.115.53	dockeradm	docker!2
5	HOI WAI TECK	LABC03 - 172.20.115.54	172.20.115.54	dockeradm	docker!2
6	KOH JIN CAI DAEMIAN	LABC03 - 172.20.115.55	172.20.115.55	dockeradm	docker!2
7	KYAW KYAW OO	LABC03 - 172.20.115.56	172.20.115.56	dockeradm	docker!2
8	LUM YOKE FAI	LABC03 - 172.20.115.57	172.20.115.57	dockeradm	docker!2
9	MUHAMMAD FADHLI BIN MOHAMED NOOR	LABC03 - 172.20.115.58	172.20.115.58	dockeradm	docker!2
10	MUHAMMAD HILMEE BIN MD ALI	LABC03 - 172.20.115.59	172.20.115.59	dockeradm	docker!2
11	NG SAY WEE	LABC03 - 172.20.115.60	172.20.115.60	dockeradm	docker!2
12	NGUI WEILY	LABC03 - 172.20.115.61	172.20.115.61	dockeradm	docker!2
13	NU'MAN HARITH BIN NORRAIMI	LABC03 - 172.20.115.62	172.20.115.62	dockeradm	docker!2
14	RULY JANUAR FACHMI	LABC03 - 172.20.115.76	172.20.115.76	dockeradm	docker!2
15	SEAH SHIH WEI GEROME	LABC03 - 172.20.115.64	172.20.115.64	dockeradm	docker!2
16	SEAN CHENG ZHI WEI	LABC03 - 172.20.115.65	172.20.115.65	dockeradm	docker!2
17	SEY KOK SIONG	LABC03 - 172.20.115.66	172.20.115.66	dockeradm	docker!2
18	TAN JOON YEE DOUGLAS	LABC03 - 172.20.115.67	172.20.115.67	dockeradm	docker!2
19	WU WAI TENG VANESSA	LABC03 - 172.20.115.68	172.20.115.68	dockeradm	docker!2
20	YAP KOON SING	LABC03 - 172.20.115.69	172.20.115.69	dockeradm	docker!2
21	YE CHENG LIM	LABC03 - 172.20.115.70	172.20.115.70	dockeradm	docker!2
22	SHAIFUL BIN ABDUL KARIM	LABC03 - 172.20.115.71	172.20.115.71	dockeradm	docker!2
23	CHAI RU YI	LABC03 - 172.20.115.72	172.20.115.72	dockeradm	docker!2
24	JWAY HWEE LING JULIE	LABC03 - 172.20.115.73	172.20.115.73	dockeradm	docker!2
25	SAMANTHA TEO XING YEE	LABC03 - 172.20.115.74	172.20.115.74	dockeradm	docker!2
26	ZIL AZZA HILMIAH BINTE RADUAN	LABC03 - 172.20.115.75	172.20.115.75	dockeradm	docker!2



Remote Desktop Connection

Computer: 172.20.113.xx

User name: None specified

You will be asked for credentials when you connect.

Replace xx with the IP address of the VM that you have been assigned.

Deploy the Application

1. Save below db-pod.yml file in the directory `~/lab09 i.e. /home/dockeradm/lab09`

```
apiVersion: v1
kind: Pod
metadata:
  name: mysql
  labels:
    name: mysql
    app: demo
spec:
  containers:
  - name: mysql
    image: mysql:latest
    ports:
    - containerPort: 3306
      protocol: TCP
    env:
    - name: "MYSQL_ROOT_PASSWORD"
      value: "password"
```

2. Create a POD with single container using the command :

```
cd /home/dockeradm/lab09
kubectl create -f db-pod.yml
```

3. Check the PODs via the below command

```
kubectl get pods
```

4. Login to the Kubernetes Dashboard and analyze the POD

5. Get the complete details of POD using the command

```
kubectl describe pod mysql
```

Part 2: Deploy multiple PODs and communication between PODs in Kubernetes

LAB EXERCISE

This LAB exercise demonstrates the concept of packaging containers into a pod and communication between pods.

