



Faculty of Engineering & Applied Science

SOFE4790U – Distributed Systems

Lab 3 – CRN 44425

Due Date: 10/30/2022

Part 2 and 3 Final Results: https://drive.google.com/file/d/1N3OrsNYgn7ZJxBsOvo5_2-XzBiq8vung/view?usp=share_link

Persistent Volume Design - https://drive.google.com/file/d/1CBY0mUD-aRuRRubDbs5IoVmWZn2eyZkN/view?usp=share_link

First Name	Last Name	Student ID
Abdul	Bhutta	100785884

Part 1 - Health Endpoint Monitoring Pattern

Problem

One of the business requirements is to periodically check for status or monitor the applications and backend services to ensure they are functioning without any issues. It is much easier to monitor with a centralized physical system than a cloud system because of having complete control of the physical system.

Solution

The problem can be solved through a health monitoring system which checks the status of the components of the system, which can be implemented at the endpoint of the application and returns the status through a response code. Below is a list of requirements to implement the pattern.

Requirements	Description
Agent	Analyzes the outcome from the application which performed the health check
Application	Check the health status of the request at the endpoint

Discussion

Summarize the problem, the solution, and the requirements for the pattern given in part 1. Which of these requirements can be achieved by the procedures shown in parts 2 and 3?

A Health Endpoint monitoring pattern is used for monitoring through periodic intervals and by sending requests to an endpoint of the cloud applications and services. The monitoring pattern will send back the request code of each service and application to determine the availability and response time (latency). To implement this pattern, you will require two key components an agent on the cloud service which will analyze and perform the health checks. The other component will be an application to check the health status of the request at the endpoint.

The requirements are achieved in part 2 since it behaves as a health endpoint monitor. It checks for errors at the endpoint while integrating the circuit breaker pattern, which is used as a contingency as failures are bound to happen. The circuit breakers allow the services not to be overloaded with too many requests and redirect the traffic to allow the service to come back online. The circuit breaker is configured to check every 3 seconds at the main endpoint which acts as a health checkpoint. Part 3 also follows the requirements as well since the decorator pattern was applied to a function which transforms the input value sent by the user and checks whether any of the fields are missing. It then changes those values to a predefined default value.

Part 2

Build Docker Image

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$ docker build . -t us.gcr.io/sofe4790u-lab3/dummyservice
Sending build context to Docker daemon  6.656kB
Step 1/7 : FROM node:carbon
--> 8eeadf3757f4
Step 2/7 : WORKDIR /usr/src/app
--> Using cache
--> ca2eab3d6b36
Step 3/7 : COPY package*.json ./
--> Using cache
--> 244a016b5a88
Step 4/7 : RUN npm install
--> Using cache
--> 383cce04775a
Step 5/7 : COPY . .
--> Using cache
--> e097f6d23148
Step 6/7 : EXPOSE 80
--> Using cache
--> 4c35150b8d2c
Step 7/7 : CMD [ "npm", "start" ]
--> Using cache
--> 1579b6abb06d
Successfully built 1579b6abb06d
Successfully tagged us.gcr.io/sofe4790u-lab3/dummyservice:latest
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$
```

Push the docker image to the container

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$ docker push us.gcr.io/sofe4790u-lab3/dummyservice
Using default tag: latest
The push refers to repository [us.gcr.io/sofe4790u-lab3/dummyservice]
994112752128: Pushed
1c6407d87b9b: Pushed
ac4e20b24eca: Pushed
d9143b82daf0: Pushed
423451ed44f2: Layer already exists
b2aaf85d6633: Layer already exists
88601a85ce11: Layer already exists
42f92c2f9c08e: 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: sha256:58f87b8f57c6112c38e3db063910cfb37b2f70240dd7f92f0fb9017645ce65b2 size: 3048
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$
```

