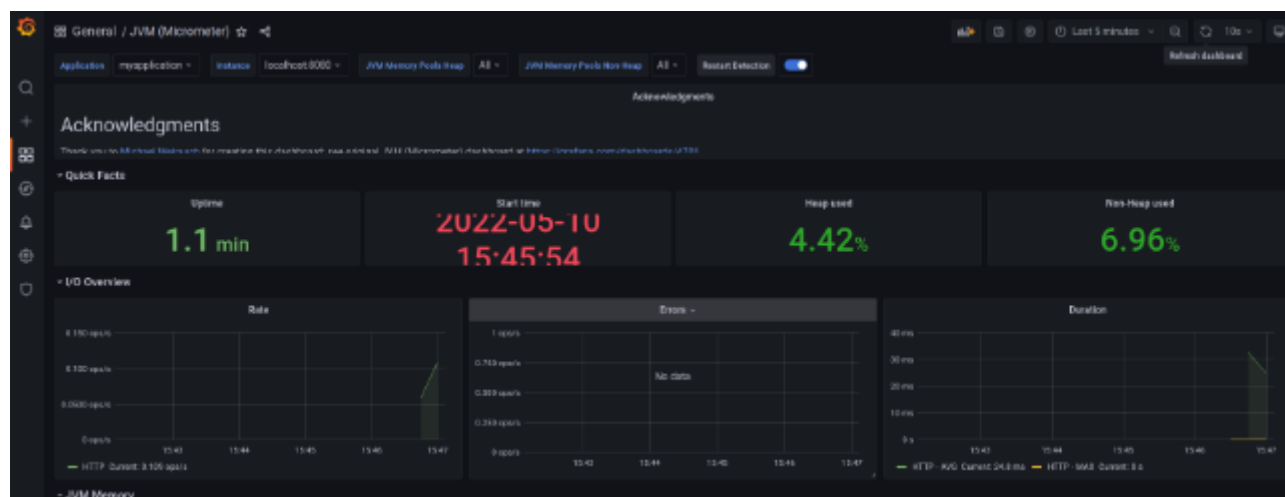


Sommaire

1. [Utilisation de Grafana](#)
 1. [Ajouter une datasource](#)
 2. [Créer un graphique / Ajouter une métrique](#)
 3. [Exporter / Importer des graphiques](#)
 4. [Paramétrer une alerte](#)
2. [Installation de Promtail](#)
3. [Configuration de Promtail](#)
4. [Récupérer des fichiers de logs](#)
5. [Créer et récupérer des logs de type syslogs d'un service](#)
6. [Afficher les logs dans Grafana](#)
7. [Filtrer les logs](#)
8. [Prometheus - Ajouter un endpoint](#)

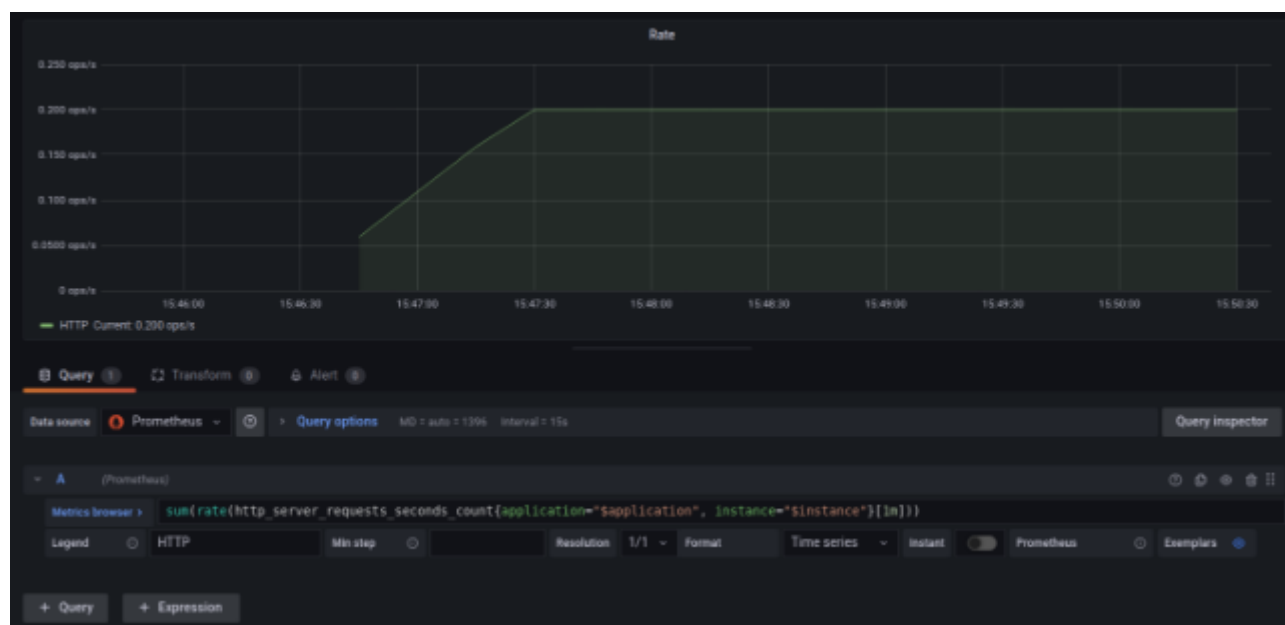
Utilisation de Grafana

Grafana permet l'affichage de plusieurs métriques récupérées par différentes source de données sous forme de tableau de bord personnalisable



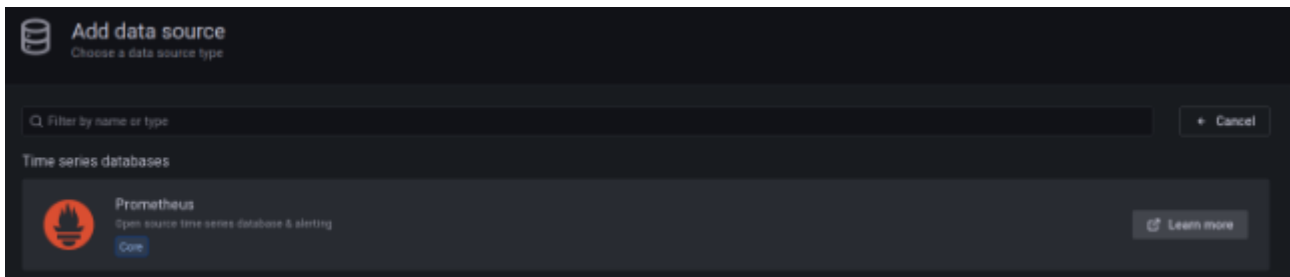
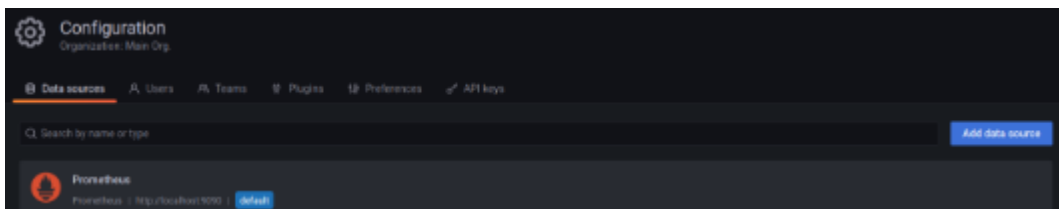
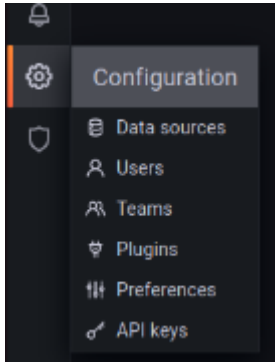
Queries

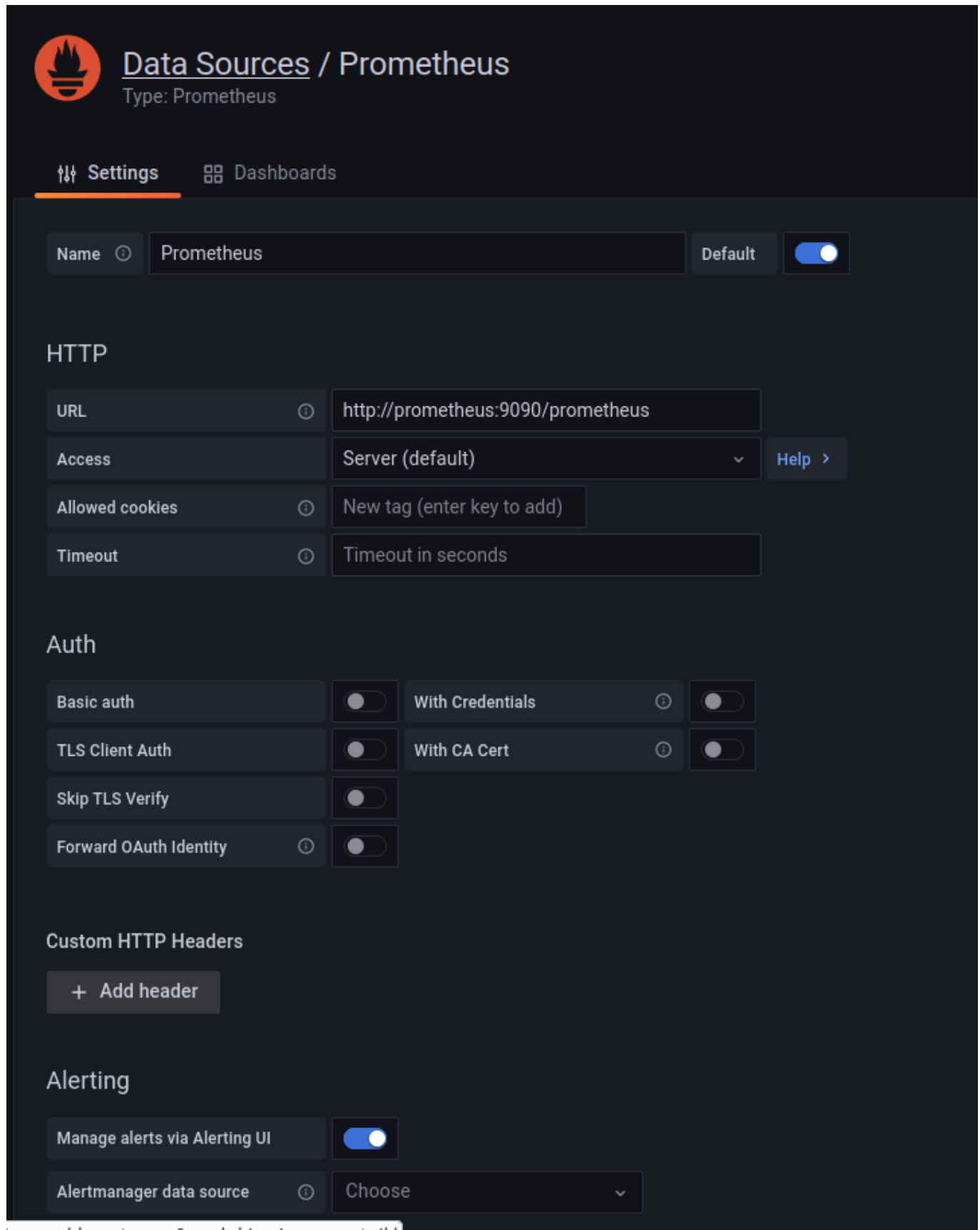
Grafana peut être utilisé pour effectuer des queries (requêtes) afin d'obtenir des données pour les afficher dans un graphique ou pour les consulter directement comme par exemple les fichiers de logs ou les résultats d'une requête MySQL.



