Cloud Computing Lab 2 Report

PES1UG19CS592

Yashi Chawla

1 Installing Docker Engine on EC2 Instance

```
pes1ug19cs592@pes1ug19cs592:~/lab2$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Figure 1: Running Docker Hello-World

2 Docker Images and Docker Files

```
pes1ug19cs592@pes1ug19cs592:~/lab2$ docker run task2
Running this inside a container !
My SRN is PES1UG19CS592
```

Figure 2: Running C Program inside Container

3 Exposing Ports and Docker Networks

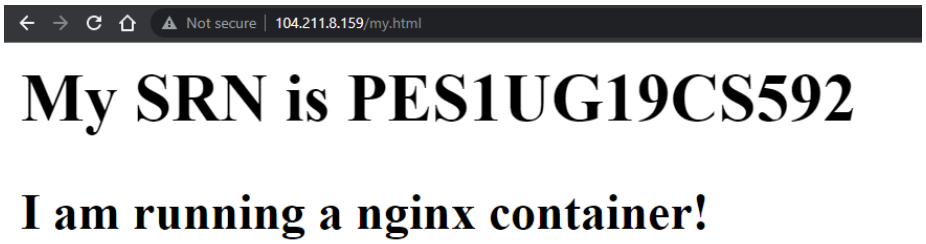

Figure 3: Accessing web page on browser

```
pes1ug19cs592@pes1ug19cs592:~/lab2/task3$ docker run -p 80:80 -t task3
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform co
nfiguration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh

10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/defa
ult.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/de
fault.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2022/02/23 09:07:28 [notice] 1#1: using the "epoll" event method
2022/02/23 09:07:28 [notice] 1#1: nginx/1.21.6
2022/02/23 09:07:28 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2022/02/23 09:07:28 [notice] 1#1: OS: Linux 5.11.0-1028-azure
2022/02/23 09:07:28 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2022/02/23 09:07:28 [notice] 1#1: start worker processes
2022/02/23 09:07:28 [notice] 1#1: start worker process 33
2022/02/23 09:07:28 [notice] 1#1: start worker process 34
2022/02/23 09:07:31 [notice] 34#34: signal 28 (SIGWINCH) received
```

Figure 4: Docker container running nginx

```
pes1ug19cs592@pes1ug19cs592:~/lab2/task3$ docker run task3a
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('6215fd16f74795d0cadad940'), 'Name:
': 'Yashi Chawla', 'SRN': 'PES1UG19CS592'}
```

Figure 5: Read and write from MongoDB

```
pes1ug19cs592@pes1ug19cs592:~/lab2/task3$ docker run -dp 27017:27017 --netw
ork=my-bridge-network --name=mongodb mongo:latest
b91499e0166239c32330ba8203c39ac7265792a66d878b3fd0d1e3f190b56fae
pes1ug19cs592@pes1ug19cs592:~/lab2/task3$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STA
TUS          PORTS                                         NAMES
b91499e01662   mongo:latest   "docker-entrypoint.s…"   54 seconds ago   Up
53 seconds   0.0.0.0:27017->27017/tcp, :::27017->27017/tcp   mongodb
```

Figure 6: MongoDB running within network

```
pes1ug19cs592@pes1ug19cs592:~/lab2/task3$ docker run --network=my-bridge-ne
twork task3b
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('6215ff5623e3a63e49406083'), 'Name:
': 'Yashi Chawla', 'SRN': 'PES1UG19CS592'}
```

Figure 7: Python file reading and writing from MongoDB within network

4 Docker Compose

