SOFE 4790U: Distributed Systems

Date: September 18, 2022

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

Philip Jasionowski 100751888

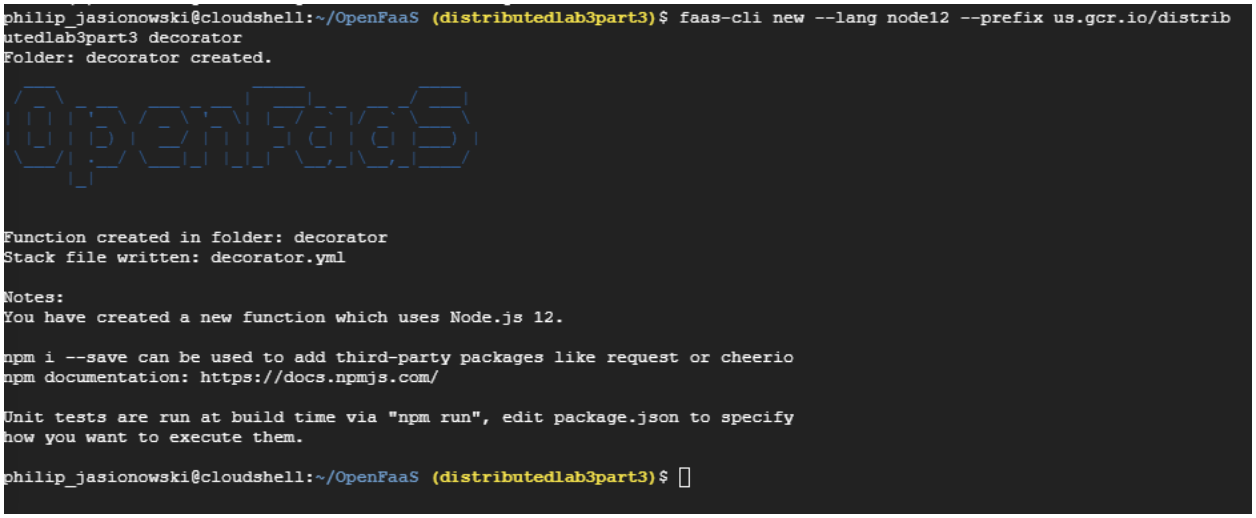
Learning Objectives:

In this lab we learned to create a service using NodeJS and how to create a docker image. We also learned how to configure a circuit breaker using Nginx. We also learned how to configure and use OpenFaas on GCP.

Links:

Video 1 (part 1): https://youtu.be/jmbVWGxWXC8
Video 1 (part 2): https://youtu.be/jmbVWGxWXC8
Video 2:

```
CLI:
 commit:  8820d8e4a15dab900d8a7e8fc271851ccb94012e
 version: 0.14.11
philip_jasionowski@cloudshell:~/OpenFaaS (distributedlab3part3)$ faas template pull
Fetch templates from repository: https://github.com/openfaas/templates.git at
2022/10/29 00:26:22 Attempting to expand templates from https://github.com/openfaas/templates.git
2022/10/29 00:26:22 Fetched 17 template(s) : [csharp dockerfile go java11 java11-vert-x node node
14 node16 node17 php7 php8 python python3 python3-debian ruby] from https://github.com/openfaas/t
philip_jasionowski@cloudshell:~/OpenFaaS (distributedlab3part3)$ fass new --list
-bash: fass: command not found
philip_jasionowski@cloudshell:~/OpenFaaS (distributedlab3part3)$ faas new --list
Languages available as templates:
- csharp
- dockerfile
- go
- java11
- java11-vert-x
- node
- node12
- node12-debian
- node14
- node16
- node17
- php7
- php8
- python
- python3
- python3-debian
- ruby

philip_jasionowski@cloudshell:~/OpenFaaS (distributedlab3part3)$ faas-cli new --lang node12 --pre
utedlab3part3 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.org/

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

philip_jasionowski@cloudshell:~/OpenFaaS (distributedlab3part3)$
```



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?

Monitoring the health of applications is important. The problem is that it's difficult to monitor cloud applications as you don't have direct control over the environment they are hosted in. This makes it hard to diagnose and check for problems or failures.

To solve this problem it is important to implement a health monitoring application which checks the health of the app. It checks by sending a request and monitoring the responses and analyzing them.

In part 2 of the lab we deployed a circuit breaker that prevents and manages failures. Failing services can recover using the circuit breaker and it can also reroute the traffic to different services. It activates if the failure amount gets over a certain set threshold

Design:

Kubernetes provides persistent volumes. Why can such a feature 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 simply hold data even after a pod has been destroyed or if it fails. GKE enables us to dynamically provision persistent volumes on demand. Persistent volumes store data at the cluster level so that they can be shared with multiple pods. This helps prevent loss of data if a pod is accidentally restarted or terminated.