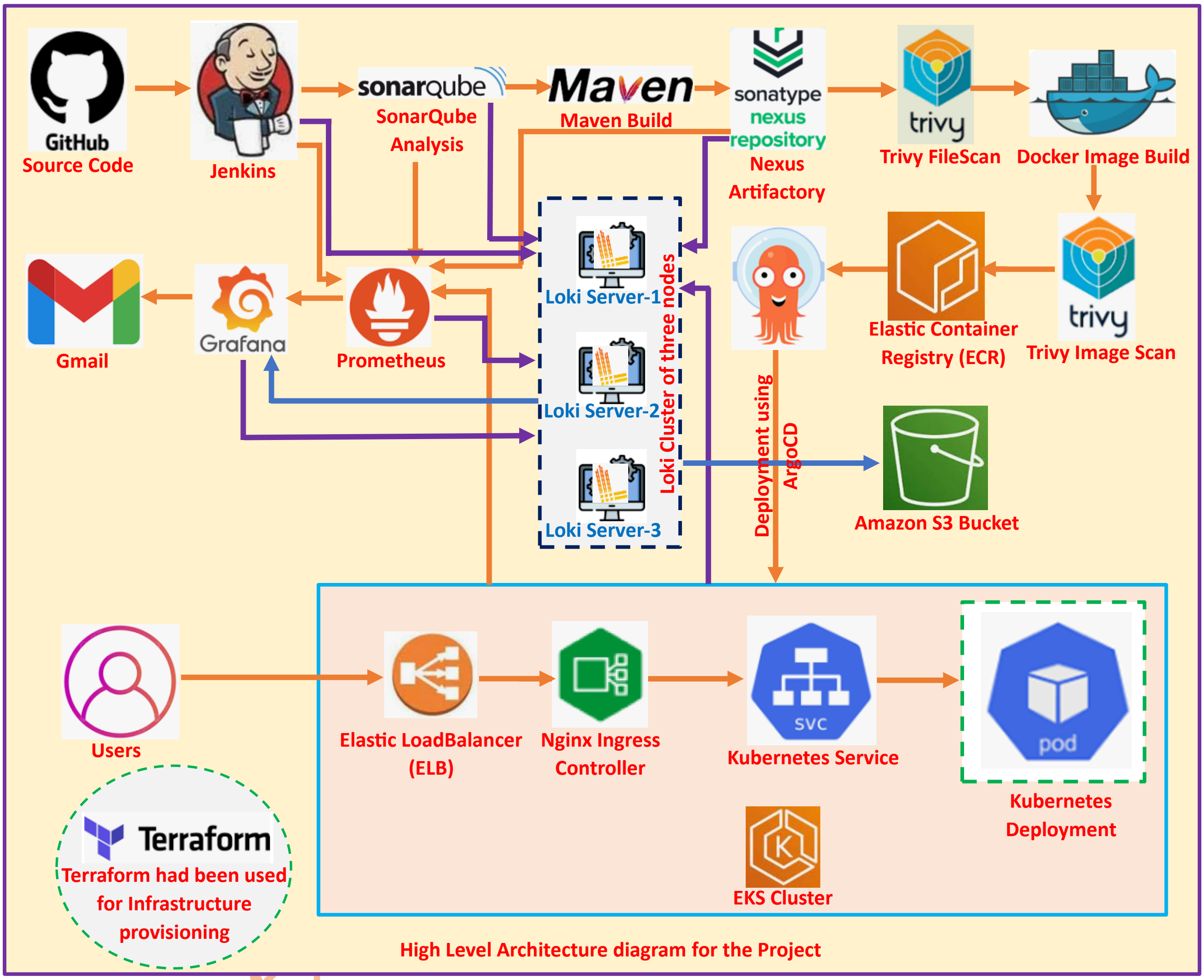
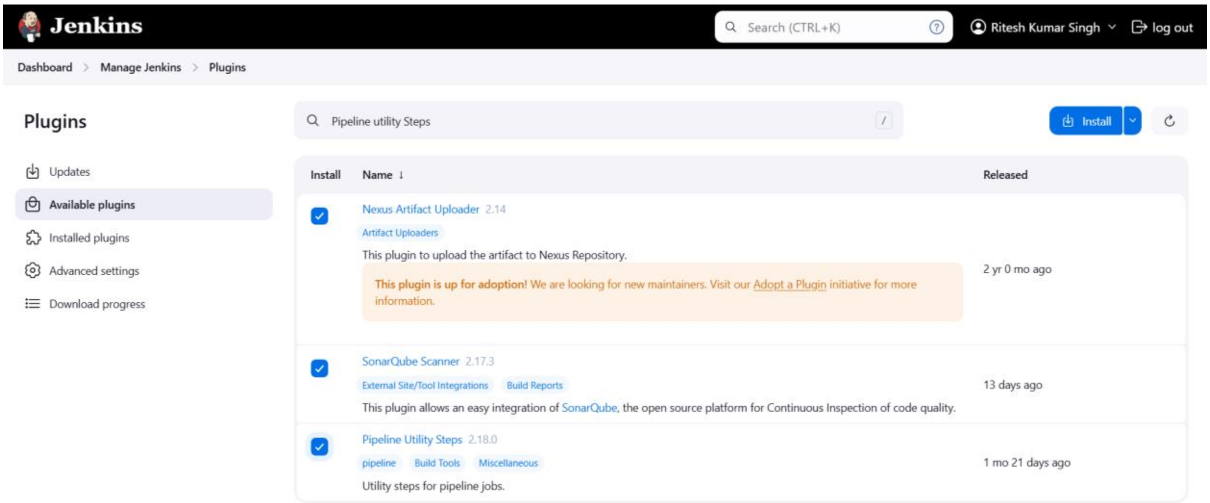
DevOps Project Blogging App Deployment Monitoring using Prometheus and Grafana and Log Aggregation using Loki, Promtail and Grafana



High Level Architecture diagram for the Project

This DevOps Project deals with creation of Infrastructure using Terraform and setup of CICD Pipeline using Jenkins, Monitoring using Prometheus and Grafana and Log Aggregation using Loki, Promtail and Grafana. SonarQube was used for Code-Analysis and Maven was used as the Build Tool. Nexus Artifactory was used to keep the Artifacts as shown in the Architecture diagram above. Trivy was used for FileScan and Docker Image Scan. The Docker Image was kept in the Elastic Container Registry (ECR) and which was deployed to EKS Cluster using the ArgoCD as shown in the high-level architecture diagram above. User was able to access the Application through the Ingress and hence the Kubernetes Service. For this project the source code was

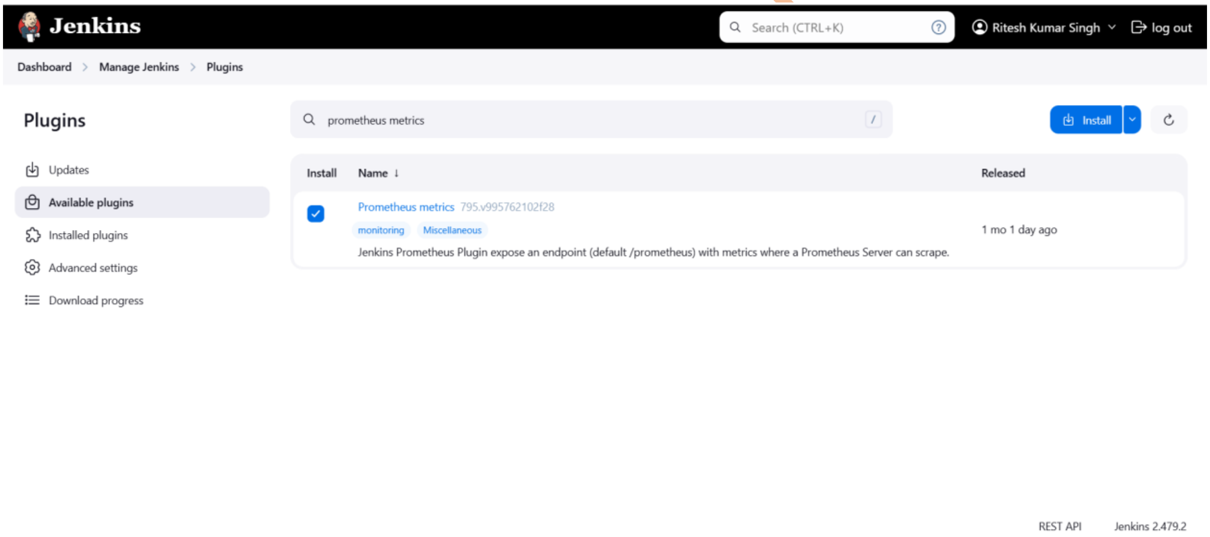
For this project I had installed Nexus Artifact Uploader, SonarQube Scanner and Pipeline Utility Step Plugin in the Jenkins as shown in the screenshot attached below.



Pipeline utility Steps

Install	Name ↓	Released
☑	Nexus Artifact Uploader 2.14 Artifact Uploaders This plugin to upload the artifact to Nexus Repository. **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	2 yr 0 mo ago
☑	SonarQube Scanner 2.17.3 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	13 days ago
☑	Pipeline Utility Steps 2.18.0 pipeline Build Tools Miscellaneous Utility steps for pipeline jobs.	1 mo 21 days ago

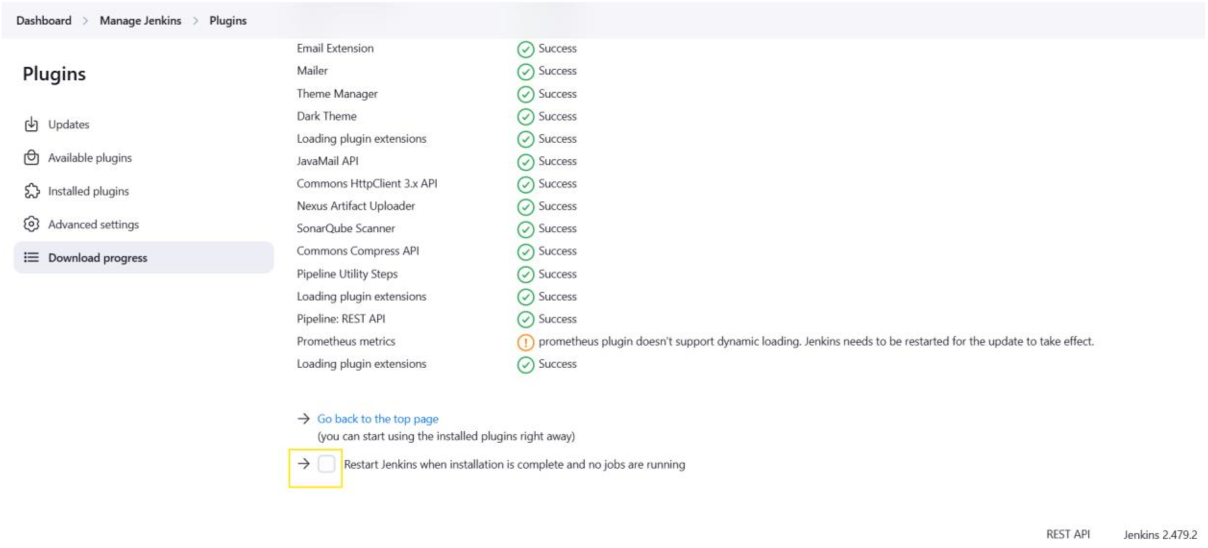
I had installed Prometheus metrics plugin in the Jenkins to monitor Jenkins Job through Prometheus and Grafana as shown in the screenshot attached below.



prometheus metrics

Install	Name ↓	Released
☑	Prometheus metrics 795.v995762102f28 monitoring Miscellaneous Jenkins Prometheus Plugin expose an endpoint (default /prometheus) with metrics where a Prometheus Server can scrape.	1 mo 1 day ago

After Installation of Prometheus metrics as explained above in Jenkins I restarted the Jenkins as shown in the screenshot attached below.



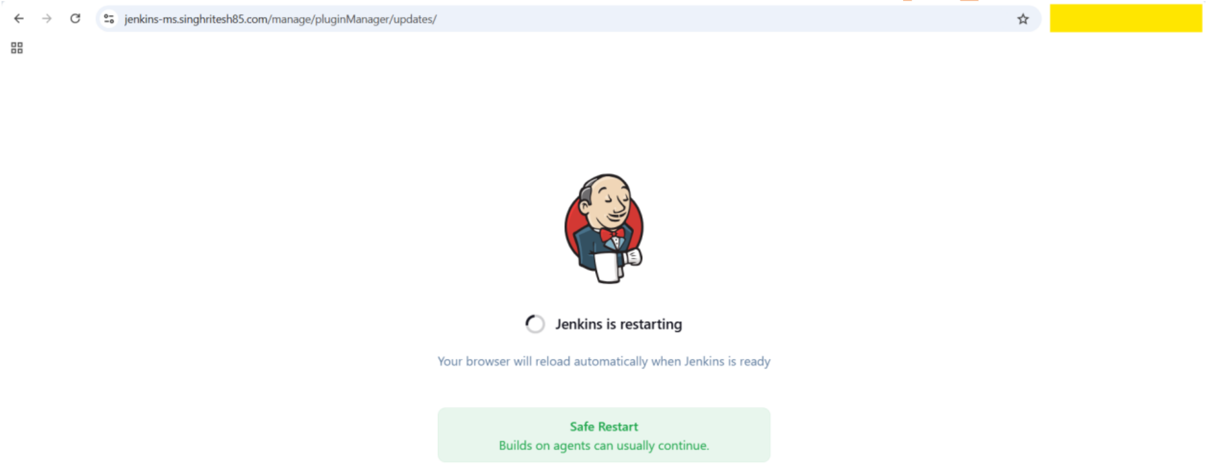
Dashboard > Manage Jenkins > Plugins

Plugins

Email Extension	Success
Mailer	Success
Theme Manager	Success
Dark Theme	Success
Loading plugin extensions	Success
JavaMail API	Success
Commons HttpClient 3.x API	Success
Nexus Artifact Uploader	Success
SonarQube Scanner	Success
Commons Compress API	Success
Pipeline Utility Steps	Success
Loading plugin extensions	Success
Pipeline: REST API	Success
Prometheus metrics	prometheus plugin doesn't support dynamic loading. Jenkins needs to be restarted for the update to take effect.
Loading plugin extensions	Success

→ Go back to the top page
(you can start using the installed plugins right away)

→ □ Restart Jenkins when installation is complete and no jobs are running



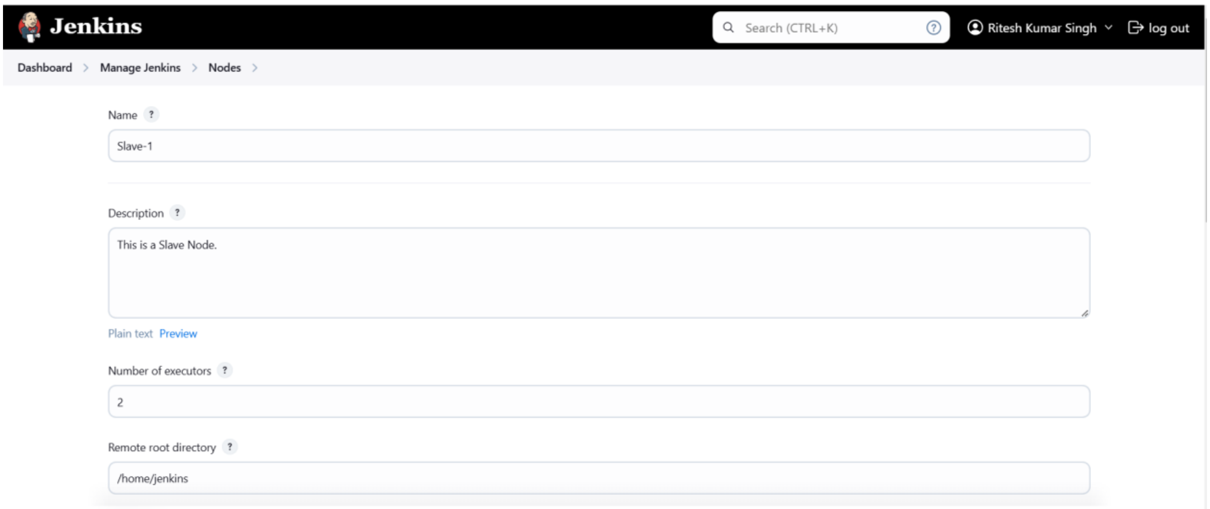
jenkins-ms.singhritesh85.com/manage/pluginManager/updates/

Jenkins is restarting

Your browser will reload automatically when Jenkins is ready

Safe Restart
Builds on agents can usually continue.

To Add Jenkins Slave with the Jenkins Master, follow the steps as written here. Go to **Manage Jenkins > Nodes**. Then provide the further details as mentioned in the screenshot attached below.