Ajouter une datasource

Pour pouvoir afficher des données, Grafana aura besoin de localiser et de requêter la source de ces dernières, pour ce faire aller dans l'onglet Configuration – Datasource sur le bord gauche de l'écran → Add data source choisir la source de données et remplir les différents champs (exemple de configuration Prometheus ci-dessous)





The screenshot shows the 'Data Sources / Prometheus' configuration page. At the top, there's a header with the Prometheus logo and the title 'Data Sources / Prometheus' with a subtitle 'Type: Prometheus'. Below this is a navigation bar with 'Settings' (active) and 'Dashboards'. The main configuration area includes a 'Name' field set to 'Prometheus', a 'Default' toggle switch, and an 'HTTP' section with fields for 'URL' (http://prometheus:9090/prometheus), 'Access' (Server (default)), 'Allowed cookies' (New tag (enter key to add)), and 'Timeout' (Timeout in seconds). An 'Auth' section follows with toggle switches for 'Basic auth', 'TLS Client Auth', 'Skip TLS Verify', and 'Forward OAuth Identity', each with a 'With Credentials' or 'With CA Cert' option. Below this is a 'Custom HTTP Headers' section with an 'Add header' button. The 'Alerting' section at the bottom has a 'Manage alerts via Alerting UI' toggle switch and an 'Alertmanager data source' dropdown menu.

Data Sources / Prometheus
Type: Prometheus

Settings Dashboards

Name ⓘ Prometheus Default ☒

HTTP

URL ⓘ http://prometheus:9090/prometheus

Access Server (default) ▾ Help >

Allowed cookies ⓘ New tag (enter key to add)

Timeout ⓘ Timeout in seconds

Auth

Basic auth ☐ With Credentials ⓘ ☐

TLS Client Auth ☐ With CA Cert ⓘ ☐

Skip TLS Verify ☐

Forward OAuth Identity ⓘ ☐

Custom HTTP Headers


+ Add header

Alerting

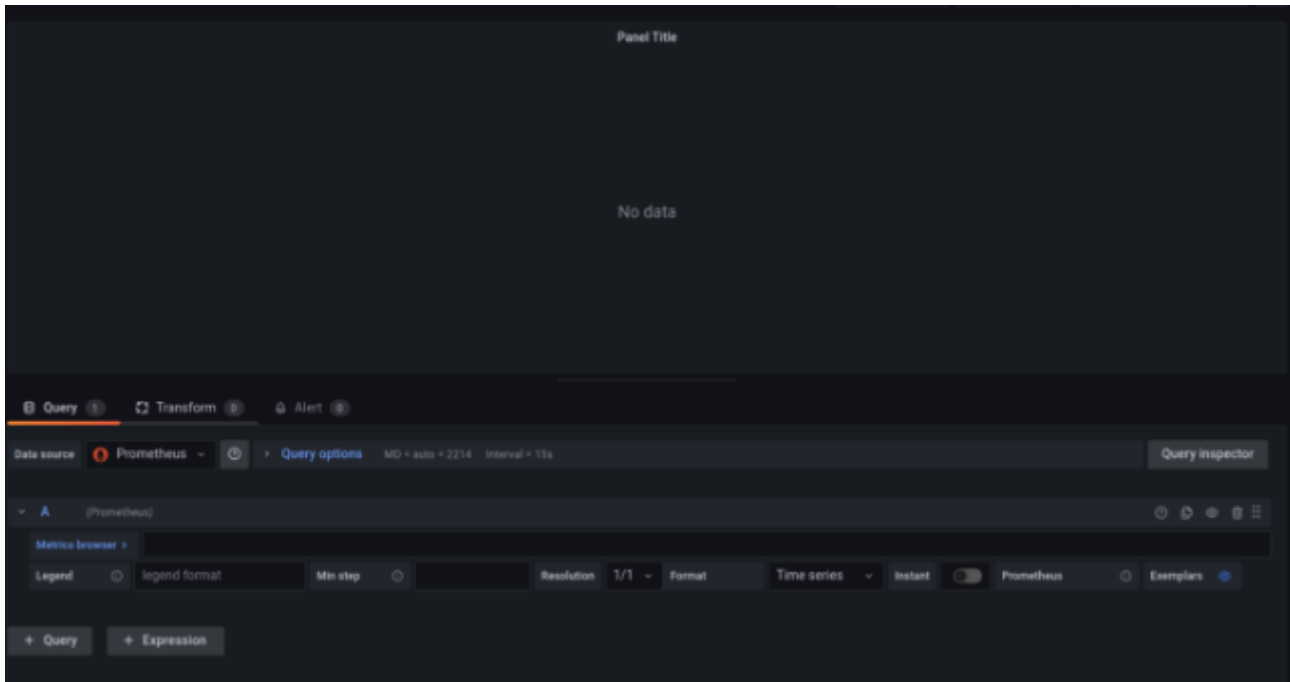
Manage alerts via Alerting UI ☒

Alertmanager data source ⓘ Choose ▾

Créer un graphique / Ajouter une métrique

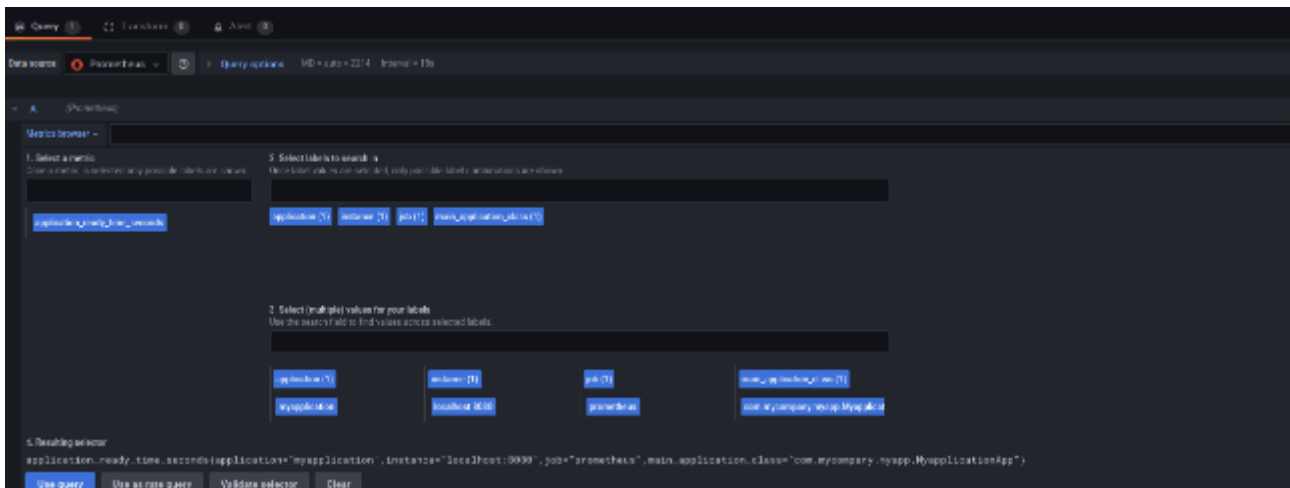
En haut à droite de l'écran, sélectionner Add Panel  → Add a New Panel dans la nouvelle fenêtre qui s'ouvre

Une fois sur l'interface de gestion des graphiques effectuez les actions suivantes :



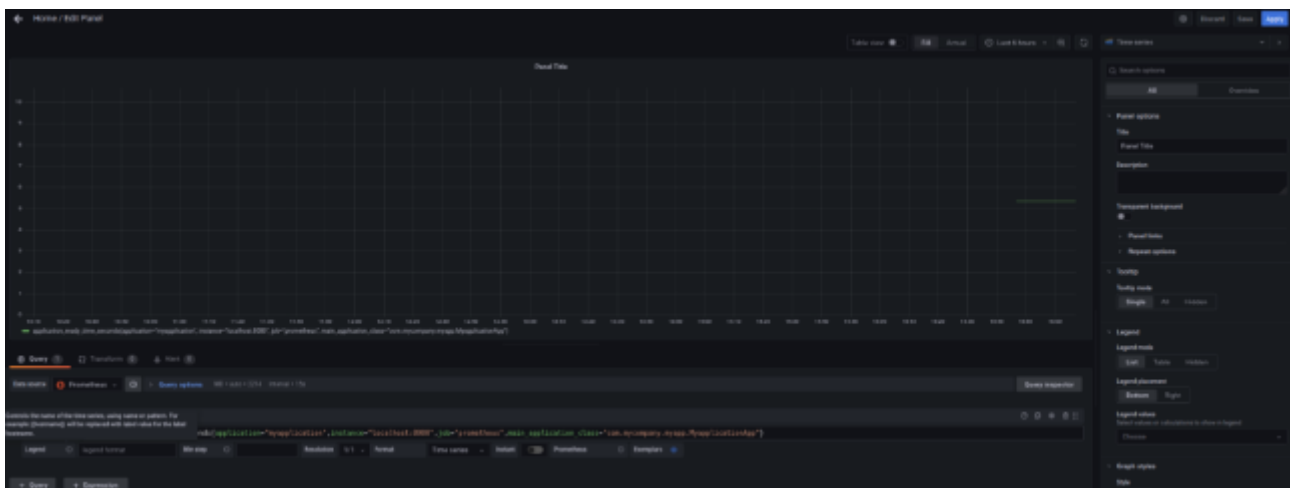
1 - Sélectionnez dans Datasource Prometheus

