SOFE 4630U: Cloud Computing
February 3, 2022
Group 11 - Group Report
Project Milestone - IaaS: Virtualization and Containerization
Fajer Zayed (100672347), Ireni Ruthirakuhan (100657302), Raveenth Maheswaran (100704540), Yale Wang (100673933)

What are docker images, containers, and registry?

Docker images	- A file that that contains the instructions of how to build a docker container - It describes what needs to be set up in the container in order to fulfill its function - Helps build/construct the docker container
Docker containers	- A virtual environment that can deploy applications on their own - The container has the ability to allow developers to package up an application - Package it with all necessary elements/components - Containers are an improvement on virtual machines because they can share access to an OS
Docker registry	- A collection of and storage for docker images - It hosts the content and makes it available to and requesting consumers - Represents the system for versioning, storing, and the distribution of docker images

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

- docker ps
    - ps = process status
    - Shows all containers currently running
- docker images
    - Shows all images created/match the argument
- docker build -t hello-world:1.0 .
    - Builds docker container
        - Builds container name and its tag version
        - Builds in the current directory (represented by .)
- Docker run hello-world:1.0

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

- docker ps -a → used to get container id
    - docker stop <container id>
        - used to stop running container
        - Can be verified by docker ps
    - docker rm -f <container id/application name>
        - Used to stop container and delete/remove it

Video Link:
https://drive.google.com/file/d/1Z1AO8fhjhKL-URceabJEHkHJUOdHRO5J/view?usp=sharing

What's a multi-container Docker application?

- Allows multiple containers to run at the same time on separate host ports since ports can be already allocated to a previous running container
- Takes advantage of the ability to compartmentalize the components of a web app deploying those components to different containers.
    - E.g. the application front end and the database
- Bound to distinct ports, without interfering with one another.

How do these containers communicate together?

- Containers communicated together via bridge networks
    - Example: `docker network create app-network`
      The network name in this case is app-network
- Containers address each other by using the IP address assigned to them in the network
- They communicate with each other via HTTP protocol
    - Containers expose their own ports and send HTTP requests to access resources

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

- Docker stop <container id>
- Docker image rm <image id>

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

- docker pull mysql
    - Pulls the official mysql image available on DockerHub
- docker build -t my-web-app:1.0 .
    - Builds an image named my-web-app
- docker run --name app -d -p 8080:8080 my-web-app:1.0
    - Creates a container based on the image followed by the port on the host machine that is binded to the container port
- docker network create app-network
    - Creates a network named app-network.
- docker network ls
    - Shows the current list of networks.
- docker network connect app-network app-db
    - Connects the app-db container to the app-network network

Video Link
https://drive.google.com/file/d/1_uls2hD1f_uYhCXzOQw-FAmfmcY05SJN/view?usp=sharing

Video Link
https://drive.google.com/file/d/1efxK_K4yj5fdjmaVgBHsNmwP8D7tnsYv/view?usp=sharing

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

- Set zone
    - Gcloud config set compute/zone us-central1-a
- Create the cluster
    - Gcloud container clusters create gk-cluster --num-nodes=1
        - Container = service
        - gk-clusters = cluster name
        - Creating 1 node
- Deploy container
    - First get credentials
        - gcloud container clusters get-credentials gk-cluster
        - Configures kubectl to use the cluster created
    - Deploying app to cluster
        - kubectl create deployment web-server --image=us.gcr.io/projectname/cad-site:version1
        - Web-server = name of app
- Expose to the internet
    - kubectl expose deployment web-server --type LoadBalancer --port 80 --target-port 8080
        - web-server = deployment name
        - Port initializes the public port 80 to the internet, the target port routes the traffic to port 8080 of the app
        - LoadBalancer type creates a Compute engineer load balancer
- See status of pods
    - Kubectl get pods
- Kubectl get service web-server
    - cp external ip then paste in browser

Video Link
https://drive.google.com/file/d/1RuxKmjnXfNccnfvbHnPdYEFIxEOf8w2K/view?usp=sharing

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

Kubernetes pod	- The basic building block of Kubernetes - Are designed to manage/support multiple containers

	- Run on nodes
Kubernetes service	- A deployed group of pods within a cluster - Used to connect the pods to the service name and IP address - Provide discovery and routing between pods - Also maintains access policies for the that IP

Kubernetes node	- Is the physical or virtual resource that hosts the Pod
	- Runs services necessary for containers that make up the cluster's workload
	- Has a kublet
	  - A process that communicates between the control plane and the node
	- Manages pods and the containers on the machine
	- Has a container run time
	  - Pulls container image from registry, unpacks container and runs the application

Kubernetes deployment	- Used to tell kubernetes how to modify/create instances of pods that hold the containerized application
	- Can be automated
	- Scales the replica pods, enable rollout of new code/rollback to earlier deployment version
	  - Provides updates for pods and replica sets
	  - Can replace a failed pod/bypass down nodes
	  - Replaces pods to make sure that the applications continue to work as expected
	  - Ensures that they are running, as expected, across all nodes within the cluster
	- Deployments are used to create new replicas/remove existing deployments
	- It has the ability to adopt their resources with new deployments

What's meant by replicas?

- A replicaSet is an abstract concept of a group of copies of Pods that are guaranteed to be running in a cluster and ensures the availability of these identical pods at any given time
- Prevents users from losing access to their application when a Pod fails
  - Provides maintainability and reliability of user and application data

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

- The four types of Kubernetes' services are ClusterIP, NodePort, LoadBalancer, and ExternalName.

Cluster IP	- A default service type that provides network connectivity within your cluster and can only be accessed inside the cluster - Internal clients are able to send requests to a stable IP - Lasts for the life of the service
Node Port	- A service that can be accessed from outside the cluster by using <NodeIP>:<NodePort> - An extension of the ClusterIP service where ClusterIP will be created for each node with the node port - Where clients have the ability to send requests to IP of node on 1/+ nodePort values - Has a status port on each node's IP
LoadBalancer	- A service that provides an external load balancer. - The load balancer routes to the ClusterIP and NodePort services that are created - Clients send requests to IP address of network load balancer
ExternalName	- A service that uses DNS names instead of default names