



DevOps



Chapitre 9 : Surveillance Continue

ESPRIT – UP ASI (Architecture des Systèmes d'Information)
Bureau E204



United Nations
Educational, Scientific and
Cultural Organization



UNESCO Chair
"Project-based learning"
ESPRIT School of engineering, Tunisia



Délivrée par la
Commission
des Titres
d'Ingénieur



CONFÉRENCE DES
GRANDES
ÉCOLES



CONCEIVE DESIGN IMPLEMENT OPERATE

► Plan du cours

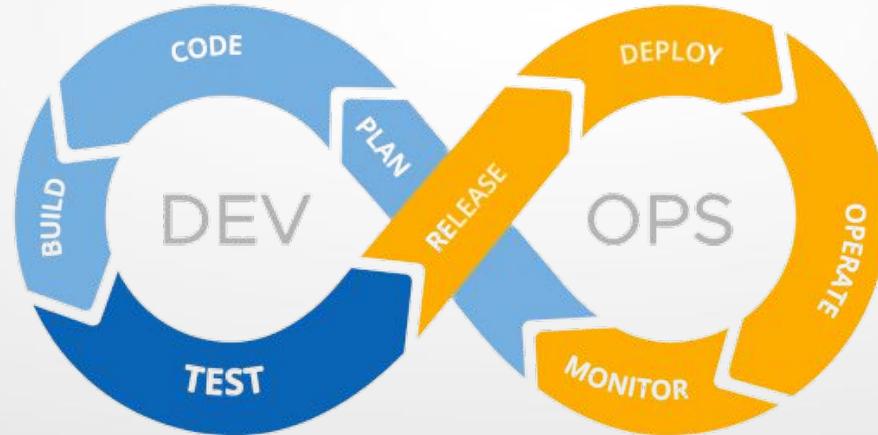
- Introduction
- Importance de la surveillance continue en DevOps
- Types de la surveillance continue
- Prometheus
 - Définition, architecture, installation, configuration
- Grafana
 - Définition, installation, configuration, dashboard
- **Travail à faire** (Surveiller Jenkins avec Prometheus et Grafana)



▶ Introduction



- Au cœur de la méthodologie DevOps, la **surveillance continue** est l'outil essentiel qui permet de maintenir un contrôle constant sur les systèmes et les applications, garantissant ainsi une prestation de services ininterrompue.



► Problématique



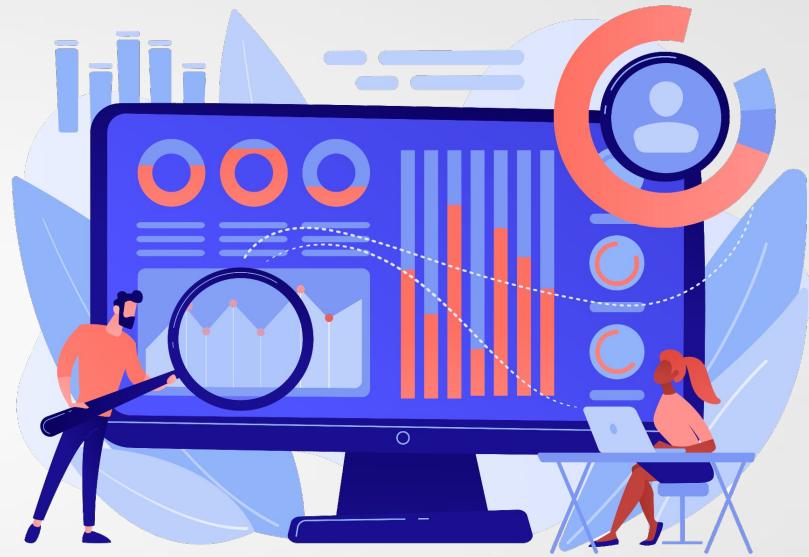
*Comment la **surveillance en temps réel** peut-il contribuer à l'optimisation des performances des applications dans un environnement DevOps, en assurant une réactivité accrue face aux problèmes potentiels et en minimisant les temps d'arrêt ?*



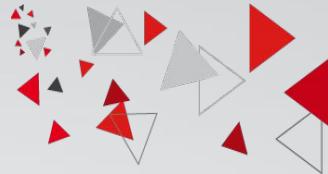
► Surveillance Continue - Importance



- › **Détection précoce des problèmes :**
 - Le monitoring repère les soucis rapidement
- › **Réaction rapide aux incidents :**
 - L'équipe réagit vite pour éviter les problèmes graves
- › **Automatisation des réparations :**
 - Les actions correctives sont automatisées
- › **Amélioration continue :**
 - Les données aident à améliorer constamment les performances
- › **Moins de temps d'arrêt :**
 - Les pannes sont minimisées, ce qui maintient l'application en ligne



► Surveillance Continue - Types



La surveillance continue en DevOps comprend *trois types de surveillance* essentiels :

› **Surveillance des performances**

Vérifier que l'application fonctionne rapidement et efficacement en termes de temps de réponse, d'utilisation des ressources (CPU, mémoire, etc.), de débit, de latence, et d'autres métriques clés.

Outils de suivi des métriques : Prometheus, Grafana



Prometheus



Grafana



► Surveillance Continue - Types



› Surveillance de l'infrastructure

Surveiller les composants matériels et logiciels qui soutiennent l'application tels que les serveurs, les bases de données, les réseaux, les conteneurs, etc.

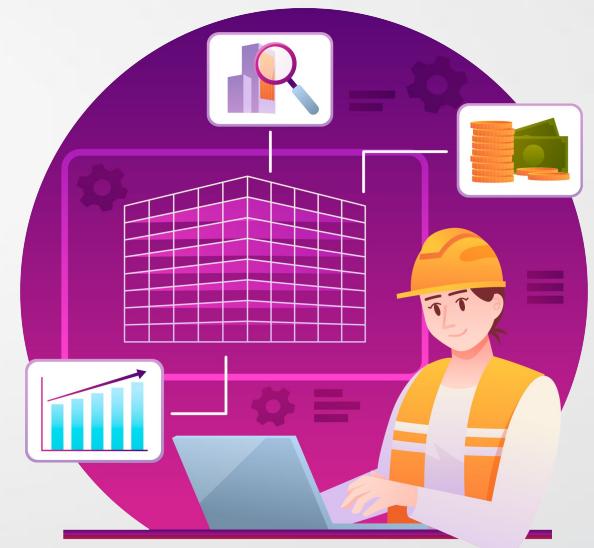
Outils de surveillance de l'infrastructure : Nagios

Outils de gestion des conteneurs : Kubernetes

Nagios®



kubernetes



► Surveillance Continue - Types



› Surveillance de sécurité

Identifier les menaces potentielles, les vulnérabilités et les activités suspectes qui pourraient compromettre la sécurité de l'application.

Outils de gestion de vulnérabilité : Nessus

Outils de détection des menaces de sécurité : ELK Stack

(Elasticsearch, Logstash, Kibana)



► Suivi des métriques - Prometheus



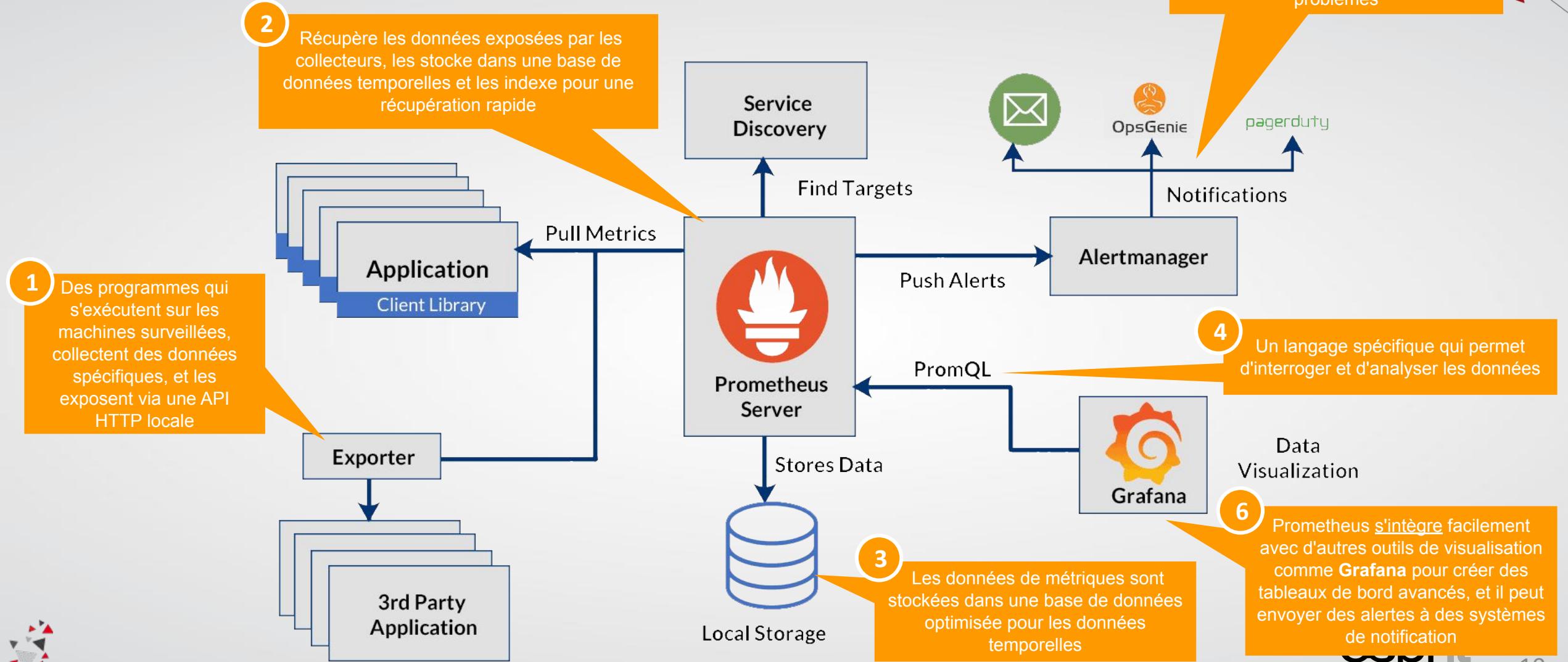
- › **Prometheus** est un logiciel libre qui surveille les systèmes informatiques, recueille des données en temps réel via HTTP, les stocke dans une base de données performante, et permet d'interroger ces données à l'aide de *PromQL*, un langage de requête simple.



