Tolu Elebute
100724471

Questions

Part 1

1. What are docker image, container, and registry?

   Docker image is used for creating a container that can be run on docker platform. It is a read-only template that contains a set of instructions for doing this.

   A Container needs to run an image to exist. They are deployed instances created from templates by image.

   A registry is a server-side application that stores and allows the distribution of docker images.

2. List the Docker commands used in the video with a brief description for each command and option.
   - FROM: Sets the bae image and initializes a new build stage
   - RUN: Create a new app directory for your application files
   - COPY: Copies the app files from your host machine to image file system
   - WORKDIR: Sets the directory for executing future commands
   - CMD: Runs the main class when your container starts

3. 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 <container-ID>
   - $ docker rm -f <container-ID>

   Video 1: https://vimeo.com/672595875/8f85bb8158

Part 2

1. What's a multi-container Docker application? A multi-container docker application allows for the running of components in their own container that docker plugs all together using standard network protocols.
2. How these containers are communicated together? For containers to communicate, they need to be of the same network.
3. What command can be used to stop the Docker application and delete its images?
   - docker image rm
   - docker stop <OPTIONS> container <container...>
4. List the new docker commands used in the video with a brief description for each command and option.
   - docker pull mysql: pulls official mysql image
   - docker run --name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql: creates a container app-db in detached

mode with environment variable MYSQL_ROOT_PASSWORD and MYSQL_DATABASE on the mysql image.
- docker run --name app -d -p 8080:8080 my-web-app:1.0: creates container in detached mode on 8080 port on my-web-app:1.0 image
- docker network create app-network: creating own network
- docker network connect app-network app-db: This connects the app container with db container on the same network
- docker compose: This automatically creates a bridge network and attaches containers to it.

Video 2: https://vimeo.com/672618803/0607d3b291

Part 3

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

   Video 3.1: https://vimeo.com/672945233/a9c3502685

   Video 3.2: https://vimeo.com/672977246/50fda8f1ab

   GCP Commands

   - docker push us.gcr.io/zippy-purpose-340120/cad-site:version1: pushes to google cloud registry (kinda like a repo)
   - gcloud container clusters create gk-cluster --num-nodes=1: creates a GKE cluster
   - kubectl create deployment web-server --image=us.gcr.io/zippy-purpose-340120/cad-site:version1: deploys an application to the cluster
   - kubectl expose deployment web-server --type LoadBalancer --port 80 --target-port 80: To expose your application while creating a compute engine load balancer for your container and initializeing port 80 public to the internet.
   - kubectl get pods: inspects running pods
   - kubectl get service web-server: inspects web-server service

Part 4

1. What is Kubernetes' pod, service, node, and deployment?
   - Node runs the services, they are a set of worker machines that host the pods
   - Pods are the most smallest and most basic deployable objects. Containers are placed into pods to run on nodes.
   - A kubernetes service enables assignment of name and an IP address to a group of pods in a cluster that perform the same function.
   - Forthe pods that hold an application that has been containerized, deployment tells kubernetes how to create or modify instances of them.

2. What’s meant by replicas? These are processes that maintain the specified number of pod instances running in a cluster constantly so that in scenarios here a pod fails or is inaccessible, users don’t lose access to their applications.
3. What are the types of Kubernetes’ services? what is the purpose of each?

   Types of kuberenetes type:

   - ClusterIP: A given cluster-internal ip address to the service makes it only reachable within the cluster.
   - NodePort: This service exposes the service outside of the cluster and does this by adding a cluster-wide port on top of ClusterIP. It can be perceived as an extension of the ClusterIP service.
   - LoadBalanacer: In extension to the NodePort service, LoadBalancer integrates NodePort with cloud-based load balancers.
   - ExternalName: ExternalName service maps a service to a DNS name.

   Video 4: https://vimeo.com/673231101/a9c5929b62