Monitoring CI/CD PHP Web Application with Grafana and Prometheus

Version	1.0.0
Prepared by	Rimah Houssameldine
Audience	ParkInnovation Team.
Date	2024-3-3

Table of Contents

Introduction:

This documentation provides step-by-step instructions for setting up monitoring for a CI/CD PHP web application using Grafana and Prometheus. Monitoring your CI/CD pipeline is crucial for ensuring the reliability and performance of your application.

Technologies Used:

- Grafana: A visualization and monitoring tool.
- Prometheus: A monitoring and alerting toolkit.
- Docker: For containerization.
- Jenkins: CI/CD automation server.
- Windows Server: Host for the PHP web application.

Prerequisites

Before proceeding, ensure you have the following:

- Access to Grafana and Prometheus servers.
- Docker installed on the target hosts.
- Jenkins server up and running on localhost:8080.
- Prometheus server up and running on localhost:9090.
- Windows Server with appropriate metrics enabled.

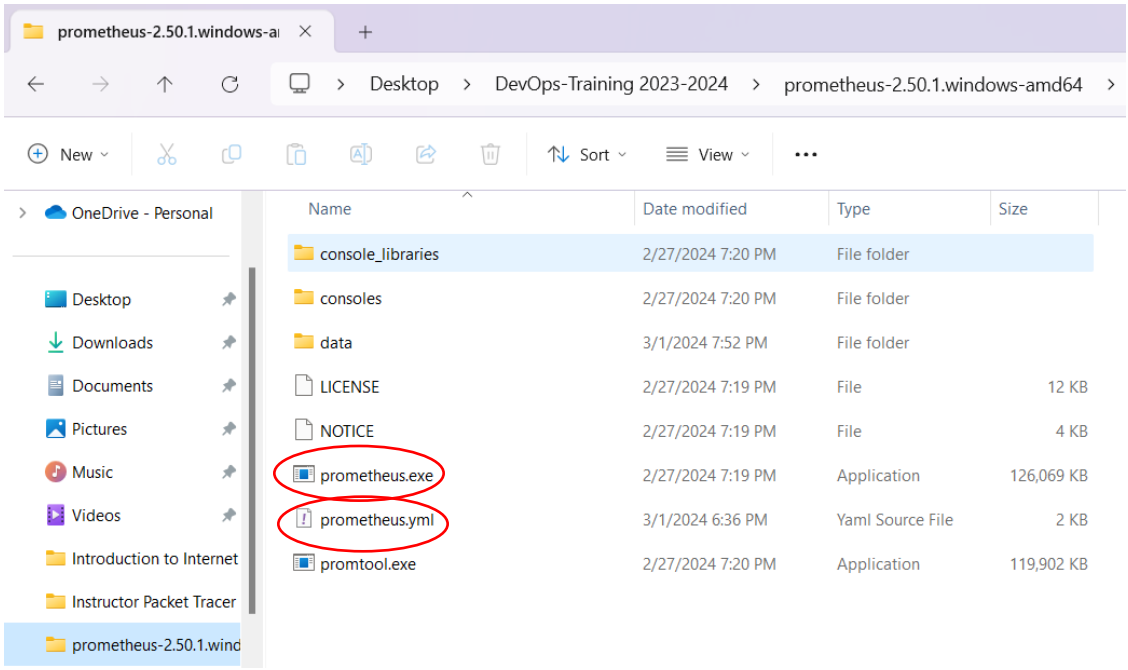
Prepared by: Rimah Houssameldine
Audience: ParkInnovation Team.

Date: 2024-3-3
Version: 1.0.0

Steps for Monitoring CI/CD PHP Web Application with Grafana and Prometheus

Setting Up Prometheus

Configure Prometheus scrape jobs for Docker, Jenkins, Windows Server, and Prometheus itself.



prometheus.yml

C: > Users > user > Desktop > DevOps-Training 2023-2024 > prometheus-2.50.1.windows-amd64 > prometheus.yml > ...

```yaml
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.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"]

      # Add this for Jenkins
  - job_name: "jenkins"
    metrics_path: "/prometheus"
    static_configs:
      - targets: ["localhost:8080"]

  - job_name: "windows_exporter"
    metrics_path: "/metrics"
    static_configs:
      - targets: ["localhost:9182"]
```

