SOFE 4630U
Cloud Computing

Project Milestone #1
2022/02/03

Iaas: Virtualization and Containerization

Sabesan Sivakumar(100701928)

What are docker image, container, and registry?

Docker Image: Is essentially a template filled with instructions on how to build a docker container.This can include the code to which it wants to run (file location), The JDK version of the base image, creation and setting the directory for application files to be executed from the app.

Docker Container: Is what allows you to package your application and all of its dependencies in an easy-to-share way. It allows users to run the application with minimal work as long as everything is packaged in the container.

Docker Registry: is the server side application which allows the user to store and distribute docker images.

List the Docker commands used in the video with a brief description for each command and option.

docker build -t hello-world:1.0 . : build command builds the container -t used to specify the image and tag in this case its hello-world with the tag of 1.0 and the . used is to specify the directory since we were working in the directory the . means current directory

docker images : Displays all images on the device with the image ID when it was created and the size of the image.

docker run hello-world:1.0 : runs the container in this case hello-world:1.0

docker ps:List all the running containers

docker logs: retrieves all the logs present

docker images: retrieves all the images alongside their id and time of when they were created.

At the end of the video, there are two running containers, what commands can be used to stop and delete those two containers?

- docker stop containerID: Stops the container
- docker rm -f containerID: Deletes the container
- Only the first 4 digits of the ID are needed

Prepare a video showing the container(s) created on your machine, displaying their logs, stopping them, and then deleting them. (Note: the JDK version must match that installed in

your machine and used to compile the java code. If you have a problem compiled it can download it from the repository from the path: "/v1/out/production/HelloWorldDocker/Main.class" and use OpenJDK:14 in your Dockerfile).

https://drive.google.com/file/d/1Lkp3_hWQCsP_wJS6IRIpr6ZlA_vJ-kkf/view?usp=sharing

What's a multi-container Docker application?

- Contains multiple containers running and communicating with each other on a docker application but have to be connected to the same network.

How are these containers communicated together?

- All containers can communicate with each other as long as they are in the same network
- In our code the containers mysql and my-web-ap are connected to the same network: app-network
- Have their own ip address so they can communicate with each other

What command can be used to stop the Docker application and delete its images?

- docker stop containerID: Used to stop the docker application also not the entire containerID is not need just the first 4 letters
- docker rm imageID -Used to delete images based of their id

List the new docker commands used in the video with a brief description for each command and option.

- docker pull mysql: pull mysql image from online
- docker build -t my-web-app:1.0 :create a image called my-web-app with a tag of 1.0
- docker run –name app -d -p 8080:8080 –network=app-network mywebapp:1.0 :runs the application on port 8080
- docker network create app-network: to create the virtual network where we can have multiple containers on the same network

Prepare a video showing the created application, run the webapp, stop the application and delete the application containers. (Note: if you have a problem generating the war file, you can download it from the repository from the path: "/v2/target/MyWebApp.war").

https://drive.google.com/file/d/1UPhVvpgq_BdQzTXvHfXIVDTfRDS9e1dc/view?usp=sharing

Prepare a video showing how the container is deployed using Dokcer and Kubernetes in GCP

https://drive.google.com/file/d/1dH1xqVokh-Px4vDxTE4PlGj3kRsKP8wf/view?usp=sharing

List all used GCP shell commands and their description in your report.

