Assignment

Structure:

```
monri-sre/
 |— app/
 │   ├— Dockerfile
 │   ├— app.py
 │   └— requirements.txt
 │
 ├— grafana/
 │
 │
 ├— prometheus/
 │   ├— alert.rules.yml
 │   └— prometheus.yml
 │
 └— docker-compose.yml
```

Building containers:

```
adotahirovic@ip-192-168-1-9 monri-sre %          docker-compose down
                                                 docker-compose up -d --build
```

Checking if we have running containers:

```
adotahirovic@ip-192-168-1-9 monri-sre % docker ps
CONTAINER ID   IMAGE                 COMMAND            CREATED        STATUS       PORTS
NAMES
4fd27b33b4cc   grafana/grafana:latest     "/run.sh"            2 minutes ago   Up 2 minutes   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   grafana
890a5627f9f4   prom/prometheus:latest     "/bin/prometheus --c…"   2 minutes ago   Up 2 minutes   0.0.0.0:9090->9090/tcp, :::9090->9090/tcp   prometheus
3eaaed68c444   monri-sre-app          "python app.py"        2 minutes ago   Up 2 minutes   0.0.0.0:5000->5000/tcp, :::5000->5000/tcp   main-app
43b1ed62b812   prom/node-exporter:latest   "/bin/node_exporter"    2 minutes ago   Up 2 minutes   0.0.0.0:9100->9100/tcp, :::9100->9100/tcp   node-exporter
```

Links that we will need: Prometheus UI: http://localhost:9090
Grafana UI: http://localhost:3000

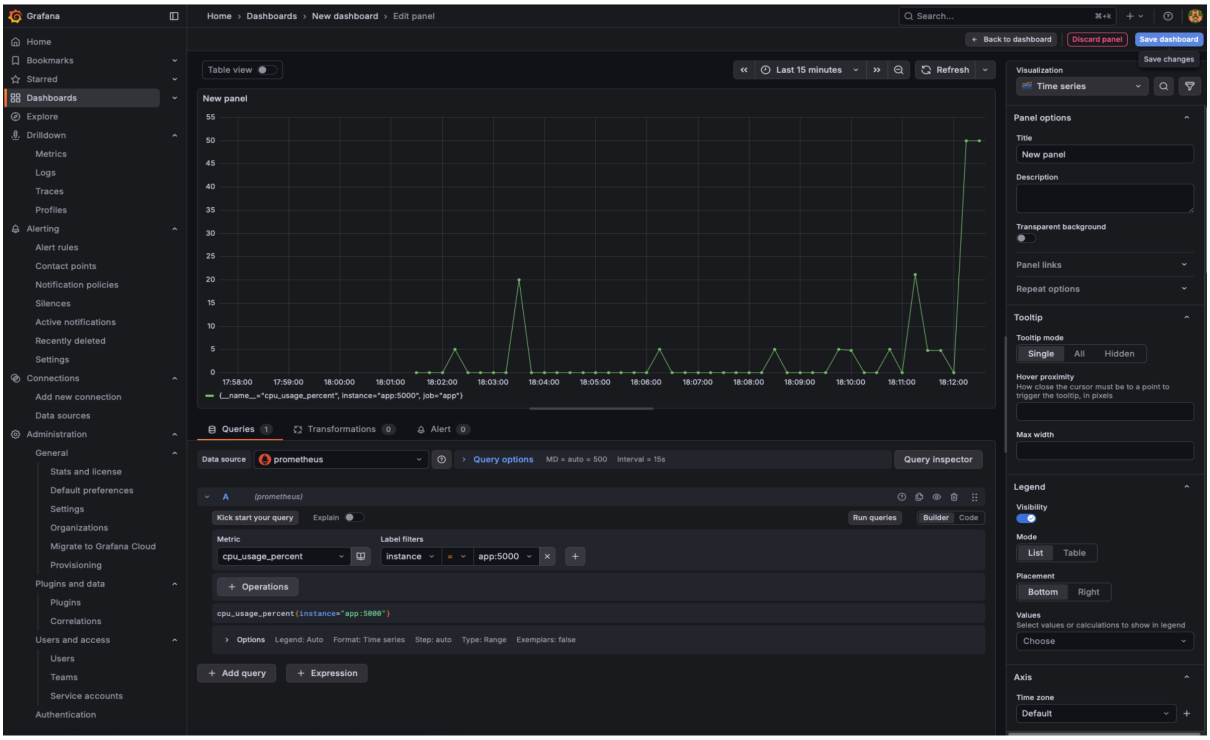
After the deployment, you can go to http://localhost:9090/targets and check if the state is UP
Then we should go into Grafana with the above link (admin is both username and password)

