Faculty of Engineering and Applied Science

SOFE 4790U Distributed Systems

CRN 44425

Lab # 3

Mamun Hossain - 100553073

Introduction:

This lab gives us an introduction to understand how to deploy a circuit breaker file and utilize the function as a service. We have also learned how to access FaaS services via the UI and invoke the requests accordingly.

Part 1: How to solve the problem and the requirements:

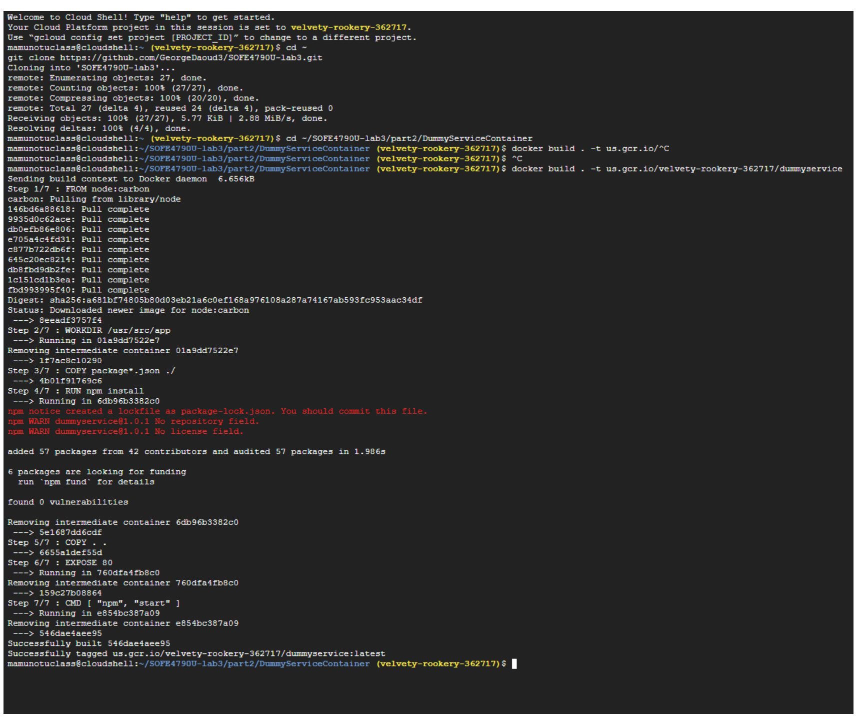
The problem at hand was the many factors that had affected cloud-hosted applications, and those factors were such as network latency, performance and availability of the underlying compute and storage systems and the network between them.

The way this problem was solved was by implementing health monitoring of the clusters and the nodes to each endpoint of the application. It will check if the application or services respond to the request of the health monitor and it will also analyze the endpoint of the tool or framework that performs the check.

The requirements to do this health monitor check is to implement an Agent that will have the appropriate certificates and in doing so, will send pings to the application endpoint to ensure that over the network everything is working, and will check for 200 (OK) status to be returned to ensure data errors are minimized.

Part 2: Procedure:

Here we can see that we initialize the setup with the required clusters and then clone the repository and input the dummy service containers and services and then push them with the docker command.



Here we are able to see the projects list to see what is in the gcloud.

```
Successfully tagged us.gcr.io/velvety-rookery-362717/dummyservice:latest
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (velvety-rookery-362717)$ gcloud projects list
PROJECT_ID: mamunhossain
NAME: MamunHossain
PROJECT_NUMBER: 418585279789

PROJECT_ID: sunny-effort-366715
NAME: SOFE4640 GoogleMapKey
PROJECT_NUMBER: 685349366104
```

Afterwards we can push it with the docker command to see what layers need to be pushed.

```
PROJECT_ID: velvety-rookery-362717
NAME: My First Project
PROJECT_NUMBER: 38427508957
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (velvety-rookery-362717)$ docker push us.gcr.io/velvety-rookery-362717/dummyservice
Using default tag: latest
The push refers to repository [us.gcr.io/velvety-rookery-362717/dummyservice]
fdb15b87583c: Pushed
abd302d5b471: Pushed
5f873b52387a: Pushed
17e0dfafdf13: Pushed
423451ed44f2: Layer already exists
b2aaf85d6633: Layer already exists
88601a85ce11: Layer already exists
42f9c2f9c08e: Layer already exists
99e8bd3efaaf: Layer already exists
bee1e39d7c3a: Layer already exists
1f59a4b2e206: Layer already exists
0ca7f54856c0: Layer already exists
ebb9ae013834: Layer already exists
latest: digest: sha256:a70f5b4579e2eb15b7dedeb145f33015a63db97bfb34533bfbf770d151aa38e6 size: 3048
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (velvety-rookery-362717)$
```