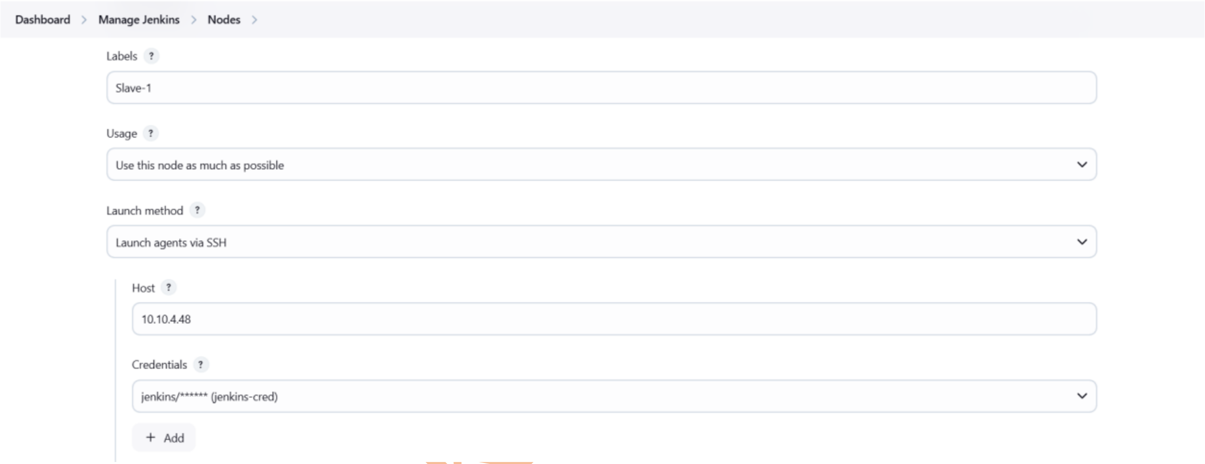


Name: Slave-1

Description: This is a Slave Node.

Number of executors: 2

Remote root directory: /home/jenkins



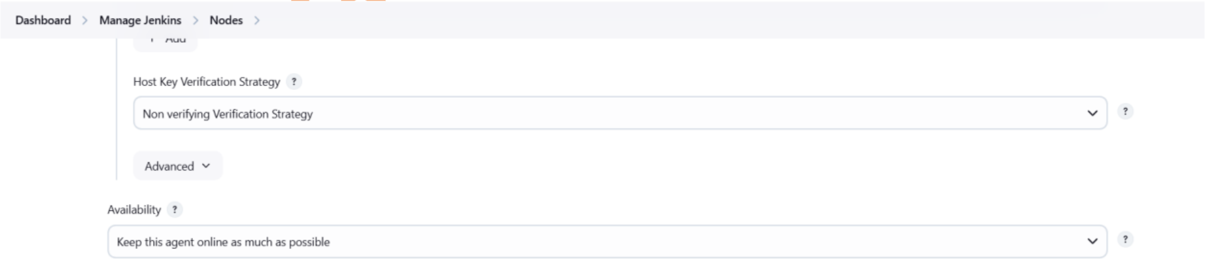
Labels: Slave-1

Usage: Use this node as much as possible

Launch method: Launch agents via SSH

Host: 10.10.4.48

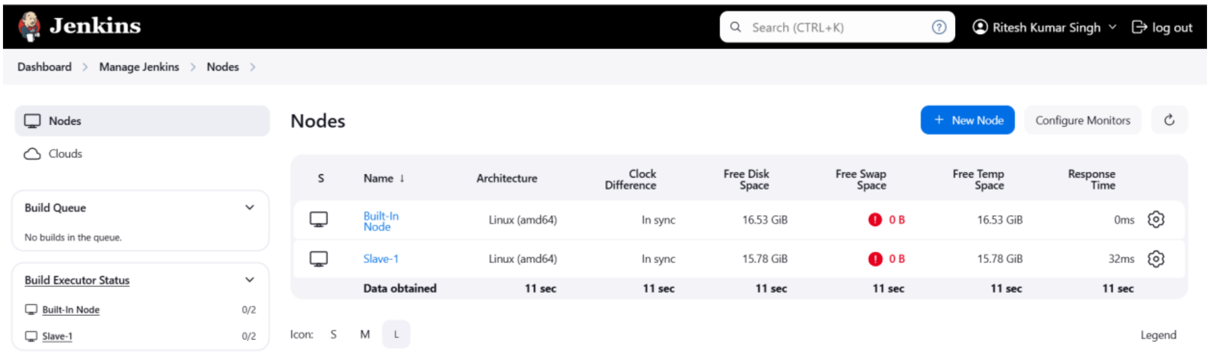
Credentials: jenkins/****** (jenkins-cred)



Host Key Verification Strategy: Non verifying Verification Strategy

Availability: Keep this agent online as much as possible

Finally, the Jenkins Slave was added to the Jenkins Master as shown in the screenshot attached below.



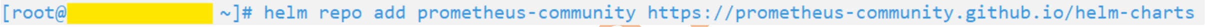
S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	16.53 GiB	0 B	16.53 GiB	0ms
	Slave-1	Linux (amd64)	In sync	15.78 GiB	0 B	15.78 GiB	32ms
	Data obtained	11 sec	11 sec	11 sec	11 sec	11 sec	11 sec

Installation of node-exporter and promtail had been done using the helm chart in the EKS Cluster as written below.

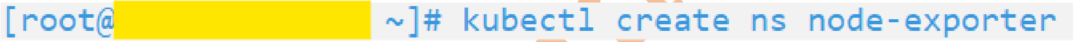
helm repo add prometheus-community https:*//prometheus-community.github.io/helm-charts*

kubectl create ns node-exporter

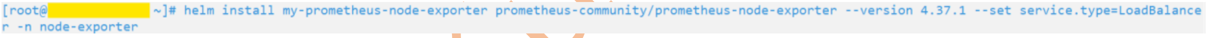
helm install my-prometheus-node-exporter prometheus-community/prometheus-node-exporter --version 4.37.1 --set service.type=LoadBalancer -n node-exporter



```
[root@[redacted] ~]# helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
```

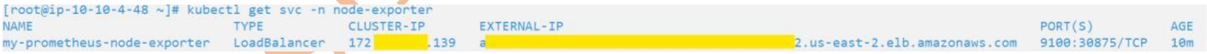


```
[root@[redacted] ~]# kubectl create ns node-exporter
```



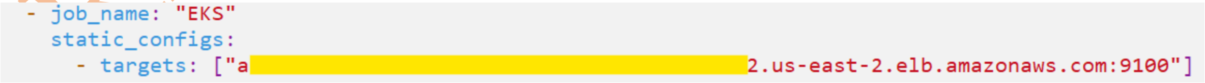
```
[root@[redacted] ~]# helm install my-prometheus-node-exporter prometheus-community/prometheus-node-exporter --version 4.37.1 --set service.type=LoadBalance
r -n node-exporter
```

Below screenshot shows the Kubernetes Service which was created for node-exporter using the helm chart.



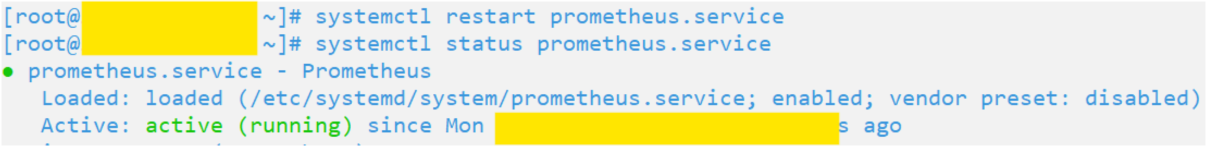
```
[root@ip-10-10-4-48 ~]# kubectl get svc -n node-exporter
NAME                          TYPE           CLUSTER-IP       EXTERNAL-IP                                                               PORT(S)          AGE
my-prometheus-node-exporter   LoadBalancer   172[redacted].139   a[redacted]2.us-east-2.elb.amazonaws.com   9100:30875/TCP   10m
```

I had updated the prometheus configuration file, **/etc/prometheus/prometheus.yml** as shown in the screenshot attached below.



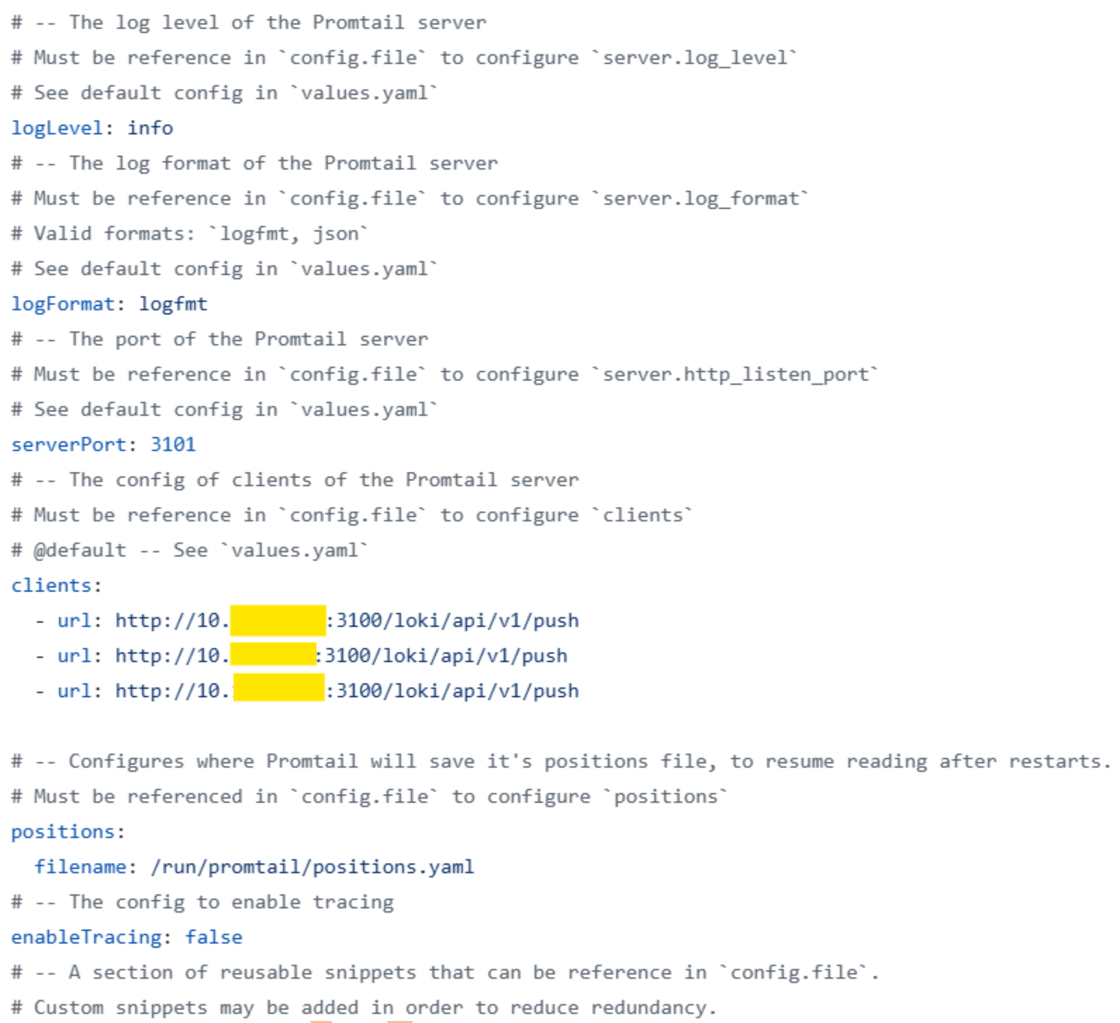
```
  - job_name: "EKS"
    static_configs:
      - targets: ["a[redacted]2.us-east-2.elb.amazonaws.com:9100"]
```

After updating the prometheus configuration file I restarted the prometheus service and checked the prometheus service status as shown in the screenshot attached below.



```
[root@[redacted] ~]# systemctl restart prometheus.service
[root@[redacted] ~]# systemctl status prometheus.service
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon [redacted]s ago
```

I installed the promtail using the helm chart as shown in the screenshot attached below. After cloning helm chart from GitHub, I updated the values.yaml file of promtail helm chart with Loki Servers Private IP Addresses as shown in the screenshot attached below.

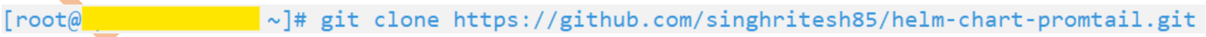


```
# -- The log level of the Promtail server
# Must be reference in `config.file` to configure `server.log_level`
# See default config in `values.yaml`
logLevel: info
# -- The log format of the Promtail server
# Must be reference in `config.file` to configure `server.log_format`
# Valid formats: `logfmt, json`
# See default config in `values.yaml`
logFormat: logfmt
# -- The port of the Promtail server
# Must be reference in `config.file` to configure `server.http_listen_port`
# See default config in `values.yaml`
serverPort: 3101
# -- The config of clients of the Promtail server
# Must be reference in `config.file` to configure `clients`
# @default -- See `values.yaml`
clients:
  - url: http://10.[redacted]:3100/loki/api/v1/push
  - url: http://10.[redacted]:3100/loki/api/v1/push
  - url: http://10.[redacted]:3100/loki/api/v1/push

# -- Configures where Promtail will save it's positions file, to resume reading after restarts.
# Must be referenced in `config.file` to configure `positions`
positions:
  filename: /run/promtail/positions.yaml
# -- The config to enable tracing
enableTracing: false
# -- A section of reusable snippets that can be reference in `config.file`.
# Custom snippets may be added in order to reduce redundancy.
```

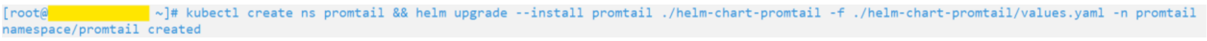
git clone https://github.com/singhritesh85/helm-chart-promtail.git

kubectl create ns promtail && helm upgrade --install promtail ./helm-chart-promtail -f ./helm-chart-promtail/values.yaml -n promtail

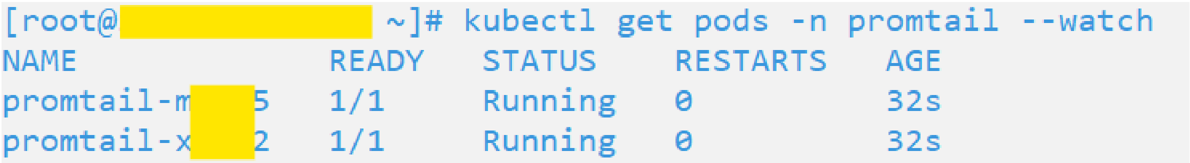
kubectl get pods -n promtail --watch



```
[root@[redacted] ~]# git clone https://github.com/singhritesh85/helm-chart-promtail.git
```

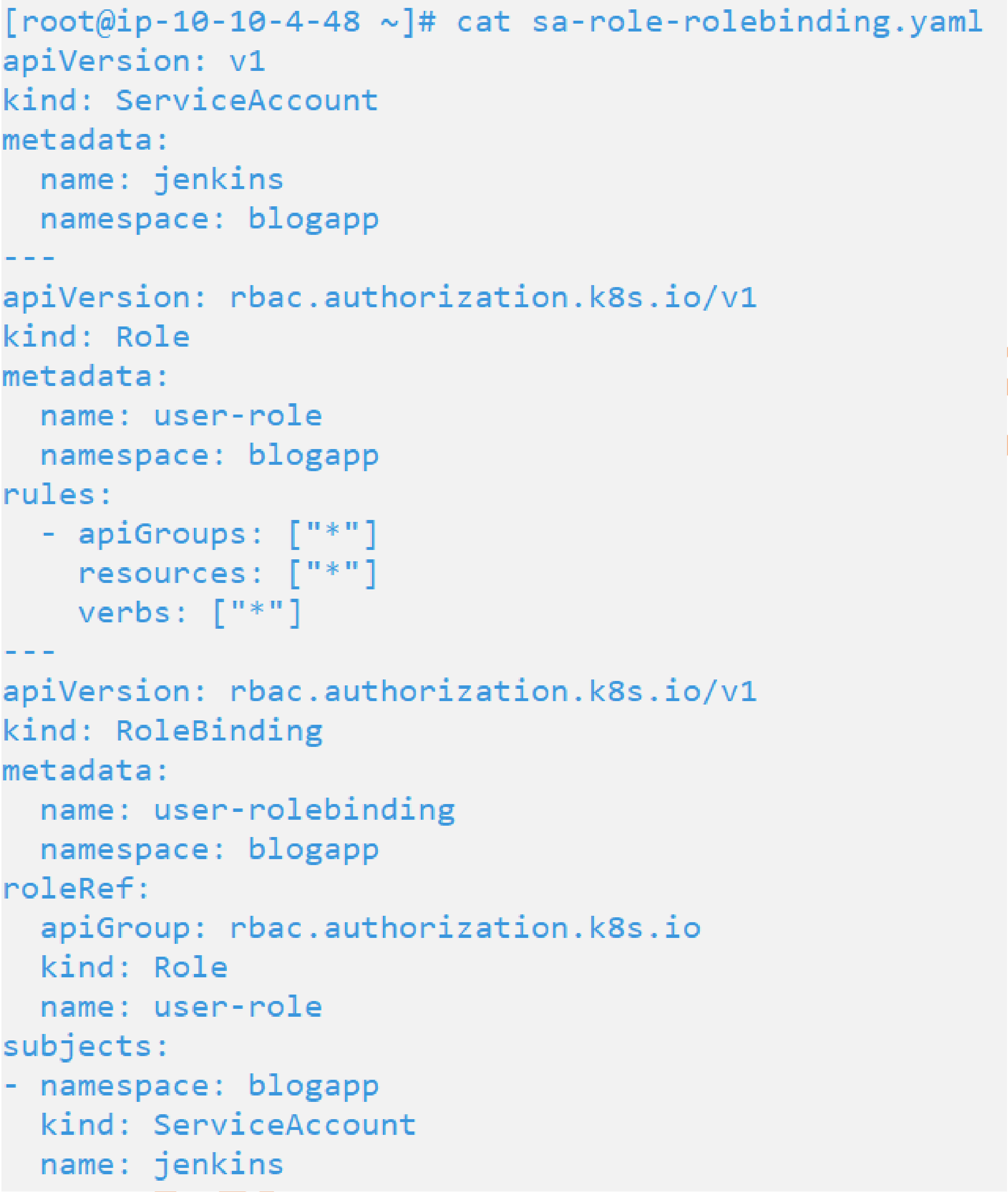


```
[root@[redacted] ~]# kubectl create ns promtail && helm upgrade --install promtail ./helm-chart-promtail -f ./helm-chart-promtail/values.yaml -n promtail
namespace/promtail created
```

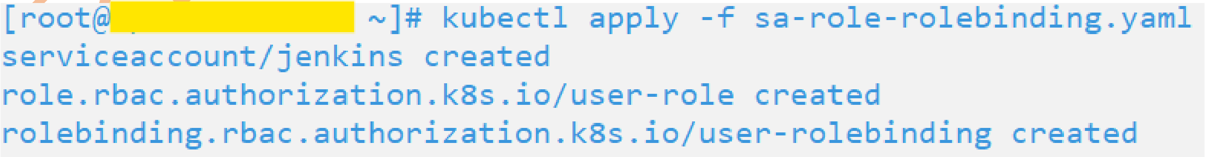


```
[root@[redacted] ~]# kubectl get pods -n promtail --watch
NAME              READY   STATUS    RESTARTS   AGE
promtail-m[redacted]5   1/1     Running   0          32s
promtail-x[redacted]2   1/1     Running   0          32s
```

I had provided restricted access to the deployment user **jenkins** using Service Account, Role and Role Binding as shown in the screenshot attached below. The deployment user had all the accesses in the namespace **blogapp** but does not had access for the entire EKS cluster. That means deployment user jenkins access was restricted to the namespace **blogapp** in the EKS Cluster.