Time to Complete
Approximately 30 Minutes

What You Need

- Lab 9 - Part 1 to be completed successfully.
- Execute the command: `sudo apt install docker-compose`
- Kubernetes Cluster should be running. If not start the cluster using the command “minikube start”
- Register a free account on Docker hub https://www.docker.com/

Build a Docker image and push it to docker hub

1. Sign in to the Docker Hub (Browser) https://www.docker.com/
   - Using the same credentials launch a new terminal to login into docker: docker login

2. Git clone all the files from the Folder **PODLab** from the provided git url https://github.com/KeyonGenesis/K8Exercises and edit the below files in the **Docker** folder as per your docker hub username.

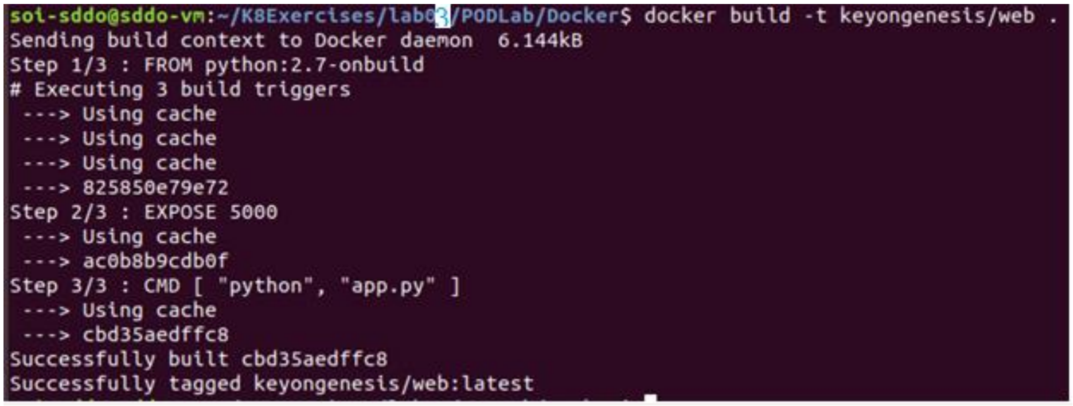
soi-sddo@sddo-vm:~$ *git clone https://github.com/KeyonGenesis/K8Exercises*
soi-sddo@sddo-vm:~$ *cd K8Exercises/lab03/PODLab/Docker*

Edit the following 2 files. Replace the username/repository from keyongenesis/web to your username/ in docker hub.

- Build.sh
- docker-compose.yml

3. From the docker directory, build the image using the below command.

```
docker build -t <DOCKER_HUB_USERNAME>/web .
```



```
soi-sddo@sddo-vm:~/K8Exercises/lab03/PODLab/Docker$ docker build -t keyongenesis/web .
Sending build context to Docker daemon  6.144kB
Step 1/3 : FROM python:2.7-onbuild
# Executing 3 build triggers
 ---> Using cache
 ---> Using cache
 ---> Using cache
 ---> 825850e79e72
Step 2/3 : EXPOSE 5000
 ---> Using cache
 ---> ac0b8b9cdb0f
Step 3/3 : CMD [ "python", "app.py" ]
 ---> Using cache
 ---> cbd35aedffc8
Successfully built cbd35aedffc8
Successfully tagged keyongenesis/web:latest
```