- › Prometheus n'est pas conçu pour faire de la restitution d'informations sous la forme de tableau de bord. Une bonne pratique est de faire appel à un outil comme **Grafana**.



Architecture - Prometheus

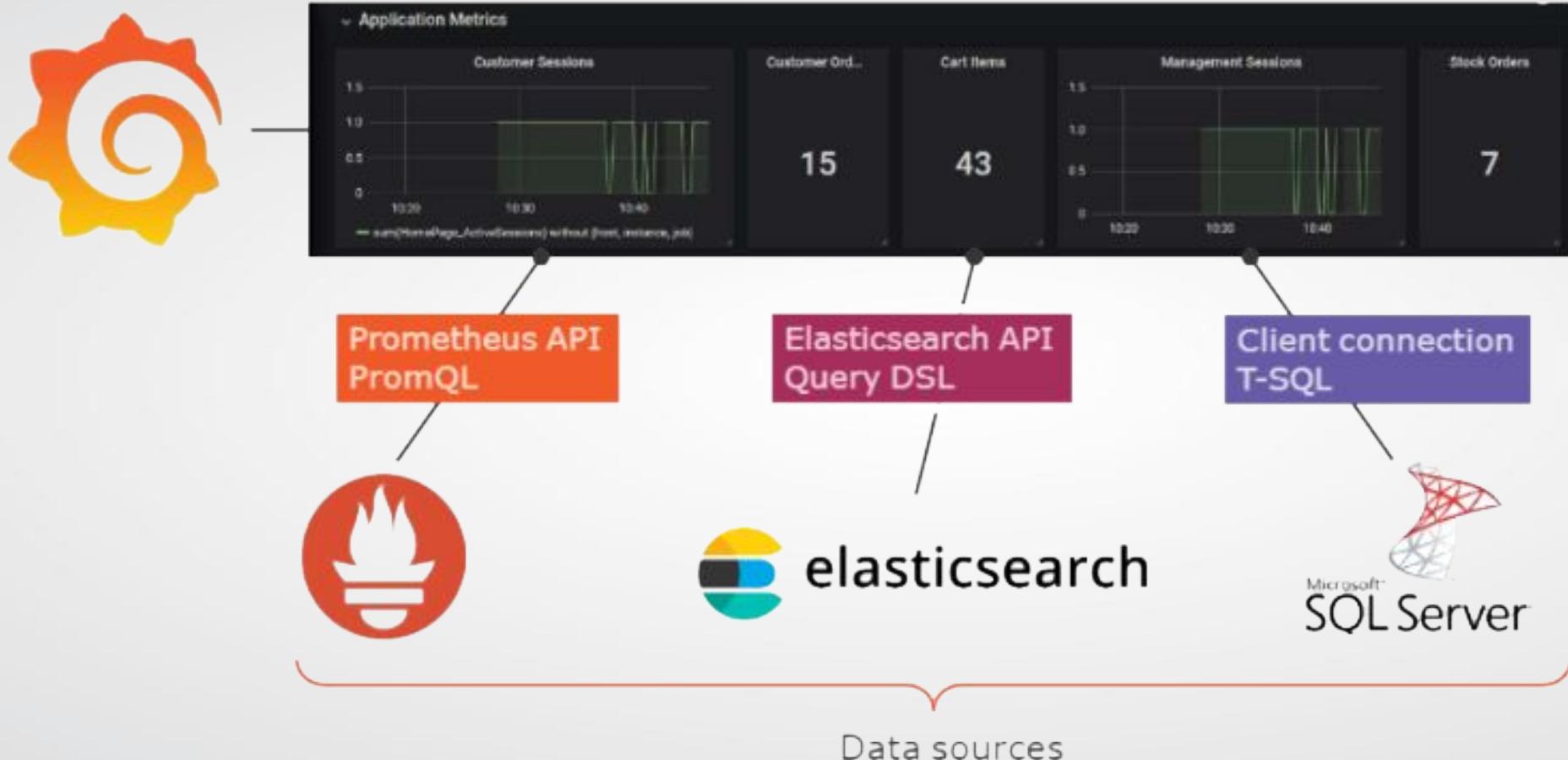


▶ Visualisation des métriques - Grafana

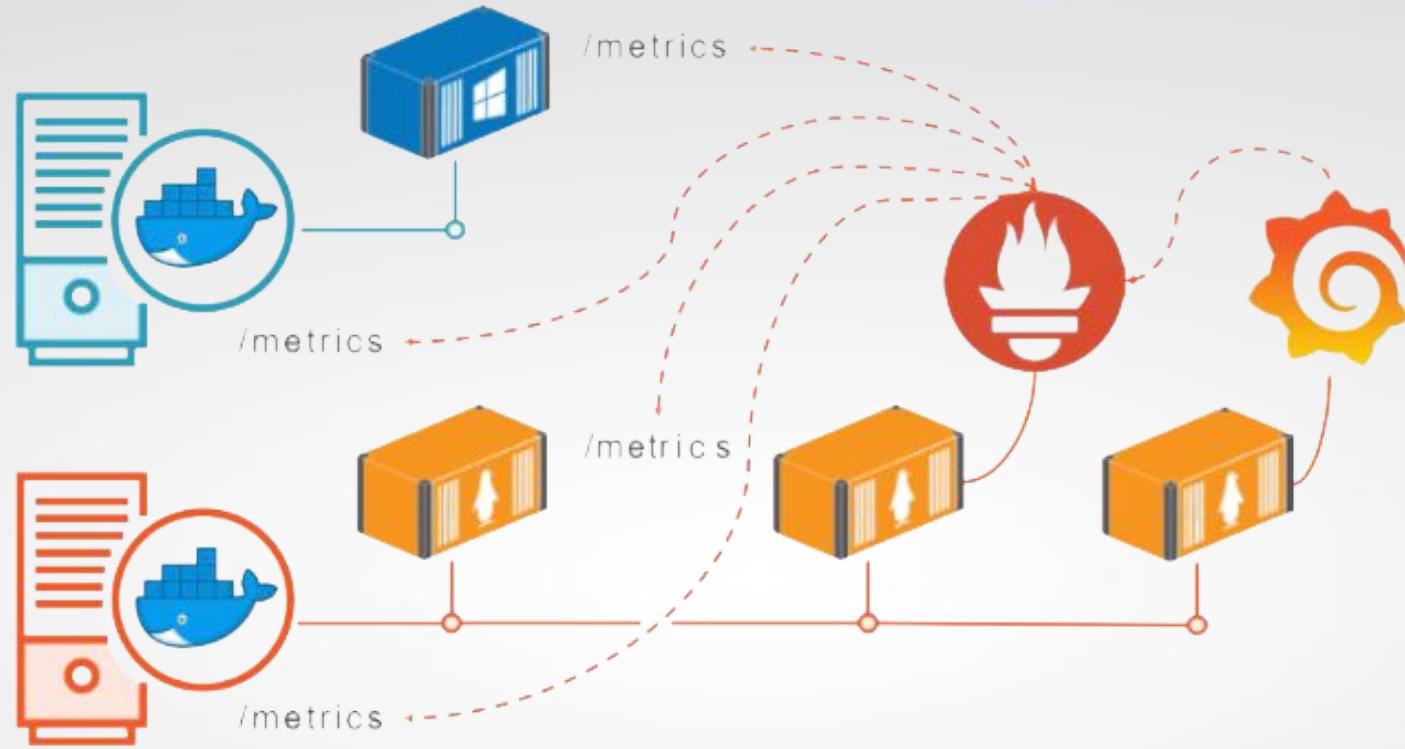
- › **Grafana** est un outil open source de monitoring informatique orienté data visualisation. Il est conçu pour générer des dashboards sur la base de métriques et données temporelles.



▶ Visualisation des métriques - Grafana



Surveillance des applications conteneurisées



- › La combinaison de Prometheus et Grafana simplifie la surveillance des applications conteneurisées avec Docker en offrant une gestion flexible, une réactivité en temps réel, et une meilleure utilisation des ressources pour des applications portables et sécurisées.



