**ML Kubernetes Deployment Pipeline**

**Task 1: Build an ML webapp and Dockerize**

1. Start a new VM in GCP
2. Install docker using following commands:

sudo apt-get update

sudo apt-get install docker.io

gcloud auth configure-docker

1. Configuration

sudo -s

gcloud auth login

gcloud config set project fb-mlops-nov-21

gcloud config set compute/zone us-west1-a

export PROJECT\_ID=fb-mlops-nov-21

echo $PROJECT\_ID

1. Run the following commands to download the code:

sudo -s

cd $HOME

git clone -b deploy\_branch https://github.com/dataninja01/Capstone-Project

cd Capstone-Project

1. Open server.py and make sure the last line of the code is :

if "serve" in sys.argv: app.run(host='0.0.0.0', port=8000, debug=False)

This code indicates that the application is exposed to port 8000. You can change the port to whichever port you would like.

1. Make sure you have the correct Dockerfile. You can read the file by entering the command “cat Dockerfile”. If there is no Dockerfile, you can make one like this:

nano Dockerfile

Copy and paste the following text into the Dockerfile. You can also change the text to fit your needs:

FROM python:3.8

RUN apt-get update

RUN cd $HOME

RUN git clone https://github.com/dataninja01/Capstone-Project

RUN cd Capstone-Project

COPY . /app

WORKDIR /app

RUN pip3 install -r requirements.txt

RUN python3 server.py

EXPOSE 8000

CMD ["python3", "server.py", "serve"]

Exit editing by entering “Ctrl+X” and “Y”.

1. Build the docker container

docker build -t gcr.io/${PROJECT\_ID}/capstone:v1 .

docker images

1. Make sure it is running successfully:

docker run --rm -p 8000:8000 gcr.io/${PROJECT\_ID}/capstone:v1

1. Open a browser with http://<External IP address>:8000

Use your deployed app

**Task 2: Push the web app to Kubernetes**

1. Follow the following commands to install kubectl and for credentials updation to push docker to Kubernetes.

Install kubectl: follow the steps at <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>

kubectl version --client

gcloud services enable container.googleapis.com

1. Make sure that you are in the correct folder “Capstone-Project”
2. Run the following commands to push the docker container. Remember you are now running as a super user (no need for sudo).

docker push gcr.io/${PROJECT\_ID}/capstone:v1

gcloud container images list

1. Install kubectl and create a Kubernetes cluster with 3 nodes

gcloud config set compute/zone us-west1-a

gcloud services enable container.googleapis.com

gcloud container clusters create credit-card-cluster

gcloud container clusters get-credentials credit-card-cluster

gcloud compute instances list

1. Deploy the docker container to kubernetes

kubectl create deployment capstone --image=gcr.io/${PROJECT\_ID}/capstone:v1

kubectl expose deployment capstone --type=LoadBalancer --port 8000

kubectl get service

1. Run kubectl get service to check the external IP for the “capstone” deployment. It might take a while for the “pending” status at the “external IP” column to disappear. Once the external IP was assigned, open a new web browser and enter http://<ExternalIP>:8000 to access our web application.
2. Commonly encounter problem:

If a web application was successfully deployed on Docker but not on Kubernetes, it could be because there was an error when deploying with Kubernetes. There are four useful commands to troubleshoot Pods (https://learnk8s.io/troubleshooting-deployments):

1. kubectl get deployment is useful to check if the application iss actually deployed. If ‘ready’ is 0/number, it means it is not successfully deployed.
2. kubectl logs <pod name> is helpful to retrieve the logs of the containers of the Pod. You can find pod name by running kubectl get pods
3. kubectl describe pod <pod name> is useful to retrieve a list of events associated with the Pod.
4. kubectl get pod <pod name> is useful to extract the YAML definition of the Pod as stored in Kubernetes.
5. kubectl exec -ti <pod name> -- bash is useful to run an interactive command within one of the containers of the Pod.
6. Sometimes it could be because there is a version conflict between the version we deployed on docker and version we deploy on the cluster. Make sure the deployed version has debug turned off.

**Task 4: Kubernetes Monitoring using Prometheus and Grafana**

For this task, we begin from the Kubernetes cluster with the exposed deployment (od-function). We will use helm to install helm charts and then view Kubernetes pod health in Prometheus and Grafana Dashboards.

The blogs we refer to as a valuable resources here are:

[1] <https://devops4solutions.com/monitor-kubernetes-cluster-using-prometheus-and-grafana/>

[2] <https://medium.com/htc-research-engineering-blog/monitoring-kubernetes-clusters-with-grafana-e2a413febefd>

1. Install helm using the following commands:

gcloud container clusters get-credentials credit-card-cluster

apt-get install wget

wget https://get.helm.sh/helm-v3.1.1-linux-amd64.tar.gz

tar zxfv helm-v3.1.1-linux-amd64.tar.gz

cp linux-amd64/helm .

1. Authenticate google cloud and install helm charts.

gcloud auth login

./helm repo add stable <https://charts.helm.sh/stable>

./helm repo add prometheus-community

./helm repo update

./helm install prometheus prometheus-community/kube-prometheus-stack

1. Get the names of pods and services running on the cluster:

kubectl get pods

kubectl get svc

1. Forward prometheus services to port 7000

kubectl port-forward --address 0.0.0.0 prometheus-prometheus-kube-prometheus-prometheus-0 7000:9090

1. Forward Grafana services to port 5000

New VM windows

gcloud auth login

gcloud config set project ${PROJECT\_ID}

gcloud container clusters get-credentials credit-card-cluster

kubectl port-forward --address 0.0.0.0 kube-prometheus-stack-1606233825-grafana-598d4d4bd6-r7pp5 5000:3000

Username: admin

Password: prom-operator

1. Add sources to Grafana (Prometheus and Kubernetes monitoring) and dashboard.

**Task 5: Clean Up**

kubectl delete --all svc

kubectl delete --all deployment

kubectl delete --all pods

docker rmi (image ID)

gcloud container clusters delete credit-card-cluster

Delete VM, VPN, firewall, and other resources we used like Cloud SQL