2 - Cliquez sur Metrics Browser



3 - Sélectionnez les métriques que vous souhaitez afficher

4 - Cliquez sur Use Query



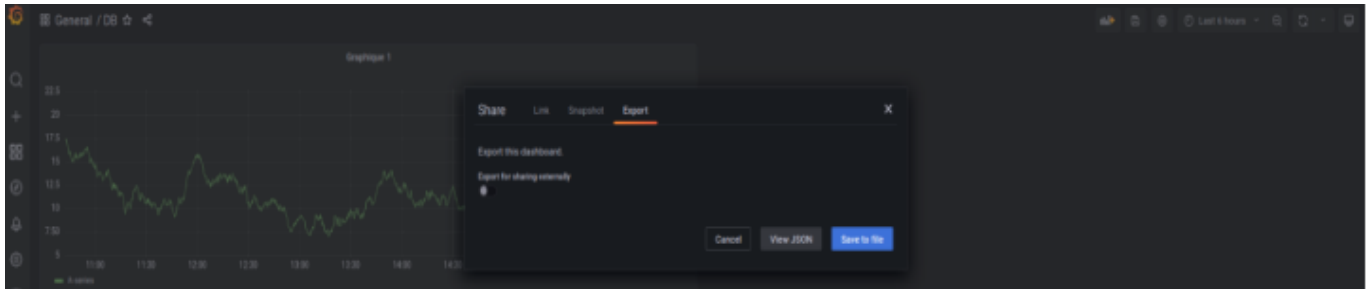
5 - Sauvegardez votre métrique avec le bouton Save en haut à droite

Exporter / Importer des graphiques

Exporter un ensemble de graphiques (dashboard).



Dans l'onglet dashboard à gauche de l'interface



1 - Cliquez sur le Dashboard que vous souhaitez exporter



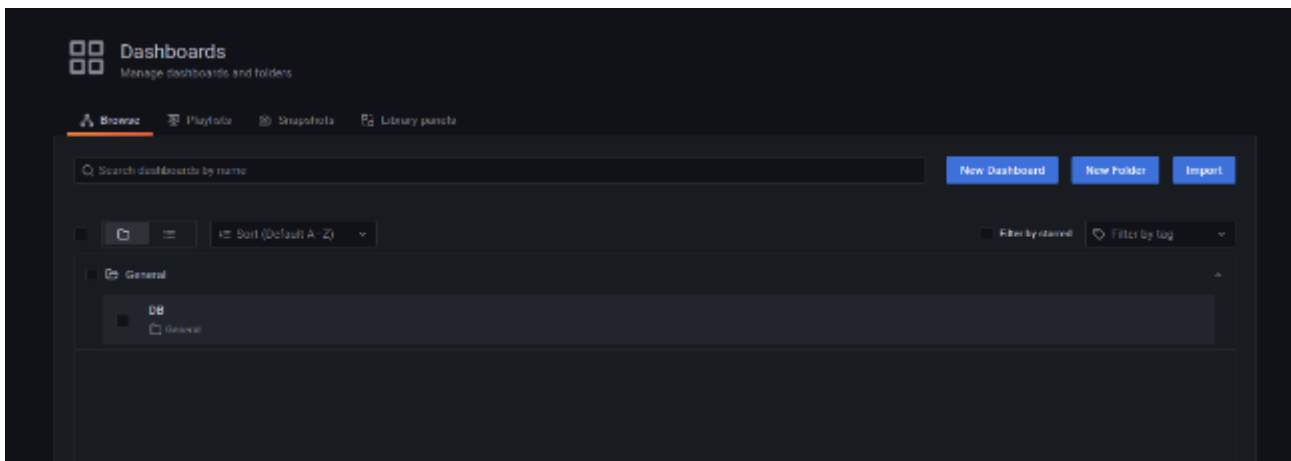
2 - Cliquez sur l'icone

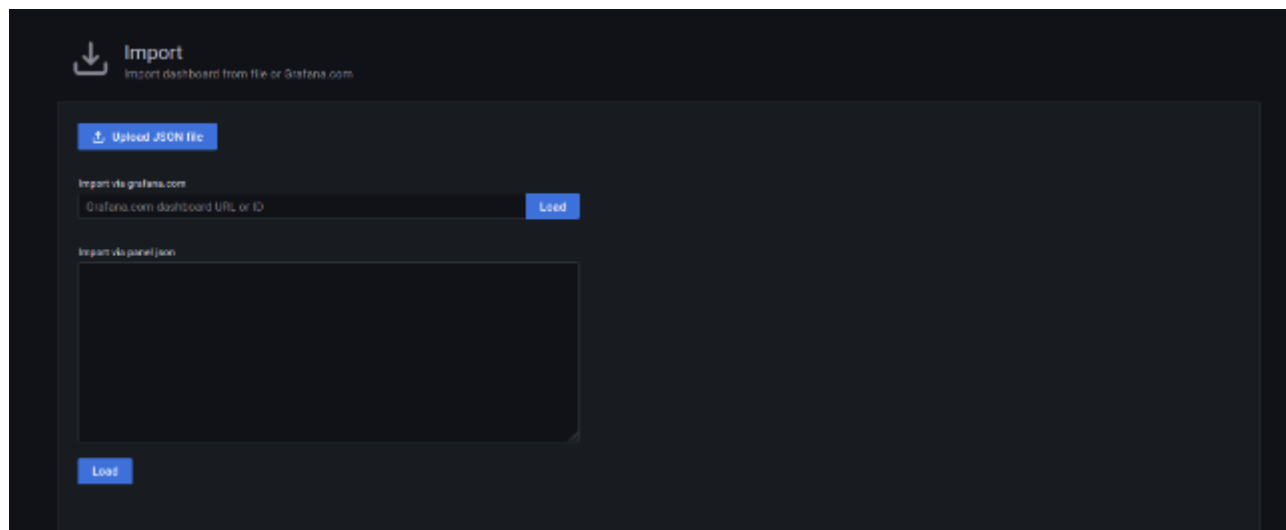
3 - Dans le menu Export, cliquez sur Save to file

Importer un ensemble de graphiques (dashboard).



Dans l'onglet dashboard à gauche de l'interface sélectionner Browse





- 1 - Cliquez sur le bouton Import
- 2 - Cliquez sur le bouton Upload JSON file
- 3 - Cliquez sur le bouton Load

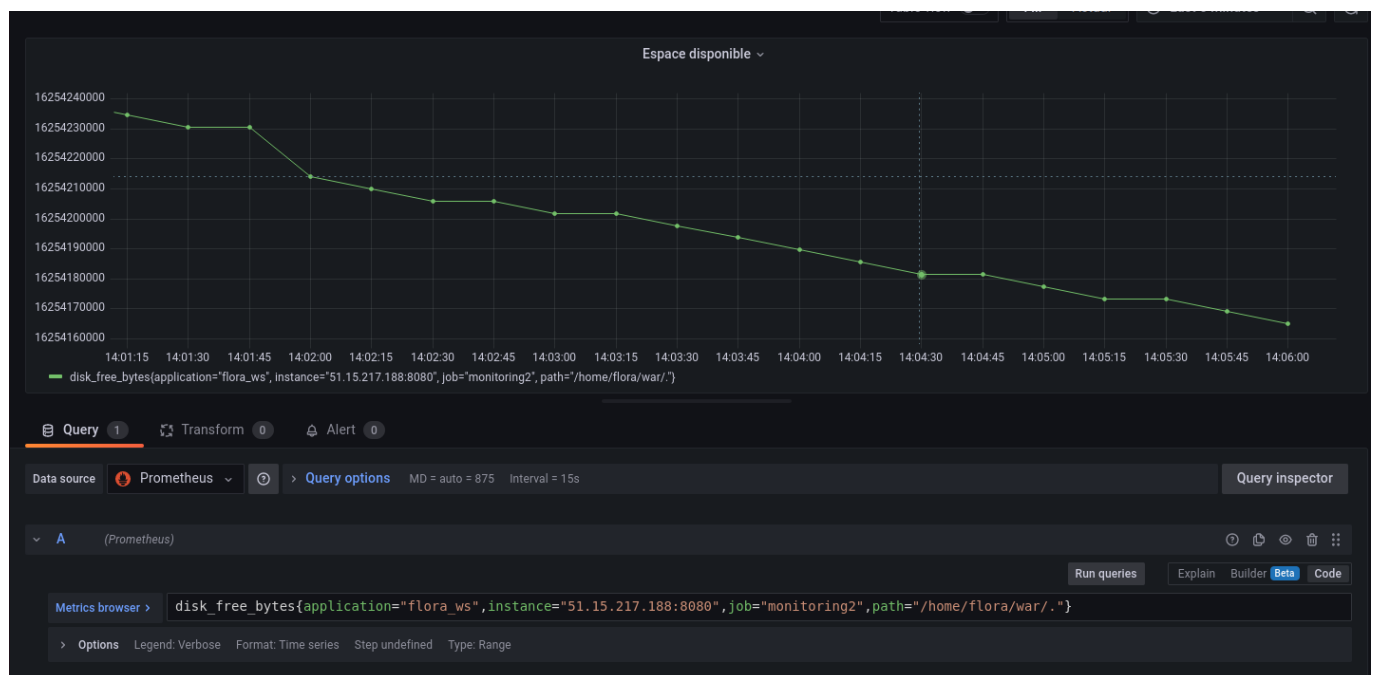
Paramétrer une alerte

Les alertes nécessitent le paramétrage du SMTP au préalable, pour ce faire :