If you go to Connections —> Data sources —> Add data source
You can add Prometheus with this connection link http://prometheus:9090 and you can save it

For the desired metrics and alerts, I decided to add them manually because it will be in my opinion, better to explain via UI, but of course, it’s easily possible to add them via Code and just deploy

Now go to dashboards and go to “+ Create dashboard” —> “add visualisation” —> choose prometheus —> in metrics select cpu_usage_precent, lable filters should be instance and app:5000 and just save it

To test this metric/graph you can do a manual CPU test with this command “docker exec -it main-app bash -c "python -c 'while True: pass’”” pls note this command, since we will need it for the alert later on



{__name__="cpu_usage_percent", instance="app:5000", job="app"}

Data source: prometheus

MD = auto = 500 Interval = 15s

Metric: cpu_usage_percent

Label filters: instance = app:5000

```
cpu_usage_percent{instance="app:5000"}
```

Options Legend: Auto Format: Time series Step: auto Type: Range Exemplars: false

Visualization: Time series

Title: New panel

Description:

Transparent background

Tooltip mode: Single All Hidden

Hover proximity: How close the cursor must be to a point to trigger the tooltip, in pixels

Max width

Visibility

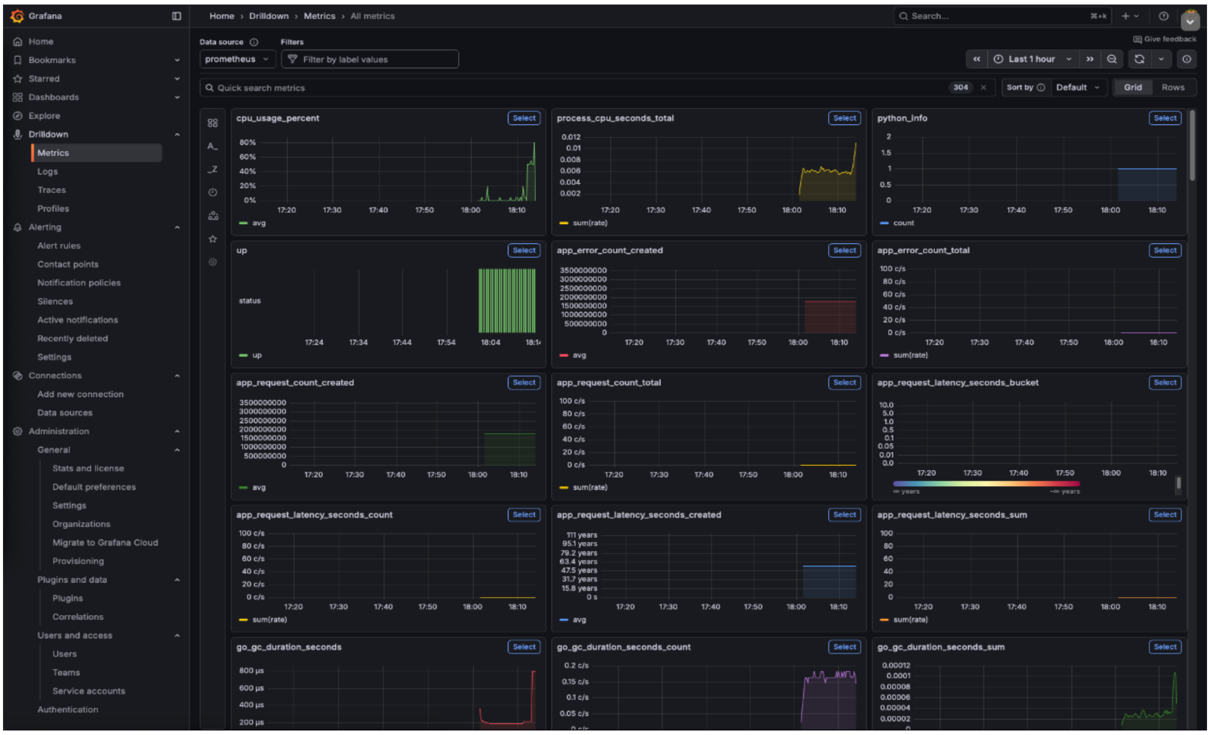
Mode: List Table

Placement: Bottom Right

Values: Select values or calculations to show in legend: Choose

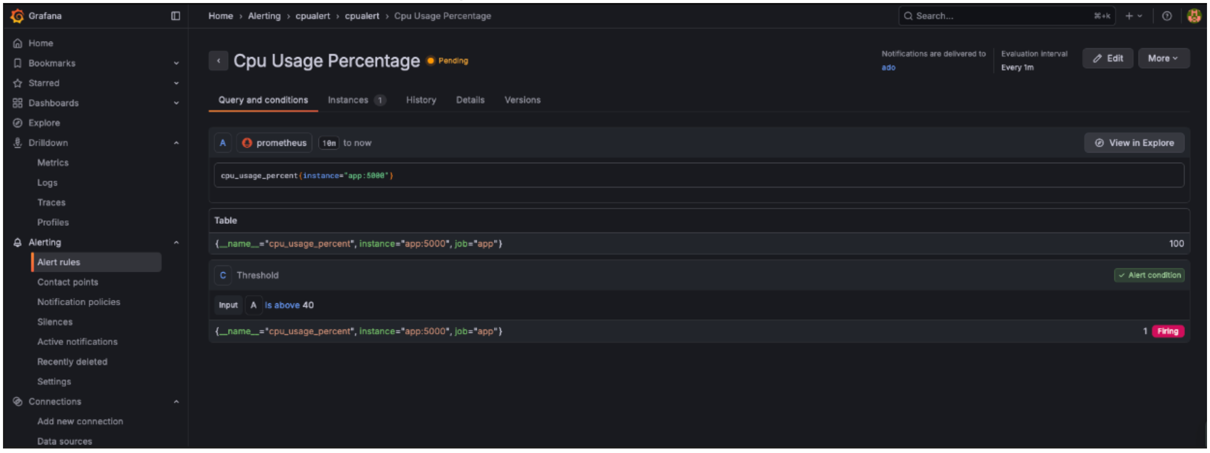
Time zone: Default

If you go into Metrics, you can see a lot of metrics



Alert creation

Let’s create an alert by going into Alerting —> Alert rules —> and set it up step by step. In this case, we did create CPU usage one, for the alert condition we will use “IS ABOVE: 40”, which should be all good. We chose a custom Contact point as well