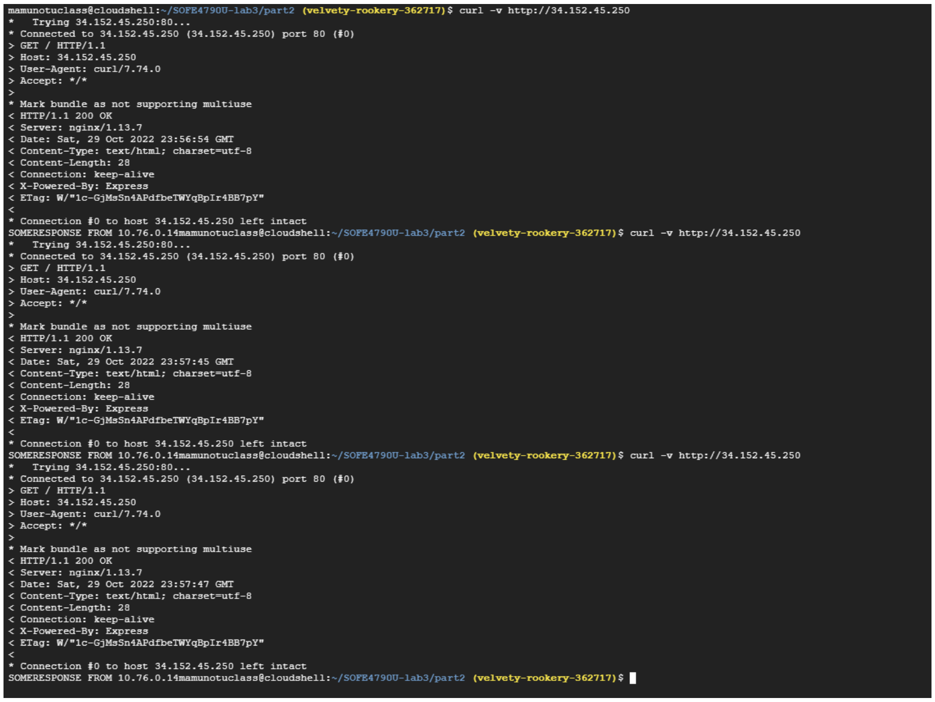
We can now expose the deployments after we create the backup deployments and while we go into the editor to make the required adjustments to the files provided in the git clone.

```
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ cd ~/SOFE4790U-lab3/part2/
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
error: error validating "backup-deployment.yaml": error validating data: ValidationError(Deployment.spec.template.spec.containers[0]): unknown field "containerPort" in io.k8s.api.core.v1.Container; if you choose to ignore these errors, turn validation off with --validate=false
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
error: error parsing backup-deployment.yaml: error converting YAML to JSON: yaml: line 20: did not find expected key
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
error: error validating "backup-deployment.yaml": error validating data: ValidationError(Deployment.spec.template.spec.containers[0]): unknown field "containerPort" in io.k8s.api.core.v1.Container; if you choose to ignore these errors, turn validation off with --validate=false
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ --validation=false
-bash: --validation=false: command not found
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
error: error parsing backup-deployment.yaml: error converting YAML to JSON: yaml: line 20: did not find expected key
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
error: error parsing backup-deployment.yaml: error converting YAML to JSON: yaml: line 20: did not find expected key
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl create -f backup-deployment.yaml
deployment.apps/backup-deployment created
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl expose deployment backup-deployment --port=80 --type=LoadBalancer --name backup-deployment
service/backup-deployment exposed
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl get pods
NAME                                      READY   STATUS    RESTARTS   AGE
ambassador-deployment-66db4f7766-56566    1/1     Running   0          21d
ambassador-deployment-66db4f7766-9chql    1/1     Running   0          21d
backup-deployment-77c7b595b7-28z58        1/1     Running   0          35s
dummy-deployment-676db84ff5-12hgd         1/1     Running   0          7m26s
experiment-deployment-7b47cbd668-q48xj    1/1     Running   0          21d
experiment-deployment-7b47cbd668-w86zb    1/1     Running   0          21d
loadbalancer-deployment-6676f9ccf6-7qsxs  1/1     Running   0          21d
loadbalancer-deployment-6676f9ccf6-fhjqc  1/1     Running   0          21d
loadbalancer-deployment-6676f9ccf6-wch4b  1/1     Running   0          21d
php-apache-779cd44bdc-sjqzh               1/1     Running   0          21d
web-deployment-6fdbb5c6bb-v9dqh           1/1     Running   0          21d
web-deployment-6fdbb5c6bb-zb8wl           1/1     Running   0          21d
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ kubectl get services
NAME                      TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)          AGE
ambassador-deployment     LoadBalancer   10.80.1.30     34.152.52.239   80:30938/TCP     21d
backup-deployment         LoadBalancer   10.80.13.62    <pending>       80:32107/TCP     18s
dummy-deployment          LoadBalancer   10.80.7.31     34.152.54.18    80:31132/TCP     6m49s
experiment-deployment     ClusterIP      10.80.15.8     <none>          80/TCP           21d
kubernetes                ClusterIP      10.80.0.1      <none>          443/TCP          21d
loadbalancer-deployment   LoadBalancer   10.80.14.172   35.203.101.45   8080:31058/TCP   21d
php-apache                ClusterIP      10.80.13.15    <none>          80/TCP           21d
web-deployment            ClusterIP      10.80.14.54    <none>          80/TCP           21d
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$
```

