How to monitor your Golang App with Prometheus (and Grafana)?

Wojciech Barczyński - Spacelift.io

15 June 2023

About me

- Lead Engineering at Spacelift.io
- Background:
  Developer and System Engineer
- Before:
  Codility.com, Hypatos.ai, Lykehq (RocketInternet)

Point of view

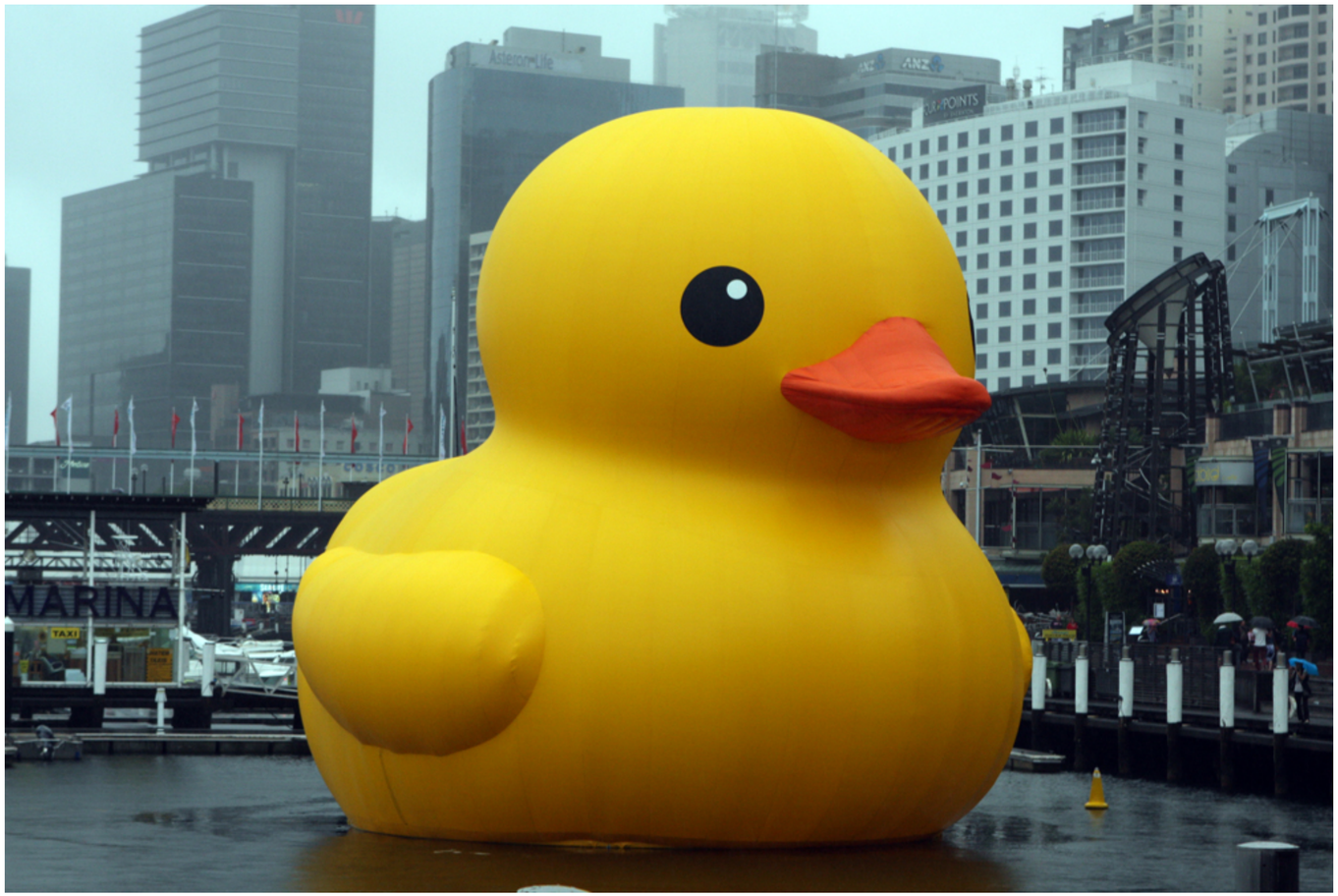
- Software Engineer,
- startups && fast-moving environment,
- Infra and platform should just work,
- Infra and platform ~ invisible.

Recommendation

- Start with monitoring;
- Metrics? Start with Up/Down, than RED and USE;
- Drop-in solution Prometheus;
- How to start? With this demo app.

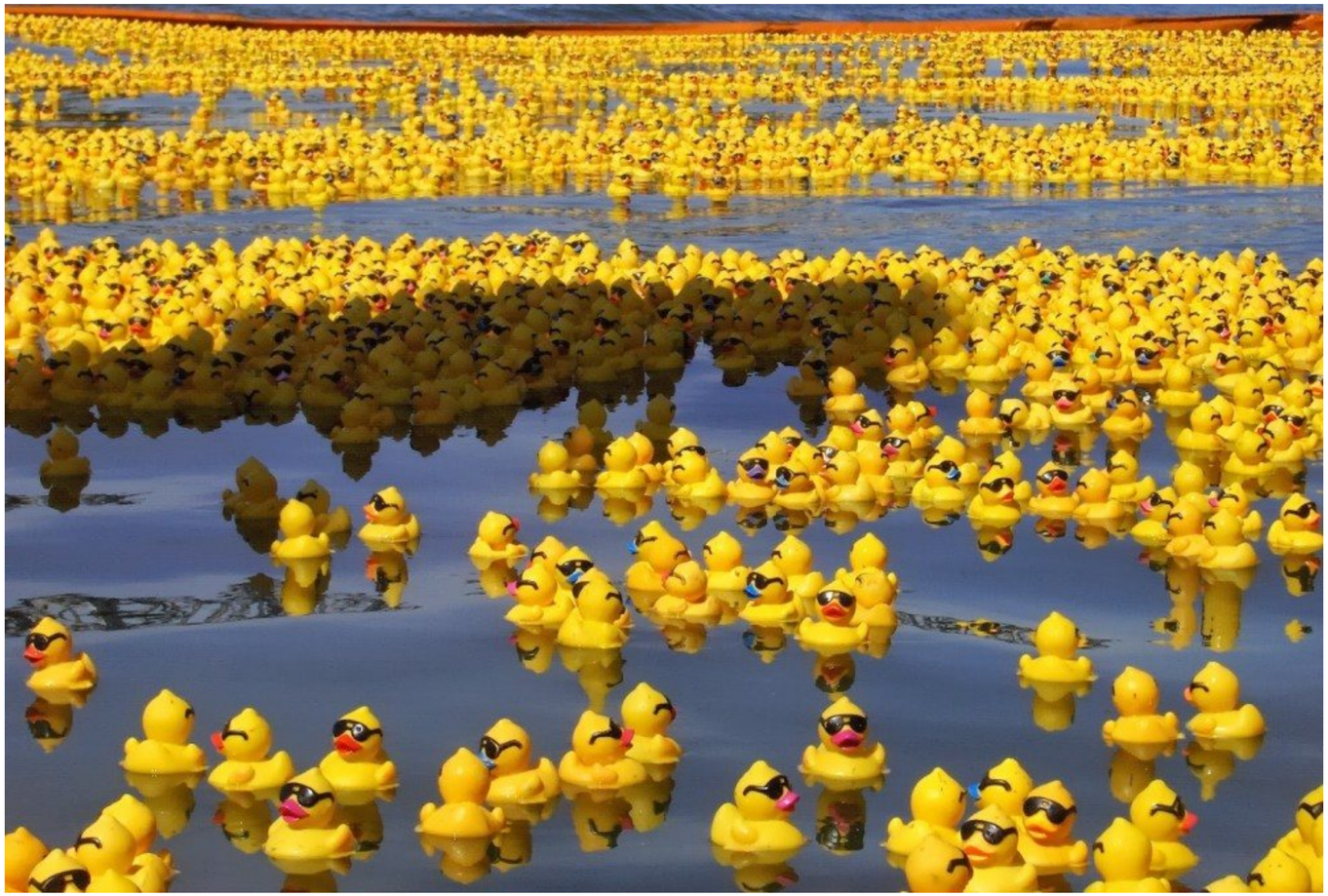
Why?

Monolit ;)



Why?

Microservices ;)



Observability

- Monitoring
- Logging
- Tracing

Observability

	Metrics	Logging	Tracing
CapEx	Medium	Low	High
OpEx	Low	High	Medium
Reaction	High	Medium	Low
Investigation	Low	Medium	High

Go for Industrial Programming by Peter Bourgon.