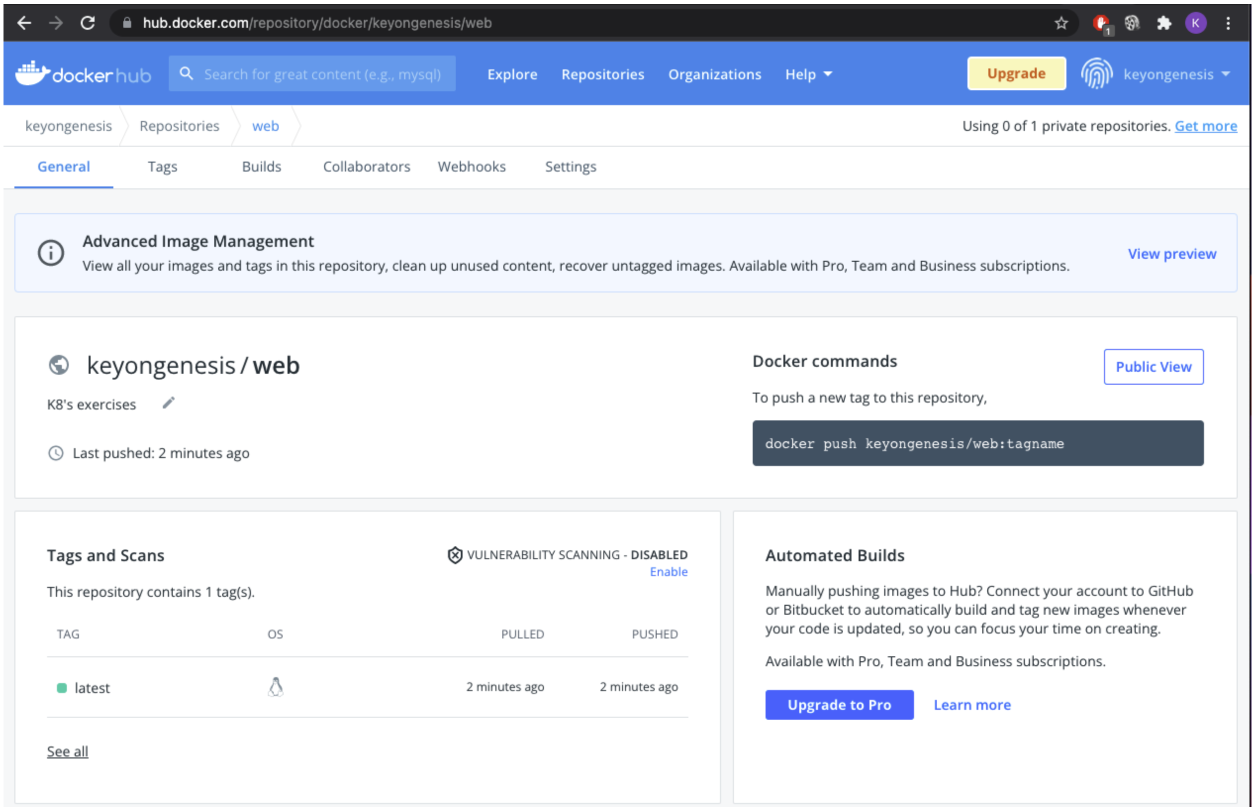
4. Once the build is successful, push the image to your docker hub.

Need to login to your docker hub account first

```
docker login
```

Username:
Password:

```
docker push <DOCKER_HUB_USERNAME>/web
```



Using 0 of 1 private repositories. Get more

Advanced Image Management
View all your images and tags in this repository, clean up unused content, recover untagged images. Available with Pro, Team and Business subscriptions. View preview

keyongenesis / **web**

K8's exercises

Last pushed: 2 minutes ago

Docker commands Public View

To push a new tag to this repository,

```
docker push keyongenesis/web:tagname
```

Tags and Scans

VULNERABILITY SCANNING - **DISABLED**
Enable

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
🟩 latest	🐧	2 minutes ago	2 minutes ago

See all

Automated Builds

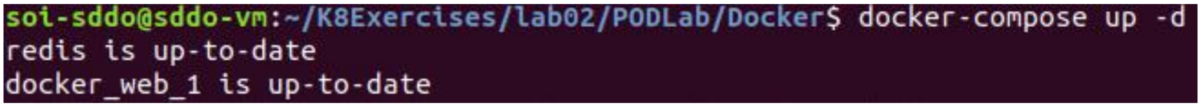
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions.

Upgrade to Pro Learn more

5. To check if the build image is working properly, use the docker compose file to create the container using below command.

