

CI\_CD SONARQUBE\_JFROG\_E  
KS\_JENKINS

---

# Prerequisite

- 1) Need AWS Account
- 2) Need Terraform latest Version
- 3) AWS-CLI-v2
- 4) kubectl
- 5) Jenkins Server - create ec2 vm in aws and install jenkins

# Jenkins server setup & installation

ssh login created new jenkins server

1) apt update

2 ) sudo apt install default-jre

3) wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -

4) sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

5) sudo apt update

6) sudo apt install jenkins

7) sudo apt install docker.io

8) sudo usermod -aG docker jenkins

9) aws configure

10) kubectl install

11) curl -LO "https://dl.k8s.io/release/\$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

12 ) sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl



```
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"  
% Total    % Received % Xferd  Average Speed   Time     Time      Time  Current  
          Dload  Upload Total Spent   Left Speed  
00  138  100  138    0     0  484      0 --:--:-- --:--:-- --:--:--  484  
00 42.9M  100 42.9M    0     0 48.8M      0 --:--:-- --:--:-- --:--:-- 184M  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$ sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$  
buntu@ip-172-31-45-194:~$ kubectl version --client  
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short. Use --output=yaml|json to get the full version.  
client Version: version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.2", GitCommit:"5835544ca568b757a8ecae5c153f317e5736700e", GitTreeState:"clean", BuildDate:"2022-09-21T14:33:40Z", GoVersion:"go1.18.1", Compiler:"gc", Platform:"linux/amd64"}  
[...]
```

```
enkins@ip-172-31-45-194:/home/ubuntu$ aws configure  
WS Access Key ID [*****CGE6]: AKIAZ53A5C4HFUDV7W46  
WS Secret Access Key [*****YlHu]: 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  
efault region name [ap-south-1]:  
efault output format [None]:  
enkins@ip-172-31-45-194:/home/ubuntu$ aws eks update-kubeconfig --name greens-cluster-cluster --region ap-south-1  
dded new context arn:aws:eks:ap-south-1:682566162190:cluster/greens-cluster-cluster to /var/lib/jenkins/.kube/config  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$  
enkins@ip-172-31-45-194:/home/ubuntu$ kubectl get nodes  
NAME                  STATUS   ROLES      AGE     VERSION  
p-172-31-34-185.ap-south-1.compute.internal   Ready   <none>   24h   v1.22.12-eks-ba74326  
enkins@ip-172-31-45-194:/home/ubuntu$ client_loop: send disconnect: Broken pipe
```

## INSTALLING SONAR-QUBE in AWS EKS CLUSTER

## First aws configure in local

Then

```
aws eks update-kubeconfig –name cluster_name –region
```

## INSTALLATION

```
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" already exists with the same configuration, skipping
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ 
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ helm install sonarqube bitnami/sonarqube
NAME: sonarqube
LAST DEPLOYED: Mon Sep 26 11:08:16 2022
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
** Please be patient while the chart is being deployed **
```

2. Open a browser and access SonarQube using the obtained URL.

3. Login with the following credentials below:

```
echo Username: user
echo Password: $(kubectl get secret --namespace default sonarqube -o jsonpath=".data.sonarqube-password" | base64 -d)
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$
```

```
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ $(kubectl get secret --namespace default sonarqube -o jsonpath=".data.sonarqube-p
assword" | base64 -d)
yMnfCHxDLA: command not found
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$
```

You get the password yMnfCHxDLA and username user

```
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
jenkins-0     2/2     Running   0          14h
sonarqube-7466c89955-z8b29 1/1     Running   0          5m30s
sonarqube-postgresql-0   1/1     Running   0          5m30s
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$
```

```
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$
```

```
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$ kubectl get svc
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP
jenkins        LoadBalancer  10.100.87.67  a31b7ccdbfd48a2abd8939e16635fd-222433920.ap-south-1.elb.amazonaws.com  8080:32268
jenkins-agent  ClusterIP   10.100.122.38 <none>
kubernetes     ClusterIP   10.100.0.1    <none>
sonarqube      LoadBalancer  10.100.118.240 af6a0d8e984d54a19969e8e614ff49d9-15825217.ap-south-1.elb.amazonaws.com  80:31930/TCP
sonarqube-postgresql 6m23s   ClusterIP   10.100.142.240 <none>
sonarqube-postgresql-hl 6m23s   ClusterIP   None       <none>
prakash@prakash-Latitude-3420:~/prakash_hank/test-projects$
```