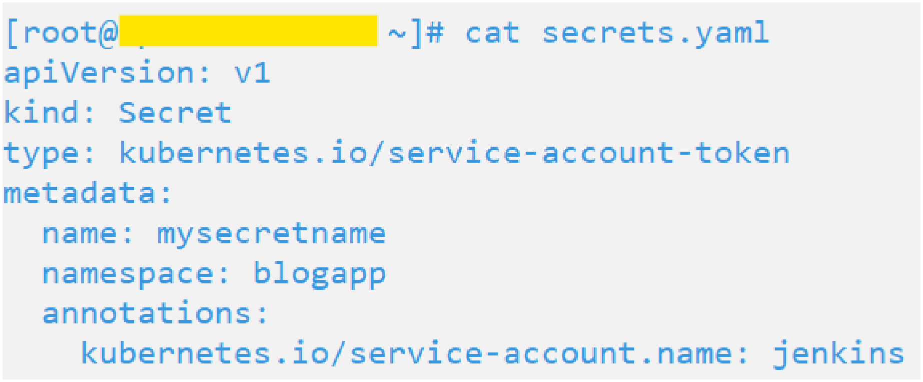


```
[root@ip-10-10-4-48 ~]# cat sa-role-rolebinding.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: jenkins
  namespace: blogapp
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: user-role
  namespace: blogapp
rules:
  - apiGroups: ["*"]
    resources: ["*"]
    verbs: ["*"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: user-rolebinding
  namespace: blogapp
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: user-role
subjects:
- namespace: blogapp
  kind: ServiceAccount
  name: jenkins
```

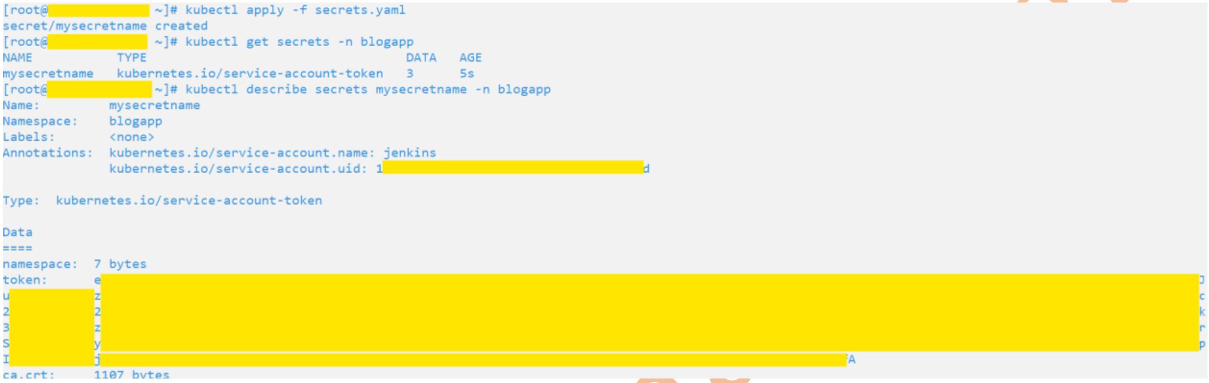


```
[root@[redacted] ~]# kubectl apply -f sa-role-rolebinding.yaml
serviceaccount/jenkins created
role.rbac.authorization.k8s.io/user-role created
rolebinding.rbac.authorization.k8s.io/user-rolebinding created
```

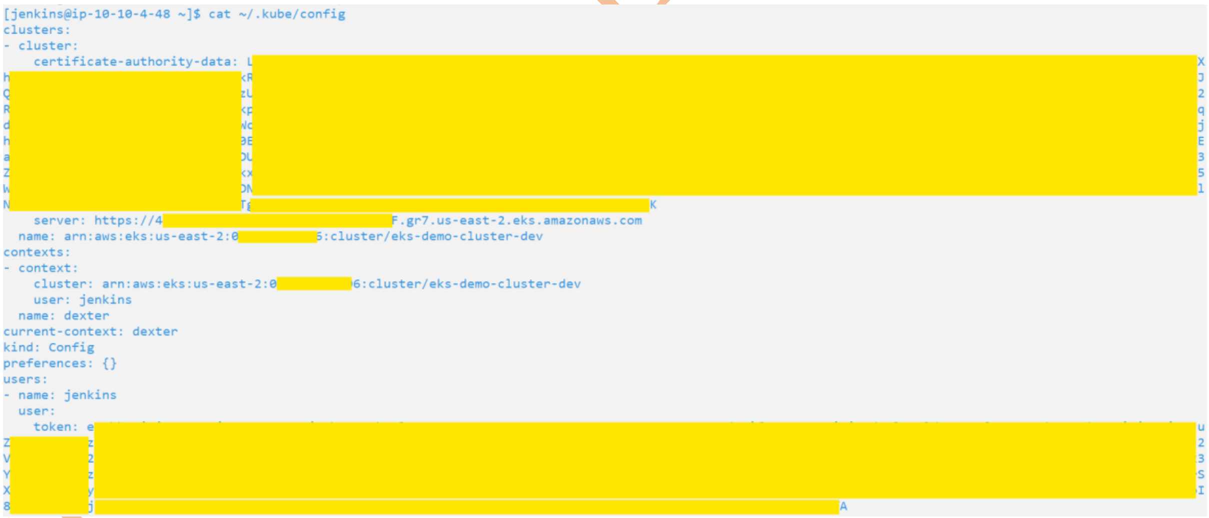
Created Kubernetes Secrets Which Token was utilized in the kubeconfig file (which was shared with the deployment user jenkins) as shown in the screenshot attached below.



```
[root@[redacted] ~]# cat secrets.yaml
apiVersion: v1
kind: Secret
type: kubernetes.io/service-account-token
metadata:
  name: mysecretname
  namespace: blogapp
  annotations:
    kubernetes.io/service-account.name: jenkins
```

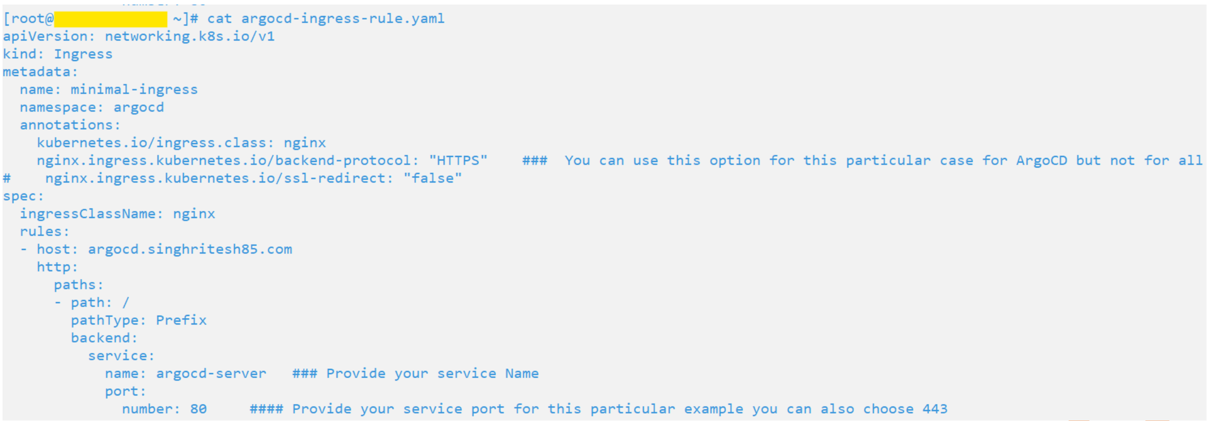


```
[root@[redacted] ~]# kubectl apply -f secrets.yaml
secret/mysecretname created
[root@[redacted] ~]# kubectl get secrets -n blogapp
NAME           TYPE                                  DATA   AGE
mysecretname   kubernetes.io/service-account-token   3      5s
[root@[redacted] ~]# kubectl describe secrets mysecretname -n blogapp
Name:         mysecretname
Namespace:    blogapp
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: jenkins
              kubernetes.io/service-account.uid: [redacted]

Type:  kubernetes.io/service-account-token

Data
====
namespace:  7 bytes
token:      [redacted]
ca.crt:     1107 bytes
```



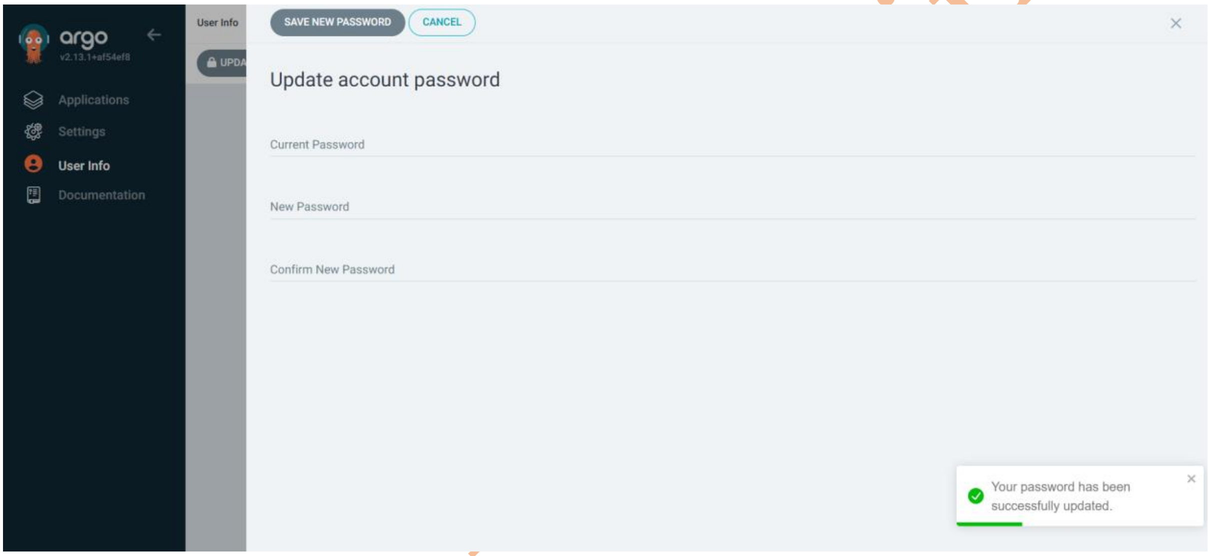
```
[jenkins@ip-10-10-4-48 ~]$ cat ~/.kube/config
clusters:
- cluster:
    certificate-authority-data: [redacted]
    server: https://4[redacted]F.gr7.us-east-2.eks.amazonaws.com
  name: arn:aws:eks:us-east-2:0[redacted]5:cluster/eks-demo-cluster-dev
contexts:
- context:
    cluster: arn:aws:eks:us-east-2:0[redacted]6:cluster/eks-demo-cluster-dev
    user: jenkins
  name: dexter
current-context: dexter
kind: Config
preferences: {}
users:
- name: jenkins
  user:
    token: [redacted]
```

Kubernetes Ingress for ArgoCD was creates using the Ingress Rule as shown in the screenshot attached below.

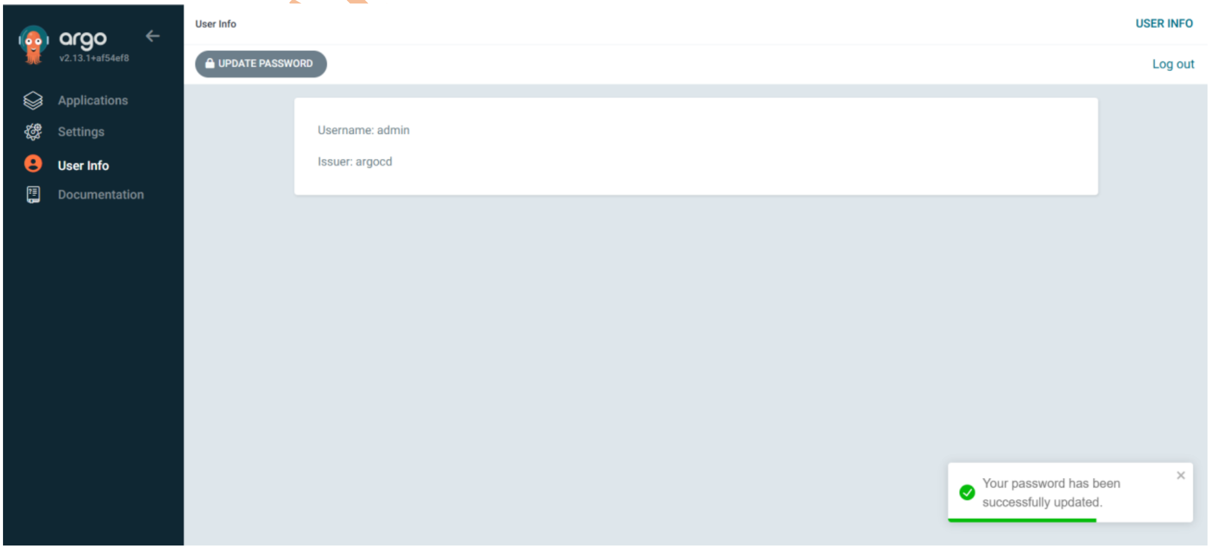


```
[root@[redacted] ~]# cat argocd-ingress-rule.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  namespace: argocd
  annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/backend-protocol: "HTTPS"    ###  You can use this option for this particular case for ArgoCD but not for all
#    nginx.ingress.kubernetes.io/ssl-redirect: "false"
spec:
  ingressClassName: nginx
  rules:
  - host: argocd.singhritesh85.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: argocd-server   ### Provide your service Name
            port:
              number: 80     #### Provide your service port for this particular example you can also choose 443
```

I had updated the ArgoCD password as shown in the screenshot attached below.



Your password has been successfully updated.

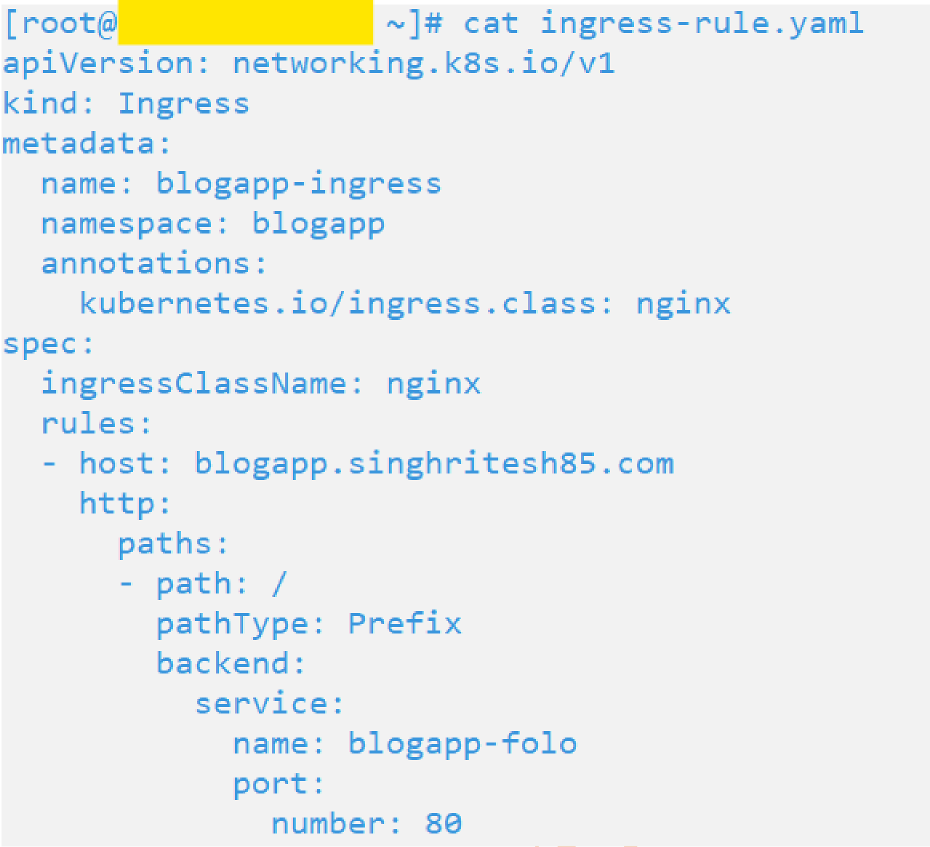


Username: admin

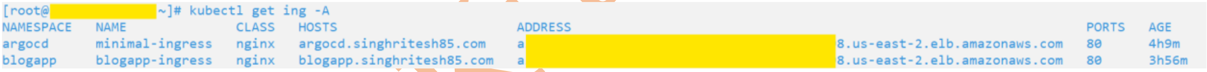
Issuer: argocd

Your password has been successfully updated.