Create Image for Dummy Service

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$ docker build . -t us.gcr.io/sofe4790u-lab3/dummyservice
Sending build context to Docker daemon  6.656kB
Step 1/7 : FROM node:carbon
carbon: Pulling from library/node
146bd6a88618: Pull complete
9935d0c62ace: Pull complete
db0efb86e806: Pull complete
e705a4c4fd31: Pull complete
c877b722db6f: Pull complete
645c20ec8214: Pull complete
db8fb9ab2afe: Pull complete
1c15fcd1b3ea: Pull complete
fbd993995f40: Pull complete
Digest: sha256:a681bf74805b80d03eb21a6c0ef168a976108a287a74167ab593fc953aac34df
Status: Downloaded newer image for node:carbon
--> 8eeadf3757f4
Step 2/7 : WORKDIR /usr/src/app
--> Running in a0338cff83ed
Removing intermediate container a0338cff83ed
--> ca2eab3d6b36
Step 3/7 : COPY package*.json ./
--> 244a016b5a88
Step 4/7 : RUN npm install
--> Running in ae670c7bca78
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN dummyservice@1.0.1 No repository field.
npm WARN dummyservice@1.0.1 No license field.

added 57 packages from 42 contributors and audited 57 packages in 2.161s

6 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

Removing intermediate container ae670c7bca78
--> 383cce04775a
Step 5/7 : COPY . .
--> e097f6d23148
Step 6/7 : EXPOSE 80
--> Running in 0972982a8aca
Removing intermediate container 0972982a8aca
--> 4c35150b8d2c
Step 7/7 : CMD [ "npm", "start" ]
--> Running in f4b842fb0d2a
Removing intermediate container f4b842fb0d2a
--> 1579b6abb06d
Successfully built 1579b6abb06d
Successfully tagged us.gcr.io/sofe4790u-lab3/dummyservice:latest
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2/DummyServiceContainer (sofe4790u-lab3)$
```

Deploy the service and expose it through a load balancer

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl delete services dummy-deployment
service "dummy-deployment" deleted
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl create -f dummy-deployment.yaml
deployment.apps/dummy-deployment created
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl expose deployment dummy-deployment --port=80 --type=LoadBalancer --name
dummy-deployment
service/dummy-deployment exposed
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl create -f backup-deployment.yaml
deployment.apps/backup-deployment created
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl expose deployment backup-deployment --port=80 --type=LoadBalancer --name
backup-deployment
service/backup-deployment exposed
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$
```

Check deployment and services status

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl get pods
NAME                                READY    STATUS    RESTARTS   AGE
backup-deployment-79cf469564-xv1x6  1/1      Running   0           53s
dummy-deployment-7bd5bc5dd-tqzrf    1/1      Running   0          109s
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl get services
^[[NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
backup-deployment                    LoadBalancer        10.80.14.197     34.95.18.116     80:31897/TCP     49s
dummy-deployment                    LoadBalancer        10.80.10.189     34.95.52.218     80:32651/TCP     106s
kubernetes                           ClusterIP            10.80.0.1        <none>           443/TCP          97m
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl get deployments
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
backup-deployment  1/1      1              1            61s
dummy-deployment   1/1      1              1           117s
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$
```

Create Configmap

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl create -f nginx-configmap.yaml
configmap/nginx-configuration created
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl create -f circuitbreaker.yaml
deployment.apps/circuitbreaker created
service/circuitbreaker created
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$
```

Get external IP Address


```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ kubectl get services
NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
backup-deployment                    LoadBalancer        10.80.14.197     34.95.18.116     80:31897/TCP     3m6s
circuitbreaker                      LoadBalancer        10.80.7.220      35.203.92.185    80:32080/TCP     93s
dummy-deployment                    LoadBalancer        10.80.10.189     34.95.52.218     80:32651/TCP     4m3s
kubernetes                           ClusterIP            10.80.0.1        <none>           443/TCP          100m
```