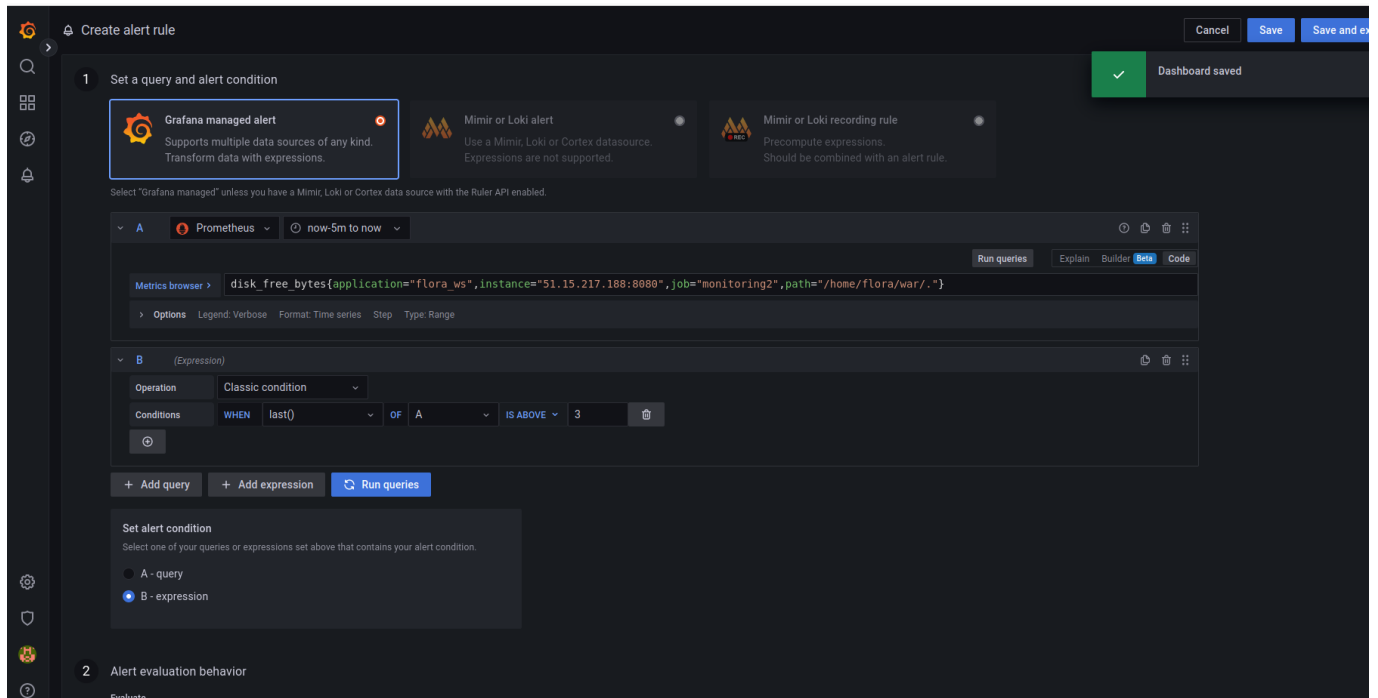
```
docker exec -ti monitoring-grafana-1 /bin/bash
```

Dans le `/etc/grafana/grafana.ini` du serveur, dans la partie SMTP/Emailing modifiez les lignes appropriées à votre serveur SMTP.

Choisissez la métrique sur laquelle vous voulez paramétrer une alerte, pour cet exemple, nous allons prendre une métrique qui mesure l'espace disque disponible sur une machine et qui va nous alerter lorsque celui ci descend sous un certain montant.



Dans l'onglet Alert, cliquez sur le bouton **Create a alerte rule from this panel**



Nous arrivons à la fenêtre de paramétrage d'alertes de Grafana, on peut apercevoir deux menu A & B , avec respectivement notre métrique et la condition d'alerte, plusieurs options de conditions comme des expressions mathématiques ou par échantillonnage sont disponibles mais pour rester dans une approche simple, voici quelques conditions d'alertes basiques :

AVG : Indique une valeur moyenne

MIN : Indique une valeur minimale

MAX : Indique une valeur maximale

SUM : Indique une somme de valeurs sous une plage de temps donnée

COUNT : Indique un nombre d'entrées sous une plage de temps donnée

LAST : Indique un nombre d'entrées sous une plage de temps donnée

MEDIAN : Prend la valeur médiane de la métrique

Après avoir paramétrer vos conditions d'alertes, **cliquez sur le bouton Run Queries pour visualiser graphiquement l'alerte et voir si elle à bien été pris en compte**, dans notre cas notre expression B sera WHEN max() OF A IS BELOW X BYTES, que l'on pourrait traduire par : Si la métrique descend en dessous de X BYTES, surveille la métrique.

The screenshot shows the 'Alert evaluation behavior' configuration page in Grafana. It is divided into two main sections: '2 Alert evaluation behavior' and '3 Add details for your alert'.

Section 2: Alert evaluation behavior

- Evaluate**: Evaluation interval applies to every rule within a group. It can overwrite the interval of an existing alert rule.
- Evaluate every**: A dropdown menu showing '1m' (highlighted with a red box) and '5m'.
- Configure no data and error handling**: A link to configure no data and error handling.
- Preview alerts**: A button to preview alerts.

Section 3: Add details for your alert

Write a summary and add labels to help you better manage your alerts

Rule name: A text input field containing 'Espace disponible'.

Folder: A dropdown menu with 'Choose' selected. A note states: 'Rules within the same group are evaluated after the same time interval.'

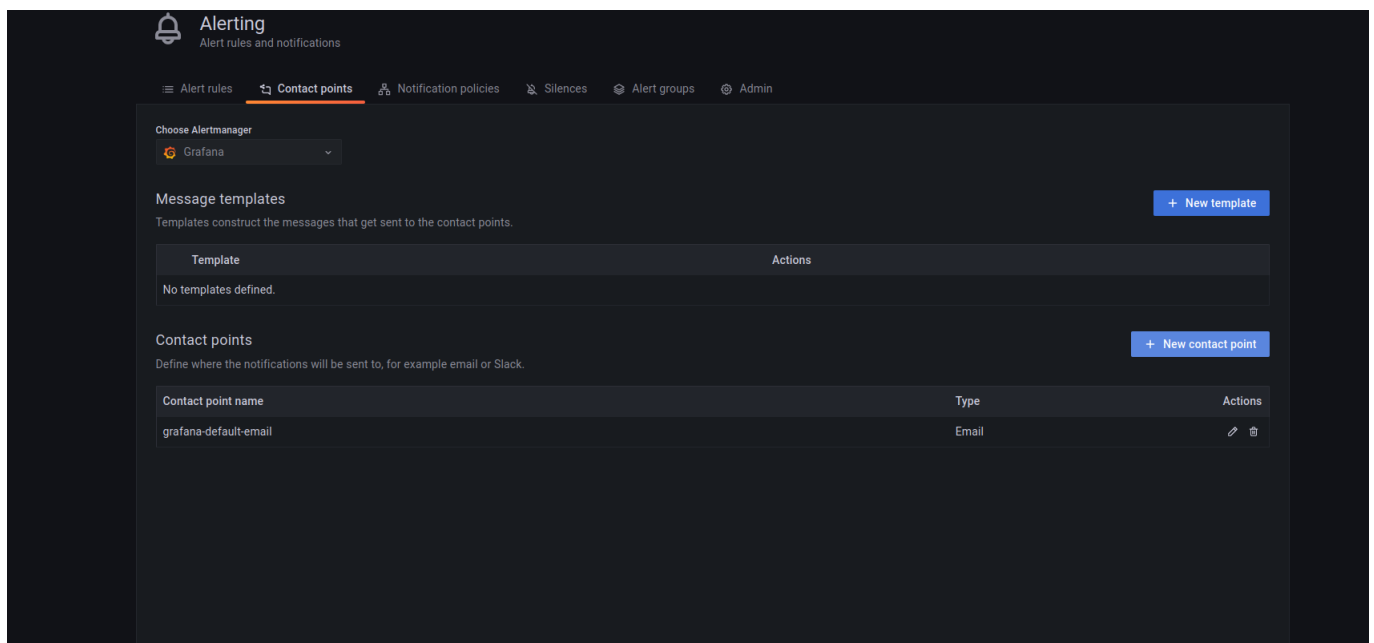
Summary and annotations

- Dashboard UID**: A dropdown menu with 'xENn2grnk' selected.
- Panel ID**: A dropdown menu with '2' selected.
- Add info**: A button to add information.

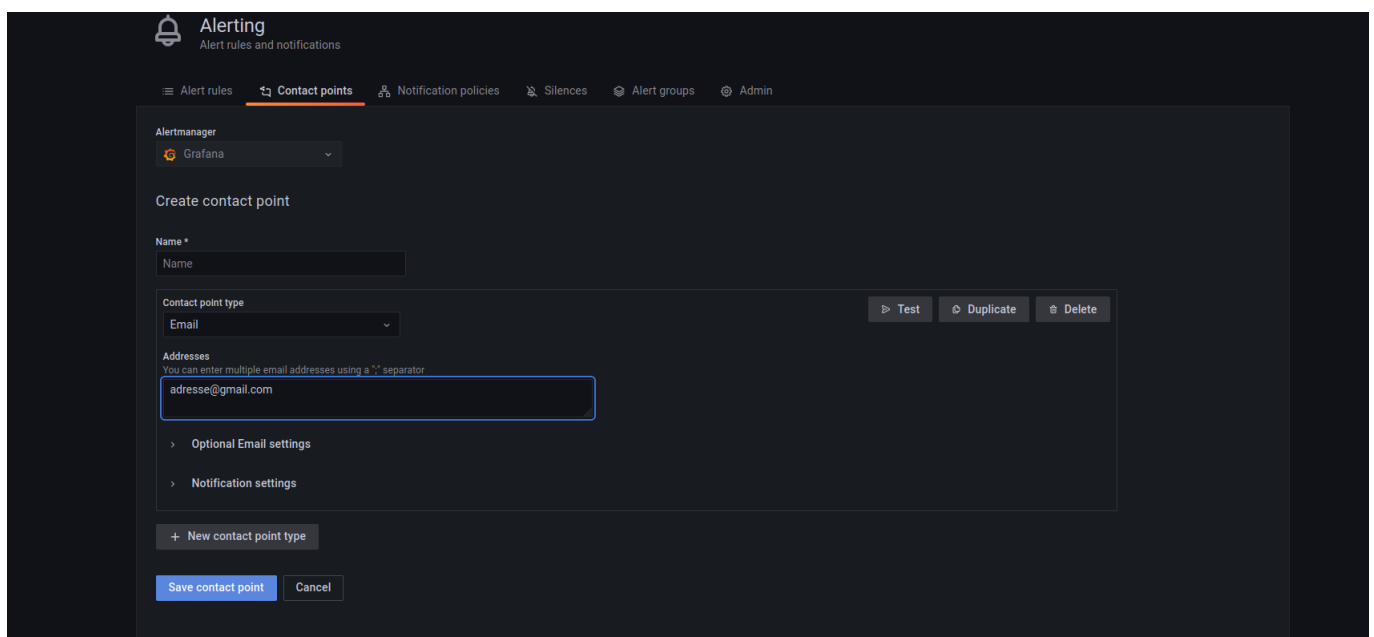
La deuxième partie est le paramétrage de la détection de l'alerte, **dans la partie 2 Alert evaluation behavior veuillez indiquer comment la détection va opérer quand l'expression aura été déclenchée**, ici, nous allons paramétrer l'alerte pour que lorsque l'expression B se déclenche, Grafana va surveiller la métrique toutes les 1 minutes pendant 5 minutes et déclencher l'alerte si celle-ci déclenche toujours l'expression

Avec notre expression, lorsque la métrique passe en dessous de X BYTES, Grafana va surveiller la métrique pendant 5 minutes et vérifier toutes les 1 minutes si l'expression est déclenchée et si c'est le cas, nous alertera.