Kubernetes Ingress for Blogging Application was created using the Ingress Rule as shown in the screenshot attached below.

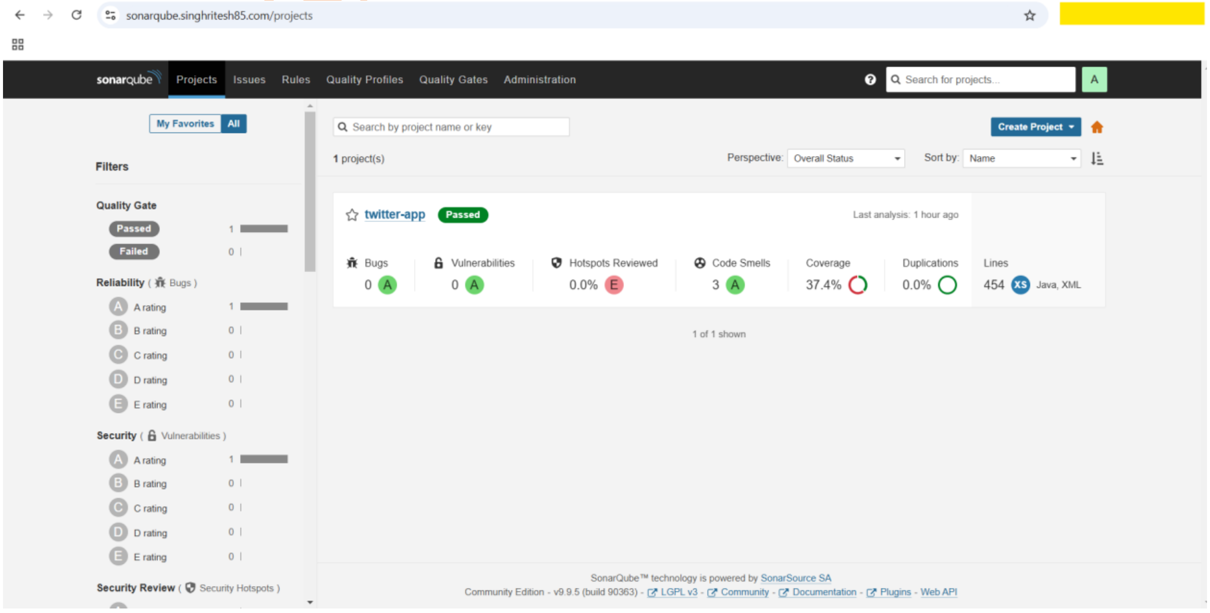


```
[root@[redacted] ~]# cat ingress-rule.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: blogapp-ingress
  namespace: blogapp
  annotations:
    kubernetes.io/ingress.class: nginx
spec:
  ingressClassName: nginx
  rules:
  - host: blogapp.singhritesh85.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: blogapp-folo
            port:
              number: 80
```



```
[root@[redacted]~]# kubectl get ing -A
NAMESPACE   NAME              CLASS   HOSTS                       ADDRESS                                                    PORTS   AGE
argocd      minimal-ingress   nginx   argocd.singhritesh85.com    a[redacted]8.us-east-2.elb.amazonaws.com    80      4h9m
blogapp     blogapp-ingress   nginx   blogapp.singhritesh85.com   a[redacted]8.us-east-2.elb.amazonaws.com    80      3h56m
```

The Screenshot for SonarQube, Nexus and ArgoCD after running the Jenkins Job is as shown in the screenshot attached below.



1 project(s)

Filters

Quality Gate
- Passed 1
- Failed 0

Reliability (Bugs)
- A rating 1
- B rating 0
- C rating 0
- D rating 0
- E rating 0

Security (Vulnerabilities)
- A rating 1
- B rating 0
- C rating 0
- D rating 0
- E rating 0

Security Review (Security Hotspots)

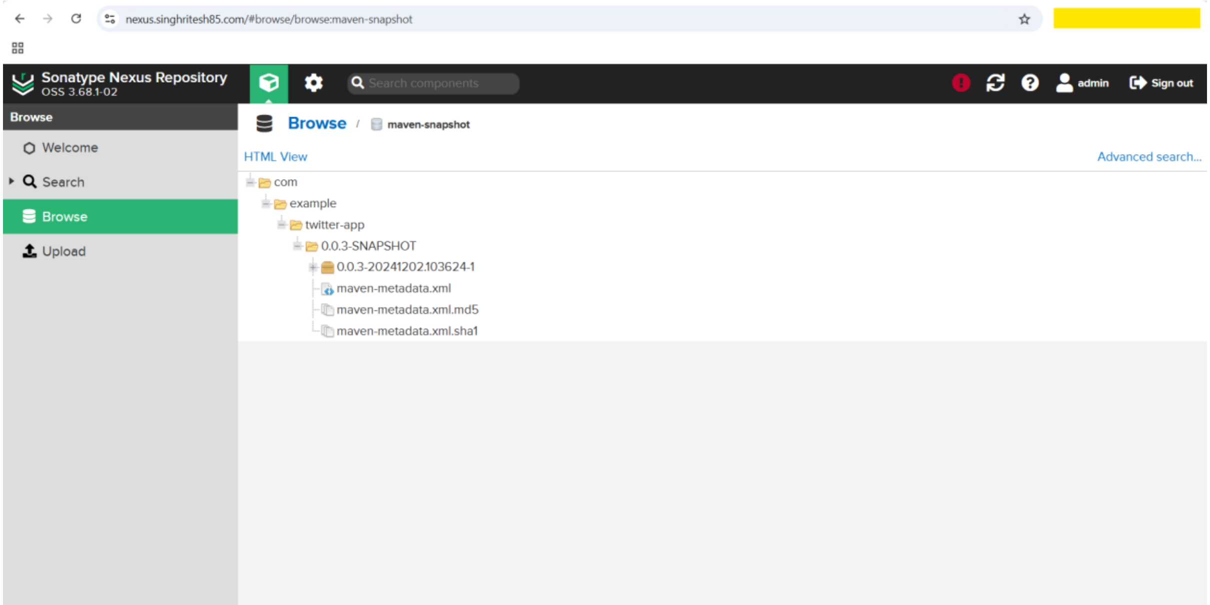
twitter-app Passed

Last analysis: 1 hour ago

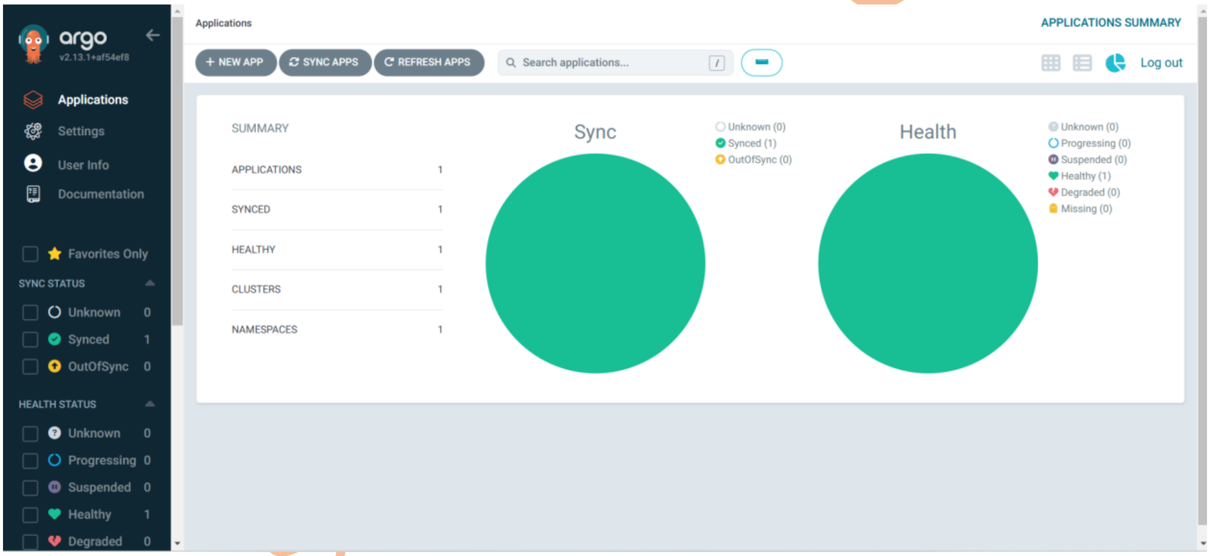
Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
0 A	0 A	0.0% E	3 A	37.4%	0.0%	454 XS Java, XML

1 of 1 shown

SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.9.5 (build 90363) - LGPL v3 - Community - Documentation - Plugins - Web API

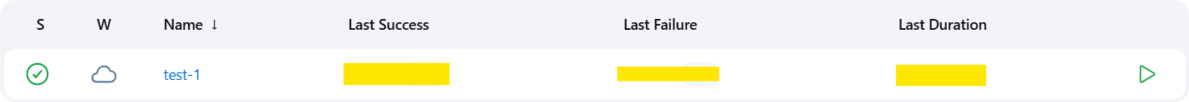


com
example
twitter-app
0.0.3-SNAPSHOT
0.0.3-20241202.103624-1
maven-metadata.xml
maven-metadata.xml.md5
maven-metadata.xml.sha1



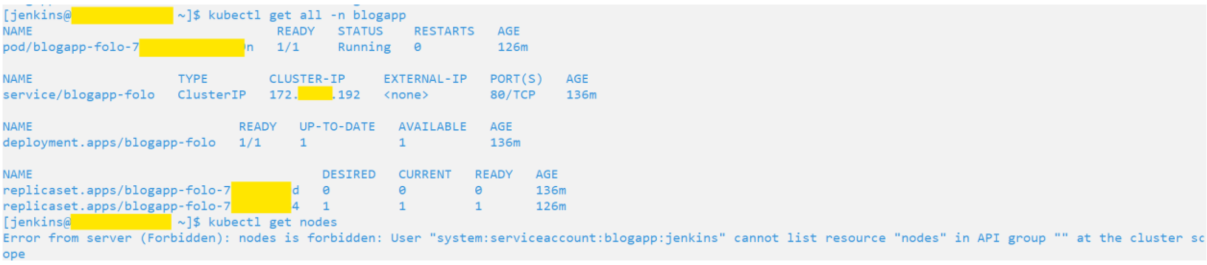
SUMMARY	
APPLICATIONS	1
SYNCED	1
HEALTHY	1
CLUSTERS	1
NAMESPACES	1

Below screenshot show Jenkins Job after its successful execution.



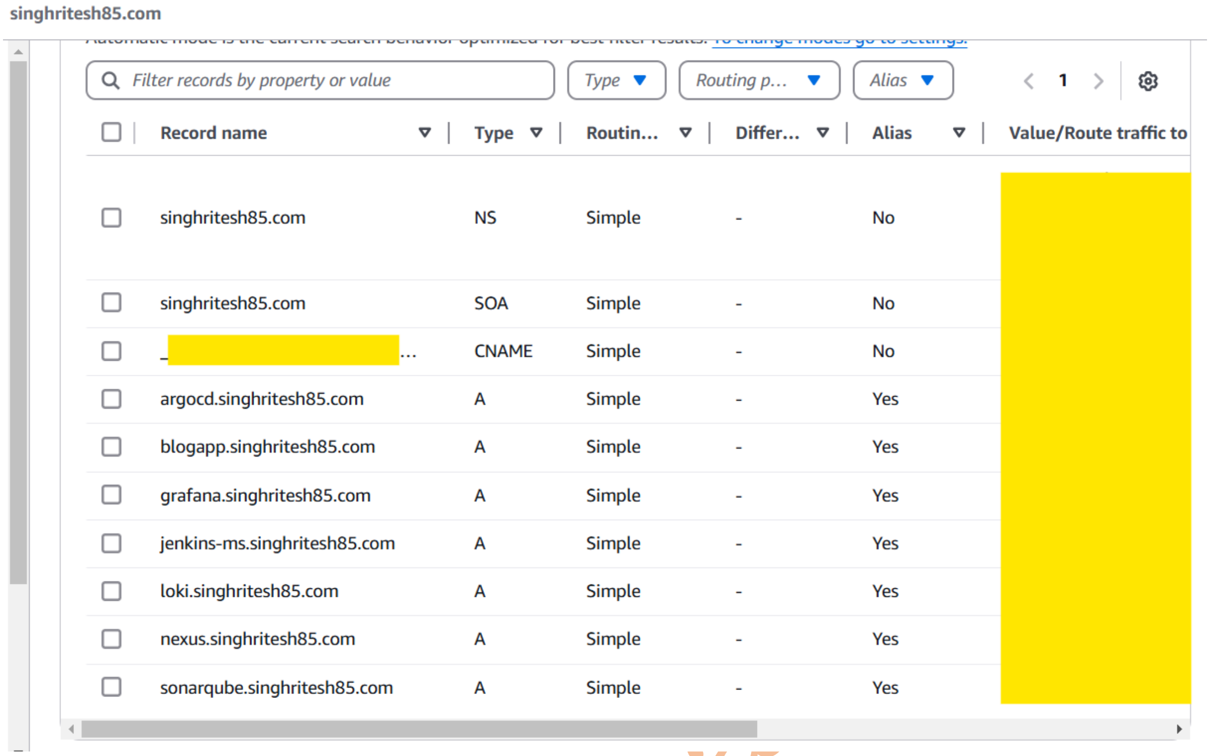
S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☁	test-1	[redacted]	[redacted]	[redacted]	▷

After successful execution of Jenkins Job Kubernetes Pod, Service and Deployment was created.



```
[jenkins@[redacted] ~]$ kubectl get all -n blogapp
NAME                                 READY   STATUS    RESTARTS   AGE
pod/blogapp-folo-7[redacted]n   1/1     Running   0          126m

NAME                   TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
service/blogapp-folo   ClusterIP   172.[redacted].192   <none>        80/TCP    136m

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/blogapp-folo   1/1     1            1           136m

NAME                                      DESIRED   CURRENT   READY   AGE
replicaset.apps/blogapp-folo-7[redacted]d   0         0         0       136m
replicaset.apps/blogapp-folo-7[redacted]4   1         1         1       126m
[jenkins@[redacted] ~]$ kubectl get nodes
Error from server (Forbidden): nodes is forbidden: User "system:serviceaccount:blogapp:jenkins" cannot list resource "nodes" in API group "" at the cluster sc
ope
```

Below screenshot shows the entry for Route53 to create the Record Set.

