Faculty of Engineering and Applied Science

SOFE 4790U Distributed Systems

Lab 3 Individual: Deploying a Circuit Breaking ambassador and a Function-as-a-Service (FaaS)

Group 19

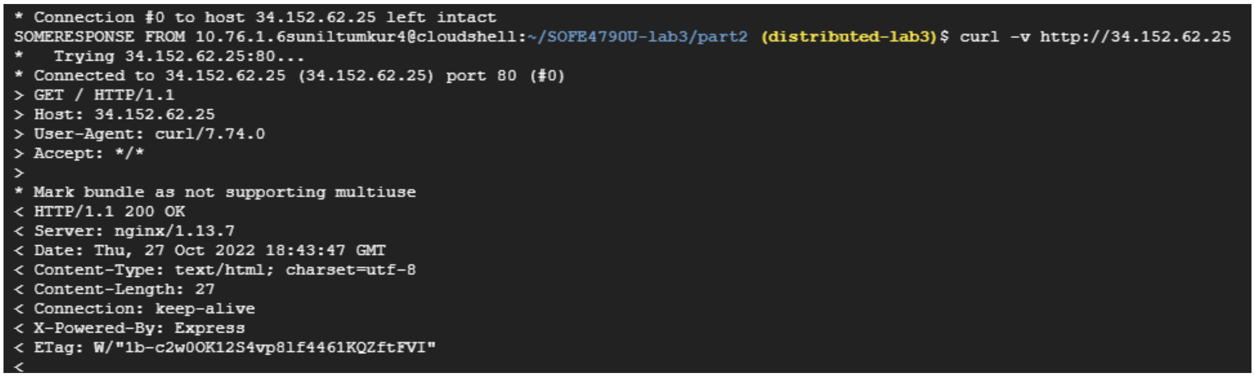
Sunil Tumkur 100620430

https://github.com/sunilt4/Distributed-Systems

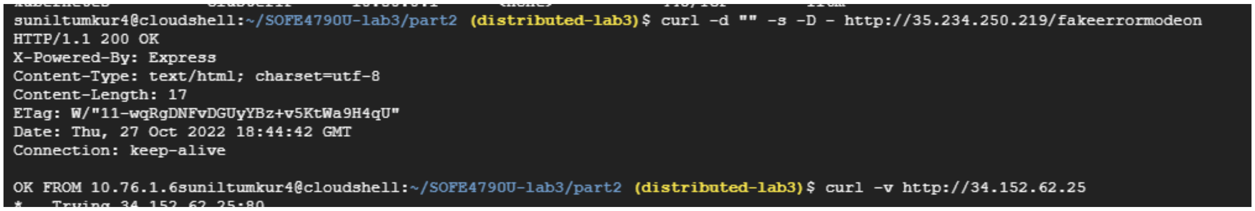
Part 1:

See Discussion below

Part 2:

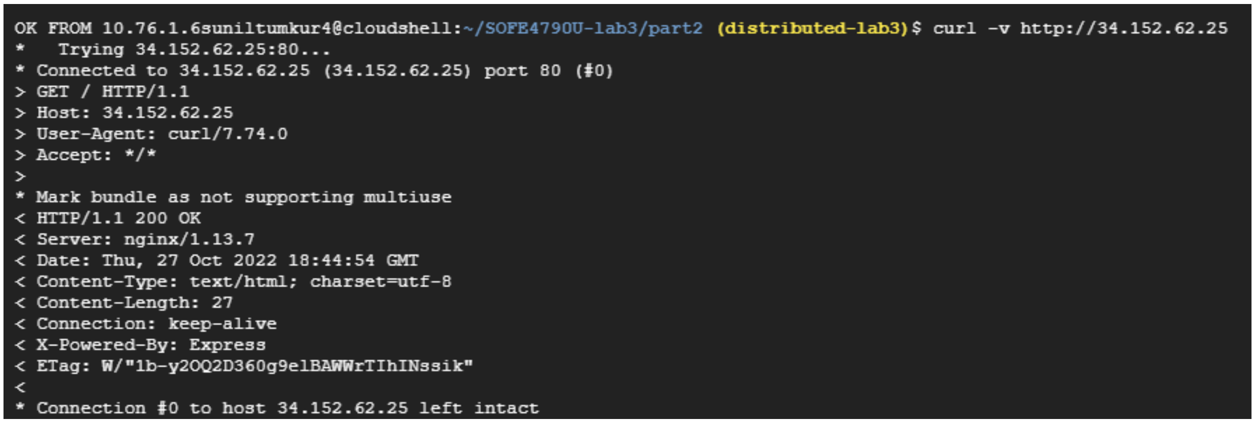


```
* Connection #0 to host 34.152.62.25 left intact
SOMERESPONSE FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -v http://34.152.62.25
*   Trying 34.152.62.25:80...
* Connected to 34.152.62.25 (34.152.62.25) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.62.25
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Thu, 27 Oct 2022 18:43:47 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 27
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1b-c2w0OK12S4vp8lf4461KQZftFVI"
<
```