Here u get sonarqube loadbalancer elasticip

And go to browser

Paste the ip get the SonarQube UI

## Log In to SonarQube

## Go to profile settings generate tokens

The screenshot shows the SonarQube interface with the following details:

- Header:** Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, Search for projects...
- User Profile:** A (Administrator)
- Current Page:** Security
- Section:** Tokens
- Text:** If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.
- Generate Tokens Form:**
  - Name: Enter Token Name
  - Type: Global Analysis Token
  - Expires in: 30 days
  - Generate button
- Success Message:** New token "jenkin" has been created. Make sure you copy it now, you won't be able to see it again!
- Copy Button:** Copy sqa\_4a396a89d3e6cef302e6d0f40692f058479f1be7
- Table:** A list of existing tokens.

Name	Type	Project	Last use	Created	Expiration
jenkin	Global		Never	September 26, 2022	September 26, 2023
- Action Buttons:** Revoke (for the jenkin token)
- Footer:** Enter a new password

Go to jenkins plugin manager install SonarQube Scanner

Dashboard > Manage Jenkins > Plugin Manager

↑ Back to Dashboard

Manage Jenkins

## Plugin Manager

Updates

Available

Installed

Advanced

sona

Install Name ↓

Released

Personalization for Blue Ocean 1.25.8

External Site/Tool Integrations User Interface

19 days ago

Blue Ocean Personalization

SonarQube Scanner 2.14

External Site/Tool Integrations Build Reports

10 mo ago

This plugin allows an easy integration of [SonarQube](#), the open source platform for Continuous Inspection of code quality.

Sonar Quality Gates 1.3.1

Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the

Install without restart

Download now and install after restart

Update information obtained: 14 hr ago

Check now

And go to jenkins global tool configuration and setup sonarqube server host and generated token

Environment variables

Tool Locations

---

### SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Environment variables Enable injection of SonarQube server configuration as build environment variables

### SonarQube installations

List of SonarQube installations

Add SonarQube

---

### Metrics

Access keys ?

Give sonar qube generated token here

Kind

Secret text



Scope ?

Global (Jenkins, nodes, items, all child items, etc)



Secret

.....

ID ?

sonar-secret

Description ?

Enable injection of SonarQube server configuration as build environmental variables

## SonarQube installations

List of SonarQube installations

Name

sonarqube



Server URL

Default is http://localhost:9000

http://af6a0d8e984d54a19969e8e614ff49d9-15825217.ap-south-1.elb.amazonaws.com

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonar-secret



+ Add

Save

Apply

Store secrets like here

## Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 sonar-secret	sonar-secret	Secret text	
 gitlab-id	saidamo/*********	Username with password	

Icon: S M L

Store jFROG username like below with username docker url

Jenkins

Search (CTRL+K)