Not a silver-bullet

but:

- Easy to setup
- Immediately value

Surprisingly: the last one implemented

Centralized Logging

- Like a debugging vs testing
- Post-mortem
- Hard to find the needle
- Hight CTO

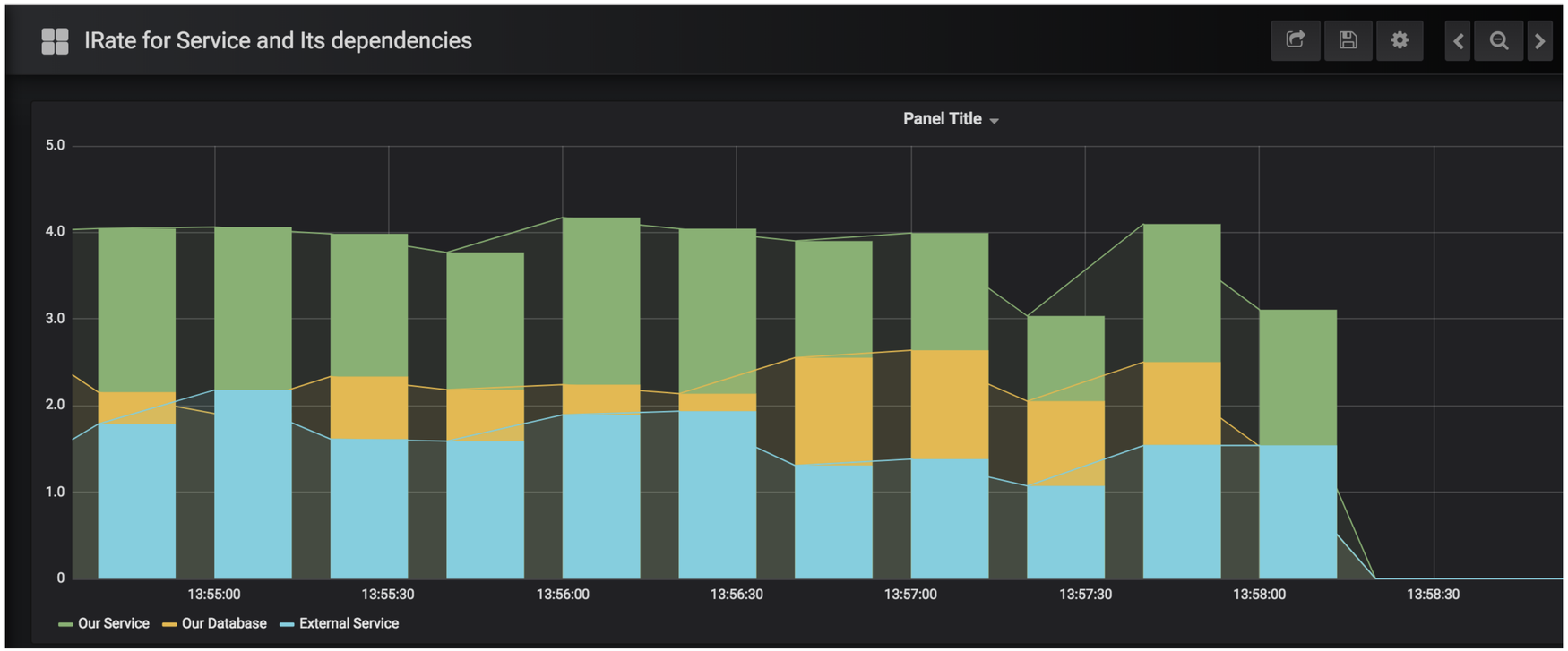
Monitoring

- Numbers
- Trends
- Dependencies
- + Actions

Metric

Name	Label	Value
traefik_requests_total	code="200", method="GET"	3001

Monitoring



Demo app

Monitoring

Example from couchbase blog

How to find the right metric?

How to find the right metric?

- USE
- RED

USE

Utilization	the average time that the resource was busy servicing work
Saturation	extra work which it can't service, often queued
Errors	the count of error events

Documented and Promoted by Berdan Gregg

USE

- **U**tilization: as a percent over a time interval: "one disk is running at 90% utilization".
- **S**aturation:
- **E**rrors:

USE

- **Utilization**:
- **Saturation**: as a queue length. eg, "the CPUs have an average run queue length of four".
- **Errors**:

USE

- **Utilization**:
- **Saturation**:
- **Errors**: scalar counts. eg, "this network interface drops packages".

USE

- **traditionaly** more instance oriented
- still useful in the microservices world

RED

Rate	How busy is your service?
Error	Errors
Duration	What is the latency of my service?

Tom Wilkie's guideline for instrumenting applications.

RED

- **Rate** - how many request per seconds handled
- **Error**
- **Duration** (distribution)

RED

- **Rate**
- **Error** - how many request per seconds handled we failed
- **Duration**

RED

- **Rate**
- **Error**
- **Duration** - how long the requests took

RED

- Follow Four Golden Signals by Google SREs [1]
- Focus on what matters for end-users, their experience
- Business metrics

[1] Latency, Traffic, Errors, Saturation (src)

RED

Not recommended for:

- batch-oriented
- streaming services

Prometheus

What prometheus is?

- Aggregation of time-series data
- Not an event-based system

Prometheus stack

- Prometheus - collect
- Alertmanager - alerts
- Grafana - visualize

Prometheus

- Plain text,
- Metrics collected over HTTP metrics/,
- Pull model [1],
- Wide support for languages,
- PromQL,
- Integration with Kubernetes, Traefik, ...

[1] See *scrape time*, push-mode possible

Metrics in plain text

```
# HELP order_mgmt_audit_duration_seconds Multiprocess metric
# TYPE order_mgmt_audit_duration_seconds summary
order_mgmt_audit_duration_seconds_count{status_code="200"} 41.
order_mgmt_audit_duration_seconds_sum{status_code="200"} 27.44
order_mgmt_audit_duration_seconds_count{status_code="500"} 1.0
order_mgmt_audit_duration_seconds_sum{status_code="500"} 0.716
# HELP order_mgmt_duration_seconds Multiprocess metric
# TYPE order_mgmt_duration_seconds summary
order_mgmt_duration_seconds_count{method="GET",path="/complex"
order_mgmt_duration_seconds_sum{method="GET",path="/complex",s
order_mgmt_duration_seconds_count{method="GET",path="/",status
order_mgmt_duration_seconds_sum{method="GET",path="/",status_c
order_mgmt_duration_seconds_count{method="GET",path="/complex"
order_mgmt_duration_seconds_sum{method="GET",path="/complex",s
```

Metrics in plain text

```
# HELP go_gc_duration_seconds A summary of the GC invocation d
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 9.01e-05
go_gc_duration_seconds{quantile="0.25"} 0.000141101
go_gc_duration_seconds{quantile="0.5"} 0.000178902
go_gc_duration_seconds{quantile="0.75"} 0.000226903
go_gc_duration_seconds{quantile="1"} 0.006099658
go_gc_duration_seconds_sum 18.749046756
go_gc_duration_seconds_count 89273
```

Exporters

- Mongodb
- Mysql
- Postgresql
- Rabbitmq
- ...
- also Blackbox exporter

Exporters and integrations

Prometheus PromQL

Working with histograms:

```
histogram_quantile(0.9,
  rate(http_req_duration_seconds_bucket[10m]))
```

Rates:

```
rate(http_requests_total{job="api-server"}[5m])
  irate(http_requests_total{job="api-server"}[5m])
```