Testing the Circuit Breaker, a) Run the command three times

```
bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ curl -v http://35.203.92.185
* Trying 35.203.92.185:80...
* Connected to 35.203.92.185 (35.203.92.185) port 80 (#0)
> GET / HTTP/1.1
> Host: 35.203.92.185
> 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 20:37:59 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-j4qqy4RWDPmP1lDpTldstkgMcg"
<
* Connection #0 to host 35.203.92.185 left intact
SOMERESPONSE FROM 10.76.0.12bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ curl -v http://35.203.92.185
* Trying 35.203.92.185:80...
* Connected to 35.203.92.185 (35.203.92.185) port 80 (#0)
> GET / HTTP/1.1
> Host: 35.203.92.185
> 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 20:38:40 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-j4qqy4RWDPmP1lDpTldstkgMcg"
<
* Connection #0 to host 35.203.92.185 left intact
SOMERESPONSE FROM 10.76.0.12bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)$ curl -v http://35.203.92.185
* Trying 35.203.92.185:80...
* Connected to 35.203.92.185 (35.203.92.185) port 80 (#0)
> GET / HTTP/1.1
> Host: 35.203.92.185
> 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 20:38:41 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 28
< Connection: keep-alive
< X-Powered-By: Express
< ETag: W/"1c-j4qqy4RWDPmP1lDpTldstkgMcg"
<
* Connection #0 to host 35.203.92.185 left intact
```

<i>b) Mimic an Error</i>	
<pre> bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)\$ curl -d "" -s -D - http://34.95.52.218/fakeerrormodeon HTTP/1.1 200 OK X-Powered-By: Express Content-Type: text/html; charset=utf-8 Content-Length: 18 ETag: W/"12-N3xK5AJr1Lw2pCRqWRZhBbPf84g" Date: Sat, 29 Oct 2022 20:39:14 GMT Connection: keep-alive OK FROM 10.76.0.12bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)\$ </pre>	
<i>c) Reset the error</i>	
<pre> bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)\$ curl -d "" -s -D - http://34.95.52.218/fakeerrormodeoff HTTP/1.1 200 OK X-Powered-By: Express Content-Type: text/html; charset=utf-8 Content-Length: 18 ETag: W/"12-N3xK5AJr1Lw2pCRqWRZhBbPf84g" Date: Sat, 29 Oct 2022 20:40:04 GMT Connection: keep-alive </pre>	
<i>d) Run command and check IP</i>	
<pre> bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part2 (sofe4790u-lab3)\$ curl -v http://35.203.92.185 * Trying 35.203.92.185:80... * Connected to 35.203.92.185 (35.203.92.185) port 80 (#0) > GET / HTTP/1.1 > Host: 35.203.92.185 > 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 20:40:38 GMT < Content-Type: text/html; charset=utf-8 < Content-Length: 28 < Connection: keep-alive < X-Powered-By: Express < ETag: W/"1c-j4qqy4RWDpImp1lDpT1dstkgMcg" < * Connection #0 to host 35.203.92.185 left intact </pre>	

Part 3

<i>Create a cluster admin role binding</i>	
<pre> bhutta_abdul@cloudshell:~/SOFE4790U-lab3/part3 (sofe4790u-lab3)\$ kubectl create clusterrolebinding "cluster-admin-(whoami)" --clusterrole=cluster-admin --user="\$(gcloud config get-value core/account)" Your active configuration is: [cloudshell-25206] clusterrolebinding.rbac.authorization.k8s.io/cluster-admin-bhutta_abdul created </pre>	
<i>Deploy OpenFaas to GKE</i>	
<pre> bhutta_abdul@cloudshell:~ (sofe4790u-lab3)\$ curl -SLsf https://dl.get-arkade.dev/ sudo sh arkade install openfaas --load-balancer 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 </pre>	
<i>Install Client</i>	

```

bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ 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'.

OpenFaas

CLI:
commit: 8820d8e4a15dab900d8a7e8fc271851ccb94012e
version: 0.14.11

```

Verify OpenFaas is working

```

bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ kubectl -n openfaas get deployments -l "release=openfaas, app=openfaas"
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
alertmanager        1/1     1             1           6m45s
basic-auth-plugin    1/1     1             1           6m45s
gateway              1/1     1             1           6m45s
nats                 1/1     1             1           6m45s
prometheus           1/1     1             1           6m45s
queue-worker         1/1     1             1           6m45s
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$