Prometheus - Installation



Prométhéus



1. Téléchargez l'image Docker de prometheus:

- › Utilisez la commande docker pull pour télécharger l'image **prometheus** depuis le Docker Hub.

```
vagrant@vagrant:~$ sudo docker pull prom/prometheus
Using default tag: latest
latest: Pulling from prom/prometheus
```

2. Exécutez le conteneur prometheus :

- › Vous pouvez maintenant lancer un conteneur prometheus en utilisant la commande docker run après la commande : sudo chmod 666 /var/run/docker.sock

```
vagrant@vagrant:~$ docker run -d --name prometheus -p 9090:9090 prom/prometheus
54b0ae3ea08ab033abf7d62b58e0cafe7980b31343e8c722dba8f830ef89312a
```

```
vagrant@vagrant:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND NAMES	CREATED	STATUS	PORTS
54b0ae3ea08a	prom/prometheus	"/bin/prometheus --c..."	9 seconds ago	Up 7 seconds	0.0.0.0:9090->9090/tcp, :::9090->9090/tcp

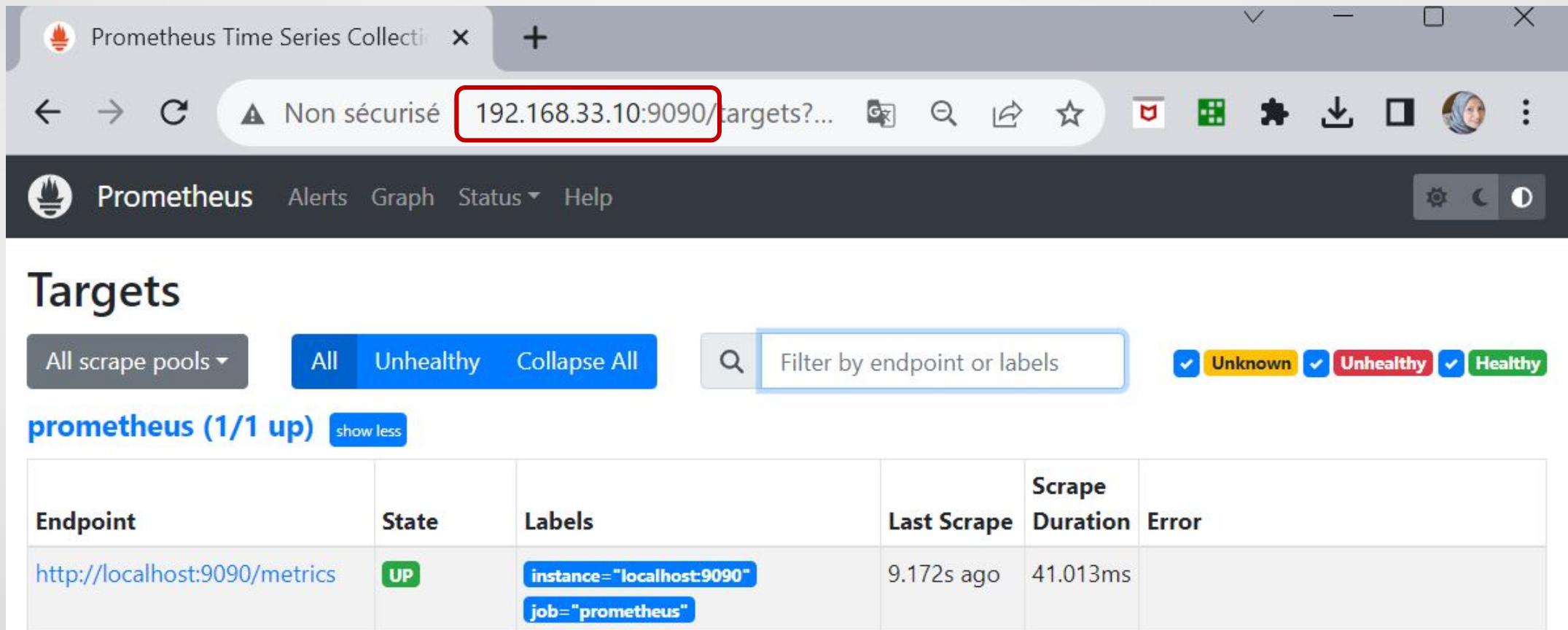
Prometheus - Installation



3. Accédez à l'interface web de Prometheus :



Vous pouvez maintenant accéder à l'interface web de Prometheus en ouvrant votre navigateur et en visitant <http://<adresse-ip-vm>:9090>.

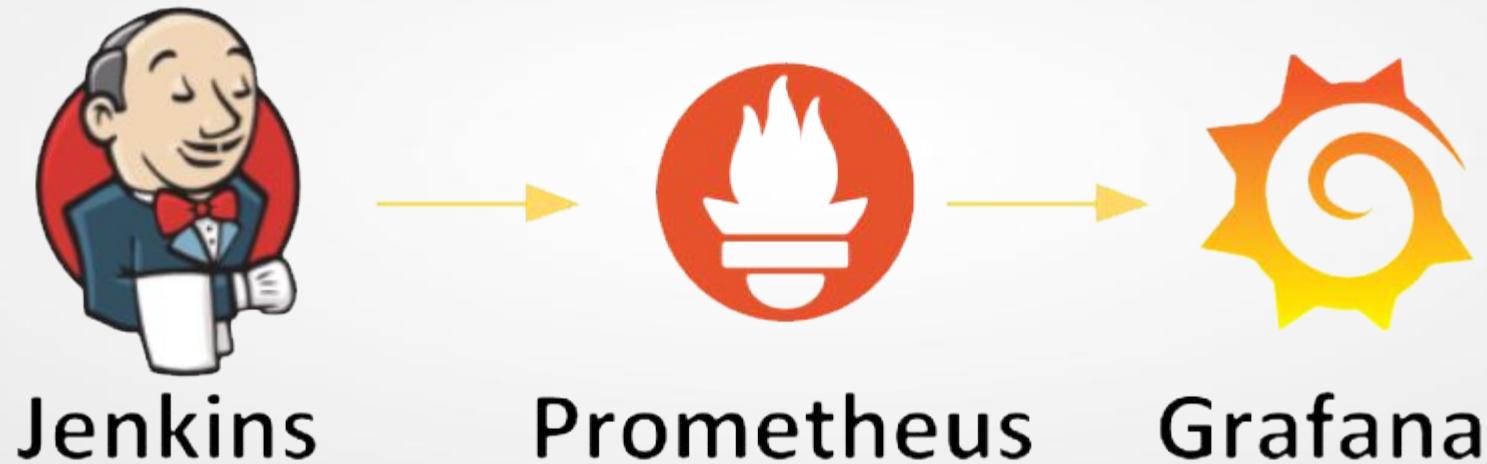
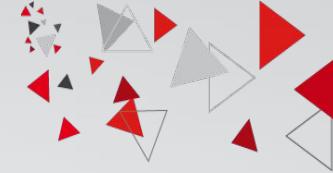


The screenshot shows the Prometheus Targets page. The URL in the address bar is highlighted with a red box: '192.168.33.10:9090/targets?...'. The page title is 'Prometheus Time Series Collector'. The navigation bar includes links for Prometheus, Alerts, Graph, Status, and Help. The main content area is titled 'Targets' and shows a single target named 'prometheus' which is '1/1 up'. The status is 'UP' and it has labels 'instance="localhost:9090"' and 'job="prometheus"'. The last scrape was 9.172s ago with a duration of 41.013ms.

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	9.172s ago	41.013ms	



Configuration de Prometheus pour récupérer les métriques de Jenkins et les visualiser via Grafana



Prometheus - Configuration



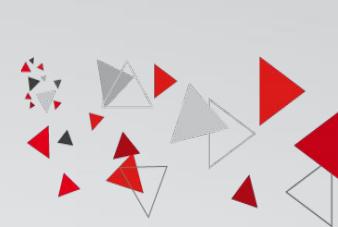
1. Installer le plugin “Prometheus metrics” dans Jenkins :

The screenshot shows the Jenkins Plugins management interface. On the left, there's a sidebar with links: Updates (30), Available plugins (highlighted in light grey), Installed plugins, and Advanced settings. The main area has a search bar at the top with the placeholder "rechercher (CTRL+K)". Below it, a search result for "prometheus" is shown in a card. The card includes the plugin name "Prometheus metrics 2.3.3", its category "monitoring", its status "Released", the date "1 mo. 12 j ago", and a short description: "Jenkins Prometheus Plugin expose an endpoint (default /prometheus) with metrics where a Prometheus Server can scrape." There are "Install" and "Uninstall" buttons at the bottom right of the card.

Vous devez redémarrer Jenkins en utilisant la commande "systemctl restart jenkins" pour que la mise à jour prenne effet.



Prometheus - Configuration



2. Configurer le serveur prometheus :

- › Pour configurer le serveur Prometheus, vous devrez modifier le fichier de configuration « `prometheus.yml` » à l'intérieur du conteneur en ajoutant une nouvelle configuration pour Jenkins.

