

**CLOUD COMPUTING
CSC-8110
COURSE-WORK REPORT**

NAME: KAILASH BALACHANDIRAN
STUDENT ID: 220243160

AIM: To understand the fundamentals of programming and deploying Docker-based (an emerging cloud virtualisation technology) application hosting environment and programming and deploying cloud infrastructures using Terraform. By completing the coursework, you will be able to gain hands-on experience on the following inter-related aspects:

- Configuring a Docker-based application hosting environment;
- Pushing and pulling images from the Docker Hub (global repository of software components' images maintained by the developers);
- Creating and deploying a complex web application stack consisting of multiple software components (e.g., database, web server, etc.);
- Monitoring the ongoing performance of your application stack using the inherent features provided by the Docker environment.
- Deploying a Kubernetes service on Azure Kubernetes Service and running a sample microservice on the cluster

Task 1: Deploy a web application component in Docker Environment

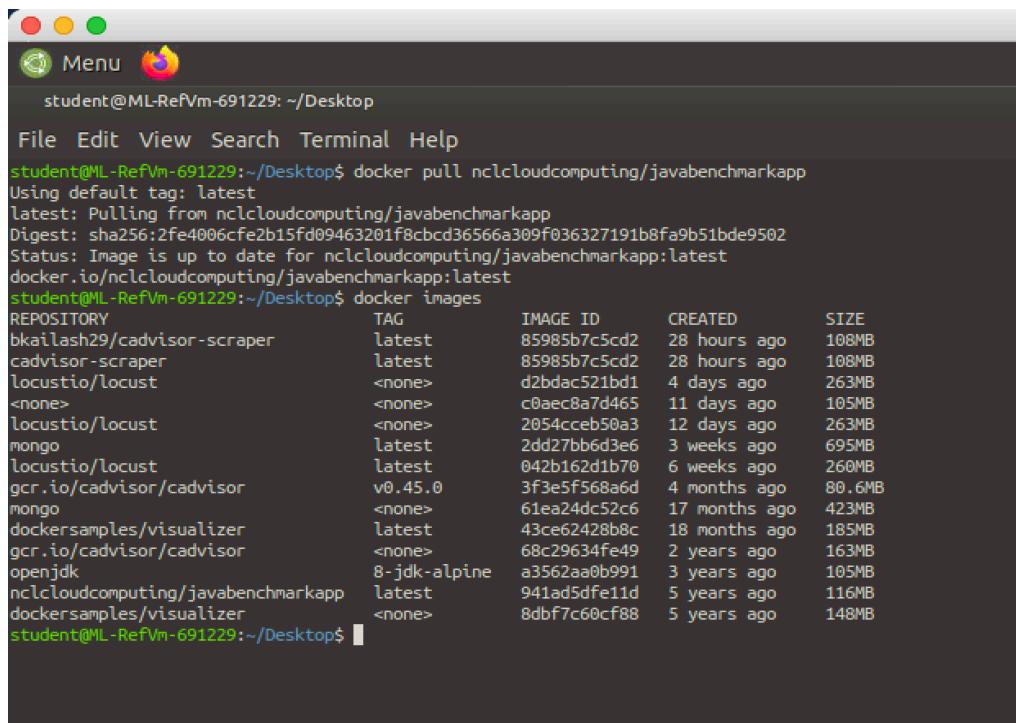
Task Objective: To understand and learn how to pull and run Docker images from Docker Hub using the command line interface. sample Java web application component image, named "nclcloudcomputing/java benchmark app", has been uploaded to the Docker Hub. The image contains a ready-to-use implementation of a web application deployed in a Tomcat server (an open-source web server). In terms of computational logic, the web application implements a prime number check on a large number. By doing so, the application can generate a high CPU and memory load.

1. Pull and run the Docker image "nclcloudcomputing/javabenchmarkapp" from Docker Hub in the virtual machine running on your laptop/PC. Perform these tasks using the command line interface (CLI).

The commands in *Figure 1.1* are executed in the terminal using docker CLI

The following command is used to pull the java web application image
“*docker pull nclcloudcomputing/javabenchmarkapp*”

The following command is used to view the pulled images in the docker
“*docker images*”



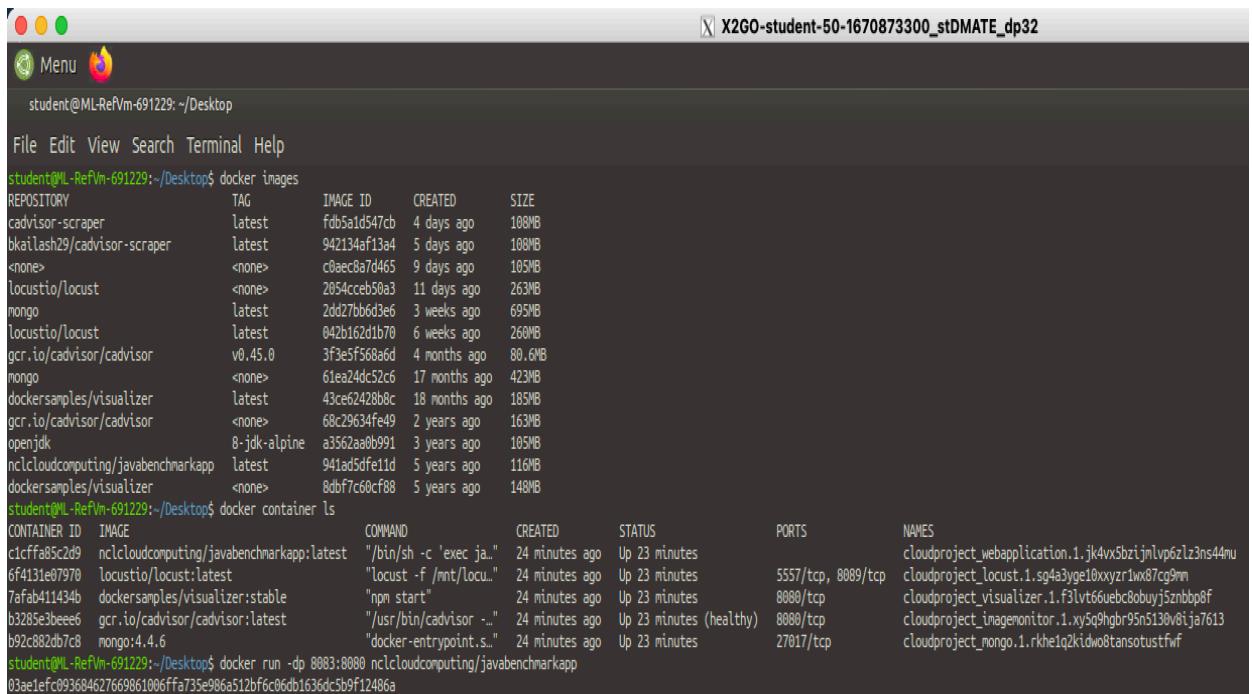
```
student@ML-RefVm-691229:~/Desktop$ docker pull nclcloudcomputing/javabenchmarkapp
Using default tag: latest
latest: Pulling from nclcloudcomputing/javabenchmarkapp
Digest: sha256:2fe4006cfe2b15fd09463201f8cbcd36566a309f036327191b8fa9b51bde9502
Status: Image is up to date for nclcloudcomputing/javabenchmarkapp:latest
docker.io/nclcloudcomputing/javabenchmarkapp:latest
student@ML-RefVm-691229:~/Desktop$ docker images
REPOSITORY          TAG      IMAGE ID   CREATED    SIZE
bkailash29/cadvisor    latest   85985b7c5cd2  28 hours ago  108MB
cadvisor-scraper       latest   85985b7c5cd2  28 hours ago  108MB
locustio/locust        <none>   d2bdac521bd1  4 days ago   263MB
<none>                <none>   c0aec8a7d465  11 days ago  105MB
locustio/locust        <none>   2054ccceb50a3  12 days ago  263MB
mongo                 latest   2dd27bb6d3e6   3 weeks ago  695MB
locustio/locust        latest   042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor v0.45.0  3f3e5f568a6d  4 months ago  80.6MB
mongo                 <none>   61ea24dc52c6  17 months ago 423MB
dockersamples/visualizer latest   43ce6242b8c   18 months ago 185MB
gcr.io/cadvisor/cadvisor <none>   68c29634fe49  2 years ago  163MB
openjdk               8-jdk-alpine a3562aa0b991  3 years ago  105MB
nclcloudcomputing/javabenchmarkapp latest   941ad5dfe11d  5 years ago  116MB
dockersamples/visualizer <none>   8dbf7c60cf88  5 years ago  148MB
student@ML-RefVm-691229:~/Desktop$
```

Figure 1.1

In *Figure 1.2* we use docker images to view the images and “*docker container ls*” to view the running containers

The following command is used to run the java web application image on port 8083 in detached mode

“*docker run -dp 8083:8080 nclcloudcomputing/javabenchmarkapp*”



```

student@ML-RefVm-691229:~/Desktop$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
cadvisor-scraper    latest   fdb5a1d547cb  4 days ago  108MB
bkailash29/cadvisor    latest   942134af13a4  5 days ago  108MB
<none>              <none>   c0aec8a7d465  9 days ago  105MB
locustio/locust      <none>   2054cce150a3  11 days ago  263MB
mongo                latest   2dd27bb6d3e6  3 weeks ago  695MB
locustio/locust      latest   042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor v0.45.0  3f3e5f568a60  4 months ago  80.6MB
mongo                <none>   61ea24dc52c6  17 months ago  423MB
dockersamples/visualizer latest   43ce6242880c  18 months ago  185MB
gcr.io/cadvisor/cadvisor <none>   68c29634fe49  2 years ago  163MB
openjdk               8-jdk-alpine  a3562aa0b0991  3 years ago  105MB
nclcloudcomputing/javabenchmarkapp latest   941ad5dfe11d  5 years ago  116MB
dockersamples/visualizer <none>   8df7c760cf80  5 years ago  148MB
student@ML-RefVm-691229:~/Desktop$ docker container ls
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
c1cffa85c2d9        nclcloudcomputing/javabenchmarkapp:latest "/bin/sh -c 'exec ja..." 24 minutes ago   Up 23 minutes          5557/tcp, 8089/tcp   cloudproject_webapplication.1.jk4vx5bzijnlvp6zlz3ns44mu
6f4131e07970        locustio/locust:latest      "locust -f /mnt/locu..." 24 minutes ago   Up 23 minutes          8080/tcp             cloudproject_locust.1.sg4a3yge10xyzrlwx87cg9mm
7afab411434b        dockersamples/visualizer:stable  "npm start"         24 minutes ago   Up 23 minutes          8080/tcp             cloudproject_visualizer.1.f31vt66uebc8obuyj5znbbp8f
b3285e3beee6        gcr.io/cadvisor/cadvisor:latest  "/usr/bin/cadvisor ..." 24 minutes ago   Up 23 minutes (healthy)  8080/tcp             cloudproject_imagemonitor.1.y5g9hgb95n5130v6lja7613
b92c882db7c8        mongo:4.4.6           "docker-entrypoint.s..." 24 minutes ago   Up 23 minutes          27017/tcp            cloudproject_mongo.1.rkhe1q2kidwo8tansotustfwf
student@ML-RefVm-691229:~/Desktop$ docker run -dp 8083:8080 nclcloudcomputing/javabenchmarkapp
03ae1efc093684627669861006ffa735e986a512bf6c06db1636dc5b9f12486a

```

Figure 1.2

- Verify whether the web application is working by pointing your web browser to [http://localhost:8083/primecheck]. The page should display the time taken to perform a prime number checking calculation.

Figure 1.3 shows that the java web application image running on port 8083 and displaying the time taken to perform the prime number check calculation



Figure 1.3

Task 2: Deploy a complex web application stack in Docker Environment

Task Objective: To understand how to create a complex web application stack consisting of multiple Docker images delivering distinct services including a web server, load generation, database server, monitoring engine, and visualisation. This task will also help you in understanding how native Docker Compose and Swarm techniques can be leveraged to manage and deploy such complex web application stacks.

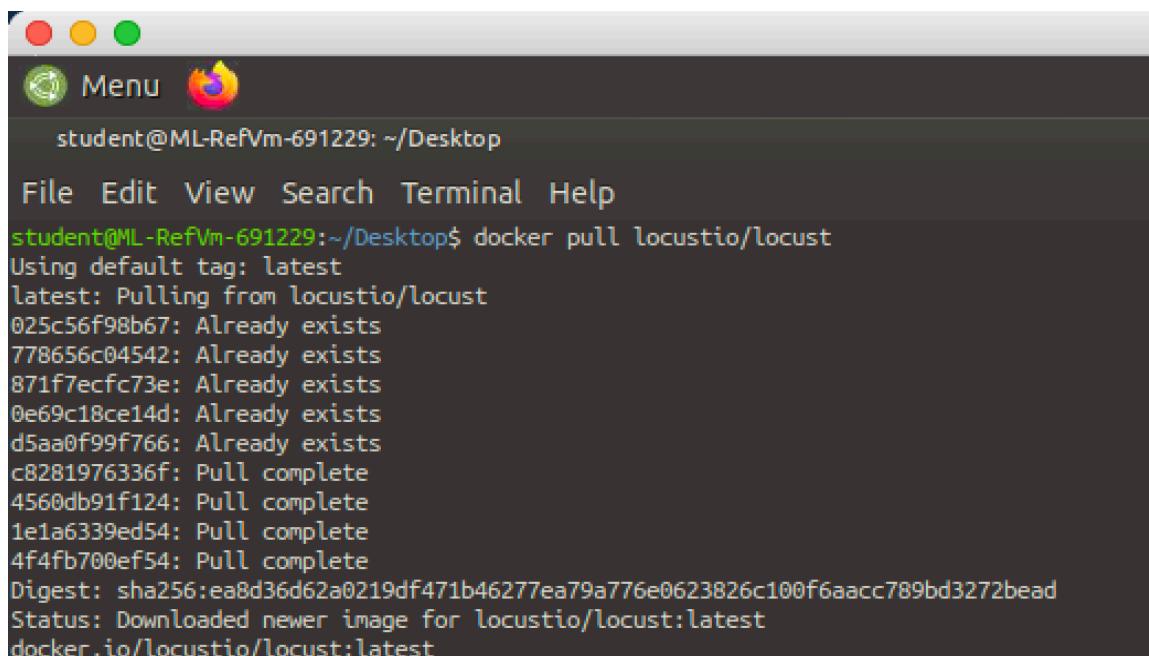
1. Pull the following images from the Docker Hub:

- a) load generator Docker image (*locustio/locust*);

Figure 2.1 shows that the docker image “*locustio/locust*” is pulled to the docker CLI from the docker hub using the command

“*docker pull locustio/locust*”

This is used to run load tests.



```
student@ML-RefVm-691229: ~/Desktop
File Edit View Search Terminal Help
student@ML-RefVm-691229:~/Desktop$ docker pull locustio/locust
Using default tag: latest
latest: Pulling from locustio/locust
025c56f98b67: Already exists
778656c04542: Already exists
871f7ecfc73e: Already exists
0e69c18ce14d: Already exists
d5aa0f99f766: Already exists
c8281976336f: Pull complete
4560db91f124: Pull complete
1e1a6339ed54: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:ea8d36d62a0219df471b46277ea79a776e0623826c100f6aacc789bd3272bead
Status: Downloaded newer image for locustio/locust:latest
docker.io/locustio/locust:latest
```

Figure 2.1

- b) MongoDB Docker image (*mongo*);

Figure 2.2 shows that the docker image “*mongo*” is pulled from the docker Hub using the command

“*docker pull mongo*”

Mongo is a NoSQL database used to store data.

```
student@ML-RefVm-691229:~/Desktop$ docker pull mongo
Using default tag: latest
latest: Pulling from library/mongo
846c0b181fff: Pull complete
ef773e84b43a: Pull complete
2bfad1efb664: Pull complete
84e59a6d63c9: Pull complete
d2f00ac700e0: Pull complete
96d33bf42f45: Pull complete
ebaa69d77b61: Pull complete
aa77b709a7d6: Pull complete
245bd0c9ace2: Pull complete
Digest: sha256:f1b54ae2acc7db563457f41443103a2d48d1ee5a13332734f82222aa719e2542
Status: Downloaded newer image for mongo:latest
docker.io/library/mongo:latest
```

Figure 2.2

c) cAdvisor monitoring Docker image (gcr.io/cadvisor/cadvisor:v0.45.0);

Figure 2.3 shows that the docker image “gcr.io/cadvisor/cadvisor:v0.45.0” is pulled from the docker Hub using the command

“*docker pull gcr.io/cadvisor/cadvisor:v0.45.0*”

cAdvisor is used to monitor containers. It provides monitoring data of CPU, memory, network, etc. and visualizes the data in graph form.

```
student@ML-RefVm-691229:~/Desktop$ docker pull gcr.io/cadvisor/cadvisor:v0.45.0
v0.45.0: Pulling from cadvisor/cadvisor
Digest: sha256:9360d7421c5e9b646ea13e5ced3f8e6da80017b0144733a04b7401dd8c01a5cb
Status: Image is up to date for gcr.io/cadvisor/cadvisor:v0.45.0
gcr.io/cadvisor/cadvisor:v0.45.0
```

Figure 2.3

d) Docker Swarm Visualizer (dockersamples/visualizer);

Figure 2.4 shows that the docker image “dockersamples/visualizer” is pulled from the docker Hub using the command

“*docker pull dockersamples/visualizer*”

A visualizer is used to ensure that all containers are up and running.

```
student@ML-RefVm-691229:~/Desktop$ docker pull dockersamples/visualizer
Using default tag: latest
latest: Pulling from dockersamples/visualizer
Digest: sha256:530c863672e7830d7560483df66beb4cbbcd375a9f3ec174ff5376616686a619
Status: Image is up to date for dockersamples/visualizer:latest
docker.io/dockersamples/visualizer:latest
```