```

Check OpenFaas is ready and get IP

```

bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ kubectl rollout status -n openfaas deploy/gateway
deployment "gateway" successfully rolled out
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ kubectl get svc -o wide gateway-external -n openfaas
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE      SELECTOR
gateway-external    LoadBalancer      10.80.5.210     34.95.26.217    8080:32273/TCP   7m33s    app=gateway
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$

```

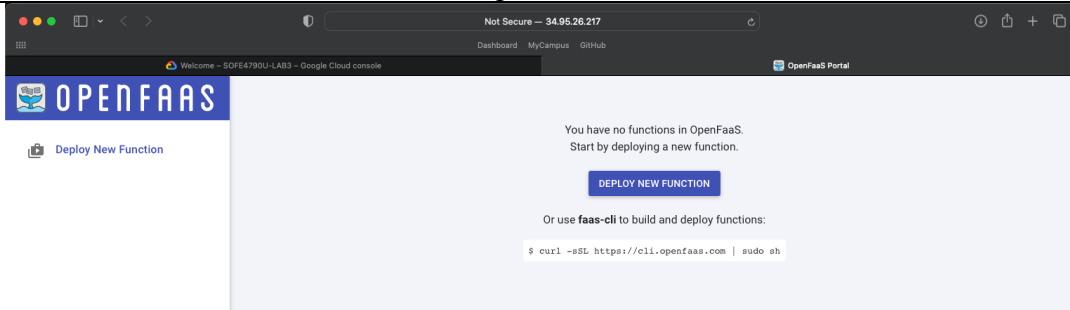
Save Login and password

```

bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ export OPENFAAS_URL="34.95.26.217:8080"
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ echo $PASSWORD
9u248zD4208eW7dnHakHdKof5
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ echo -n $PASSWORD | faas-cli login --username admin --password-stdin
Calling the OpenFaaS server to validate the credentials...
WARNING! You are not using an encrypted connection to the gateway, consider using HTTPS.
credentials saved for admin http://34.95.26.217:8080
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$

```

OpenFaas



Create new directory and pull all available templates

```

bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ mkdir ~/OpenFaaS
bhutta_abdul@cloudshell:~ (sofe4790u-lab3)$ cd ~/OpenFaaS
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli template pull
Fetch templates from repository: https://github.com/openfaas/templates.git at
2022/10/30 00:40:07 Attempting to expand templates from https://github.com/openfaas/templates.git
2022/10/30 00:40:08 Fetched 17 template(s) > (csharp dockerfile go javall javall-vert-x node node12 node12-debian node14 node16 node17 php7 php8 python pyth
on3 python3-debian ruby) from https://github.com/openfaas/templates.git
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli new --list
Languages available as templates:
- csharp
- dockerfile
- go
- javall
- javall-vert-x
- node
- node12
- node12-debian
- node14
- node16
- node17
- php7
- php8
- python
- python3
- python3-debian
- ruby
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$

```

Empty NodeJS function

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli new --lang node12 --prefix us.gcr.io/sofe4790u-lab3 main
Folder: main created.
```



```
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.
```

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$
```

Update handler.js file

```
OpenFaaS > main > handler.js > ...
1  'use strict'
2
3  module.exports = async (event, context) => {
4      var parameters=JSON.stringify(event.body)
5      return context
6      .status(200)
7      .succeed(parameters)
8  }
```