Admin log out

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

## Global credentials (unrestricted)

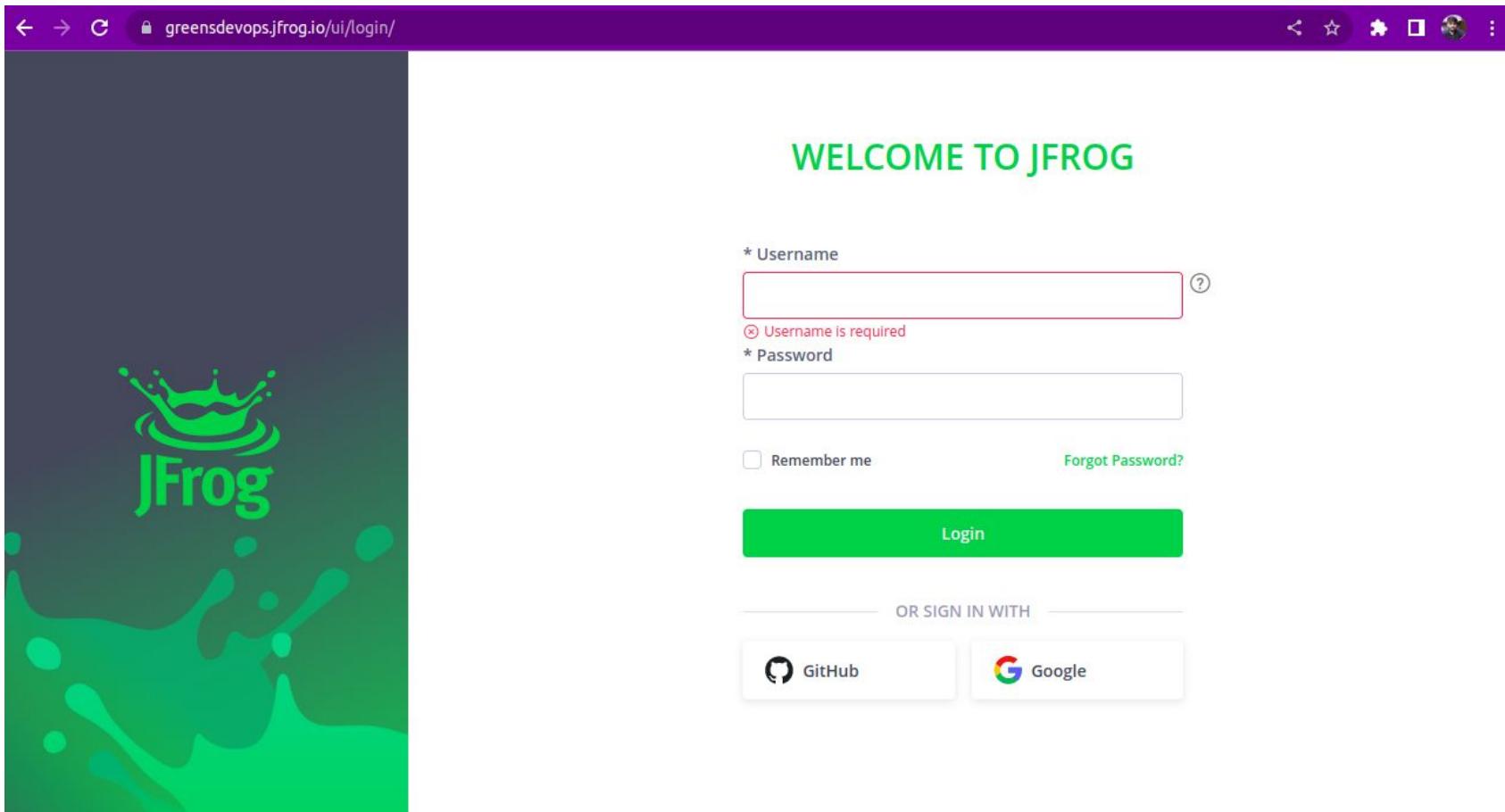
+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
gitlab-id	saidamo/*********	Username with password	
sonar-token	sonar-token	Secret text	
docker-jfrog	damokrishan@gmail.com greensdevops.jfrog.io/*********	Username with password	
aws-id	aws-id	Secret text	
aws-key	aws-key	Secret text	

Icon: S M L

Lets go to jfrog setup



The image shows a screenshot of a web browser displaying the JFrog login page. The URL in the address bar is `greensdevops.jfrog.io/ui/login/`. The page features a large green header with the JFrog logo on the left. The main content area has a white background with a green title "WELCOME TO JFROG" centered at the top. Below the title are two input fields: a red-bordered "Username" field with a required error message and a grey-bordered "Password" field. There is also a "Remember me" checkbox and a "Forgot Password?" link. A large green "Login" button is positioned below the password field. At the bottom, there is a "OR SIGN IN WITH" section featuring "GitHub" and "Google" social login buttons.

greensdevops.jfrog.io/ui/login/

# WELCOME TO JFROG

\* Username

(\*) Username is required

\* Password

Remember me      [Forgot Password?](#)

[Login](#)

OR SIGN IN WITH

 GitHub

 Google

 Search Packages[Upgrade Now](#)

AL

Project  
All

## Repositories

[Set Me Up](#)[+ Add Repositories](#)

Artifactory

Quick Setup

Repositories

Packages

Artifacts

Builds

Xray

Distribution

Pipelines

Learning Center

Local

Remote

Virtual

Federated

A remote repository serves as a caching proxy for a repository managed at a remote URL (which may itself be another Artifactory remote repository).