Figure 2.4

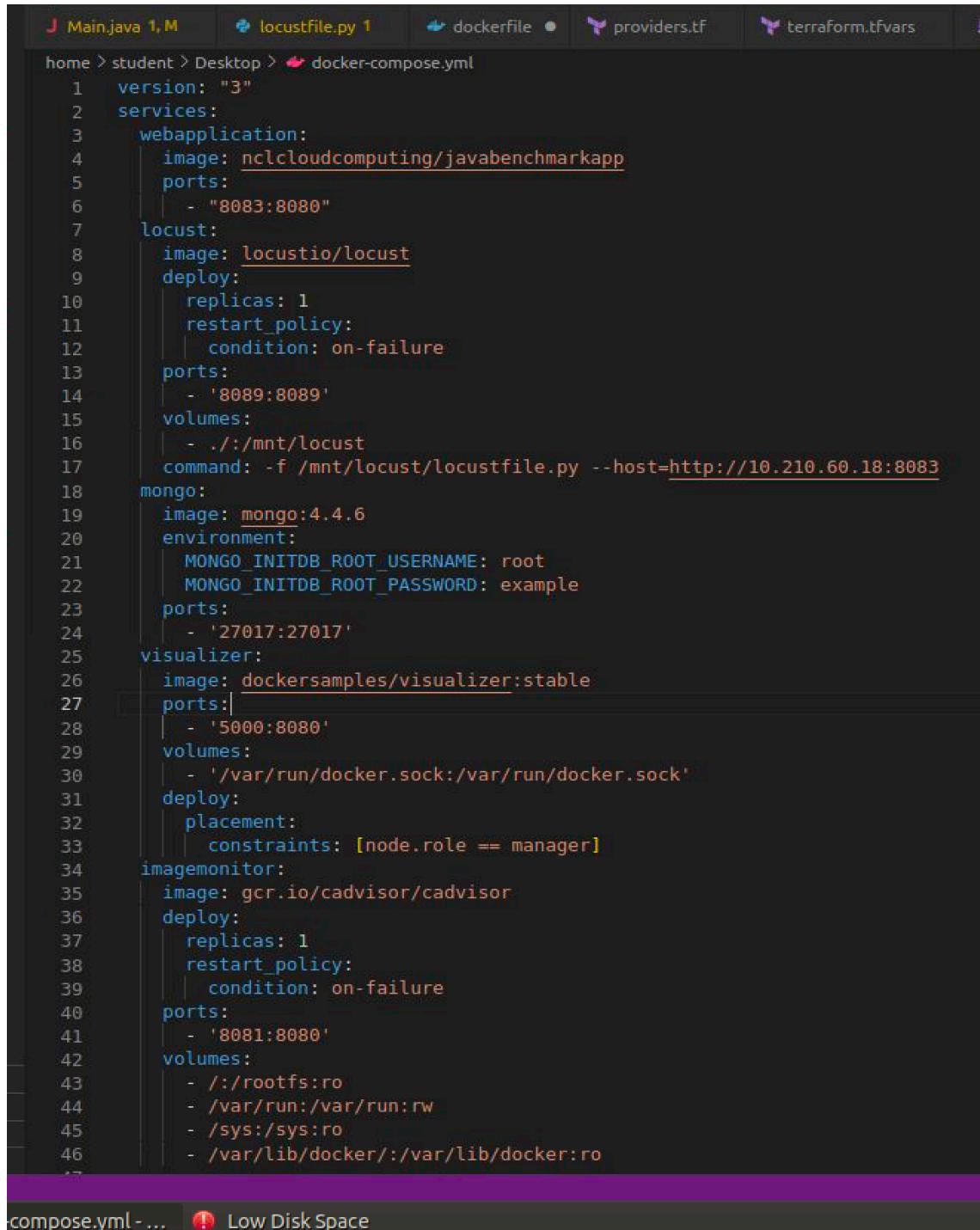
Figure 2.5 shows that all the above-mentioned docker images are pulled to the docker CLI, it can be viewed with the command “*docker images*”

```
student@ML-RefVm-691229:~/Desktop$ docker pull locustio/locust
Using default tag: latest
latest: Pulling from locustio/locust
025c56f98b67: Already exists
778656c04542: Already exists
871f7ecfc73e: Already exists
0e69c18ce14d: Already exists
d5aa0f99f766: Already exists
c8281976336f: Pull complete
4560db91f124: Pull complete
1e1a6339ed54: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:ea8d36d62a0219df471b46277ea79a776e0623826c100f6aacc789bd3272bead
Status: Downloaded newer image for locustio/locust:latest
docker.io/locustio/locust:latest
student@ML-RefVm-691229:~/Desktop$ docker pull mongo
Using default tag: latest
latest: Pulling from library/mongo
846c0b181fff: Pull complete
ef773e84b43a: Pull complete
2bfad1efb664: Pull complete
84e59a6d63c9: Pull complete
dzf00ac700e0: Pull complete
96d33bf42f45: Pull complete
ebaa69d77b61: Pull complete
aa77b709a7d6: Pull complete
245bd0c9ace2: Pull complete
Digest: sha256:f1b5a4e2acc7db563457f41443103a2d48d1ee5a13332734f82222aa719e2542
Status: Downloaded newer image for mongo:latest
docker.io/library/mongo:latest
student@ML-RefVm-691229:~/Desktop$ docker pull gcr.io/cadvisor/cadvisor:v0.45.0
v0.45.0: Pulling from cadvisor/cadvisor
Digest: sha256:9360d7421c5e9b646ea13e5ced3f8e6da80017b0144733a04b7401dd8c01a5cb
Status: Image is up to date for gcr.io/cadvisor/cadvisor:v0.45.0
gcr.io/cadvisor/cadvisor:v0.45.0
student@ML-RefVm-691229:~/Desktop$ docker pull dockersamples/visualizer
Using default tag: latest
latest: Pulling from dockersamples/visualizer
Digest: sha256:530c863672e7830d7560483df66beb4cbbcd375a9f3ec174ff5376616686a619
Status: Image is up to date for dockersamples/visualizer:latest
docker.io/dockersamples/visualizer:latest
student@ML-RefVm-691229:~/Desktop$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
locustio/locust     latest   104d48691d70  13 hours ago  263MB
bkallash29/cadvisor-scraprer  latest   85985b7c5cd2  28 hours ago  108MB
cadvisor-scraprer  latest   85985b7c5cd2  28 hours ago  108MB
locustio/locust     <none>   d2bdac521bd1  4 days ago   263MB
mongo               latest   0850fead9327  4 days ago   700MB
<none>              <none>   c0aec8a7d465  11 days ago  105MB
locustio/locust     <none>   2054ccbeb50a3  12 days ago  263MB
mongo               <none>   2dd27bb6d3e6  3 weeks ago  695MB
locustio/locust     <none>   042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor  v0.45.0  3f3e5fs568a6d  4 months ago  80.6MB
mongo               <none>   61ea24dc52c6  17 months ago  423MB
dockersamples/visualizer  latest   43cce6242b8c8  18 months ago  185MB
gcr.io/cadvisor/cadvisor  <none>   68c29634fe49  2 years ago  163MB
openjdk              8-jdk-alpine  a356za0@06991  3 years ago  105MB
nclcloudcomputing/javabenchmarkapp  latest   941ad5dfe11d  5 years ago  116MB
dockersamples/visualizer  <none>   8dbf7c60cf88  5 years ago  148MB
student@ML-RefVm-691229:~/Desktop$
```

Figure 2.5

2. Define a Docker compose file which contains all necessary configurations and instructions for deploying and instantiating the above (step 1) Docker images (web server, locust, MongoDB, cAdvisor, Docker Swarm Visualizer);

A “*docker-compose.yml*” file is created that follows the web application topology shown below. Only one instance is created per image. These images are connected to each other through a bridged network. Swarm creates a default bridge network by default. A default configured network named web net is created in the YAML file. Required volumes and commands are added to ensure the service is running.



```
home > student > Desktop > docker-compose.yml
1 version: "3"
2 services:
3   webapplication:
4     image: nclcloudcomputing/javabenchmarkapp
5     ports:
6       - "8083:8080"
7   locust:
8     image: locustio/locust
9     deploy:
10    replicas: 1
11    restart_policy:
12      condition: on-failure
13    ports:
14      - '8089:8089'
15    volumes:
16      - ./mnt/locust
17    command: -f /mnt/locust/locustfile.py --host=http://10.210.60.18:8083
18   mongo:
19     image: mongo:4.4.6
20     environment:
21       MONGO_INITDB_ROOT_USERNAME: root
22       MONGO_INITDB_ROOT_PASSWORD: example
23     ports:
24       - '27017:27017'
25   visualizer:
26     image: dockersamples/visualizer:stable
27     ports:
28       - '5000:8080'
29     volumes:
30       - '/var/run/docker.sock:/var/run/docker.sock'
31     deploy:
32       placement:
33         constraints: [node.role == manager]
34   imagemonitor:
35     image: gcr.io/cadvisor/cadvisor
36     deploy:
37       replicas: 1
38       restart_policy:
39         condition: on-failure
40     ports:
41       - '8081:8080'
42     volumes:
43       - /:/rootfs:ro
44       - /var/run:/var/run:rw
45       - /sys:/sys:ro
46       - /var/lib/docker/:/var/lib/docker:ro
47
```

Figure 2.6

3. Create a web application topology, on a single Docker swarm node by using the above (step 2) Docker compose configurations.

Docker Swarm is a container orchestration tool that allows users to manage deployed containers. Swarm follows a master/slave architecture, with manager nodes managing other nodes. In this task, we will run a swarm on the manager node. Initialize the swarm with the command

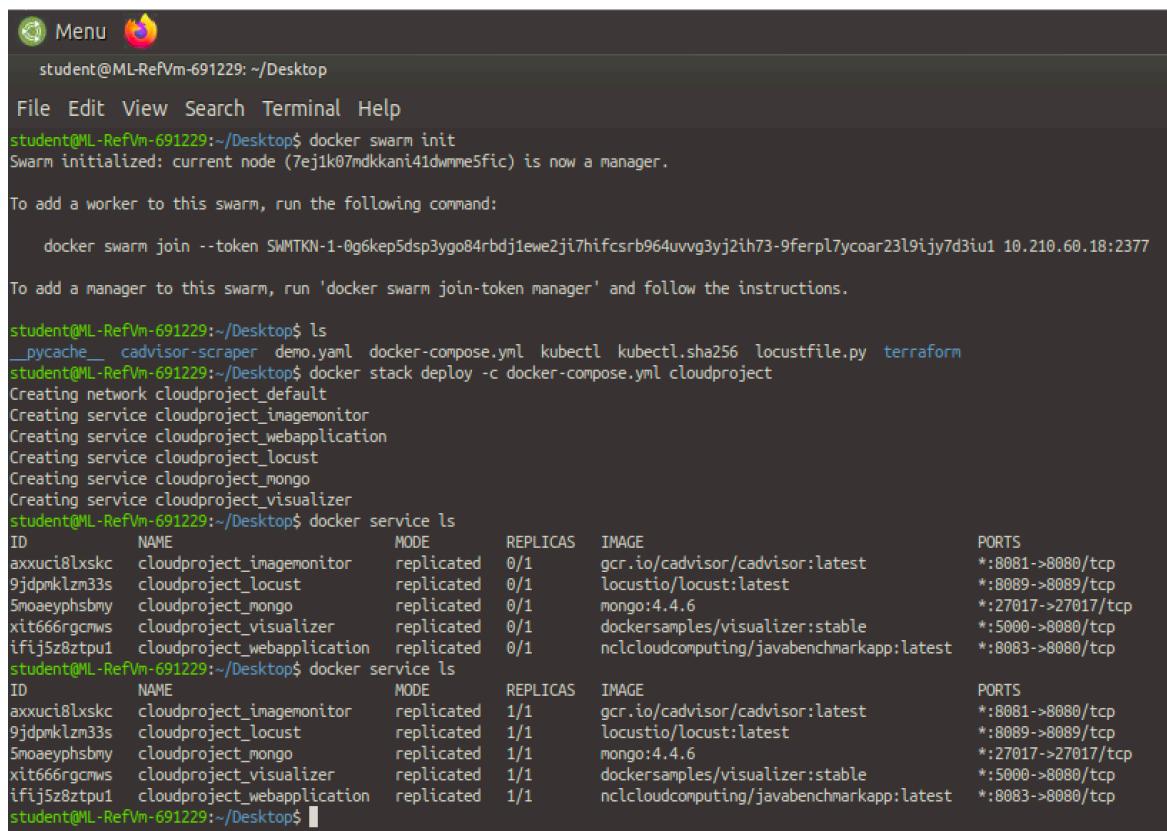
“docker swarm init”

Deploy a new docker stack by running the following command

“docker stack deploy -c docker-compose.yml”

The following command is used to view the running services in the swarm node

“docker service ls”



```

student@ML-RefVm-691229:~/Desktop$ docker swarm init
Swarm initialized: current node (7ej1k07ndkkani41dwmmefic) is now a manager.

To add a worker to this swarm, run the following command:

  docker swarm join --token SWMTKN-1-0g6kep5dsp3ygo84rbdj1ewe2ji7hifcsrb964uvvg3yj2ih73-9ferpl7ycoar23l9ijy7d3iu1 10.210.60.18:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

student@ML-RefVm-691229:~/Desktop$ ls
__pycache__  cAdvisor-scraper  demo.yaml  docker-compose.yml  kubectl  kubectl.sha256  locustfile.py  terraform
student@ML-RefVm-691229:~/Desktop$ docker stack deploy -c docker-compose.yml cloudproject
Creating network cloudproject_default
Creating service cloudproject_imagemonitor
Creating service cloudproject_webapplication
Creating service cloudproject_locust
Creating service cloudproject_mongo
Creating service cloudproject_visualizer
student@ML-RefVm-691229:~/Desktop$ docker service ls
ID          NAME      MODE      REPLICAS  IMAGE
axxuci8lxskc  cloudproject_imagemonitor  replicated  0/1      gcr.io/cadvisor/cadvisor:latest      *:8081->8080/tcp
9jdpmklzn33s  cloudproject_locust     replicated  0/1      locustio/locust:latest                *:8089->8089/tcp
5moaeyphsbmy  cloudproject_mongo     replicated  0/1      mongo:4.4.6                         *:27017->27017/tcp
xit666rgcmws  cloudproject_visualizer  replicated  0/1      dockersamples/visualizer:stable        *:5000->8080/tcp
ifij5z8ztpu1  cloudproject_webapplication  replicated  0/1      nclcloudcomputing/javabenchmarkapp:latest  *:8083->8080/tcp
student@ML-RefVm-691229:~/Desktop$ docker service ls
ID          NAME      MODE      REPLICAS  IMAGE
axxuci8lxskc  cloudproject_imagemonitor  replicated  1/1      gcr.io/cadvisor/cadvisor:latest      *:8081->8080/tcp
9jdpmklzn33s  cloudproject_locust     replicated  1/1      locustio/locust:latest                *:8089->8089/tcp
5moaeyphsbmy  cloudproject_mongo     replicated  1/1      mongo:4.4.6                         *:27017->27017/tcp
xit666rgcmws  cloudproject_visualizer  replicated  1/1      dockersamples/visualizer:stable        *:5000->8080/tcp
ifij5z8ztpu1  cloudproject_webapplication  replicated  1/1      nclcloudcomputing/javabenchmarkapp:latest  *:8083->8080/tcp
student@ML-RefVm-691229:~/Desktop$ 
```

Figure 2.7

Figure 2.8 shows that the “mongo” image is running on port 27017, using the web browser “<http://localhost:27017>”

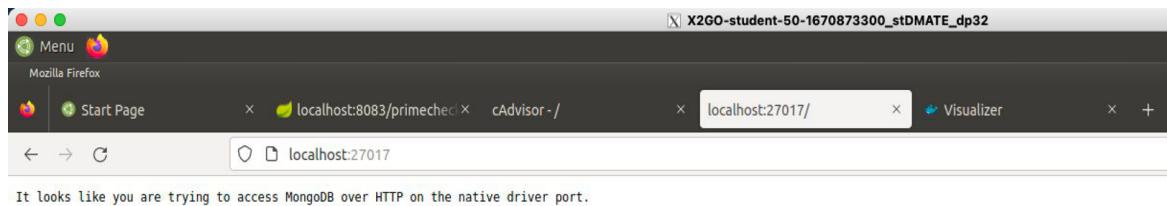


Figure 2.8

Figure 2.9 shows that the “`gcr.io/cadvisor/cadvisor:v0.45.0`” image is running on port 8081, using the web browser “`http://localhost:8081`”

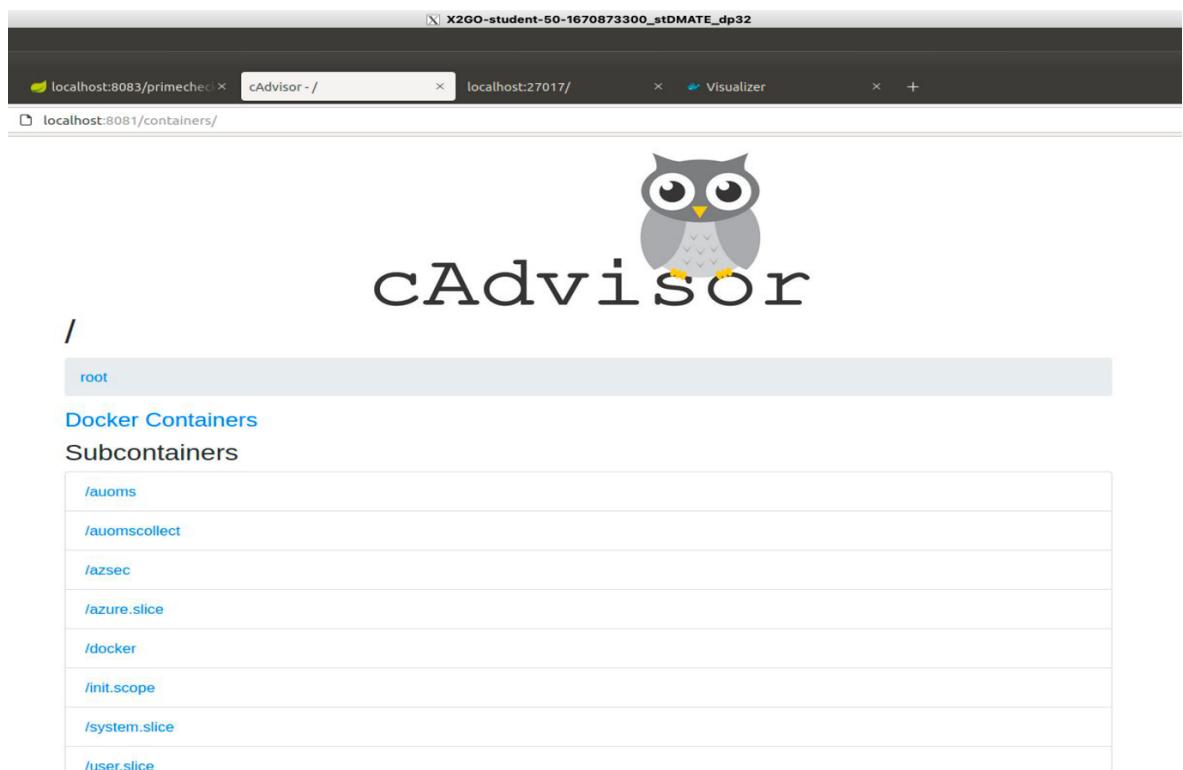


Figure 2.9

The screenshot shows the cAdvisor Docker Containers interface. At the top, there's a navigation bar with tabs for 'localhost:8083/primecheck' (highlighted), 'cAdvisor - Docker Containers' (active), 'localhost:27017/' (disabled), and 'Visualizer' (disabled). Below the navigation bar is a search bar with the placeholder 'localhost:8081/docker/'. The main content area features a large owl logo and the text 'cAdvisor Docker Containers'. A 'Docker Containers' tab is selected. Under 'Subcontainers', there is a list of five entries, each with a URL and a short description:

- cloudproject_imagemonitor.1.ydltpaswiek89ydimhiazv... (/docker/b92c5cf8c6a303a7db30e73b6fc527f3c0077daa1dff5fa013144375c2b79e8)
- cloudproject_mongo.1.sjhkisvqdjbd3p7eksu49h8wf (/docker/bbbe56669e3aeac53bc06aa642f2dabfa503922ceb3edd4e4256d8f257f7a3fd)
- cloudproject_locust.1.jp45wxxyiteg3curys6tcrdj4 (/docker/75e7c6cb6f2aba03518909774e8c3328dbb17e40a1df4150cb924420a56a230f)
- cloudproject_webapplication.1.4sgg1mb5udjc1a1t79ig... (/docker/da5f902b6dd63f3f353cb33ac3f1c4485ftbd1cc191b395aefb994a0169979f3b)
- cloudproject_visualizer.1.bthxty5r5qjcx8tycv6xls03... (/docker/2847751b2ffd7ae272c4ef457acbb5242b3753fc2547b83a63ed6a1377e46ba0)

Below this is a 'Driver Status' section with three items:

- Docker Version 20.10.18
- Docker API Version 1.41
- Kernel Version 5.15.0-1023-azure

Figure 2.10

This screenshot shows the same cAdvisor interface as Figure 2.10, but the 'Docker Containers' tab is not selected. Instead, the 'Driver Status' tab is active. It displays the same three items: Docker Version 20.10.18, Docker API Version 1.41, and Kernel Version 5.15.0-1023-azure.

Below the driver status is a 'Storage' section with the following details:

- Driver: overlay2
- userxattr: false
- Backing Filesystem: extfs
- Supports d_type: true
- Native Overlay Diff: false

At the bottom of the interface, there is a 'Images' section.