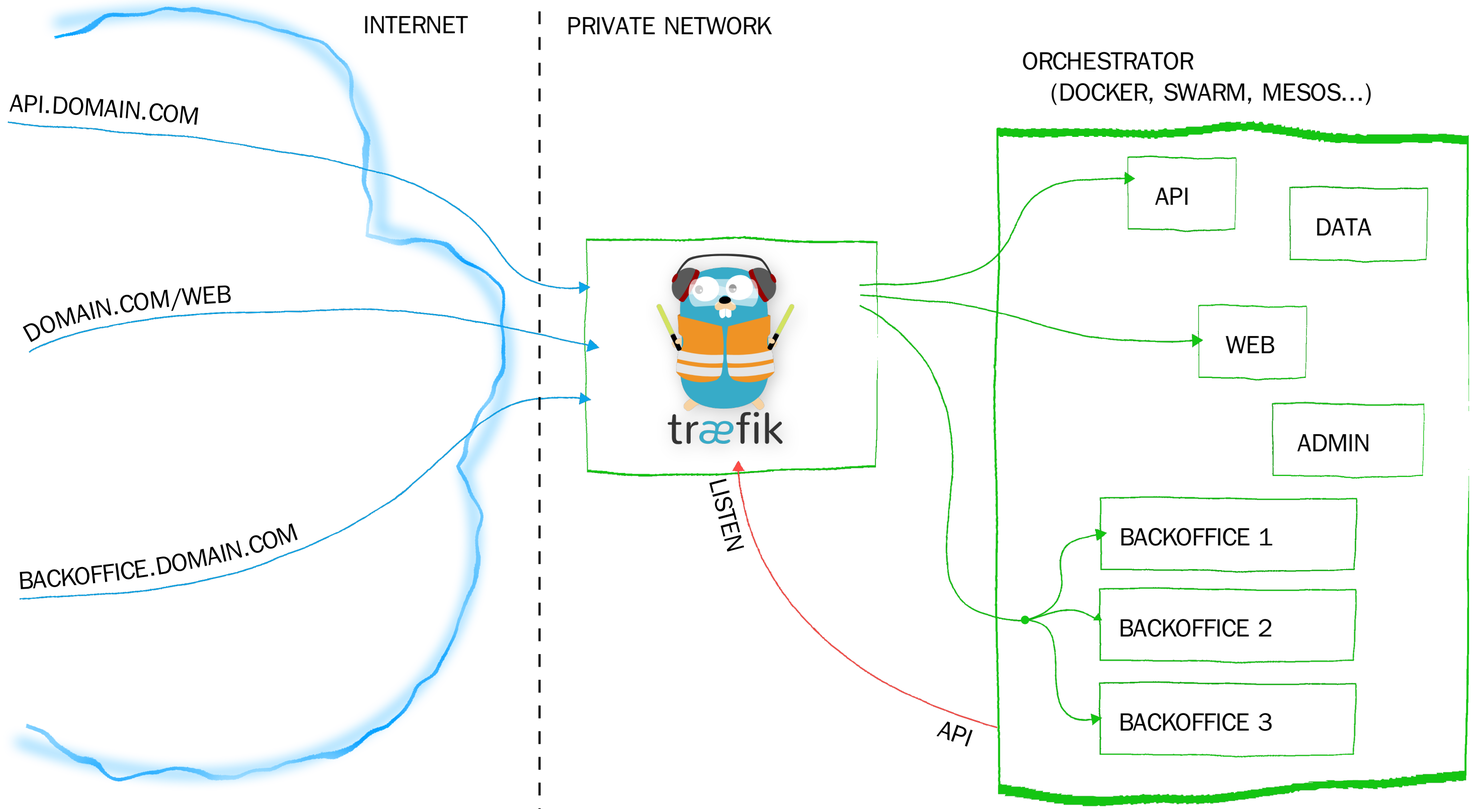
More complex - `redict_linear` and `holt_winters`.

Prometheus PromQL

Alarming:

```
ALERT ProductionAppServiceInstanceDown
IF up { environment = "production", app =~ ".+"} == 0
FOR 4m
ANNOTATIONS {
     summary = "Instance of {{$labels.app}} is down",
     description = " Instance  {{$labels.instance}} of app
}
```

Cloud-native projects integration



traefik: `--web.metrics.prometheus`

Metrics

- Counter - just up
- Gauge - up/down
- Histogram
- Summary

Histogram

`traefik_duration_seconds_bucket {method="GET,code="200"}`

{le="0.1"}	2229
{le="0.3"}	107
{le="1.2"}	100
{le="5"}	4
{le="+Inf"}	2
_sum	
_count	2342

Summary

`http_request_duration_seconds`

{quantile="0.5"}	4
{quantile="0.9"}	5
http_request_duration_seconds_sum	9
http_request_duration_seconds_count	3

Histogram / Summary:

- Latency of services
- Request or Request size

Histograms recommended

RED

Metric + PromQL:

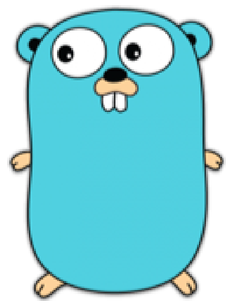
```
sum(irate(order_mgmt_duration_seconds_count
{job=~".*"}[1m])) by (status_code)
```

Metric and label naming

Best practices on metric names:

- service name is your prefix `user_`
- state the bae unit `_seconds` and `_bytes`

Prometheus + Golang

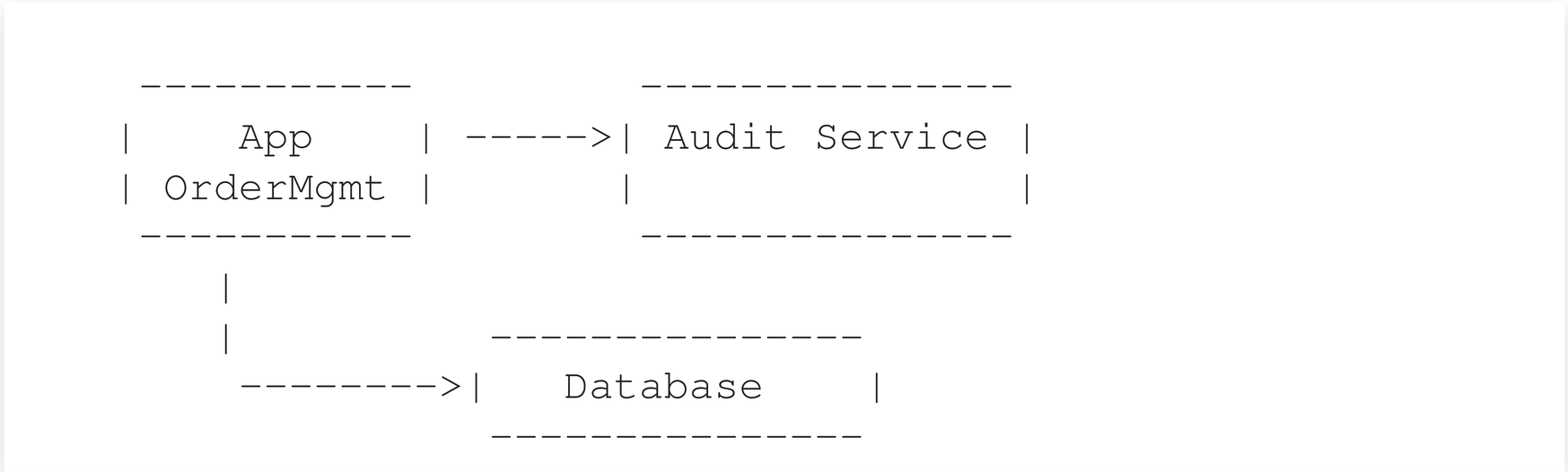


Go

Golang client

- github.com/prometheus/client_golang

Demo: Simple REST service



Demo

- http://127.0.0.1:8080/hello - service
- http://127.0.0.1:8080/metrics
- http://127.0.0.1:9090 - prometheus
- http://127.0.0.1:3000 - grafana (admin:secret)
- http://127.0.0.1:9093 - alertmanager

Demo

```
☁  demo ⚡ make start

☁  demo ⚡ docker ps
CONTAINER ID        IMAGE                               PORTS
5f824d1bc789        grafana/grafana:10.0.0      0.0.0.0:3000->3
d681a414a8b6        prom/prometheus:v2.44.0      0.0.0.0:9090->
ea0d9233e159        prom/alertmanager:v0.25.0   0.0.0.0:9093->9
732c59fb3753        order-manager    0.0.0.0:8080->8080/tcp
```

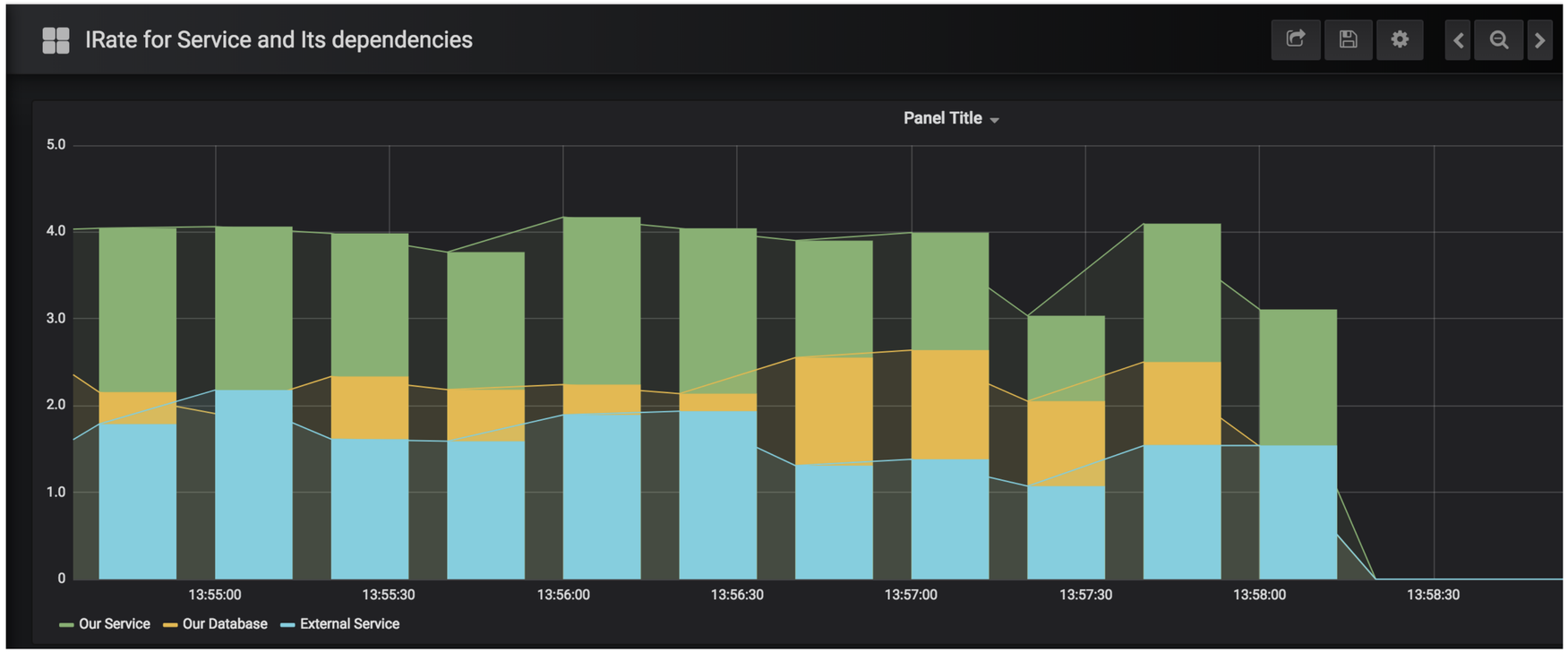
Demo: generate calls

```
☁  demo  ⚡ make srv_wrk_random
```