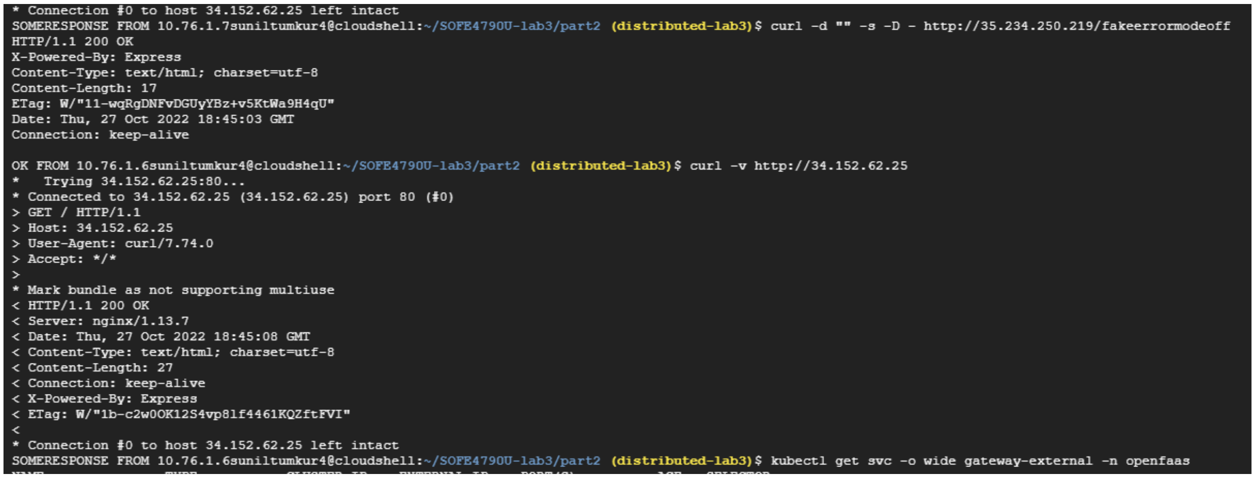


```
suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -d "" -s -D - http://35.234.250.219/fakeerrormodeon
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 17
ETag: W/"11-wqRgDNFvDGUyYBz+v5KtWa9H4qU"
Date: Thu, 27 Oct 2022 18:44:42 GMT
Connection: keep-alive

OK FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -v http://34.152.62.25
*   Trying 34.152.62.25:80
```



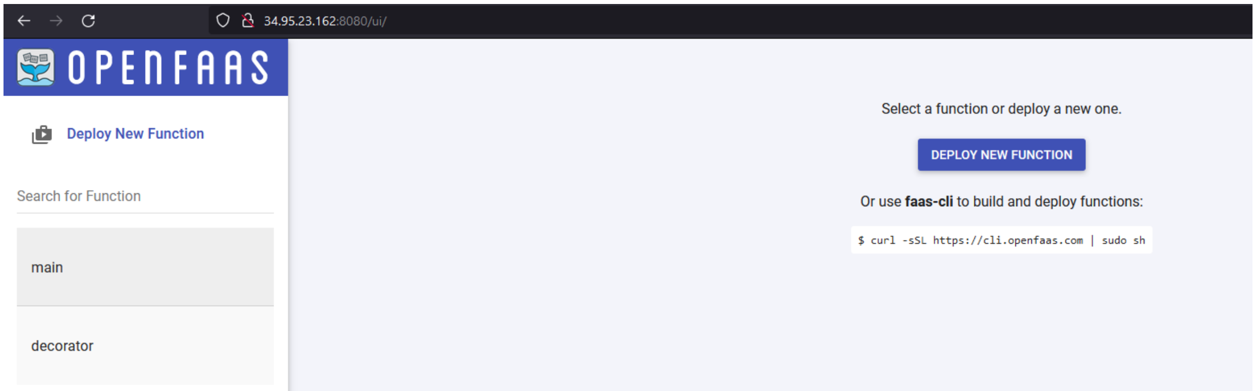
```
OK FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -v http://34.152.62.25
*   Trying 34.152.62.25:80...
* Connected to 34.152.62.25 (34.152.62.25) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.62.25
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Thu, 27 Oct 2022 18:44:54 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 27
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1b-y2OQ2D360g9elBAWWrTIhINssik"
<
* Connection #0 to host 34.152.62.25 left intact
```



