Faculty of Engineering and Applied Science

SOFE 4790U Computer Networks

Group 17 CRN 44425

Lab #1

Name	Student #
Owen Musselman	100657709

Objectives:

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

MySQL Implementation

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get deployment
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
mysql-deployment   1/1     1            1           4m8s
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
mysql-deployment-7467c475f8-pzxdj   1/1     Running   0          4m26s
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Checking the created deployment(s) and pod(s).

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl logs mysql-deployment-7467c475f8-pzxdj 2>&1 |grep GENERATED
[Entrypoint] GENERATED ROOT PASSWORD: 2uZP=.J1h_7%CUN0=l,eq%RSz00*P15N
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Searching for the automatically generated password.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl exec -it mysql-deployment-7467c475f8-pzxdj -- mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.30

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Logging into mysql by using the generated password.

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'adminPass'
Query OK, 0 rows affected (0.01 sec)
```

Changing the generated password with a new password.

```
mysql> exit
Bye
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Exiting mysql.

```
mysql> CREATE USER 'user'@'%' IDENTIFIED BY 'sofe4790u';
Query OK, 0 rows affected (0.06 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'user'@'%' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)
```

Creating a new user and assigning permissions to this newly created user.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get service
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)          AGE
kubernetes      ClusterIP      10.108.0.1      <none>          443/TCP          47h
mysql-service   LoadBalancer   10.108.10.233   34.152.15.114   3306:32582/TCP   4m4s
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Checking the service status.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ mysql -uuser -psofe4790u -h 34.152.15.114
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Connect to mysql server using the external IP address.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl delete deployment mysql-deployment
deployment.apps "mysql-deployment" deleted
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl delete service mysql-service
service "mysql-service" deleted
```

Deleting the deployment.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl apply -f mysql.yaml
service/mysql-service created
deployment.apps/mysql-deployment created
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Deploy mysql service by executing the mysql.yaml file.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get service
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)          AGE
kubernetes      ClusterIP      10.108.0.1     <none>          443/TCP          2d
mysql-service   LoadBalancer   10.108.8.33    34.152.15.114   3306:31301/TCP   3m42s
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Checking the IP addresses of the service.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
mysql-deployment-5496fdc956-6p7lr   1/1     Running   0          8m11s
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get deployment
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
mysql-deployment   1/1     1            1           10m
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$
```

Checking the pod(s), and deployment(s).

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ mysql -uuser -psofe4790u -h 34.152.15.114
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Logging into the mysql server using the new external IP address.

```
mysql> use myDB;
Database changed
mysql> create table person( id int, age int, name varchar(50));
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> insert into person values(1,30,'tom');
Query OK, 1 row affected (0.04 sec)

mysql> insert into person values(2,23,'adam');
Query OK, 1 row affected (0.03 sec)

mysql> insert into person values(3,79,'Joe');
Query OK, 1 row affected (0.04 sec)

mysql> select * from person where age>=30;
+------+------+------+
| id   | age  | name |
+------+------+------+
|    1 |   30 | tom  |
|    3 |   79 | Joe  |
+------+------+------+
2 rows in set (0.03 sec)
```

Changing the database to be used, creating a table and inserting 3 items into the table.

MongoDB Implementation

```yaml
apiVersion: v1
kind: Service
metadata:
  name: mongodb-service
spec:
  type: LoadBalancer
  ports:
    - port: 3306
  selector:
    app: database
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: database
  template:
    metadata:
      labels:
        app: database
    spec:
      containers:
        - image: mongo:4.0.8
          name: mongodb
          env:
            - name: MONGO_INITDB_ROOT_USERNAME
              value: Owen
            - name: MONGO_INITDB_ROOT_PASSWORD
              value: randomPass
          ports:
            - containerPort: 3306
              name: mongodb
```