0 Repositories

Search

<input type="checkbox"/> Repository Key ↑	Type	Project	URL	Replica
---	------	---------	-----	---------

**No results were found**

Try to change your search



AL Project  
All

Artifactory

Quick Setup

## Repositories

Packages

Artifacts

Builds

Xray

Distribution

Pipelines

Learning Center

## Select Package Type

Search by package type



Alpine



Bower



CRAN



Cargo



Chef



CocoaPods



Composer



Conan



Conda



Debian



Docker



Gems



Generic



GitLfs



Go



Gradle



Helm



Ivy



Maven



npm



NuGet



Opkg



P2



Pub

Close

Create Remote Repository



Search Packages

Upgrade Now



AL Project All

Artifactory

Quick Setup

Repositories

Packages

Artifacts

Builds

Xray

Distribution

Pipelines

Learning Center

Repositories > New remote Repository

## New Remote Repository

Basic

Advanced

Replications

Docker

\* Repository Key

greenstest

\* URL

Cancel

Create Remote Repository



Search Packages

Upgrade Now



AL Project All

Artifactory

Quick Setup

Repositories

Packages

Artifacts

Builds

Xray

Distribution

Pipelines

Learning Center

Repositories

Set Me Up

+ Add Repositories

Local    Remote    Virtual    Federated

A remote repository serves as a caching proxy for a repository managed at a remote URL (which may itself be another Artifactory remote repository).

1 Repositories

Search

<input type="checkbox"/> Repository Key ↑	Type	Project	URL	Replicatio
<input type="checkbox"/> greenstest	Docker		https://registry-1.docker.io/	



AL Project All

Artifactory

Quick Setup

Repositories

Packages

Artifacts

Builds

Xray

Distribution

Pipelines

Learning Center

Search Packages

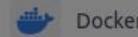
Repositories

Local    Remote    Virtual    Federated

A remote repository serves as a caching proxy for a re

1 Repositories

Name    Type    Project



Docker

## Set Up a Docker client



Repository

greenstest

Client

Docker

Configure

Pull

In your Terminal, Run the following command.

when prompted - provide your username(damokrishan@gmail.com) and

API Key

```
1 | docker login -udamokrishan@gmail.com greensdevops.jfrog.io
```

💡 [Learn more about working with Docker client](#)

← Select a different package type

Done

# Create private secret registry for docker

Go to jenkins

Create new pipeline

Dashboard > cicd >

## Configuration

General

Advanced Project Options

Pipeline

### Definition

Pipeline script from SCM

SCM ?

Git

### Repositories ?

Repository URL ?

https://gitlab.com/saidamo/eks-ci-cd-sonarqube-jenkins.git

Credentials ?

saidamo/\*\*\*\*\*\*\*\*

+ Add

Advanced...

Save

Apply

Finally i put the code and jenkins file inside the gitlab please refer  
After deployment completed u get svc load balancer like below

```
prakash@prakash-Latitude-3420:~/Downloads$  
prakash@prakash-Latitude-3420:~/Downloads$  
prakash@prakash-Latitude-3420:~/Downloads$ kubectl get pods  
NAME          READY   STATUS    RESTARTS   AGE  
green-eks-5fd68f7846-8j7w8  1/1     Running   0          24s  
green-eks-67cbf58579-sv7wx  1/1     Terminating   0          8m5s  
sonarqube-6cc47bbf68-ts48m  1/1     Running   0          27h  
sonarqube-postgresql-0     1/1     Running   0          27h  
prakash@prakash-Latitude-3420:~/Downloads$ kubectl get svc  
NAME           TYPE      CLUSTER-IP        EXTERNAL-IP          PORT(S)  
AGE  
green-eks       LoadBalancer  10.100.129.138  ac6ab9800b15349da85f061acfd056b6-1312106747.ap-south-1.elb.amazonaws.com  80:32520/  
TCP,443:32673/TCP  21m  
kubernetes      ClusterIP    10.100.0.1      <none>               443/TCP  
sonarqube       LoadBalancer  10.100.254.151  a0d7e5425c6f24b65beca8dc46032860-718461882.ap-south-1.elb.amazonaws.com  80:31358/  
TCP,9001:32556/TCP  27h  
sonarqube-postgresql  ClusterIP    10.100.188.139  <none>               5432/TCP  
sonarqube-postgresql-hl  ClusterIP    None          <none>               5432/TCP  
27h  
prakash@prakash-Latitude-3420:~/Downloads$  
prakash@prakash-Latitude-3420:~/Downloads$  
prakash@prakash-Latitude-3420:~/Downloads$
```

Thanks