```
vagrant@vagrant:~$ docker exec -it prometheus sh
/prometheus $ tee -a /etc/prometheus/prometheus.yml <<EOF
>   - job_name: jenkins
>     metrics_path: /prometheus
>     static_configs:
>       - targets: ['192.168.33.10:8080']
> EOF
```



Vous pouvez obtenir la commande complète à partir du fichier "prometheus.yml.txt" sur le Drive.

Prometheus - Configuration



3. Vérifier la configuration :

- › Pour vérifier votre configuration, vous pouvez utiliser la commande suivante :

```
docker exec prometheus cat /etc/prometheus/prometheus.yml
```

```
# A scrape configuration containing exactly one endpoint to scrape:  
# Here it's Prometheus itself.  
scrape_configs:  
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.  
  - job_name: "prometheus"  
  
    # metrics_path defaults to '/metrics'  
    # scheme defaults to 'http'.  
  
    static_configs:  
      - targets: ["localhost:9090"]  
  - job_name: jenkins  
    metric_path: /prometheus  
    static_configs:  
      - targets: ['192.168.33.10:8080']
```



Prometheus - Configuration



4. Redémarrer le service prometheus :

- › Après avoir sauvegardé la configuration, redémarrez le service Prometheus pour qu'il prenne en compte les nouvelles modifications.

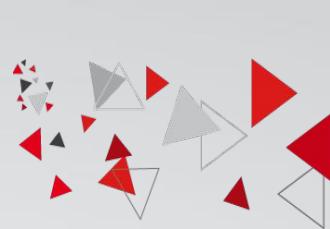
```
vagrant@vagrant:~$ docker restart prometheus
prometheus
vagrant@vagrant:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
502b7083a909        prom/prometheus   "/bin/prometheus --c..."   17 minutes ago    Up 5 seconds      0.0.0.0:9090->9090/tcp,
vagrant@vagrant:~$ |
```



Prometheus



Prometheus - Configuration



- › Après le redémarrage, vous pouvez maintenant accéder à l'interface web de prometheus en visitant <http://<adresse-ip-vm>:9090/targets>

Screenshot of the Prometheus Targets page showing two targets: Jenkins and Prometheus.

The top navigation bar includes links for Prometheus, Alerts, Graph, Status, Help, and configuration icons.

The main section is titled "Targets" and displays two entries:

- Jenkins (1/1 up)**:
 - Endpoint: <http://192.168.33.10:8080/prometheus>
 - State: UP
 - Labels: instance="192.168.33.10:8080" job="jenkins"
 - Last Scrape: 7.835s ago
 - Scrape Duration: 58.149ms
 - Error: None
- prometheus (1/1 up)**:
 - Endpoint: <http://localhost:9090/metrics>
 - State: UP
 - Labels: instance="localhost:9090" job="prometheus"
 - Last Scrape: 17.215s ago
 - Scrape Duration: 14.190ms
 - Error: None

Prometheus - Configuration



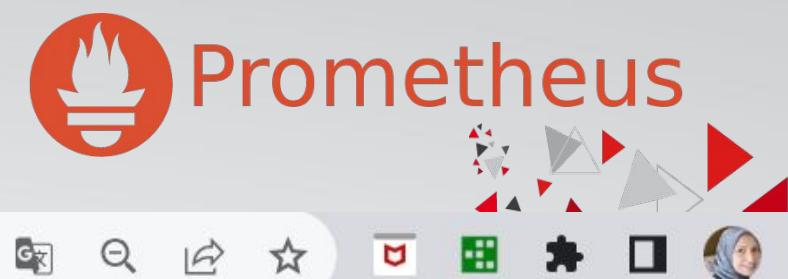
- › Pour visualisez les métriques de surveillance de Jenkins, vous pouvez accéder à l'interface web de prometheus en visitant <http://<adresse-ip-vm>:8080/prometheus>

The screenshot shows the Prometheus Targets page. At the top, there is a navigation bar with links for Prometheus, Alerts, Graph, Status, and Help. Below the navigation bar, the title "Targets" is displayed. There are three buttons: "All scrape pools ▾", "All" (which is selected), "Unhealthy", and "Collapse All". To the right of these buttons is a search bar with the placeholder "Filter by endpoint or labels". Below this, a section titled "jenkins (1/1 up)" contains a "show less" button. A table lists the target configuration for Jenkins. The columns are: Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. The first row shows the configuration for Jenkins:

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.33.10:8080/prometheus	UP	instance="192.168.33.10:8080" job="jenkins"	7.835s ago	58.149ms	



Prometheus - métriques



◀ ▶ ⌂

⚠ Non sécurisé | 192.168.33.10:8080/prometheus/



```
# HELP jvm_gc_collection_seconds Time spent in a given JVM garbage collector in seconds.
# TYPE jvm_gc_collection_seconds summary
jvm_gc_collection_seconds_count{gc="G1 Young Generation",} 115.0
jvm_gc_collection_seconds_sum{gc="G1 Young Generation",} 8.16
jvm_gc_collection_seconds_count{gc="G1 Old Generation",} 0.0
jvm_gc_collection_seconds_sum{gc="G1 Old Generation",} 0.0
# HELP default_jenkins_version_info Jenkins Application Version
# TYPE default_jenkins_version_info gauge
default_jenkins_version_info{version="2.414.1",} 1.0
# HELP default_jenkins_up Is Jenkins ready to receive requests
# TYPE default_jenkins_up gauge
default_jenkins_up 1.0
# HELP default_jenkins_uptime Time since Jenkins machine was initialized
# TYPE default_jenkins_uptime gauge
default_jenkins_uptime 6322901.0
# HELP default_jenkins_nodes_online Jenkins nodes online status
# TYPE default_jenkins_nodes_online gauge
# HELP default_jenkins_builds_duration_milliseconds_summary Summary of Jenkins build times in milliseconds by Job
# TYPE default_jenkins_builds_duration_milliseconds_summary summary
default_jenkins_builds_duration_milliseconds_summary_count{jenkins_job="SpringAOP",repo="NA",buildable="true",} 3.0
default_jenkins_builds_duration_milliseconds_summary_sum{jenkins_job="SpringAOP",repo="NA",buildable="true",} 22143.0
default_jenkins_builds_duration_milliseconds_summary_count{jenkins_job="ski/main",repo="NA",buildable="true",} 2.0
default_jenkins_builds_duration_milliseconds_summary_sum{jenkins_job="ski/main",repo="NA",buildable="true",} 236995.0
default_jenkins_builds_duration_milliseconds_summary_count{jenkins_job="Timesheet-DevOps",repo="NA",buildable="true",} 20.0
default_jenkins_builds_duration_milliseconds_summary_sum{jenkins_job="Timesheet-DevOps",repo="NA",buildable="true",} 3820222.0
# HELP default_jenkins_builds_success_build_count_total Successful build count
# TYPE default_jenkins_builds_success_build_count_total counter
default_jenkins_builds_success_build_count_total{jenkins_job="SpringAOP",repo="NA",buildable="true",} 3.0
default_jenkins_builds_success_build_count_total{jenkins_job="Timesheet-DevOps",repo="NA",buildable="true",} 8.0
# HELP default_jenkins_builds_failed_build_count_total Failed build count
# TYPE default_jenkins_builds_failed_build_count_total counter
default_jenkins_builds_failed_build_count_total{jenkins_job="ski/main",repo="NA",buildable="true",} 2.0
default_jenkins_builds_failed_build_count_total{jenkins_job="Timesheet-DevOps",repo="NA",buildable="true",} 12.0
# HELP default_jenkins_builds_health_score Health score of a job
# TYPE default_jenkins_builds_health_score gauge
```

Prometheus - métriques



- › L'exploitation et la compréhension des données textuelles mises à disposition par Prometheus peuvent s'avérer complexes.
- › Pour faciliter cette tâche, nous allons utiliser **Grafana**, un outil qui récupère ces données, les organise de manière visuelle et les présente sous forme de graphiques.



Grafana - Installation



1. Exécutez le conteneur grafana:

- › Vous pouvez maintenant lancer un conteneur grafana en utilisant la commande :

```
docker run -d --name grafana -p 3000:3000 grafana/grafana
```

```
vagrant@vagrant:~$ docker run -d --name grafana -p 3000:3000 grafana/grafana
Unable to find image 'grafana/grafana:latest' locally
latest: Pulling from grafana/grafana
7264a8db6415: Already exists
3cf7ed17dad5: Pull complete
```