Altered files for backup-deployment.yaml

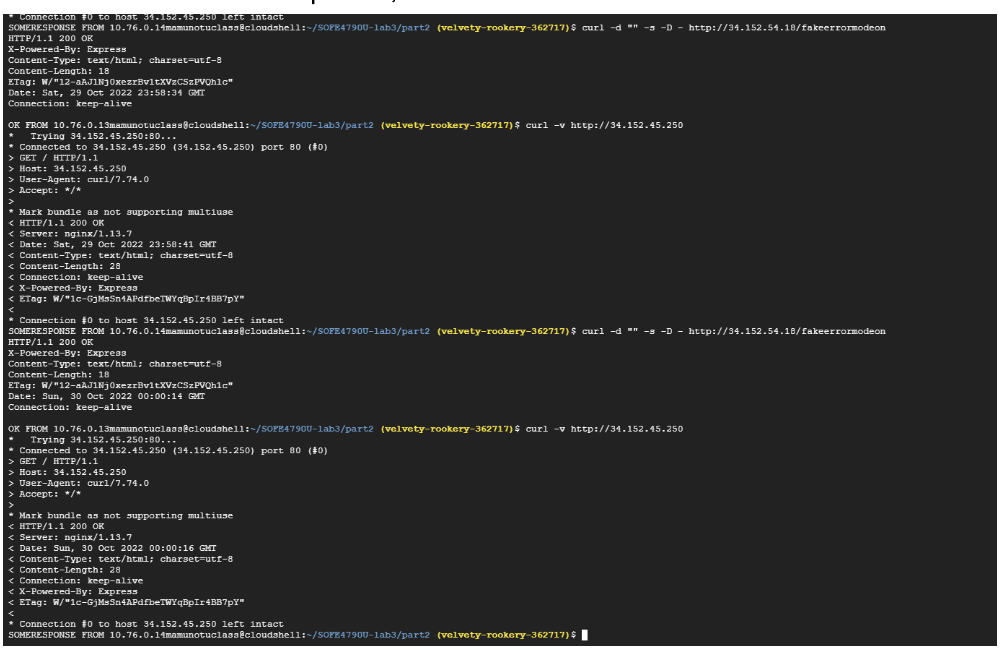
```
SOFE4790U-lab3 > part2 > backup-deployment.yaml > ...
12      run: backup-deployment
13    nplate:
14    netadata:
15      labels:
16        run: backup-deployment
17    spec:
18      containers:
19      - name: backup-depoyment
20        image: us.gcr.io/vast-alcove-367116/dummyservice
21        ports:
22        - containerPort: 80
23        livenessProbe:
24          httpGet:
25            # The /alive endpoint is the one we will not touch in our test case, a
26            path: /alive
27            port: 80
```

Here, we can see after we create the configmap, we can use the curl command to execute the circuit breaker UP and check it in the local machine in the last line of each response.



```
mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -v http://34.152.45.250
*   Trying 34.152.45.250:80...
* Connected to 34.152.45.250 (34.152.45.250) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.45.250
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Sat, 29 Oct 2022 23:56:54 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-GjMsSn4APdfbeTWYqBpIr4BB7pY"
<
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -v http://34.152.45.250
*   Trying 34.152.45.250:80...
* Connected to 34.152.45.250 (34.152.45.250) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.45.250
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Sat, 29 Oct 2022 23:57:45 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-GjMsSn4APdfbeTWYqBpIr4BB7pY"
<
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -v http://34.152.45.250
*   Trying 34.152.45.250:80...
* Connected to 34.152.45.250 (34.152.45.250) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.45.250
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Sat, 29 Oct 2022 23:57:47 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-GjMsSn4APdfbeTWYqBpIr4BB7pY"
<
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$
```

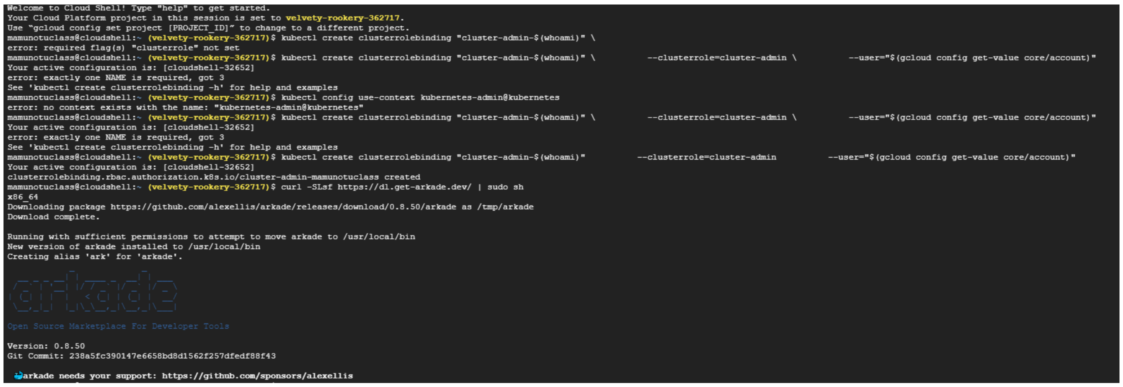
While we see the last line remaining unchanged after each test, but the dummy service having a different line in each response, check test the circuit breaker one more time.



```
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -d "" -s -D - http://34.152.54.18/fakeerrormodeon
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 18
ETag: W/"12-aAJ1Nj0xezrBv1tXVzCSzPVQh1c"
Date: Sat, 29 Oct 2022 23:58:34 GMT
Connection: keep-alive

OK FROM 10.76.0.13mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -v http://34.152.45.250
*   Trying 34.152.45.250:80...
* Connected to 34.152.45.250 (34.152.45.250) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.45.250
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Sat, 29 Oct 2022 23:58:41 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-GjMsSn4APdfbeTWYqBpIr4BB7pY"
<
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -d "" -s -D - http://34.152.54.18/fakeerrormodeon
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 18
ETag: W/"12-aAJ1Nj0xezrBv1tXVzCSzPVQh1c"
Date: Sun, 30 Oct 2022 00:00:14 GMT
Connection: keep-alive

OK FROM 10.76.0.13mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$ curl -v http://34.152.45.250
*   Trying 34.152.45.250:80...
* Connected to 34.152.45.250 (34.152.45.250) port 80 (#0)
> GET / HTTP/1.1
> Host: 34.152.45.250
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.13.7
< Date: Sun, 30 Oct 2022 00:00:16 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-GjMsSn4APdfbeTWYqBpIr4BB7pY"
<
* Connection #0 to host 34.152.45.250 left intact
SOMERESPONSE FROM 10.76.0.14mamunotuclass@cloudshell:~/SOFE4790U-lab3/part2 (velvety-rookery-362717)$
```

