Project Documentation

Project Title: Docker-Project **Completion Date**: 22nd August 2023

Project Summary:

Docker-project to demonstrate skills in docker. In this project, I created a custom Redis image. I pushed that image on Dockerhub to be used anywhere and by anyone. Then I deployed that image on a Kubernetes cluster.

Development Platform: Terraform

Developer: Shubham Thorat
+91 7976438345

Tools Used

1. Docker
2. Dockerhub
3. Minikube

Problem Statement

You are working as a DevOps engineer in an IT firm. You have been asked to create a Redis-based Docker image and deploy it on a Kubernetes cluster.

Solution specification

1. Redis Docker Image Creation:
    - Develop a Dockerfile that outlines the steps to create a Redis-based image.
    - Configure the Dockerfile to include the necessary Redis setup, dependencies, and configurations.
    - Utilize official Redis base images or compatible images for building the custom image.
2. Image Push to Docker Hub:
    - Build the Redis Docker image using the Dockerfile.
    - Tag the image with a meaningful and descriptive name.
    - Log in to Docker Hub using appropriate credentials.
    - Push the tagged image to Docker Hub to make it available for public access.
3. Kubernetes Cluster Deployment:
    - Prepare a Kubernetes Deployment YAML file specifying desired attributes like replica count, and image details.
    - Create a Kubernetes Service YAML file for a NodePort service type to access the redis-cli.
    - Apply the Deployment configuration and Service configuration using kubectl apply to launch the Redis containers.

Implementation

Step1: Created Dockerfile to build the image

```
FROM ubuntu
EXPOSE 6379
RUN apt update && apt install redis-server -y
WORKDIR /root
ENTRYPOINT redis-server --protected-mode no
~
```

Step2: Created Image from the docker file by running following command

$docker build –tag my-redis .

The command will execute instruction from docker file which is in current directory.

–tag my-redis specifies that the name of the image should be my-redis.

We can also provide more information in a tag by a colon(:) like

my-redis:v1

my-redis:v2

my-redis:anything

```
Activities  Terminal                Aug 22 00:39                                  0.00 KB/s  100 %
                                    docker build --tag my-redis .
henry  master

[~/Documents/Study/projects/Simpli/DCA/docker-project]
henry  master  docker build --tag my-redis .
Sending build context to Docker daemon  71.17kB
Step 1/5 : FROM ubuntu
 ---> 01f29b872827
Step 2/5 : EXPOSE 6379
 ---> Running in 7f84c1a25cae
Removing intermediate container 7f84c1a25cae
 ---> 44acc1d4f372
Step 3/5 : RUN apt update && apt install redis-server -y
 ---> Running in f46039c7c710

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [44.0 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [980 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [164 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 Packages [1792 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [17.5 MB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [860 kB]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [897 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [266 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1136 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1241 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [49.8 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [923 kB]
Get:17 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [25.6 kB]
Get:18 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [49.2 kB]
Fetched 26.5 MB in 9s (3069 kB/s)
Reading package lists...
Building dependency tree...
Reading state information...
2 packages can be upgraded. Run 'apt list --upgradable' to see them.

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists...
Building dependency tree
```