Build Docker image

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ cd ~/OpenFaaS
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli build -f main.yml
[0] > Building main.
Clearing temporary build folder: ./build/main/
Preparing: ./main/ build/main/function
Building: us.gcr.io/sofe4790u-lab3/main:latest with node12 template. Please wait..
Sending build context to Docker daemon 12.8kB
Step 1/31 : FROM --platform=$(TARGETPLATFORM:-linux/amd64) ghcr.io/openfaas/of-watchdog:0.9.10 as watchdog
0.9.10: Pulling from openfaas/of-watchdog
c4fc21d17d12: Pulling fs layer
c4fc21d17d12: Download complete
c4fc21d17d12: Pull complete
Digest: sha256:5d1ed766546ff5510614c695d48e3e0bafefee01ec8c04dc3a51297cb75bb57a
Status: Downloaded newer image for ghcr.io/openfaas/of-watchdog:0.9.10
--> 9f97468a4531
Step 2/31 : FROM --platform=$(TARGETPLATFORM:-linux/amd64) node:12-alpine as ship
12-alpine: Pulling from library/node
```

Push to container

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ docker push us.gcr.io/sofe4790u-lab3/main
Using default tag: latest
The push refers to repository [us.gcr.io/sofe4790u-lab3/main]
c4cc512dfbdb: Pushed
f3e52fa4f0ee: Pushed
c17d15683f0c: Pushed
4ce34a584422: Pushed
9a941d3d77c7: Pushed
6229463858ce: Pushed
57021fbca4bc: Pushed
2900e05f5c62: Pushed
d86bda25f4c1: Pushed
5ee04ae9a725: Pushed
4c5614929cb1: Pushed
7f30cde3f699: Layer already exists
fe810f5902cc: Layer already exists
dfd8c046c602: Layer already exists
4fc242d58285: Layer already exists
latest: digest: sha256:28f6b6df3fcb179a238b0be33a1b5c8479b09efbcd5a1ed8a136e106d3ac253 size: 3659
```


Deploy to OpenFaaS

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli deploy -f main.yml
Deploying: main.
WARNING! You are not using an encrypted connection to the gateway, consider using HTTPS.

Deployed. 202 Accepted.
URL: http://34.95.26.217:8080/function/main
```

Send JSON object

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ curl http://34.95.26.217:8080/function/main -H 'Content-Type: application/json' -d '{"Name": "Square", "Color": "Red", "Dimensions": 2}'
{"Name": "Square", "Color": "Red", "Dimensions": 2}bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$
```

Create new empty NodeJS function

```
bhutta_abdul@cloudshell:~/ (sofe4790u-lab3)$ cd ~/OpenFaaS
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli new --lang node12 --prefix us.gcr.io/sofe4790u-lab3 decorator
Folder: decorator created.
```



```
Function created in folder: decorator
Stack file written: decorator.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.

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ cd ~/OpenFaaS
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli build -f decorator.yml
[0] > Building decorator.
```

Clearing temporary build folder: ./build/decorator/

Preparing: ./decorator/ build/decorator/function

Building: us.gcr.io/sofe4790u-lab3/decorator: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: sha256:5dled766546ff5510614c695d48e3e0bafefee01ec8c04dc3a51297cb75bb57a

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

Update handler.js

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

```
1 'use strict'
2 const request = require('sync-request');
3
4 module.exports = async (event, context) => {
5   var obj = event.body;
6   if (obj['Name'] === undefined) {
7     obj['Name'] = 'Nameless';
8   }
9   if (obj['Color'] === undefined) {
10    obj['Color'] = 'Transparent';
11  }
12  var res = request('POST', 'http://34.95.26.217:8080/function/main', {
13    body: JSON.stringify(obj)
14  });
15  console.log(res["body"].toString())
16  return context.status(200).succeed(res["body"].toString('utf8', 1, res["body"].length-1).replace(/\\/"/g, '\\\\'))
17 }
```


Update package.json

```
OpenFaaS > decorator > package.json > ...
```

```
1 {
2   "name": "openfaas-function", "version": "1.0.0",
3   "description": "OpenFaaS Function", "main": "handler.js",
4   "scripts": {
5     "test": "echo \"Error: no test specified\" && exit 0",
6     "keywords": [],
7     "author": "OpenFaaS Ltd", "license": "MIT", "dependencies": {
8     "sync-request": "^6.1.0"
9   }
}
```