Figure 2.11

Figure 2.12 shows that the “dockersamples/visualizer” image is running on port 5000, using the web browser “<http://localhost:5000>”

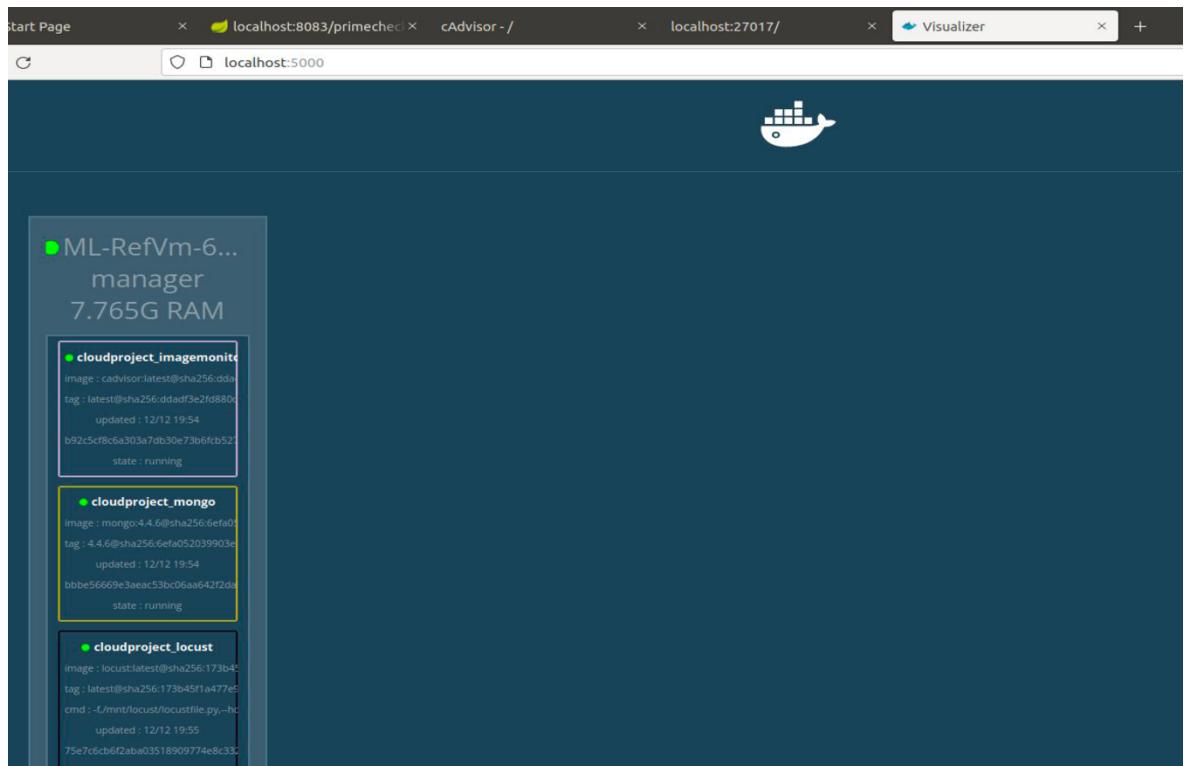


Figure 2.12

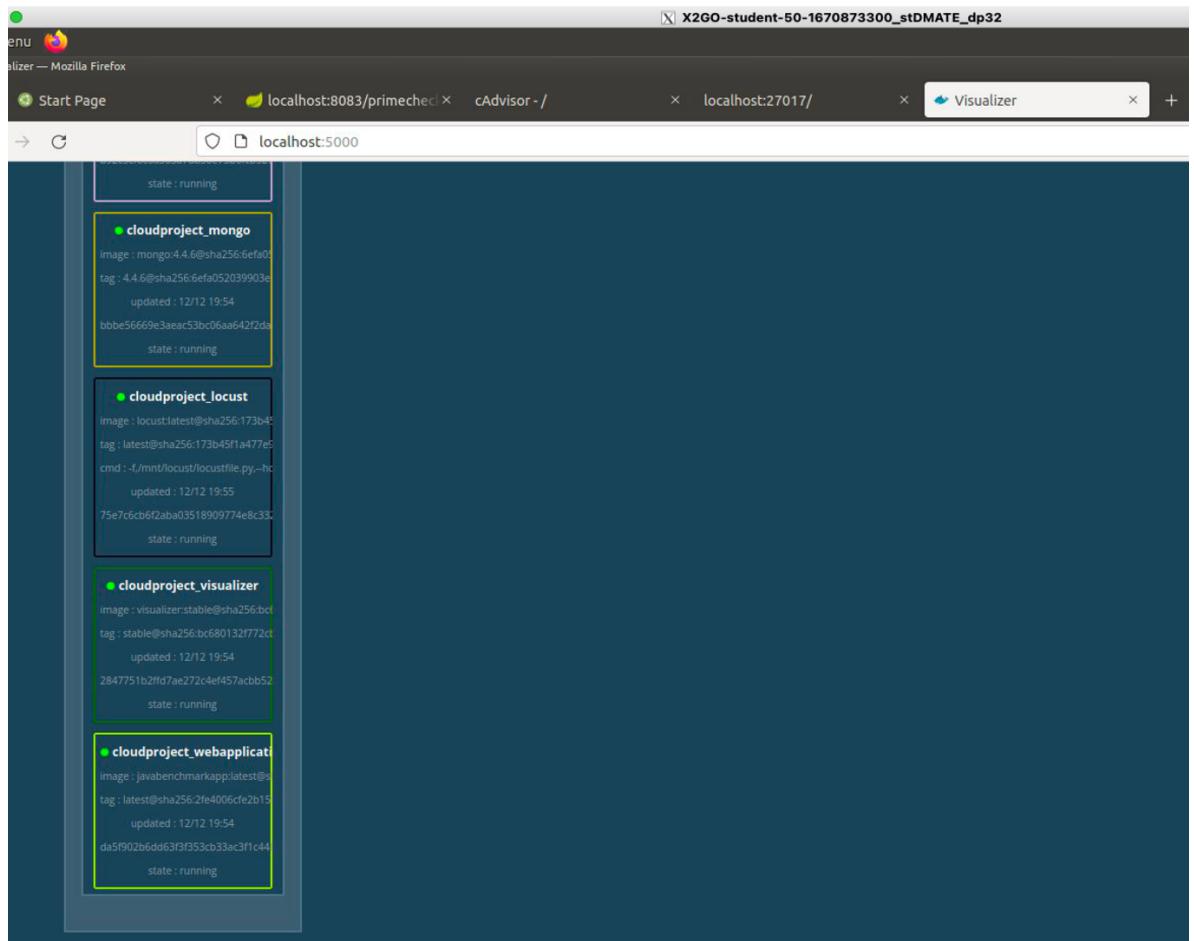


Figure 2.13

Locust: Use locust (locustio/locust) to generate load. In *Figure 2.14* We create locusfile.py to generate the load of the Benchmark app API (<http://10.210.60.18:8083>).

```

Main.java 1, M locustfile.py 1 X dockerfile provider
home > student > Desktop > locustfile.py > ...
1  from locust import HttpUser, task
2
3  class HelloWorldUser(HttpUser):
4      @task
5      def hello_world(self):
6          self.client.get("/primecheck")

```

Figure 2.14

Figure 2.15 the Locust UI, enter the number of concurrent users you want the system to spawn. It also includes the spawning rate. The host is automatically filled from the locust file.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)
GET	/primecheck	67	0	2900	3200
Aggregated					
		67	0	2900	3200

Start new load test

Number of users (peak concurrency)
1

Spawn rate (users started/second)
1

Host (e.g. http://www.example.com)
http://10.210.60.18:8083

Advanced options

Start swarming

Figure 2.15

Click Start Swarming to generate load.

The screenshots below are from running the load generator for a few minutes.