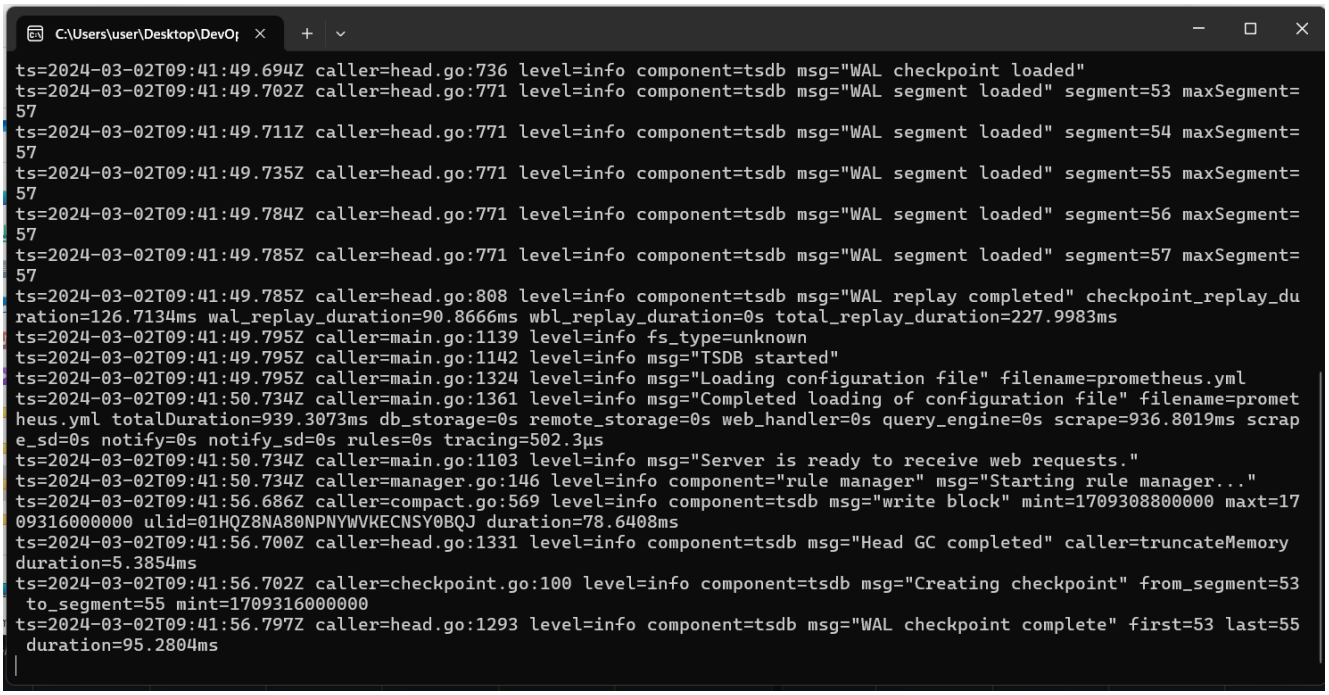
prometheus.yml

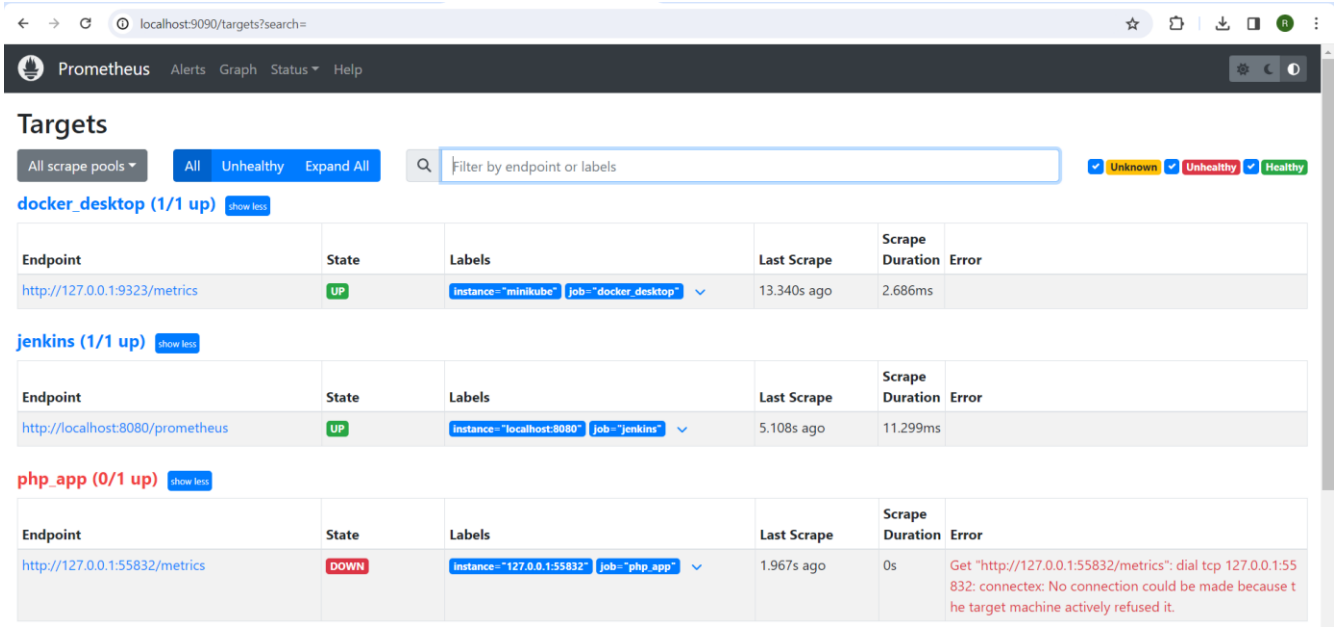
C: > Users > user > Desktop > DevOps-Training 2023-2024 > prometheus-2.50.1.windows-amd64 > prometheus.yml > ...

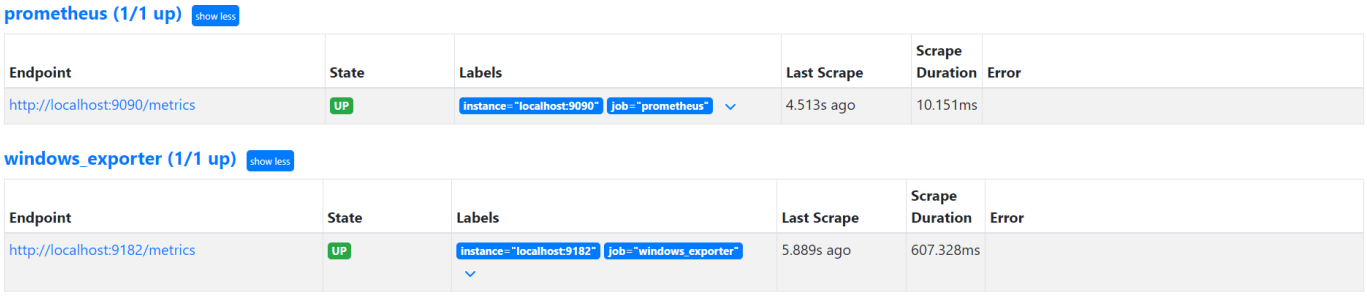
```
21  scrape_configs:
42    - job_name: 'php_app'
43      static_configs:
44      - targets: ["127.0.0.1:55832"]  # Change this to match your PHP application's address
45      metrics_path: "/metrics"        # Change this to match the path where your metrics are exposed
46
47
48    - job_name: "docker_desktop"
49      metrics_path: "/metrics"
50      static_configs:
51      - targets: ["127.0.0.1:9323"]
52      relabel_configs:
53      - source_labels: [__address__]
54        target_label: instance
55        replacement: "minikube"
56      metric_relabel_configs:
57      - source_labels: [node]
58        target_label: node
59        regex: '(minikube)'
60        replacement: $1
61      - source_labels: [container_name]
62        target_label: container
63        regex: '.+'
64        replacement: "minikube"
65      - source_labels: [__name__]
66        target_label: __name__
67        regex: 'node_memory_.*'
68        action: keep
69
70
```

This YAML configuration file for Prometheus sets the global scrape and evaluation intervals to 15 seconds each, along with a default scrape timeout of 10 seconds. Alerting configurations, although present, are commented out, indicating no specific alert manager is configured. Rule files are also commented out, suggesting that no additional rule files are loaded. The file then defines scrape configurations for various job types including Prometheus itself, Jenkins, Windows Server, a PHP application, and Docker Desktop. Each job specifies the targets and metrics paths for scraping metrics, with additional configurations for modifying labels and filtering metrics as required.

Verify Prometheus configuration by accessing Prometheus web UI (http://prometheus-server-ip:9090).