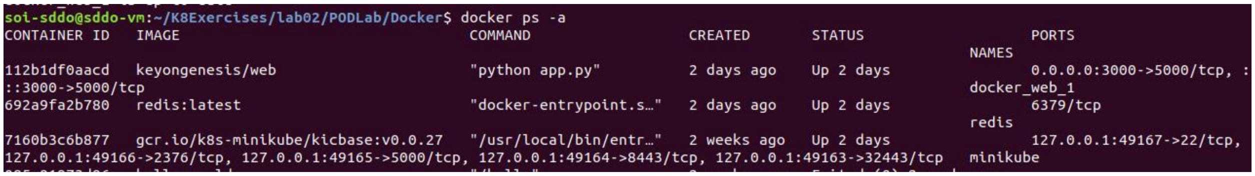
```
docker-compose up -d
```



```
soi-sddo@sddo-vm:~/K8Exercises/lab02/PODLab/Docker$ docker-compose up -d
redis is up-to-date
docker_web_1 is up-to-date
```

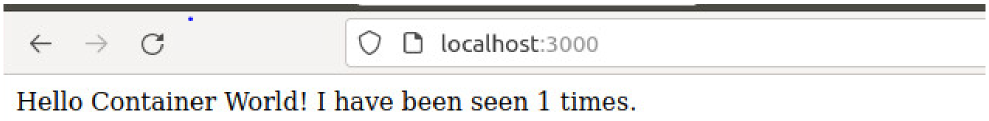
6. Check the container using the below command.

```
docker ps -a
```



```
soi-sddo@sddo-vm:~/K8Exercises/lab02/PODLab/Docker$ docker ps -a
CONTAINER ID   IMAGE                                 COMMAND                  CREATED       STATUS              PORTS
                                                                                                             NAMES
112b1df0aacd   keyongenesis/web                      "python app.py"          2 days ago    Up 2 days                  0.0.0.0:3000->5000/tcp, :
::3000->5000/tcp                                                                                             docker_web_1
692a9fa2b780   redis:latest                          "docker-entrypoint.s…"   2 days ago    Up 2 days                  6379/tcp
                                                                                                             redis
7160b3c6b877   gcr.io/k8s-minikube/kicbase:v0.0.27   "/usr/local/bin/entr…"   2 weeks ago   Up 2 days                  127.0.0.1:49167->22/tcp,
127.0.0.1:49166->2376/tcp, 127.0.0.1:49165->5000/tcp, 127.0.0.1:49164->8443/tcp, 127.0.0.1:49163->32443/tcp   minikube
```

7. Test the application using the url http://localhost:3000/ and the output will be as : Hello Container World! I have been seen 1 times.



Hello Container World! I have been seen 1 times.

Now your build image is ready to deploy in Kubernetes.

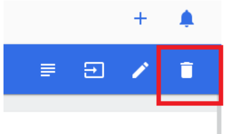
Part 3: Deploy the Application in Kubernetes

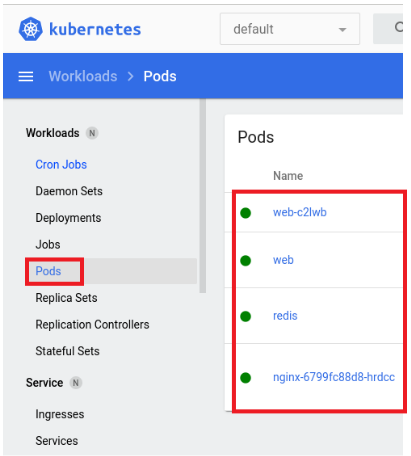
LAB EXERCISE

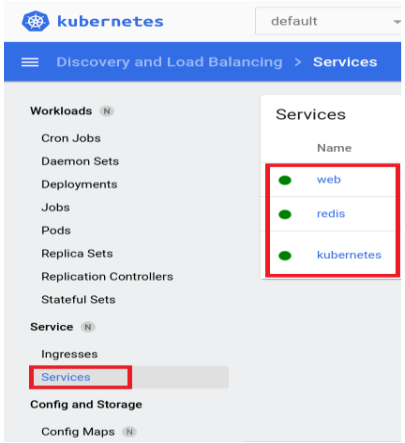
This LAB exercise demonstrates the concept of packaging containers into a pod and communication between pods.