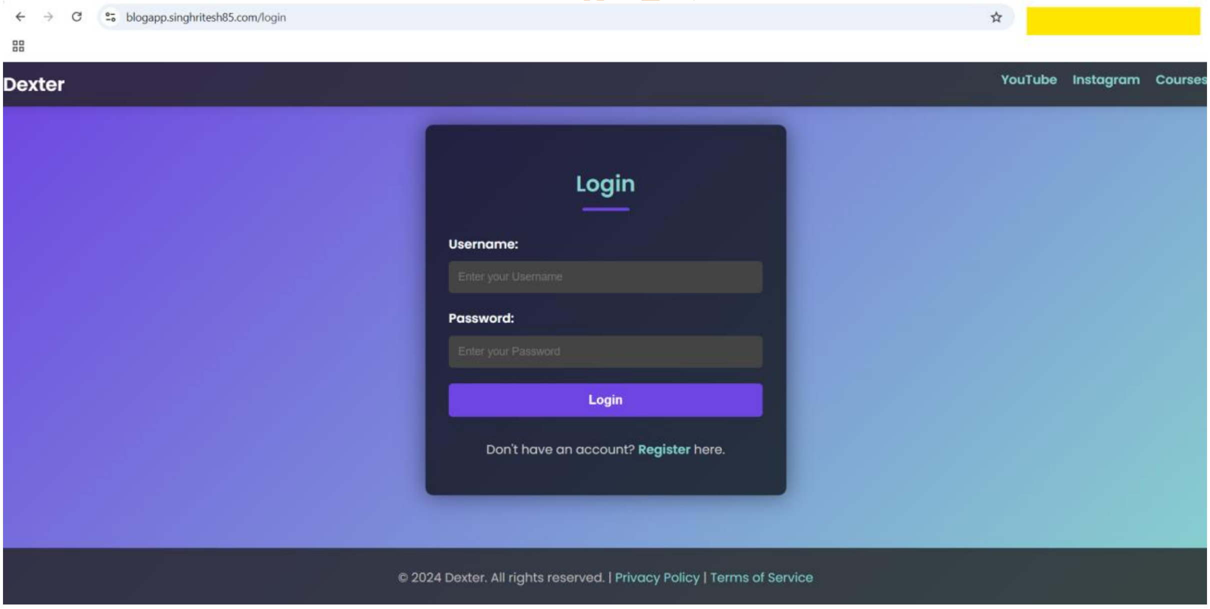


singhritesh85.com

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

□	Record name	Type	Routin...	Differ...	Alias	Value/Route traffic to
□	singhritesh85.com	NS	Simple	-	No	[redacted]
□	singhritesh85.com	SOA	Simple	-	No	[redacted]
□	_[redacted]...	CNAME	Simple	-	No	[redacted]
□	argocd.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	blogapp.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	grafana.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	jenkins-ms.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	loki.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	nexus.singhritesh85.com	A	Simple	-	Yes	[redacted]
□	sonarqube.singhritesh85.com	A	Simple	-	Yes	[redacted]

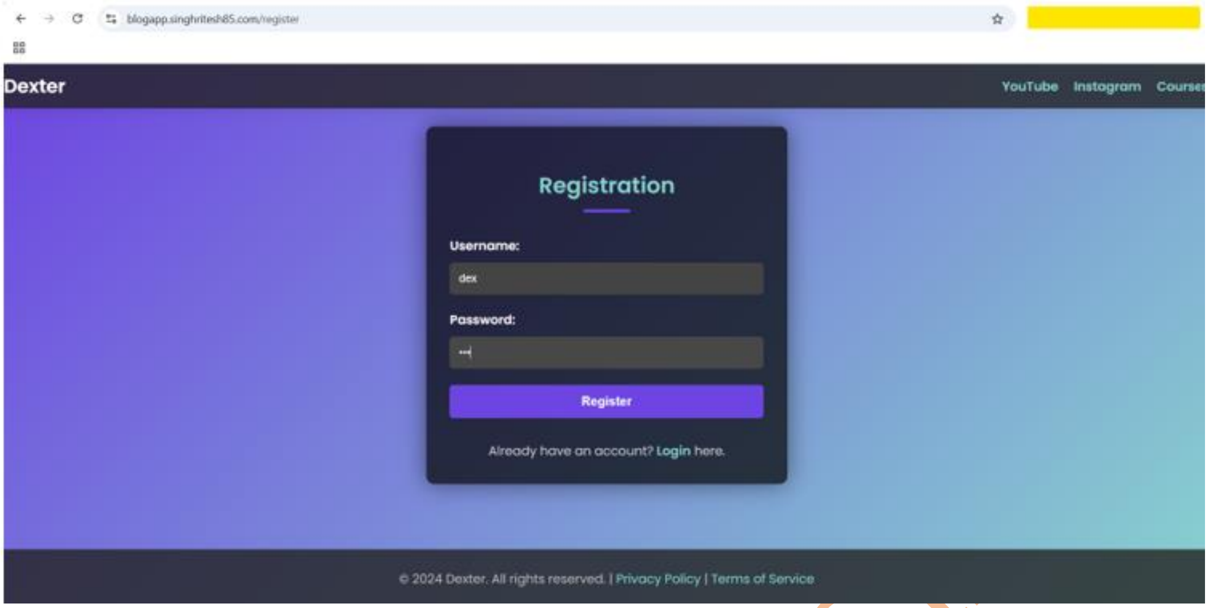
Finally, using the URL I started accessing the Application.



blogapp.singhritesh85.com/login [redacted]

Dexter
YouTube
Instagram
Courses
Login
Username:
Enter your Username
Password:
Enter your Password
Login
Don't have an account? Register here.
© 2024 Dexter. All rights reserved. | Privacy Policy | Terms of Service

I registered a new user and logged in through that user and checked that I was able to read the blogs or create a new blogs.



blogapp.singhritesh85.com/register

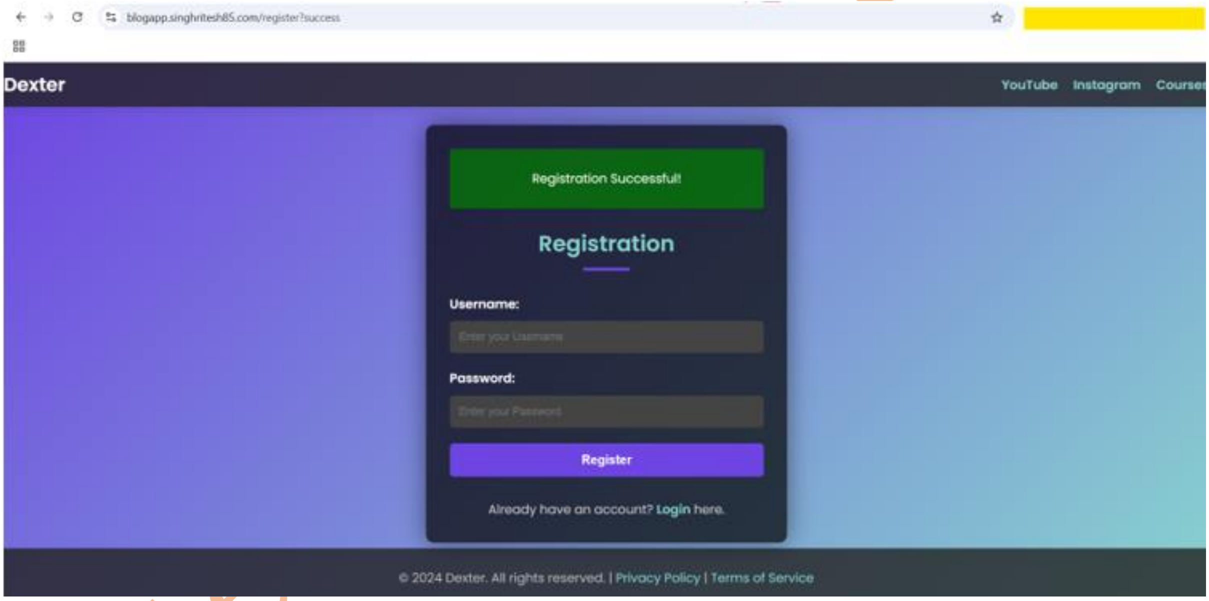
Registration

Username: dex

Password: •••

Already have an account? Login here.

© 2024 Dexter. All rights reserved. | Privacy Policy | Terms of Service



blogapp.singhritesh85.com/register?success

Registration Successful!

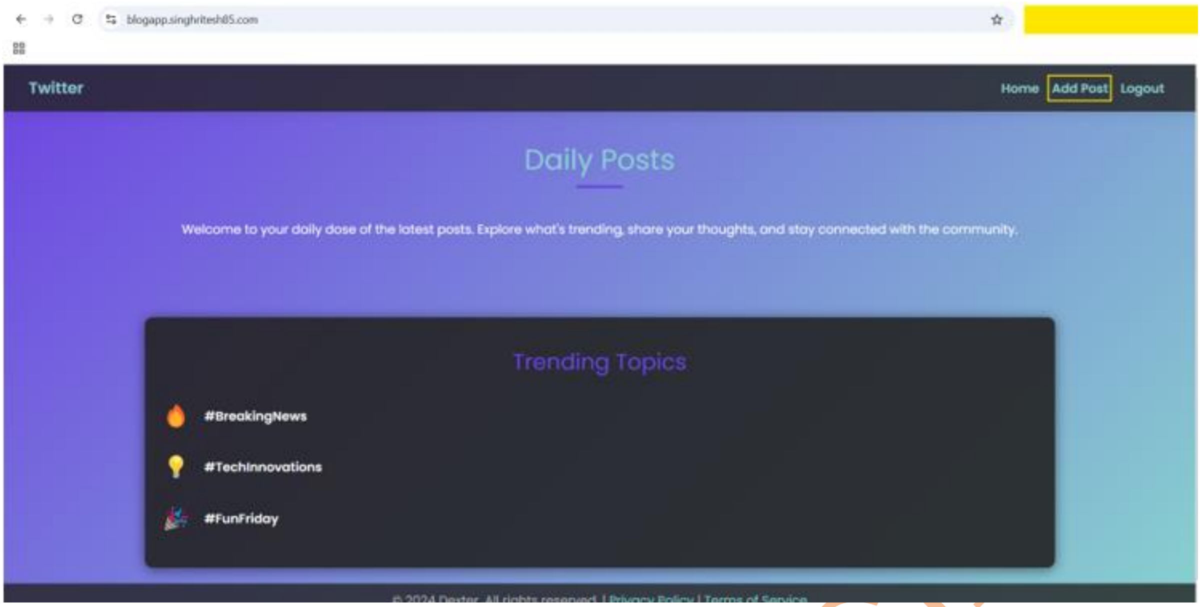
Registration

Username: Enter your Username

Password: Enter your Password

Already have an account? Login here.

© 2024 Dexter. All rights reserved. | Privacy Policy | Terms of Service

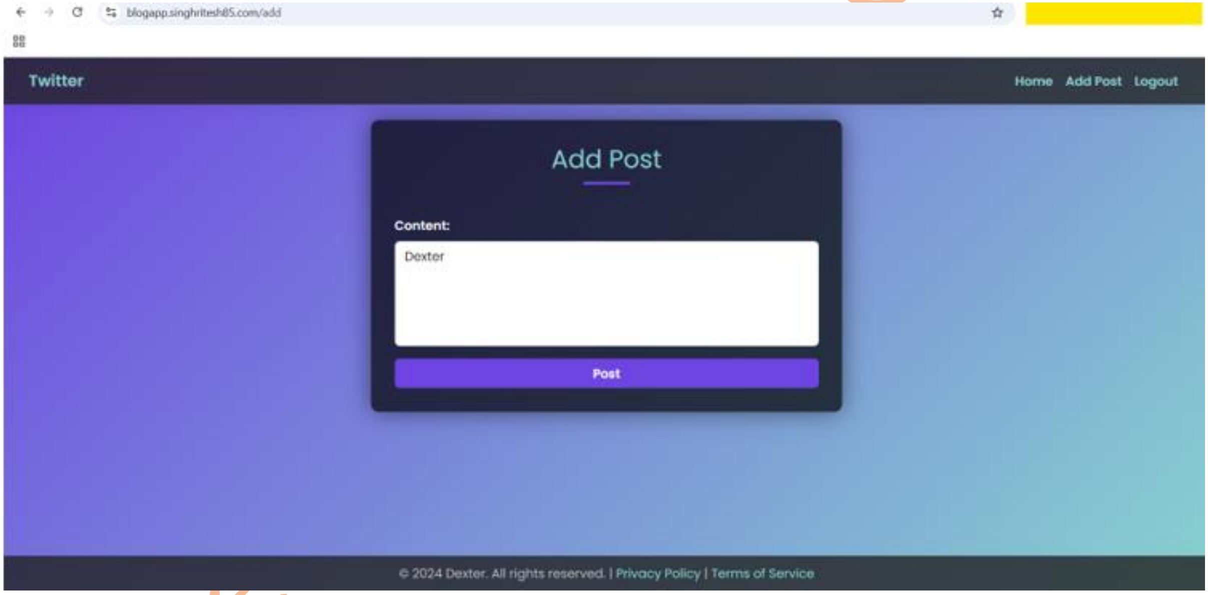


Daily Posts

Welcome to your daily dose of the latest posts. Explore what's trending, share your thoughts, and stay connected with the community.

Trending Topics

- #BreakingNews
- #TechInnovations
- #FunFriday



Add Post

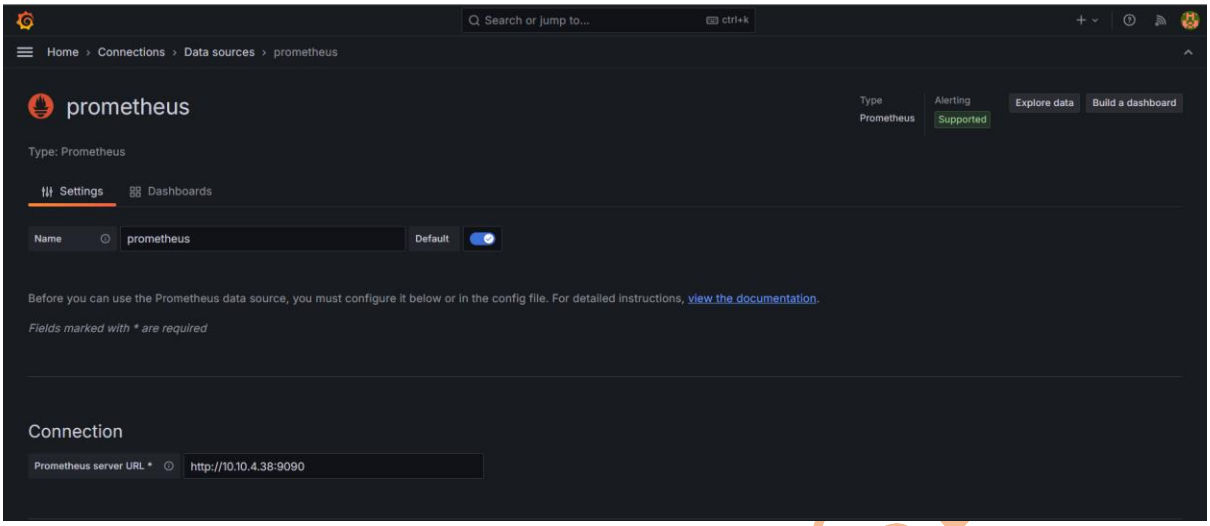
Content:

Dexter

© 2024 Dexter. All rights reserved. | Privacy Policy | Terms of Service

Monitoring Using Prometheus and Grafana and Log Aggregation using Loki

For Monitoring Tool I had used Prometheus and Grafana. To monitor SonarQube I had used SonarQube-Prometheus-Exporter which was installed using terraform at the path **/opt/sonarqube/extensions/plugins**. It was downloaded from the link https://github.com/dmeiners88/sonarqube-prometheus-exporter/releases/download/v1.0.0-SNAPSHOT-2018-07-04/sonar-prometheus-exporter-1.0.0-SNAPSHOT.jar. These steps had been covered in the terraform **user_data_sonarqube.sh**. It is basically a bootstrap script for SonarQube Server. For Monitoring Jenkins, you need to install the plugin Prometheus metrics and then restart Jenkins, these steps already been discussed at the starting. The configuration for prometheus had already been done in the terraform. I had taken sonarqube username and password as **admin** and **Admin123** respectively, you can choose as of your own choice and update the terraform script accordingly (Prometheus needs username and password to extract the metrics from SonarQube). I had provided the terraform script with this GitHub Repository. Below Screenshot shows how I had integrated Prometheus and Loki with Grafana.



prometheus

Type: Prometheus

Type: Prometheus | Alerting: Supported

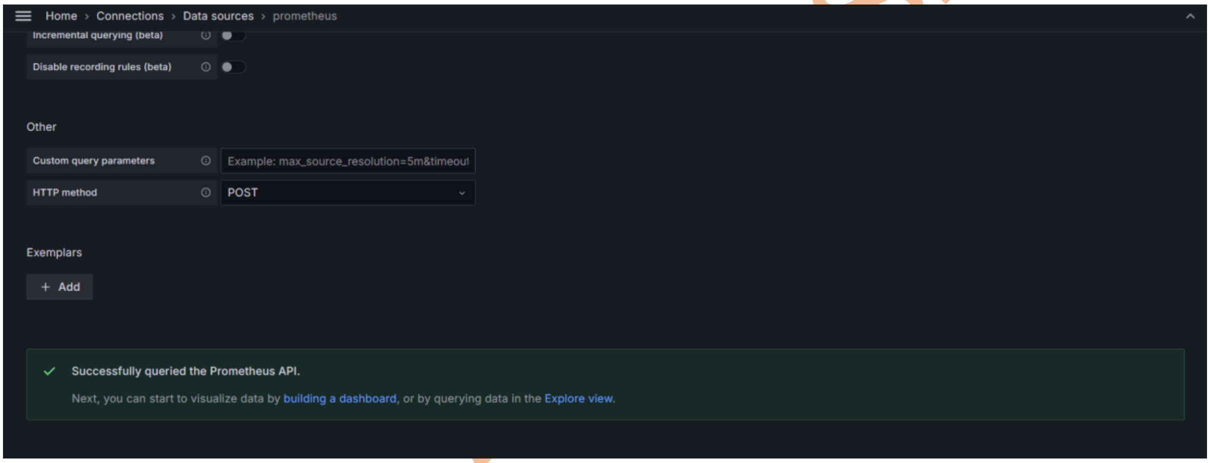
Name: prometheus | Default: on

Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, view the documentation.

*Fields marked with * are required*

Connection

Prometheus server URL *: http://10.10.4.38:9090



Incremental querying (beta)

Disable recording rules (beta)

Other

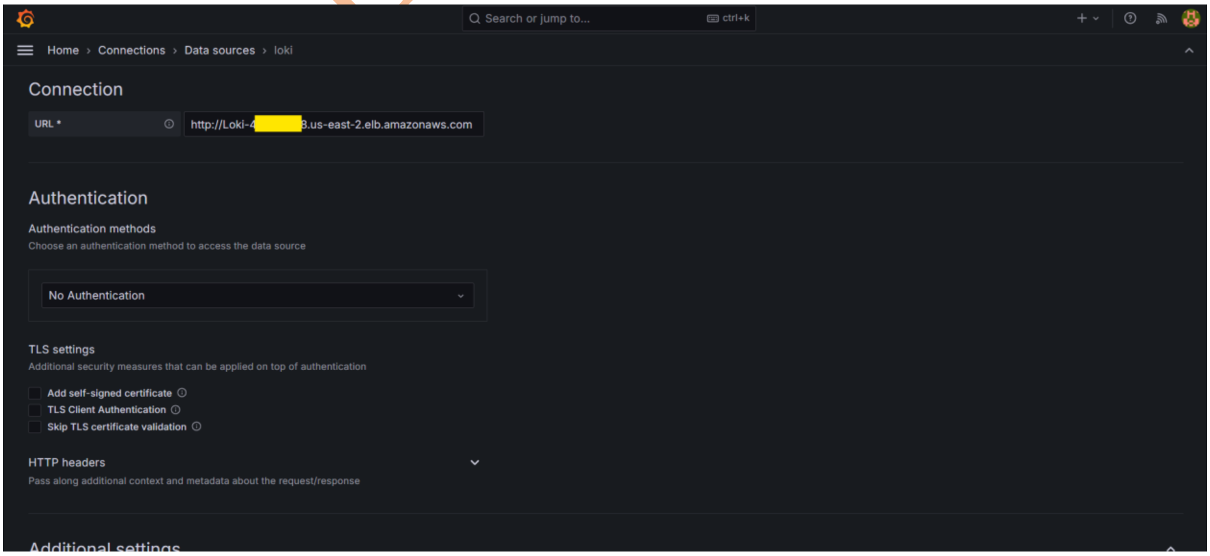
Custom query parameters: Example: max_source_resolution=5m&timeout

HTTP method: POST

Exemplars

✓ Successfully queried the Prometheus API.

