Cloud Computing Project Milestone 1

SHAYAN SEPASDAR

100722542

PART 1

Answer the following questions:

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

   Docker image: A docker image is a template that's used to create and run a container.

   Container: A standard component that allows you to package your application and it's dependencies in a easy way to share

   Registry: the location that containers can be stored.

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

   docker build -t helloworld:1.0 .

   It builds a container called hello world with version 1

   docker ps / docker ps -a

   Shows all the available containers , -a for the ones running in background

   docker images

   Shows all created images

   docker run helloworld:1.0 / docker run -d helloworld:2.0

Runs the container, -d for running in background

docker logs containerID

Displays the log for the give container id

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

Stop: docker stop <container id>

Delete: docker rm <container id>

VIDEO 1

https://drive.google.com/file/d/1o8khjdJceu8PVsiJ_Igu0IN7Nul1UGrP/view?usp=sharing

PART 2

1.What's a multi-container Docker application?How are these containers communicated together?

A multiple container docker application is an application that connect multiple containers for a purpose.

Container communicate through use of a bridge network

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

Docker-compose stop

Docker-compose rm

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

The following list contains the commands used within Video 2 and their respective explanations:

`docker pull mysql`
Retrieves MySQL image from Docker Hub.
`docker run --name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql`
Runs a MySQL container from the officially pulled image.
The --name option to give the container a name.

-d to run the container in detached mode.
The -e option specifies that it will be followed by a configuration environment variable. Then MySQL root password and database name are set.

`mvn install`

Maven command used to download dependencies, build and create a .war file which is used within the container image.

`docker run --name app -d -p 8080:8080 my-web-app:1.0`

Start running a container from the "my-web-app" image, version 1.0.
Creates the name label "app".
Binds the container port 8080 to the host port 8080 to allow communication.

`docker network create app-network`

Create a dedicated bridge network between two containers and specify its name as "app-network"

`docker network ls`

Displays all the running networks within Docker.

VIDEO 2
https://drive.google.com/file/d/1mTvoJA8lS0p_m4t0gsdjRUQR9UlkRiKn/view?usp=sharing

PART 3

The following list contains the commands used within video 3 and their respective explanations:

`gcloud config set project project_name`

Set the current project in the cloud shell to be a newly created project for your website cloud computing milestone

`gcloud config set compute/zone us-central1-a`

Set a default compute zone for GK cluster to enable the kubectl command and related GCP services

`docker run -d -p 8080:80 nginx:latest`

Run the nginx server (in detached version using -d) by exposing port 8080 of the cloud shell port 80 for the nginx web server using the following command

`docker cp index.html container_id:/usr/share/nginx/html`

Copy the index.html file onto the nginx server file system

`docker commit container_id image_name:tag`

Commit the newly added file changes to a newly created docker image using the following command

```
docker tag image_name:version_tag
us.gcr.io/project_name/image_name:version
```

Tag the source image such that it refers to a target image in the GCP container registry

```
docker push us.gcr.io/project_name/image_name:version
```

Push the image onto the registry

```
gcloud container clusters create gk-cluster --num-nodes=1
```

Create a Google Kubernetes cluster and name that cluster gk-cluster. The number of nodes have been specified as 1

```
gcloud container clusters get-credentials gk-cluster
```

Get the authentication credentials to deploy the container by provisioning kubectl which communicates with the API

```
kubectl create deployment web-server
--image=us.gcr.io/project_name/image_name:version_tag
```

Create a deployment service to the cluster and name it web-server. --image specifies the container image to use

```
kubectl expose deployment web-server --type LoadBalancer --port
80 --target-port 80
```

Expose your service running within a cluster over an internet port.

PART 4

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

Kubernetes Pod: A group of one or more containers that run within the same network.

Kubernetes Service: service that deploys a group of pods (cluster) on a unique external IP address.

Kubernetes Node: a virtual (or physical) machine that is doing work for the cluster.

Kubernetes Deployment: instructions given to tell kubernetes how to deploy the pods.

What's meant by replicas?

Replicas are a number of pods that are running the same application.

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

LoadBalancer: using a cloud provider to host a Kubernetes cluster.

ExternalName: mapping a service to a DNS name specified.

NodePort: Setting up a load balancing solution exposed on a static port that can be defined or

automatically assigned.

ClusterIP: Can handle communications between the front-end and back-end of an app