```
vagrant@vagrant:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
NAMES
e4a1121b5d45        grafana/grafana   "/run.sh"          4 minutes ago     Up 4 minutes      0.0.0.0:3000->3000/tcp, ...
3000->3000/tcp     grafana
502b7083a909        prom/prometheus  "/bin/prometheus --c..."  2 hours ago       Up 2 hours        0.0.0.0:9090->9090/tcp, ...
9090->9090/tcp     prometheus

```

- › Vous avez la possibilité d'installer Prometheus et Grafana directement sur votre machine pour effectuer la surveillance.

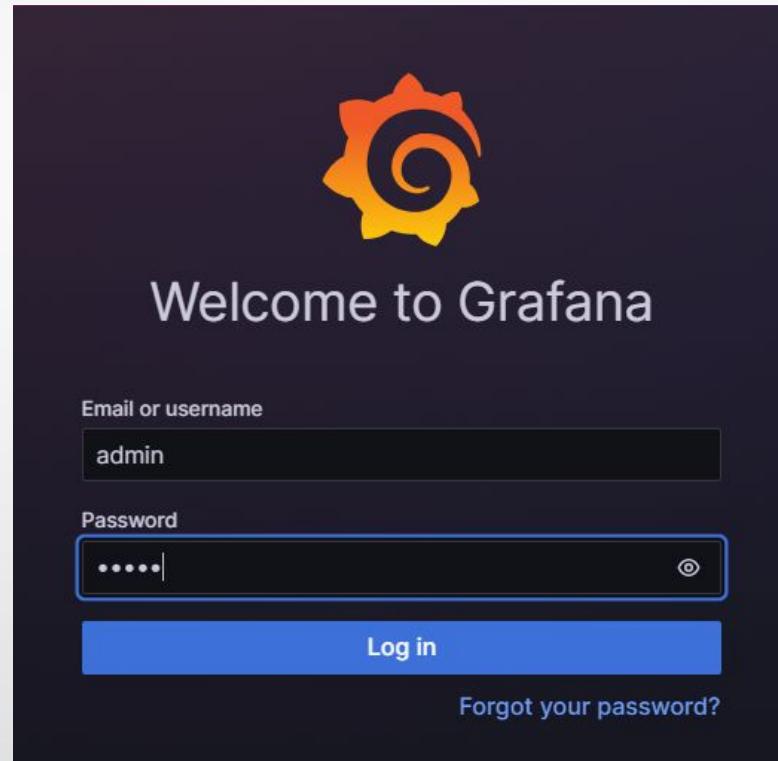


Grafana - Installation



3. Accédez à l'interface web de Grafana:

- › Vous pouvez maintenant accéder à l'interface web de Grafana en ouvrant votre navigateur et en visitant <http://<adresse-ip-vm>:3000>
- › L'utilisateur et le mot de passe par défaut sont admin/admin

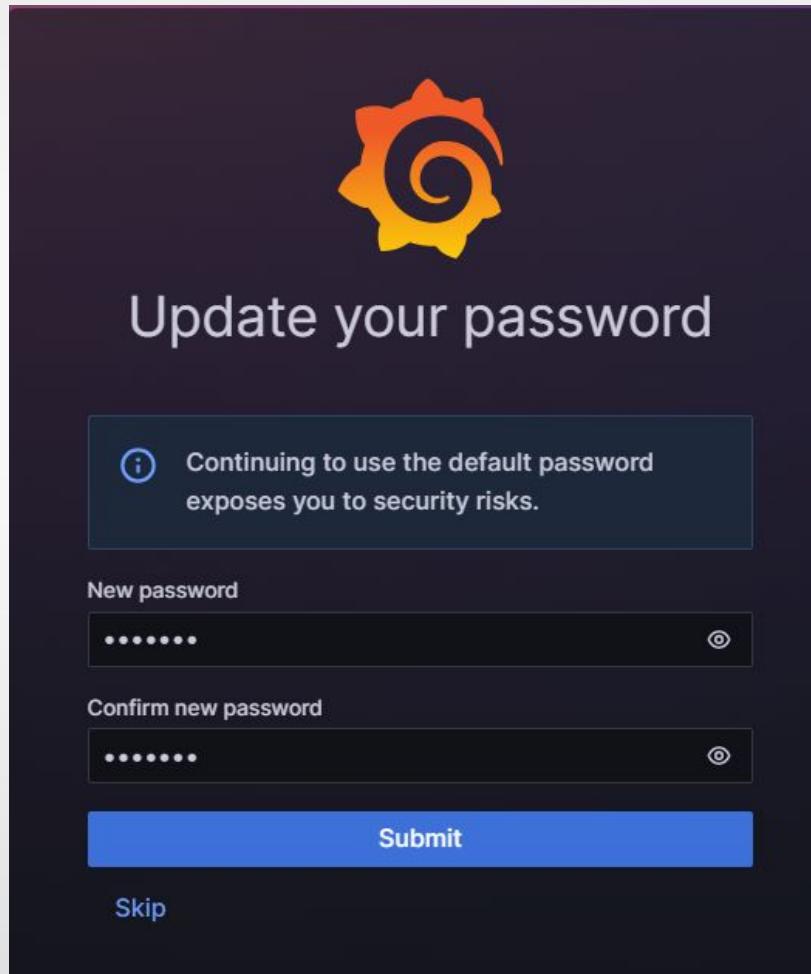


Grafana - Installation

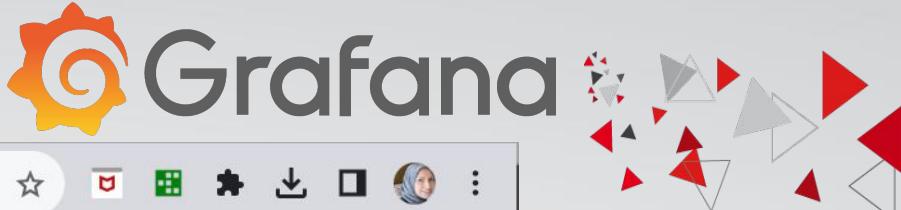


4. Changer le mot de passe de Grafana:

- › Vous pouvez changer le mot de passe par `grafana` par exemple.



Grafana - Installation



The screenshot shows the Grafana home page with a dark theme. At the top left is the Grafana logo. The top bar includes a back/forward button, a non-secure connection warning, the URL 192.168.33.10:3000/?orgId=1, and a toolbar with various icons like search, refresh, and settings. On the left, there's a sidebar with a 'Home' link and a 'Basic' section describing the setup guide. The main content area features three panels: 'TUTORIAL' (with 'DATA SOURCE AND DASHBOARDS' and 'Grafana fundamentals' sections), 'DATA SOURCES' (with 'Add your first data source' and a database icon), and 'DASHBOARDS' (with 'Create your first dashboard' and a dashboard icon). Below these are sections for 'Dashboards' (Starred dashboards, Recently viewed dashboards) and 'Latest from the blog' (an article by Maciej Nawrocki on integrating Spring Boot with Grafana).



Grafana - Configuration



- › La première étape consiste à intégrer une **source de données (Prometheus)**

The image shows two screenshots of the Grafana interface. The left screenshot is the 'Welcome to Grafana' screen, which includes a 'Basic' sidebar with instructions and a 'DATA SOURCES' section with a red box around it. The right screenshot is the 'Add data source' configuration screen, showing a list of time series databases with 'Prometheus' highlighted by a red box.

Welcome to Grafana

Need help? [Documentation](#) [Tutorials](#) [Community](#) [Public Slack](#)

Basic

The steps below will guide you to quickly finish setting up your Grafana installation.

TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals

Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.

DATA SOURCES

Add your first data source

Learn how in the docs

Remove this pane

Home > Connections > Data sources > Add data source

Add data source

Choose a data source type

Filter by name or type Cancel

Time series databases

Prometheus	Open source time series database & alerting
Graphite	Open source time series database
InfluxDB	Open source time series database

Core

Grafana - Configuration

Récupérez l'adresse IP du conteneur docker Grafana

The screenshot shows the Grafana interface with the path Home > Connections > Data sources > Jenkins. A modal window titled 'Configure your Prometheus data source below' is open, containing instructions to skip effort and get Prometheus (and Loki) as fully-managed, scalable, and hosted data sources from Grafana Labs with the free-forever Grafana Cloud plan. Below the modal, there is a section for 'Alerting supported' with a red box around the 'Name' input field, which is set to 'Jenkins'. There is also a 'Default' button and a toggle switch. At the bottom, there is a 'Connection' section with a red box around the 'Prometheus server URL' input field, which contains 'http://192.168.33.10:9090'.

Cliquez sur Save & test

The screenshot shows the Grafana Query editor. It includes sections for 'Performance' (Prometheus type: Choose, Cache level: Low, Incremental querying (beta): off, Disable recording rules (beta): off), 'Other' (Custom query parameters: Example: max_source_resolution=5m&timeout, HTTP method: POST), and 'Exemplars' (+ Add). At the bottom, there are 'Delete' and 'Save & test' buttons, with the 'Save & test' button highlighted by a red box. A success message at the bottom states 'Successfully queried the Prometheus API.'



► Grafana - configuration



Pour importer un tableau de bord dans Grafana, suivez les étapes suivantes :

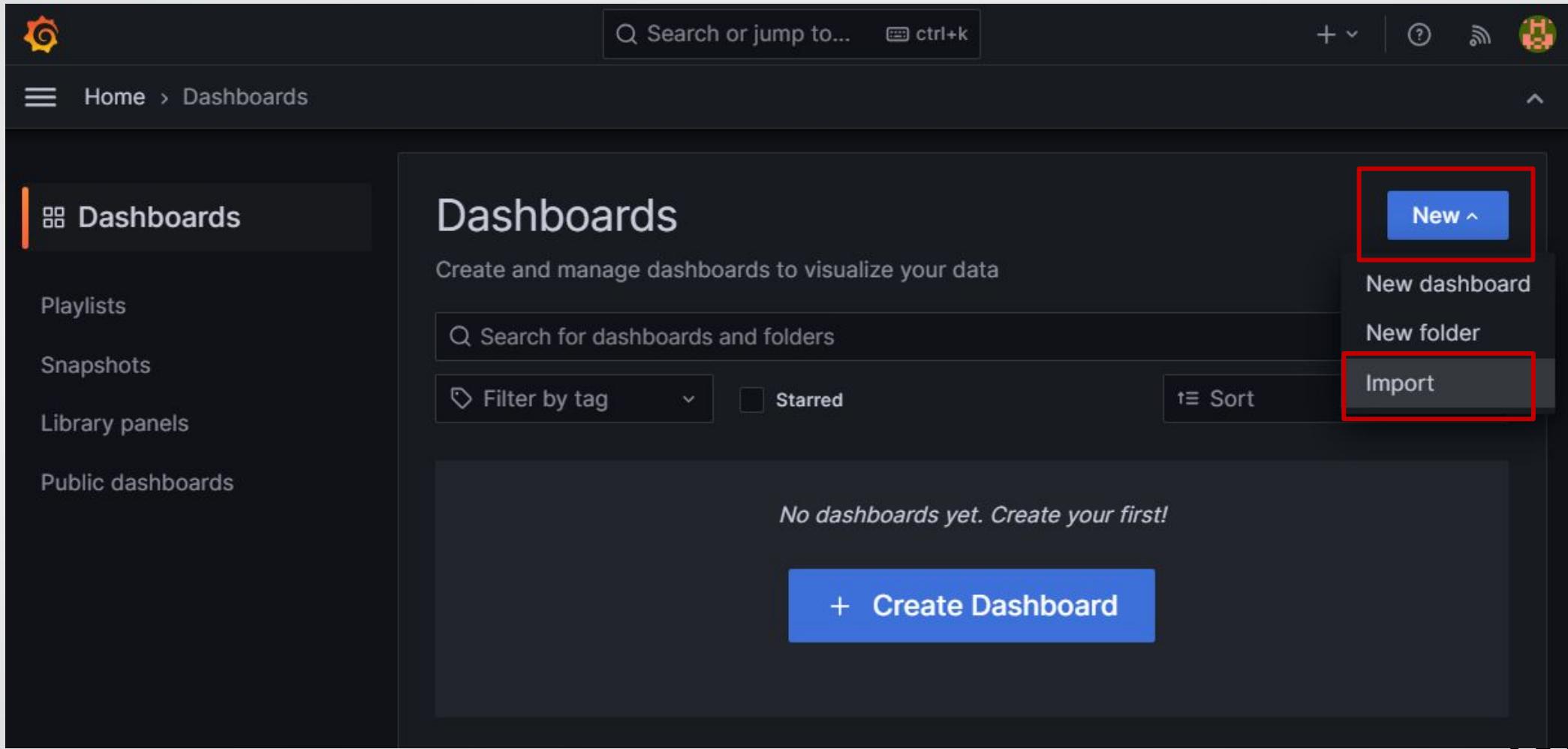
- › Accédez à l'URL *http://@IP_VM:3000/dashboards* dans votre navigateur
- › Recherchez l'option "Import" sur la page
- › Utilisez l'identifiant **9964** pour importer le tableau de bord souhaité
- › Une fois l'importation terminée, le tableau de bord sera disponible pour une utilisation immédiate dans Grafana.



► Grafana - Configuration



- › Cliquez sur "Import"



A screenshot of the Grafana interface showing the 'Dashboards' page. The left sidebar has 'Dashboards' selected. The main area shows a search bar, a 'New' button, and an 'Import' button. The 'Import' button is highlighted with a red box. The text 'No dashboards yet. Create your first!' is displayed at the bottom.



► Grafana - Configuration

- › Entrez l'identifiant **9964** du modèle de tableau de bord à importer



The screenshot shows the Grafana interface with the following details:

- Header:** Search bar with placeholder "Search or jump to..." and keyboard shortcut "ctrl+k".
- Top Bar:** Home, Dashboards, Import dashboard (highlighted), +, ?, and a gear icon.
- Left Sidebar:** Dashboards (selected), Playlists, Snapshots, Library panels, and Public dashboards.
- Main Content:**
 - Import dashboard:** Title, sub-instruction "Import dashboard from file or Grafana.com", and a file upload area with an "Upload dashboard JSON file" button and a note "Drag and drop here or click to browse". Accepted file types: .json, .txt.
 - Input Field:** A search bar containing the identifier "9964" with a red border around it, and a "Load" button to its right.
 - Bottom Panel:** "Import via dashboard JSON model" section showing a partial JSON model:

```
{  
  "title": "Example - Repeating Dictionary variables",  
  "uid": "_0HnEoN4z",  
  "panels": [...]  
  ...  
}
```



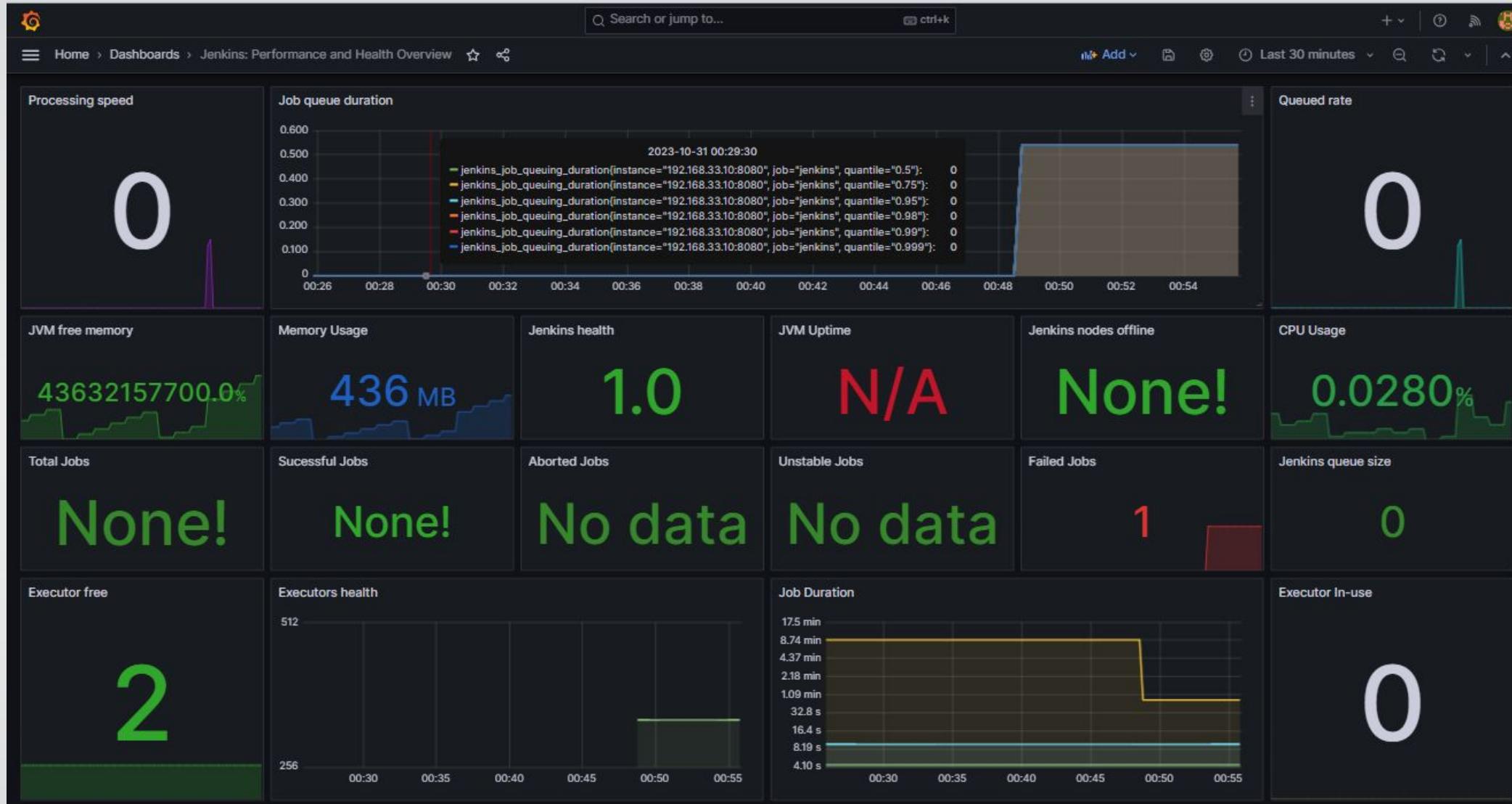
► Grafana - Configuration

- › Sélectionnez la source de données Jenkins et cliquez sur Import

The screenshot shows the 'Import dashboard' screen in Grafana. The left sidebar has 'Dashboards' selected. The main area displays the 'Import dashboard' form. It shows a title 'Importing dashboard from Grafana.com' and details 'Published by haryan' and 'Updated on 2023-08-24 10:34:53'. Under 'Options', the 'Name' field contains 'Jenkins: Performance and Health Overview' and the 'Folder' dropdown is set to 'Dashboards'. A section for 'Unique identifier (UID)' explains its purpose and shows 'haryan-jenkins' with a 'Change uid' button. Below this, a dropdown