Build docker image

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ cd ~/OpenFaaS
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ faas-cli build -f decorator.yml
[0] > Building decorator.
Clearing temporary build folder: ./build/decorator/
Preparing: ./decorator/ build/decorator/function
Building: us.gcr.io/sofe4790u-lab3/decorator:latest with node12 template. Please wait..
Sending build context to Docker daemon 12.8kB
Step 1/31 : FROM --platform=${TARGETPLATFORM:-linux/amd64} ghcr.io/openfaas/of-watchdog:0.9.10 as watchdog
0.9.10: Pulling from openfaas/of-watchdog
c4fc21d17d12: Pulling fs layer
c4fc21d17d12: Verifying Checksum
c4fc21d17d12: Download complete
c4fc21d17d12: Pull complete
Digest: sha256:5dled766546ff5510614c695d48e3e0bafefee01ec8c04dc3a51297cb75bb57a
Status: Downloaded newer image for ghcr.io/openfaas/of-watchdog:0.9.10
--> 9f97468a4531
Step 2/31 : FROM --platform=${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
```

Push to container

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ docker push us.gcr.io/sofe4790u-lab3/decorator
Using default tag: latest
The push refers to repository [us.gcr.io/sofe4790u-lab3/decorator]
de2351a82d23: Pushed
8b1412e6966b: Pushed
b6d4c4d8c4a1: Pushed
a63e9eed078c: Pushed
7c23f000b1cc: Pushed
293dda09737: Pushed
583972de9644: Pushed
b6e9fd53cb5f: Pushed
d86bda25f4c1: Pushed
4d5ce352abac: Pushed
aded21115157: Pushed
7f30cde3f699: Layer already exists
fe810f5902cc: Layer already exists
dfd8c046c602: Layer already exists
4fc242d58285: Layer already exists
latest: digest: sha256:ac1a5c5d7cc28a30b27002e4781012c39647ace7f7db4d4bad0d3df259035396 size: 3663
```

Deploy to OpenFaaS

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ 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://34.95.26.217:8080/function/decorator
```

Send a JSON object and will return transparent

```
bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$ curl http://34.95.26.217:8080/function/decorator -H 'Content-Type: application/json' -d '{ "Name": "Square", "Dimensions": 2 }'
{"Name": "Square", "Dimensions": 2, "Color": "Transparent"}bhutta_abdul@cloudshell:~/OpenFaaS (sofe4790u-lab3)$
```

Design

Kubernetes provides persistent volumes. Why such a feature can be important? How to implement it? Provide an example in which persistent volumes are needed. Configure a YAML file to implement the example. Run it and test the creation of persistent volume and its ability to provide the required functionality within the example.

Persistent volumes allow the administrator to save data and use it as a plugin which is independent from the container and has its own lifecycle. The data can be shared between pods even after the pod has been restarted while keeping the data from the previous session can still be accessed. An example of persistent volume is used with application that require a database and if the application is closed, we still require the data to be stored. An example of persistent volume implementation is provided by Kubernetes official website (<https://kubernetes.io/docs/tasks/configure-pod-container/configure-persistent-volume-storage/>).

Using Minikube to create a single node cluster

```
bhutta.abdul@cloudshell:~ (sofe4790u-lab3)$ 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
* Automatically selected the docker driver. Other choices: none, ssh
* Using Docker driver with root privileges
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Downloading Kubernetes v1.25.0 preload ...
  > preloaded-images-k8s-v18-v1...: 385.37 MiB / 385.37 MiB 100.00% 204.83
  > gcr.io/k8s-minikube/kicbase: 0 B [ ] ?% ? p/s 8.9s
* Creating docker container (CPUs=2, Memory=4000MB) ...
* Preparing Kubernetes v1.25.0 on Docker 20.10.17 ...
- kubelet.cgroups-per-qos=false
- kubelet.enforce-node-allocatable=""
- Generating certificates and keys ...
- Booting up control plane ...
- Configuring RBAC rules ...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: default-storageclass, storage-provisioner
* Done! kubect1 is now configured to use "minikube" cluster and "default" namespace by default
bhutta.abdul@cloudshell:~ (sofe4790u-lab3)$ minikube ssh
docker@minikube:~$
```

Create index file and test output

