**EXPERIMENT-8&9:**

8.Integrate Kubernetes and Docker

9.Automate the process of running containerized application for exercise 7 using kubernetes

* Login to aws account
* Create a t2 medium instance
* Follow the commands
* Connect to your instance

\* sudo apt update

\* sudo apt upgrade -y

\* sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software- properties-common

\* curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

\*sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"

\* sudo apt-get update

\* sudo apt-get install docker-ce docker-ce-cli containerd.io

\* docker --version

\* sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

\* sudo chmod +x /usr/local/bin/docker-compose

\* docker-compose --version

\* curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

\* sudo install minikube-linux-amd64 /usr/local/bin/minikube

\* git clone https://github.com/Iliyaz-Syed/k8s-main

\* cd k8s-main/

\* minikube start

\* sudo su

\* sudo minikube start --force

\* sudo docker ps

\* eval $(sudo minikube docker-env)

\* docker build -t flaskimage:latest .

\* kubectl apply -f flask-app-deployment.yaml

\* snap install kubectl

\* minikube kubectl --

\* minikube kubectl --ssh

\* kubectl options

\* curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"

\* chmod +x ./kubectl

\* mv ./kubectl /usr/local/bin/kubectl

\* kubectl options

\* kubectl apply -f flask-app-deployment.yaml

\* kubectl get services

\* minikube tunnel --cleanup

\* kubectl get po

\* kubectl apply -f flask-app-deployment.yaml

\* minikube service flask-app-service

\* kubectl get services

\* eval $(minikube docker-env)

\* kubectl apply -f flask-app-deployment.yaml

\* kubectl get po

\* minikube logs

\* docker images

\* cat docker-compose.yml

\* vim flask-app-deployment.yaml

-set imagePullPolicy to Never from IfNotPresent

\* eval $(minikube docker-env)

\* docker build -t flaskimage:latest .

\* kubectl rollout restart deployment flask-app

\* kubectl get pods

\* kubectl apply -f flask-app-deployment.yaml

\* kubectl get po

\* docker images

\* kubectl delete pod --selector=app=flask-app

\* kubectl delete pod --selector=app=flask-app

\* kubectl get pods

\* minikube logs

\* eval $(minikube docker-env)

\* docker build -t flaskimage:latest .

\* vim flask-app-deployment.yaml

-set type to NodePort from LoadBalancer

\* docker images

\* kubectl delete pods --selector=app=flask-app

\* kubectl delete pods --selector=app=flask-app

\* kubectl delete pods --selector=app=flask-app

\* kubectl get po

\* kubectl get svc

\* minikube tunnel --cleanup

\* kubectl delete pods --selector=app=flask-app

\* kubectl get po

\* kubectl apply -f flask-app-deployment.yaml

\* kubectl get po

\* minikube logs

\* kubectl get po

\* kubectl get svc

\* apt install gnome-terminal

\* minikube tunnel --cleanup

\* sudo ufw allow 6006/tcp

\* minikube tunnel --cleanup

\* sudo ufw allow 6006/tcp

\* minikube tunnel --cleanup

\* kubectl get svc

\* vim flask-app-deployment.yaml

\* kubectl apply -f flask-app-deployment.yaml

\* kubectl get po

\* minikube tunnel --cleanup

\* vim flask-app-deployment.yaml

- set type to LoadBalancer from NodePort

\* minikube tunnel --cleanup

\* minikube service list

\* curl <url>

\* minikube start --driver=none

\* minikube tunnel

\* kubectl edit svc flask-app-service

\* minikube tunnel

* **Open gitbash and connect to the same instance again and follow the commands without logging into root**.

\* sudo kubectl get svc

\* curl <url>

\* sudo apt update

\* sudo apt install nginx

\* sudo vim /etc/nginx/sites-available/default

server {

listen 80;

server\_name <your instance publicip>;

location / {

proxy\_pass <url>;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwardeed-For $proxy\_add\_x\_forwarded\_for;

}

}

\* sudo system restart nginx

* **In new tab enter your instance public ip**