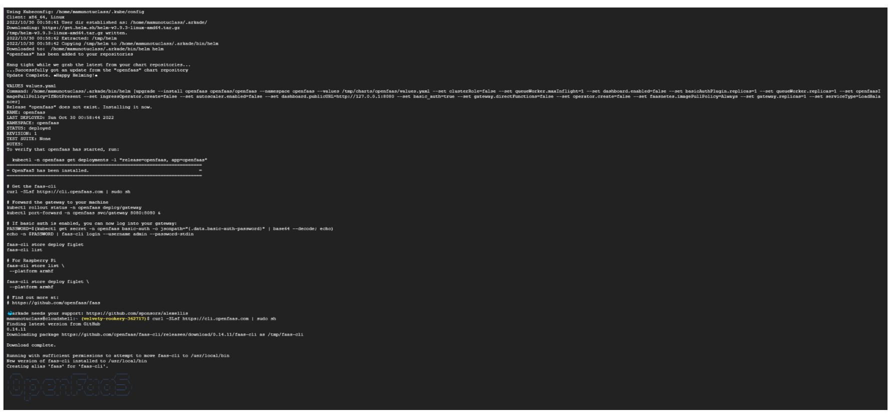
Part 3:

We initialized the cluster admin role binding to then deploy OpenFaaS through the GKE, afterwards installing FaaS and as we can see below the confirmation of that and ArKade. We then use the commands to test that FaaS has been successfully initialized.



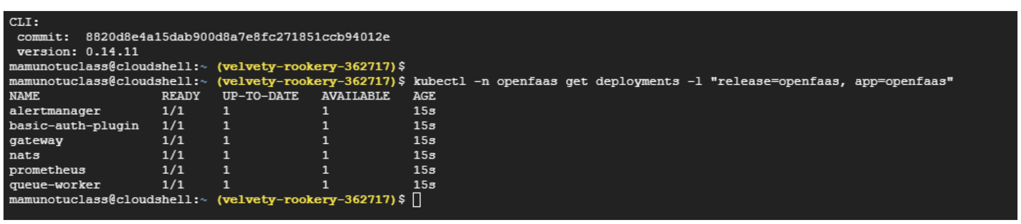
```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to velvety-rookery-362717.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl create clusterrolebinding "cluster-admin-$(whoami)" \
error: required flag(s) "clusterrole" not set
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl create clusterrolebinding "cluster-admin-$(whoami)" \         --clusterrole=cluster-admin \         --user="$(gcloud config get-value core/account)"
Your active configuration is: [cloudshell-32652]
error: exactly one NAME is required, got 3
See 'kubectl create clusterrolebinding -h' for help and examples
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl config use-context kubernetes-admin@kubernetes
error: no context exists with the name: "kubernetes-admin@kubernetes"
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl create clusterrolebinding "cluster-admin-$(whoami)" \         --clusterrole=cluster-admin \         --user="$(gcloud config get-value core/account)"
Your active configuration is: [cloudshell-32652]
error: exactly one NAME is required, got 3
See 'kubectl create clusterrolebinding -h' for help and examples
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl create clusterrolebinding "cluster-admin-$(whoami)"         --clusterrole=cluster-admin         --user="$(gcloud config get-value core/account)"
Your active configuration is: [cloudshell-32652]
clusterrolebinding.rbac.authorization.k8s.io/cluster-admin-mamunotuclass created
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ curl -SLsf https://dl.get-arkade.dev/ | sudo sh
x86_64
Downloading package https://github.com/alexellis/arkade/releases/download/0.8.50/arkade as /tmp/arkade
Download complete.

Running with sufficient permissions to attempt to move arkade to /usr/local/bin
New version of arkade installed to /usr/local/bin
Creating alias 'ark' for 'arkade'.

Open Source Marketplace For Developer Tools

Version: 0.8.50
Git Commit: 238a5fc390147e6658bd8d1562f257dfedf88f43

 arkade needs your support: https://github.com/sponsors/alexellis
```

The installation process of FaaS and ArKade.



```
Using Kubeconfig: /home/mamunotuclass/.kube/config
Client: x86_64, Linux
2022/10/30 00:58:41 User dir established as: /home/mamunotuclass/.arkade/
Downloading: https://get.helm.sh/helm-v3.9.3-linux-amd64.tar.gz
/tmp/helm-v3.9.3-linux-amd64.tar.gz written.
2022/10/30 00:58:42 Extracted: /tmp/helm
2022/10/30 00:58:42 Copying /tmp/helm to /home/mamunotuclass/.arkade/bin/helm
Downloaded to:  /home/mamunotuclass/.arkade/bin/helm helm
"openfaas" has been added to your repositories

Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "openfaas" chart repository
Update Complete. *Happy Helming!*

VALUES values.yaml
Command: /home/mamunotuclass/.arkade/bin/helm [upgrade --install openfaas openfaas/openfaas --namespace openfaas --values /tmp/charts/openfaas/values.yaml --set clusterRole=false --set queueWorker.maxInflight=1 --set dashboard.enabled=false --set basicAuthPlugin.replicas=1 --set queueWorker.replicas=1 --set openfaasImagePullPolicy=IfNotPresent --set ingressOperator.create=false --set autoscaler.enabled=false --set dashboard.publicURL=http://127.0.0.1:8080 --set basic_auth=true --set gateway.directFunctions=false --set operator.create=false --set faasnetes.imagePullPolicy=Always --set gateway.replicas=1 --set serviceType=LoadBalancer]
Release "openfaas" does not exist. Installing it now.
NAME: openfaas
LAST DEPLOYED: Sun Oct 30 00:58:44 2022
NAMESPACE: openfaas
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
To verify that openfaas has started, run:

  kubectl -n openfaas get deployments -l "release=openfaas, app=openfaas"
=======================================================================
= OpenFaaS has been installed.                                        =
=======================================================================

# Get the faas-cli
curl -SLsf https://cli.openfaas.com | sudo sh

# Forward the gateway to your machine
kubectl rollout status -n openfaas deploy/gateway
kubectl port-forward -n openfaas svc/gateway 8080:8080 &

# If basic auth is enabled, you can now log into your gateway:
PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)
echo -n $PASSWORD | faas-cli login --username admin --password-stdin

faas-cli store deploy figlet
faas-cli list

# For Raspberry Pi
faas-cli store list \
 --platform armhf

faas-cli store deploy figlet \
 --platform armhf

# Find out more at:
# https://github.com/openfaas/faas

 arkade needs your support: https://github.com/sponsors/alexellis
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ curl -SLsf https://cli.openfaas.com | sudo sh
Finding latest version from GitHub
0.14.11
Downloading package https://github.com/openfaas/faas-cli/releases/download/0.14.11/faas-cli as /tmp/faas-cli

Download complete.

Running with sufficient permissions to attempt to move faas-cli to /usr/local/bin
New version of faas-cli installed to /usr/local/bin
Creating alias 'faas' for 'faas-cli'.
```