```
docker@minikube:~$ sudo mkdir /mnt/data
docker@minikube:~$ sudo sh -c "echo 'Hello from Kubernetes storage' > /mnt/data/index.html"
docker@minikube:~$ cat /mnt/data/index.html
Hello from Kubernetes storage
docker@minikube:~$
```

Persistent Volume Yaml file

```
pv-volume.yaml > ...
1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    name: task-pv-volume
5  labels:
6    type: local
7  spec:
8    storageClassName: manual
9  capacity:
10   storage: 10Gi
11  accessModes:
12   - ReadWriteOnce
13  hostPath:
14   path: "/mnt/data"
15
```

Create and get the information for the persistent volume

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl apply -f https://k8s.io/examples/pods/storage/pv-volume.yaml
persistentvolume/task-pv-volume created
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl get pv task-pv-volume
```

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM	STORAGECLASS	REASON	AGE
task-pv-volume	10Gi	RWO	Retain	Available		manual		7s

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $
```

Persistent Volume Claim Yaml file

```
pv-claim.yaml > ...
1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    name: task-pv-claim
5  spec:
6    storageClassName: manual
7    accessModes:
8      - ReadWriteOnce
9    resources:
10     requests:
11       storage: 3Gi
12
```

Create and bound the persistent volume claim to persistent volume

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl apply -f https://k8s.io/examples/pods/storage/pv-claim.yaml
persistentvolumeclaim/task-pv-claim created
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl get pv task-pv-volume
```

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM	STORAGECLASS	REASON	AGE
task-pv-volume	10Gi	RWO	Retain	Bound	default/task-pv-claim	manual		2m23s

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl get pvc task-pv-claim
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
task-pv-claim	Bound	task-pv-volume	10Gi	RWO	manual	19s

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $
```

Creating a POD Yaml file

```
pv-pod.yaml > {} spec > containers > ...
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: task-pv-pod
5  spec:
6    volumes:
7      - name: task-pv-storage
8        persistentVolumeClaim:
9          Created a minute ago
10         claimName: task-pv-claim
11    containers:
12      - name: task-pv-container
13        image: nginx
14        ports:
15          - containerPort: 80
16            name: "http-server"
17        volumeMounts:
18          - mountPath: "/usr/share/nginx/html"
19            name: task-pv-storage
```

Create Pod and verify its running

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $ kubectl get pod task-pv-pod
```

NAME	READY	STATUS	RESTARTS	AGE
task-pv-pod	0/1	ContainerCreating	0	3s

```
bhutta_abdul@cloudshell:~ (sofe4790u-lab3) $
```

Create the pod and verify its running within the container

```
bhutta_abdul@cloudshell:~ (sofe4790u-1ab3)$ kubectl apply -f https://k8s.io/examples/pods/storage/pv-pod.yaml
pod/task-pv-pod created
bhutta_abdul@cloudshell:~ (sofe4790u-1ab3)$ kubectl get pod task-pv-pod
NAME          READY   STATUS             RESTARTS   AGE
task-pv-pod   0/1     ContainerCreating   0           3s
bhutta_abdul@cloudshell:~ (sofe4790u-1ab3)$ 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 (4874 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:/# apt install curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (7.74.0-1.3+deb11u3).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
```

Verify output

```
root@task-pv-pod:/# curl http://localhost/
Hello from Kubernetes storage
root@task-pv-pod:/# █
```

Videos links

Part 2 and 3 Final Results: https://drive.google.com/file/d/1N3OrsNYgn7ZJxBsOvo5_2-XzBiq8vunq/view?usp=share_link
Persistent Volume Design - https://drive.google.com/file/d/1CBY0mUD-aRuRRubDbs5IoVmWZn2eyZkN/view?usp=share_link