C:\Users\user\Desktop\DevOp
ts=2024-03-02T09:41:49.694Z caller=head.go:736 level=info component=tsdb msg="WAL checkpoint loaded"
ts=2024-03-02T09:41:49.702Z caller=head.go:771 level=info component=tsdb msg="WAL segment loaded" segment=53 maxSegment=57
ts=2024-03-02T09:41:49.711Z caller=head.go:771 level=info component=tsdb msg="WAL segment loaded" segment=54 maxSegment=57
ts=2024-03-02T09:41:49.735Z caller=head.go:771 level=info component=tsdb msg="WAL segment loaded" segment=55 maxSegment=57
ts=2024-03-02T09:41:49.784Z caller=head.go:771 level=info component=tsdb msg="WAL segment loaded" segment=56 maxSegment=57
ts=2024-03-02T09:41:49.785Z caller=head.go:771 level=info component=tsdb msg="WAL segment loaded" segment=57 maxSegment=57
ts=2024-03-02T09:41:49.785Z caller=head.go:808 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=126.7134ms wal_replay_duration=90.8666ms wbl_replay_duration=0s total_replay_duration=227.9983ms
ts=2024-03-02T09:41:49.795Z caller=main.go:1139 level=info fs_type=unknown
ts=2024-03-02T09:41:49.795Z caller=main.go:1142 level=info msg="TSDB started"
ts=2024-03-02T09:41:49.795Z caller=main.go:1324 level=info msg="Loading configuration file" filename=prometheus.yml
ts=2024-03-02T09:41:50.734Z caller=main.go:1361 level=info msg="Completed loading of configuration file" filename=prometheus.yml totalDuration=939.3073ms db_storage=0s remote_storage=0s web_handler=0s query_engine=0s scrape=936.8019ms scrape_sd=0s notify=0s notify_sd=0s rules=0s tracing=502.3µs
ts=2024-03-02T09:41:50.734Z caller=main.go:1103 level=info msg="Server is ready to receive web requests."
ts=2024-03-02T09:41:50.734Z caller=manager.go:146 level=info component="rule manager" msg="Starting rule manager..."
ts=2024-03-02T09:41:56.686Z caller=compact.go:569 level=info component=tsdb msg="write block" mint=1709308800000 maxt=1709316000000 ulid=01HQZ8NA80NPNYWVKECNSY0BQJ duration=78.6408ms
ts=2024-03-02T09:41:56.700Z caller=head.go:1331 level=info component=tsdb msg="Head GC completed" caller=truncateMemory duration=5.3854ms
ts=2024-03-02T09:41:56.702Z caller=checkpoint.go:100 level=info component=tsdb msg="Creating checkpoint" from_segment=53 to_segment=55 mint=1709316000000
ts=2024-03-02T09:41:56.797Z caller=head.go:1293 level=info component=tsdb msg="WAL checkpoint complete" first=53 last=55 duration=95.2804ms


localhost:9090/targets?search=
Prometheus
Alerts
Graph
Status
Help
Targets
All scrape pools
All
Unhealthy
Expand All
Filter by endpoint or labels
Unknown
Unhealthy
Healthy
docker_desktop (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://127.0.0.1:9323/metrics
UP
instance="minikube"
job="docker_desktop"
13.340s ago
2.686ms
jenkins (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://localhost:8080/prometheus
UP
instance="localhost:8080"
job="jenkins"
5.108s ago
11.299ms
php_app (0/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://127.0.0.1:55832/metrics
DOWN
instance="127.0.0.1:55832"
job="php_app"
1.967s ago
0s
Get "http://127.0.0.1:55832/metrics": dial tcp 127.0.0.1:55832: connectex: No connection could be made because the target machine actively refused it.



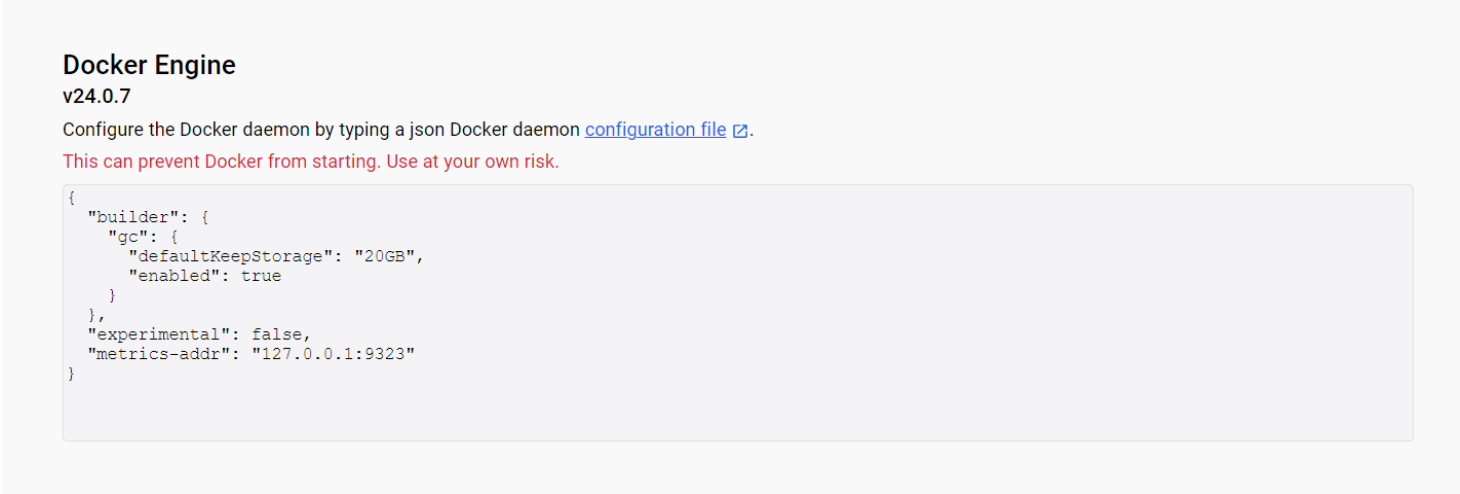
prometheus (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	4.513s ago	10.151ms	

windows_exporter (1/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9182/metrics	UP	instance="localhost:9182" job="windows_exporter"	5.889s ago	607.328ms	

Docker Engine Configuration



Docker Engine

v24.0.7

Configure the Docker daemon by typing a json Docker daemon configuration file.

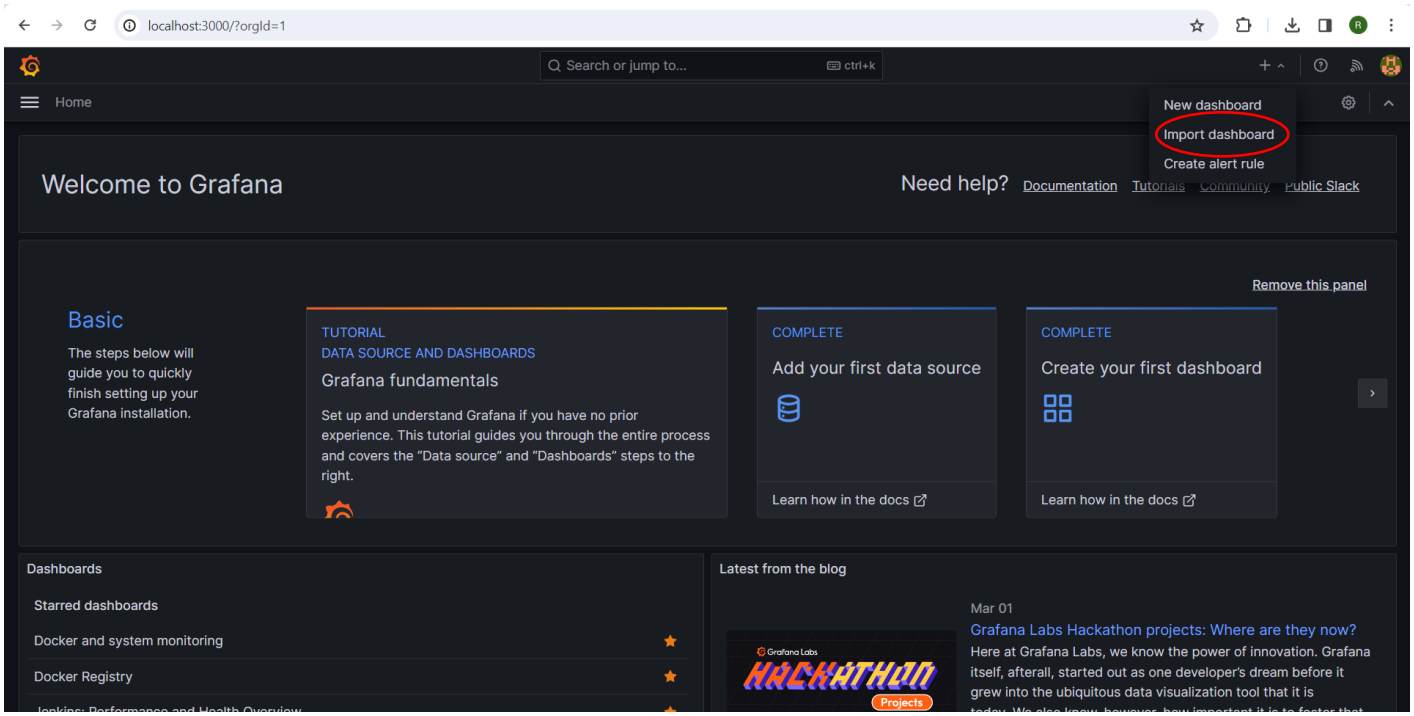
This can prevent Docker from starting. Use at your own risk.

```
{
  "builder": {
    "gc": {
      "defaultKeepStorage": "20GB",
      "enabled": true
    }
  },
  "experimental": false,
  "metrics-addr": "127.0.0.1:9323"
}
```

