ACTIVIDAD 2 - MI PRIMER DASHBOARD GRAFANA

Cristóbal Suárez Abad

OPTATIVA - 2º ASIR

Introducción y Objetivos: Una vez que Prometheus está recolectando métricas, necesitamos una herramienta que nos permita visualizar esos datos de forma profesional y atractiva. Grafana es el estándar de la industria para crear tableros (dashboards) que permiten detectar anomalías de un vistazo.

1. Despliegue de Grafana

Actualiza tu archivo `docker-compose.yml` para incluir el servicio de visualización Grafana..

- **Persistencia:** Crea un volumen llamado `grafana-data` para no perder tus dashboards al reiniciar los contenedores.

Basándonos en la web oficial de Grafana[1], incluimos esto en el docker-compose.yml del ejercicio anterior:

```
  grafana:
    image: grafana/grafana-enterprise:latest
    container_name: grafana
    restart: unless-stopped
    ports:
      - "3000:3000"
    networks:
      - back-tier
    volumes:
      - grafana-data:/var/lib/grafana

volumes:
  grafana-data: # Volumen para GRAFANA
```

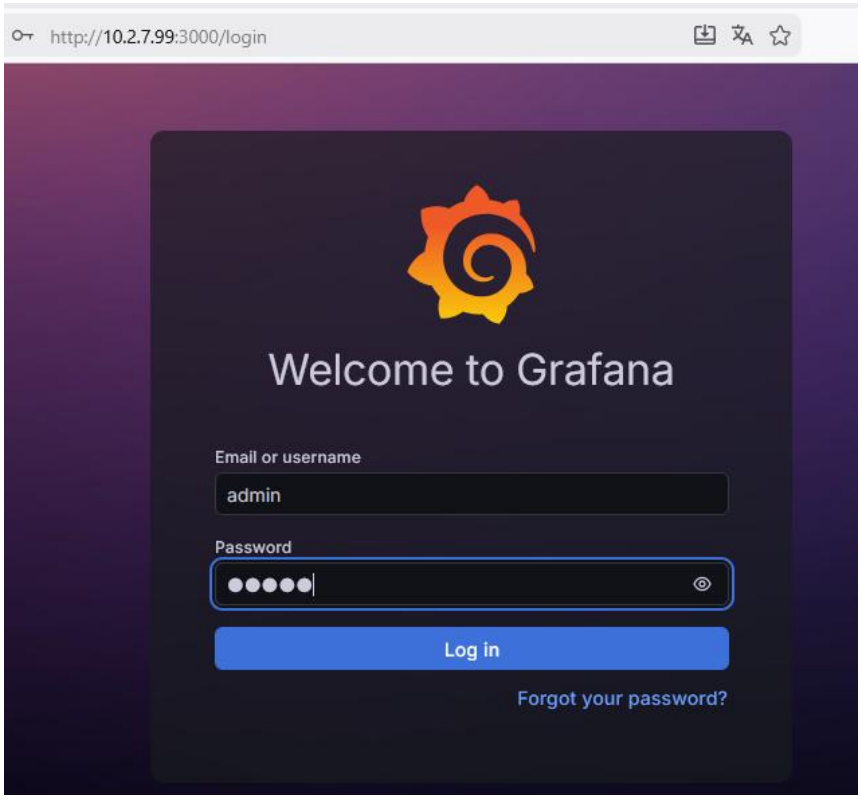
[1] https://grafana.com/docs/grafana/latest/setup-grafana/installation/docker/

2. Configuración del Data Source

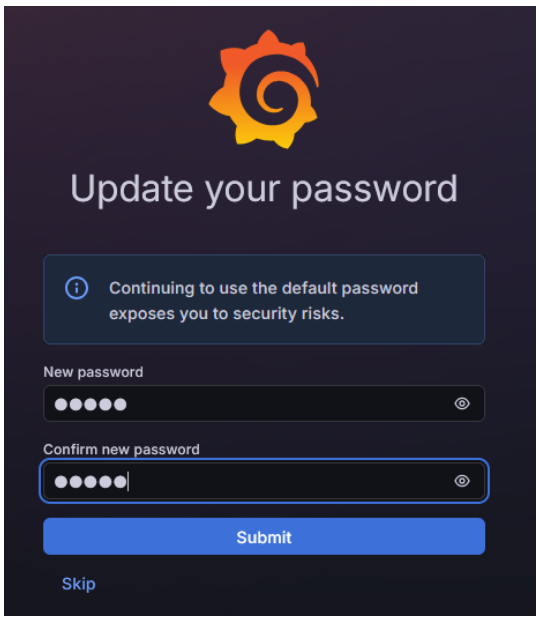
1. Accede a la aplicación web de Grafana

http://10.2.7.99:3000/login

El usuario y la contraseña es “admin” y “admin”.