Successful verification of FaaS.

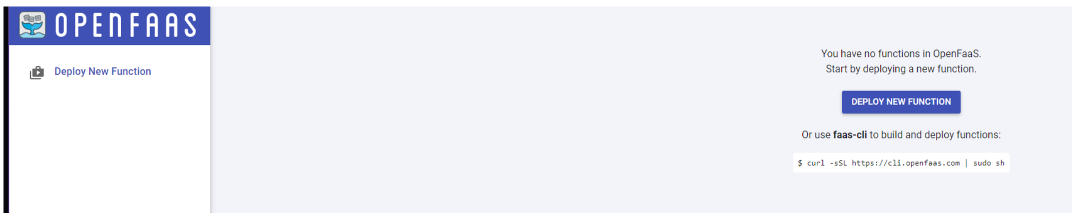


```
CLI:
 commit:  8820d8e4a15dab900d8a7e8fc271851ccb94012e
 version: 0.14.11
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl -n openfaas get deployments -l "release=openfaas, app=openfaas"
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
alertmanager        1/1     1            1           15s
basic-auth-plugin   1/1     1            1           15s
gateway             1/1     1            1           15s
nats                1/1     1            1           15s
prometheus          1/1     1            1           15s
queue-worker        1/1     1            1           15s
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$
```

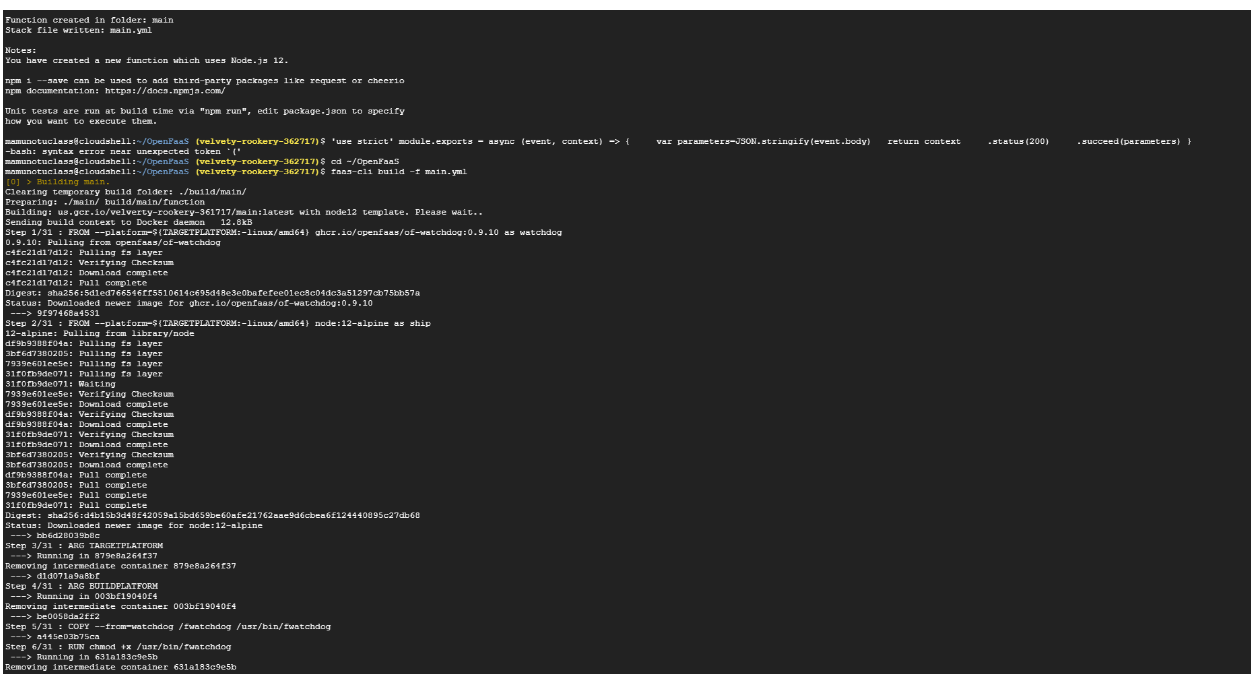
Using the instructions, we will use the commands to find out the password to our localized FaaS UI and from there we will open it after the installation.

```
CLI:
 commit:  8820d8e4a15dab900d8a7e8fc271851ccb94012e
 version: 0.14.11
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl -n openfaas get deployments -l "release=openfaas, app=openfaas"
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
alertmanager        1/1     1            1           15s
basic-auth-plugin   1/1     1            1           15s
gateway             1/1     1            1           15s
nats                1/1     1            1           15s
prometheus          1/1     1            1           15s
queue-worker        1/1     1            1           15s
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl rollout status -n openfaas deploy/gateway
deployment "gateway" successfully rolled out
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl get svc -o wide gateway-external -n openfaas
NAME               TYPE           CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE    SELECTOR
gateway-external   LoadBalancer   10.80.1.204   35.203.95.73   8080:32225/TCP   6m9s   app=gateway
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ kubectl get services
NAME                       TYPE           CLUSTER-IP     EXTERNAL-IP      PORT(S)          AGE
ambassador-deployment      LoadBalancer   10.80.1.30     34.152.52.239    80:30938/TCP     21d
backup-deployment          LoadBalancer   10.80.13.62    35.234.247.71    80:32107/TCP     110m
circuitbreaker             LoadBalancer   10.80.13.78    34.152.45.250    80:30484/TCP     109m
dummy-deployment           LoadBalancer   10.80.7.31     34.152.54.18     80:31132/TCP     117m
experiment-deployment      ClusterIP      10.80.15.8     <none>           80/TCP           21d
kubernetes                 ClusterIP      10.80.0.1      <none>           443/TCP          21d
loadbalancer-deployment    LoadBalancer   10.80.14.172   35.203.101.45    8080:31058/TCP   21d
php-apache                 ClusterIP      10.80.13.15    <none>           80/TCP           21d
web-deployment             ClusterIP      10.80.14.54    <none>           80/TCP           21d
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ export OPENFAAS_URL="35.203.95.73:8080"
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ export OPENFAAS_URL="35.203.95.73:8080" PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo) echo $PASSWORD echo -n $PASSWORD | faas-cli login --username admin --password-stdin
-bash: export: `-n': not a valid identifier
Must provide a non-empty password via --password or --password-stdin
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ export OPENFAAS_URL="35.203.95.73:8080" PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo) echo $PASSWORD echo -n $PASSWORD | faas-cli login --username admin --password-stdin
-bash: export: `-n': not a valid identifier
Must provide a non-empty password via --password or --password-stdin
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ export OPENFAAS_URL="35.203.95.73:8080"
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ echo $PASSWORD
O8L2T4PTk11i7wYisy9K9O6C0
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ echo -n $PASSWORD | faas-cli login --username admin --password-stdin
Calling the OpenFaaS server to validate the credentials...
WARNING! You are not using an encrypted connection to the gateway, consider using HTTPS.
credentials saved for admin http://35.203.95.73:8080
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$ ^C
mamunotuclass@cloudshell:~ (velvety-rookery-362717)$
```