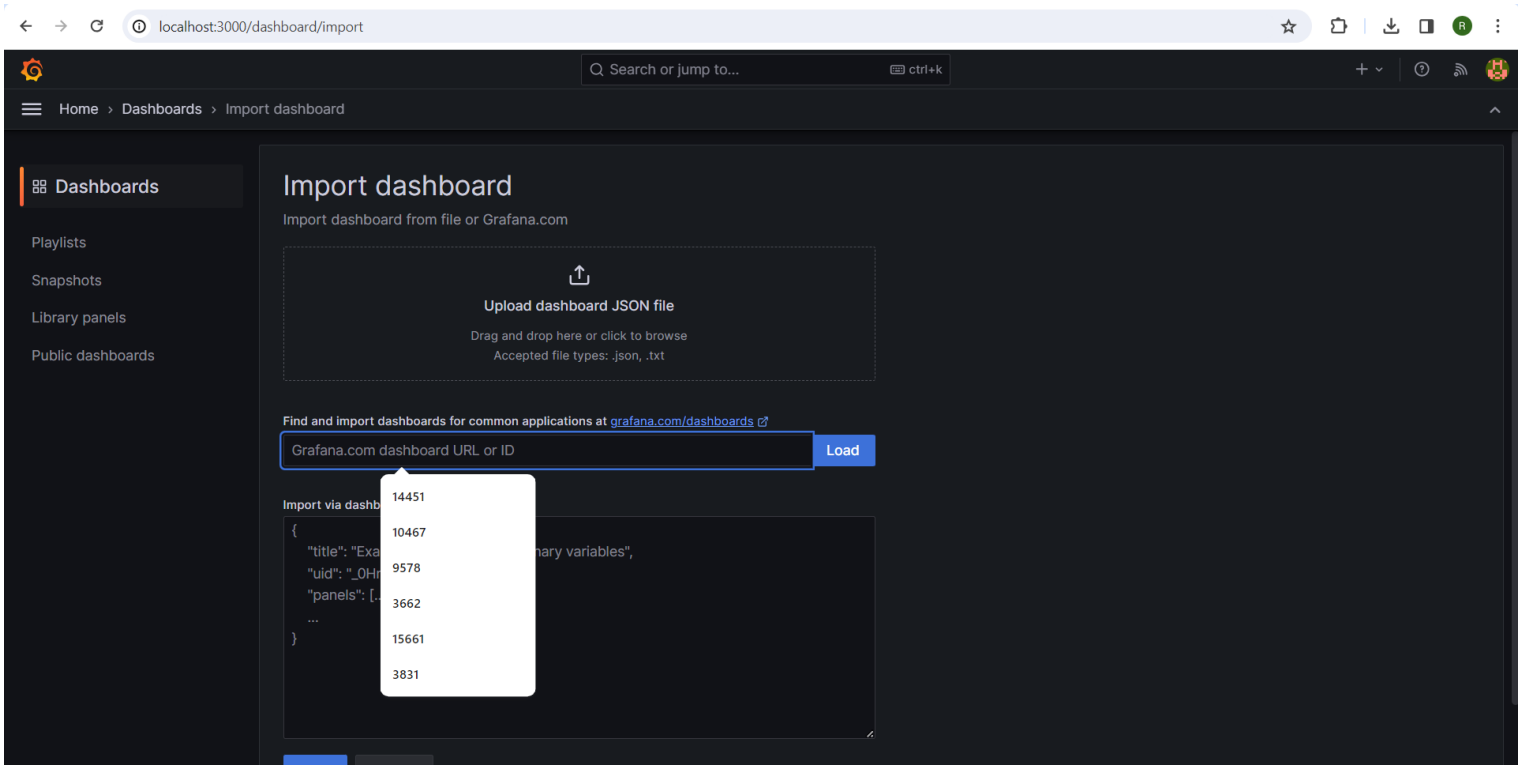
This JSON snippet configures settings for a container runtime environment, likely Docker. Within the "builder" section, it defines parameters related to garbage collection, including the default amount of storage to retain ("20GB") and whether garbage collection is enabled ("true"). The "experimental" parameter is set to "false," indicating that experimental features are disabled. Lastly, the "metrics-add" parameter specifies the address and port ("127.0.0.1:9323") where metrics are exposed, facilitating monitoring of the container runtime environment. Overall, this configuration snippet enables efficient resource management, ensures stability by disabling experimental features, and provides metrics for monitoring purposes.