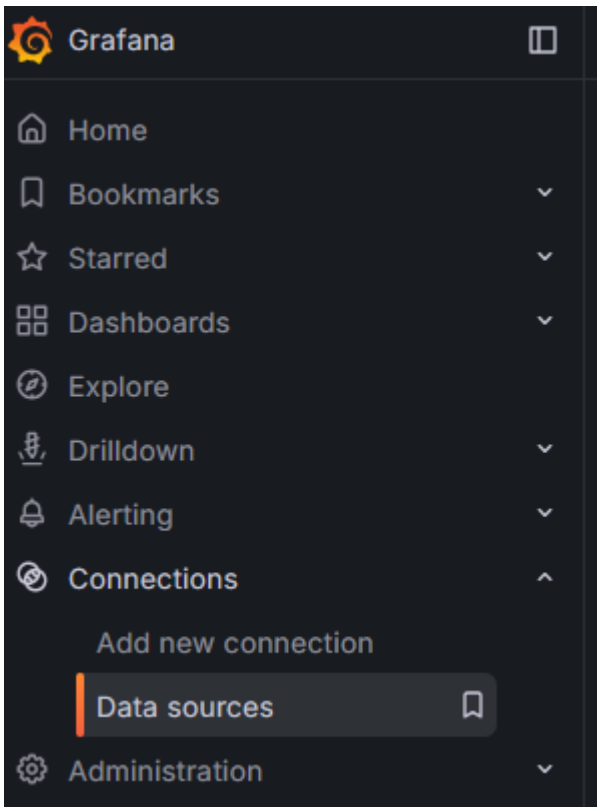


Te pedirá que establezcas una nueva contraseña para “admin”.