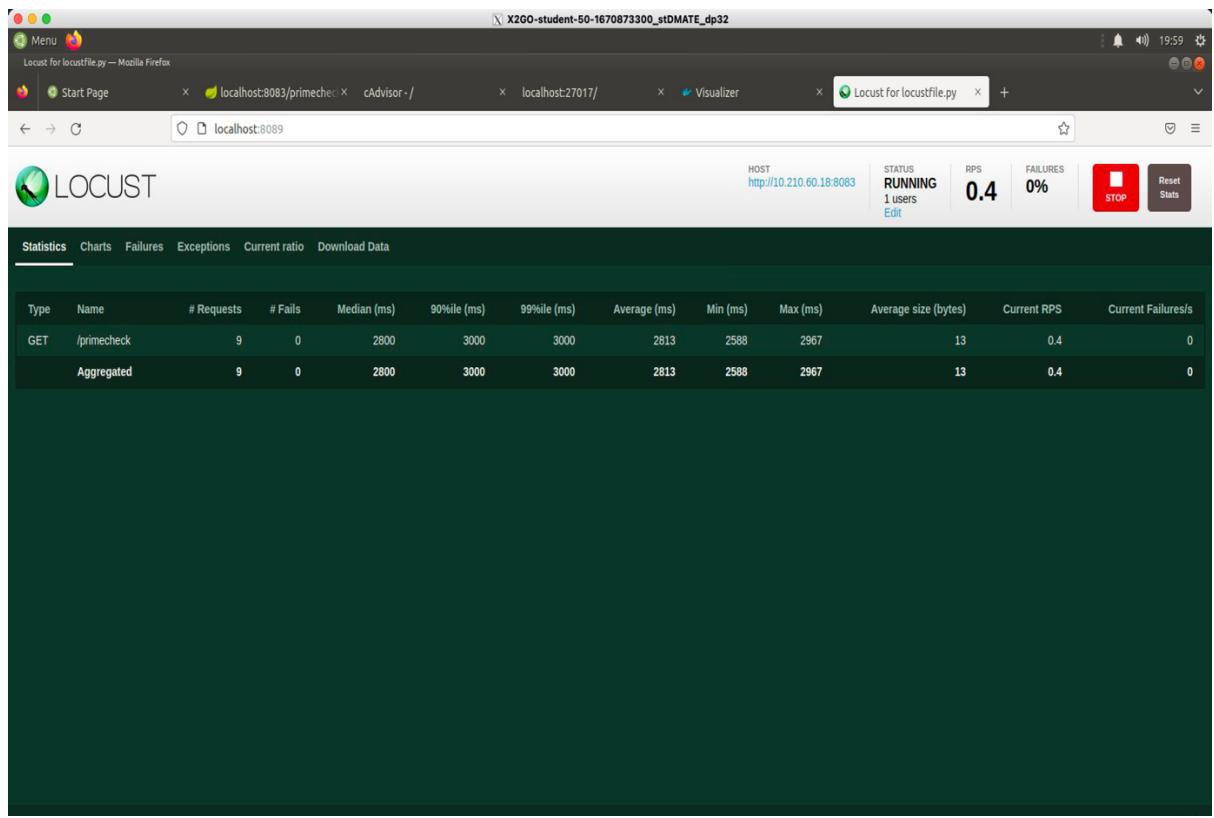


Figure 2.16

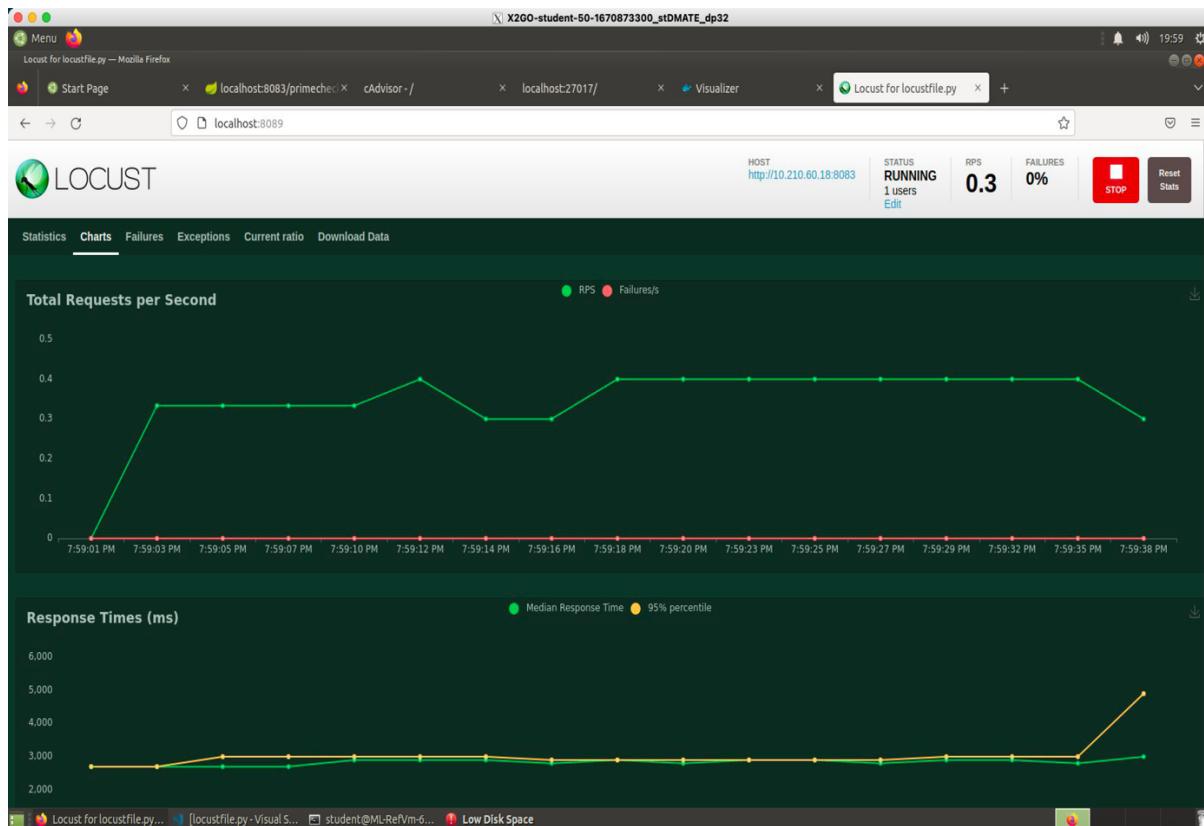


Figure 2.17

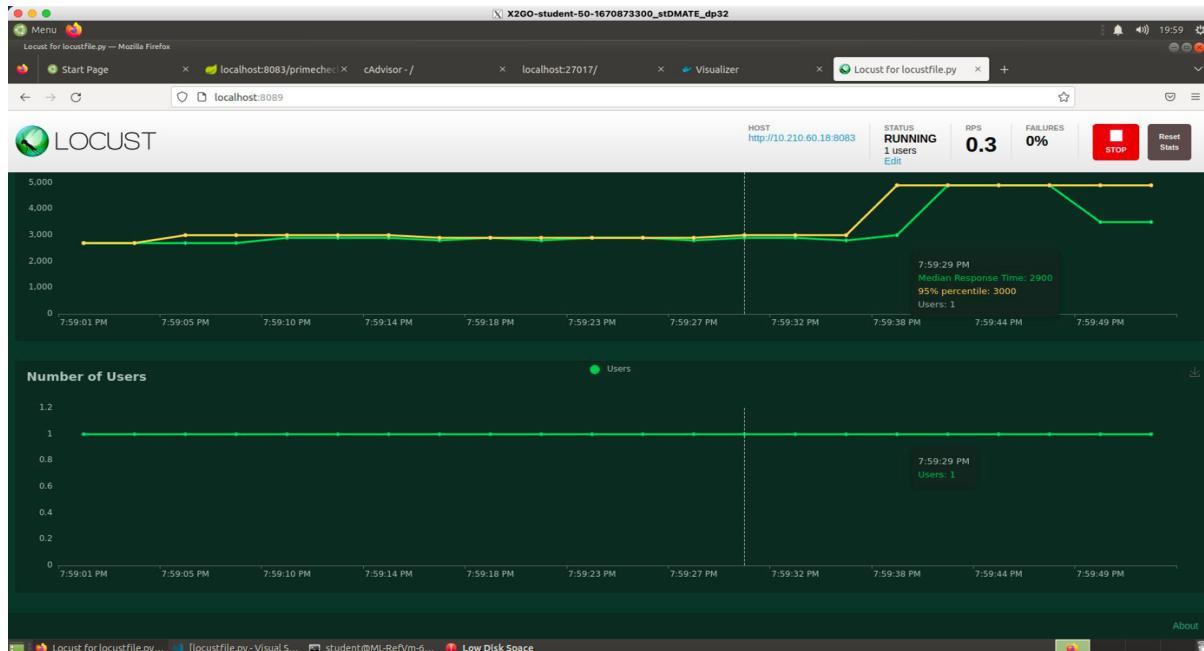


Figure 2.18

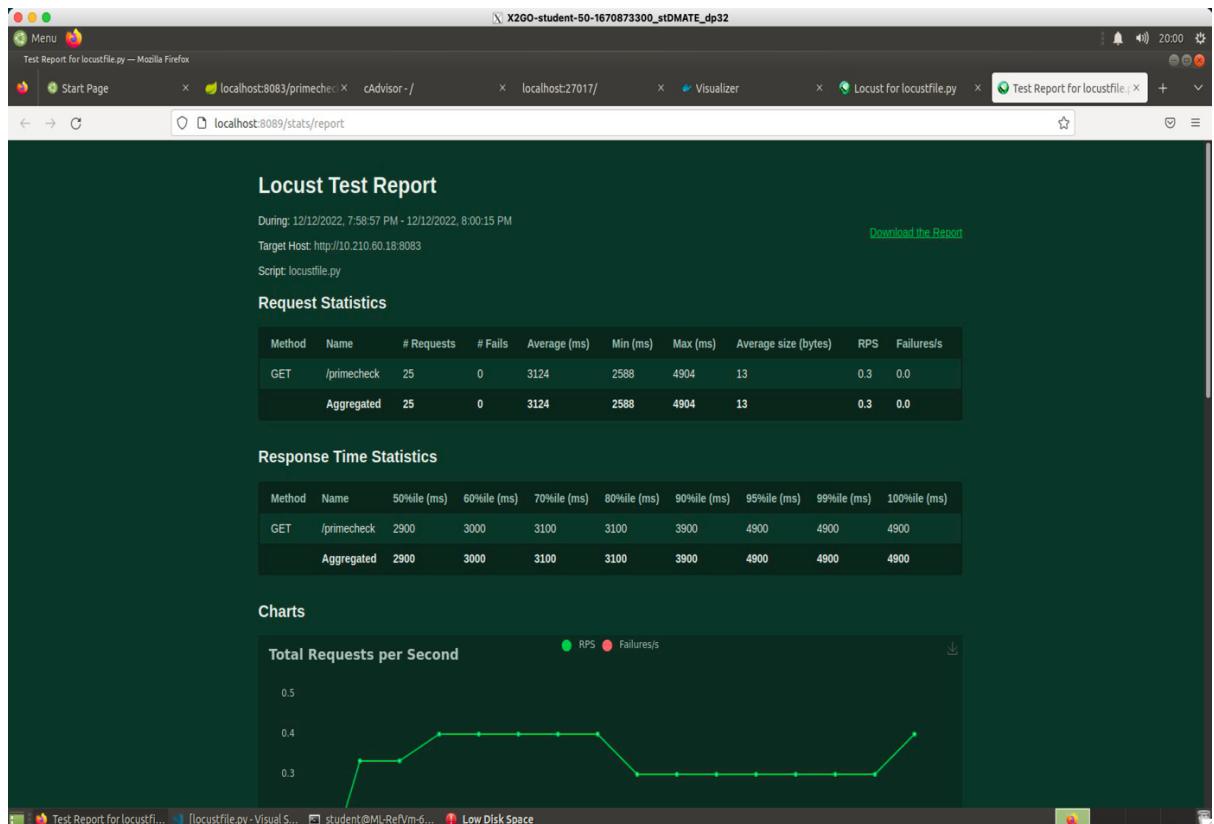


Figure 2.19

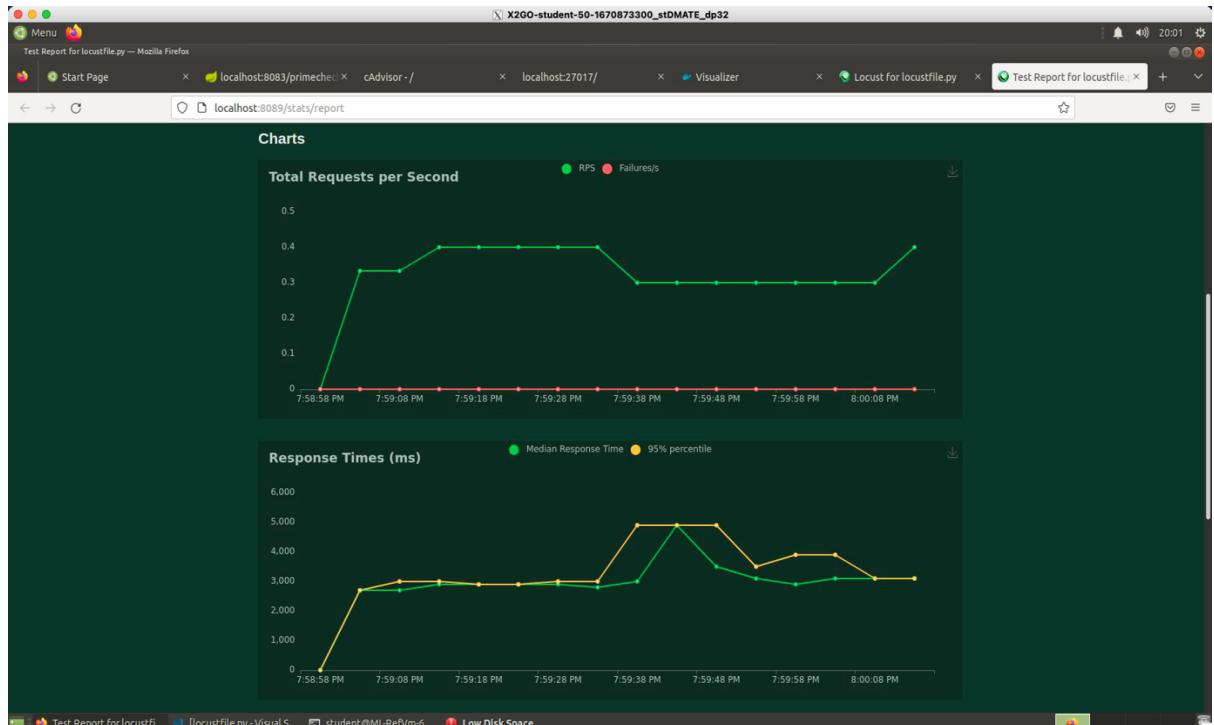


Figure 2.20

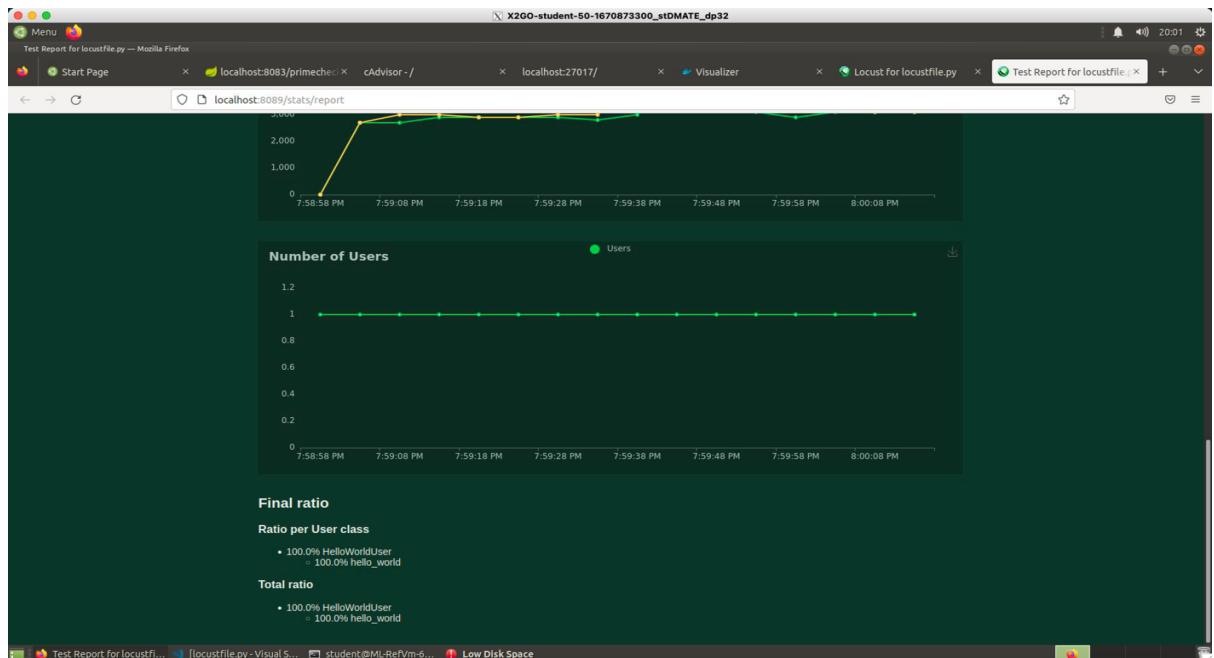


Figure 2.21

Stop docker swarm with the following command
“*docker swarm leave --force*”

Task 3: Build your own Docker image and push it to the Docker Hub

Task Objective: To understand how to build a Docker image from scratch and how to push it to Docker Hub so that other developers can view, use and extend the same.

1. You need to build a Docker image of a Java program. In terms of the application logic, the program implements features including how to extract the monitoring information from the Docker cAdvisor container (a running instance of `gcr.io/cadvisor/cadvisor` image) and how to insert this monitoring information into a MongoDB Docker container (a running instance of `mongo` image). The cAdvisor is able to monitor run-time performance (CPU load, network I/O and memory usage) of all instances of Docker images running within a Docker Swarm.

The below figure shows the results detected by the cAdvisor user interface.

The screenshots are after running the load generator for a few minutes

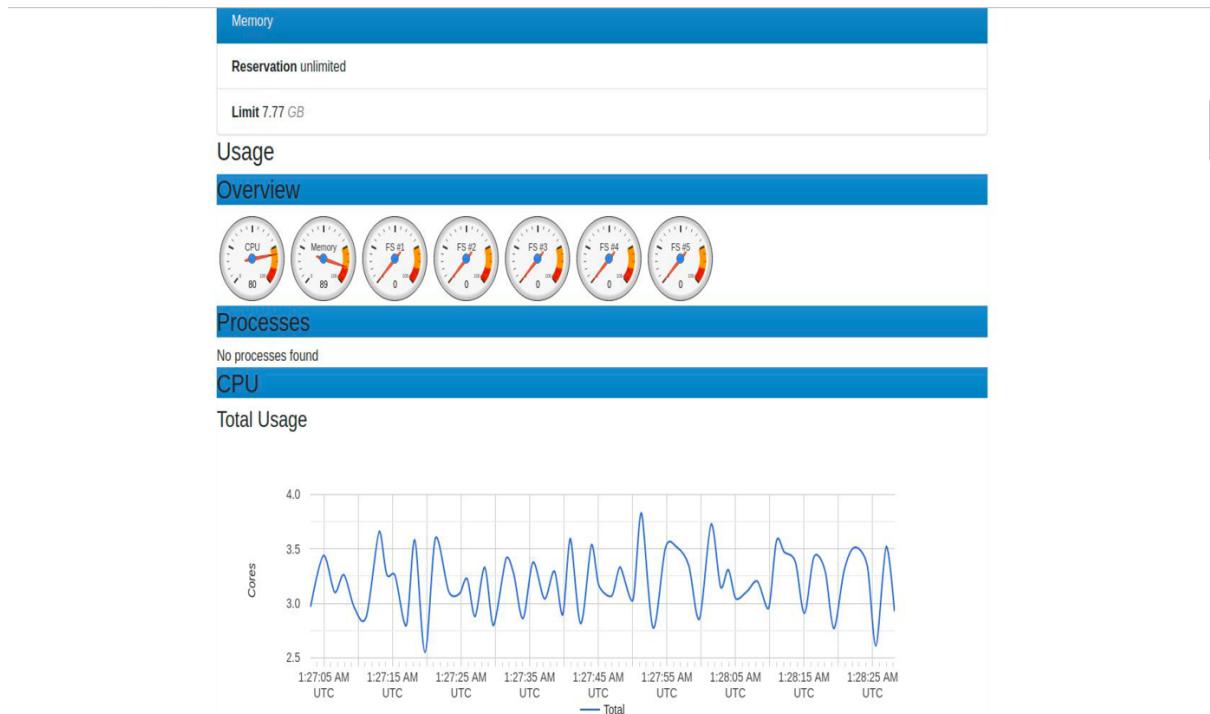


Figure 3.1

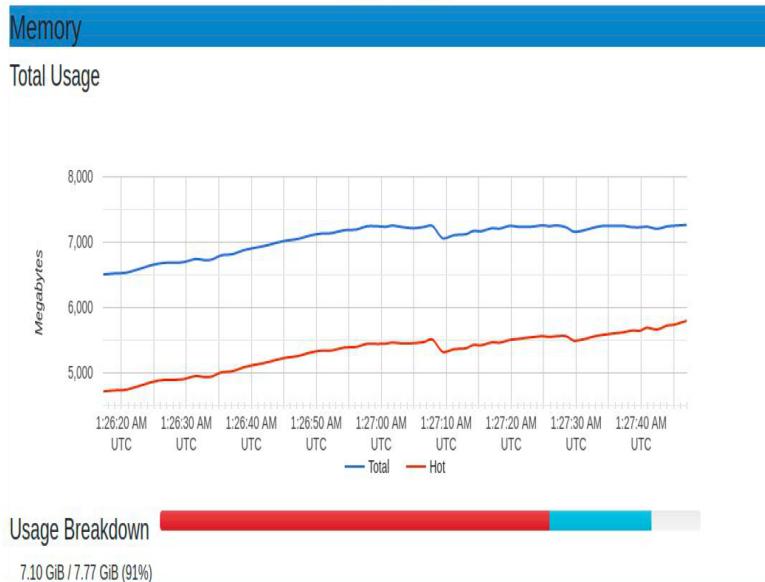


Figure 3.2

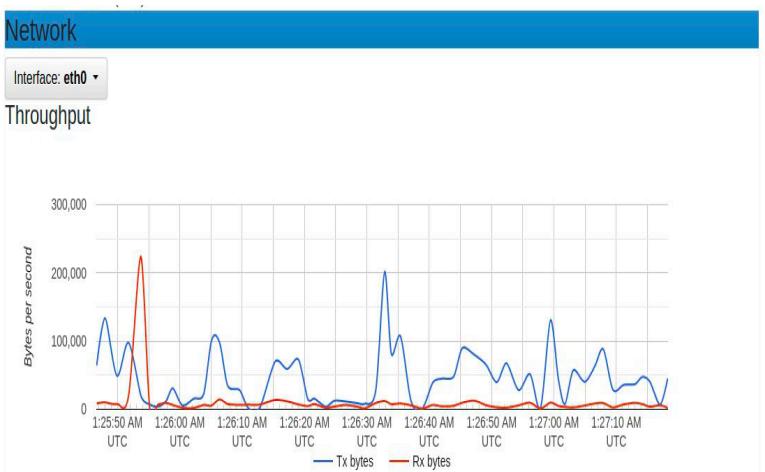


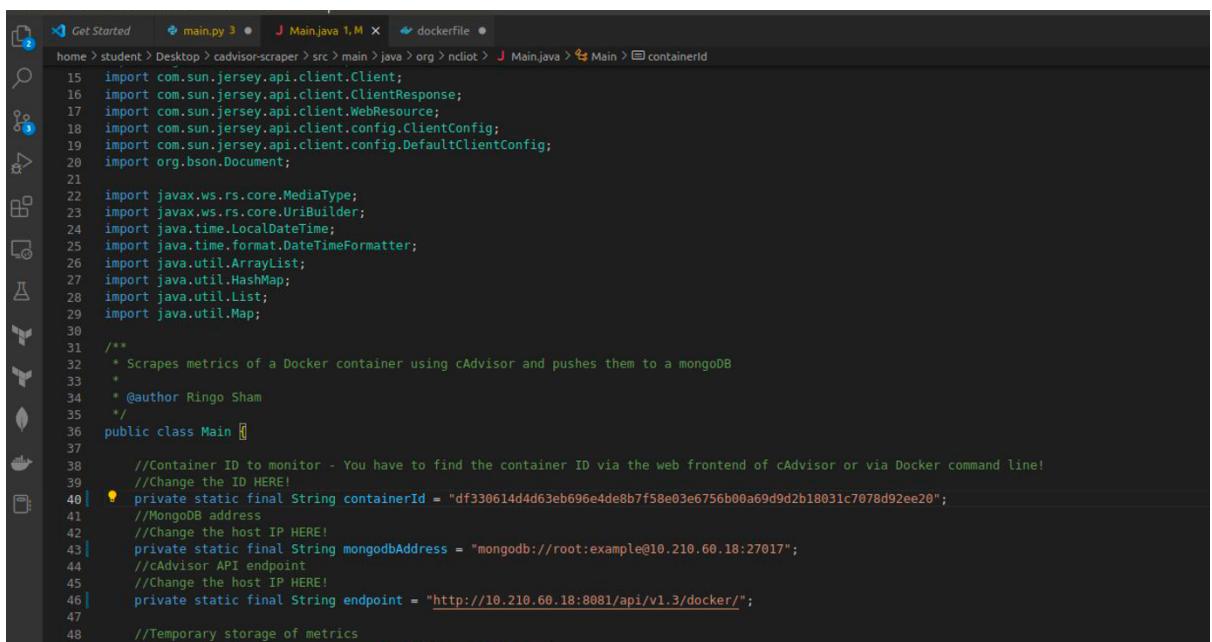
Figure 3.3

We use cAdvisor to monitor benchmark app containers where data is consumed by the cAdvisor scraper application using the APIs provided by cAdvisor.

2. Edit the provided Java code using the IntelliJ editor (already installed in the provided virtual image). The instructions in Java code that need editing include *container ID*, *MongoDB Address*, and *endpoint* address.

The application sends requests to cAdvisor, which monitors web applications. cAdvisor's API is used to extract and process the data which is later transferred to mongo DB. Requests are sent every minute because cAdvisor returns metrics at the last minute.

In the main.java file, make the following changes and build the image. Change the container field, which is the container ID of the Java web application that needs to be monitored. Also, change the MongoDB address which is the mongo DB address and the endpoint which is the cAdvisors endpoint.



```

15 import com.sun.jersey.api.client.Client;
16 import com.sun.jersey.api.client.ClientResponse;
17 import com.sun.jersey.api.client.WebResource;
18 import com.sun.jersey.api.client.config.ClientConfig;
19 import com.sun.jersey.api.client.config.DefaultClientConfig;
20 import org.json.Document;
21
22 import javax.ws.rs.core.MediaType;
23 import javax.ws.rs.core.UriBuilder;
24 import java.time.LocalDateTime;
25 import java.time.format.DateTimeFormatter;
26 import java.util.ArrayList;
27 import java.util.HashMap;
28 import java.util.List;
29 import java.util.Map;
30
31 /**
32  * Scrapes metrics of a Docker container using cAdvisor and pushes them to a mongoDB
33  *
34  * @author Ringo Sham
35  */
36 public class Main {
37
38     //Container ID to monitor - You have to find the container ID via the web frontend of cAdvisor or via Docker command line!
39     //Change the ID HERE!
40     private static final String containerId = "df330614d4d63eb696e4de8b7f58e03e6756b00a69d9d2b18031c7078d92ee20";
41     //MongoDB address
42     //Change the host IP HERE!
43     private static final String mongoDBAddress = "mongodb://root@example@10.210.60.18:27017";
44     //cAdvisor API endpoint
45     //Change the host IP HERE!
46     private static final String endpoint = "http://10.210.60.18:8081/api/v1.3/docker/";
47
48     //Temporary storage of metrics

```

Figure 3.4

Figure 3.2 shows that the java web application address

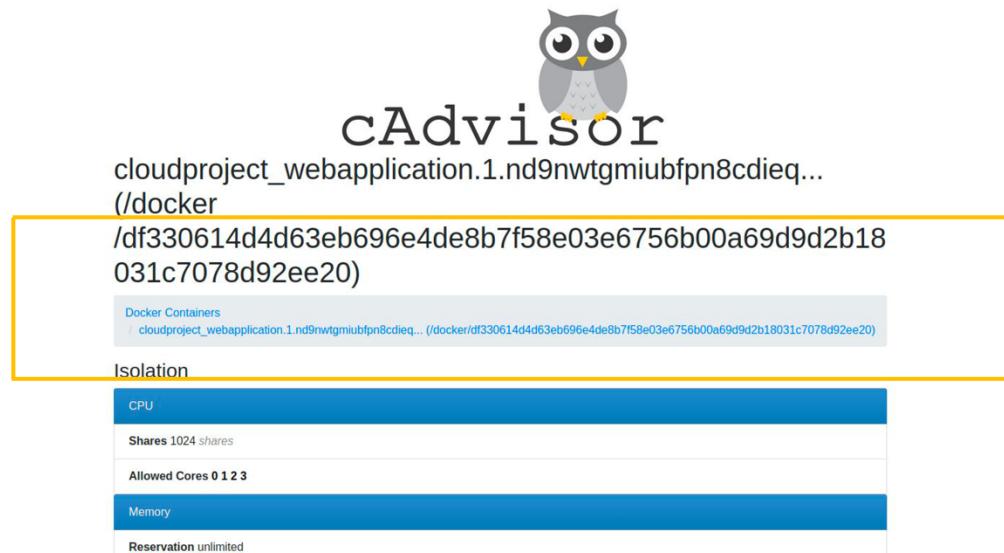


Figure 3.5

```

Get Started   main.py 3  Main.java 1,M  dockerfile
home > student > Desktop > cadvisor-scraper > src > main > java > org > ncilot > J Main.java > ↵ Main > containerId
15 import com.sun.jersey.api.client.Client;
16 import com.sun.jersey.api.client.ClientResponse;
17 import com.sun.jersey.api.client.WebResource;
18 import com.sun.jersey.api.client.config.ClientConfig;
19 import com.sun.jersey.api.client.config.DefaultClientConfig;
20 import org.json.Document;
21
22 import javax.ws.rs.core.MediaType;
23 import javax.ws.rs.core.UriBuilder;
24 import java.time.LocalDateTime;
25 import java.time.format.DateTimeFormatter;
26 import java.util.ArrayList;
27 import java.util.HashMap;
28 import java.util.List;
29 import java.util.Map;
30
31 /**
32 * Scraps metrics of a Docker container using cAdvisor and pushes them to a mongoDB
33 *
34 * @author Ringo Sham
35 */
36 public class Main {
37
38     //Container ID to monitor - You have to find the container ID via the web frontend of cAdvisor or via Docker command line!
39     //Change the ID HERE!
40     private static final String containerId = "df330614d4d63eb696e4de8b7f58e03e6756b00a69d9d2b18031c7078d92ee20";
41     //MongoDB address
42     //Change the host IP HERE!
43     private static final String mongoDBAddress = "mongodb://root:example@10.210.60.18:27017";
44     //cAdvisor API endpoint
45     //Change the host IP HERE!
46     private static final String endpoint = "http://10.210.60.18:8081/api/v1.3/docker/";
47
48     //Temporary storage of metrics
        ...
    }

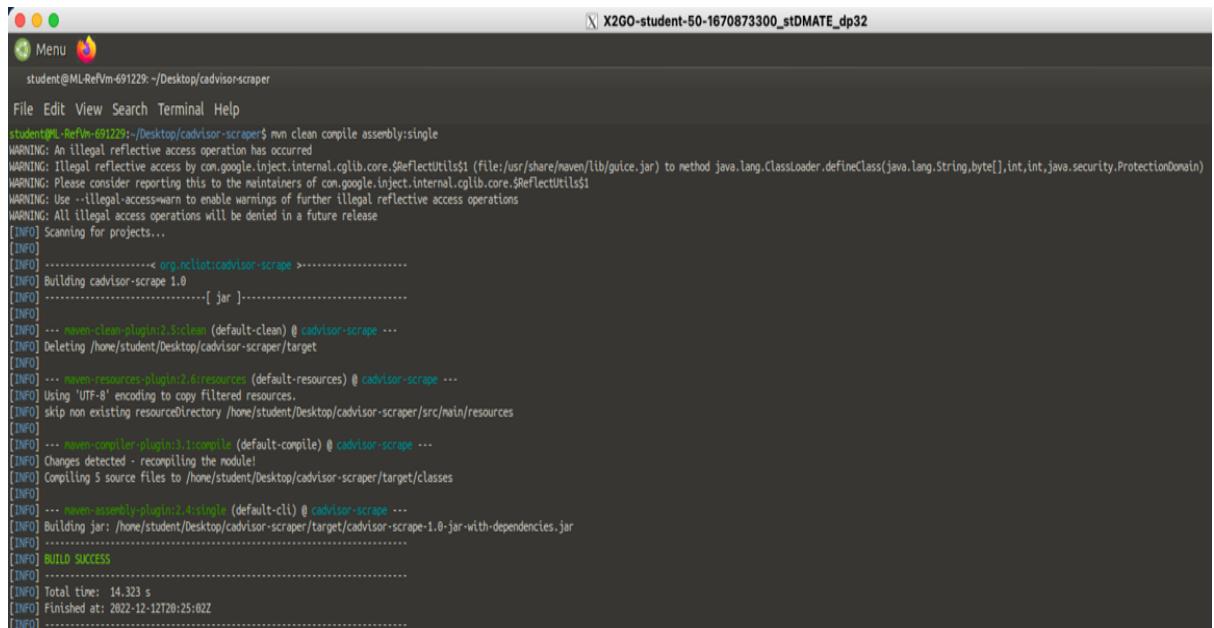
```

A yellow box highlights the line of code defining the container ID: `private static final String containerId = "df330614d4d63eb696e4de8b7f58e03e6756b00a69d9d2b18031c7078d92ee20";`.

Figure 3.6

Figure 3.4 shows that the maven is built in the docker CLI java web application and sends requests to cAdvisor, which monitors web applications. with the following cmd, we have the dependencies under the fully built jar.