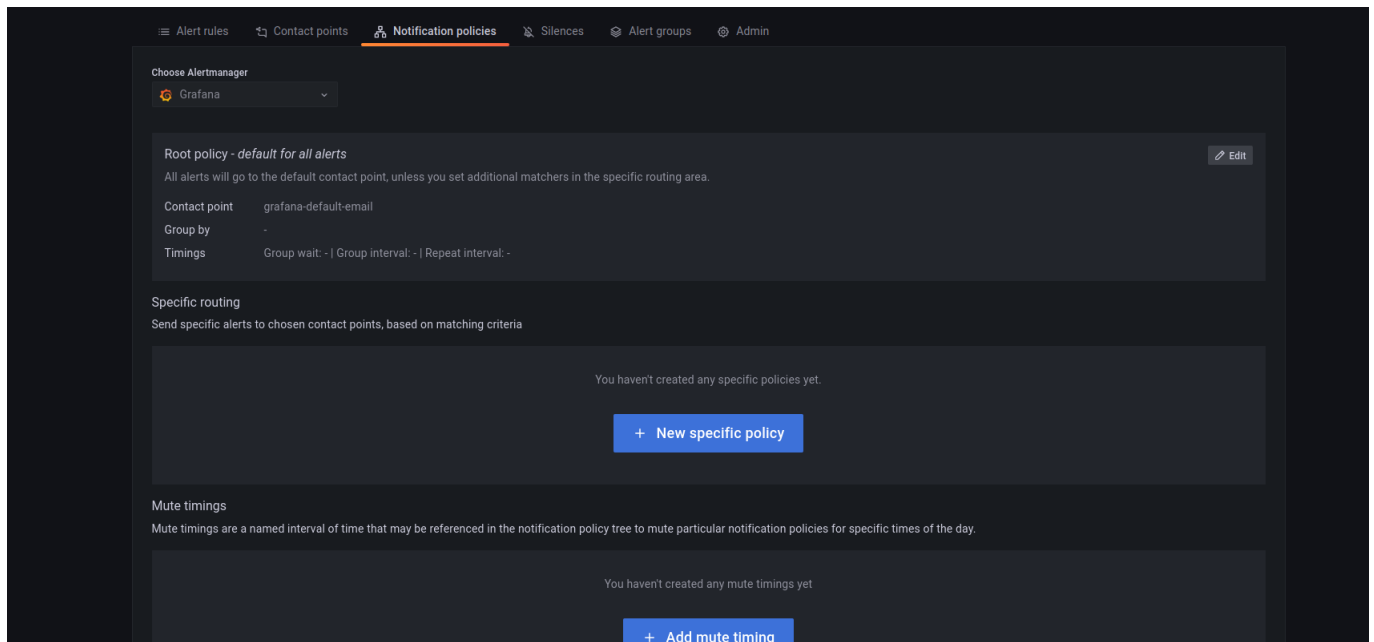
Les parties 3 et 4 servent à ajouter des détails et organiser les alertes, **pour créer une alerte il faudra obligatoirement sélectionner un fichier dans lequel enregistrer l'alerte et lui assigner un groupe, cliquez sur Save and Exit une fois que vous avez fini.**



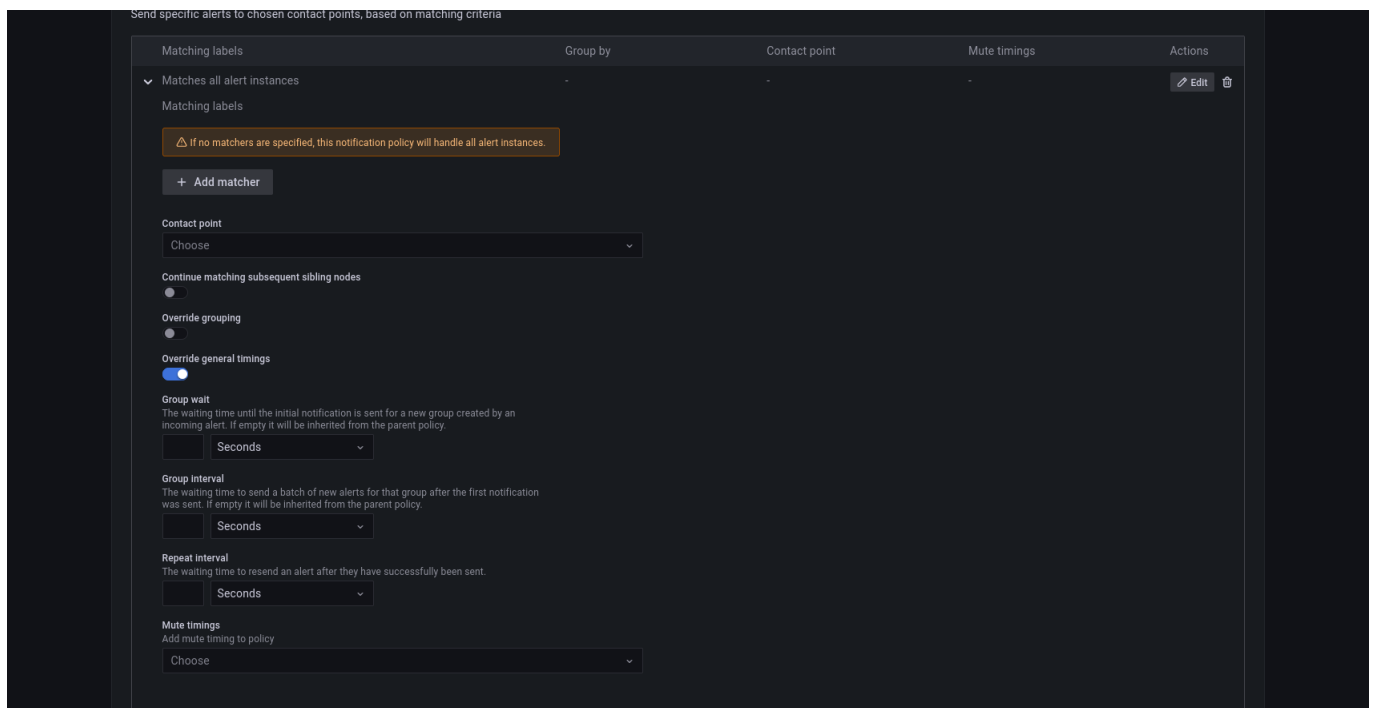
Maintenant que nous avons paramétré notre alerte, il nous faut lui assigner un point de contact sur lequel Grafana enverra un message d'alerte qui peut prendre plusieurs formes (Mail, Bot Slack, etc...), dans notre cas nous enverrons un mail : **Dans le menu Alerte (icône de cloche) dans la barre à gauche de l'interface, choisissez Contact Points et cliquez sur New contact Point**



Nommez votre point de contact, choisissez son type, et les paramètres nécessaires (Dans notre cas, nous choisirons Email et nous indiquerons une adresse de réception) et cliquez sur Save contact point

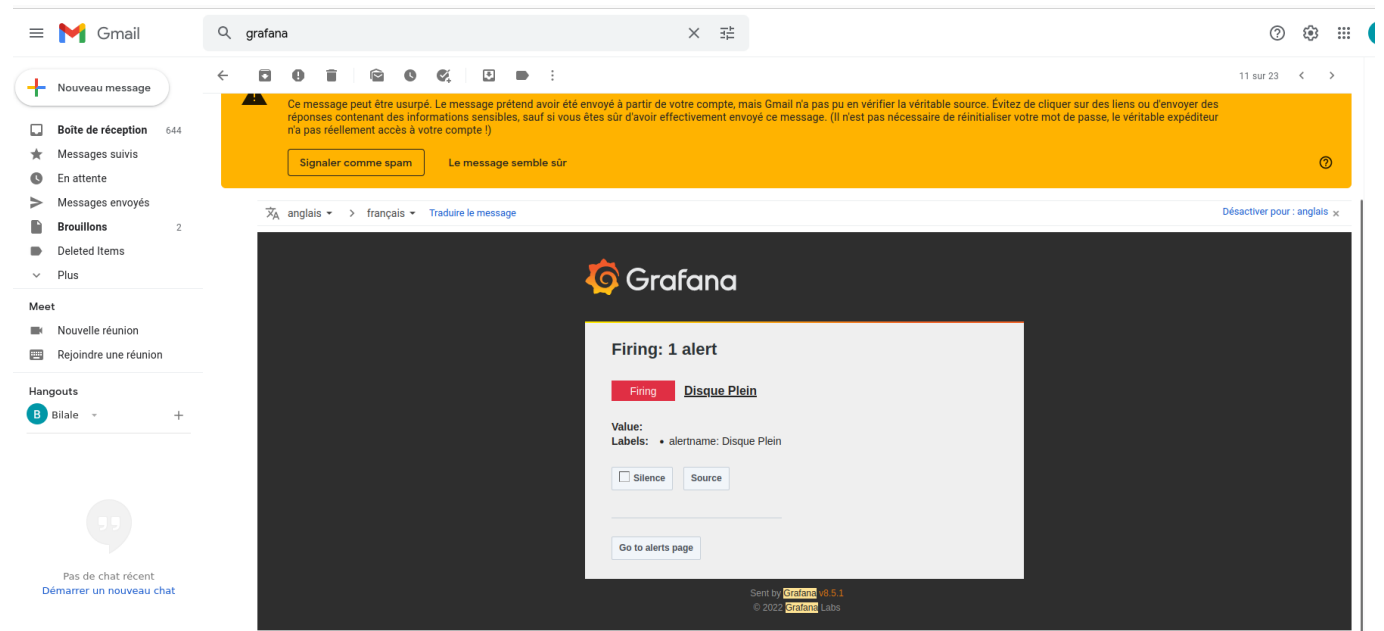


Maintenant il va falloir paramétrer les réglages de notification, **dans le menu Notification Policies cliquez sur le bouton New Specific Policy**



Sur cette fenêtre, **paramétrez votre Contact Point précédemment crée, et activez Override general timings, paramétrez la fréquence à laquelle l'alerte vous sera notifié**, dans notre cas, nous allons choisir de mettre Group Wait à 5 secondes et le reste à 0 pour n'avoir qu'un seul mail d'alerte.