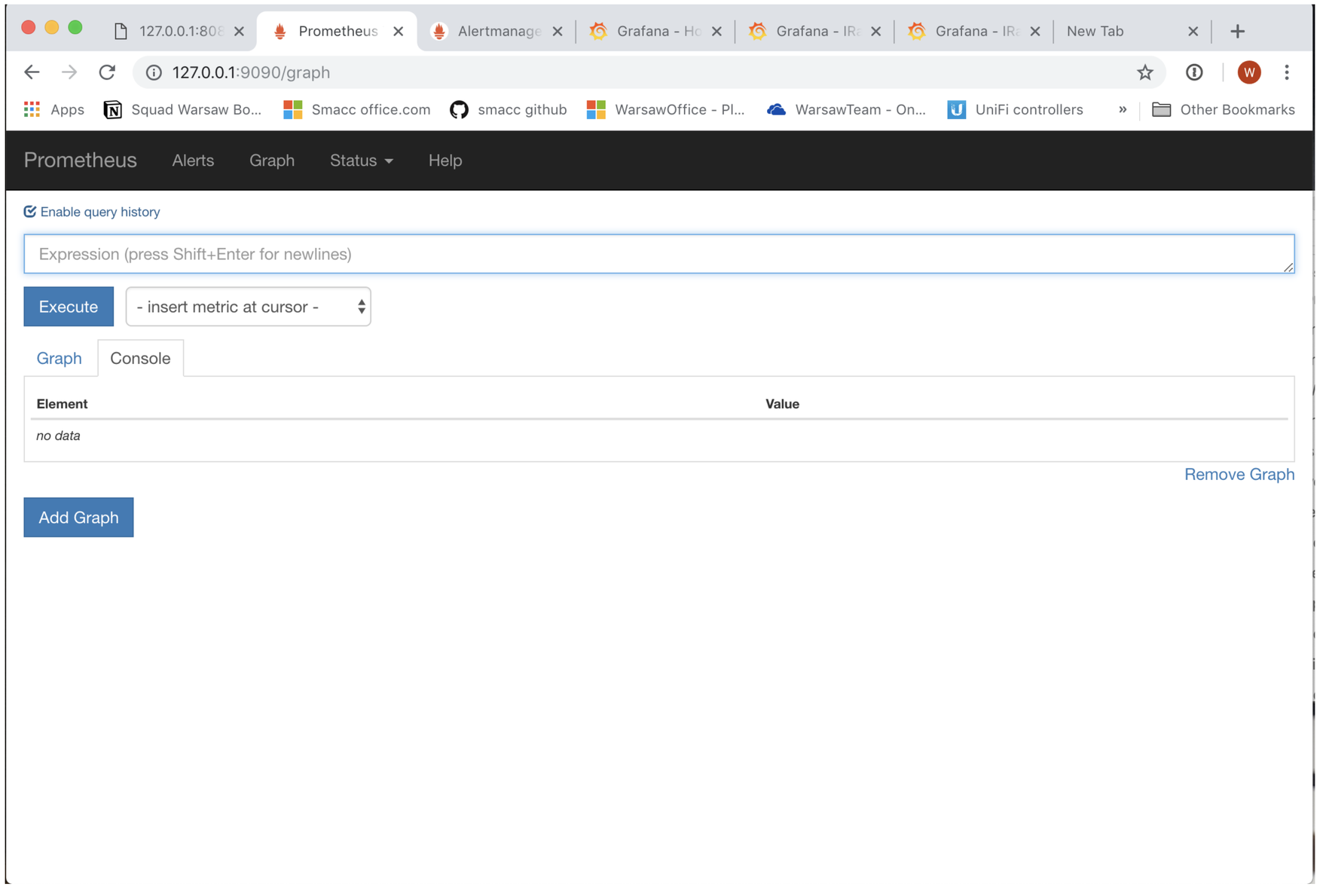
With error injection

```
# HELP order_mgmt_database_duration_seconds Multiprocess metric
# TYPE order_mgmt_database_duration_seconds histogram
order_mgmt_database_duration_seconds_sum{sql_state="0",status_code="0"} 338.9096608161926
order_mgmt_database_duration_seconds_sum{sql_state="HY000",status_code="1001"} 92.89293241500854
order_mgmt_database_duration_seconds_bucket{le="0.1",sql_state="0",status_code="0"} 72.0
order_mgmt_database_duration_seconds_bucket{le="0.25",sql_state="0",status_code="0"} 168.0
order_mgmt_database_duration_seconds_bucket{le="0.5",sql_state="0",status_code="0"} 287.0
order_mgmt_database_duration_seconds_bucket{le="0.75",sql_state="0",status_code="0"} 459.0
order_mgmt_database_duration_seconds_bucket{le="0.9",sql_state="0",status_code="0"} 573.0
order_mgmt_database_duration_seconds_bucket{le="1.0",sql_state="0",status_code="0"} 646.0
order_mgmt_database_duration_seconds_bucket{le="2.5",sql_state="0",status_code="0"} 648.0
order_mgmt_database_duration_seconds_bucket{le="+Inf",sql_state="0",status_code="0"} 648.0
order_mgmt_database_duration_seconds_count{sql_state="0",status_code="0"} 648.0
order_mgmt_database_duration_seconds_bucket{le="0.1",sql_state="HY000",status_code="1001"} 24.0
order_mgmt_database_duration_seconds_bucket{le="0.25",sql_state="HY000",status_code="1001"} 86.0
order_mgmt_database_duration_seconds_bucket{le="0.5",sql_state="HY000",status_code="1001"} 134.0
order_mgmt_database_duration_seconds_bucket{le="0.75",sql_state="HY000",status_code="1001"} 190.0
order_mgmt_database_duration_seconds_bucket{le="0.9",sql_state="HY000",status_code="1001"} 217.0
order_mgmt_database_duration_seconds_bucket{le="1.0",sql_state="HY000",status_code="1001"} 225.0
order_mgmt_database_duration_seconds_bucket{le="2.5",sql_state="HY000",status_code="1001"} 225.0
order_mgmt_database_duration_seconds_bucket{le="+Inf",sql_state="HY000",status_code="1001"} 225.0
order_mgmt_database_duration_seconds_count{sql_state="HY000",status_code="1001"} 225.0
# HELP order_mgmt_audit_duration_seconds Multiprocess metric
# TYPE order_mgmt_audit_duration_seconds summary
order_mgmt_audit_duration_seconds_count{status_code="200"} 490.0
order_mgmt_audit_duration_seconds_sum{status_code="200"} 231.30700039863586
order_mgmt_audit_duration_seconds_count{status_code="500"} 158.0
order_mgmt_audit_duration_seconds_sum{status_code="500"} 99.42281174659729
# HELP order_mgmt_duration_seconds Multiprocess metric
# TYPE order_mgmt_duration_seconds summary
order_mgmt_duration_seconds_count{method="GET",path="/complex",status_code="200"} 490.0
order_mgmt_duration_seconds_sum{method="GET",path="/complex",status_code="200"} 471.54110622406006
order_mgmt_duration_seconds_count{method="GET",path="/complex",status_code="503"} 383.0
order_mgmt_duration_seconds_sum{method="GET",path="/complex",status_code="503"} 291.444673538208
```