Step3: I ran the container from a my-redis image

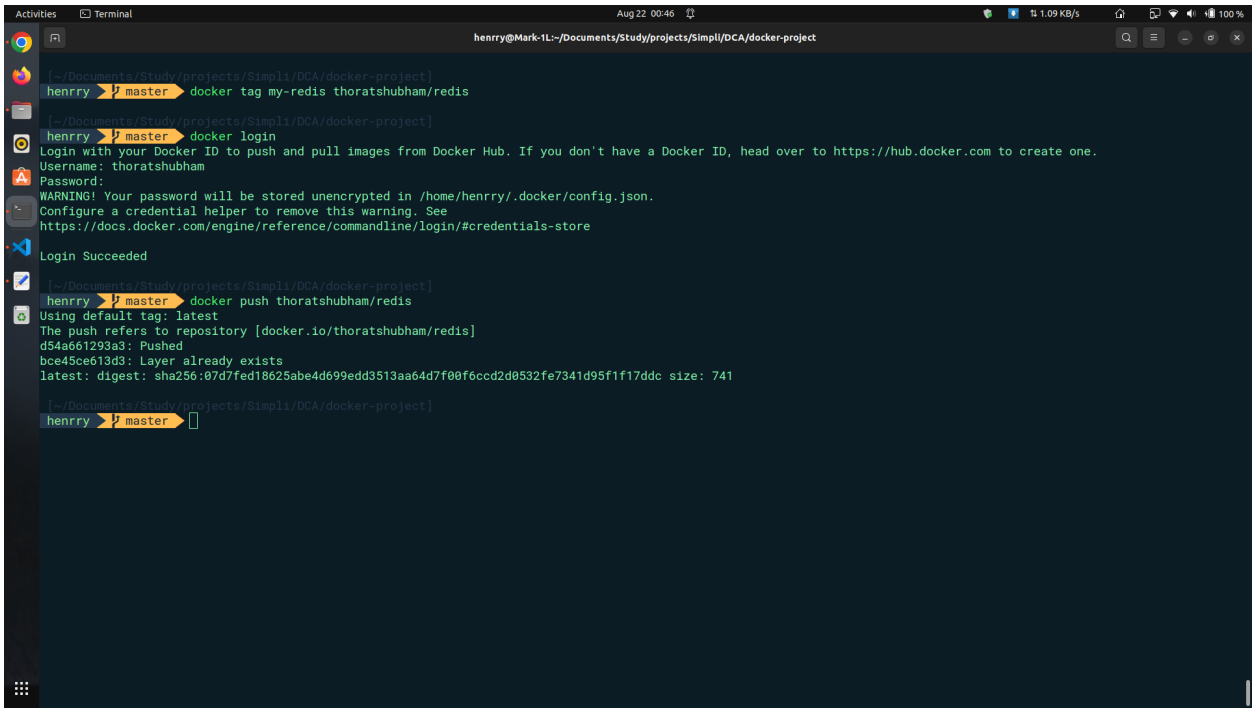
Step4: Tested if i can access with redis-cli.

```
Activities  Terminal                Aug 22 00:43                                  1.10 KB/s  100 %
                                    henry@Mark-1L:~/Documents/Study/projects/Simpli/DCA/docker-project

[~/Documents/Study/projects/Simpli/DCA/docker-project]
henry  master  docker run -d -p 6379:6379 my-redis
640e81307b1eb90f6522c4a5ca9130affecee899034456dfa20da6c642bc95ab

[~/Documents/Study/projects/Simpli/DCA/docker-project]
henry  master  docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED             STATUS              PORTS
                                                                                                                          NAMES
640e81307b1e   my-redis                             "/bin/sh -c 'redis-s…"   5 seconds ago       Up 3 seconds        0.0.0.0:6379->6379/tcp, :::6379->6379/tcp
                                                                                                                          happy_lederberg
18ace0f7cb2b   gcr.io/k8s-minikube/kicbase:v0.0.40   "/usr/local/bin/entr…"   About an hour ago   Up About an hour    127.0.0.1:32772->22/tcp, 127.0.0.1:32771->2376/tcp,
 127.0.0.1:32770->5000/tcp, 127.0.0.1:32769->8443/tcp, 127.0.0.1:32768->32443/tcp   minikube

[~/Documents/Study/projects/Simpli/DCA/docker-project]
henry  master  redis-cli -h localhost -p 6379
localhost:6379> set name shubham
OK
localhost:6379> get name
"shubham"
localhost:6379> set status "It Works!!!"
OK
localhost:6379> get status
"It Works!!!"
localhost:6379> exit

[~/Documents/Study/projects/Simpli/DCA/docker-project]
henry  master
```

Step5:changed the tag of the image from my-redis to thoratshubham/redis

To push the image on dockerhub

Step6: Logged in to my docker hub account through “docker login” command and pushed the image



Step7: created kubernetes cluster from minikube.

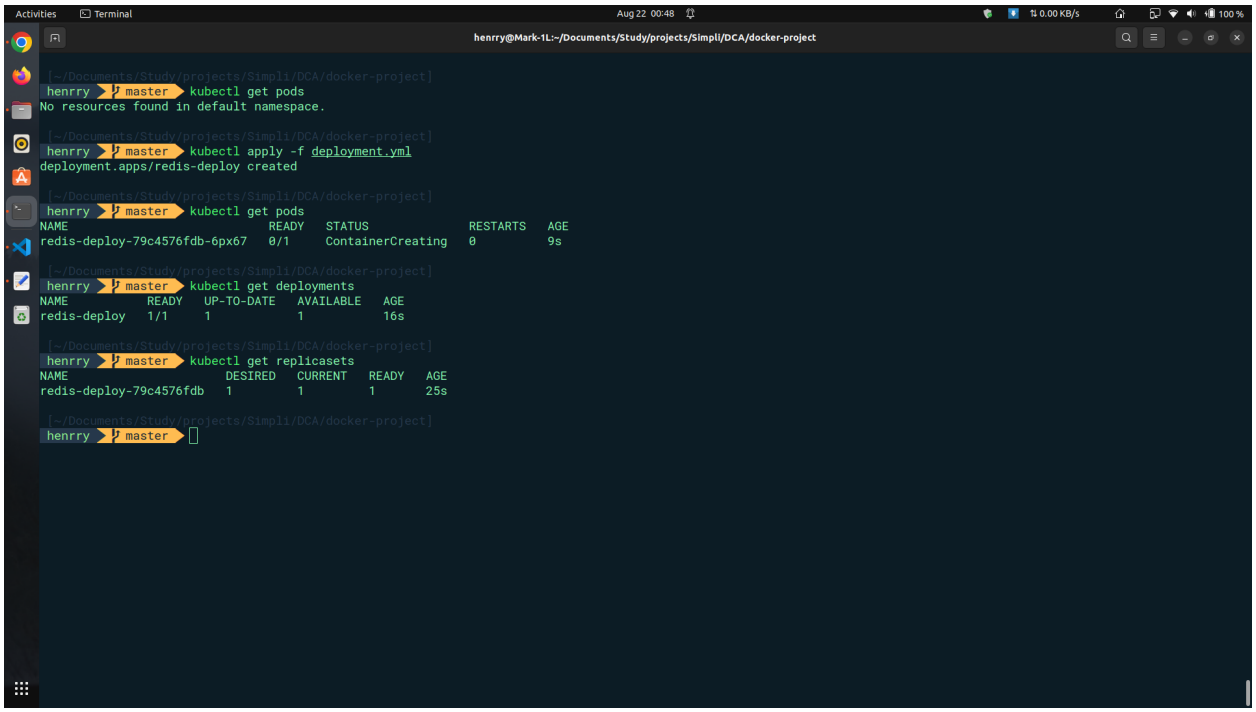
Step8: created deployment to lauch the redis image inside pod.