Votre alerte est maintenant paramétrée, lorsqu'elle sera déclenchée voici un exemple de mail que vous pourrez recevoir :



Envoyer et afficher des logs avec Promtail / Loki / Grafana

Promtail

Promtail est un agent à déployer sur la machine dont vous souhaitez récupérer les logs, nous allons l'installer via paquet avec :

Installation de Promtail

```
curl -s https://api.github.com/repos/grafana/loki/releases/latest | grep
browser_download_url | cut -d '"' -f 4 | grep promtail-linux-amd64.zip | wget -i
-
unzip promtail-linux-amd64.zip
sudo mv promtail-linux-amd64 /usr/local/bin/promtail
promtail --version
sudo nano /etc/promtail-local-config.yaml
```

Configuration de Promtail

Note : La configuration de cette procédure est basée sur du local, pensez à l'adapter à votre configuration

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /data/loki/positions.yaml

clients:
  - url: http://localhost:3100/loki/api/v1/push

scrape_configs:
  - job_name: system
    static_configs:
      - targets:
          - localhost
    labels:
      job: varlogs
      __path__: /var/log/*log
```

Après avoir ajouter le fichier de configuration :

```
sudo tee /etc/systemd/system/promtail.service<<EOF

[Unit]
Description=Promtail service
After=network.target

[Service]
Type=simple
User=root
ExecStart=/usr/local/bin/promtail -config.file /etc/promtail-local-config.yaml

[Install]
WantedBy=multi-user.target

EOF

sudo systemctl daemon-reload
sudo systemctl enable promtail.service
sudo systemctl start promtail.service
```

Promtail envoi maintenant le contenu du dossier /var/log à Loki que l'on pourra afficher avec Grafana par la suite.

Récupérer des fichiers de logs

Après avoir installé Promtail, modifiez le `/etc/promtail-config.yaml` et veillez à remplacer la configuration en fonction de la votre (IP, Authentification si présente, chemin de fichiers)

Prenons pour notre exemple, de récupérer les fichiers logs de NGINX, situés dans `/var/log/nginx` :

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

#Si le dossier n'existe pas, le créer manuellement ou remplacer data par root
positions:
  filename: /data/loki/positions.yaml

clients:
  - url: https://adresseclienteloki/api/v1/push
#Si une authentification est nécessaire pour accéder à l'url, la rajouter :
  basic_auth:
    username: USERNAME
    password: PASSWORD

#Pour ajouter une instance qui va récupérer des logs nous allons ajouter une
configuration avec :
scrape_configs:
#Le job_name permet d'attribuer un nom à la tâche de récupération de configuration
afin de mieux les différencier dans le fichier yaml
  - job_name: Fichiers de logs
    static_configs:
#La target est la machine hôte sur laquelle on veut récupérer les fichiers
      - targets:
          - localhost
#Le label est le nom de la tâche de récupération de fichiers (soit ici le job qui
récupère /var/log/*.log) lorsqu'elle va apparaître dans Grafana
    labels:
      job: Logs de NGINX
      __path__: /var/log/nginx/*log
```

```
sudo systemctl restart promtail.service
```


Créer et récupérer des logs de type syslogs d'un service

Pour récupérer les logs de systemd ajoutez dans le /etc/promtail-config.yaml

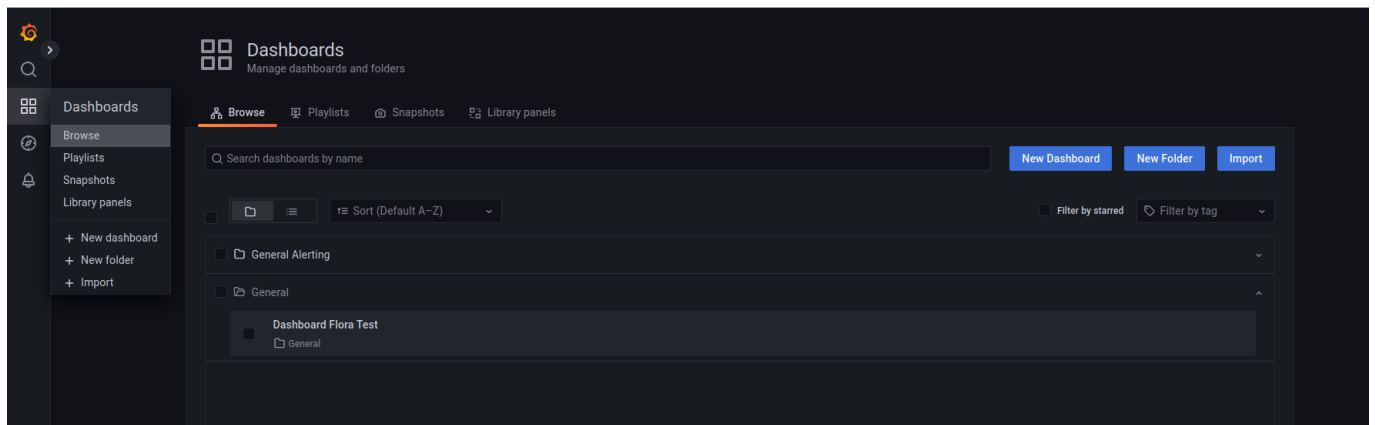
```
- job_name: journal
  journal:
    json: false
#La variable max_age permet d'afficher seulement les entrées dans les journaux qui
#datent d'il y'a 12 heures
    max_age: 12h
    path: /var/log/journal
    labels:
      job: systemd-journal
  relabel_configs:
    - source_labels: ['sudo apt install gnome-tweaks gnome-tweak-toolcmd']
```

```
sudo systemctl restart promtail.service
```

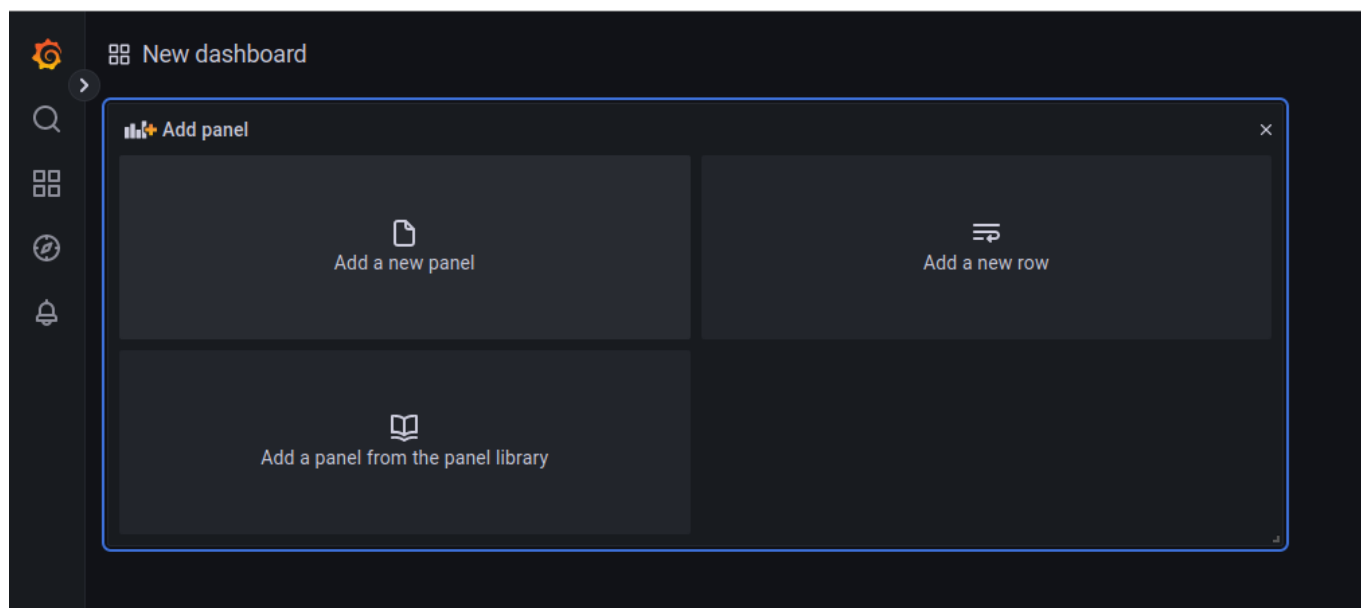
Afficher les logs dans Grafana

Maintenant que nous avons récupérés nos fichiers, nous allons les afficher avec Grafana Après avoir installé Loki et Grafana et configurer les Datasources, accédez à l'interface et créez une métrique :

