Assignment 1

1. Create 3 instances: Ubuntu 20, VM.Standard.E4.Flex 4CPU 8GB ram
2. VCN – port – 80, 8080, 5000, 6443, 10250
3. sudo apt-get install -y apt-transport-https ca-certificates curl gnupg2 software-properties-common
4. Add Firewall
   * sudo apt install firewalld
   * sudo systemctl enable firewalld
   * sudo firewall-cmd --permanent --zone=public --add-port=6443/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=5000/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=80/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=10250/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=8080/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=6783/tcp
   * sudo firewall-cmd --permanent --zone=public --add-port=6783/udp
   * sudo firewall-cmd --permanent --zone=public --add-port=6784/udp
   * sudo firewall-cmd --reload
   * sudo firewall-cmd –state
5. Install Docker
   * sudo apt-get install -y apt-transport-https ca-certificates curl gnupg2 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 $(lsb\_release -cs) stable"
   * sudo apt-get update
   * sudo apt-get install -y docker.io
   * sudo usermod -aG docker ${USER}
   * Restart the terminal to give user docker access
6. Install python requirements file (for testing on local machine)
   * sudo apt install python3-pip
   * pip install flask numpy opencv-python-headless
7. Running the server python
   * python3 objectDetectionServer.py
8. Running the client
   * python3 iWebLens\_client.py inputfolder/ [http://168.138.8.154/api/object\_detection 4](http://168.138.8.154/api/object_detection%204)
9. Building the docker image
   * docker build -t asgdocker .
   * docker run --rm asgdocker:latest
10. Installing kubernetes and setting up master-node
    * sudo systemctl enable docker
    * sudo systemctl status docker
    * sudo systemctl daemon-reload
    * sudo systemctl restart docker
    * curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
    * sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
    * sudo apt-get install kubeadm kubelet kubectl
    * sudo apt-mark hold kubeadm kubelet kubectl
    * kubeadm version
    * sudo vi /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

* + sudo systemctl daemon-reload
  + sudo systemctl restart docker
  + sudo systemctl restart kubelet
  + sudo hostnamectl set-hostname master
  + sudo swapoff -a
  + sudo sed -i '/ swap / s/^/#/' /etc/fstab

1. Initializing the kubernetes cluster (On Master)
   * sudo kubeadm init --pod-network-cidr=10.244.0.0/16
   * mkdir -p $HOME/.kube
   * sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
   * sudo chown $(id -u):$(id -g) $HOME/.kube/config
2. Add the weave network manager (On Master)
   * kubectl apply -f [https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')](https://cloud.weave.works/k8s/net?k8s-version=$(kubectl%20version%20|%20base64%20|%20tr%20-d%20'\n'))
3. Joining the worker nodes to the cluster
   * sudo kubeadm join 10.0.0.139:6443 --token 3nu34s.wdxgnvfcza5nxr8n --discovery-token-ca-cert-hash sha256:42a31def4a35774b35c5f954bacfac578ddf6454dfb331cecaba84ec1b7a3fcf
4. Check if worker nodes have joined the cluster (Run on master)
   * kubectl get nodes
5. Save the docker to the local system
   * docker save asgdocker > asgdocker.tar
6. Use WinSCP to move the .tar images to the worker nodes
7. Deploy the docker apps on your worker nodes and map the ports (on worker nodes)
   * docker load -i asgdocker.tar
   * docker run -p 5000:80 -d asgdocker
8. Add the node port (30125) on all nodes
   * sudo firewall-cmd --permanent --zone=public --add-port=30125/tcp
   * sudo firewall-cmd --reload
   * sudo firewall-cmd --list-all
9. Create the deployment and service yaml files
   * my-deployment.yml:

labels:

app: object-dectection

spec:

replicas: 1

selector:

matchLabels:

app: object-dectection

template:

metadata:

labels:

app: object-dectection

spec:

containers:

- image: asgdocker:latest

name: my-deployment

ports:

- containerPort: 5000

resources:

limits:

memory: "512Mi"

cpu: "0.5"

requests:

memory: "512Mi"

* + object-detection-service.yml:

apiVersion: v1

kind: Service

metadata:

name: object-detection

spec:

type: NodePort

selector:

app: object-dectection

ports:

- protocol: TCP

port: 80

targetPort: 5000

nodePort: 30125

1. Run the deployment and service
   * kubectl create -f my-deployment.yml
   * kubectl create -f my-service.yml
2. Check the deployment, service and pods
   * kubectl get service
   * kubectl get deployment
   * kubectl get po
3. Now, once the service is running, run the client command
   * python3 client/iWebLens\_client.py client/inputfolder/ http://168.138.8.154:30125/api/object\_detection 4