Next, you can start to visualize data by building a dashboard, or by querying data in the Explore view.



Connection

URL *: http://Loki-[redacted].us-east-2.elb.amazonaws.com

Authentication

Authentication methods

Choose an authentication method to access the data source

No Authentication

TLS settings

Additional security measures that can be applied on top of authentication

□ Add self-signed certificate

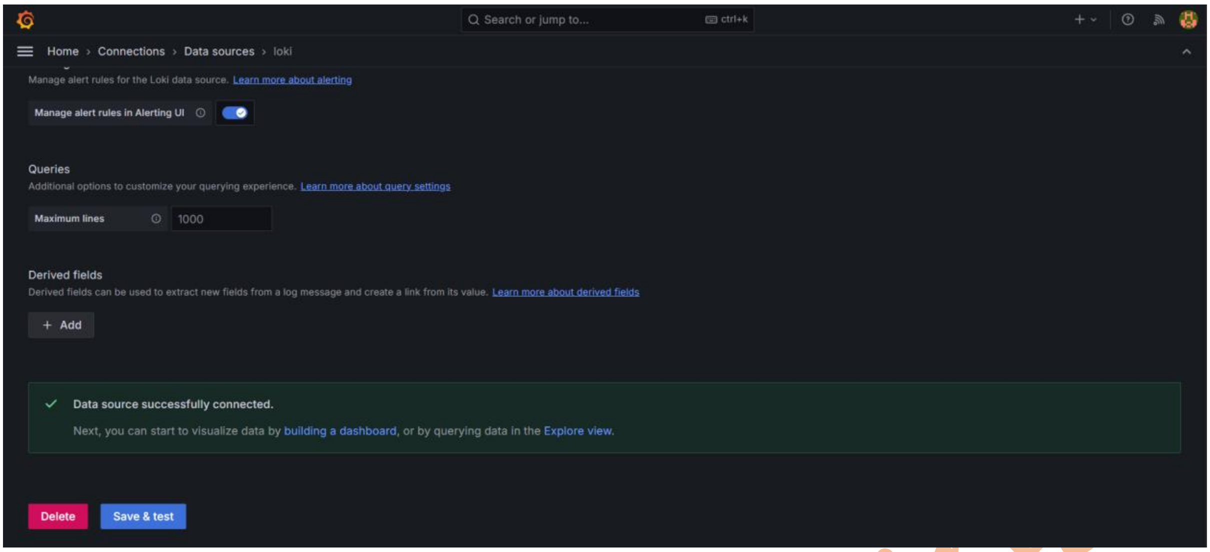
□ TLS Client Authentication

□ Skip TLS certificate validation

HTTP headers

Pass along additional context and metadata about the request/response

Additional settings



Home › Connections › Data sources › loki

Manage alert rules for the Loki data source. Learn more about alerting

Manage alert rules in Alerting UI: on

Queries

Additional options to customize your querying experience. Learn more about query settings

Maximum lines: 1000

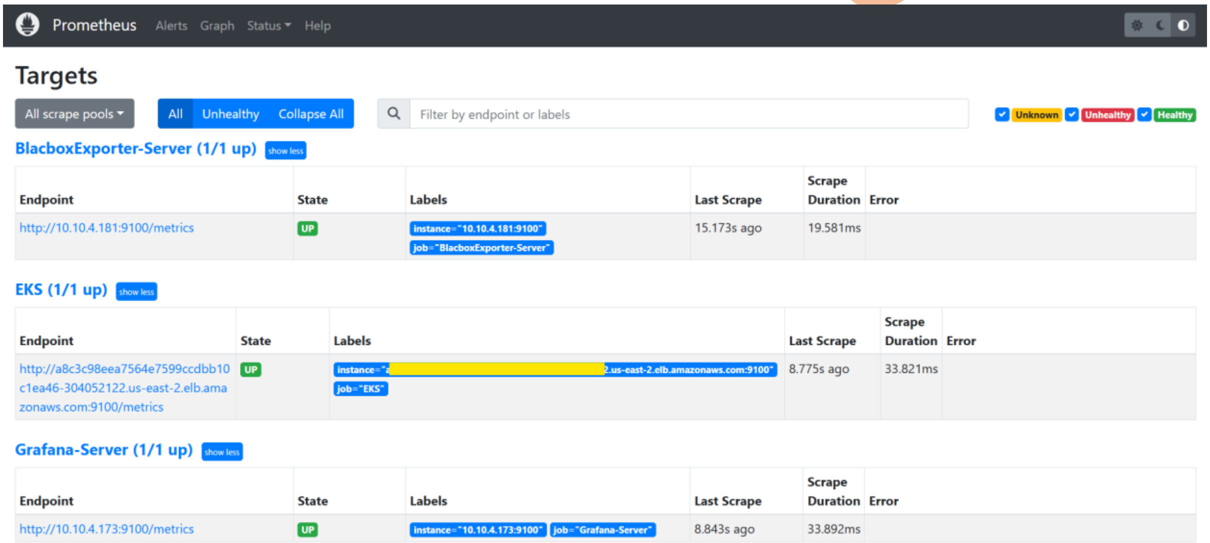
Derived fields

Derived fields can be used to extract new fields from a log message and create a link from its value. Learn more about derived fields

✓ Data source successfully connected.

Next, you can start to visualize data by building a dashboard, or by querying data in the Explore view.

Finally, I checked the Prometheus Console and I was able to see all the Targets was UP.



Targets

BlacboxExporter-Server (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://10.10.4.181:9100/metrics	UP	instance="10.10.4.181:9100" job="BlacboxExporter-Server"	15.173s ago	19.581ms	

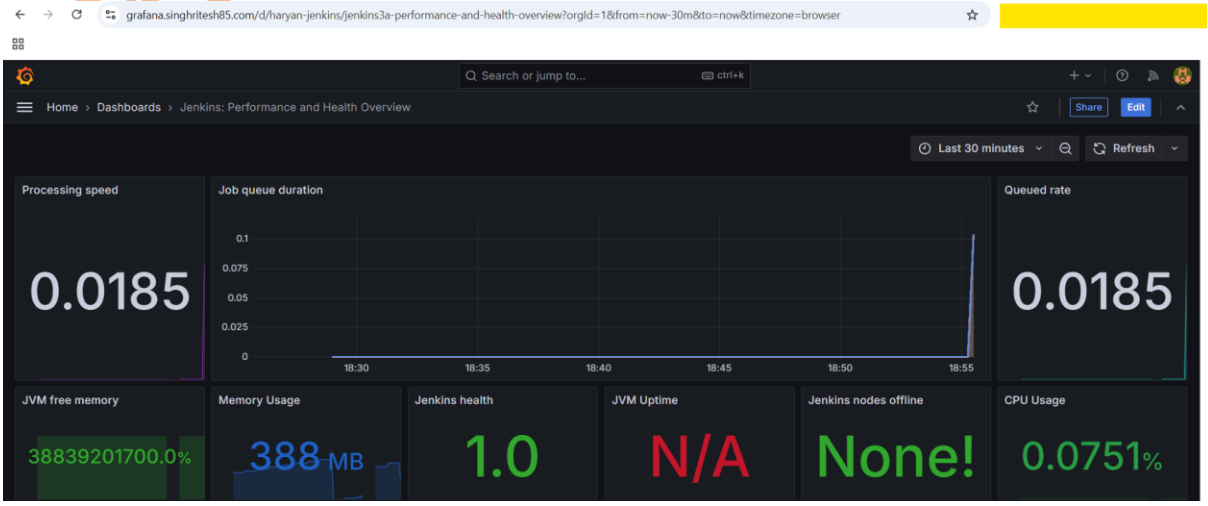
EKS (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://a8c3c98eea7564e7599ccdbb10c1ea46-304052122.us-east-2.elb.amazonaws.com:9100/metrics	UP	instance="a[redacted]2.us-east-2.elb.amazonaws.com:9100" job="EKS"	8.775s ago	33.821ms	

Grafana-Server (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://10.10.4.173:9100/metrics	UP	instance="10.10.4.173:9100" job="Grafana-Server"	8.843s ago	33.892ms	

For Monitoring Jenkins Job using Prometheus I created the Grafana Dashboard using the Grafana ID **9964**.

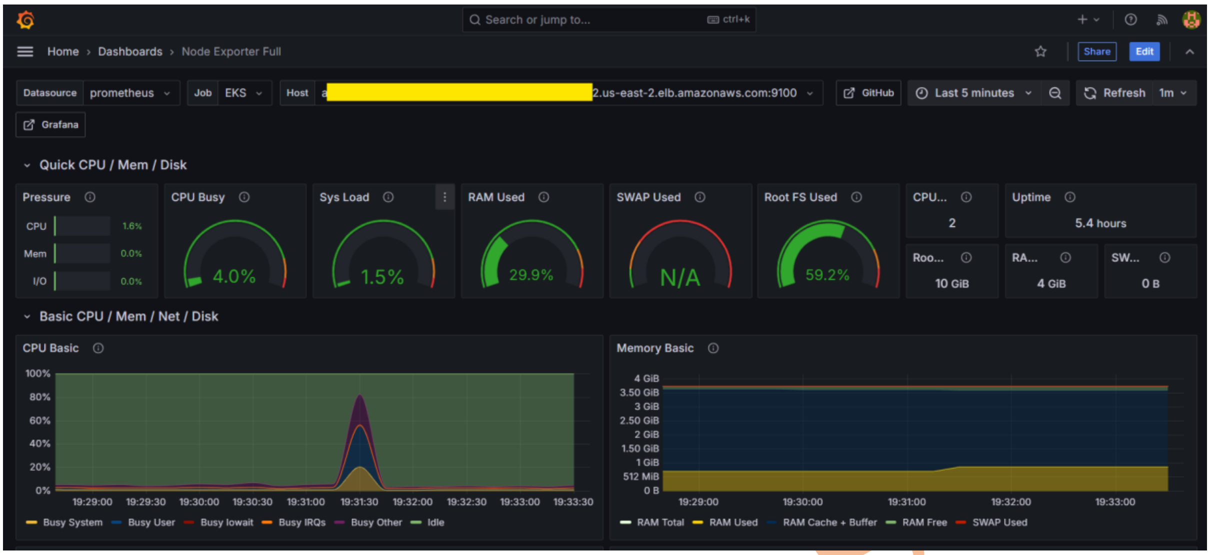


grafana.singhritesh85.com/d/haryan-jenkins/jenkins3a-performance-and-health-overview?orgId=1&from=now-30m&to=now&timezone=browser

[redacted]

Home › Dashboards › Jenkins: Performance and Health Overview

For Monitoring all the Servers and EKS Cluster health using the Node Exporter I used Grafana ID **1860**.

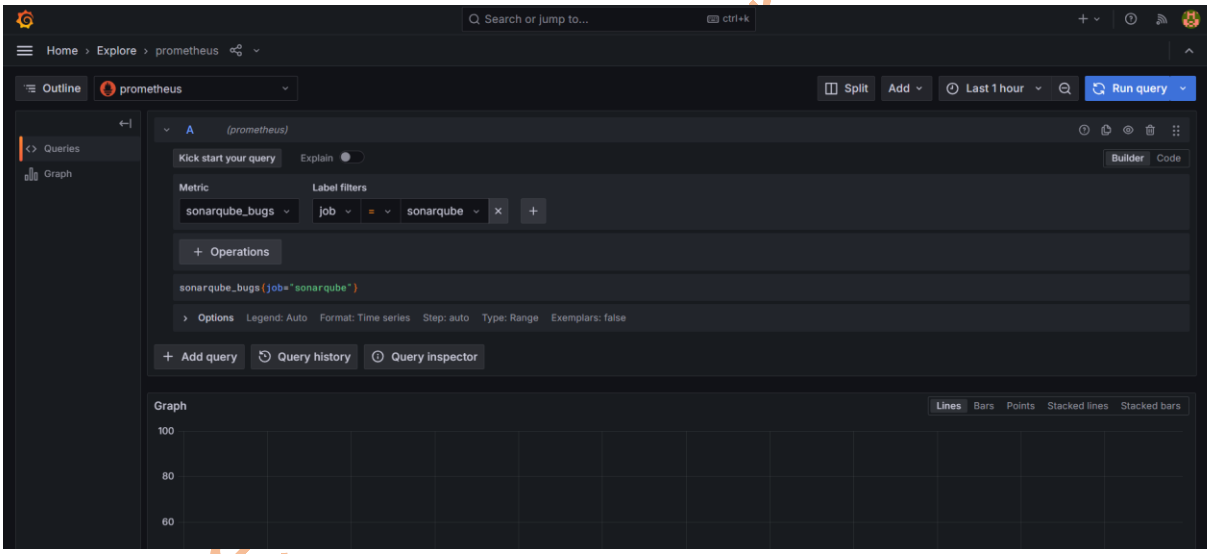


Datasource prometheus
Job EKS
Host a[redacted]2.us-east-2.elb.amazonaws.com:9100
Last 5 minutes
Refresh 1m

Pressure	CPU Busy	Sys Load	RAM Used	SWAP Used	Root FS Used	CPU...	Uptime
CPU 1.6% Mem 0.0% I/O 0.0%	4.0%	1.5%	29.9%	N/A	59.2%	2	5.4 hours

Roo...	RA...	SW...
10 GiB	4 GiB	0 B

Grafana Metrics I started exploring as shown in the screenshot attached below.

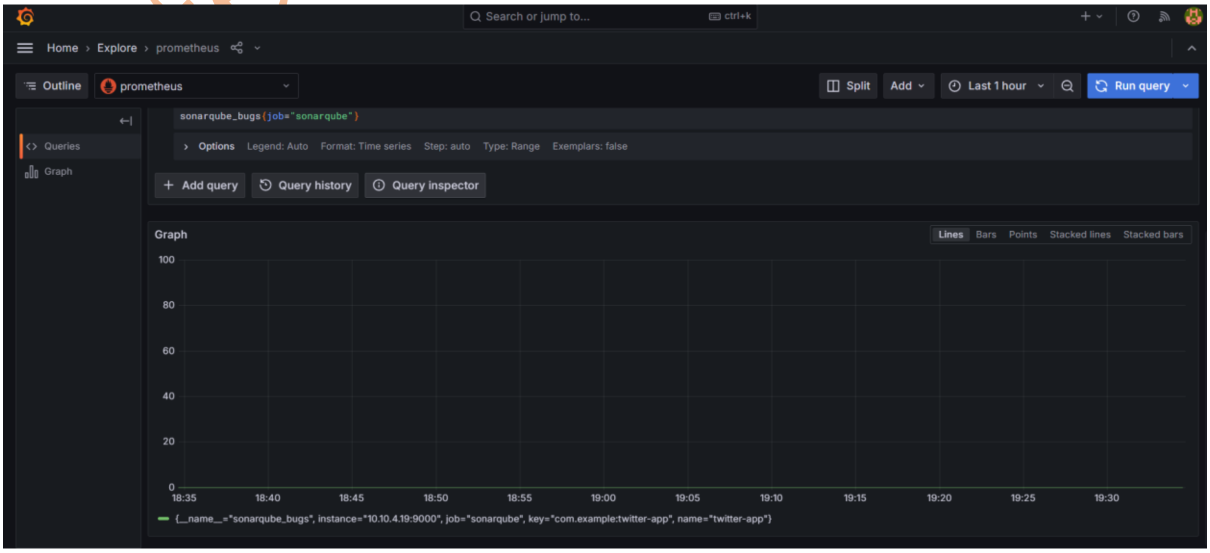


prometheus
Last 1 hour

Metric: sonarqube_bugs
Label filters: job = sonarqube

```
sonarqube_bugs{job="sonarqube"}
```

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



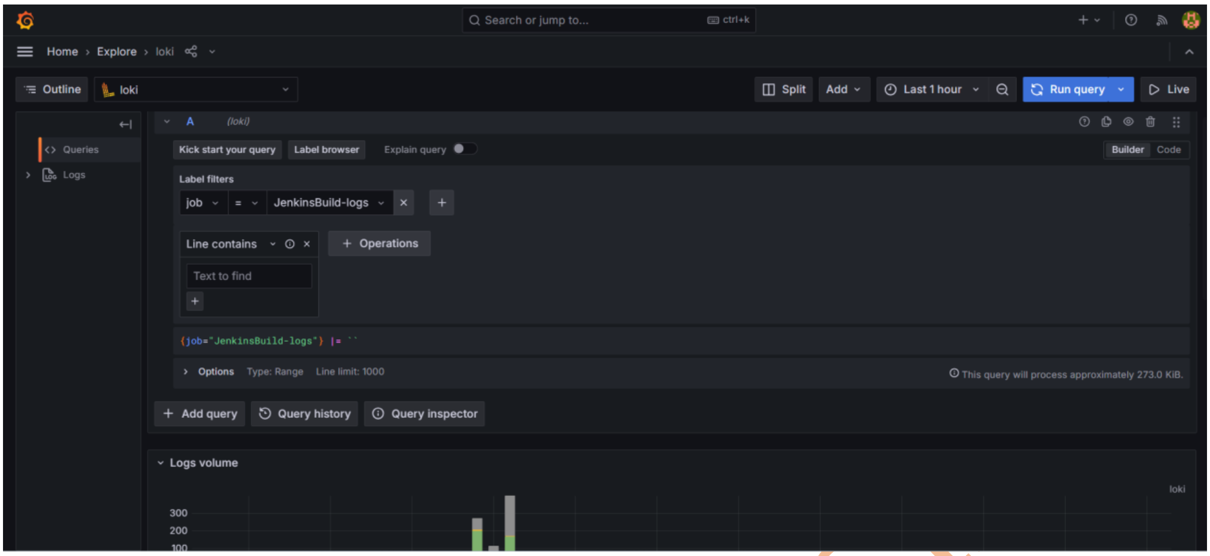
prometheus
Last 1 hour

```
sonarqube_bugs{job="sonarqube"}
```

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

{__name__="sonarqube_bugs", instance="10.10.4.19:9000", job="sonarqube", key="com.example:twitter-app", name="twitter-app"}

Logs using Loki through Grafana I started exploring as shown in the screenshot attached below.