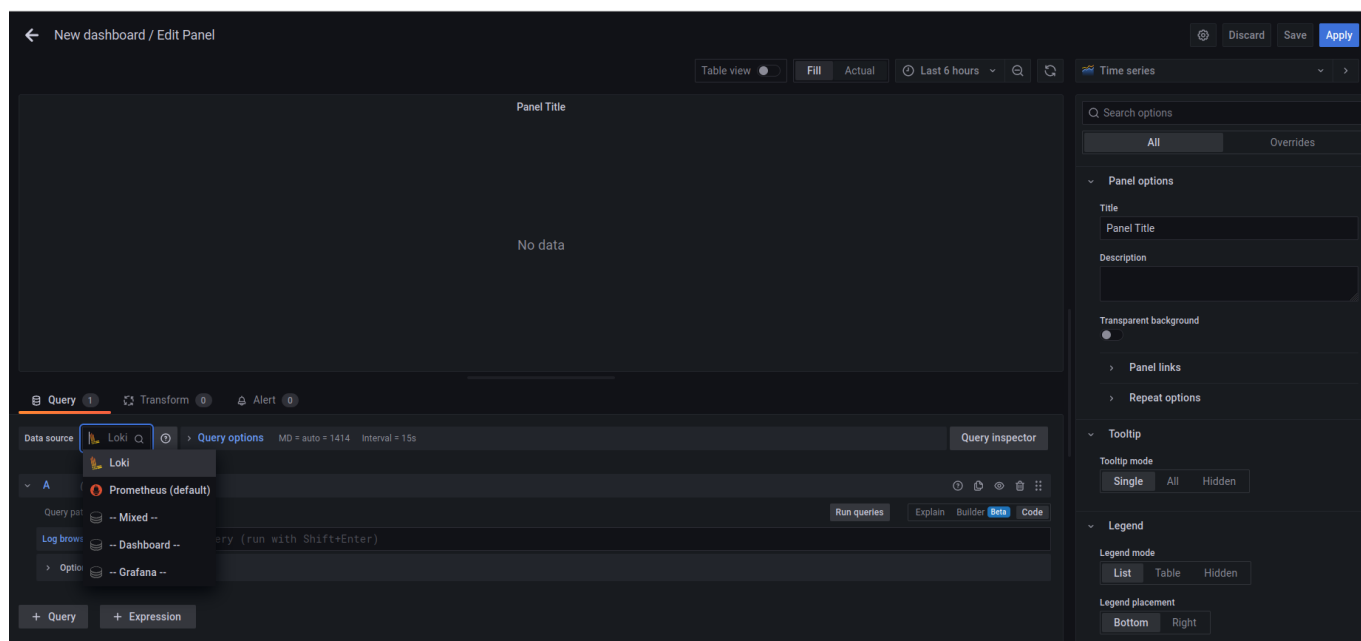
1 - Dans l'onglet Dashboard à gauche de l'écran, Cliquez sur Browse et New Dashboard dans la nouvelle fenêtre



2 - Cliquez sur Add A New Panel



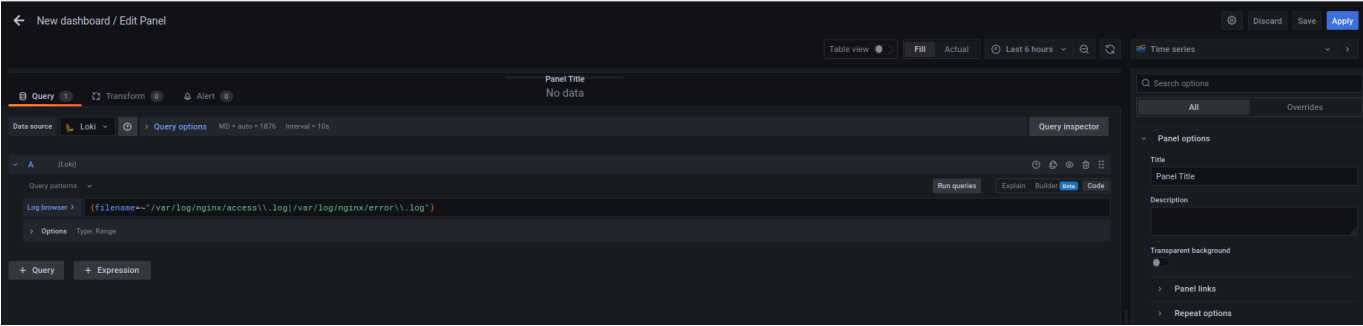
3 - Choisissez Loki comme Datasource



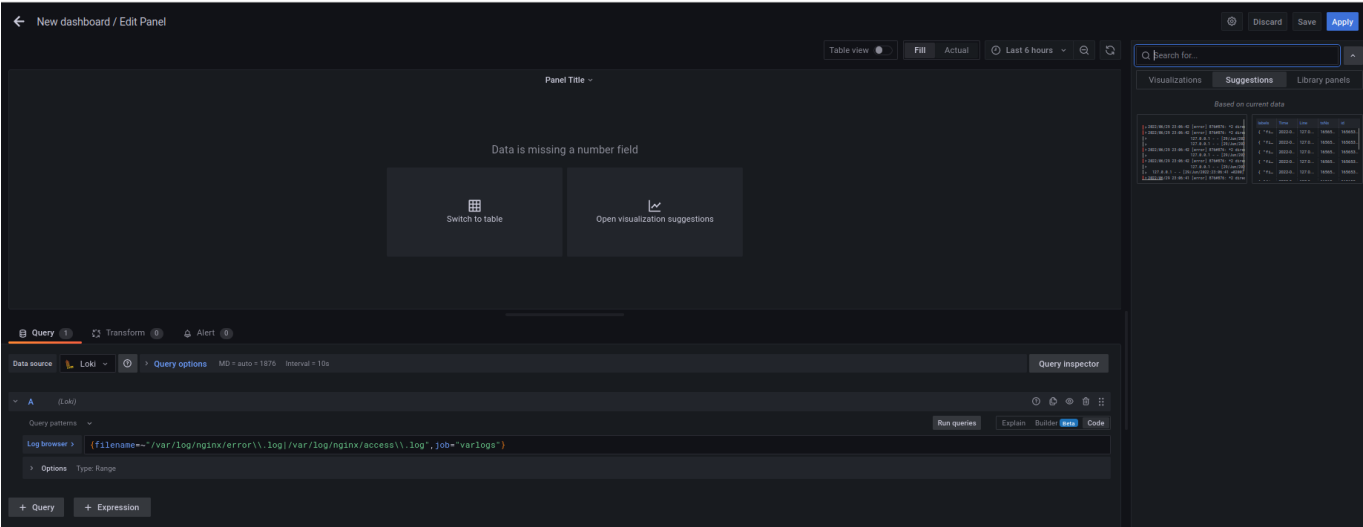
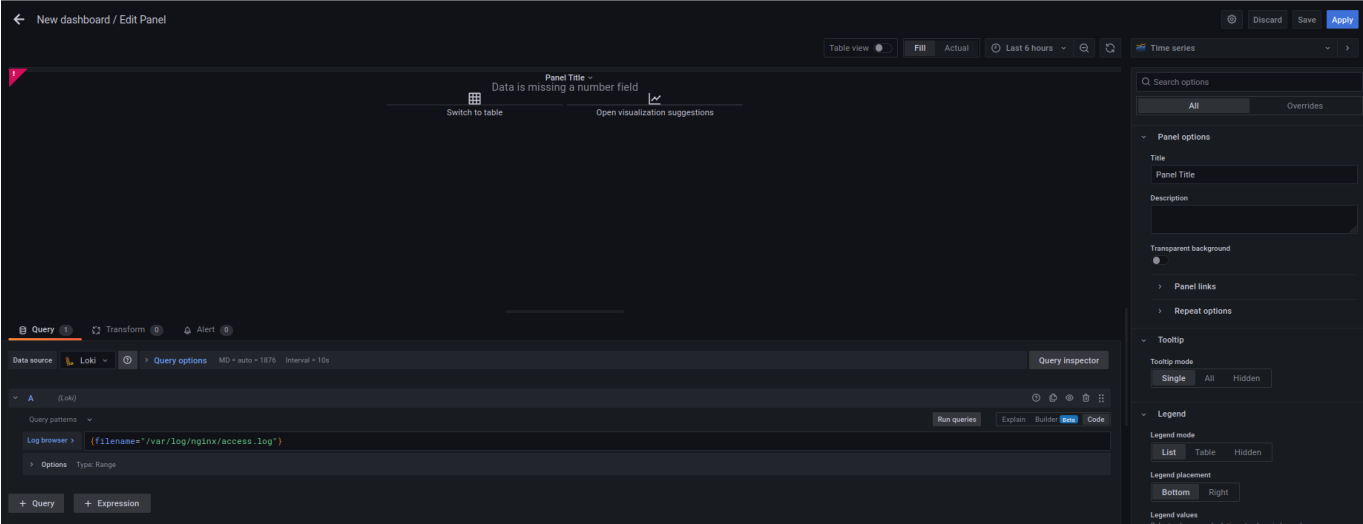
Nos fichiers de logs ont bien été transmis via Promtail, à noter que le labels / job mis précédemment sert à les différencier et les regrouper (Logs de NGINX n'affichera que /var/log/nginx/*log et systemd-journal n'affichera que les logs de systemd)

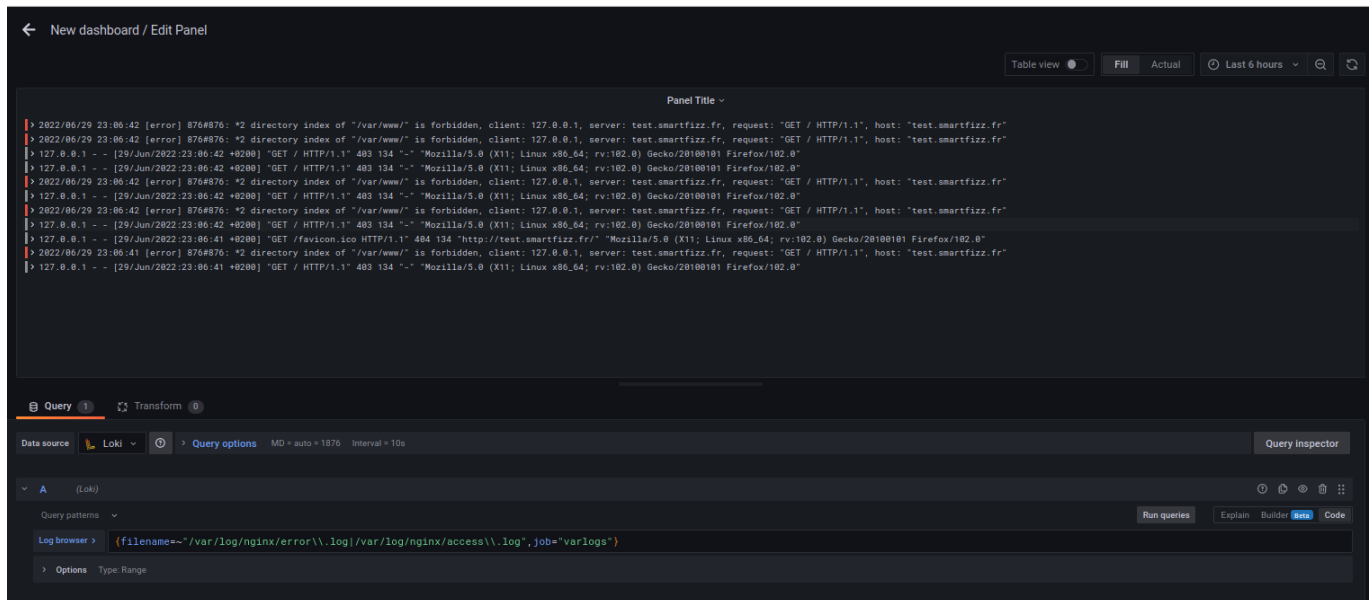
4 - Cliquez sur filename, job, unit et sélectionner les logs que vous souhaitez afficher puis sur Show logs, et enfin sur Run Queries