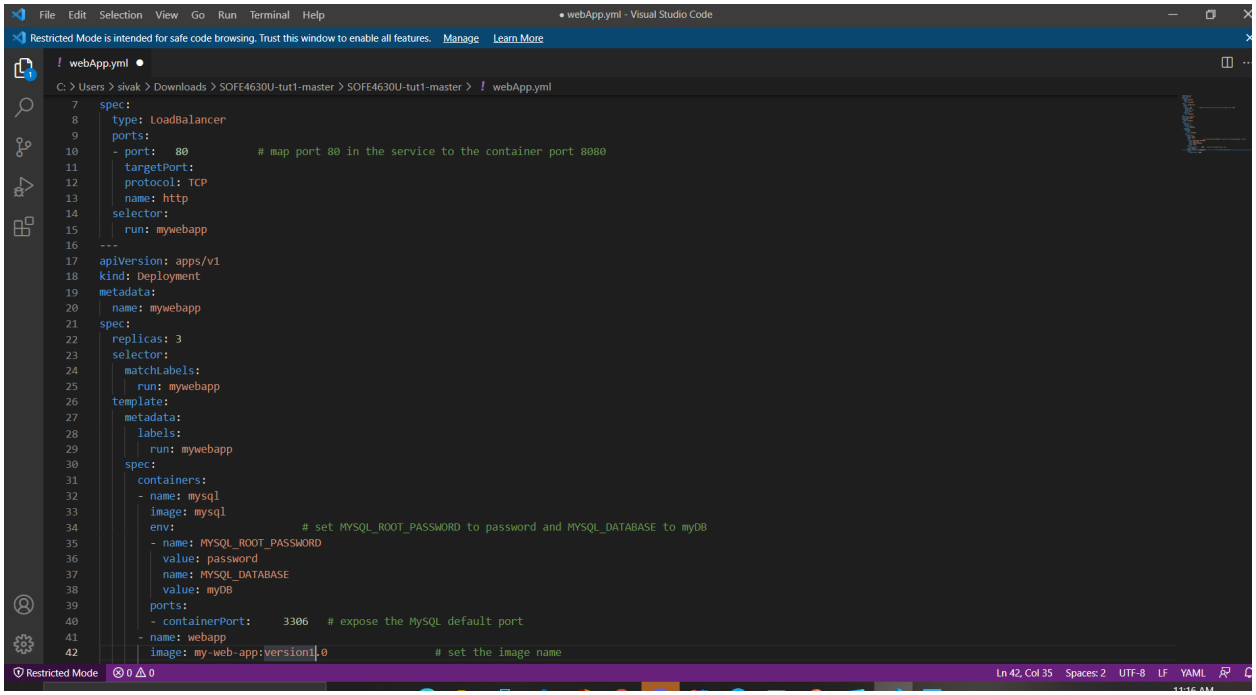
- gcloud config set project projectID : this sets current working directory
- gcloud config set compute/zone us-central1-a: Sets current working directory time zone
- gcloud container clusters create gk-cluster –num-nodes=1: creates a cluster on google cloud with 1 node
- gcloud container clusters get-credentials gk-cluster: allows the command
- kubectl create deployment web-server –image=us.gcr.io/projectID/cad-site:version 1: deploying the application to the server
- kubectl expose deployment web-server –type LoadBalancer –port 80 –target-port 80: exposes the application to port 80 and enables the website to be accessed
- kubectl get pods: Displays all the current pods
- kubectl get service web-server: retrieve the name type cluster-ip external ip ports and age of all the current services

Prepare a Kubernetes YML (or YAML) file to load the webApp used in steps 6:8 and deploy it using the Kubernetes engine on GCP. The file is a little different than that used by docker-compose.

- The hostname of all containers is the same and can be accessed by localhost, the address of the MySQL should be changed to localhost and recompiled. (Note: if you have a problem generating the war file, you can download it from the repository from the path "/KGS/target/MyWebApp.war").
- Create a new image using the new war file and push it to Google Container Registry.
- Follow the comments and fill the missing lines in the "/webApp.yml" file.
- Apply the YML file into Kubernetes and run the server (what is the appropriate Cloud shell command?).

Prepare another video describing the YML file and showing how it's deployed on GCP.

Was not able to get this step to work as I was having troubles to tag and push to the container registry. Shown below is a screenshot of the YML file



What is Kubernetes' pod, service, node, and deployment?

Pod: Contains a group of containers and each container will be able to share the same resources and network as long as they're in the same pod.

Service: It gives an interface access to the pods it holds. This can allow them to have access to their network or external processes and service

Node: The pods that we mentioned earlier are what runs on a node. Each node has its own control panel that has all the requirements to run each pod. It can by virtual or physical depending on the cluster

Deployment: Is what updates the Pods to ensure that all the pods are loosely coupled and that if one pod were to fail the application can still run efficiently since they are replicas in multiple pods.

What's meant by replicas?

Replicas are needed so that if a pod fails or the application gets busy with replicas we can run multiple pods with the replicas of containers so they can take over and the application can still run smoothly.

What are the types of Kubernetes' services? What is the purpose of each?

Cluster IP: Is the default service, where everything happens internally within the cluster.

Node Port: A service through each node port where the kubernetes can send and receive data by the port number. This service is mainly beneficial in development as the kubernetes will still be able to send data even if the service is not running.

ExternalName: A service where it is connected to each other via DNS name and not its ip address which is more commonly used.

Load Balancer: This service is automatically created in kubernetes and it fills the service with the external IP address that the cloud has already created its routes.