Cpu Usage Percentage ● Pending

Notifications are delivered to ado

Evaluation interval Every 1m

A prometheus 10m to now

```
cpu_usage_percent{instance="app:5000"}
```

Table

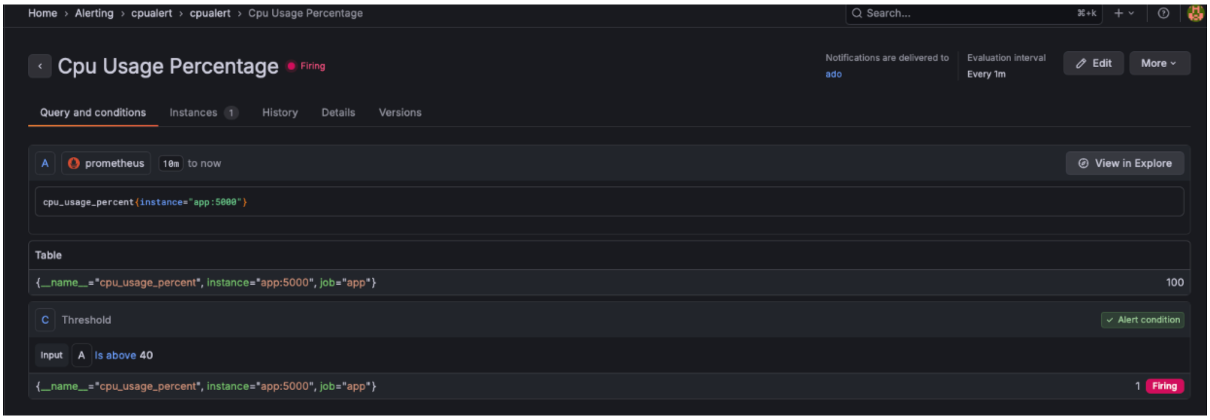
{__name__="cpu_usage_percent", instance="app:5000", job="app"} 100

C Threshold ✓ Alert condition

Input A Is above 40

{__name__="cpu_usage_percent", instance="app:5000", job="app"} 1 Firing

After doing manual load on the CPU docker exec -it main-app bash -c "python -c 'while True: pass'" with this command we can se alert is triggered, its in Firing state



Cpu Usage Percentage ● Firing

Notifications are delivered to: ado

Evaluation interval: Every 1m

A prometheus 10m to now

```
cpu_usage_percent{instance="app:5000"}
```

Table

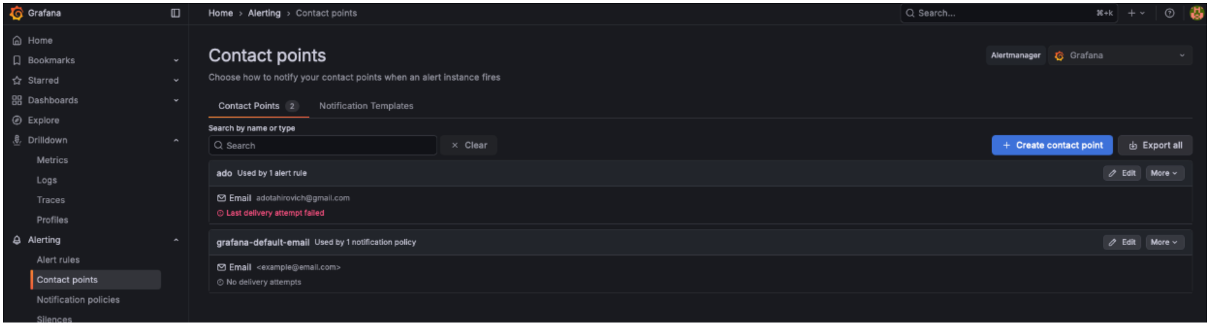
```
{__name__="cpu_usage_percent", instance="app:5000", job="app"}    100
```

C Threshold ✓ Alert condition

Input A Is above 40

```
{__name__="cpu_usage_percent", instance="app:5000", job="app"}    1 Firing
```

If we are not sure by “Firing” lets doublecheck in Contact points



Contact points

Choose how to notify your contact points when an alert instance fires

Search by name or type

ado Used by 1 alert rule

Email adotahirovich@gmail.com

Last delivery attempt failed

grafana-default-email Used by 1 notification policy

Email <example@email.com>

No delivery attempts

FYI this error is because I did not set up my emailing, which would need a bit of time, and I didnt have much time since we had a few problems at work...