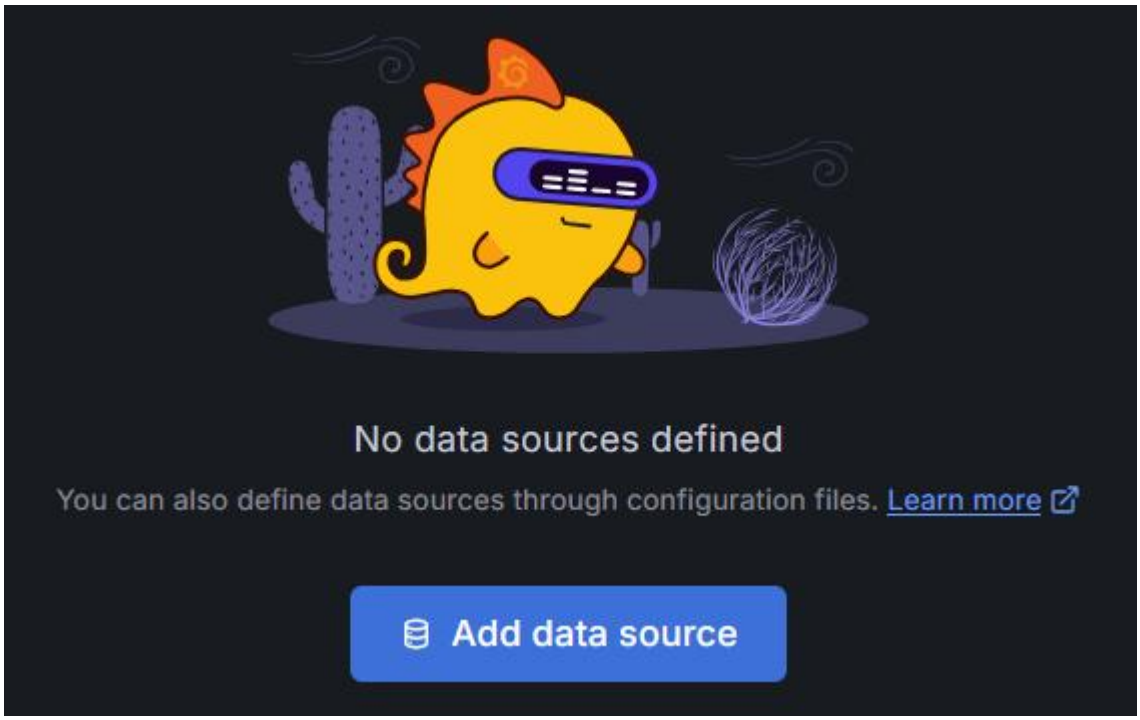


2. Añade como origen de datos Prometheus.

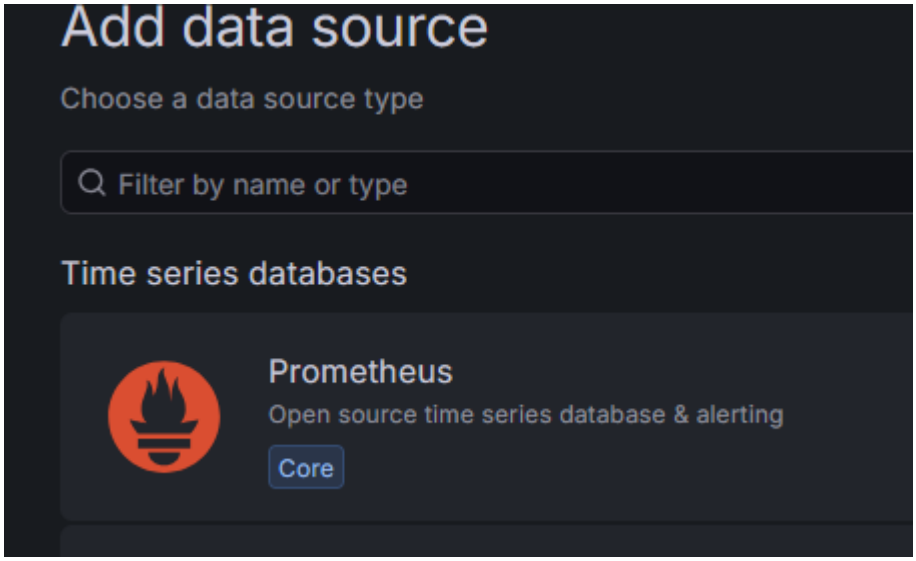
Panel de la izquierda: Connections → Data Sources



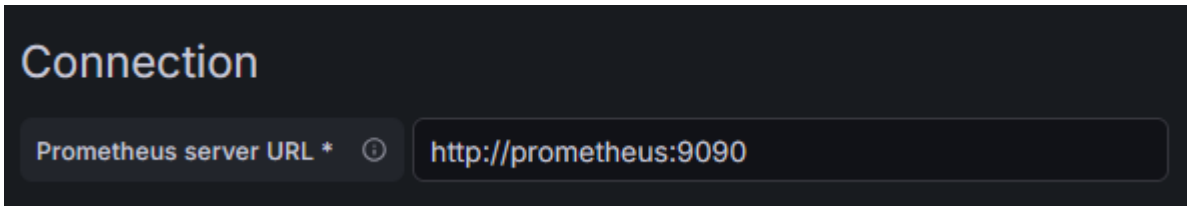
Añadimos:



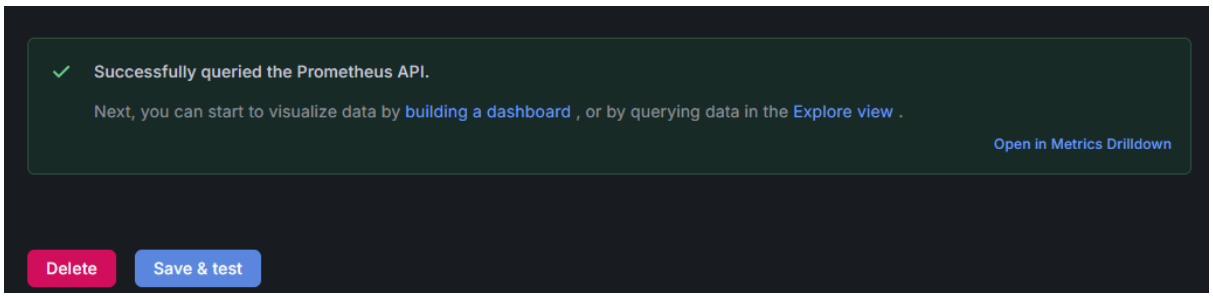
Añadimos a Prometheus:



En conexión ponemos http://prometheus:9090 Es como se identifica en la red interna de docker que hemos creado antes al contenedor de Prometheus. Su nombre y el puerto.



Después le damos a “Save & Test”.