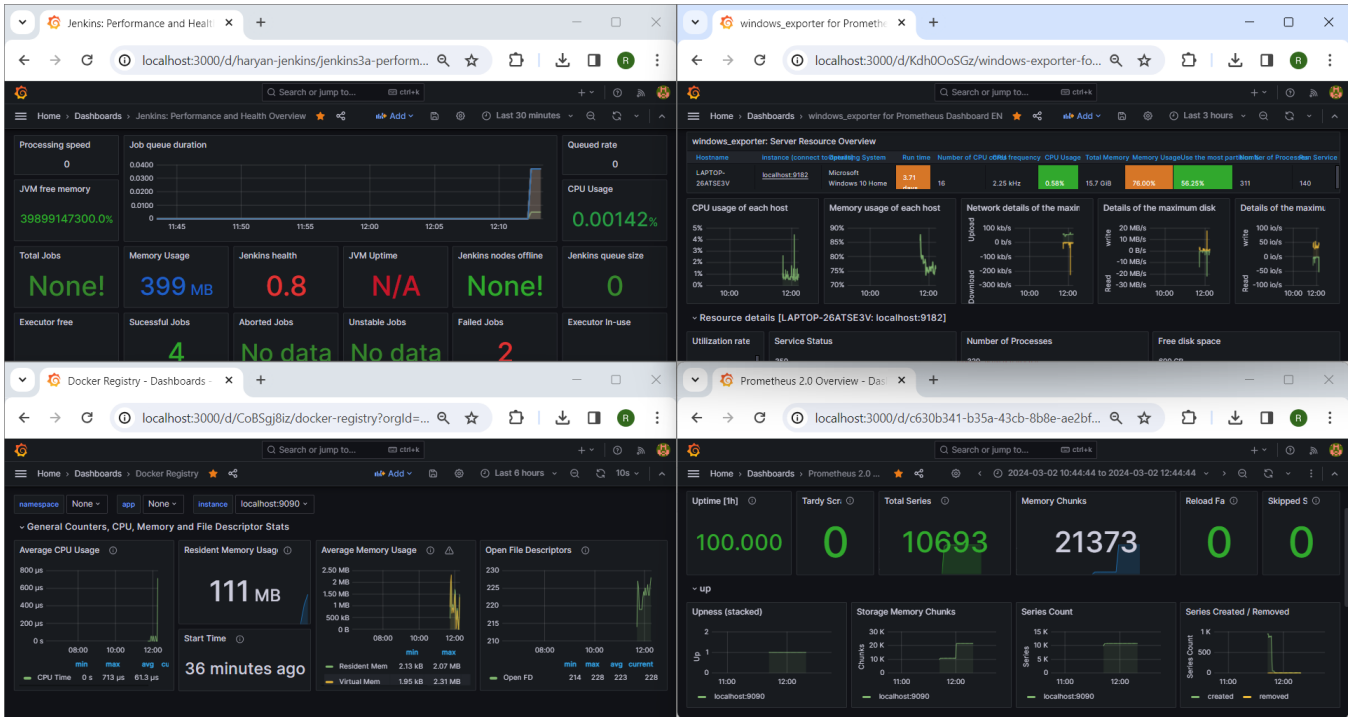
Setting Up Grafana

Verify Grafana configuration by accessing Grafana web UI (http://grafana-server-ip:3000).

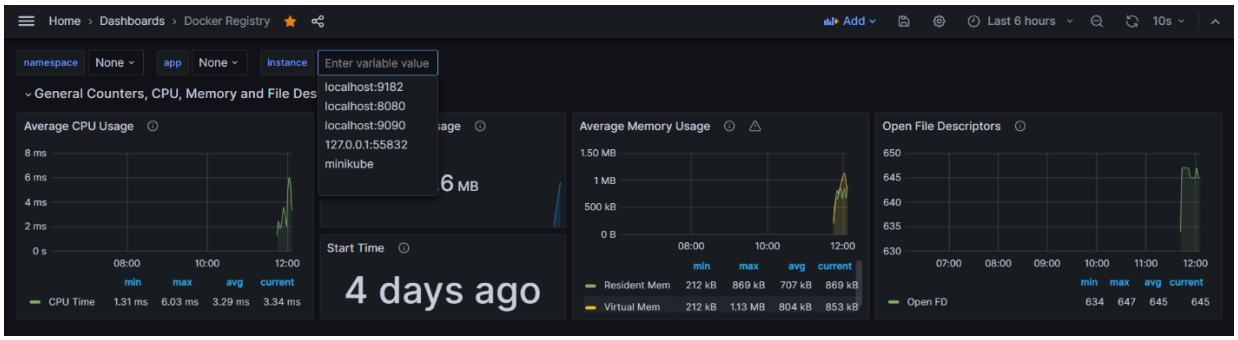


Create Grafana dashboard for Docker metrics visualization, Jenkins metrics visualization, Windows Server metrics visualization, and Prometheus server metrics visualization.


localhost:3000/dashboard/import
Search or jump to...
ctrl+k
Home › Dashboards › Import dashboard
Dashboards
Playlists
Snapshots
Library panels
Public dashboards
Import dashboard
Import dashboard from file or Grafana.com
Upload dashboard JSON file
Drag and drop here or click to browse
Accepted file types: .json, .txt
Find and import dashboards for common applications at grafana.com/dashboards
Grafana.com dashboard URL or ID
Load
Import via dashb
14451
10467
9578
3662
15661
3831

