Week 11

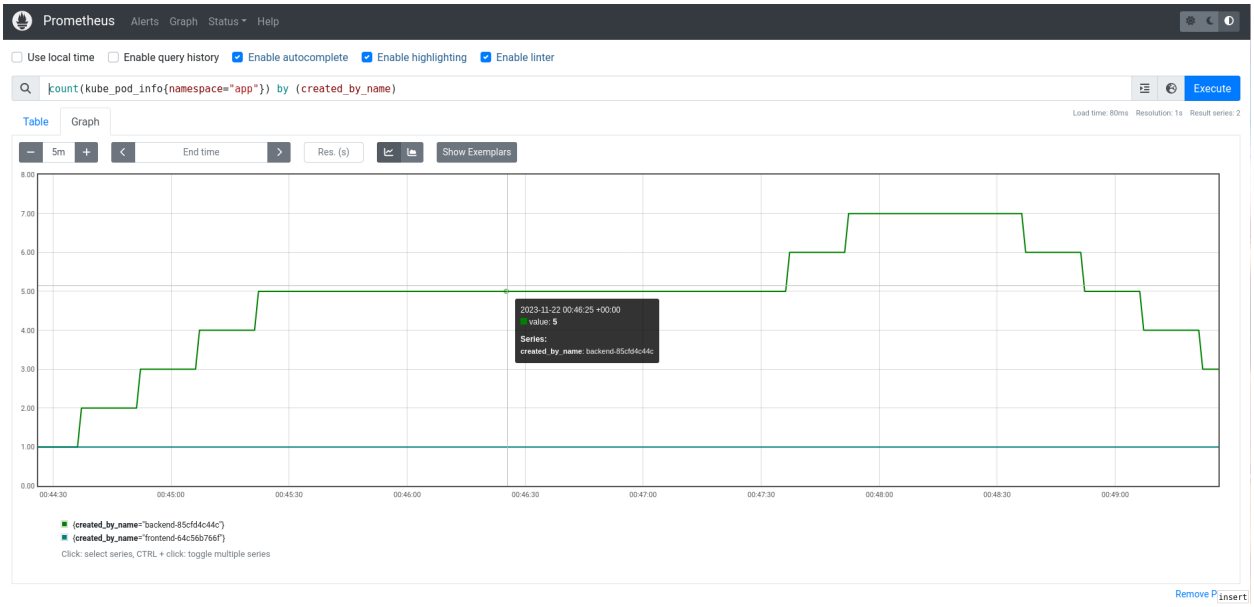
Learnings

Prometheus

I set up Prometheus using this tutorial: https://devopscube.com/setup-prometheus-monitoring-on-kubernetes/. Originally I was letting a package manager configure everything for prometheus for me, but when I ran into issues, it was hard to diagnose since I had no clue what I was touching.

After setting up Prometheus I need to set up kube-state-metrics so I can get information (like how many pods are running in a namespace). I folllowed this tutorial: https://devopscube.com/setup-kube-state-metrics/. This allowed me to view the amount of pods running per namespace using the following PromQL command in the Prometheus UI.

```
count(kube_pod_info{namespace="app"}) by (created_by_name)
```



The following command will show the CPU utilization for the backend pods in the app namespace

```
sum by (namespace, pod) (
  rate(container_cpu_usage_seconds_total{namespace="app",pod=~"backend-.*"}[5m])
)
```

Kubernetes

Namespaces

Provide a mechanism for isolating groups of resources within a single cluster. Kubernetes starts with four initial namespaces:

1. *default*: So you can start using the cluster without first creating a namespace.
2. *kube-node-lease*: Holds the “Lease” objects which allow the kubelet to send heartbeats so that the control plan can detect node failure.
3. *kube-public*: ?
4. *kube-system*: For objects created by the Kubernetes system. For example, this is where the metrics-server goes when you initialize it so that prometheus can use it.

To create a namespace, you can run the following command to define the kubernetes namespaces in the yaml files.

```
kubectl apply -f kubernetes/namespaces
```

Then, to use the namespace, first you must create a context. The following command creates a context called app and sets it to map to the “app” namespace from the folder `kubernetes/namespaces`.

```
kubectl config set-context app --namespace=app \
  --cluster=minikube \
  --user=minikube
```

Finally, to set the current context to the app name space (so you can deploy your pods from there), you can run the following command:

```
kubectl config use-context app
```

Now you can do all the normal things you would with a namespace like applying deployments, services and autoscalers.

Config Maps

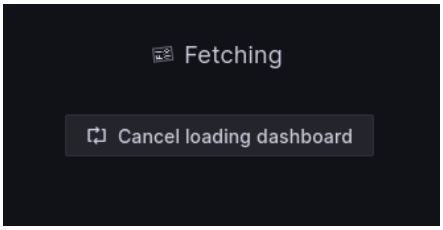
Kubernetes config maps allow you to externalize some configuration so that you don't need to redeploy each time you need to change some configuration.

Jobs

Creates one or more Pods and will continue to retry execution of the Pods until a specified number of them successfully terminate.

Problems

Grafana is taking an extremely long time to load.




```
E1121 06:32:18.150974   11667 portforward.go:370] error creating forwarding stream for port 3
000 -> 3000: Timeout occurred
E1121 06:32:24.247463   11667 portforward.go:347] error creating error stream for port 3000 -
> 3000: Timeout occurred
E1121 06:32:24.247550   11667 portforward.go:347] error creating error stream for port 3000 -
> 3000: Timeout occurred
Handling connection for 3000
Handling connection for 3000
E1121 06:32:24.259170   11667 portforward.go:347] error creating error stream for port 3000 -
> 3000: Timeout occurred
E1121 06:32:24.259197   11667 portforward.go:347] error creating error stream for port 3000 -
> 3000: Timeout occurred
Handling connection for 3000
E1121 06:32:24.259993   11667 portforward.go:347] error creating error stream for port 3000 -
> 3000: Timeout occurred
Handling connection for 3000
Handling connection for 3000
```

It looks like Prometheus is getting restarted constantly because it's being deemed unhealthy? This is not a problem I've had when starting minikube previously

●	prometheus-grafana-6fdbffb5c9-djvnr.17999fc9a1814b27	Unhealthy	Readiness probe failed: Get "http://10.244.0.135:3000/api/health": dial tcp 10.244.0.135:3000: connect: connection refused	kubelet minikube	spec.containers{grafana}	2	7 minutes ago	7 minutes ago

Solution:

I restarted the port-forward, waiting until the grafana pods were actually up.

Additional

Commands for memory

Restarting a deployment

```
kubectl rollout restart deployment <deployment_name>
```

Unapplying an applied yaml file

```
kubectl delete -f <file_location>
```