Label filters

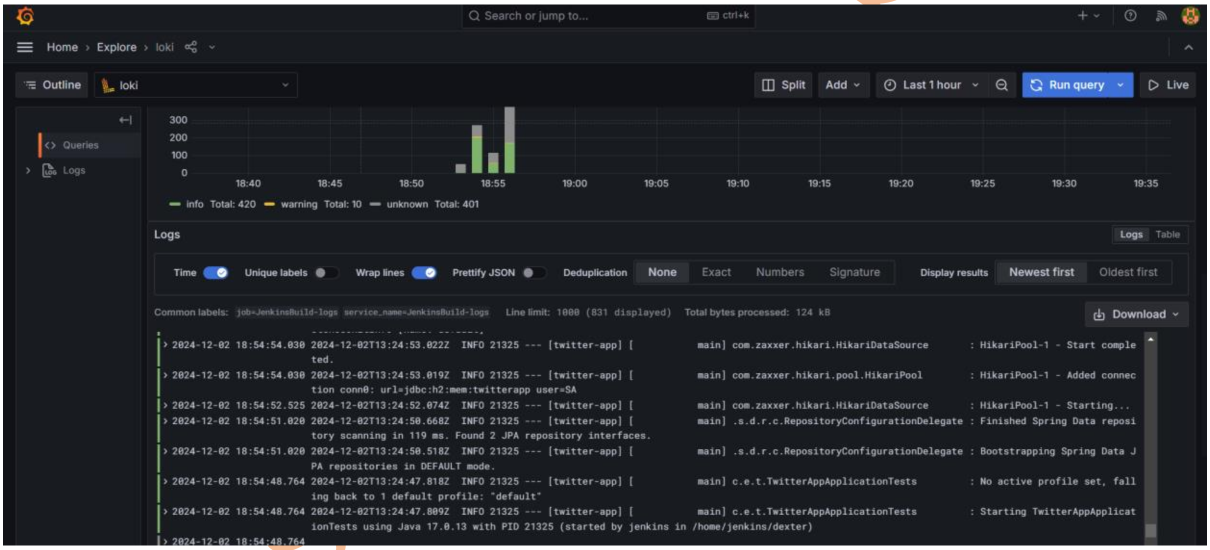
job = JenkinsBuild-logs

Line contains: Text to find

```
{job="JenkinsBuild-logs"} |= ``
```

Options Type: Range Line limit: 1000

This query will process approximately 273.0 KiB.



info Total: 420 warning Total: 10 unknown Total: 401

Time ● Unique labels ○ Wrap lines ● Prettify JSON ○ Deduplication: [✓ None] Exact Numbers Signature Display results: [✓ Newest first] Oldest first

Common labels: job=JenkinsBuild-logs service_name=JenkinsBuild-logs Line limit: 1000 (831 displayed) Total bytes processed: 124 kB

```
> 2024-12-02 18:54:54.030 2024-12-02T13:24:53.022Z  INFO 21325 --- [twitter-app] [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start comple
                          ted.
> 2024-12-02 18:54:54.030 2024-12-02T13:24:53.019Z  INFO 21325 --- [twitter-app] [           main] com.zaxxer.hikari.pool.HikariPool        : HikariPool-1 - Added connec
                          tion conn0: url=jdbc:h2:mem:twitterapp user=SA
> 2024-12-02 18:54:52.525 2024-12-02T13:24:52.074Z  INFO 21325 --- [twitter-app] [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
> 2024-12-02 18:54:51.020 2024-12-02T13:24:50.668Z  INFO 21325 --- [twitter-app] [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data reposi
                          tory scanning in 119 ms. Found 2 JPA repository interfaces.
> 2024-12-02 18:54:51.020 2024-12-02T13:24:50.518Z  INFO 21325 --- [twitter-app] [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data J
                          PA repositories in DEFAULT mode.
> 2024-12-02 18:54:48.764 2024-12-02T13:24:47.818Z  INFO 21325 --- [twitter-app] [           main] c.e.t.TwitterAppApplicationTests         : No active profile set, fall
                          ing back to 1 default profile: "default"
> 2024-12-02 18:54:48.764 2024-12-02T13:24:47.809Z  INFO 21325 --- [twitter-app] [           main] c.e.t.TwitterAppApplicationTests         : Starting TwitterAppApplicat
                          ionTests using Java 17.0.13 with PID 21325 (started by jenkins in /home/jenkins/dexter)
> 2024-12-02 18:54:48.764
```

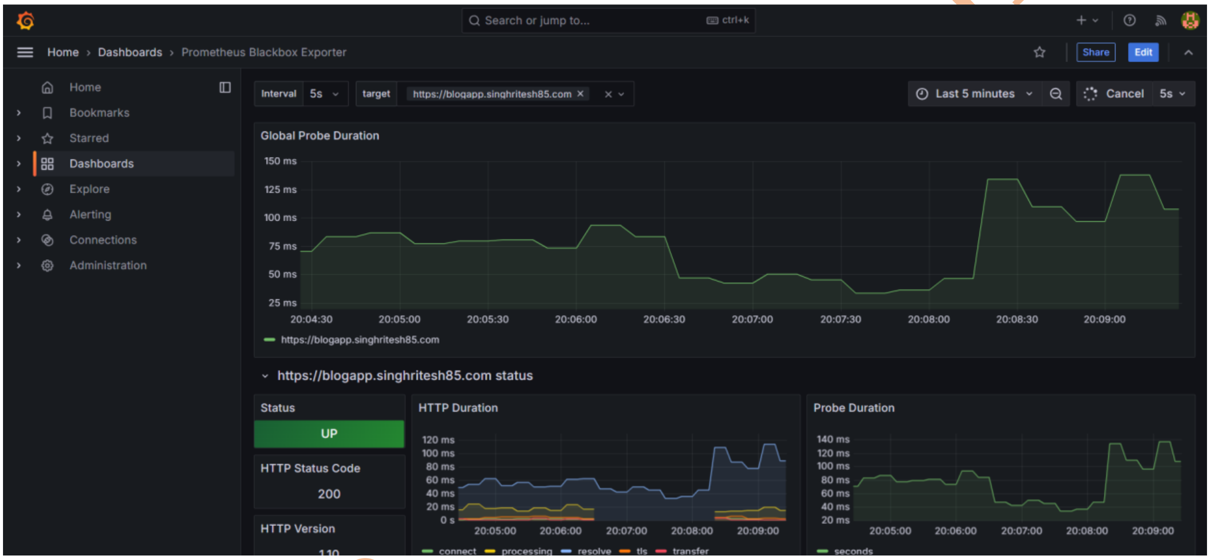
To achieve synthetic monitoring using Prometheus Blackbox Exporter I updated the **/etc/resolv.conf** file for Blackbox Exporter Server as shown in the screenshot attached below. I had used Google's Public DNS Server which is shown in the attached screenshot below.

```
[root@[redacted] ~]# cat /etc/resolv.conf
; generated by /usr/sbin/dhclient-script
search us-east-2.compute.internal
options timeout:2 attempts:5
nameserver 8.8.8.8   #10.10.0.2
```

Finally, I was able to perform the synthetics monitoring on the Blogging Application URL as shown in the screenshot attached below. Application URL **https://blogapp.singhritesh85.com** had been monitored using blackbox exporter.

I had installed Blackbox Exporter on a different server and not on the Prometheus Server. **The module name** is monitor_website.yml present of the blackbox exporter server at the path (/opt/blackbox_exporter_linux_amd64/monitor_website.yml). Prometheus blackbox operator is used for endpoint monitoring (Synthetic Monitoring) across the protocol http, https, TCP and ICMP. In this project I am monitoring the Application URL **https://blogapp.singhritesh85.com** with the help of Prometheus Blackbox-Exporter. Prometheus blackbox exporter will send the metrics to Prometheus. For this project Prometheus acts as a DataSource for Grafana and send metrics to Grafana which we can see with the help of Charts and Graphs.

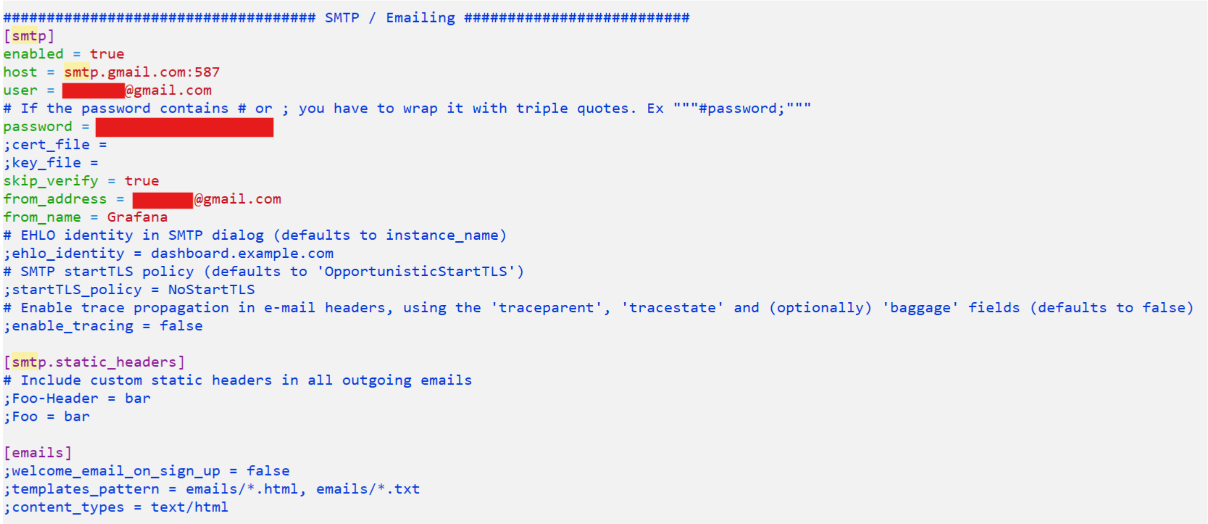
To create the Grafana Dashboard for Application URL Monitoring using blackbox exporter I had used the Grafana ID **7587** and below is the created Dashboard.



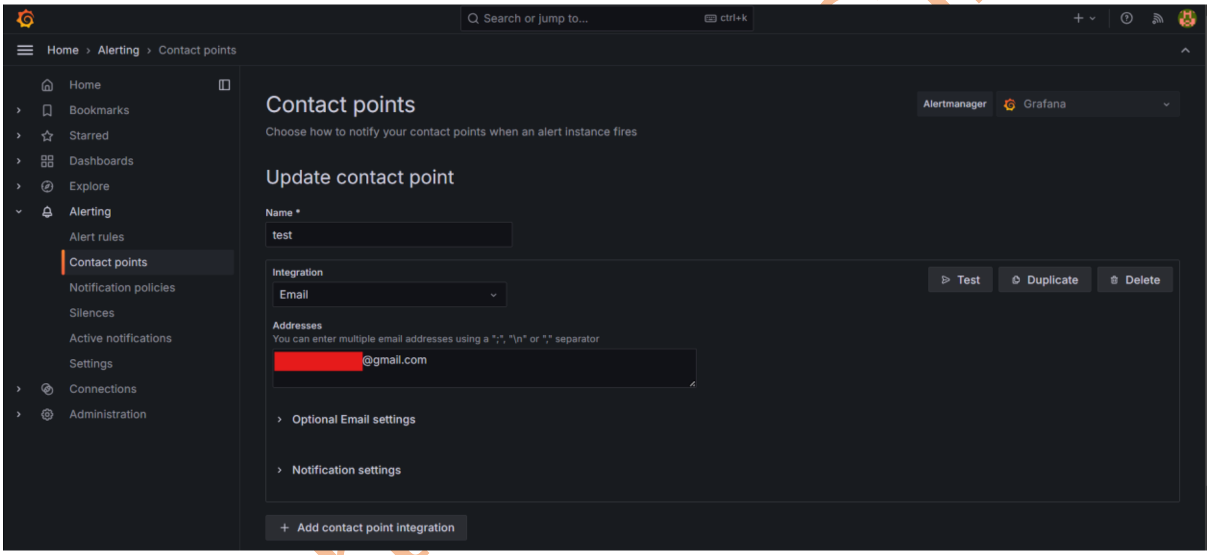
Configuration of Alerts in Grafana

To configure Alerts in Grafana, first I created **contact points** with the Email ID and changed smtp settings in the configuration file /etc/grafana/grafana.ini of Grafana as shown in the screenshot attached below.

```
[root@[redacted] ~]# cat /etc/grafana/grafana.ini
```



```
#################################### SMTP / Emailing ##########################
[smtp]
enabled = true
host = smtp.gmail.com:587
user = [redacted]@gmail.com
# If the password contains # or ; you have to wrap it with triple quotes. Ex """#password;"""
password = [redacted]
;cert_file =
;key_file =
skip_verify = true
from_address = [redacted]@gmail.com
from_name = Grafana
# EHLO identity in SMTP dialog (defaults to instance_name)
;ehlo_identity = dashboard.example.com
# SMTP startTLS policy (defaults to 'OpportunisticStartTLS')
;startTLS_policy = NoStartTLS
# Enable trace propagation in e-mail headers, using the 'traceparent', 'tracestate' and (optionally) 'baggage' fields (defaults to false)
;enable_tracing = false

[smtp.static_headers]
# Include custom static headers in all outgoing emails
;Foo-Header = bar
;Foo = bar

[emails]
;welcome_email_on_sign_up = false
;templates_pattern = emails/*.html, emails/*.txt
;content_types = text/html
```



Contact points

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

Update contact point

Name *: test

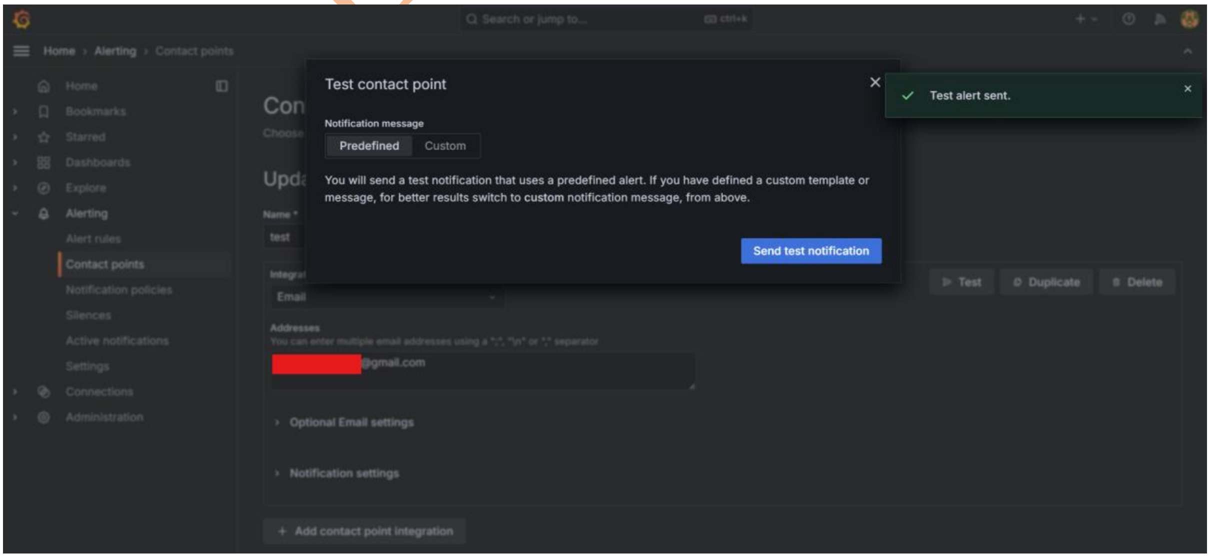
Integration: Email

Addresses
You can enter multiple email addresses using a ";", "\n" or "," separator

[redacted]@gmail.com

› Optional Email settings

› Notification settings



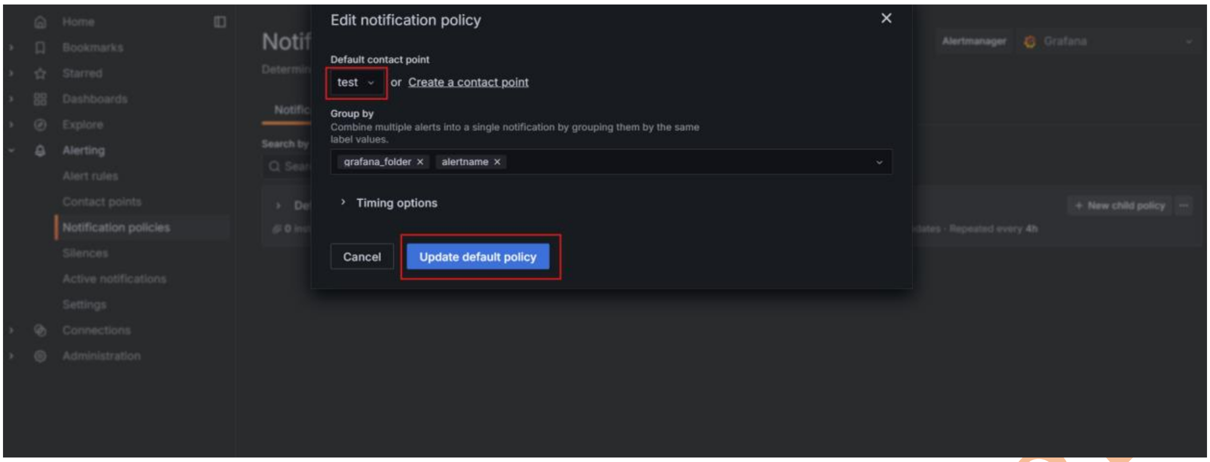
Test contact point

Notification message: Predefined

You will send a test notification that uses a predefined alert. If you have defined a custom template or message, for better results switch to custom notification message, from above.