YAML file for mongodb setup.

```
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
mongodb-deployment-7945646c67-575s7   1/1     Running   0          3m20s
mysql-deployment-5496fdc956-6p7lr     1/1     Running   0          5d
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl exec -it mongodb-deployment-7945646c67-575s7 -- sh
# mongo
MongoDB shell version v4.0.8
connecting to: mongodb://127.0.0.1:27017/?gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("654d2933-0b70-45fb-898f-82ca1974e814") }
MongoDB server version: 4.0.8
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        http://docs.mongodb.org/
Questions? Try the support group
        http://groups.google.com/group/mongodb-user
> use admin
switched to db admin
> db.auth('Owen', 'randomPass')
1
> show db
2022-09-16T21:06:09.795+0000 E QUERY    [js] Error: don't know how to show [db] :
shellHelper.show@src/mongo/shell/utils.js:1066:11
shellHelper@src/mongo/shell/utils.js:766:15
@(shellhelp2):1:1
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> ^C
bye
# ^C
# exit
slikcaustic1@cloudshell:~ (bubbly-granite-362015)$ kubectl exec -it mongodb-deployment-7945646c67-575s7 -- sh
```

Getting pod name, executing using the mongodb pod name. Then entering the mongo shell and logging in as admin using the user created in the YAML file.

```
> db.auth('Owen', 'randomPass')
1
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use mngTest
switched to db mngTest
> db.admin.insert({id:1, age: 30, name: "Tom"})
WriteResult({ "nInserted" : 1 })
> db.admin.insert({id:2, age: 23, name: "Adam"})
WriteResult({ "nInserted" : 1 })
> db.admin.insert({id:3, age: 79, name: "Joe"})
WriteResult({ "nInserted" : 1 })
> db.admin.find()
{ "_id" : ObjectId("6324f12b9cb5407c19ea8c60"), "id" : 1, "age" : 30, "name" : "Tom" }
{ "_id" : ObjectId("6324f15b9cb5407c19ea8c61"), "id" : 2, "age" : 23, "name" : "Adam" }
{ "_id" : ObjectId("6324f16d9cb5407c19ea8c62"), "id" : 3, "age" : 79, "name" : "Joe" }
> db.admin.find({age: {$gte: 30}})
{ "_id" : ObjectId("6324f12b9cb5407c19ea8c60"), "id" : 1, "age" : 30, "name" : "Tom" }
{ "_id" : ObjectId("6324f16d9cb5407c19ea8c62"), "id" : 3, "age" : 79, "name" : "Joe" }
```

Entering data into mongo collection and querying to find people that are 30 and over.

Discussion:

Terminologies of Docker:

- Docker Image: A set of instructions in which docker images are created.

- Docker Container: It is a runtime instance of a docker image. They are useful as they let devs to bundle their components together like libraries and other dependencies.
- Docker Hub: The place where you can find docker images. The hub also allows for the sharing of images that might be useful to other devs.

Terminologies of Kubernetes:

- Cluster: A set of Worker nodes and their Master node.
- Master Node: Controls a set of Worker nodes.
- Worker Node: This node is running the containerized application.
- Pod: Where the containerized application is.
- Service: Manages the connectivity to the created pods.
- Deployment: Manages the replicas of a pod to meet a standard of state management.

Advantages of using Docker over VMs	**Disadvantages of using Docker over VMs**
More efficient - Uses less computational resources than VMs.	Less secure in terms of application separation. Applications share one OS in Docker.
No single point of failure.	Persistent data storage is hard. When the container shuts down, the data will disappear.
Lower cost than	Latency issues. Communications and data sent over a network and can be lost or delayed due to the network.
Higher portability and scalability.	Only microservice applications are able to utilize Docker to its fullest.

Design:

Mongodb was deployed using YAML, where the user information was specified, along with the replicas, the mongodb version to use for the image, in the case 4.0.8. The YAML file also specifies what pod traffic should be sent to, in this case send the traffic to the mongo pod, not the mysql pod.

Conclusion:

After completing the objectives for this lab it has increased our familiarity with Docker images and containers, and Kubernetes tools as well as gaining experience with the Google Cloud Platform.