“mvn clean compile assembly:single”



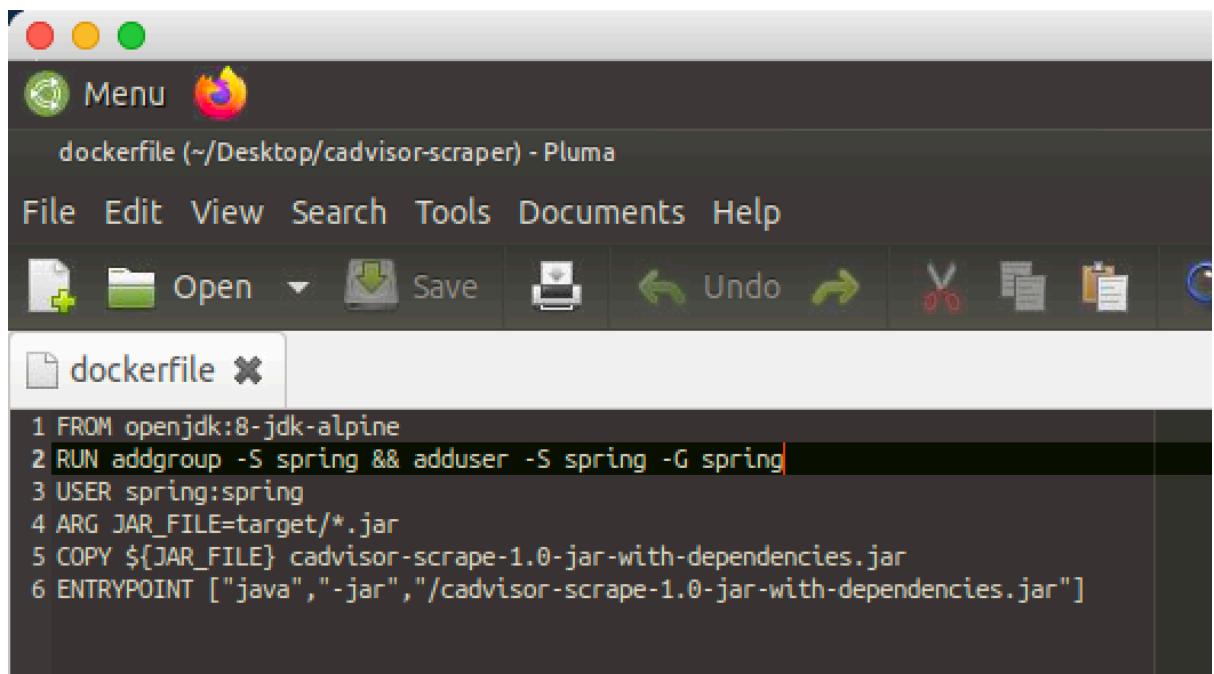
```

student@X2GO-student-50-1670873300_stDMATE_dp32: ~
student@X2GO-student-50-1670873300_stDMATE_dp32: ~/Desktop/cadvisor-scraper
File Edit View Search Terminal Help
student@X2GO-student-50-1670873300_stDMATE_dp32: ~/Desktop/cadvisor-scraper$ mvn clean compile assembly:single
WARNING: Illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$ReflectUtils$1 (file:/usr/share/nexus/lib/guice.jar) to method java.lang.ClassLoader.defineClass(java.lang.String,byte[],int,int,java.security.ProtectionDomain)
WARNING: Please consider reporting this to the maintainers of com.google.inject.internal.cglib.core.$ReflectUtils$1
WARNING: Use -Dillegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO]
[INFO] .....< org.ncliot:cadvisor-scraper >.....
[INFO] Building cadvisor-scraper 1.0
[INFO] .....[ Jar ].....
[INFO]
[INFO] ... maven-clean-plugin:2.5:clean (default-clean) @ cadvisor-scraper ...
[INFO] Deleting /home/student/Desktop/cadvisor-scraper/target
[INFO]
[INFO] ... maven-resources-plugin:2.6:resources (default-resources) @ cadvisor-scraper ...
[INFO] Using 'UTF-8' encoding to copy filtered resources
[INFO] skip non existing resource directory /home/student/Desktop/cadvisor-scraper/src/main/resources
[INFO]
[INFO] ... maven-compiler-plugin:1.1:compile (default-compile) @ cadvisor-scraper ...
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 5 source files to /home/student/Desktop/cadvisor-scraper/target/classes
[INFO]
[INFO] ... maven-assembly-plugin:2.4:single (default-cli) @ cadvisor-scraper ...
[INFO] Building jar: /home/student/Desktop/cadvisor-scraper/target/cadvisor-scraper-1.0-jar-with-dependencies.jar
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 14.323 s
[INFO] Finished at: 2022-12-12T20:25:02Z
[INFO]

```

Figure 3.7

Figure 3.5 shows that the docker file is created for the java application with the c-Advisor scraper dependencies mentioned on it



```

dockerfile (~~/Desktop/cadvisor-scraper) - Pluma
File Edit View Search Tools Documents Help
Open Save Undo Redo
dockefile * x
1 FROM openjdk:8-jdk-alpine
2 RUN addgroup -S spring && adduser -S spring -G spring
3 USER spring:spring
4 ARG JAR_FILE=target/*.jar
5 COPY ${JAR_FILE} cadvisor-scraper-1.0-jar-with-dependencies.jar
6 ENTRYPOINT ["java", "-jar", "/cadvisor-scraper-1.0-jar-with-dependencies.jar"]

```

Figure 3.8

Figure 3.6 shows to build the docker image using the following command

“docker build -t cadvisor-scraper .”

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker build -t cadvisor-scraper .
Sending build context to Docker daemon 3.709MB
Step 1/6 : FROM openjdk:8-jdk-alpine
--> a3562aa0b991
Step 2/6 : RUN addgroup -S spring && adduser -S spring -G spring
--> Using cache
--> c24531e9c807
Step 3/6 : USER spring:spring
--> Using cache
--> e0ed6181388a
Step 4/6 : ARG JAR_FILE=target/*.jar
--> Using cache
--> 139d65b66269
Step 5/6 : COPY ${JAR_FILE} cAdvisor-scrape-1.0-jar-with-dependencies.jar
--> 971e5868a4e3
Step 6/6 : ENTRYPOINT ["java", "-jar", "/cAdvisor-scrape-1.0-jar-with-dependencies.jar"]
--> Running in 5c91ea0642af
Removing intermediate container 5c91ea0642af
--> 85985b7c5cd2
Successfully built 85985b7c5cd2
Successfully tagged cAdvisor-scraper:latest
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$
```

Figure 3.9

After building the image successfully as shown in figure 3.6 using the following command we can view the built image in the docker CLI

“docker images”

The following command is used to tag and rename the image name as shown in Figure 3.7

“docker tag --imageid --name”

```

student@ML-RefVm-691229:~/Desktop/cadvisor-scraping
[INFO] Building jar: /home/student/Desktop/cadvisor-scraping/target/cadvisor-scrape-1.0-jar-with-dependencies.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 14.323 s
[INFO] Finished at: 2022-12-12T20:25:02Z
[INFO] -----
student@ML-RefVm-691229:~/Desktop/cadvisor-scraping$ docker build -t cAdvisor-scraping .
Sending build context to Docker daemon 3.709MB
Step 1/6 : FROM openjdk:8-jdk-alpine
--> a3562aa0b991
Step 2/6 : RUN addgroup -S spring && adduser -S spring -G spring
--> Using cache
--> c24531e9c807
Step 3/6 : USER spring:spring
--> Using cache
--> e0ed6181388a
Step 4/6 : ARG JAR_FILE=target/*.jar
--> Using cache
--> 139d65b66260
Step 5/6 : COPY ${JAR_FILE} cadvisor-scrape-1.0-jar-with-dependencies.jar
--> 971e5868a4e3
Step 6/6 : ENTRYPOINT ["java","-jar","/cAdvisor-scrape-1.0-jar-with-dependencies.jar"]
--> Running in 5c91ea0642af
Removing intermediate container 5c91ea0642af
--> 85985b7c5cd2
Successfully built 85985b7c5cd2
Successfully tagged cAdvisor-scraping:latest
student@ML-RefVm-691229:~/Desktop/cadvisor-scraping$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
cAdvisor-scraping   latest   85985b7c5cd2  15 seconds ago  108MB
locustio/locust     <none>   d2bdac521bd1  3 days ago   263MB
<none>              <none>   c0aec8a7d465  9 days ago   105MB
locustio/locust     <none>   2054cc6b50a3  11 days ago  263MB
mongo               latest   2dd27bb6d3e6  3 weeks ago  695MB
locustio/locust     latest   042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor   v0.45.0   3f3e5f568a6d  4 months ago  80.6MB
mongo               <none>   61ea24dc52c6  17 months ago 423MB
dockersamples/visualizer  latest   43ce62428b8c  18 months ago 185MB
gcr.io/cadvisor/cadvisor   <none>   68c29634fe49  2 years ago   163MB
openjdk              8-jdk-alpine  a3562aa0b991  3 years ago   105MB
nclcloudcomputing/javabenchmarkapp  latest   941ad5dfe11d  5 years ago   116MB
dockersamples/visualizer  <none>   8dbf7c60cf88  5 years ago   148MB
student@ML-RefVm-691229:~/Desktop/cadvisor-scraping$ docker tag 85985b7c5cd2 bkailash29/cAdvisor-scraping
student@ML-RefVm-691229:~/Desktop/cadvisor-scraping$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
bkailash29/cAdvisor-scraping  latest   85985b7c5cd2  57 seconds ago  108MB
cAdvisor-scraping   latest   85985b7c5cd2  57 seconds ago  108MB
locustio/locust     <none>   d2bdac521bd1  3 days ago   263MB
<none>              <none>   c0aec8a7d465  9 days ago   105MB
locustio/locust     <none>   2054cc6b50a3  11 days ago  263MB
mongo               latest   2dd27bb6d3e6  3 weeks ago  695MB
locustio/locust     latest   042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor   v0.45.0   3f3e5f568a6d  4 months ago  80.6MB
mongo               <none>   61ea24dc52c6  17 months ago 423MB
dockersamples/visualizer  latest   43ce62428b8c  18 months ago 185MB
gcr.io/cadvisor/cadvisor   <none>   68c29634fe49  2 years ago   163MB
openjdk              8-jdk-alpine  a3562aa0b991  3 years ago   105MB
nclcloudcomputing/javabenchmarkapp  latest   941ad5dfe11d  5 years ago   116MB
dockersamples/visualizer  <none>   8dbf7c60cf88  5 years ago   148MB
student@ML-RefVm-691229:~/Desktop/cadvisor-scraping$ 
```

[Mozilla Firefox] [Main.java - Visual Studio... Low Disk Space] [cAdvisor-]

Figure 3.10

Figure 3.8 shows the image after renaming with the name
“*bkailash29/cadvisor-scraping*”

```

student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker tag 85985b7c5cd2 bkailash29/cadvisor-scraper
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
bkailash29/cadvisor-scraper    latest    85985b7c5cd2  57 seconds ago  108MB
cadvisor-scraper      latest    85985b7c5cd2  57 seconds ago  108MB
locustio/locust        <none>   d2bdac521bd1  3 days ago   263MB
<none>              <none>   c0aec8a7d465  9 days ago   105MB
locustio/locust        <none>   2054ceb50a3   11 days ago  263MB
mongo                latest    2dd27bb6d3e6   3 weeks ago  695MB
locustio/locust        latest    042b162d1b70  6 weeks ago  260MB
gcr.io/cadvisor/cadvisor  v0.45.0  3f3e5f568a6d   4 months ago  80.6MB
mongo                <none>   61ea24dc52c6   17 months ago 423MB
dockersamples/visualizer  latest    43ce62428b8c   18 months ago 185MB
gcr.io/cadvisor/cadvisor  <none>   68c29634fe49  2 years ago  163MB
openjdk               8-jdk-alpine  a3562aa0b991  3 years ago  105MB
nclcloudcomputing/javabenchmarkapp  latest    941ad5dfe11d  5 years ago  116MB
dockersamples/visualizer  <none>   8dbf7c60cf88  5 years ago  148MB

```

Figure 3.11

3. After code modification, pack the program as a standalone Docker image and push it to the Docker Hub. Name the image as cAdvisor-scraper.

To push an image to the docker hub, first log into docker with the following command:

“docker login --username=username -p password”

Use the following command to push the image to Docker Hub.

“docker push bkailash29/cadvisor-scraper”

Use the following command to push the image to Docker Hub.

“docker pull bkailash29/cadvisor-scraper”

Use the following command to run the image to Docker Hub.

“docker run bkailash29/cadvisor-scraper”

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/student/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker push bkailash29/cadvisor-scraper
Using default tag: latest
The push refers to repository [docker.io/bkailash29/cadvisor-scraper]
ed0bb6839e4f: Layer already exists
997b5e96800: Layer already exists
ceaf9e1ebef5: Layer already exists
9b9b7f3d56a0: Layer already exists
f1b5933fe4b5: Layer already exists
latest: digest: sha256:774e0c1a421e14c7702399b1041bcfbaf5ab0f54fc154091666e7a9585dccc1 size: 1366
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker pull bkailash29/cadvisor-scraper
Using default tag: latest
latest: Pulling from bkailash29/cadvisor-scraper
Digest: sha256:774e0c1a421e14c7702399b1041bcfbaf5ab0f54fc154091666e7a9585dccc1
Status: Image is up to date for bkailash29/cadvisor-scraper:latest
docker.io/bkailash29/cadvisor-scraper:latest
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker run bkailash29/cadvisor-scraper
Dec 12, 2022 8:34:50 PM com.mongodb.diagnostics.logging.Loggers shouldUseSLF4J
WARNING: SLF4J not found on the classpath. Logging is disabled for the 'org.mongodb.driver' component
Connecting to mongodb://root@example:10.210.60.18:27017
cAdvisor endpoint: http://10.210.60.18:8081/api/v1.3/docker/
Container ID: bbb56669e3aeac53bc06aa642f2dabfa503922ceb3edd4e4256d8f257f7a3fd
Sending request to cAdvisor at 2022-12-12T20:34:52.027
Sending request to cAdvisor at 2022-12-12T20:35:51.361
Sending request to cAdvisor at 2022-12-12T20:36:46.672
```

Figure 3.12

After the image is successfully pushed to Docker Hub, you can find the image on the Docker Hub website as shown in Figures 3.10 and 3.11, The URL is:

“<https://hub.docker.com/repository/docker/bkailash29/cadvisor-scraper>”

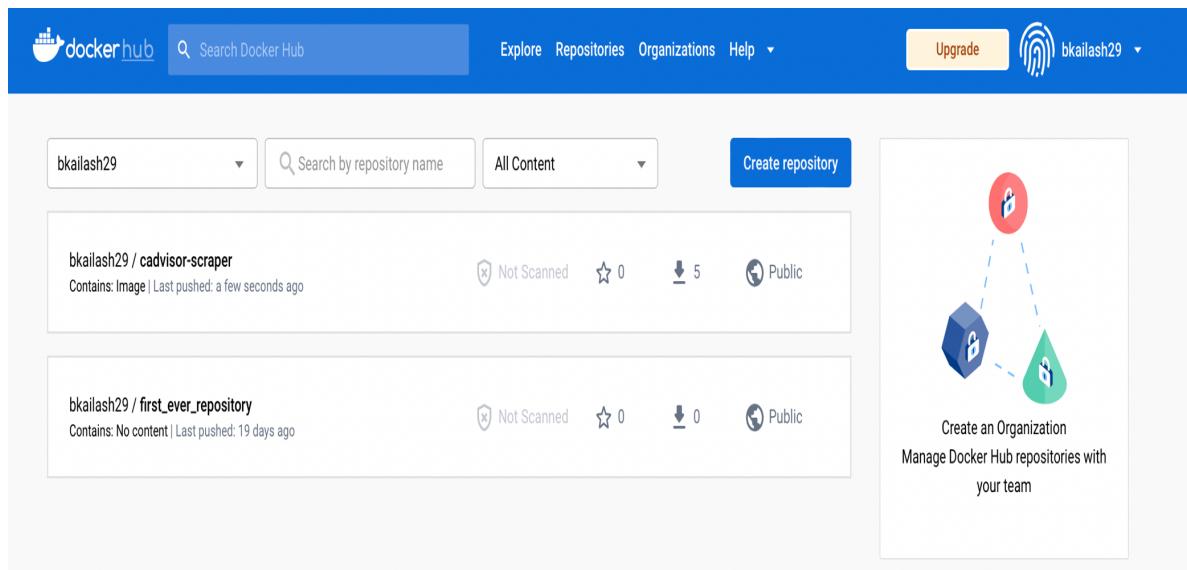


Figure 3.13

This screenshot shows the detailed view of the 'cadvisor-scraper' repository under the user 'bkailash29'. The top navigation bar includes 'Search Docker Hub', 'Explore', 'Repositories', 'Organizations', 'Help', 'Upgrade', and a user dropdown. The repository path 'bkailash29 / cadvisor-scraper' is shown in the breadcrumb. A message indicates 'Using 0 of 1 private repositories. [Get more](#)'. The main content area has tabs for General, Tags, Builds, Collaborators, Webhooks, and Settings. The General tab is selected. It features a note to add a short description, a 'Docker commands' section with a 'Public View' button, and a 'Tags and scans' section listing one tag ('latest'). The right side contains sections for 'Automated Builds' and 'Docker commands' with a command example: `docker push bkailash29/cadvisor-scraper:tagname`.

Figure 3.14

Task 4: Fully deploy and run the complex web application stack and undertake performance benchmarking activities

Task Objective: To learn and understand how to deploy complex web application stack, how to benchmark their run-time performance, and how to store the benchmarking results into a database server.

1. Pull the cAdvisor-scraper image, created in Task 3, back to the virtual machine on your laptop/PC.

Use the following command to push the image to Docker Hub.

“docker push bkailash29/cadvisor-scraper”

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker pull bkailash29/cadvisor-scraper
Using default tag: latest
latest: Pulling from bkailash29/cadvisor-scraper
Digest: sha256:774e0c1a421e14c7702399b1041bcfbaf5ab0f54cfcc154091666e7a9585dccc1
Status: Image is up to date for bkailash29/cadvisor-scraper:latest
docker.io/bkailash29/cadvisor-scraper:latest
```

Figure 4.1

2. Make necessary changes to cAdvisor-scraper so that it monitors the swarm

We can see in Figure 4.2 the data sent to Mongo DB by the web application. I have used mongo express to show the data. The application creates tables for the CPU, memory, each disk and each network interface.

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker run bkailash29/cadvisor-scraper
Dec 12, 2022 8:34:50 PM com.mongodb.diagnostics.logging.Loggers shouldUseSLF4J
WARNING: SLF4J not found on the classpath. Logging is disabled for the 'org.mongodb.driver' component
Connecting to mongodb://root@example@10.210.60.18:27017
cAdvisor endpoint: http://10.210.60.18:8081/api/v1.3/docker/
Container ID: bbbe56669e3aeac53bc06aa642f2dabfa503922ceb3edd4e4256d8f257f7a3fd
Sending request to cAdvisor at 2022-12-12T20:34:52.027
Sending request to cAdvisor at 2022-12-12T20:35:51.361
Sending request to cAdvisor at 2022-12-12T20:36:46.672
```

Figure 4.2

3. Launch cAdvisor-scraper as a container separately outside of the swarm

Since we left the swarm the container ID has changed. Therefore, we change the container ID of the benchmark app and build and run the cAdvisor-scarper app separately in a container as shown in Figure 4.3

CONTAINER ID	IMAGE	CMD	CREATED	STATUS	PORTS	NAMES
383acd04b574	bkailash29/cadvisor-scraper	"java -jar /cadvisor..."	57 minutes ago	Up 57 minutes		distracted_lunerie
75e7c6c6f2a	locustio/locust:latest	"locust -f /mnt/locu..."	2 hours ago	Up 2 hours	5557/tcp, 8089/tcp	cloudproject_locust.1.jp45wxyiteg3curys6rcrdjr4
2847751b2ffd	dockersamples/visualizer:stable	"npm start"	2 hours ago	Up 2 hours	8080/tcp	cloudproject_visualizer.1.bthxtys5jcx0tycv6xls03k
bbbe56669e3a	mongo:4.4.6	"docker-entrypoint.s..."	2 hours ago	Up 2 hours	27017/tcp	cloudproject_mongo.1.sjhkisvqdjb3p7eksu49h0uf
b92c5cf8c6a3	gcr.io/cadvisor/cadvisor:latest	"/usr/bin/cadvisor -..."	2 hours ago	Up 2 hours (healthy)	8080/tcp	cloudproject_imagemonitor.1.ydltpaswjeke89ydhiazzvkqj
da5f902b6dd6	ncclcloudcomputing/javabenchmarkapp:latest	"/bin/sh -c 'exec ja..."	2 hours ago	Up 2 hours		cloudproject_webapplication.1.4sgg1mb5udjclalt79lg0qbj

Figure 4.3

4. Send dynamic workload (e.g., http requests) to the Web Application Image Instance (in other words Web Application Container) by using the Load Generator Instance.