Before starting part 3, you need to clean up the services and pods created in part 2. You may clean them up via `minikube dashboard`

Click on each pod or service, and then click on the bin icon.







Time to Complete

Approximately 30 Minutes

1. Copy all the files from the Folder PODLab from git Location and store it locally. https://github.com/KeyonGenesis/K8Exercises

2. Edit the below files in the folder **Kubernetes** as per your docker hub username.
Edit the following files. Replace the username/repository from keyongenesis/web to your username/web in docker hub.
   - web-pod.yml
   - web-rc.yml

4. Open the command prompt as administrator and create POD service as below

```
cd K8Exercises/lab03/PodLab/Kubernetes/

kubectl create -f db-pod.yml
kubectl create -f db-svc.yml

kubectl create -f web-pod.yml
kubectl create -f web-svc.yml
kubectl create -f web-rc.yml
```

<Insert screen capture of results>

5. Open the command prompt as administrator and create POD service as below

```
kubectl get pods
```

<Insert screen capture of results>

6. Get the list of exposed services.

```
kubectl get svc
```

<Insert screen capture of results>

7. Get service url for web application using below command.

```
minikube service web --url
```

8. Access the url listed above from the browser and verify the result.
Hello Container World! I have been seen 1 times.

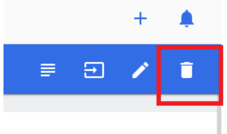
Part 4: Deployment in Kubernetes

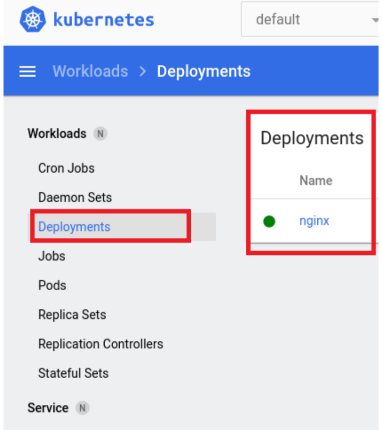
LAB EXERCISE

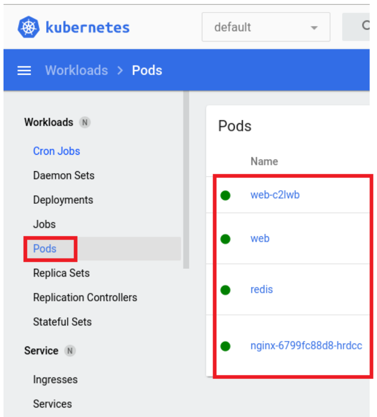
This LAB exercise demonstrates the concept of how to use the service deployment and expose it.

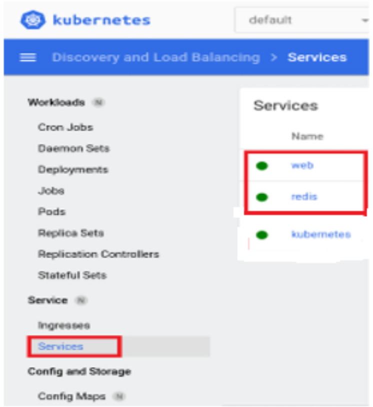
Before starting part 4, you need to clean up the services and pods created in part 3. You may clean them up via `minikube dashboard`

Click on Deployment, Pods and Services, and then click on the bin icon.









