SOFE 4790U: Distributed Computing

Date: October 30th, 2022

Lab 3: API Gateway & Failure Management

Zachary Carman

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?

The problem described in part one that as developers we need to monitor web applications and back-end services. It is good practice to montior the health of your application to make sure it was functioning correctly. This is difficult with services that are on the cloud due to a lack of control of the hosting environment and dependencies. The solution to this is to implement health monitoring through requests to endpoints of the application. This endpoint will return the application status as well as and components or services used. Other checks include checking cloud storage or database status and response time as well as checking other resources and services. Avalibility and response time can also be checked.

Requirements:

- They can connect with application endpoints and receive results.
- Be able to takin in not only status but response time as well.
- Be able to see resources and services located outside of the application
- Be able to check for the expiration of SSL certificates
- These endpoint checks need to be run for multiple different endpoints
- Checks need to be run quickly so that the system status does not change
- Validate response code and contents
- Validate URL returned from DNS lookup
- Measure response time

In part 2 we make it possible to check the service endpoint status in step 7 by setting up a circuit breaker to handle user requests in the event of an error. This is also the same in part 3 where we can test the endpoint health by creating sync-requests.

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 are used to provide data storage even when the container goes away. These live within a Kubernetes cluster and can remain after a pod to retain data. This can be done by provisioning persistent volme claim manifests and store data at the cluster level to be shrard by pods. This can be uses to amintain stateful applications.
To implement we need to create a yaml file, apply the file, and finally validate that it is available. We can add volume mounts and volumes to our deployments to save dat even after deployment deletion. I will create a wordpress applcation and a mysql application that will store the data even after the pod has been deleted thanks to persistent volumes.

Deliverables:

1. A report that includes the discussion and the design parts.
See Above.

2. An audible video of about 7 minutes maximum showing the final results of following the lab steps. It should include showing the deployments, services, and pods created in the lab with their test cases.
https://drive.google.com/file/d/1x324QMZkDvZ_cl_GQnQ7BphN9ZaZZk85/view?usp=sharing

3. Another audible video of 3 minutes maximum shows the configuration and implementation of persistent volumes using GKE.
https://drive.google.com/file/d/11vT3tEEOXpfzrBtHEM5JlH8wzN3xmMey/view?usp=sharing