Figure 4.4 shows that connecting to the web application container by using the load generator instance by using the following command

docker exec -it --mango containerid mongo "mangodb://--username:--password@10.210.60.18:27017/"

docker exec -it --bbbe56669e3a mongo "mangodb://root@example@10.210.60.18:27017/"

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker exec -it bbbe56669e3a mongo "mangodb://root@example@10.210.60.18:27017/"
MongoDB shell version v4.4.6
connecting to: mongodb://10.210.60.18:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "_id" : UUID("3a5ce9cb-7188-4c16-bc00-8b066fd2d74d") }
MongoDB server version: 4.4.6
...
The server generated these startup warnings when booting:
2022-12-12T19:54:35.195+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-12-12T19:54:44.839+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest setting it to 'never'
...
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
```

Figure 4.4

5. Verify that the cAdvisor-scraper instance is logging the run-time performance statistics to the MongoDB instance.

Figure 4.5 shows the performance statistics of the MongoDB instance

```
student@ML-RefVm-691229:~/Desktop/cadvisor-scraper$ docker exec -it bbbbe56669e3a mongo "mongodb://root:example@10.210.60.18:27017/"  
MongoDB shell version v4.4.6  
connecting to: mongodb://10.210.60.18:27017/?compressors=disabled&gssapiServiceName=mongodb  
Implicit session: session { "id" : UUID("3a5ce9cb-7188-4c16-bc00-8b066fd2d74d") }  
MongoDB server version: 4.4.6  
...  
The server generated these startup warnings when booting:  
2022-12-12T19:54:35.195+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem  
2022-12-12T19:54:44.839+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest setting it to 'never'  
...  
...  
Enable MongoDB's free cloud-based monitoring service, which will then receive and display  
metrics about your deployment (disk utilization, CPU, operation statistics, etc).  
  
The monitoring data will be available on a MongoDB website with a unique URL accessible to you  
and anyone you share the URL with. MongoDB may use this information to make product  
improvements and to suggest MongoDB products and deployment options to you.  
  
To enable free monitoring, run the following command: db.enableFreeMonitoring()  
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()  
...  
> show dbs  
admin 0.000GB  
config 0.000GB  
local 0.000GB  
metrics 0.002GB  
> use metrics  
switched to db metrics  
> show collections  
bbbe5666-cpu  
bbbe5666-disk_/_dev/sdb  
bbbe5666-memory  
bbbe5666-network_eth0  
bbbe5666-network_eth1  
bbbe5666-network_eth2  
>
```

Figure 4.5

Task 5: Deploy Kubernetes using Terraform and deploy a microservice.

Use Terraform to achieve the following tasks

1. Enable the Azure subscription

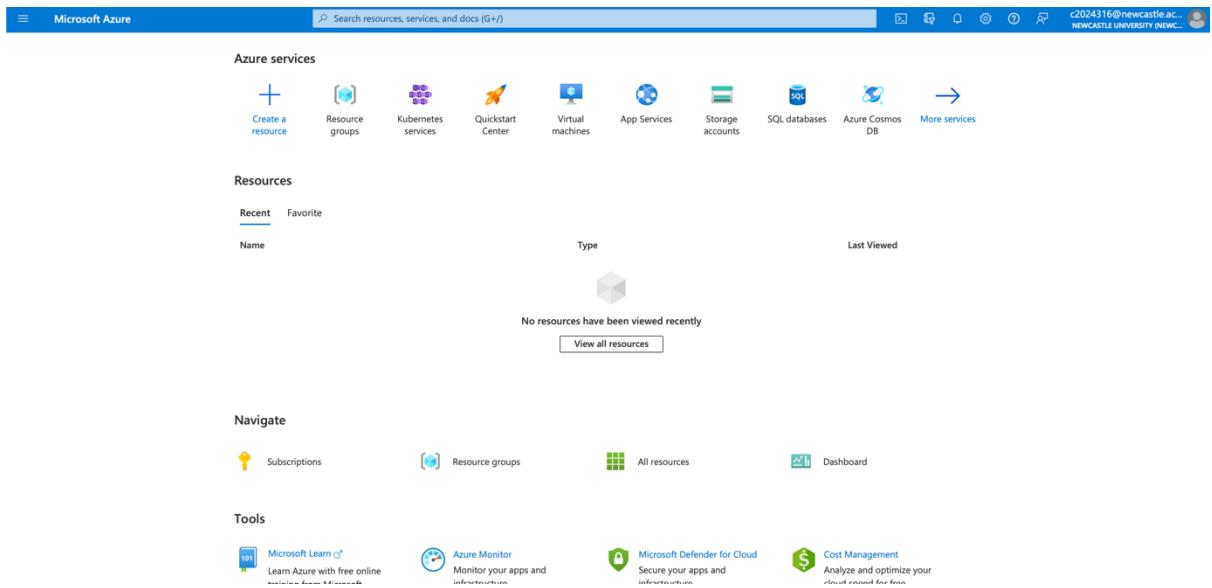
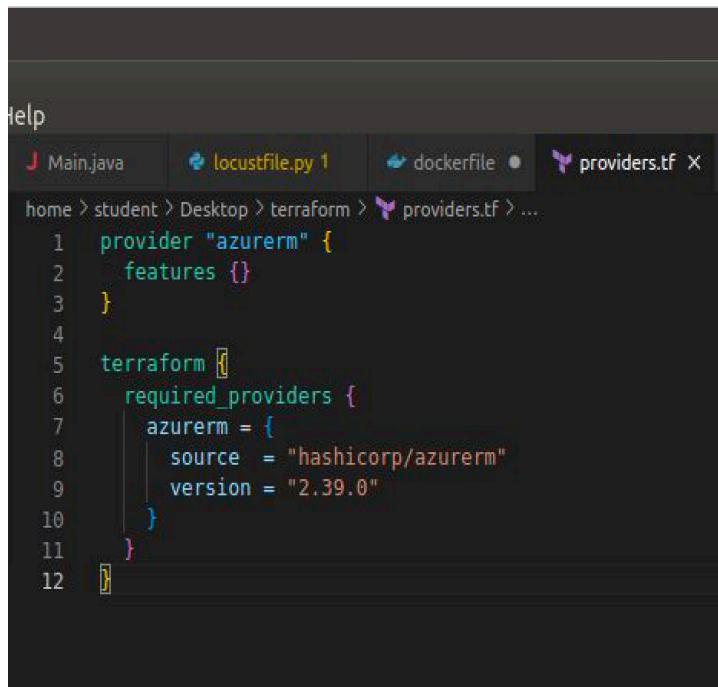


Figure 5.1

2. Create an Azure Kubernetes cluster named "csc8110" using HashiCorp language (HCL)

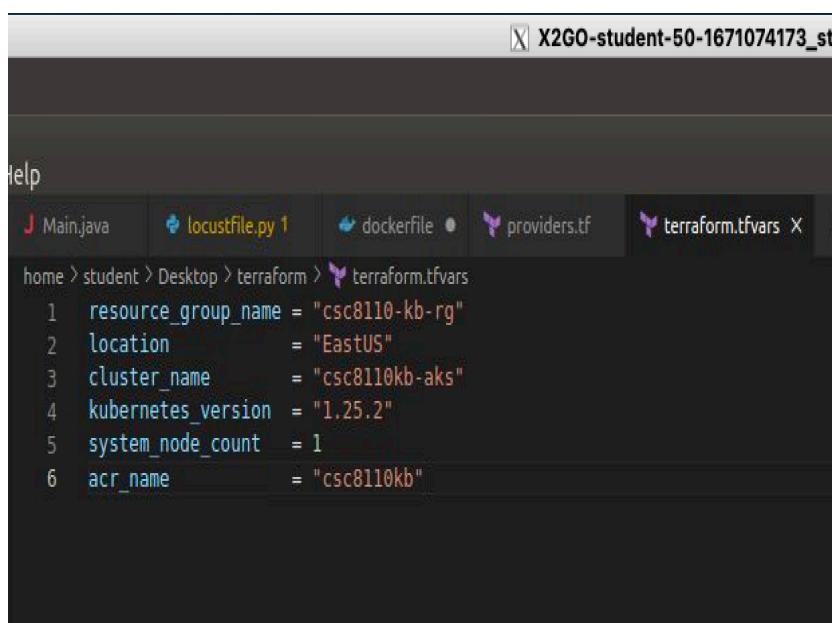
The following figures are the files to create the Azure Kubernetes cluster



```
provider "azurerm" {
  features {}
}

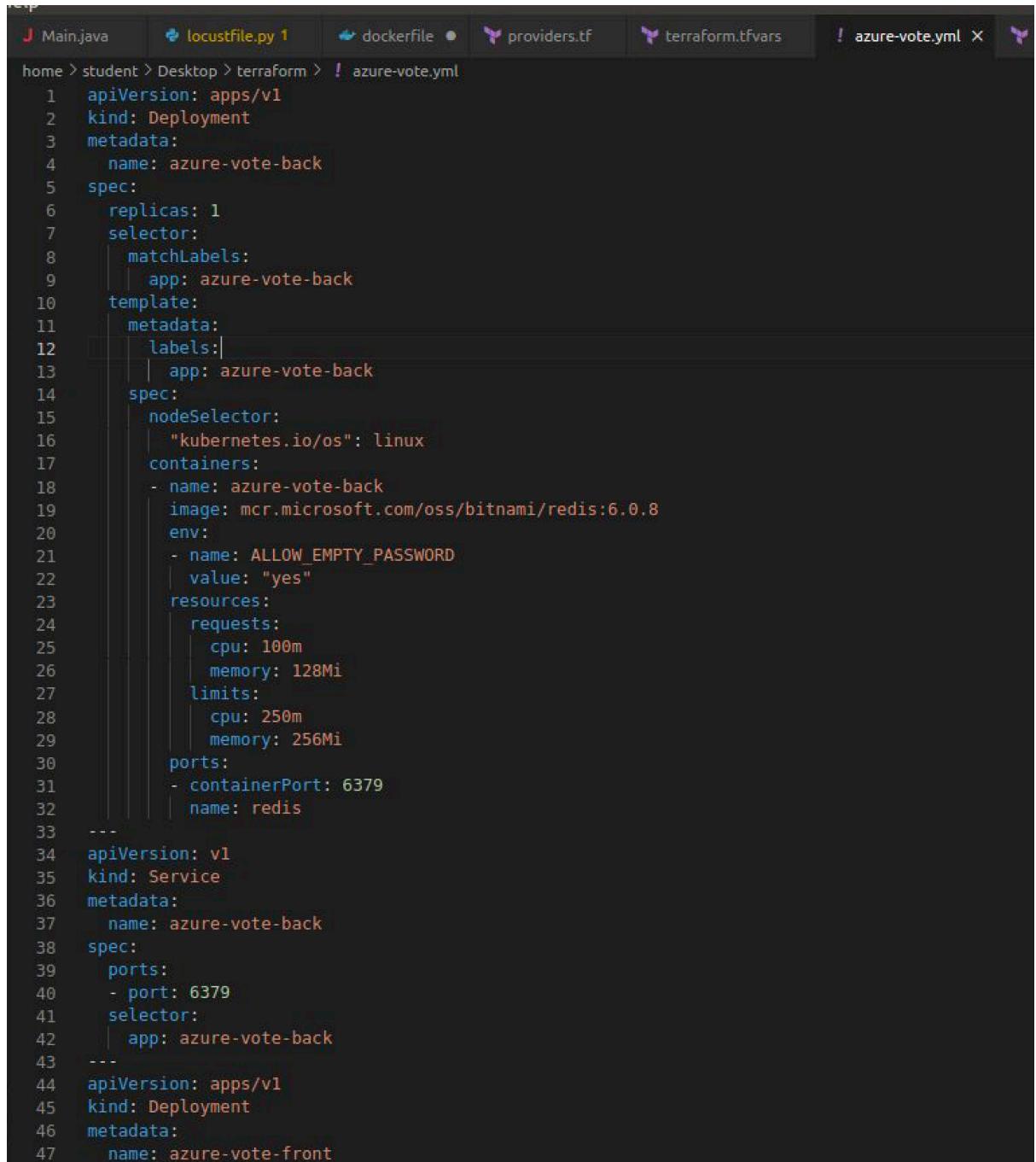
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "2.39.0"
    }
  }
}
```

Figure 5.2



```
resource_group_name = "csc8110-kb-rg"
location          = "EastUS"
cluster_name       = "csc8110kb-aks"
kubernetes_version = "1.25.2"
system_node_count  = 1
acr_name          = "csc8110kb"
```

Figure 5.3

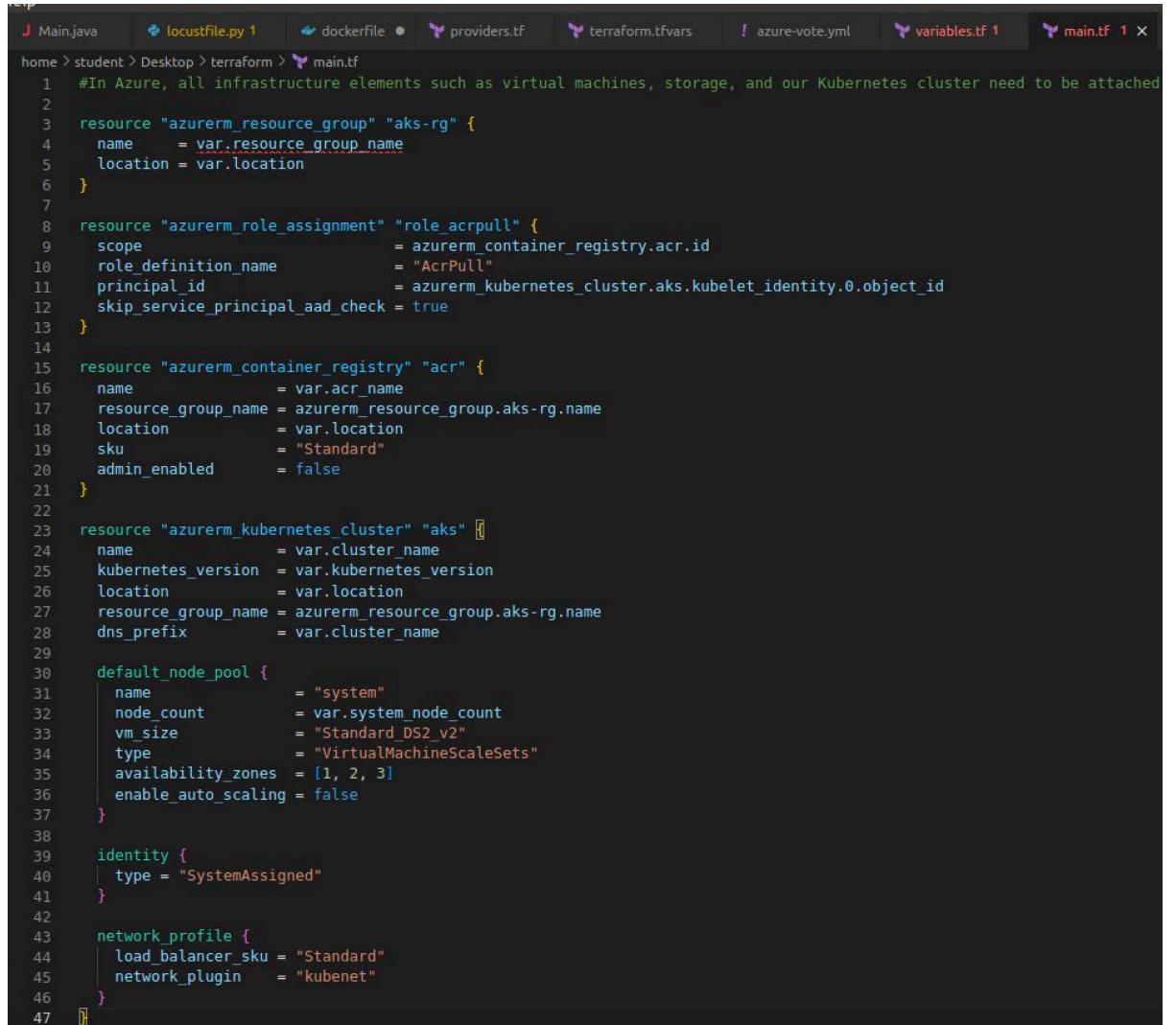


```
home > student > Desktop > terraform > ! azure-vote.yml
  1 apiVersion: apps/v1
  2 kind: Deployment
  3 metadata:
  4   name: azure-vote-back
  5 spec:
  6   replicas: 1
  7   selector:
  8     matchLabels:
  9       app: azure-vote-back
10   template:
11     metadata:
12       labels:
13         app: azure-vote-back
14     spec:
15       nodeSelector:
16         "kubernetes.io/os": linux
17       containers:
18         - name: azure-vote-back
19           image: mcr.microsoft.com/oss/bitnami/redis:6.0.8
20           env:
21             - name: ALLOW_EMPTY_PASSWORD
22               value: "yes"
23             resources:
24               requests:
25                 cpu: 100m
26                 memory: 128Mi
27               limits:
28                 cpu: 250m
29                 memory: 256Mi
30             ports:
31               - containerPort: 6379
32                 name: redis
33 ...
34 ...
35 apiVersion: v1
36 kind: Service
37 metadata:
38   name: azure-vote-back
39 spec:
40   ports:
41     - port: 6379
42   selector:
43     app: azure-vote-back
44 ...
45 apiVersion: apps/v1
46 kind: Deployment
47 metadata:
48   name: azure-vote-front
```

Figure 5.4

```
home > student > Desktop > terraform > variables.tf
1 variable "resource_group_name" {
2   type     = string
3   description = "RG name in Azure"
4 }
5 variable "location" {
6   type     = string
7   description = "Resources location in Azure"
8 }
9 variable "cluster_name" {
10  type     = string
11  description = "AKS name in Azure"
12 }
13 variable "kubernetes_version" {
14  type     = string
15  description = "Kubernetes version"
16 }
17 variable "system_node_count" [
18  type     = number
19  description = "Number of AKS worker nodes"
20 ]
21 variable "acr_name" {
22  type     = string
23  description = "ACR name"
24 }
```

Figure 5.5

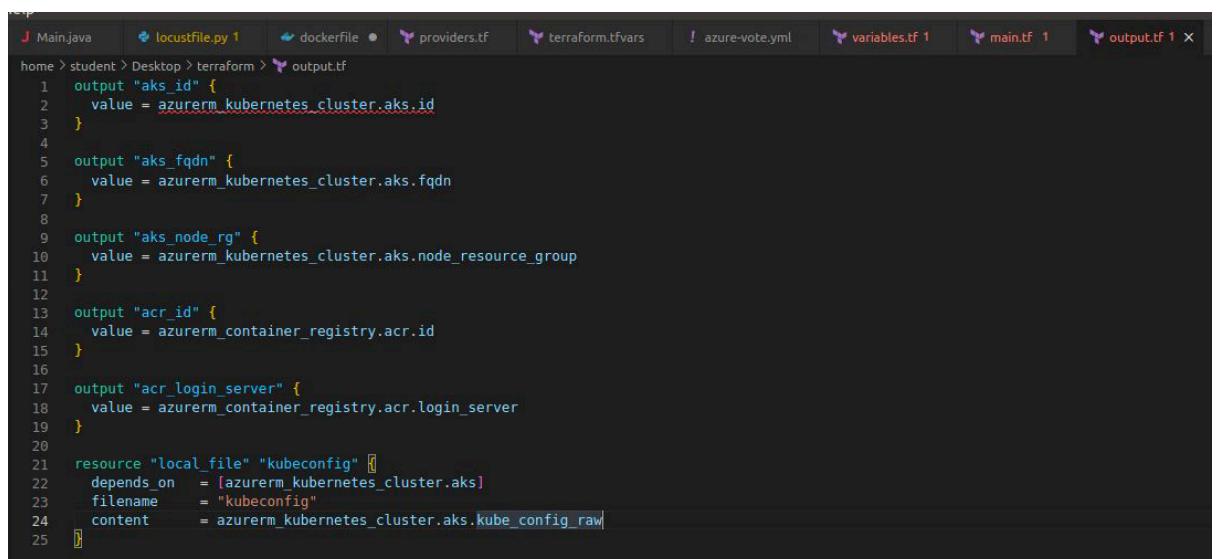