Time to Complete

Approximately 30 Minutes

What You Need

- Lab 01 to be completed successfully.
- Kubernetes Cluster should be running. If not start the cluster using the command “minikube start”

1. Deploy the Application

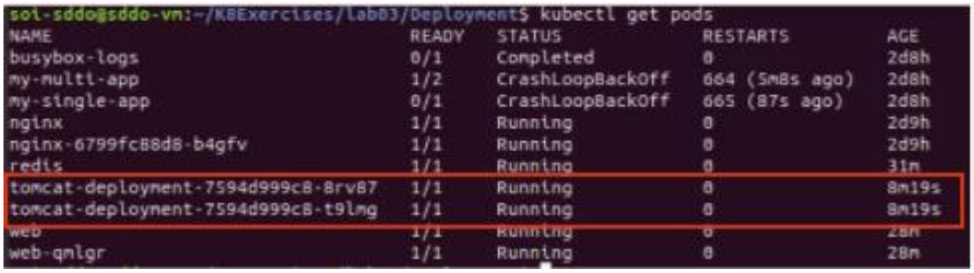
1.1 Save this below *deployment.yaml* file in local storage directory `K8Exercises/lab03/Deployment`

```
apiVersion: apps/v1
```

```
kind: Deployment
metadata:
 name: tomcat-deployment
spec:
  selector:
    matchLabels:
      app: tomcat
   replicas: 2
     template:
       metadata:
         labels:
           app: tomcat
       spec:
         containers:
         - name: tomcat
           image: tomcat:9.0
           ports:
      - containerPort: 8080
```

2. Create a POD service using the command:
   `kubectl create -f deployment.yaml`

3. Check the PODs using the below command:
   `kubectl get pods`



```
soi-sddo@sddo-vm:~/K8Exercises/lab03/Deployment$ kubectl get pods
NAME                                  READY   STATUS             RESTARTS        AGE
busybox-logs                          0/1     Completed          0               2d8h
my-multi-app                          1/2     CrashLoopBackOff   664 (5m8s ago)  2d8h
my-single-app                         0/1     CrashLoopBackOff   665 (87s ago)   2d8h
nginx                                 1/1     Running            0               2d9h
nginx-6799fc88d8-b4gfv                1/1     Running            0               2d9h
redis                                 1/1     Running            0               31m
tomcat-deployment-7594d999c8-8rv87    1/1     Running            0               8m19s
tomcat-deployment-7594d999c8-t9lmg    1/1     Running            0               8m19s
web                                   1/1     Running            0               28m
web-qmlgr                             1/1     Running            0               28m
```

Once the tomcat POD is running, then expose the service.

4. Expose the service using the command:
   `kubectl expose deployment tomcat-deployment --type=NodePort`

4.1. Verify the Service is created and is available on a node port:
   `kubectl get service tomcat-deployment`

5. Get the minikube service url for this tomcat service using the command:
   `minikube service tomcat-deployment --url`

6. Open the browser and access the service URL.

Check the tomcat home page is loading and version.

Upgrade the Application with different tomcat version

7. Update deployment of object tomcat to version 8.0 using the command:

```
kubectl set image deployment/tomcat-deployment tomcat=tomcat:8.0
```

8. Check the PODs are running using the below command:

```
kubectl get pods
```

It will take some time, since the image of the 8.0 version of tomcat needs to be pulled. Once the tomcat POD is running, then expose the service.

9. Expose the service using the command

```
kubectl expose deployment tomcat-deployment --type=NodePort
```

10. Get the minikube service url for this tomcat service using the command:

```
minikube service tomcat-deployment –-url
```

Open the browser and access the service URL. Check the tomcat home page is loading.

Rollout the previous deployed version

11. Rollout the changes applied to the deployment object using the below command.

```
kubectl rollout undo deployment/tomcat-deployment
```

12. Get the minikube service url for this tomcat service.

13. Open the browser and access the service URL.
    Check the tomcat home page is loading and version.

--End of Lab Exercise --

External documentations

Those documentations can help you to go further in this topic :

- Kubernetes official documentation of the command line
- Kubernetes official command line cheat sheet
- Kubernetes official documentation of namespace
- Kubernetes official documentation of labels
- Kubernetes official documentation of recommended labels
- Kubernetes API reference documentation
- Kubectl official Reference documentation
- Kubernetes official documentation on declarative object management
- Kubernetes official documentation on imperative object management