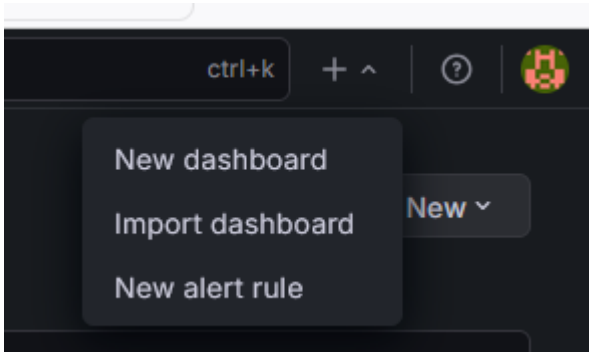


3. Creación del "Speed-Dashboard"

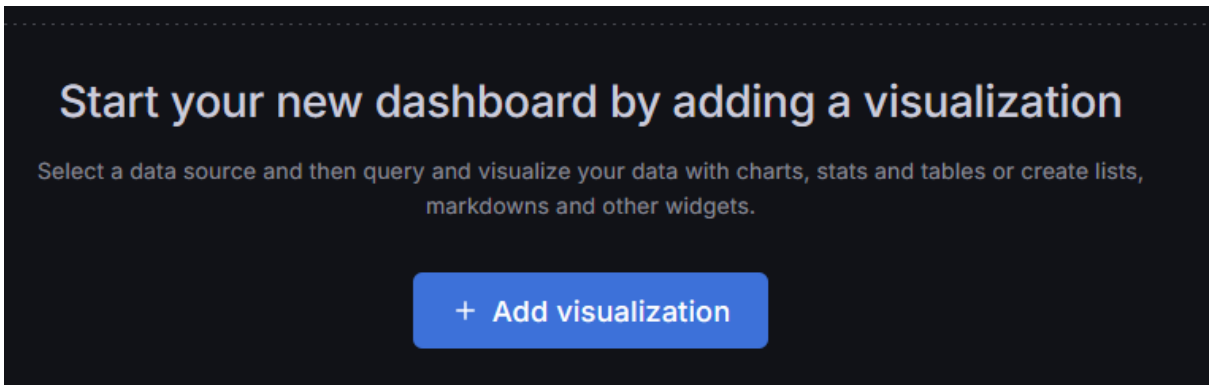
A continuación, crear un dashboard que nos dé información siguiente:

1. Muestra la memoria usada en los contenedores.
2. Cambia la unidad de medida a la correspondiente.
3. **Organización:** Ajusta el tamaño de los paneles, dales nombres descriptivos y guarda el dashboard como "Estado de Microservicios".

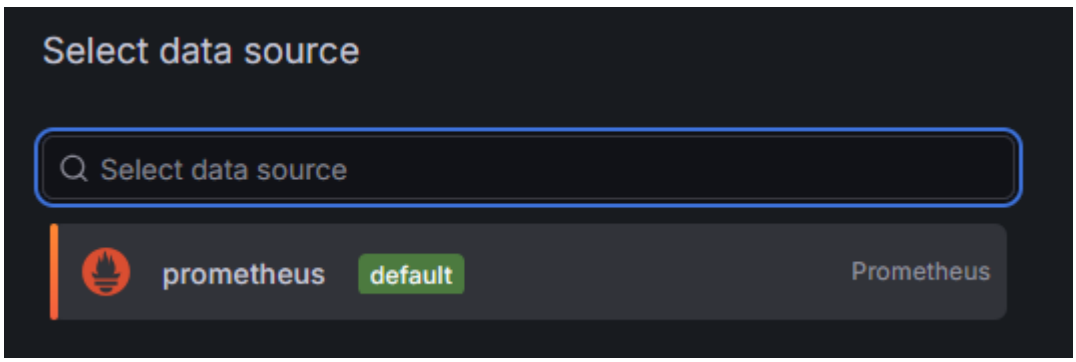
Esquina superior derecha:



Añadir visualización:

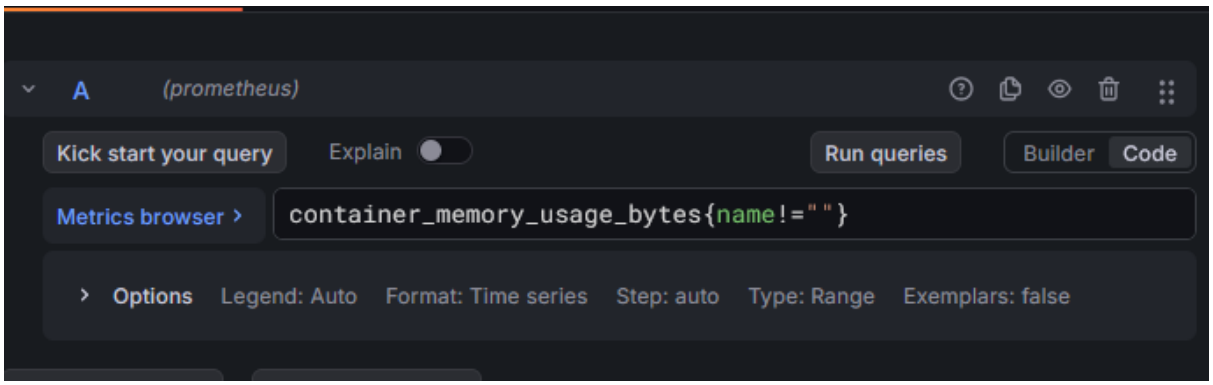


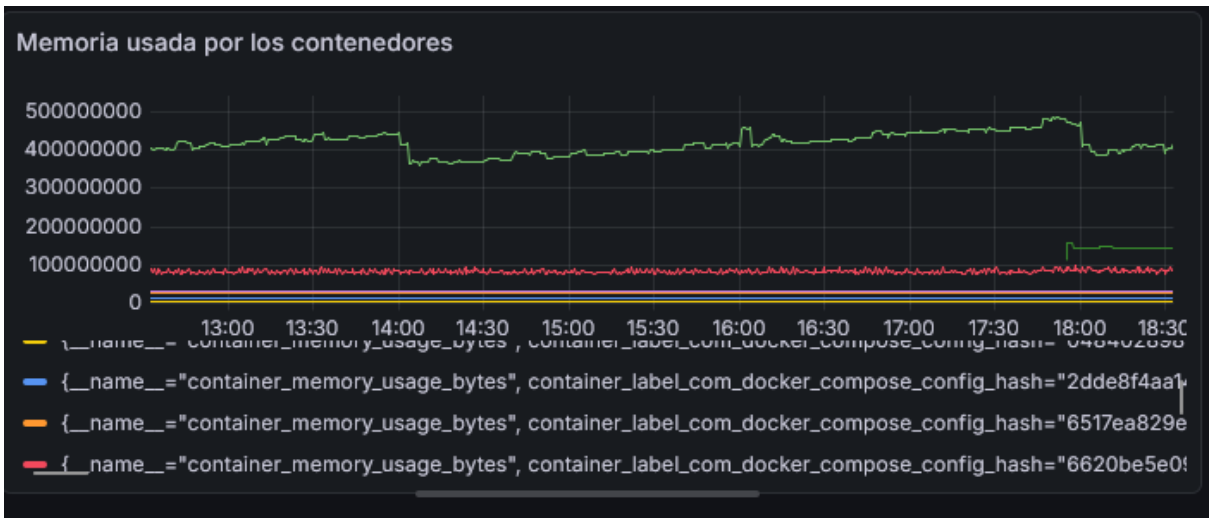
Seleccionamos a Prometheus:



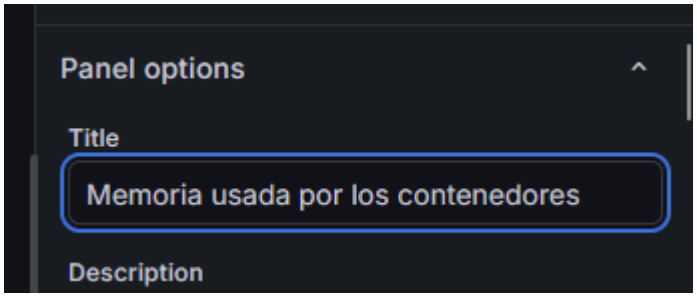
Una vez dentro, dale a “Code” y pon en “Metric browser”:

container_memory_usage_bytes{name!=""}

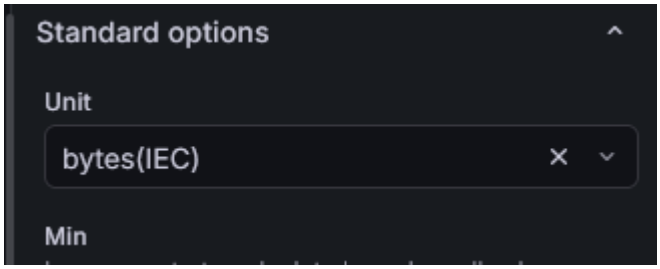


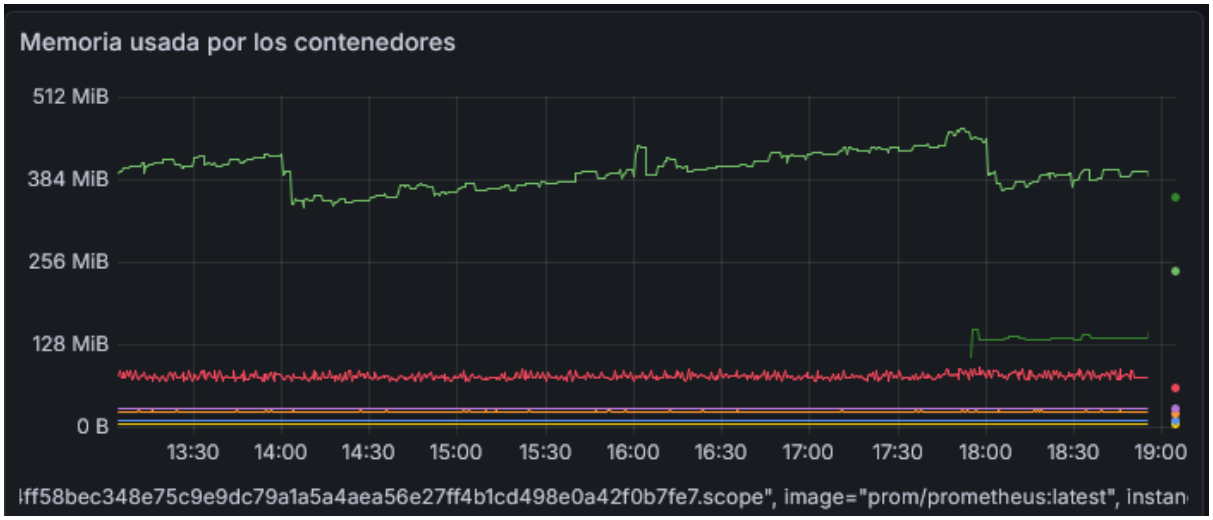


En “Panel Options” → Title:

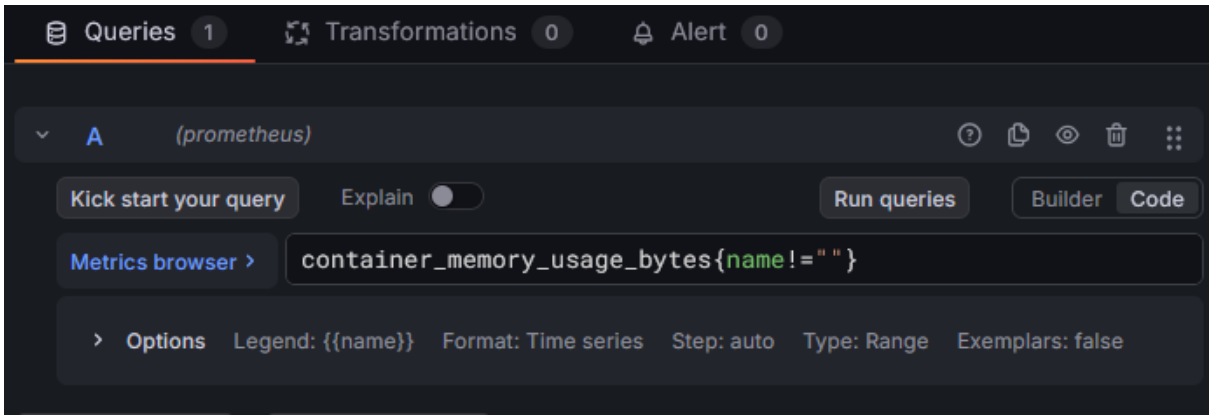


Nos vamos a “Standard options” → Data → bytes(IEC)

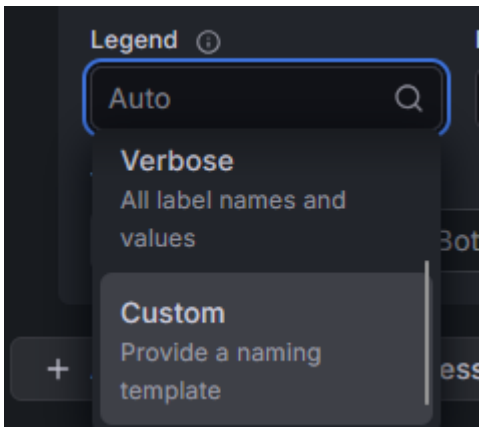




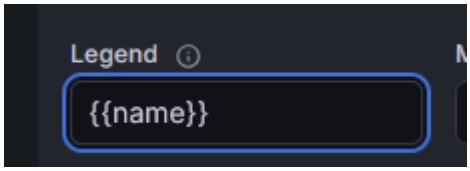
En la parte de “Queries”, le damos a “Legend”



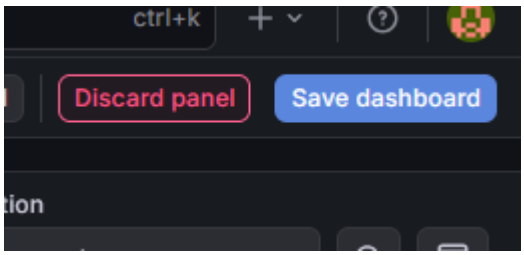
En Legend → Custom.



Y luego ponemos: {{name}}



Guardamos el Dashboard



Verificación:

- Muestra tu dashboard funcionando. Intenta realizar algunas peticiones al Frontend o al Backend y observa si hay alguna variación en las gráficas en tiempo real.

- **Peticiones al Frontend:**

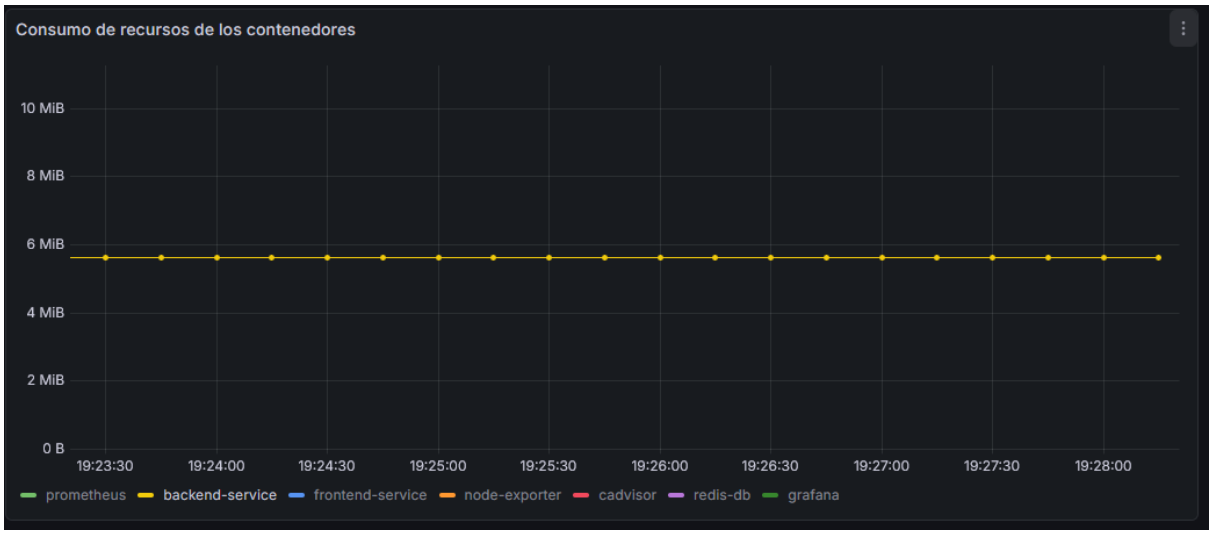
for i in {1..100}; do curl -s http://10.2.7.99:8080 > /dev/null; done