```

Main.java locustfile.py 1 dockerfile providers.tf terraform.tfvars ! azure-vote.yml variables.tf 1 main.tf 1 X
home > student > Desktop > terraform > main.tf
1 #In Azure, all infrastructure elements such as virtual machines, storage, and our Kubernetes cluster need to be attached
2
3 resource "azurerm_resource_group" "aks-rg" {
4   name     = var.resource_group_name
5   location = var.location
6 }
7
8 resource "azurerm_role_assignment" "role_acrpull" {
9   scope                  = azurerm_container_registry.acr.id
10  role_definition_name  = "AcrPull"
11  principal_id          = azurerm_kubernetes_cluster.aks.kubelet_identity.0.object_id
12  skip_service_principal_aad_check = true
13 }
14
15 resource "azurerm_container_registry" "acr" {
16   name     = var.acr_name
17   resource_group_name = azurerm_resource_group.aks-rg.name
18   location      = var.location
19   sku           = "Standard"
20   admin_enabled = false
21 }
22
23 resource "azurerm_kubernetes_cluster" "aks" [
24   name     = var.cluster_name
25   kubernetes_version = var.kubernetes_version
26   location      = var.location
27   resource_group_name = azurerm_resource_group.aks-rg.name
28   dns_prefix    = var.cluster_name
29
30   default_node_pool {
31     name        = "system"
32     node_count = var.system_node_count
33     vm_size     = "Standard_DS2_v2"
34     type        = "VirtualMachineScaleSets"
35     availability_zones = [1, 2, 3]
36     enable_auto_scaling = false
37   }
38
39   identity {
40     type = "SystemAssigned"
41   }
42
43   network_profile {
44     load_balancer_sku = "Standard"
45     network_plugin    = "kubenet"
46   }
47 ]

```

Figure 5.6



```

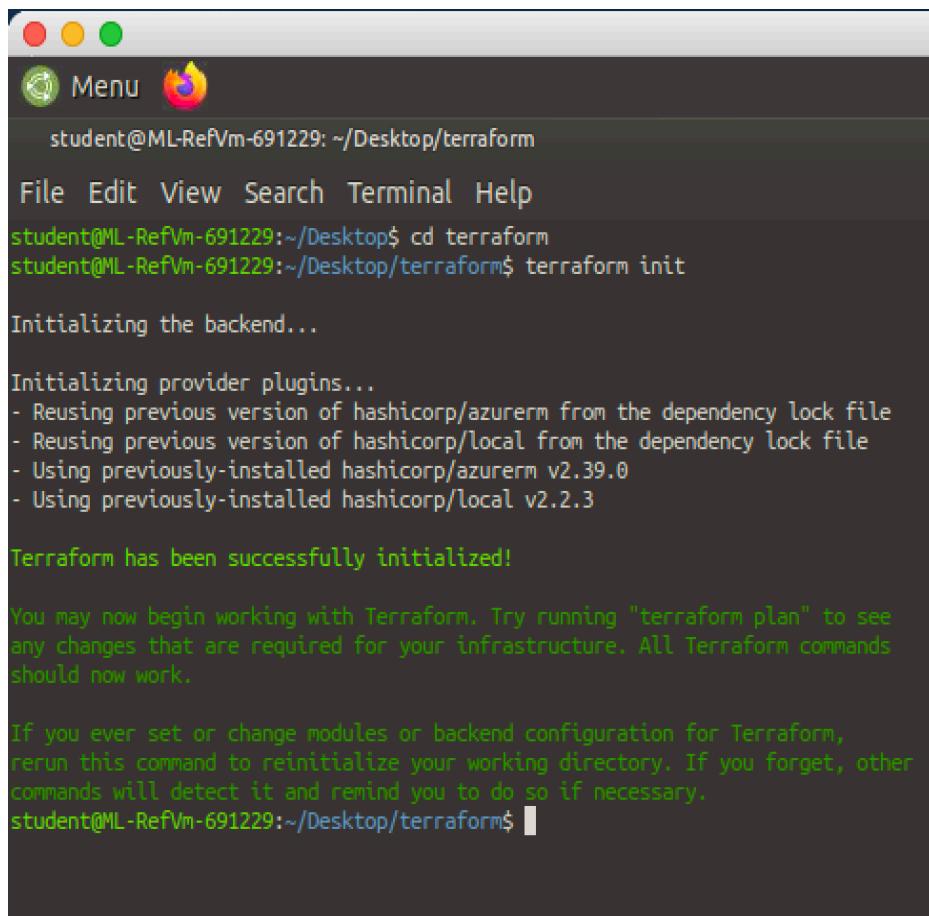
Main.java locustfile.py 1 dockerfile providers.tf terraform.tfvars ! azure-vote.yml variables.tf 1 main.tf 1 output.tf 1 X
home > student > Desktop > terraform > output.tf
1 output "aks_id" {
2   value = azurerm_kubernetes_cluster.aks.id
3 }
4
5 output "aks_fqdn" {
6   value = azurerm_kubernetes_cluster.aks.fqdn
7 }
8
9 output "aks_node_rg" {
10  value = azurerm_kubernetes_cluster.aks.node_resource_group
11 }
12
13 output "acr_id" {
14   value = azurerm_container_registry.acr.id
15 }
16
17 output "acr_login_server" {
18   value = azurerm_container_registry.acr.login_server
19 }
20
21 resource "local_file" "kubeconfig" [
22   depends_on  = [azurerm_kubernetes_cluster.aks]
23   filename    = "kubeconfig"
24   content     = azurerm_kubernetes_cluster.aks.kube_config_raw
25 ]

```

Figure 5.7

Initiate the terraform using the following command in the docker CLI as shown in figure 5.8

“terraform init”



The screenshot shows a terminal window with a dark background and light-colored text. At the top, there are three colored circles (red, yellow, green) and icons for 'Menu' and a browser. The terminal title is 'student@ML-RefVm-691229: ~/Desktop/terraform'. The user runs the command 'cd terraform' followed by 'terraform init'. The output shows the initialization process: 'Initializing the backend...', 'Initializing provider plugins...', and a list of reused providers: 'Reusing previous version of hashicorp/azurerm from the dependency lock file', 'Reusing previous version of hashicorp/local from the dependency lock file', 'Using previously-installed hashicorp/azurerm v2.39.0', and 'Using previously-installed hashicorp/local v2.2.3'. A success message 'Terraform has been successfully initialized!' is displayed, along with instructions to run 'terraform plan' to see changes. The prompt 'student@ML-RefVm-691229:~/Desktop/terraform\$' is at the bottom.

```
student@ML-RefVm-691229:~/Desktop/terraform
student@ML-RefVm-691229:~/Desktop/terraform$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/azurerm from the dependency lock file
- Reusing previous version of hashicorp/local from the dependency lock file
- Using previously-installed hashicorp/azurerm v2.39.0
- Using previously-installed hashicorp/local v2.2.3

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
student@ML-RefVm-691229:~/Desktop/terraform$
```

Figure 5.8

The following command executes the terraform

“terraform plan”

```

student@ML-RefVm-691229: ~/Desktop/terraform
File Edit View Search Terminal Help
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
student@ML-RefVm-691229:~/Desktop/terraform$ terraform plan
azurerm_resource_group.aks-rg: Refreshing state... [id=subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg]
azurerm_container_registry.acr: Refreshing state... [id=subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb-rg]
azurerm_kubernetes_cluster.aks: Refreshing state... [id=subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks]
azurerm_role_assignment.role_acrpull: Refreshing state... [id=subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/30a1f0]
local_file.kubeconfig: Refreshing state... [id=41e69095e3493ea3ad779fd58fd06322663c396]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last "terraform apply" which may have affected this plan:

# azurerm_container_registry.acr has been deleted
resource "azurerm_container_registry" "acr" {
  id          = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb" -> null
  login_server = "csc8110kb.azurecr.io" -> null
  name        = "csc8110kb"
  tags         = {}
}

# azurerm_kubernetes_cluster.aks has been deleted
resource "azurerm_kubernetes_cluster" "aks" {
  fqdn      = "csc8110kb-aks-3605d80c.hcp.eastus.azurek8s.io" -> null
  id          = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks"
  kube_config_raw = (sensitive value)
  kubernetes_version = [
    {
      object_id = "b248395c-97e4-4c24-98c2-9879ce537ea" -> null
    }
  ]
  name        = "csc8110kb-aks"
  node_resource_group = "MC_csc8110-kb-rg_csc8110kb-aks_eastus" -> null
  tags         = {}

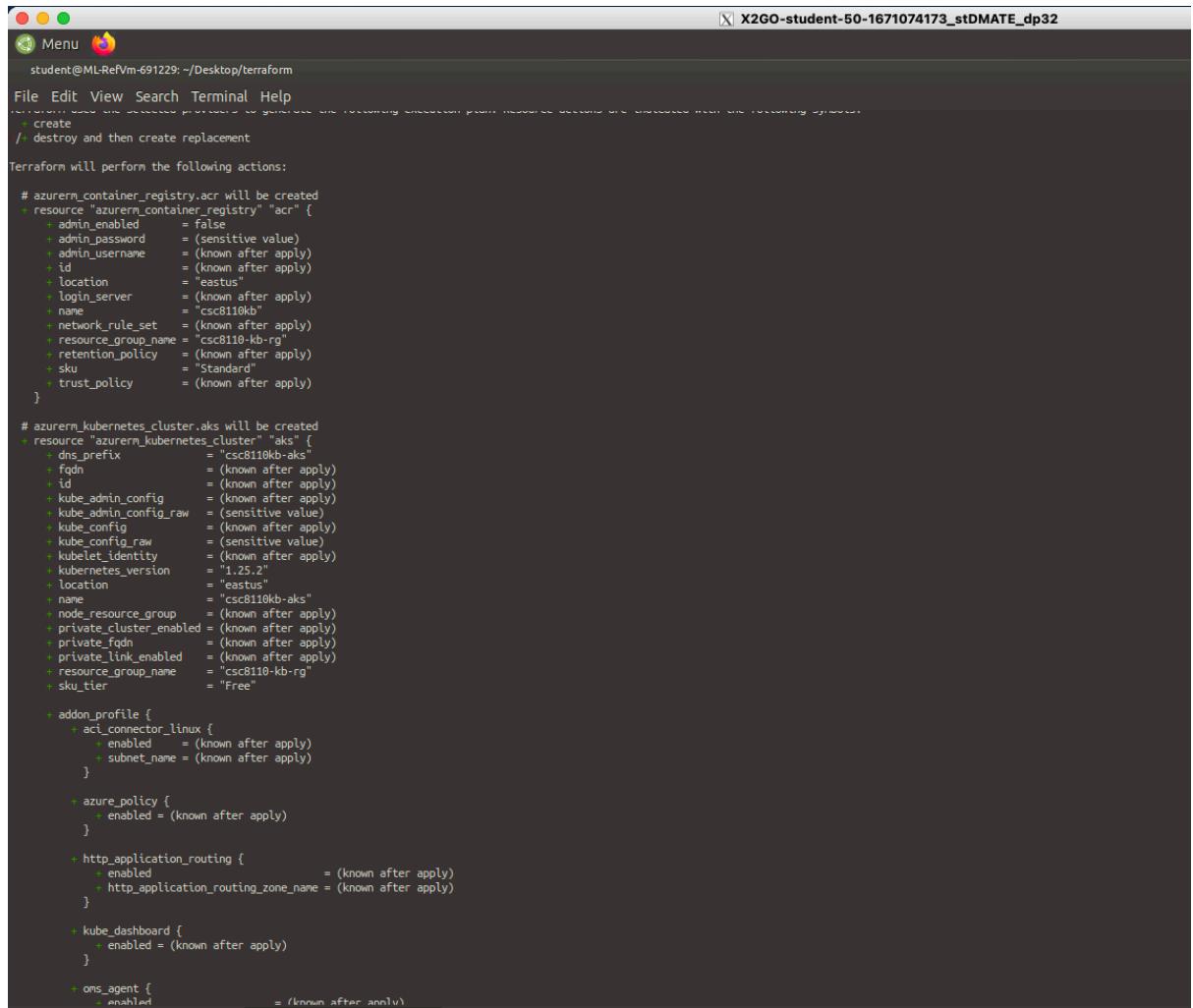
# (11 unchanged attributes hidden)

# (5 unchanged blocks hidden)
}

# azurerm_resource_group.aks-rg has been deleted
resource "azurerm_resource_group" "aks-rg" {
  id          = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg"
  name        = "csc8110-kb-rg" -> null
  tags         = {}
}

```

Figure 5.9



The screenshot shows a terminal window titled "X2GO-student-50-1671074173_stDMATE_dp32". The window displays a Terraform plan for creating resources. The output includes resource definitions and their properties, such as "azurerm_container_registry.acr" and "azurerm_kubernetes_cluster.aks", along with their specific configurations like "admin_enabled", "name", and "sku". The terminal also indicates that the plan will be applied using the "apply" command.

```
student@ML-RefVm-691229: ~/Desktop/terraform
File Edit View Search Terminal Help
Terraform uses the selected provider to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
-/ destroy and then create replacement
Terraform will perform the following actions:

# azurerm_container_registry.acr will be created
+ resource "azurerm_container_registry" "acr" {
    + admin_enabled      = false
    + admin_password     = (sensitive value)
    + admin_username     = (known after apply)
    + id                 = (known after apply)
    + location           = "eastus"
    + login_server       = (known after apply)
    + name               = "csc8110kb"
    + network_rule_set   = (known after apply)
    + resource_group_name = "csc8110-kb-rg"
    + retention_policy   = (known after apply)
    + sku                = "Standard"
    + trust_policy       = (known after apply)
}

# azurerm_kubernetes_cluster.aks will be created
+ resource "azurerm_kubernetes_cluster" "aks" {
    + dns_prefix          = "csc8110kb-aks"
    + fqdn                = (known after apply)
    + id                  = (known after apply)
    + kube_admin_config   = (known after apply)
    + kube_admin_config_raw = (sensitive value)
    + kube_config          = (known after apply)
    + kube_config_raw     = (sensitive value)
    + kubelet_identity    = (known after apply)
    + kubernetes_version  = "1.25.2"
    + location             = "eastus"
    + name                = "csc8110kb-aks"
    + node_resource_group = (known after apply)
    + private_cluster_enabled = (known after apply)
    + private_fqdn         = (known after apply)
    + private_link_enabled = (known after apply)
    + resource_group_name = "csc8110-kb-rg"
    + sku_tier             = "free"

    + addon_profile {
        + aci_connector_linux {
            + enabled = (known after apply)
            + subnet_name = (known after apply)
        }
    }

    + azure_policy {
        + enabled = (known after apply)
    }

    + http_application_routing {
        + enabled          = (known after apply)
        + http_application_routing_zone_name = (known after apply)
    }

    + kube_dashboard {
        + enabled = (known after apply)
    }

    + ons_agent {
        + enabled = (known after apply)
    }
}
```

Figure 5.10

```

student@ML-RefVm-691229: ~/Desktop/terraform
File Edit View Search Terminal Help
+ outbound_ports_allocated = (known after apply)
}
}
+ role_based_access_control {
+ enabled = (known after apply)
+ azure_active_directory {
+ admin_group_object_ids = (known after apply)
+ client_app_id          = (known after apply)
+ managed                 = (known after apply)
+ server_app_id           = (known after apply)
+ server_app_secret        = (sensitive value)
+ tenant_id                = (known after apply)
}
}
+ windows_profile {
+ admin_password = (sensitive value)
+ admin_username = (known after apply)
}
}

# azurerm_resource_group.aks-rg will be created
resource "azurerm_resource_group" "aks-rg" {
+ id          = (known after apply)
+ location    = "eastus"
+ name        = "csc8110-kb-rg"
}

# azurerm_role_assignment.role_acrpull will be created
resource "azurerm_role_assignment" "role_acrpull" {
+ id          = (known after apply)
+ name        = (known after apply)
+ principal_id = (known after apply)
+ principal_type = (known after apply)
+ role_definition_id = (known after apply)
+ role_definition_name = "AcrPull"
+ scope       = (known after apply)
+ skip_service_principal_aad_check = true
}

# local_file.kubeconfig must be replaced
resource "local_file" "kubeconfig" {
- content      = (sensitive value) -> (sensitive value) # forces replacement
- id          = "41e69095e3493ea63ad779fd58fd06322663c396" -> (known after apply)
# (3 unchanged attributes hidden)
}

Plan: 5 to add, 0 to change, 1 to destroy.

Changes to Outputs:
- acri_id      = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb" -> (known after apply)
- acri_login_server = "csc8110kb.azurecr.io" -> (known after apply)
- aks_fqdn     = "csc8110kb-aks-3605d88c.hcp.eastus.azmk8s.io" -> (known after apply)
- aks_id        = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks" -> (known after apply)
- aks_node_rg   = "k8s-csc8110-kb-rg_csc8110kb-aks_eastus" -> (known after apply)

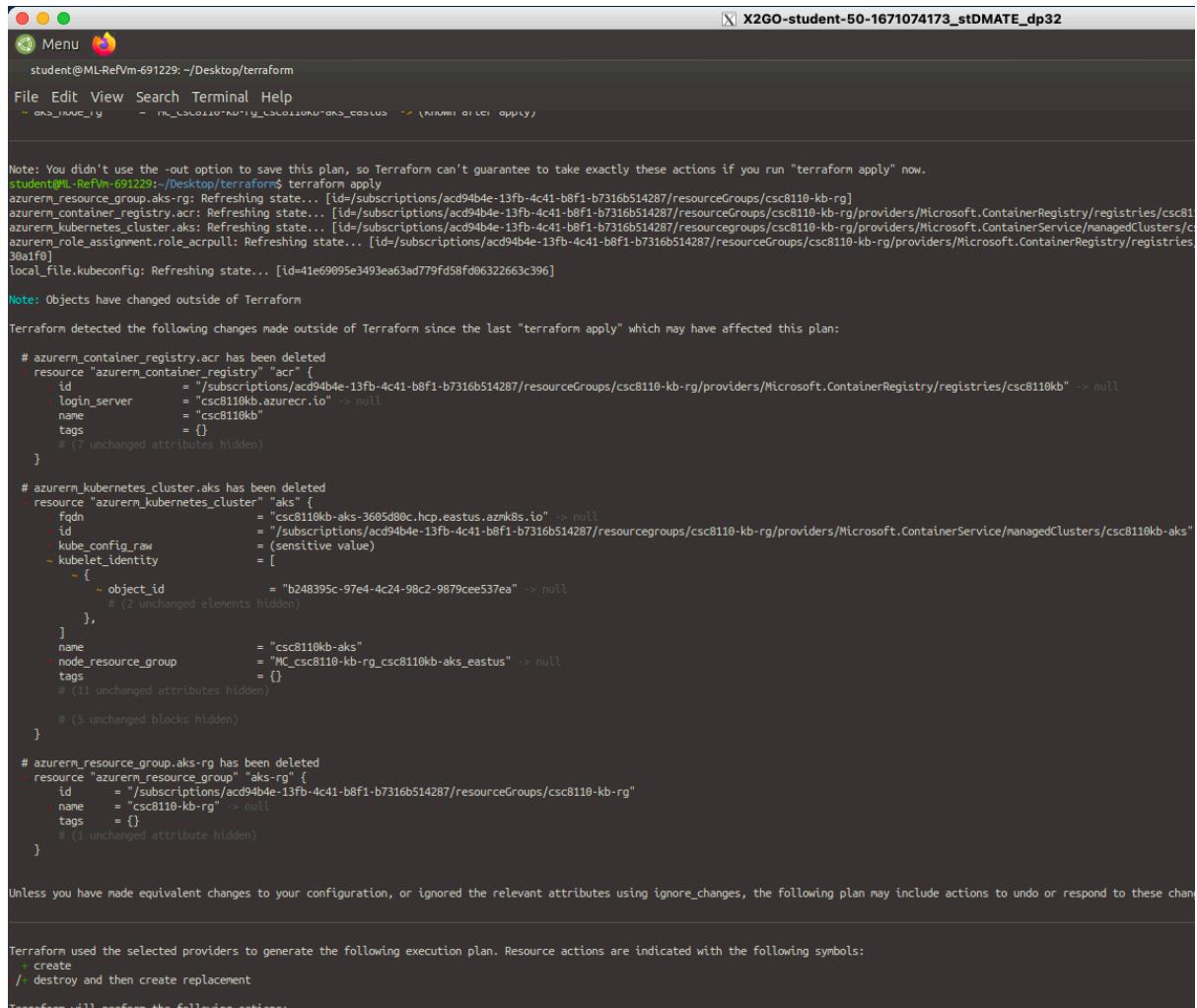
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
student@ML-RefVm-691229: ~/Desktop/terraform$ 

```

Figure 5.11

To save the executed plan run the following command as shown in the following figures

“terraform apply”



The screenshot shows a terminal window titled "X2GO-student-50-1671074173_stDMATE_dp32". The window displays the output of a Terraform "apply" command. The output includes notes about objects changing outside of Terraform, detected changes, and a detailed execution plan for resources like Container Registry, Kubernetes Cluster, and Resource Group. The terminal is running on a student VM.