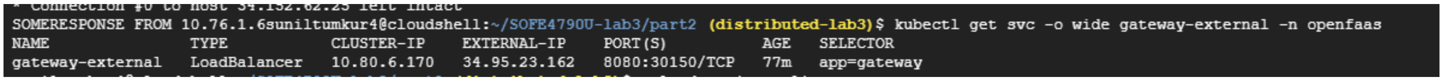
```
* Connection #0 to host 34.152.62.25 left intact
SOMERESPONSE FROM 10.76.1.7suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -d "" -s -D - http://35.234.250.219/fakeerrormodeoff
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 17
ETag: W/"11-wqRgDNFvDGUyYBz+v5KtWa9H4qU"
Date: Thu, 27 Oct 2022 18:45:03 GMT
Connection: keep-alive

OK FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ curl -v http://34.152.62.25
*   Trying 34.152.62.25:80...
* Connected to 34.152.62.25 (34.152.62.25) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.62.25
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Thu, 27 Oct 2022 18:45:08 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 27
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1b-c2w0OK12S4vp81f4461KQZftFVI"
<
* Connection #0 to host 34.152.62.25 left intact
SOMERESPONSE FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ kubectl get svc -o wide gateway-external -n openfaas
```

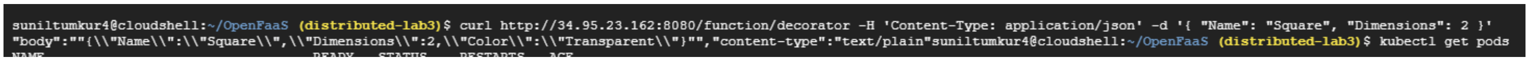
Part 3:



```
$ curl -sSL https://cli.openfaas.com | sudo sh
```



```
SOMERESPONSE FROM 10.76.1.6suniltumkur4@cloudshell:~/SOFE4790U-lab3/part2 (distributed-lab3)$ kubectl get svc -o wide gateway-external -n openfaas
NAME               TYPE           CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE   SELECTOR
gateway-external   LoadBalancer   10.80.6.170   34.95.23.162   8080:30150/TCP   77m   app=gateway
```



```
suniltumkur4@cloudshell:~/OpenFaaS (distributed-lab3)$ curl http://34.95.23.162:8080/function/decorator -H 'Content-Type: application/json' -d '{ "Name": "Square", "Dimensions": 2 }'
"body":""{\\"Name\\":\\"Square\\",\\"Dimensions\\":2,\\"Color\\":\\"Transparent\\"}"","content-type":"text/plain"suniltumkur4@cloudshell:~/OpenFaaS (distributed-lab3)$ kubectl get pods
```

Discussion

Article: https://learn.microsoft.com/en-us/azure/architecture/patterns/health-endpoint-monitoring

1) Problem being solved
   - In cloud applications there may be cases where we cannot monitor the services being ran in the cloud
   - We need to find a way to monitor these services to make sure that they stay online and available

2) How is it solved?
   - We can solve this problem by using health monitoring techniques. In order to accomplish this we send requests to an endpoint of the application we are working with. After sending these requests we will perform a check to see if we get back the intended result [1]
   - We can check if the status of the application returns a successful HTTP response, or if it returns an error
   - Check for latency to see if everything is running smoothly.
   - Checking the cloud storage and database to see its availability and response time.
   - The response that is sent back from the health check, we can also check for the body so see if it matches information we are looking for in an application to ensure its working successfully.

3) Requirements for the pattern
   - API to act in the middle between making the health check request which needs to connect to the backend to retrieve the data to make sure we our test matches the expected output generated from the backend

4) Which of these requirements can be achieved by the procedures shown in parts 2 and 3?
   - Making requests

Design

Why are Kubernetes persistent volumes so important?

- It is important because we need a way for the data to persist when our container goes away. For example if we get rid of the container or even restart it. The pods can then still have access to the data.

How to implement this?

- Need to create a PVC (Persistent volume chain) and then point the volume we created to a certain within the pod so it can reach it [2].
- Storage plugins. Kubernetes gives you the ability to use plugins in which it will manage and take the volumes from the user/host or consume the existing storage [3]
- Data volume containers [3]
- Directory mounts [3]

Provide an example in which persistent volumes are needed?

- With kubernetes, applications that are configured are usually stateless, but persistent volumes are needed when we are dealing with stateful applications.
- Example would be a shopping cart application

Video Link 1:
https://drive.google.com/file/d/17H2AWC0IRoUtm2VCXPr_6n1INSBS5Io7/view?usp=share_link

Video Link 2:

https://drive.google.com/file/d/1OcKtkL7_c3Sdr0H325xy6HjNYvYVdqNA/view?usp=share_link

References

[1] EdPrice-MSFT, “Health Endpoint Monitoring Pattern - Azure Architecture Center,” *Azure Architecture Center | Microsoft Learn*. [Online]. Available: https://learn.microsoft.com/en-us/azure/architecture/patterns/health-endpoint-monitoring. [Accessed: 20-Oct-2022].

[2] “Kubernetes persistent volumes: Tutorial and examples,” *RSS*. [Online]. Available: https://www.containiq.com/post/kubernetes-persistent-volumes. [Accessed: 20-Oct-2022].

[3] “What are kubernetes persistent volumes?,” *NetApp*. [Online]. Available: https://www.netapp.com/devops-solutions/what-is-kubernetes-persistent-volumes/. [Accessed: 20-Oct-2022].