✓ Test alert sent.

The Default **Notification Policy** had been changed as shown in the screenshot attached below.



Edit notification policy

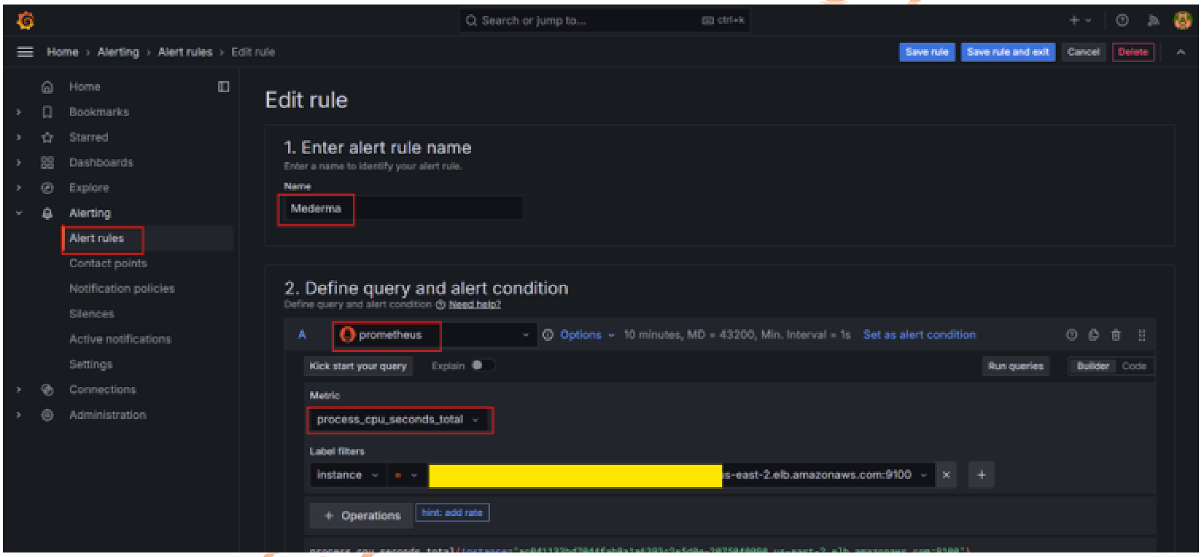
Default contact point: test or Create a contact point

Group by: Combine multiple alerts into a single notification by grouping them by the same label values.

grafana_folder, alertname

Timing options

Configure **Alert Rule** as shown in the screenshot attached below.



Edit rule

1. Enter alert rule name

Enter a name to identify your alert rule.

Name: Mederma

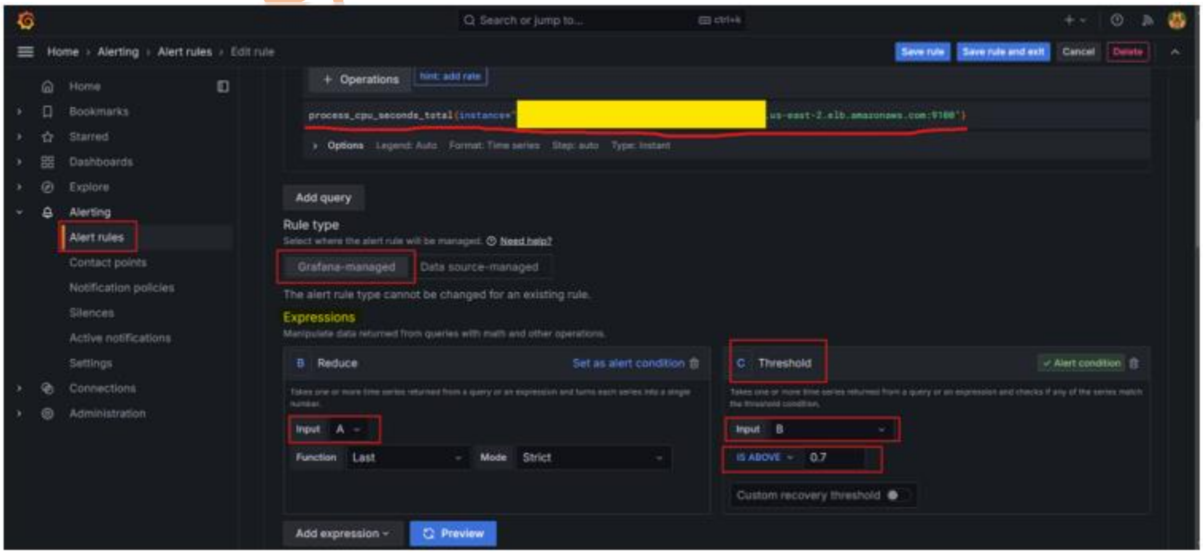
2. Define query and alert condition

Define query and alert condition Need help?

A prometheus Options 10 minutes, MD = 43200, Min. Interval = 1s Set as alert condition

Metric: process_cpu_seconds_total

Label filters: instance = [redacted]s-east-2.elb.amazonaws.com:9100



```
process_cpu_seconds_total{instance="[redacted]us-east-2.elb.amazonaws.com:9100"}
```

Options Legend: Auto Format: Time series Step: auto Type: Instant

Rule type

Select where the alert rule will be managed. Need help?

Grafana-managed | Data source-managed

The alert rule type cannot be changed for an existing rule.

Expressions

Manipulate data returned from queries with math and other operations.

B Reduce — Takes one or more time series returned from a query or an expression and turns each series into a single number.

Input: A

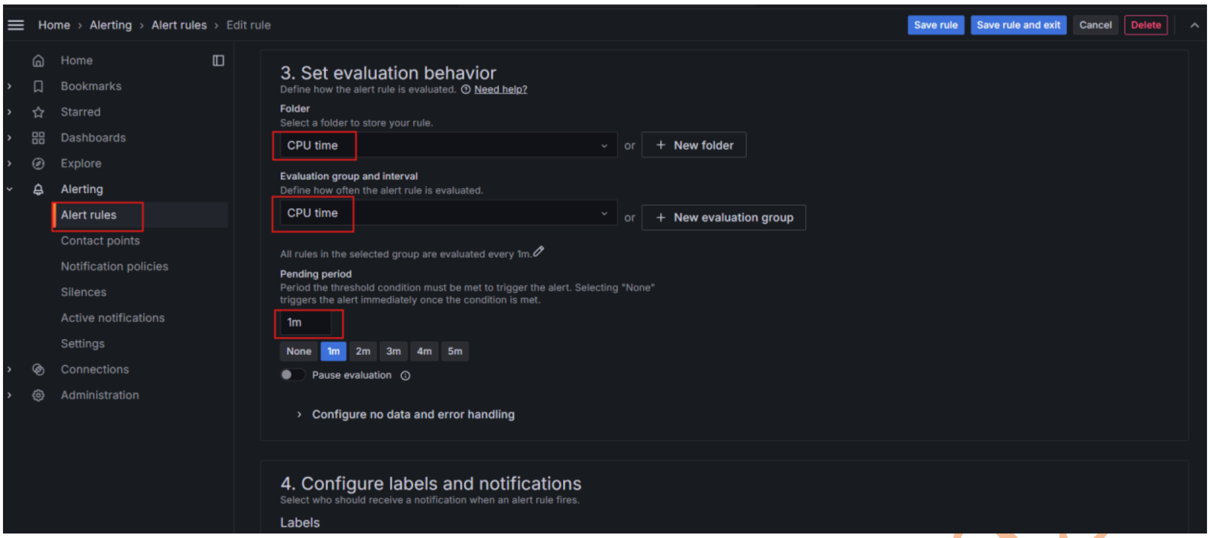
Function: Last Mode: Strict

C Threshold — Takes one or more time series returned from a query or an expression and checks if any of the series match the threshold condition.

Input: B

IS ABOVE 0.7

Custom recovery threshold



3. Set evaluation behavior

Define how the alert rule is evaluated. Need help?

Folder

Select a folder to store your rule.

CPU time or + New folder

Evaluation group and interval

Define how often the alert rule is evaluated.

CPU time or + New evaluation group

All rules in the selected group are evaluated every 1m.

Pending period

Period the threshold condition must be met to trigger the alert. Selecting "None" triggers the alert immediately once the condition is met.

1m

None 1m 2m 3m 4m 5m

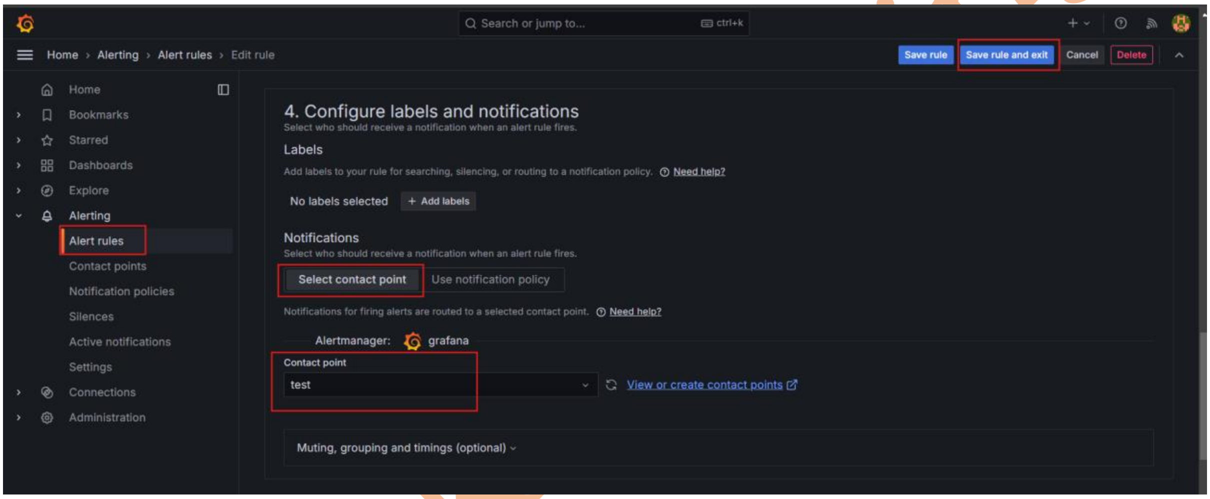
Pause evaluation

› Configure no data and error handling

4. Configure labels and notifications

Select who should receive a notification when an alert rule fires.

Labels



4. Configure labels and notifications

Select who should receive a notification when an alert rule fires.

Labels

Add labels to your rule for searching, silencing, or routing to a notification policy. Need help?

No labels selected + Add labels

Notifications

Select who should receive a notification when an alert rule fires.

Select contact point Use notification policy

Notifications for firing alerts are routed to a selected contact point. Need help?

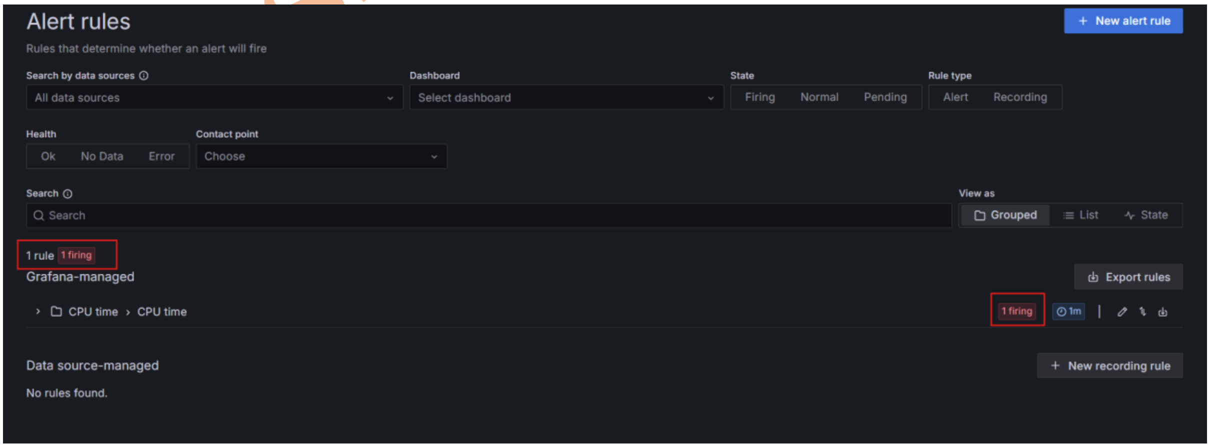
Alertmanager: grafana

Contact point

test View or create contact points

Muting, grouping and timings (optional)

If the Alert Rule is in firing state after condition crosses the threshold condition, then Grafana console screenshot will be showing the same as shown in the screenshot attached below.



Alert rules

Rules that determine whether an alert will fire

Search by data sources: All data sources

Dashboard: Select dashboard

State: Firing Normal Pending

Rule type: Alert Recording

Health: Ok No Data Error

Contact point: Choose

Search: Search

View as: Grouped List State

1 rule 1 firing

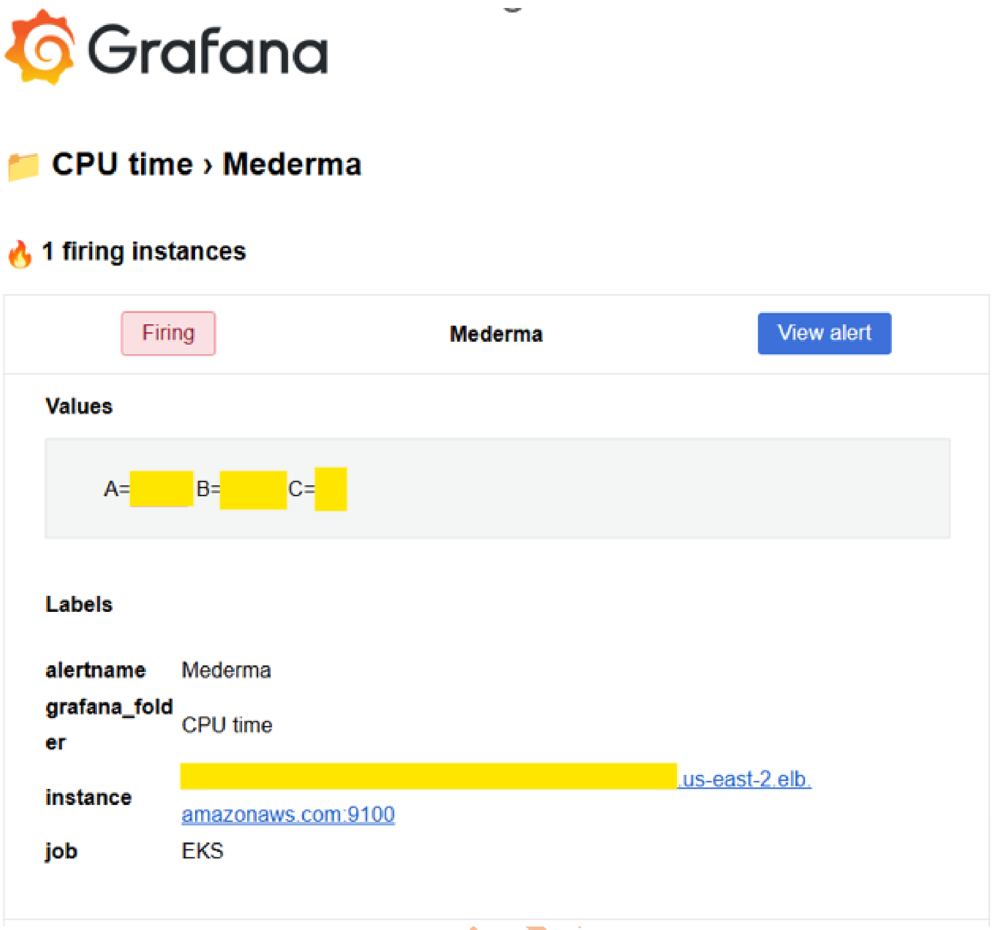
Grafana-managed

› CPU time › CPU time 1 firing 1m

Data source-managed

No rules found.

An Email will be sent to the Email ID as shown in the screenshot attached below.



Grafana

CPU time › Mederma

🔥 1 firing instances

Values

A=[redacted] B=[redacted] C=[redacted]

Labels

alertname	Mederma
grafana_folder	CPU time
instance	[redacted] us-east-2.elb.amazonaws.com:9100
job	EKS