- **Peticiones al Backend:**

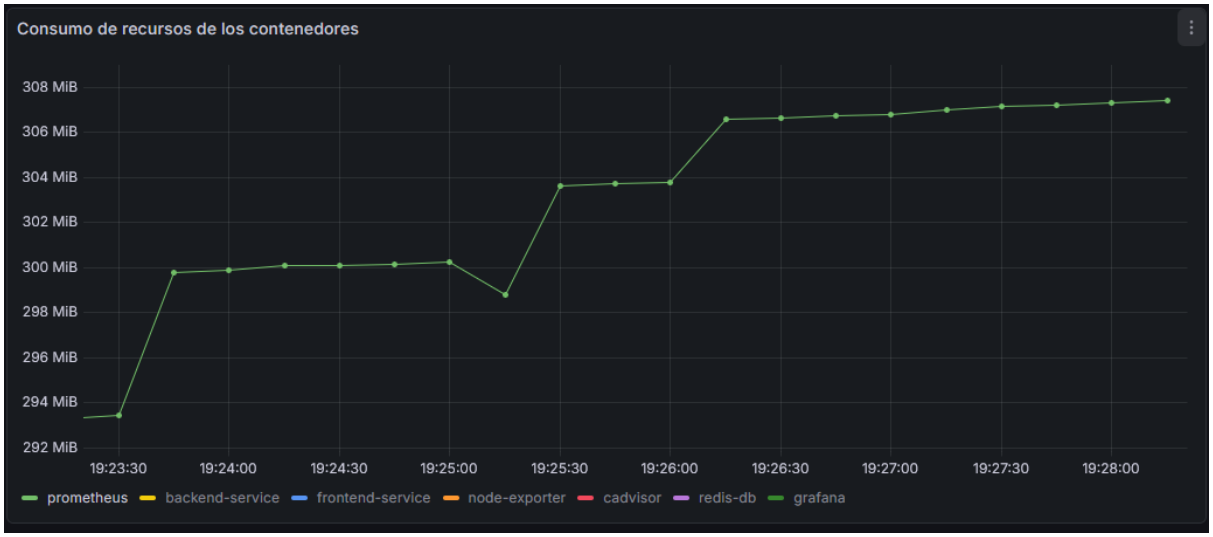
for i in {1..100}; do curl -s http://10.2.7.99:9090 > /dev/null; done