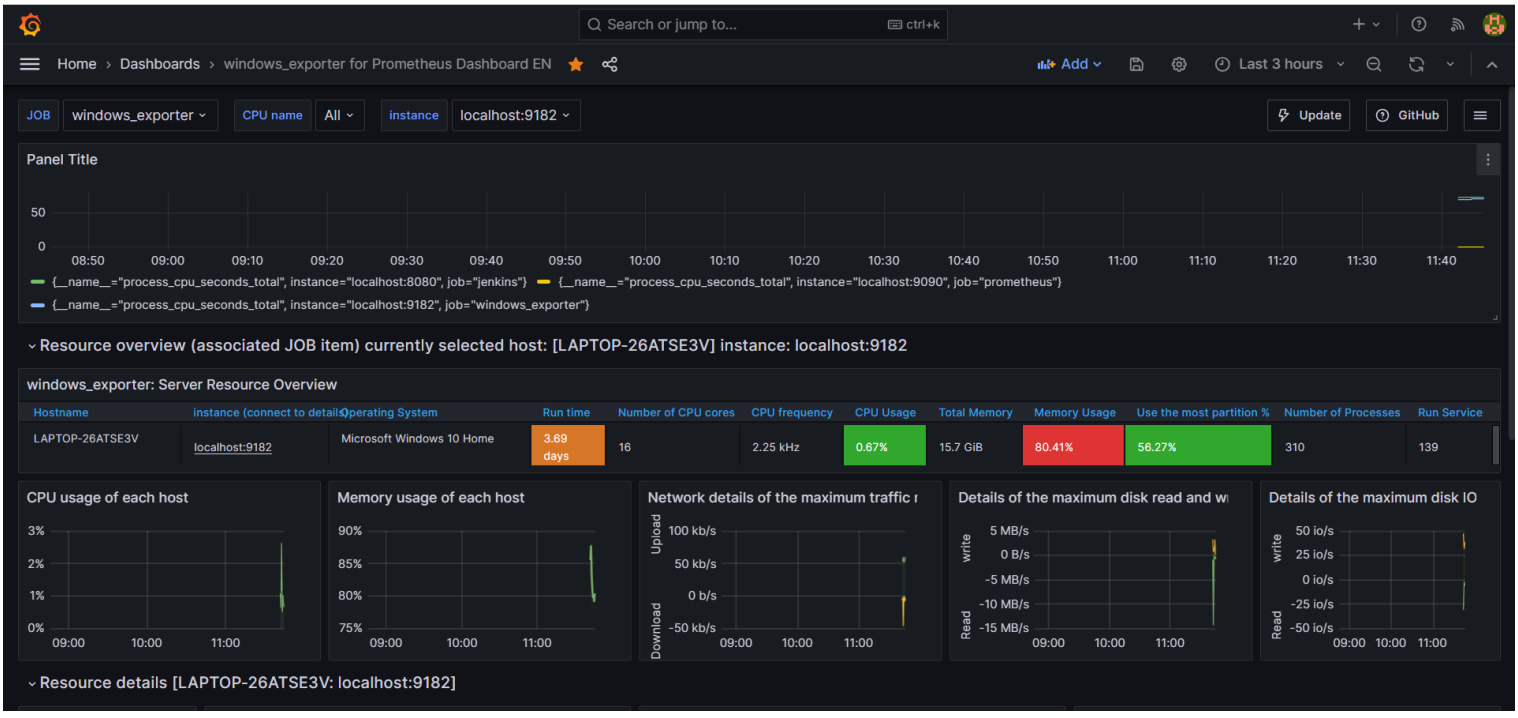

Jenkins: Performance and Healt
localhost:3000/d/haryan-jenkins/jenkins3a-perform...
Home › Dashboards › Jenkins: Performance and Health Overview
Processing speed
Job queue duration
Queued rate
JVM free memory
39899147300.0%
CPU Usage
0.00142%
Total Jobs
None!
Memory Usage
399 MB
Jenkins health
0.8
JVM Uptime
N/A
Jenkins nodes offline
None!
Jenkins queue size
0
Executor free
Sucessful Jobs
4
Aborted Jobs
No data
Unstable Jobs
No data
Failed Jobs
2
Executor In-use
windows_exporter for Prometh
localhost:3000/d/Kdh0OoSGz/windows-exporter-fo...
Home › Dashboards › windows_exporter for Prometheus Dashboard EN
windows_exporter: Server Resource Overview
CPU usage of each host
Memory usage of each host
Network details of the maxir
Details of the maximum disk
Resource details [LAPTOP-26ATSE3V: localhost:9182]
Utilization rate
Service Status
Number of Processes
Free disk space
Docker Registry - Dashboards
localhost:3000/d/CoBSgj8iz/docker-registry?orgId=...
Home › Dashboards › Docker Registry
General Counters, CPU, Memory and File Descriptor Stats
Average CPU Usage
Resident Memory Usage
111 MB
Average Memory Usage
Open File Descriptors
Start Time
36 minutes ago
Prometheus 2.0 Overview - Das
localhost:3000/d/c630b341-b35a-43cb-8b8e-ae2bf...
Home › Dashboards › Prometheus 2.0 ...
Uptime [1h]
100.000
Tardy Scr
0
Total Series
10693
Memory Chunks
21373
Reload Fa
0
Skipped S
0
up
Upness (stacked)
Storage Memory Chunks
Series Count
Series Created / Removed