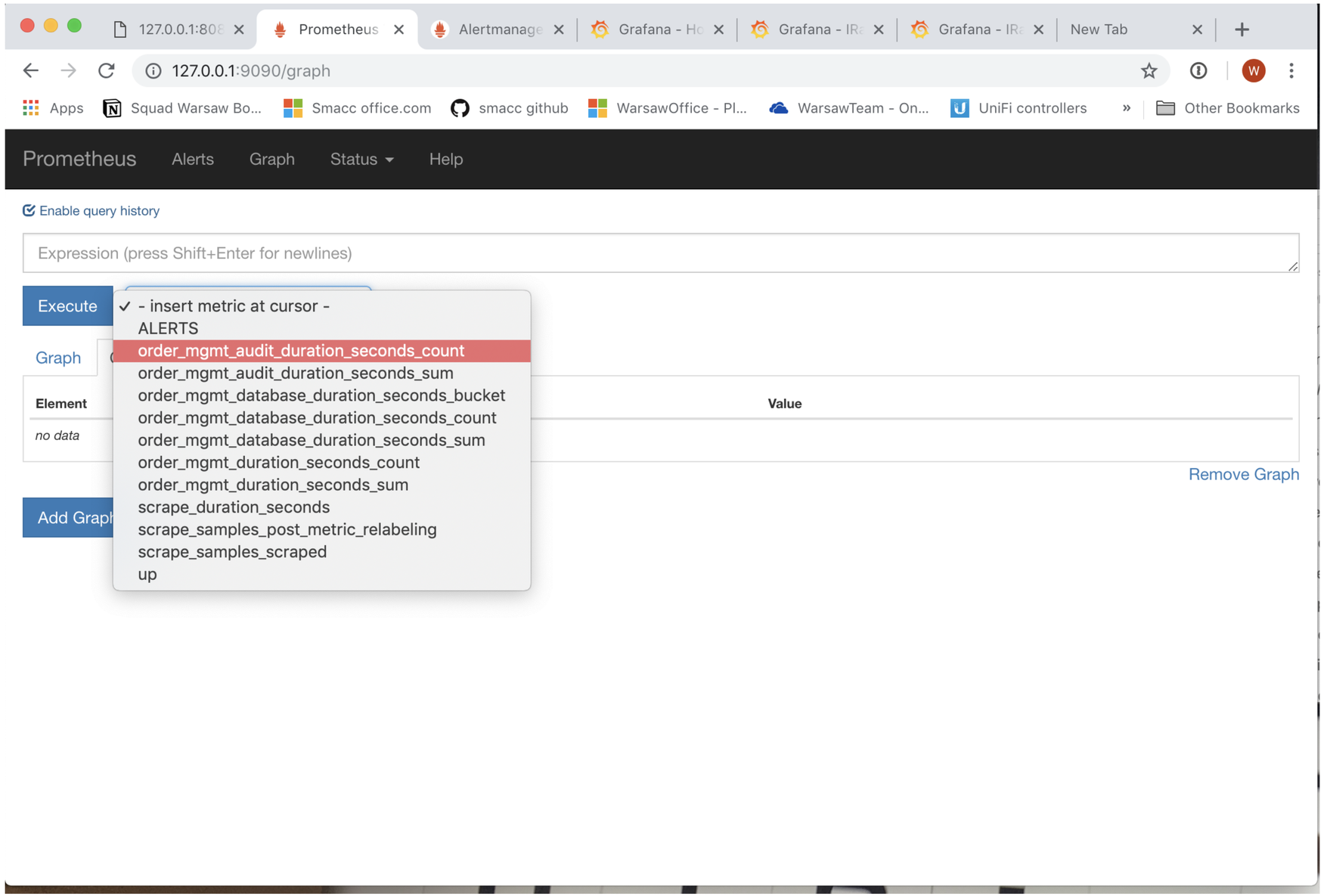
Grafana


IRate for Service and Its dependencies
Panel Title
5.0
4.0
3.0
2.0
1.0
0
13:55:00
13:55:30
13:56:00
13:56:30
13:57:00
13:57:30
13:58:00
13:58:30
Our Service
Our Database
External Service

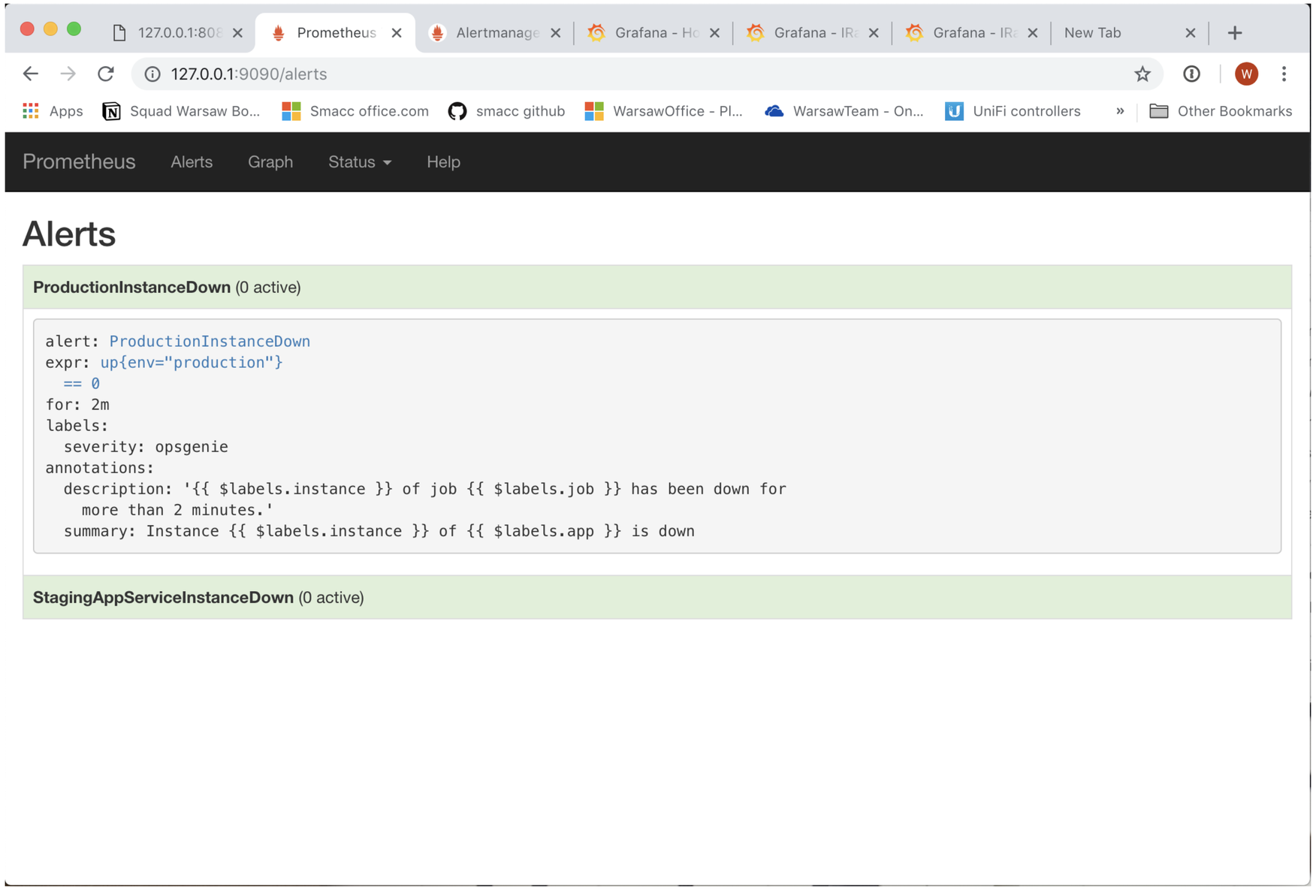
Prometheus


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:9090/graph
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Enable query history
Expression (press Shift+Enter for newlines)
Execute
- insert metric at cursor -
Graph
Console
Element
Value
no data
Remove Graph
Add Graph

Prometheus


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:9090/graph
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Enable query history
Expression (press Shift+Enter for newlines)
Execute
- insert metric at cursor -
ALERTS
order_mgmt_audit_duration_seconds_count
order_mgmt_audit_duration_seconds_sum
order_mgmt_database_duration_seconds_bucket
order_mgmt_database_duration_seconds_count
order_mgmt_database_duration_seconds_sum
order_mgmt_duration_seconds_count
order_mgmt_duration_seconds_sum
scrape_duration_seconds
scrape_samples_post_metric_relabeling
scrape_samples_scraped
up
Graph
Element
Value
no data
Remove Graph
Add Graph

