Ontario Tech University

Faculty of Engineering

SOFE 4790U

Distributed System Labs

Fall 2022

Prepared by George Daoud and Mohamed EL-darieby

Lab 1: Introduction to Google Kubernetes Engine (GKE)

Objective:

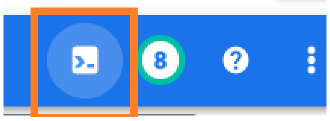
1. Get familiar with Docker images and containers.
2. Learns various Kubernetes tools.
3. Learn how to use Google Cloud Platform (GCP).
4. Compose YAML files to deploy cloud applications.

Repository:

https://github.com/GeorgeDaoud3/SOFE4790U-lab1

Procedure:

1. Watch The following video to understand Docker terminologies, https://youtu.be/rOTqprHv1YE
2. To manage Docker images and applications, we will use Kubernetes, watch the following video to get familiar with Kubernetes and its components https://youtu.be/cC46cg5FFAM
3. Let's start our first application on Google Cloud Platform (GCP). Start a free trial using a Gmail account following the following video. https://youtu.be/P2ADJdk5mYo
4. Although there is a graphical tool to create a Kubernetes cluster within GCP, commands will be used in the Lab.
   a) Within GCP, open the console terminal

   b) Set the default compute zone to northamerica-northeast1-b

   ```
   gcloud config set compute/zone northamerica-northeast1-b
   ```

   c) Enable GKE by typing "Kubernetes Engine", select "Kubernetes Engine API", click Enable.

   d) Create a three-nodes cluster on GKE.

   ```
   gcloud container clusters create openfaas --num-nodes=3
   ```

   Note: if Authorization windows popped up, click Authorize

   Note: if you got an error that there is no available resources to create the nodes, you need to change the default compute zone (e.g. to us-central1-a)
5. To deploy your first application, a pre-made MySQL image will be used like.

   ```
   kubectl create deployment mysql-deployment --image mysql/mysql-server --port=3306
   ```

where **mysql/mysql-server** is the name of Docker image, **3306** is the port the number that will be exposed from the docker image to the outside world, and **mysql-deployment** is the name that will be used by Kubernetes to access this deployment.

By default, only one pod will be created per deployment. The status of the deployment can be checked by the following command

```
kubectl get deployment
```

While the status of pods can be accessed by the following command

```
kubectl get pods
```

check that the deployment is available and that the pod is running successfully (it may take some time until everything is settled down)

6. To access the MySQL logs,

a) According to the docker image documentation, as we didn't specify the root password, it will be generated randomly. To get that password, the logs generated locally by the pod should be accessed and filtered it for a certain line using the following command

```
kubectl logs <pod-name> 2>&1 |grep GENERATED
```

accessing the logs of a pod helps a lot of troubleshooting it in case of error or if it crashed.

b) You can access the database by run the command **mysql** within the pod, by using the following command

```
 kubectl exec -it  <pod-name>  -- mysql -uroot -p
```

Kubernetes **exec** command allows you to execute a certain command within a certain pod in interactive (**-i** option) and by using the Linux terminal (**-t** option). The command we want to execute is **mysql** which open a cli interface to the MySQL database. It has two options, the first is **-u** followed by the username, i.e. **root**. The second is **-p** which asks you to enter the root password you got in a). Note, there is no whitespace between the **-u** and **root**.

c) The first step after successfully login, is to change the **root** password, using the following MySQL command.

```
ALTER USER 'root'@'localhost' IDENTIFIED BY <new-password> ;
```

d) Then you can run any MySQL command, like

```
show databases;
```

For displaying all available schemas

e) To exit MySQL cli, execute

```
exit
```

f) To login again to the cli, we must use the new password and you can add it to the **mysql** command like

```
kubectl exec -it  <pod-name>  -- mysql -uroot -p<root-password>
```

Again, there is no whitespaces between -p and the password

g) To create a new user, use the following MySQL command

```
CREATE USER 'user'@'%' IDENTIFIED BY 'sofe4790u';
GRANT ALL PRIVILEGES ON *.* TO 'user'@'%' WITH GRANT OPTION;
```

h) Now exit the MySQL CLI, if you already logged into it.

7. To give a deployment an IP address

a) A load Balancer service should be created to that deployment

```
kubectl expose deployment mysql-deployment --type=LoadBalancer --name=mysql-service
```

You can add two options **--port** that specify the service port number and **--target-port** that specifies the pod port number. If not specifies both will be the same as the port numbers already exposed via the deployment command.

b) To check the status of the service, use this command

```
kubectl get service
```

It may take some time until the external IP address is changed from pending to a valid IP address. You may need to repeat the previous command

c) Once you get a valid external IP address, you can use it to connect to the deployed MySQL server from any machine. For example, to connect to it from the GCP console, you can use the following command.

```
mysql -uuser -psofe4790u -h<IP-address>
```

8. A more advanced deployment.

The other way to create the application is to configure a **YAML** file containing the deployment and the service and all their parameters and then build it using Kubernetes.

a) Let’s first delete the deployment and the services previously created

```
kubectl delete deployment mysql-deployment
kubectl delete service mysql-service
```

b) Create a file containing the following script using the editor available in GCP and name it **mysql.yaml** using the editor available in GCP. Note: the whitespaces included in the script are <u>crucial</u> (you can download the file from the repository if you have problems)

```
apiVersion: v1
kind: Service
metadata:
  name: mysql-service
spec:
  type: LoadBalancer
  ports:
    - port: 3306
  selector:
    app: mysql
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - image: mysql/mysql-server
          name: mysql
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: password
            - name: MYSQL_USER
              value: user
            - name: MYSQL_PASSWORD
              value: sofe4790u
            - name: MYSQL_DATABASE
              value: myDB
          ports:
            - containerPort: 3306
              name: mysql
```

The first part includes the parameters of the service. While the second part is the deployment parameters. It includes a field for the replica count which is set to 1. This represent the number of pods that will be created from that image while the template field include the descriptions of the pod. It includes key-value pairs called environment variables. Those variables are used to customize the container and they dependent on the image design which means they will be different from a Docker image to another and can be found on the Docker image documentation. The first variable sets the root password to password while the next two variables will create a user called **user** with a password **sofe4790u** while the last one will set the default schema to **myDB**.

c) Deploy it to GKE

```
kubectl apply -f mysql.yaml
```

d) Check the status of the service, deployment, and the pod.

e) Login into MySQL server via the service new IP address.

f) Try to run the following SQL statements

```
use myDB;
create table person( id int, age int, name varchar(50));
insert into person values(1,30,'tom');
insert into person values(2,23,'adam');
insert into person values(3,79,'Joe');
select * from person where age>=30;
```

g) Exit the MySQL CLI

h) (optional) after creating a video for submission, you can delete the deployment by using the following command

```
kubectl delete -f mysql.yaml
```

Discussion:

Summarize what you have learned about docker and Kubernetes including the used terminologies and their descriptions. What ‘s the advantages and disadvantages of using docker images against using virtual machines.

Design:

MongoDB is anther type of databases. It’s required to deploy it using GKE using a YAML file. If you used any Kubernetes tool in your deployment that is not included in the lab you should describe it and why you used it

Deliverables:

1. A report that includes the discussion and the design parts.

2. An **audible** video of about 3 minutes maximum showing the final results of following the lab steps. It should include showing the deployment, service, and pod of MySQL and the execute of MySQL commands (step 12).
3. Another **audible** video of 5 minutes maximum showing the deployment and using of MongoDB within GCP.

++ Lab Ends ++