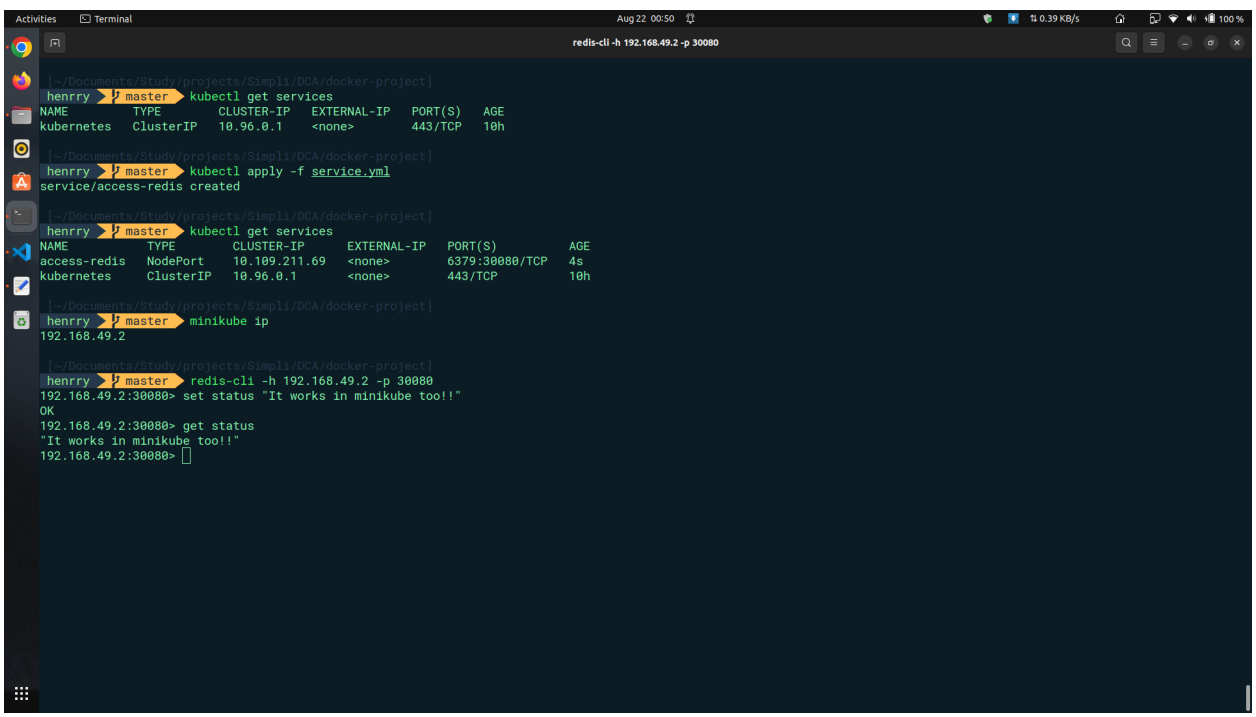
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
  template:
    metadata:
      labels:
        app: redis
    spec:
      containers:
      - name: test-redis
        image: thoratshubham/redis
        ports:
          - containerPort: 6379
~
```

Step9: created service to access the redis image from outside

```yaml
apiVersion: v1
kind: Service
metadata:
  name: access-redis
spec:
  type: NodePort
  selector:
    app: redis
  ports:
    - protocol: TCP
      port: 6379
      targetPort: 6379
      nodePort: 30080
~
```

Step10: using kubectl a cli tool created a deployment for redis image and NodePort service to access the redis from outside.





As you can see in the screenshot it works.