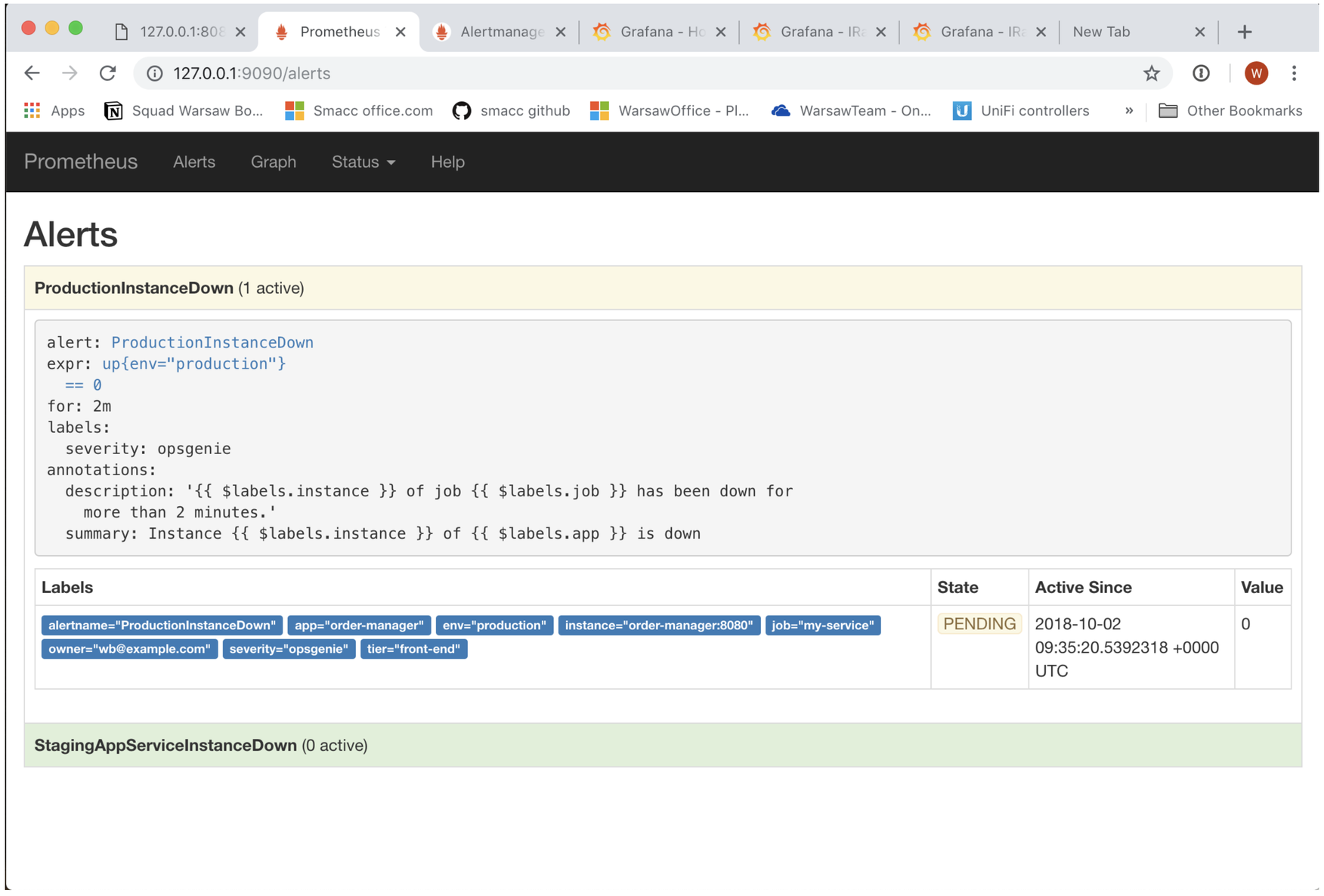
Prometheus


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:9090/alerts
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Alerts
ProductionInstanceDown (0 active)
alert: ProductionInstanceDown
expr: up{env="production"}
== 0
for: 2m
labels:
severity: opsgenie
annotations:
description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for
more than 2 minutes.'
summary: Instance {{ $labels.instance }} of {{ $labels.app }} is down
StagingAppServiceInstanceDown (0 active)

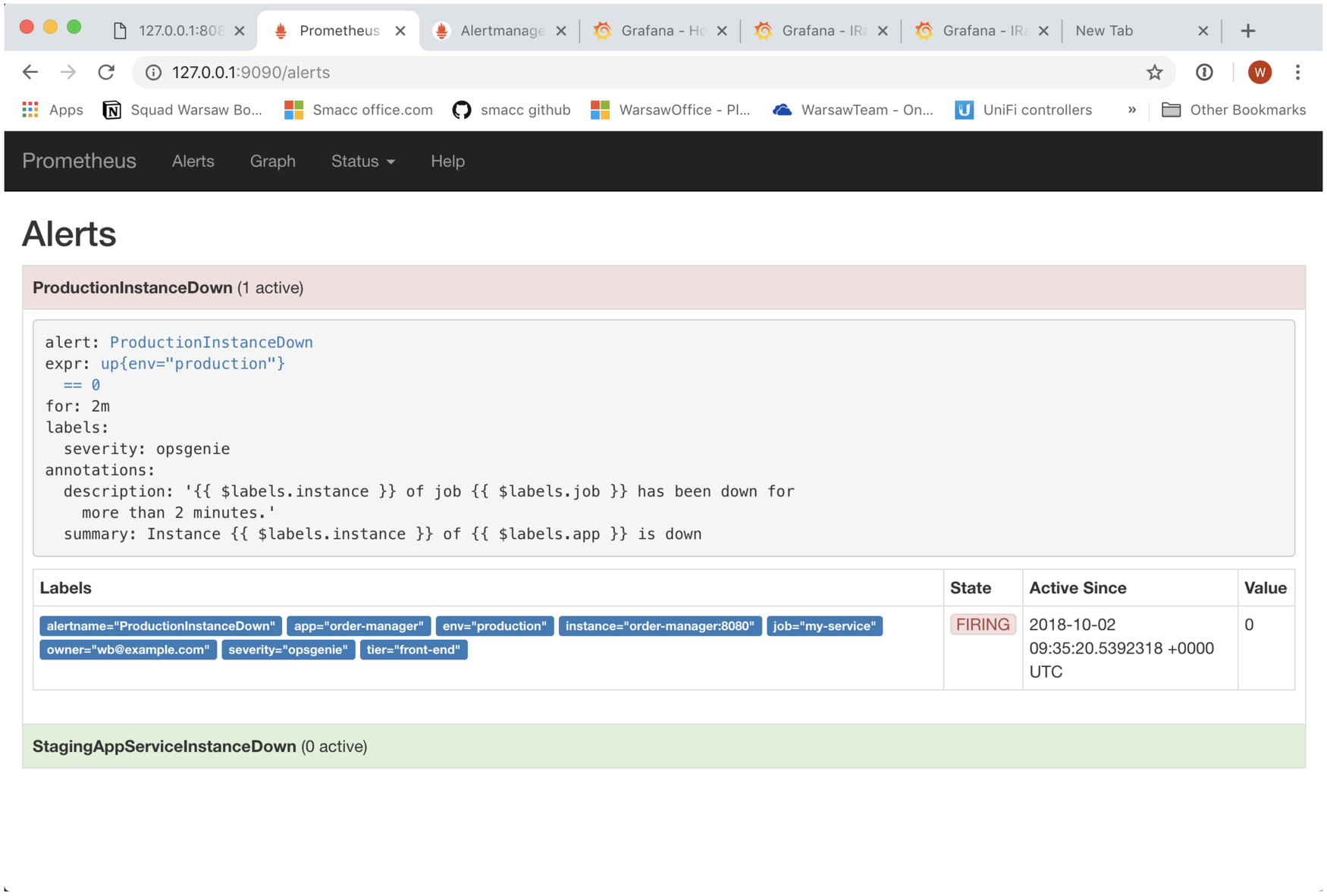
Kill the service

```
☁  demo ⚡ docker stop talk-golang-prom-red_alertmanager_1
```