Apenas se nota porque no afecta mucho al consumo de RAM en NGINX

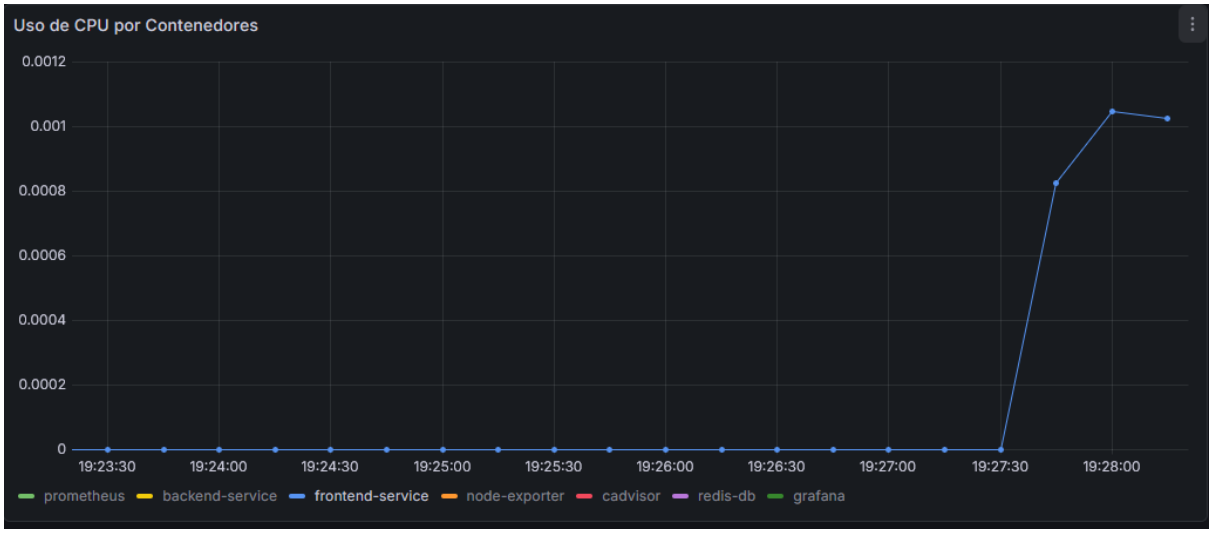
Y en el backend service

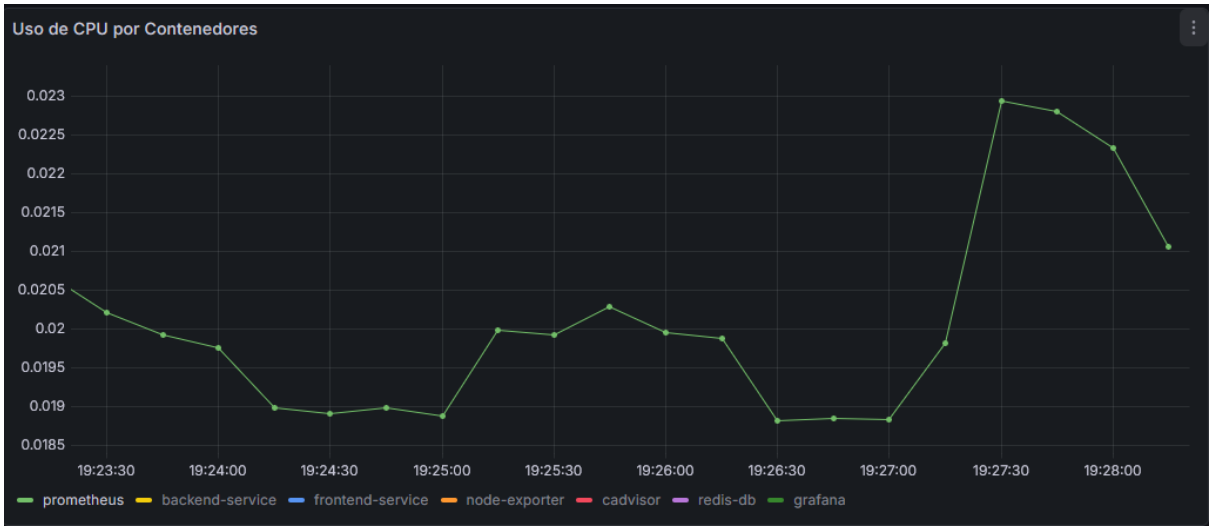


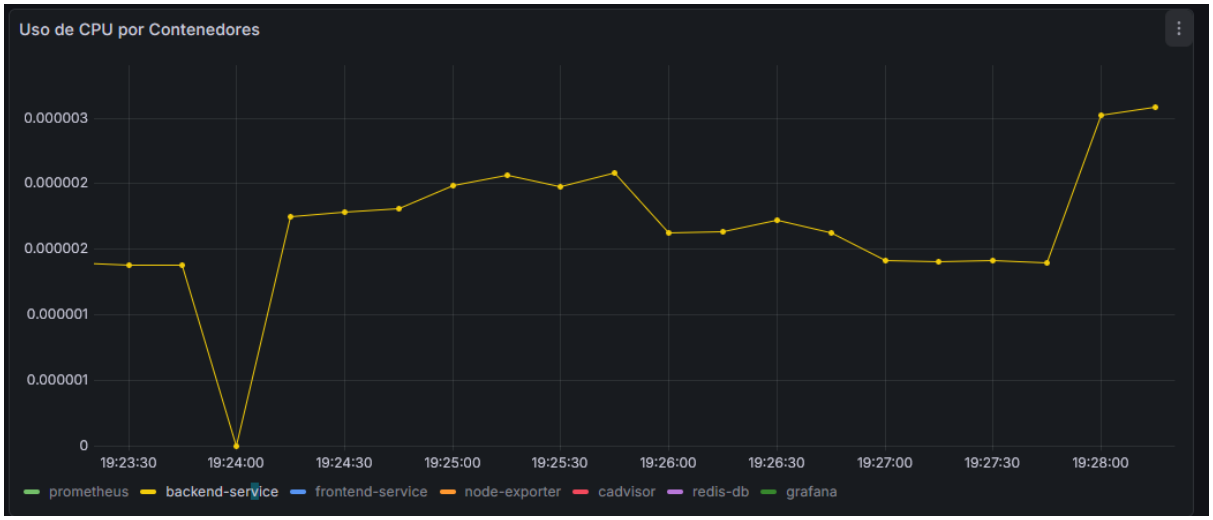
Pero si en el de Prometheus:



Pero si se nota una mayor variación si tenemos una gráfica que mide el consumo de CPU.







Para añadir el panel de CPU, poner esto en el Query:

rate(container_cpu_usage_seconds_total{name!=""}[1m])