```
msg":"Connection accepted","attr":{"remote":"172.19.0.3:36072","uuid":"20d6bef7-1e2b-449a-b5bf-00381e1f6663","connecti
onId":3,"connectionCount":3}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:46.872+00:00"},"s":"I",  "c":"NETWORK",  "id":51800,   "ctx":"conn3","msg
":"client metadata","attr":{"remote":"172.19.0.3:36072","client":"conn3","doc":{"driver":{"name":"PyMongo","version":"
4.0.1"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython
 3.10.2.final.0"}}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:46.873+00:00"},"s":"I",  "c":"NETWORK",  "id":51800,   "ctx":"conn2","msg
":"client metadata","attr":{"remote":"172.19.0.3:36070","client":"conn2","doc":{"driver":{"name":"PyMongo","version":"
4.0.1"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython
 3.10.2.final.0"}}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:46.873+00:00"},"s":"I",  "c":"STORAGE",  "id":20320,   "ctx":"conn2","msg
":"createCollection","attr":{"namespace":"sample_db.sample_collection","uuidDisposition":"generated","uuid":{"uuid":{"
$uuid":"8b7ef801-38ca-4daf-acbf-0f8b0f7f8d8b"}},"options":{}}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:46.901+00:00"},"s":"I",  "c":"INDEX",    "id":20345,   "ctx":"conn2","msg
":"Index build: done building","attr":{"buildUUID":null,"namespace":"sample_db.sample_collection","index":"_id_","comm
itTimestamp":null}}
pycode_1   | Inserted into the MongoDB database!
pycode_1   | Fecthed from MongoDB:  {'_id': ObjectId('6216377e62b2bef1410c4156'), 'Name:': 'Yashi Chawla', 'SRN': 'PES
1UG19CS592'}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:47.372+00:00"},"s":"I",  "c":"-",        "id":20883,   "ctx":"conn1","msg
":"Interrupted operation as its client disconnected","attr":{"opId":17}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:47.373+00:00"},"s":"I",  "c":"NETWORK",  "id":22944,   "ctx":"conn1","msg
":"Connection ended","attr":{"remote":"172.19.0.3:36068","uuid":"16d25dab-e86f-473d-8cd1-243dad842d43","connectionId":
1,"connectionCount":2}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:47.377+00:00"},"s":"I",  "c":"NETWORK",  "id":22944,   "ctx":"conn3","msg
":"Connection ended","attr":{"remote":"172.19.0.3:36072","uuid":"20d6bef7-1e2b-449a-b5bf-00381e1f6663","connectionId":
3,"connectionCount":1}}
mongodb_1  | {"t":{"$date":"2022-02-23T13:32:47.377+00:00"},"s":"I",  "c":"NETWORK",  "id":22944,   "ctx":"conn2","msg
":"Connection ended","attr":{"remote":"172.19.0.3:36070","uuid":"40b838c6-fb95-46c8-874c-ec35737d5fc7","connectionId":
```

Figure 8: Python-MongoDB application running as docker-compose

```
ame":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython 3
.10.2.final.0"}}}
pycode_1    | Inserted into the MongoDB database!
pycode_1    | Fecthed from MongoDB:  {'_id': ObjectId('6216377e62b2bef1410c4156'), 'Name:'
: 'Yashi Chawla', 'SRN': 'PES1UG19CS592'}
mongodb_1   | {"t":{"$date":"2022-02-23T13:34:57.719+00:00"},"s":"I",  "c":"NETWORK",  "id
":22943,   "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.19.0.5:4761
2","uuid":"f9b48763-0409-47c4-9298-535cd9598cdd","connectionId":4,"connectionCount":4}}
mongodb_1   | {"t":{"$date":"2022-02-23T13:34:57.720+00:00"},"s":"I",  "c":"NETWORK",  "id
":51800,   "ctx":"conn4","msg":"client metadata","attr":{"remote":"172.19.0.5:47612","cli
ent":"conn4","doc":{"driver":{"name":"PyMongo","version":"4.0.1"},"os":{"type":"Linux","n
ame":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython 3
.10.2.final.0"}}}
```

Figure 9: Application read and write after scaling – Container 1

```
ame":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython 3
.10.2.final.0"}}}
pycode_2    | Inserted into the MongoDB database!
pycode_2    | Fecthed from MongoDB:  {'_id': ObjectId('6216377e62b2bef1410c4156'), 'Name:'
: 'Yashi Chawla', 'SRN': 'PES1UG19CS592'}
mongodb_1   | {"t":{"$date":"2022-02-23T13:34:57.737+00:00"},"s":"I",  "c":"NETWORK",  "id
":22943,   "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.19.0.4:3547
4","uuid":"8c6a06ec-6716-4920-adc6-d7a8546fc9b3","connectionId":7,"connectionCount":7}}
```

Figure 10: Application read and write after scaling – Container 2

```
ame":"Linux","architecture":"x86_64","version":"5.11.0-1028-azure"},"platform":"CPython 3
.10.2.final.0"}}}
pycode_3    | Inserted into the MongoDB database!
pycode_3    | Fecthed from MongoDB:  {'_id': ObjectId('6216377e62b2bef1410c4156'), 'Name:'
: 'Yashi Chawla', 'SRN': 'PES1UG19CS592'}
mongodb_1   | {"t":{"$date":"2022-02-23T13:34:58.021+00:00"},"s":"I",  "c":"-",        "id
":20883,   "ctx":"conn1","msg":"Interrupted operation as its client disconnected","attr":
{"opId":14}}
```

Figure 11: Application read and write after scaling – Container 3