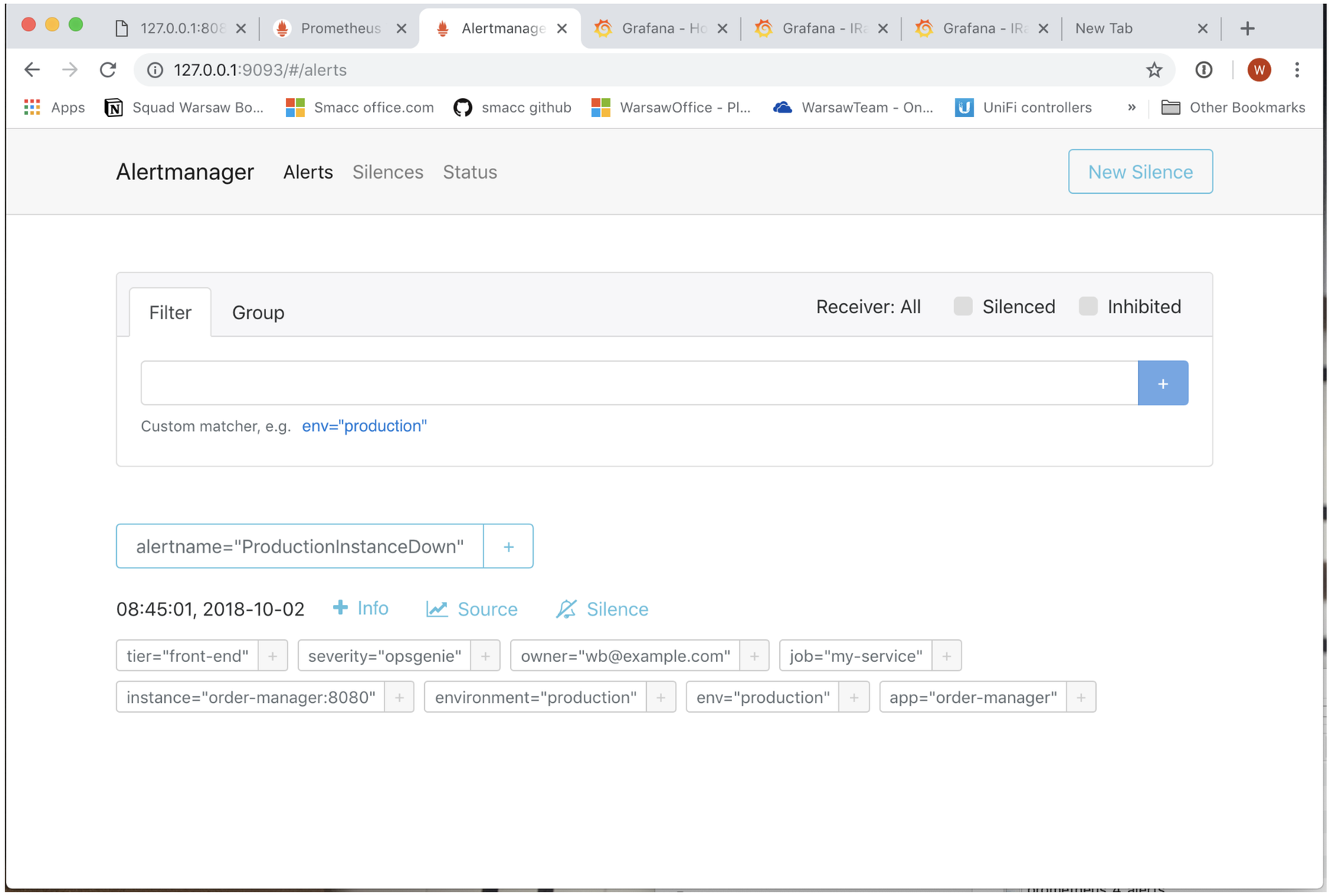
Prometheus


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:9090/alerts
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Alerts
ProductionInstanceDown (1 active)
alert: ProductionInstanceDown
expr: up{env="production"}
== 0
for: 2m
labels:
severity: opsgenie
annotations:
description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for
more than 2 minutes.'
summary: Instance {{ $labels.instance }} of {{ $labels.app }} is down
Labels
State
Active Since
Value
alertname="ProductionInstanceDown"
app="order-manager"
env="production"
instance="order-manager:8080"
job="my-service"
owner="wb@example.com"
severity="opsgenie"
tier="front-end"
PENDING
2018-10-02 09:35:20.5392318 +0000 UTC
0
StagingAppServiceInstanceDown (0 active)

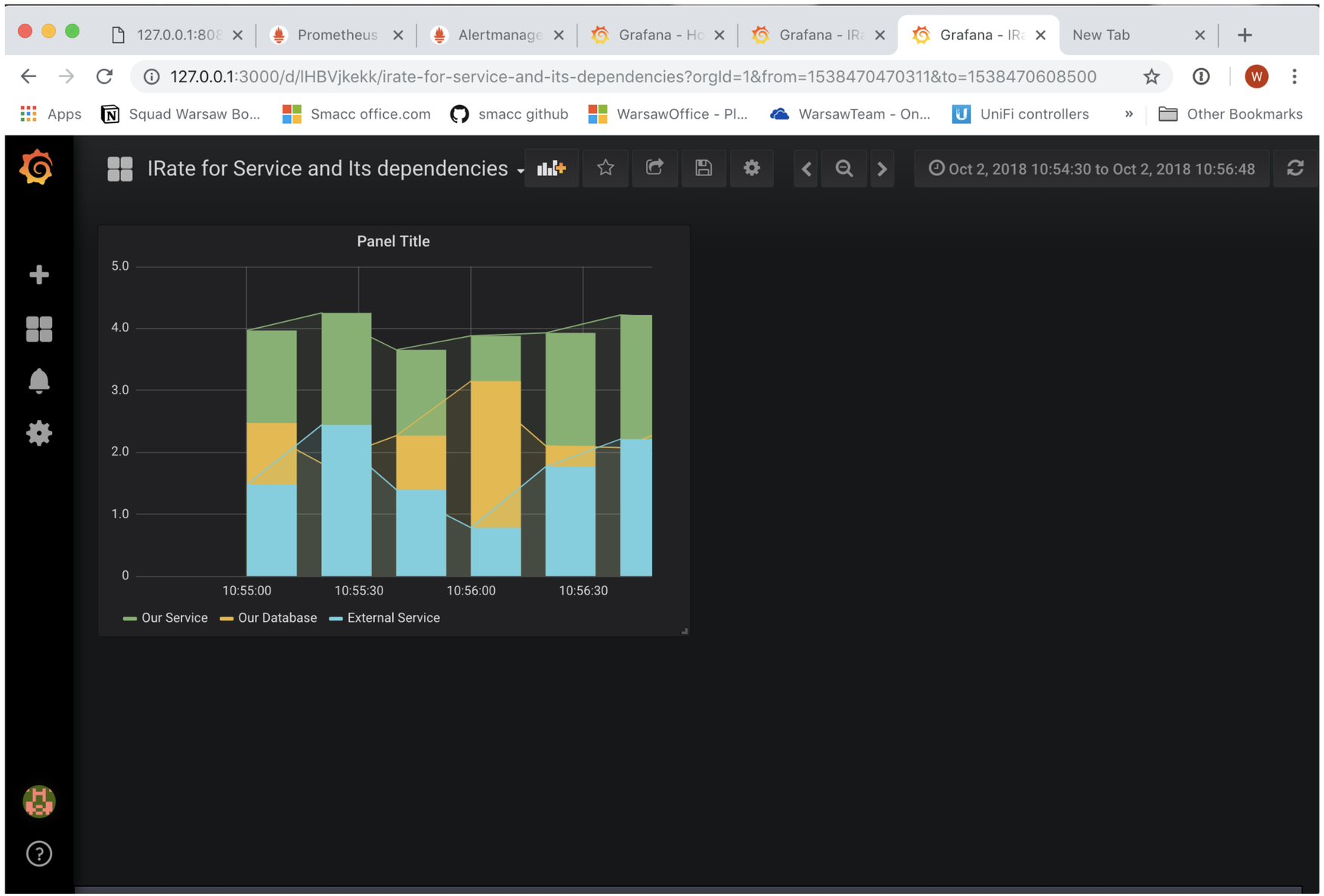
Prometheus


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:9090/alerts
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Alerts
ProductionInstanceDown (1 active)
alert: ProductionInstanceDown
expr: up{env="production"}
== 0
for: 2m
labels:
severity: opsgenie
annotations:
description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for
more than 2 minutes.'
summary: Instance {{ $labels.instance }} of {{ $labels.app }} is down
Labels
State
Active Since
Value
alertname="ProductionInstanceDown"
app="order-manager"
env="production"
instance="order-manager:8080"
job="my-service"
owner="wb@example.com"
severity="opsgenie"
tier="front-end"
FIRING
2018-10-02 09:35:20.5392318 +0000 UTC
0
StagingAppServiceInstanceDown (0 active)