```

student@ML-RefVm-691229:~/Desktop/terraform$ terraform apply
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
azurerm_resource_group.aks-rg: Refreshing state... [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg]
azurerm_container_registry.acr: Refreshing state... [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110-kb-rg]
azurerm_kubernetes_cluster.aks: Refreshing state... [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110-kb-rg]
azurerm_role_assignment.role_acrpull: Refreshing state... [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110-kb-rg]
local_file.kubeconfig: Refreshing state... [id=41e69095e3493ea63ad779fd58fd06322663c396]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last "terraform apply" which may have affected this plan:

# azurerm_container_registry.acr has been deleted
- resource "azurerm_container_registry" "acr" {
  - id          = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb" -> null
  - login_server = "csc8110kb.azurecr.io" -> null
  - name        = "csc8110kb"
  - tags         = {}

  # (7 unchanged attributes hidden)
}

# azurerm_kubernetes_cluster.aks has been deleted
- resource "azurerm_kubernetes_cluster" "aks" {
  - fqdn      = "csc8110kb.aks-3605d80c.hcp.eastus.azmk8s.io" -> null
  - id        = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks"
  - kube_config_raw = (sensitive value)
  - kublet_identity = [
    ~ {
      - object_id   = "b248395c-97e4-4c24-98c2-9879cee537ea" -> null
      # (2 unchanged elements hidden)
    },
  ]
  - name        = "csc8110kb-aks"
  - node_resource_group = "MC_csc8110-kb-rg_csc8110kb-aks_eastus" -> null
  - tags         = {}

  # (11 unchanged attributes hidden)
}

# azurerm_resource_group.aks-rg has been deleted
- resource "azurerm_resource_group" "aks-rg" {
  - id          = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg"
  - name        = "csc8110-kb-rg" -> null
  - tags         = {}

  # (1 unchanged attribute hidden)
}

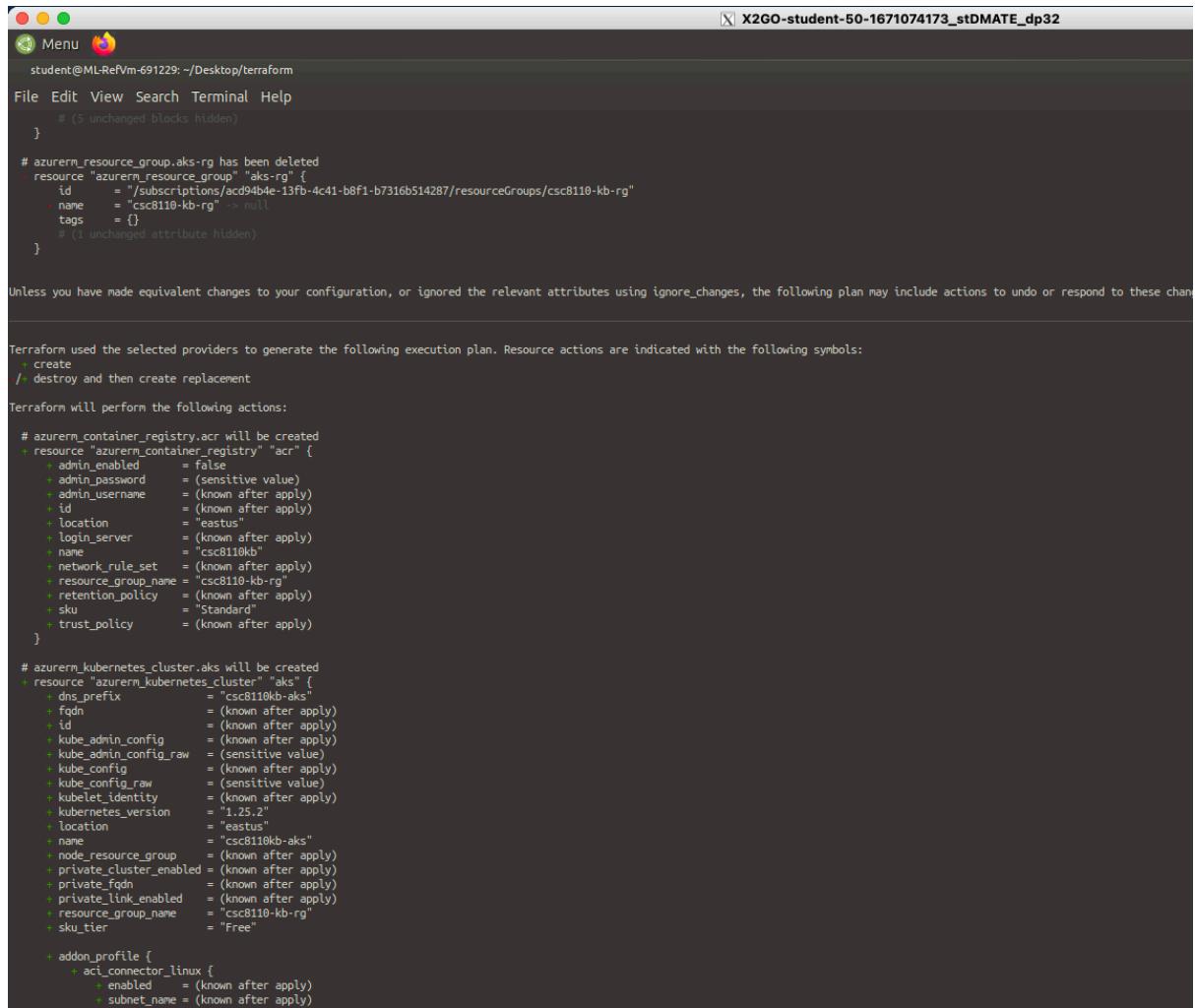
Unless you have made equivalent changes to your configuration, or ignored the relevant attributes using ignore_changes, the following plan may include actions to undo or respond to these changes.

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  create
  / destroy and then create replacement

Terraform will perform the following actions:

```

Figure 5.12



```

X2GO-student-50-1671074173_stDMATE_dp32
student@ML-RefVm-691229: ~/Desktop/terraform

File Edit View Search Terminal Help
# (5 unchanged blocks hidden)

# azurerm_resource_group.aks-rg has been deleted
resource "azurerm_resource_group" "aks-rg" {
  id      = "/subscriptions/acd9404e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg"
  name    = "csc8110-kb-rg"
  tags    = {}
}

Unless you have made equivalent changes to your configuration, or ignored the relevant attributes using ignore_changes, the following plan may include actions to undo or respond to these changes.

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
-/ destroy and then create replacement

Terraform will perform the following actions:

# azurerm_container_registry.acr will be created
resource "azurerm_container_registry" "acr" {
  admin_enabled        = false
  admin_password       = (sensitive value)
  admin_username       = (known after apply)
  id                  = (known after apply)
  location            = "eastus"
  login_server         = (known after apply)
  name                = "csc8110kb"
  network_rule_set    = (known after apply)
  resource_group_name = "csc8110-kb-rg"
  retention_policy     = (known after apply)
  sku                 = "Standard"
  trust_policy         = (known after apply)
}

# azurerm_kubernetes_cluster.aks will be created
resource "azurerm_kubernetes_cluster" "aks" {
  dns_prefix           = "csc8110kb-aks"
  id                  = (known after apply)
  kube_admin_config    = (known after apply)
  kube_admin_config_raw = (sensitive value)
  kube_config          = (known after apply)
  kube_config_raw      = (sensitive value)
  kubelet_identity     = (known after apply)
  kubernetes_version   = "1.25.2"
  location             = "eastus"
  name                = "csc8110kb-aks"
  node_resource_group  = (known after apply)
  private_cluster_enabled = (known after apply)
  private_fqdn          = (known after apply)
  private_link_enabled  = (known after apply)
  resource_group_name   = "csc8110-kb-rg"
  sku_tier              = "Free"

  addon_profile {
    aci_connector_linux {
      enabled    = (known after apply)
      subnet_name = (known after apply)
    }
  }
}

```

Figure 5.13

```

Menu
student@ML-RefVm-691229: ~/Desktop/terraform
File Edit View Search Terminal Help

+ windows_profile {
  admin_password = (sensitive value)
  admin_username = (known after apply)
}

# azurerm_resource_group.aks_rg will be created
+ resource "azurerm_resource_group" "aks_rg" {
  id          = (known after apply)
  location    = "eastus"
  name        = "csc8110-kb-rg"
}

# azurerm_role_assignment.role_acrpull will be created
+ resource "azurerm_role_assignment" "role_acrpull" {
  id          = (known after apply)
  name        = (known after apply)
  principal_id = (known after apply)
  principal_type = (known after apply)
  role_definition_id = (known after apply)
  role_definition_name = "AcRpull"
  scope       = (known after apply)
  skip_service_principal_aad_check = true
}

# local_file.kubeconfig must be replaced
/+ resource "local_file" "kubeconfig" {
  ~ content      = (sensitive value) -> (sensitive value) # Forces replacement
  ~ id           = "41e69995e3493ea63ad779fd58fd06322663c396" -> (known after apply)
  # (3 unchanged attributes hidden)
}

Plan: 5 to add, 0 to change, 1 to destroy.

Changes to Outputs:
- acrio = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb" -> (known after apply)
- acri_login_server = "csc8110kb.azurecr.io" -> (known after apply)
- aks_fqdn = "csc8110kb-aks-360508c.hcp.eastus.azmk8s.io" -> (known after apply)
- aks_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks" -> (known after apply)
- aks_node_rg = "MC_csc8110-kb-rg_csc8110kb-aks_eastus" -> (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

local_file.kubeconfig: Destroying... [id=41e69995e3493ea63ad779fd58fd06322663c396]
local_file.kubeconfig: Destruction complete after 0s
azurerm_resource_group.aks_rg: Creating...
azurerm_resource_group.aks_rg: Creation complete after 2s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg]
azurerm_container_registry.acr: Creating...
azurerm_kubernetes_cluster.aks: Creating...
azurerm_container_registry.acr: Still creating... [10s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [10s elapsed]
azurerm_container_registry.acr: Creation complete after 14s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb]
azurerm_kubernetes_cluster.aks: Still creating... [20s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [30s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [40s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [50s elapsed]

```

Figure 5.14

```

Enter a value: yes

local_file.kubeconfig: Destroying... [id=41e69095e3493ea63ad779fd58fd06322663c396]
local_file.kubeconfig: Destruction complete after 0s
azurerm_resource_group.aks-rg: Creating...
azurerm_resource_group.aks-rg: Creation complete after 2s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg]
azurerm_container_registry.acr: Creating...
azurerm_kubernetes_cluster.aks: Creating...
azurerm_container_registry.acr: Still creating... [10s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [10s elapsed]
azurerm_container_registry.acr: Creation complete after 14s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb]
azurerm_kubernetes_cluster.aks: Still creating... [20s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [30s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [40s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [50s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m0s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m10s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m20s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m30s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m40s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [1m50s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m0s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m10s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m20s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m30s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m40s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [2m50s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m0s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m10s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m20s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m30s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m40s elapsed]
azurerm_kubernetes_cluster.aks: Still creating... [3m50s elapsed]
azurerm_kubernetes_cluster.aks: Creation complete after 4m14s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks]
azurerm_role_assignment.role_acrpull: Creating...
local_file.kubeconfig: Creating...
local_file.kubeconfig: Creation complete after 0s [id=577c98b18a1d7c924f70c1a5af7243c3859ee799]
azurerm_role_assignment.role_acrpull: Still creating... [10s elapsed]
azurerm_role_assignment.role_acrpull: Still creating... [20s elapsed]
azurerm_role_assignment.role_acrpull: Creation complete after 25s [id=/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/roleAssignments/1-8193d9a48fffb]

Apply complete! Resources: 5 added, 0 changed, 1 destroyed.

Outputs:

acr_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb"
acr_login_server = "csc8110kb.azurecr.io"
aks_fqdn = "csc8110kb-aks-f490ab55.hcp.eastus.azurek8s.io"
aks_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks"
aks_node_rg = "MC_csc8110-kb-rg_csc8110kb-aks_eastus"
student@ML-RefVm-691229:~/Desktop/terraform$ 
```

Figure 5.15

To run the kubeconfig file following command is used

“az aks get-credentials --resource-group csc8110-kb-rg –name csc8110kb-aks”

```

acr_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerRegistry/registries/csc8110kb"
acr_login_server = "csc8110kb.azurecr.io"
aks_fqdn = "csc8110kb-aks-f490ab55.hcp.eastus.azurek8s.io"
aks_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourceGroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks"
aks_node_rg = "MC_csc8110-kb-rg_csc8110kb-aks_eastus"
student@ML-RefVm-691229:~/Desktop/terraform$ az aks get-credentials --resource-group csc8110-kb-rg --name csc8110kb-aks
A different object named csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): y
A different object named clusterUser_csc8110-kb-rg_csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): yes
A different object named clusterUser_csc8110-kb-rg_csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): y
Merged "csc8110kb-aks" as current context in /home/student/.kube/config
student@ML-RefVm-691229:~/Desktop/terraform$ 
```

Figure 5.16

The following command is used to get the Kubernetes node

“kubectl get nodes”

```

aks_rguri = "https://csc8110kb-aks-14508531.hcp.eastus.azmkube.net"
aks_id = "/subscriptions/acd94b4e-13fb-4c41-b8f1-b7316b514287/resourcegroups/csc8110-kb-rg/providers/Microsoft.ContainerService/managedClusters/csc8110kb-aks"
aks_node_rg = "MC_csc8110-kb-rg_csc8110kb-aks_eastus"
student@ML-RefVm-691229:~/Desktop/terraform$ az aks get-credentials --resource-group csc8110-kb-rg --name csc8110kb-aks
A different object named csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): y
A different object named clusterUser_csc8110-kb-rg_csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): yes
A different object named clusterUser_csc8110-kb-rg_csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): y
Merged "csc8110kb-aks" as current context in /home/student/.kube/config
student@ML-RefVm-691229:~/Desktop/terraform$ kubectl get nodes
NAME           STATUS   ROLES   AGE    VERSION
aks-system-12618104-vmss000000 Ready   agent   7m59s   v1.25.2
student@ML-RefVm-691229:~/Desktop/terraform$
```

Figure 5.17

The following command is used to create the service and deploy the application by running the azure-vote.yml file

“kubectl apply -f azure-vote.yml”

```

Overwrite? (y/n): yes
A different object named clusterUser_csc8110-kb-rg_csc8110kb-aks already exists in your kubeconfig file.
Overwrite? (y/n): y
Merged "csc8110kb-aks" as current context in /home/student/.kube/config
student@ML-RefVm-691229:~/Desktop/terraform$ kubectl get nodes
NAME           STATUS   ROLES   AGE    VERSION
aks-system-12618104-vmss000000 Ready   agent   7m59s   v1.25.2
student@ML-RefVm-691229:~/Desktop/terraform$ kubectl get service azure-vote-front --watch
Error from server (NotFound): services "azure-vote-front" not found
student@ML-RefVm-691229:~/Desktop/terraform$ kubectl apply -f azure-vote.yml
deployment.apps/azure-vote-back created
service/azure-vote-back created
deployment.apps/azure-vote-front created
service/azure-vote-front created
student@ML-RefVm-691229:~/Desktop/terraform$
```

Figure 5.18

3. Deploy any sample microservice application once the cluster is ready

Figure 5.19 shows that the microservice application was created and running in 20.242.152.59

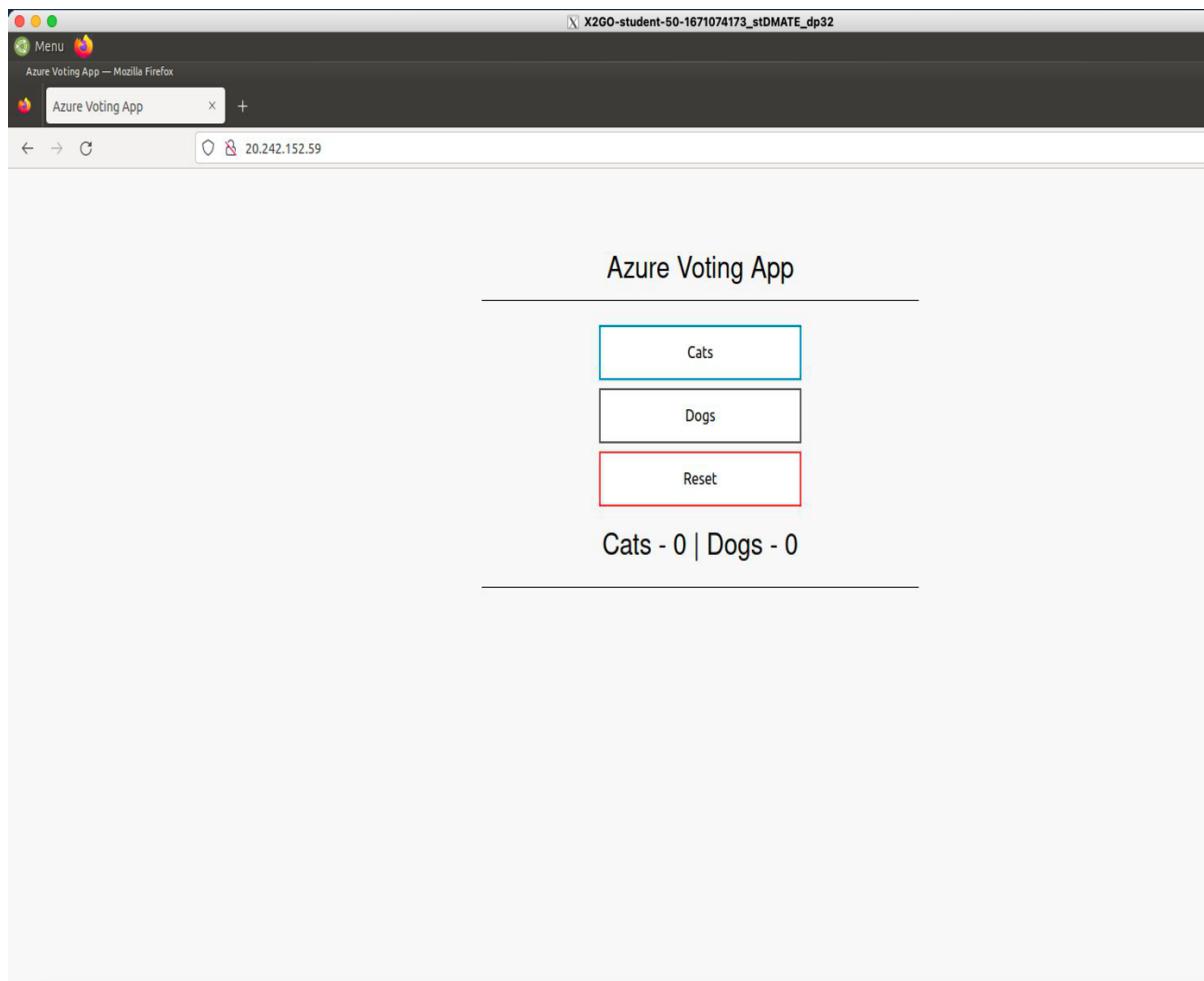


Figure 5.19

The screenshot shows the Microsoft Azure Resource Groups page. At the top, there's a search bar and a user profile for 'c2024316@newcastle.ac... NEWCASTLE UNIVERSITY (NEWC...)'. Below the header, the title 'Resource groups' is displayed with a 'Home >' link. A toolbar includes 'Create', 'Manage view', 'Refresh', 'Export to CSV', 'Open query', and 'Assign tags'. Filter options like 'Subscription equals all' and 'Location equals all' are present. The main area lists three resource groups:

Name	Subscription	Location
csc8110-kb-rg	Azure for Students	East US
MC_csc8110-kb-rg_csc8110kb-aks_eastus	Azure for Students	East US
NetworkWatcherRG	Azure for Students	East US

Figure 5.20

To view my course-work demonstration video click on the below zoom link

<https://newcastleuniversity.zoom.us/rec/share/J3A5tQfW6wJKQHsKSo5kDABemThXRcPiKWCM7RHkMib5uj2K-SdS3CXkvSB3-Lds.8xsYHYLtaSNXK8UD>

Passcode: w9cDJS&q