menu is open, showing 'Prometheus' at the top and 'Jenkins' listed under it. This 'Jenkins' option is highlighted with a red box. At the bottom of the form are 'Import' and 'Cancel' buttons, with 'Import' also highlighted with a red box.



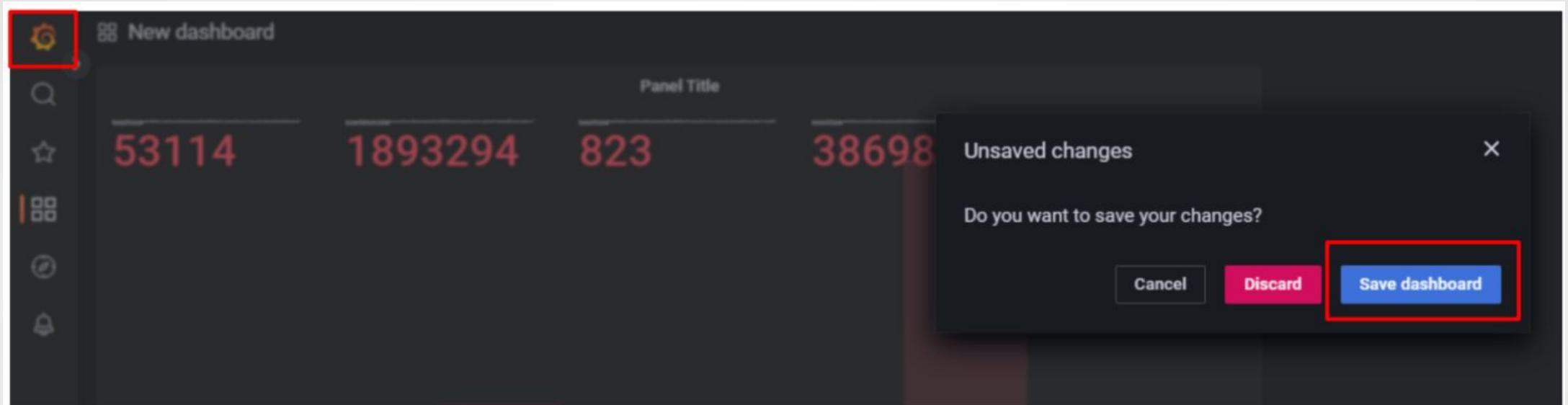
Grafana - Dashboard



► Grafana - Dashboard



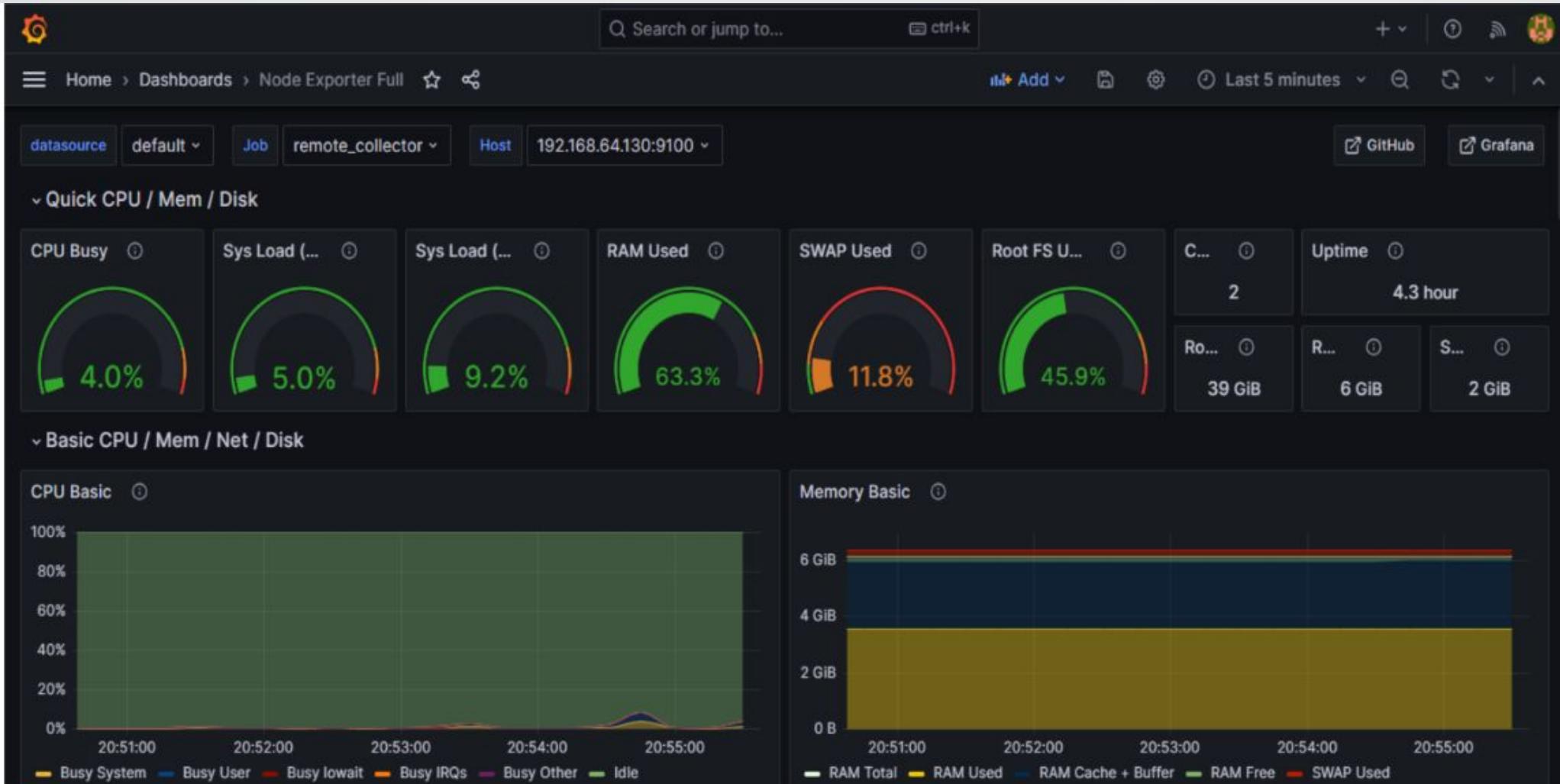
- › Sauvegardez le dashboard en cliquant sur le logo de Grafana



Grafana - Exemple Dashboard



- › Ubuntu system performance monitor

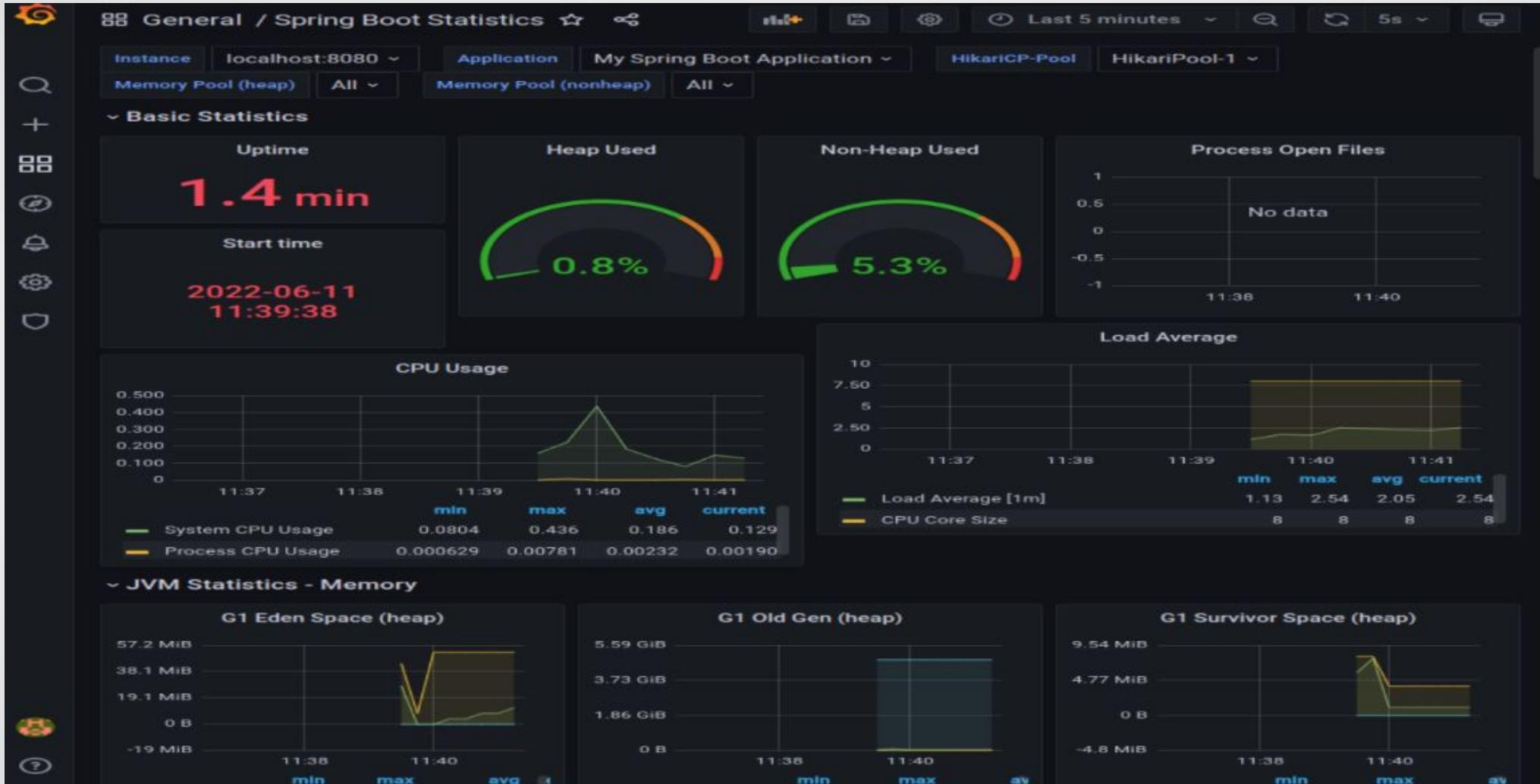


Grafana



Grafana - Exemple Dashboard

- Spring Boot Application system performance monitor



Grafana

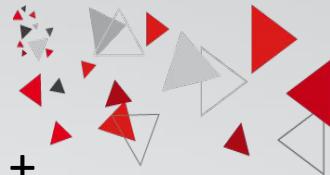


Grafana - Exemple Dashboard

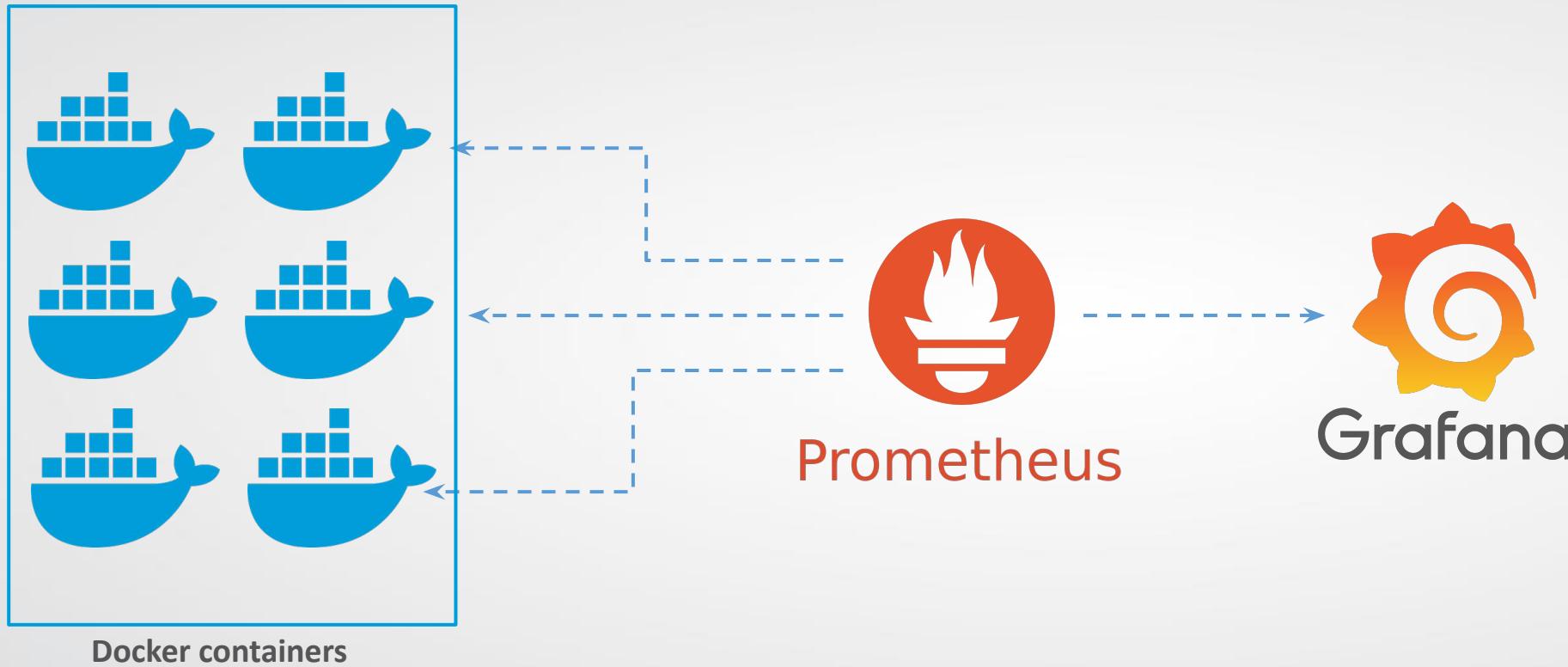
- Spring Boot Application system performance monitor