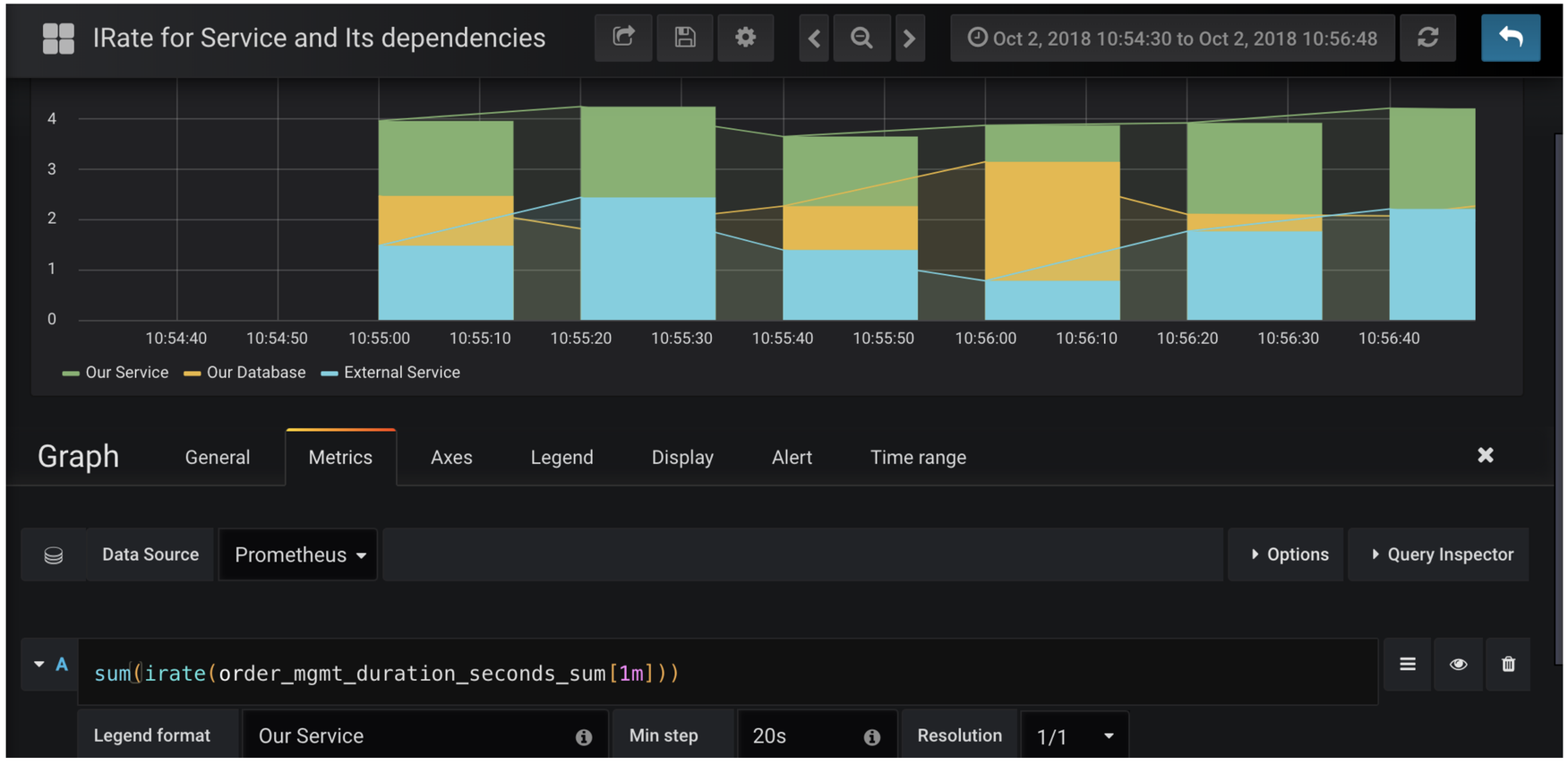
Alertmanager



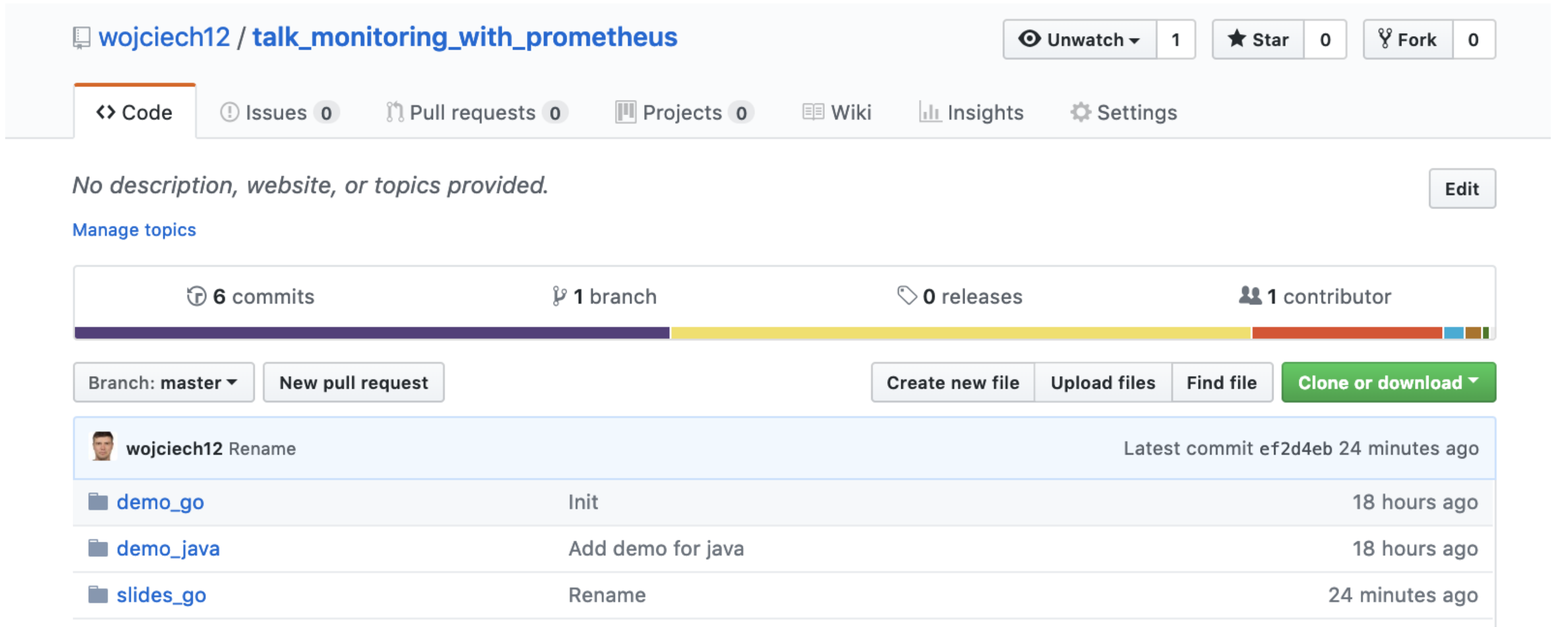
Grafana


127.0.0.1:808
Prometheus
Alertmanage
Grafana - Ho
Grafana - IRa
Grafana - IRa
New Tab
127.0.0.1:3000/d/lHBVjkekk/irate-for-service-and-its-dependencies?orgId=1&from=1538470470311&to=1538470608500
Apps
Squad Warsaw Bo...
Smacc office.com
smacc github
WarsawOffice - Pl...
WarsawTeam - On...
UniFi controllers
Other Bookmarks
IRate for Service and Its dependencies
Oct 2, 2018 10:54:30 to Oct 2, 2018 10:56:48
Panel Title
5.0
4.0
3.0
2.0
1.0
0
10:55:00
10:55:30
10:56:00
10:56:30
Our Service
Our Database
External Service

Grafana


IRate for Service and Its dependencies
Oct 2, 2018 10:54:30 to Oct 2, 2018 10:56:48
4
3
2
1
0
10:54:40
10:54:50
10:55:00
10:55:10
10:55:20
10:55:30
10:55:40
10:55:50
10:56:00
10:56:10
10:56:20
10:56:30
10:56:40
Our Service
Our Database
External Service
Graph
General
Metrics
Axes
Legend
Display
Alert
Time range
Data Source
Prometheus
Options
Query Inspector
A
sum(irate(order_mgmt_duration_seconds_sum[1m]))
Legend format
Our Service
Min step
20s
Resolution
1/1

Github


wojciech12 / talk_monitoring_with_prometheus
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
No description, website, or topics provided.
Edit
Manage topics
6 commits
1 branch
0 releases
1 contributor
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
wojciech12 Rename
Latest commit ef2d4eb 24 minutes ago
demo_go Init 18 hours ago
demo_java Add demo for java 18 hours ago
slides_go Rename 24 minutes ago

Demo

Everything Included:

- Prometheus dashboard and config
- AlertManager dashboard and config
- Simulate the successful and failed calls
- Examples of queries for rate

Best practices

- Start simple (up/down), later add more complex rules
- Sum over Summaries with Q leads to incorrect results, see prom docs

Summary

- Monitoring saves your time
- Checking logs == debuging vs having tests
- Logging -> high TCO

Questions?

Backup slides

Prometheus - Push model

- See:

https://prometheus.io/docs/instrumenting/pushing/

Good for short living jobs in your cluster.

Designing metric names

Which one is better?

- request_duration{app=my_app}
- my_app_request_duration

see documentation on best practises for metric naming and instrumentation

Designing metric names

Which one is better?

- order_mgmt_db_duration_seconds_sum
- order_mgmt_duration_seconds_sum{dep_name='db'}

Prometheus + K8S = <3

Labels are propagated from k8s to prometheus

Integration with Prometheus

cat memcached-0-service.yaml

```
---
  apiVersion: v1
  kind: Service
  metadata:
    name: memcached-0
    labels:
      app: memcached
      kubernetes.io/name: "memcached"
      role: shard-0
    annotations:
      prometheus.io/scrape: "true"
      prometheus.io/scheme: "http"
      prometheus.io/path: "metrics"
      prometheus.io/port: "9150"
```

https://github.com/skarab7/kubernetes-memcached