Docker metrics visualization

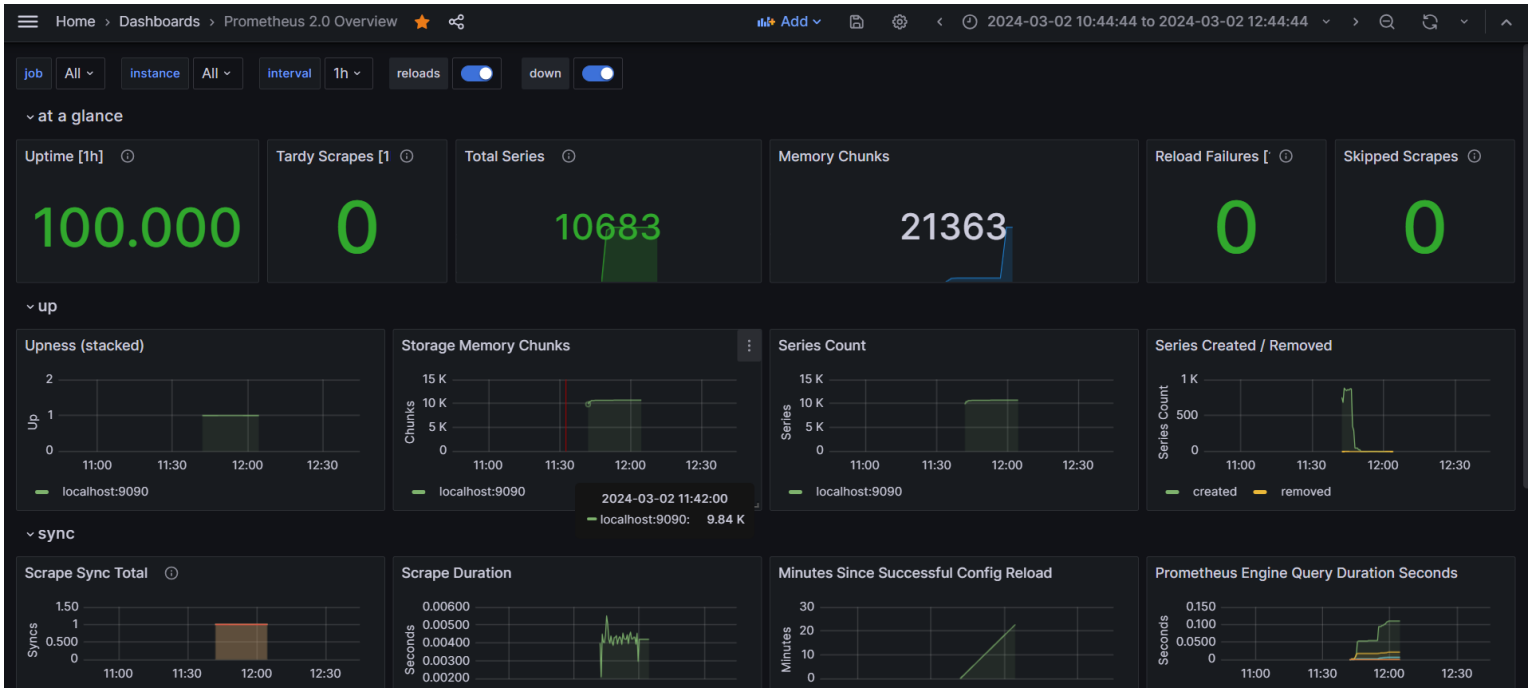


Jenkins metrics visualization

Windows Server metrics visualization

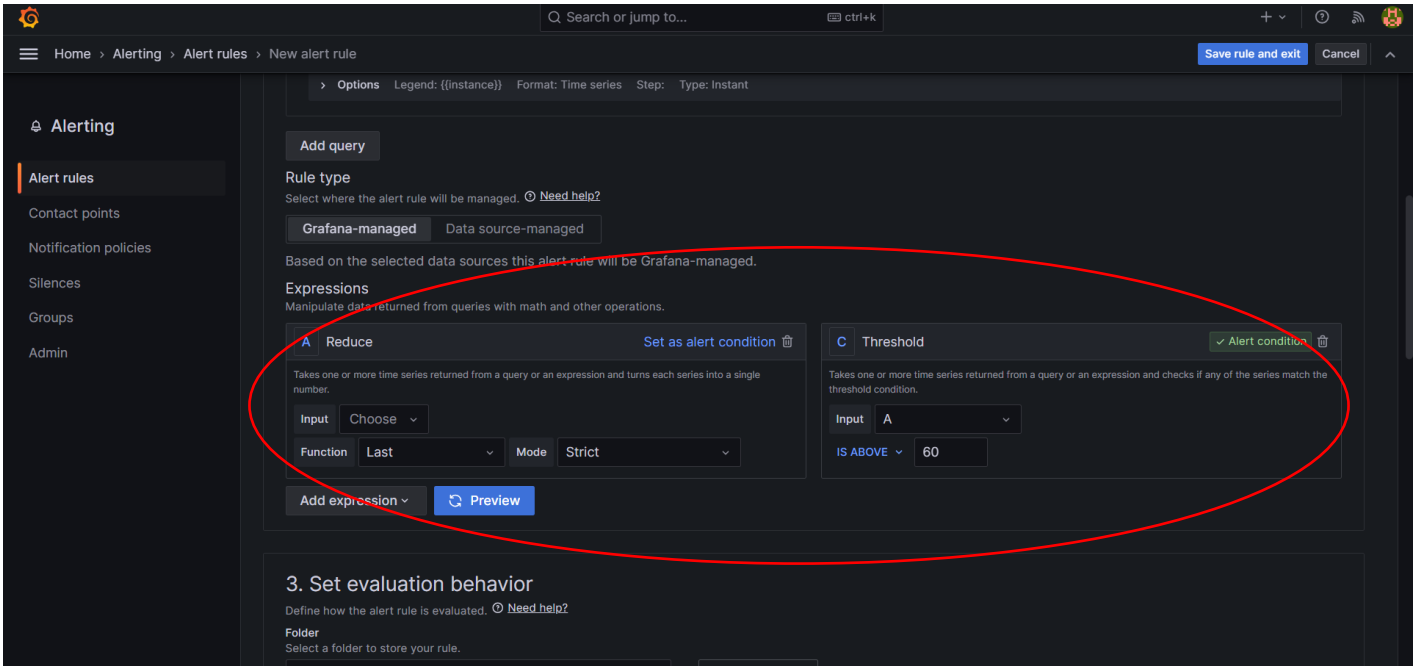


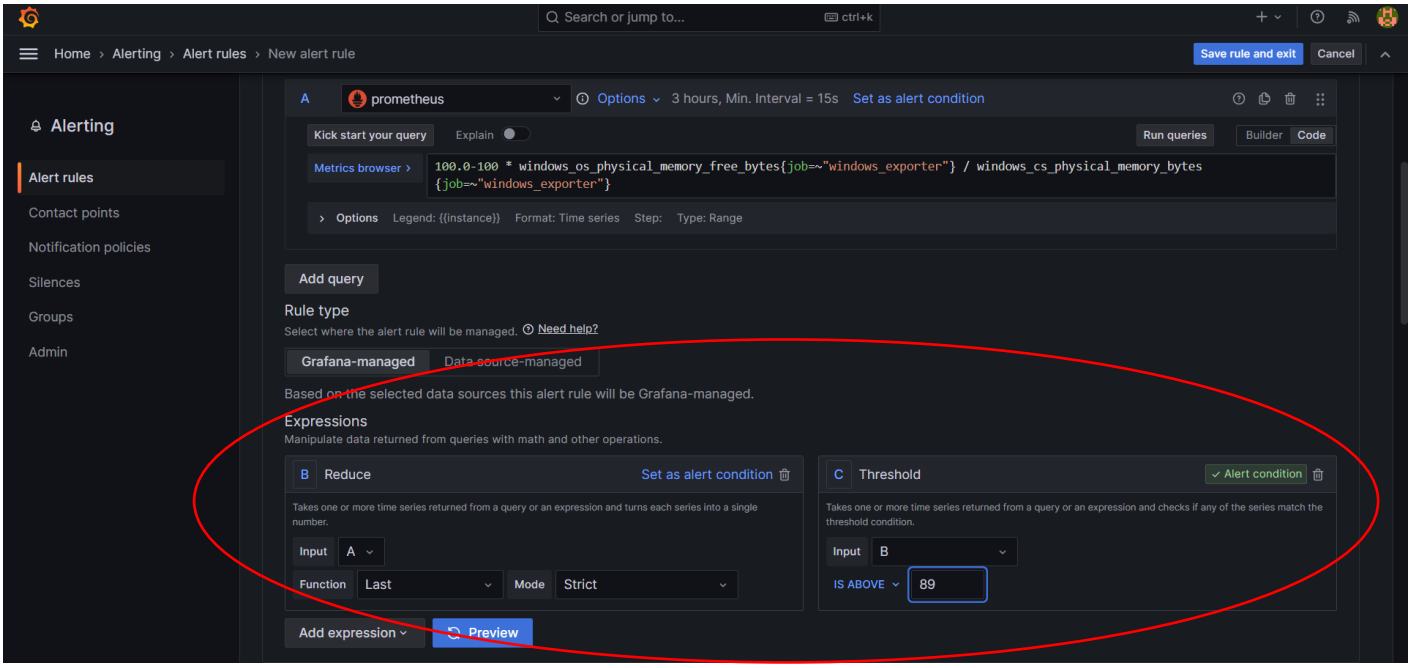
Prometheus server metrics visualization



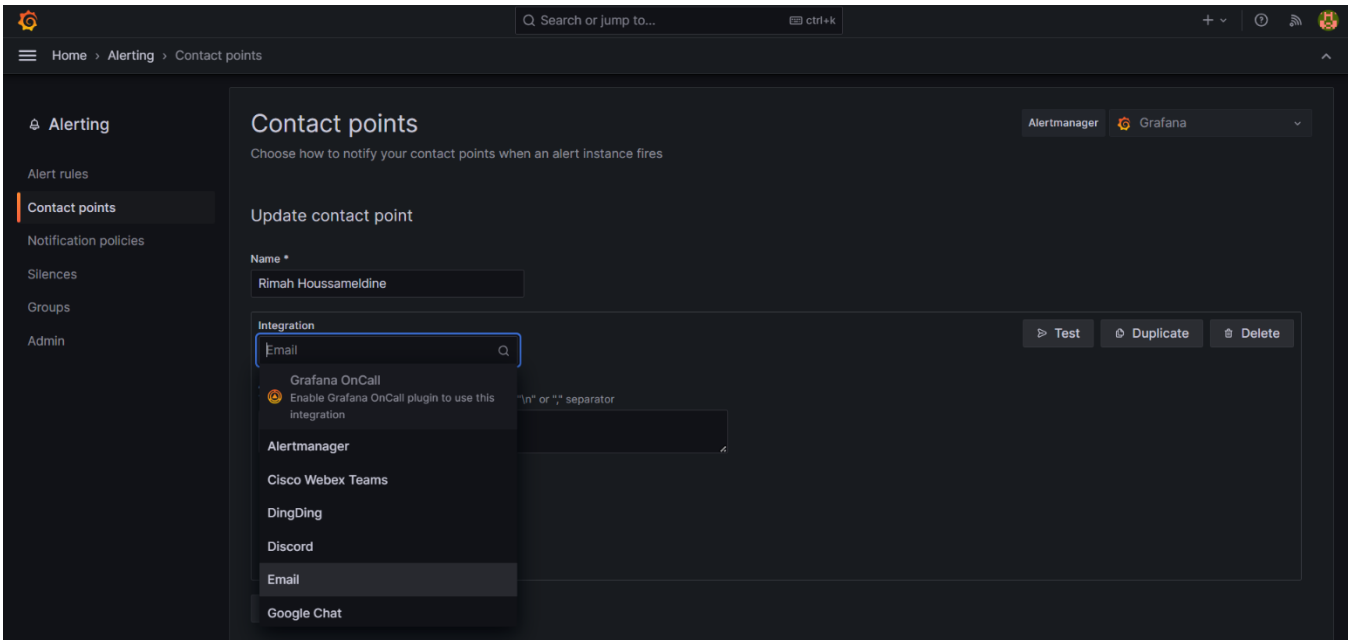
Alerting Setup

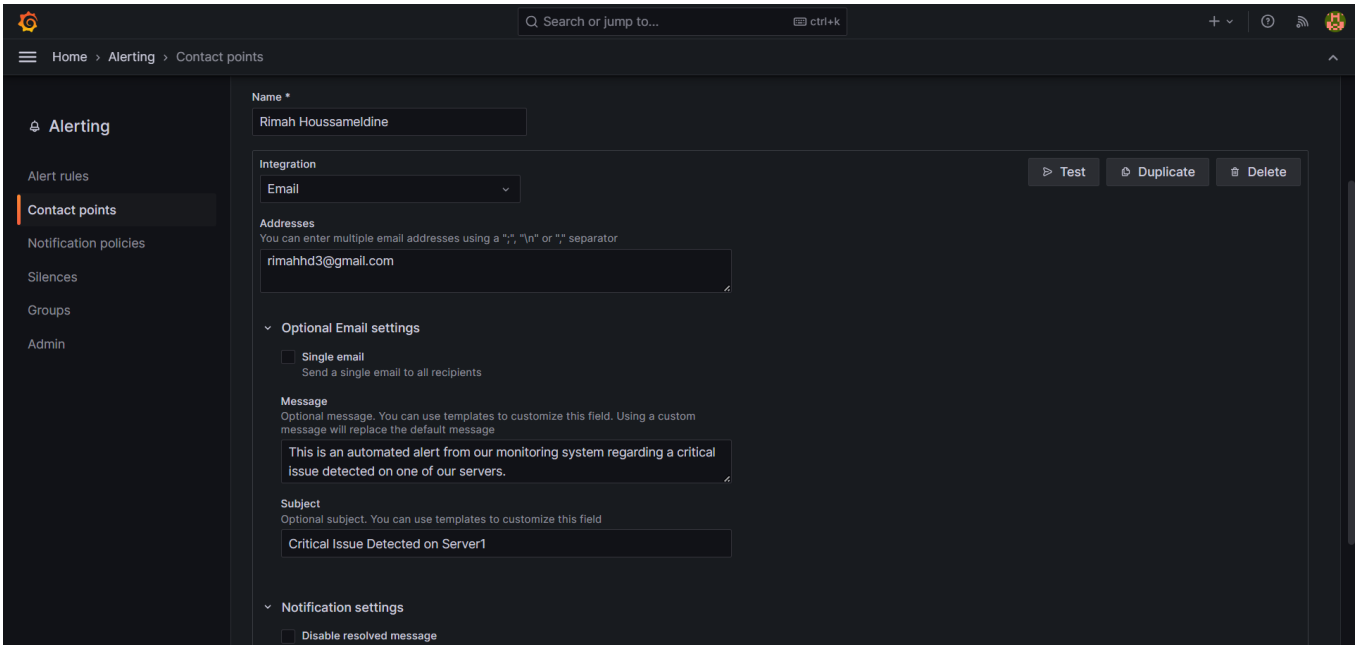
Set up alerting rules in Prometheus for critical metrics.



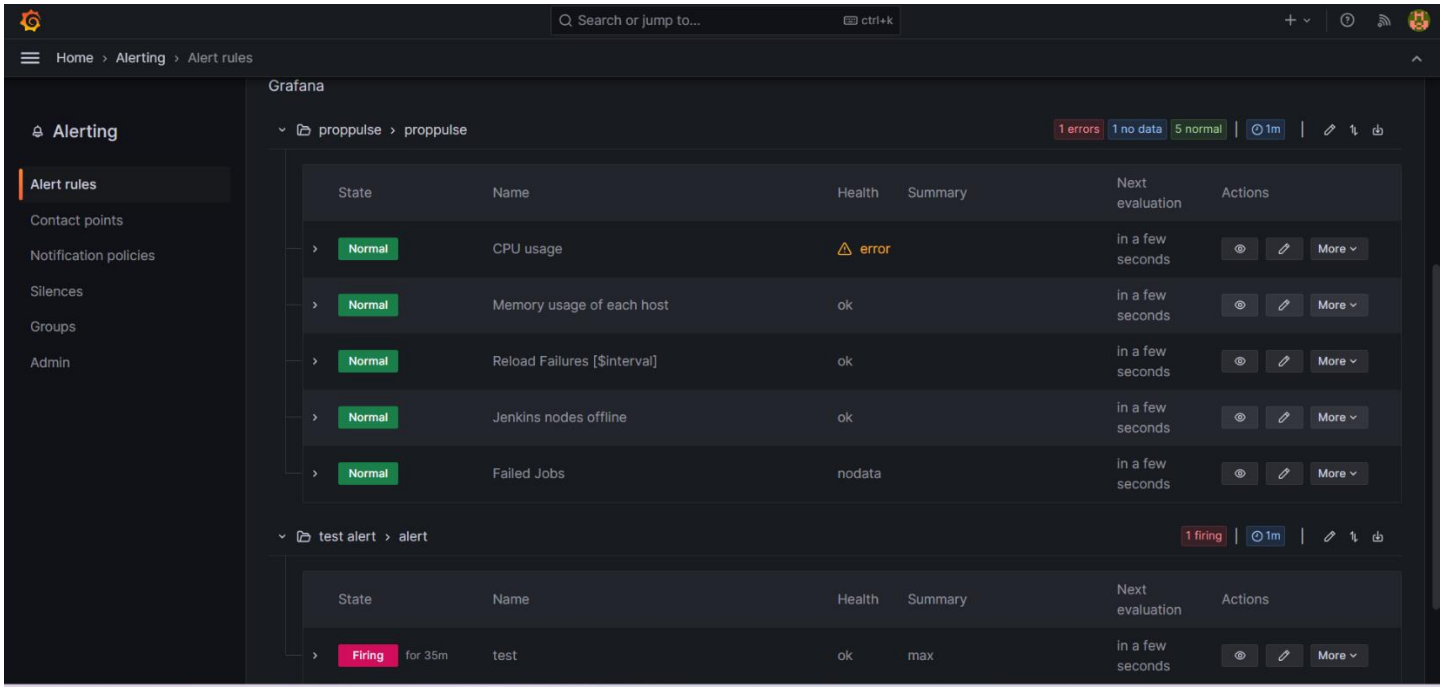


Configure alert notification channels in Grafana (email, Slack, etc.).





Test alerting setup to ensure proper functioning.



Best Practices

Regularly review and optimize dashboards and alerting rules.
Ensure monitoring covers key performance indicators (KPIs) for your application.
Implement automated monitoring checks as part of your CI/CD pipeline.

Conclusion

Monitoring your CI/CD PHP web application with Grafana and Prometheus enhances visibility into your deployment pipeline, enabling proactive issue detection and resolution. Follow the steps outlined in this documentation to establish a robust monitoring infrastructure for your application.