19 / 25



5 - Cliquez sur l'option Open Visualization Suggestions et choisissez le type de format dans lequel vos logs seront affichés

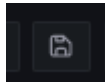




The screenshot shows the Grafana dashboard interface. At the top, there's a navigation bar with a back arrow and the text "New dashboard / Edit Panel". Below this, there's a panel titled "Panel Title" displaying log data. The logs show several error messages, including "directory index of '/var/www/' is forbidden" and "200 OK". Below the panel, there's a "Query" section with a "Data source" dropdown set to "Loki". The "Query options" section shows "MD = auto = 1876" and "Interval = 10s". The "Query patterns" section shows a query: `{filename=~"/var/log/nginx/error\\.log|/var/log/nginx/access\\.log",job="varlogs"}`. The "Options" section shows "Type: Range".

Les logs sont maintenant disponibles, pour sauvegarder votre métrique, Cliquez sur Apply et sauvegardez

vosre Dashboard avec l'icône



Filtrer les logs

Builder

Pour afficher certaines informations dans les logs affichés, Loki utilise son propre langage LogQL (<https://grafana.com/docs/loki/latest/logql/>) pour manipuler les données.

Reprenons la métrique créée précédemment du fichier de log `/var/log/nginx/access.log` avec les entrées suivantes :

The screenshot displays the Grafana Loki interface. At the top, a panel titled "Panel Title" shows a list of log entries. Each entry is a single line of text representing an HTTP access log, such as: `> 127.0.0.1 - - [30/Jun/2022:00:22:10 +0200] "GET / HTTP/1.1" 200 134 "-" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"`. Below the log list, the interface shows the "Query" tab with 1 query and the "Transform" tab with 0 transforms. The "Data source" is set to "Loki". The "Query options" section shows "MD = auto = 1876" and "Interval = 10s". The "Query patterns" section is expanded, showing the "Log browser" button and the query `{filename="/var/log/nginx/access.log",job="Logs de NGINX"}`. The "Options" section is also expanded, showing "Type: Range". At the bottom, there are buttons for "+ Query" and "+ Expression".

Si on voudrait afficher que les entrées renvoyant un code HTTP 200 , Cliquez dans la partie Builder où l'on pourra choisir certains filtres dans la section + Operations comme Line contains avec 200 + un espace pour prendre uniquement le code et les afficher :

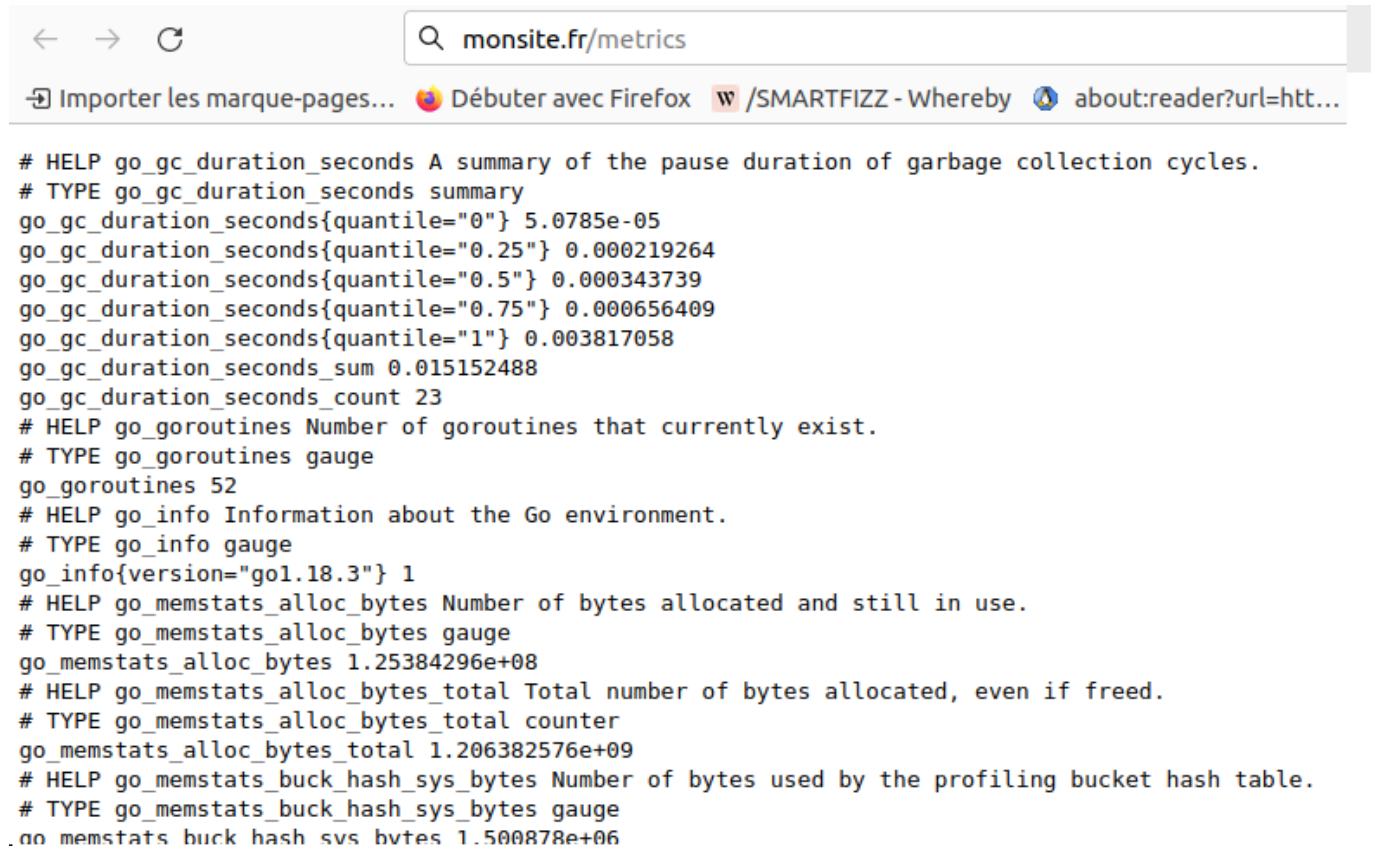
The screenshot shows the Grafana Loki Builder interface. At the top, there's a panel titled "Panel Title" displaying log entries with HTTP status 200 highlighted in yellow. Below this, the "Query" section shows a single query. The "Data source" is set to "Loki". The "Query options" section shows "MD = auto = 1260" and "Interval = 15s". The "Labels" section shows a filter for "filename" with a value of "/var/log/nginx/error\\.log" and "job" with a value of "Logs de NGINX". The "Line contains" section has a text input field containing "200". The "Raw query" section shows the generated query: `{filename=~"/var/log/nginx/error\\.log|/var/log/nginx/access\\.log", job="Logs de NGINX"} |= "200"`. The "Options" section shows "Type: Range".

A noter que l'on peut choisir d'autres informations de l'entrée comme /Jun/ pour n'avoir que les entrées datant de Juin par exemple :

The screenshot shows the Grafana Loki Builder interface with the "Line contains" filter set to "/Jun/". The "Raw query" section shows the generated query: `{filename=~"/var/log/nginx/error\\.log|/var/log/nginx/access\\.log", job="Logs de NGINX"} |= '/Jun/'`. The "Options" section shows "Type: Range".

Prometheus - Ajouter un endpoint

Une fois Prometheus installé et opérationnel, il va exposer des metrics en local sur 127.0.0.1:9090/metrics, pour récupérer les metrics d'une application / d'un outil disponibles par exemple sur monsite.fr/metrics voici une configuration d'exemple :



```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 5.0785e-05
go_gc_duration_seconds{quantile="0.25"} 0.000219264
go_gc_duration_seconds{quantile="0.5"} 0.000343739
go_gc_duration_seconds{quantile="0.75"} 0.000656409
go_gc_duration_seconds{quantile="1"} 0.003817058
go_gc_duration_seconds_sum 0.015152488
go_gc_duration_seconds_count 23
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 52
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.18.3"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 1.25384296e+08
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 1.206382576e+09
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.500878e+06
```

Pour pouvoir ajouter le endpoint sur lequel les metrics sont disponibles, modifiez le fichier de configuration prometheus.yml situé dans monitoring/prometheus/prometheus.yml et rajoutez dans scrape_configs :

```
nano monitoring/prometheus/prometheus.yml
```

```
scrape_configs:
  - job_name: "Metrics de monsite"
    static_configs:
      - targets: ["monsite.fr"]

    metrics_path: "/metrics"
```


Note : Si le endpoint `monsite.fr/metrics` nécessite une authentification basic-auth pour y avoir accès la préciser dans le fichier au niveau de `metrics_path` comme suivant :

```
basic_auth:  
  username: 'user'  
  password: 'password'
```

Redémarrez le docker avec

```
docker restart monitoring_prometheus_1
```

Vérifiez que le endpoint à bien été ajouté sur `127.0.0.1:9090/prometheus/targets`