Faculty of Engineering & Applied Science

Course #Title: Cloud Computing

Date: February 3rd, 2022

Project Milestone #1

Title: IaaS: Virtualization and Containerization

Group #2

Student names: Richard Said - 100659431, Kaamran Minhas - 100593277, Taha Hashmat - 100689792, Adam Wong Chew Onn - 100598499, Sarthak Sharma - 100604428

Google Link for Final Videos:

https://drive.google.com/drive/folders/1CSU6L1t2Bk5dTk1K7PIkFcMW5dSEE1pD?usp=sharing

GitHub Link for Individual and Final Submissions:

https://github.com/tahahashmat/Group2-ProjectMilestone1

Docker - HelloWorld:

Q5. What are docker images, container, and registry?

Docker images are essential building blocks that act as classes in docker when running in containers.
Docker containers act as virtual machines without a guest OS that contain all the code required for applications to run.
Docker registry is an open sourced compilation or collection of repositories with their own, push, pull, and get, commands.

Q6. List the Docker commands used in the video with a brief description for each command and option

1. docker version - this simply outputs the version and status of docker used in the terminal
2. docker build -t hello-world:1.0 . : Builds an image with the name hello-world and tag 1.0.
3. docker images - lists the current docker images
4. docker run hello-world:1.0 : creates a docker container based on the hello-world:1.0 image and terminates it
5. docker ps - lists containers currently running
6. docker ps -a: shows a list of all containers, running and terminated
7. docker build -t hello-world:2.0 . - this command builds the docker file specifying the tag 2 and hello world image name
8. docker run -d hello-world:2.0: creates a container in detached mode meaning a container starts and runs in the background
9. docker logs <container id> - see what is happening in the container

Q7. At the end of the video, there are two running containers, what commands can be used to stop and delete those two containers?

- Docker kill <container id> - will stop the containers from running or being active
- Command to delete a container: docker container rm *container ID*

Docker - WebApp:

Q8. What's a multi-container Docker application?

A multi-container docker application is an application where multiple independent containers, each serving a specific purpose, are able to communicate with each other such as an SQL database and a separate container sending no receiving data.

Q9. How are these containers communicated together?

The containers each perform specific tasks therefore a "network" is needed for them to communicate with each other and docker creates this thing called a bridge which creates this network.

Q10. What command can be used to stop the Docker application and delete its images?

- sudo docker create --name firstkill bitnami/wordpress - this will kill or stop the docker container using its name.
- Command to delete docker images: docker image -rmi *image ID*

Q11. List the new docker commands used in the video with a brief description for each command and option.

1. docker pull mysql - pulls in a mysql image from dockerhub
2. docker run --name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql: creates a container called "app-db" which will set a root password for mysql database and gives the database name "myDB" with mysql as the image name
3. docker run --name app -d my-web-app:1.0: runs the app in detached mode using the name option for the container.
4. docker rm -f - stops containers forcefully
5. docker run –name app -d -p 8080:8080 my-web-app:1.0 - specifying a port on the host machine and the container port that the host port will be bound to
6. docker run --name app -d -p 8081:8080 my-web-app:1.0: Docker instead sets the host machine port to 8081.
7. docker network create app-network - creates a new network alongside 3 other default networks
8. docker network ls: lists all the networks in docker for the containers.
9. docker network connect app-network app-db: connecting db container to the network
10. nvm clean install - building the application file
11. docker run --name app -d -p 8080:8080 network=app-network my-web-app:1.0: re-creating application container with network in same command.
12. docker compose up -d: creates and starts the containers

GCP Shell:

Q13. List all used GCP shell commands and their description in your report.

1. docker run -p 8080:80 nginx:latest: simply begins running docker container in port 8080 as host.
2. cat index.html - copying all the index.html
3. docker cp index.html<container-id>:/usr/share/nginx/html/: specifies the html file that will be used to make the side into container id.
4. docker commit 09c7b89bad81 cad/web:version1 - committed the container and created a new image
5. docker tag cad/web:version1 us.gcr.io/youtube-demo-255723/cad-site:version1: specifies tag and name of container
6. docker push us.gcr.io/synthetic-nova-340209/cad-site:version1 - pushing the docker image tagged in the last command
7. gcloud config set project youtube-demo-255723: configure the project name to default.
8. gcloud config set compute/zone us-central1-a - specifies location of a cluster
9. gcloud container clusters create gk-cluster --num-nodes=1: actually creates the gk cluster for the containers.
10. gcloud container clusters get-credentials gk-cluster - deploying the container and gives credential to the gk cluster
11. Kubectl create deployment web-server --image=us.gcr.io/youtube-demo-255723/cad-site:version1: deploys the web server using the image.
12. kubectl expose deployment web-server --type LoadBalancer --port 80 --target-port 8080 - exposing the deployment to the internet
13. kubectl get pods: shows you the running pods.
14. kubectl get service web-server - specified server with external IP

Kubernetes:

Q17. What is Kubernetes' pod, service, node, and deployment?

- **Pods:** are the smallest and most basic deployable units in kubernetes that can not only hold multiple containers but containers that can communicate with each other in a network.
- **Service:** since pods contain their own IP addresses that cannot be dynamically changed, created or destroyed which is what's called a kubernetes service. This service is essentially an abstract depiction of a cluster or pods group in docker.
- **Deployment:** This is an instruction kubernetes of the creation and modification of instances for pods. A pod that fails to run can be replaced by a replica, continuing the streamline.

Q18. What's meant by replicas?

Replicas are what keep the streamline going when a pod fails or suddenly becomes no longer responsive. Replicas also keep the static number of pods in the cluster the same as it will replace the unresponsive pod avoiding complexities.

Q19. What are the types of Kubernetes' services? What is the purpose of each?

- ClusterIP - ONLY services that are within clusters are shown
- NodePort - Each node's IP will expose the service from the port (that is static).
- LoadBalancer - The load balancer provided by the cloud provider will expose the service.
- ExternalName- Returns a CNAME record instead of a cluster IP
- Headless- A service that returns the specific IP address of different pods in the cluster