Verification of successful entry into FaaS UI.



After verification, we can go into the editor and alter the main/handler.js file to convert the event.body parameters of the JSON object to a string, and get the command that we want.



```
Function created in folder: main
Stack file written: main.yml

Notes:
You have created a new function which uses Node.js 12.

npm i --save can be used to add third-party packages like request or cheerio
npm documentation: https://docs.npmjs.com/

Unit tests are run at build time via "npm run", edit package.json to specify
how you want to execute them.

mamunotuclass@cloudshell:~/OpenFaaS (velvety-rookery-362717)$ 'use strict' module.exports = async (event, context) => {     var parameters=JSON.stringify(event.body)   return context    .status(200)    .succeed(parameters) }
-bash: syntax error near unexpected token `('
mamunotuclass@cloudshell:~/OpenFaaS (velvety-rookery-362717)$ cd ~/OpenFaaS
mamunotuclass@cloudshell:~/OpenFaaS (velvety-rookery-362717)$ faas-cli build -f main.yml
[0] > Building main.
Clearing temporary build folder: ./build/main/
Preparing: ./main/ build/main/function
Building: us.gcr.io/velverty-rookery-361717/main:latest with node12 template. Please wait..
Sending build context to Docker daemon   12.8kB
Step 1/31 : FROM --platform=${TARGETPLATFORM:-linux/amd64} ghcr.io/openfaas/of-watchdog:0.9.10 as watchdog
0.9.10: Pulling from openfaas/of-watchdog
c4fc21d17d12: Pulling fs layer
c4fc21d17d12: Verifying Checksum
c4fc21d17d12: Download complete
c4fc21d17d12: Pull complete
Digest: sha256:5d1ed766546ff5510614c695d48e3e0bafefee01ec8c04dc3a51297cb75bb57a
Status: Downloaded newer image for ghcr.io/openfaas/of-watchdog:0.9.10
 ---> 9f97468a4531
Step 2/31 : FROM --platform=${TARGETPLATFORM:-linux/amd64} node:12-alpine as ship
12-alpine: Pulling from library/node
df9b9388f04a: Pulling fs layer
3bf6d7380205: Pulling fs layer
7939e601ee5e: Pulling fs layer
31f0fb9de071: Pulling fs layer
31f0fb9de071: Waiting
7939e601ee5e: Verifying Checksum
7939e601ee5e: Download complete
df9b9388f04a: Verifying Checksum
df9b9388f04a: Download complete
31f0fb9de071: Verifying Checksum
31f0fb9de071: Download complete
3bf6d7380205: Verifying Checksum
3bf6d7380205: Download complete
df9b9388f04a: Pull complete
3bf6d7380205: Pull complete
7939e601ee5e: Pull complete
31f0fb9de071: Pull complete
Digest: sha256:d4b15b3d48f42059a15bd659be60afe21762aae9d6cbea6f124440895c27db68
Status: Downloaded newer image for node:12-alpine
 ---> bb6d28039b8c
Step 3/31 : ARG TARGETPLATFORM
 ---> Running in 879e8a264f37
Removing intermediate container 879e8a264f37
 ---> d1d071a9a8bf
Step 4/31 : ARG BUILDPLATFORM
 ---> Running in 003bf19040f4
Removing intermediate container 003bf19040f4
 ---> be0058da2ff2
Step 5/31 : COPY --from=watchdog /fwatchdog /usr/bin/fwatchdog
 ---> a445e03b75ca
Step 6/31 : RUN chmod +x /usr/bin/fwatchdog
 ---> Running in 631a183c9e5b
Removing intermediate container 631a183c9e5b
```