► Surveillance - Architecture



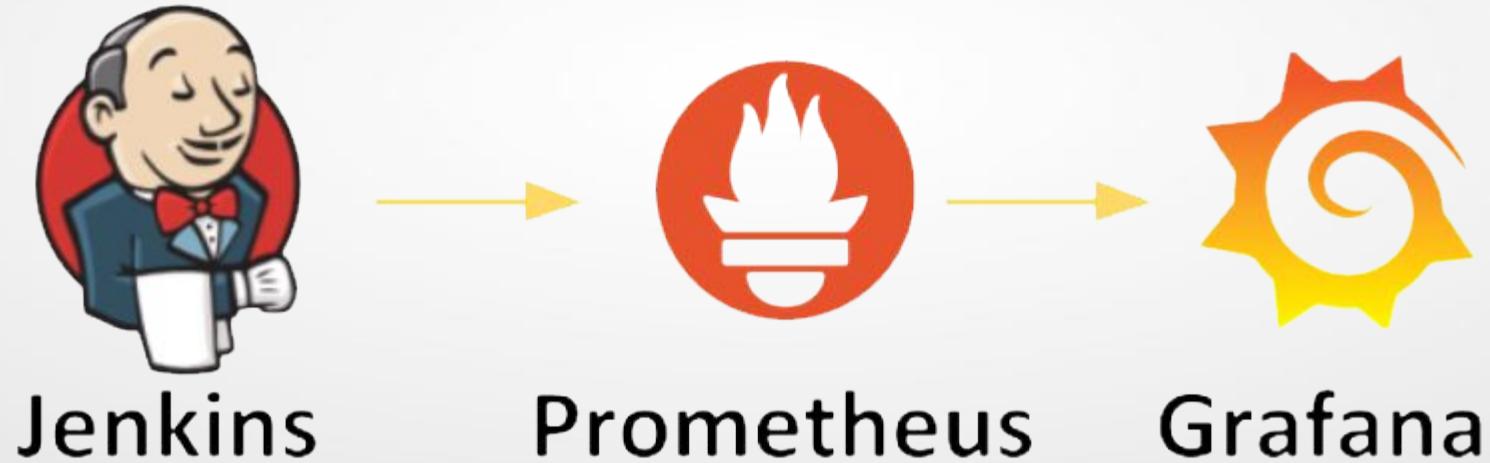
- › Vous pouvez surveiller les trois conteneurs de votre application finale (Backend Spring Boot + MySQL). Prometheus collecte les métriques en temps réel, tandis que Grafana les visualise.



► Travail à faire

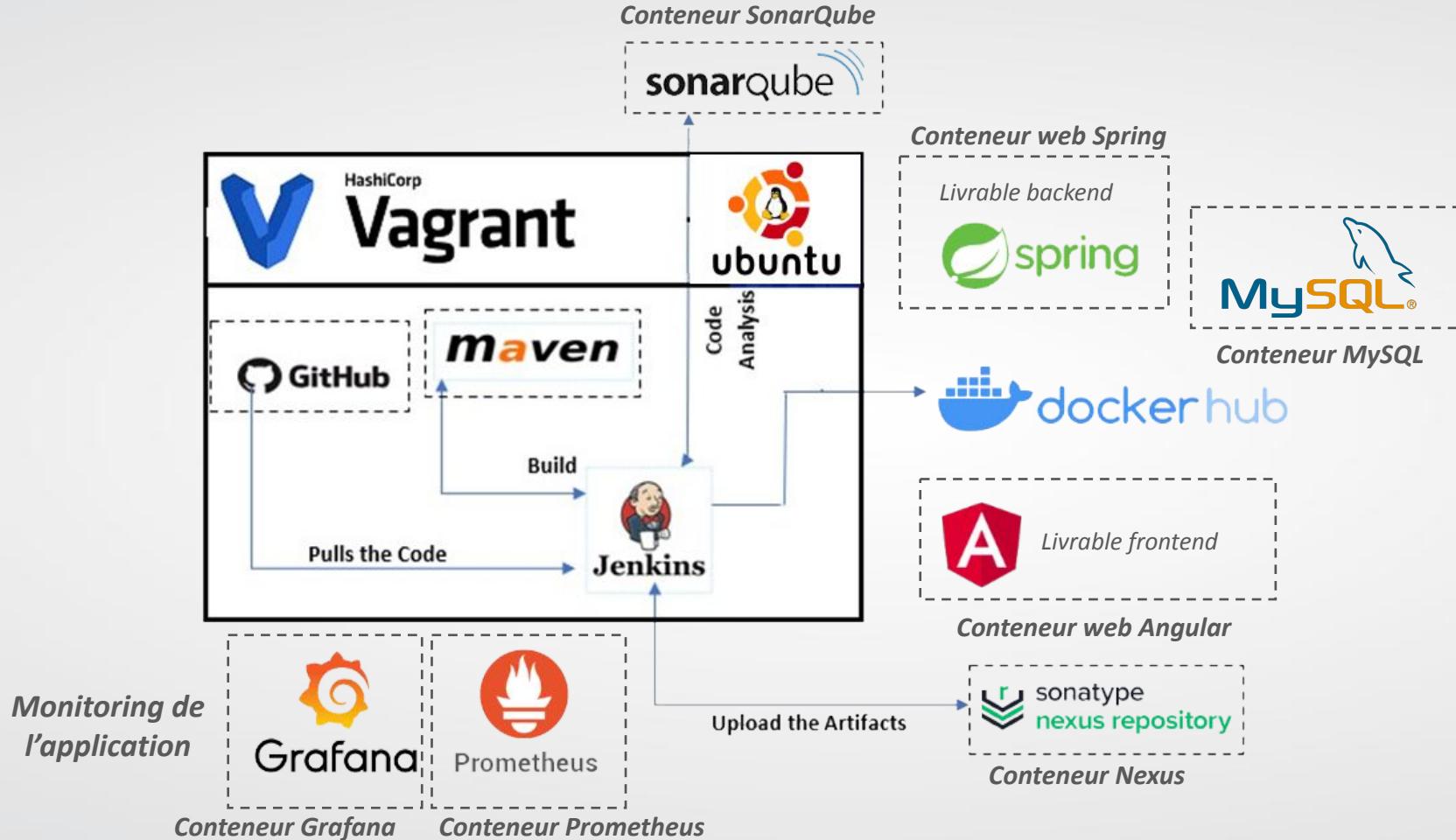


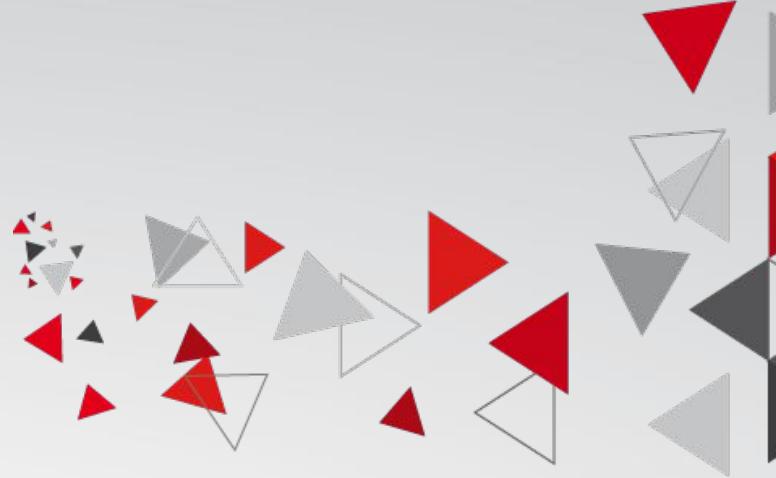
1. Installez et configurez Prometheus
2. Installez et configurez Grafana
3. Surveiller Jenkins avec Prometheus et Grafana



Solution finale

- Créez des tableaux de bord pour surveiller les divers serveurs ainsi que votre application Spring Boot





*"Apprendre par le projet, c'est découvrir
par l'action, créer par la compréhension, et
réussir par la persévérance."*



ESPRIT – UP ASI (Architecture des Systèmes d'Information)
Bureau E204