Here we are building the main.yml file after editing the FaaS main/handler.js file.

```
Digest: sha256:d4b15b3d48f42059a15bd659be60afe21762aae9d6cbea6f124440895c27db68
Status: Downloaded newer image for node:12-alpine
 ---> bb6d28039b8c
Step 3/31 : ARG TARGETPLATFORM
 ---> Running in 879e8a264f37
Removing intermediate container 879e8a264f37
 ---> d1d071a9a8bf
Step 4/31 : ARG BUILDPLATFORM
 ---> Running in 003bf19040f4
Removing intermediate container 003bf19040f4
 ---> be0058da2ff2
Step 5/31 : COPY --from=watchdog /fwatchdog /usr/bin/fwatchdog
 ---> a445e03b75ca
Step 6/31 : RUN chmod +x /usr/bin/fwatchdog
 ---> Running in 631a183c9e5b
Removing intermediate container 631a183c9e5b
 ---> dcb64a05fe38
Step 7/31 : RUN apk --no-cache add curl ca-certificates     && addgroup -S app && adduser -S -g app app
 ---> Running in 81e4b0a9eef0
fetch https://dl-cdn.alpinelinux.org/alpine/v3.15/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.15/community/x86_64/APKINDEX.tar.gz
(1/5) Installing ca-certificates (20220614-r0)
(2/5) Installing brotli-libs (1.0.9-r5)
(3/5) Installing nghttp2-libs (1.46.0-r0)
(4/5) Installing libcurl (7.80.0-r4)
(5/5) Installing curl (7.80.0-r4)
Executing busybox-1.34.1-r5.trigger
```

Edited OpenFaaS handler.js file.

OpenFaaS > main > handler.js > ...

```
'use strict'

module.exports = async (event, context) => {
    var parameters=JSON.stringify(event.body)
  return context
    .status(200)
    .succeed(parameters)
}
```

We then run the command and get the string input as desired

```
mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ curl http://34.95.52.188:8080/function/main -H 'Content-Type: application/json' -d '{ "Name": "Square", "Color": "Red", "Dimensions": 2
 }'
{"Name":"Square","Color":"Red","Dimensions":2}mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$
```

After repeating the same processes as the previous task, we defined the decorator with the file provided, making minor alterations with the IP and the file itself to make it work and give the desired output shown below.

```
mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ cd ~/OpenFaaS
mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ faas-cli new --lang node12 --prefix us.gcr.io/vast-alcove-367116 decorator
Folder: decorator already exists
mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ faas-cli deploy -f decorator.yml
Deploying: decorator.
WARNING! You are not using an encrypted connection to the gateway, consider using HTTPS.

Deployed. 202 Accepted.
URL: http://34.95.52.188:8080/function/decorator

mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ docker push us.gcr.io/vast-alcove-367116/decorator
Using default tag: latest
The push refers to repository [us.gcr.io/vast-alcove-367116/decorator]
806f0e209477: Layer already exists
384b0350834f: Layer already exists
b4020ba7625e: Layer already exists
b1e14583d7e8: Layer already exists
3607cda1296a: Layer already exists
6eb1fbae3ab2: Layer already exists
aff2fd231f3f: Layer already exists
95f3c2add12d: Layer already exists
d86bda25f4c1: Layer already exists
270a2a5c9a25: Layer already exists
bbce70513b0a: Layer already exists
7f30cde3f699: Layer already exists
fe810f5902cc: Layer already exists
dfd8c046c602: Layer already exists
4fc242d58285: Layer already exists
latest: digest: sha256:c627286de2e12d7d409ce23bfb6cc48f80148f758620b2cd94ac15a7e2cd85c3 size: 3663
mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ faas-cli deploy -f decorator.yml
Deploying: decorator.
WARNING! You are not using an encrypted connection to the gateway, consider using HTTPS.

Deployed. 202 Accepted.
URL: http://34.95.52.188:8080/function/decorator

mamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ curl http://34.95.52.188:8080/function/decorator -H 'Content-Type: application/json' -d '{ "Name": "Square", "Dimensions": 2 }'
04 page not foundmamunotuclass@cloudshell:~/OpenFaaS (vast-alcove-367116)$ cd~
-bash: cd~: command not found
```

File alteration of decorator/handler.js.

OpenFaaS > decorator > handler.js > <unknown> > ...

```javascript
'use strict'
const request = require('sync-request');

module.exports = async (event, context) => {
    var obj = event.body;
    if (obj['Name'] === undefined) {
        obj['Name'] = 'Nameless';
    }
    if (obj['Color'] === undefined) {
        obj['Color'] = 'Transparent';
    }
    var res = request('POST', 'http://34.95.52.188:8080/function/decorator', {
        body: JSON.stringify(obj)
      });
      console.log(res["body"].toString())
    return context.status(200).succeed(res["body"].toString('utf8', 1, res["body"].length-
1).replace(/\\\"/g,'\"'));
}
```

Discussion:

In summary, the main problem at hand was to understand how latency works and how the applications need to have health monitoring checks to ensure that it is working to its highest functionality. We need to understand that we cannot always monitor the hardware in the system for every device, as that can end up being costly and wasting time in the long run. With the health monitoring system we had discussed in part 1, we can use it effectively to ensure that we can understand and monitor the network latency, performance and availability of the underlying compute and storage systems with the networks between it.

We will be gathering different values and monitor the health checks as we did in the lab. In summary for part 1, we were able to confirm the need for implementing health monitoring of clusters and the nodes to each endpoint of the application to ensure that all the checkpoints were in place to monitor the node health. The way we were able to accomplish this with part 2 and 3 was by implementing several services: Backup service, implemented service, and a circuit breaker. We had used the circuit breaker several times and then paired it with a dummy service to ensure that the circuit breaker was working as it was intended to do. The way this would work is that the circuit breaker would be tested to make sure that the endpoints have the same value so it can be deemed to be a healthy endpoint.

For part 3, we had used the decorator function and the main function to distinguish the difference between the two and their interactions with the application and to view the values that get sent to the applications via JSON files that get converted into strings. The decorator endpoint ensures the validity of the checkpoint for the health monitoring by responding to any errors in the system and fixing them accordingly.

Design:

As we know, Kubernetes provides persistent volumes and it is necessary to have these volumes in a distributed system because if these volumes were not present, any data that has been previously created or altered will become null and void when the container restarts its processes. An example of when persistent volumes are needed is when let's say for example you want to be a developer at an enterprise, you want to make sure that your project is scalable. This way, if the consumer requests for something that can have several new environments to ensure that the product they want can run in different environments to their hearts desire. As we can see below, we will show how to configure a YAML file to implement the example and then running it and test it with persistent volume. We will be using “minikube” based off a guide we had found on the Internet doing our own research.

We install minikube using the commands and then check for all the pods to be working to make sure that the application will be working.

```
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ minikube start
* minikube v1.27.0 on Debian 11.5 (amd64)
  - MINIKUBE_FORCE_SYSTEMD=true
  - MINIKUBE_HOME=/google/minikube
  - MINIKUBE_WANTUPDATENOTIFICATION=false
! Kubernetes 1.25.0 has a known issue with resolv.conf. minikube is using a workaround that should work for most use cases.
! For more information, see: https://github.com/kubernetes/kubernetes/issues/112135
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Updating the running docker "minikube" container ...
* Preparing Kubernetes v1.25.0 on Docker 20.10.17 ...
  - kubelet.cgroups-per-qos=false
  - kubelet.enforce-node-allocatable=""
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl get pods
No resources found in default namespace.
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl get pods -A
NAMESPACE     NAME                               READY   STATUS    RESTARTS      AGE
kube-system   coredns-565d847f94-n6fwd           1/1     Running   1 (57s ago)   3m36s
kube-system   etcd-minikube                      1/1     Running   2 (57s ago)   3m48s
kube-system   kube-apiserver-minikube            1/1     Running   2 (47s ago)   3m48s
kube-system   kube-controller-manager-minikube   1/1     Running   2 (57s ago)   3m48s
kube-system   kube-proxy-v9wj7                   1/1     Running   2 (57s ago)   3m37s
kube-system   kube-scheduler-minikube            1/1     Running   1 (62s ago)   3m48s
kube-system   storage-provisioner                1/1     Running   3 (43s ago)   3m46s
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ minikube dashboard
* Enabling dashboard ...
  - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
  - Using image docker.io/kubernetesui/dashboard:v2.6.0
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:41731/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
  http://127.0.0.1:41731/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/
```

We can now enter the node and create persistent volumes as seen below that the pods can utilize as persistent volumes.

```
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl apply -f https://k8s.io/examples/pods/storage/pv-volume.yaml
persistentvolume/task-pv-volume created
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl apply -f https://k8s.io/examples/pods/storage/pv-claim.yaml
persistentvolumeclaim/task-pv-claim created
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl apply -f https://k8s.io/examples/pods/storage/pv-pv-pod
error: unable to read URL "https://k8s.io/examples/pods/storage/pv-pv-pod", server reported 404 Not Found, status code=404
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl apply -f https://k8s.io/examples/pods/storage/pv-pod.yaml
pod/task-pv-pod created
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl get pvc task-pv-claim
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         45s
mamunotuclass@cloudshell:~ (vast-alcove-367116)$
```

From here, we can see the node being re entered by the command as seen below to ensure the persistent volume is being used as needed as shown created earlier into the node.

```
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   10Gi       RWO            manual         45s
mamunotuclass@cloudshell:~ (vast-alcove-367116)$ kubectl exec -it task-pv-pod -- /bin/bash
root@task-pv-pod:/# apt update
Get:1 http://deb.debian.org/debian bullseye InRelease [116 kB]
Get:2 http://deb.debian.org/debian-security bullseye-security InRelease [48.4 kB]
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:4 http://deb.debian.org/debian bullseye/main amd64 Packages [8184 kB]
Get:5 http://deb.debian.org/debian-security bullseye-security/main amd64 Packages [193 kB]
Get:6 http://deb.debian.org/debian bullseye-updates/main amd64 Packages [14.6 kB]
Fetched 8600 kB in 2s (4735 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
root@task-pv-pod:/#
```

Below are the yaml files in the editor

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: task-pv-pod
spec:
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: task-pv-claim
  containers:
    - name: task-pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
```