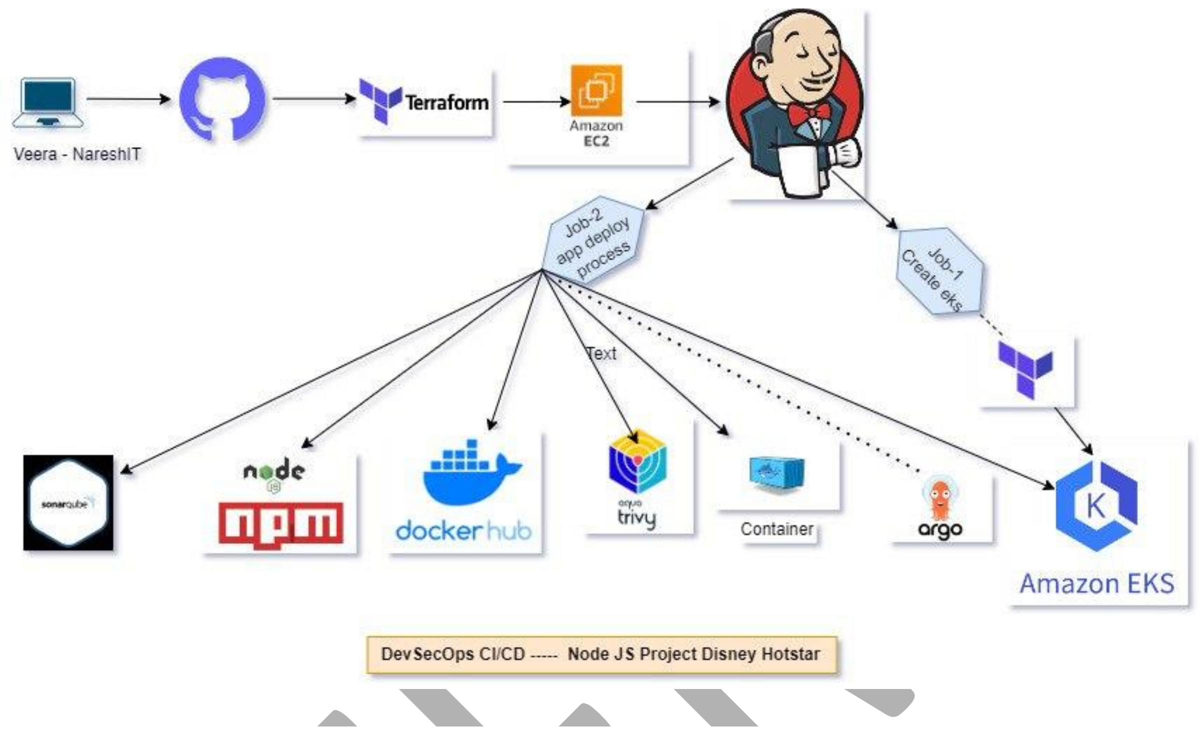
DevOps Project



Project reference Github

https://github.com/nareshdevopscloud/Hotstar-DevOps-Project-NodeJS

Create IAM user and configure credentials into local machine

Install Terraform and VScode into local machine

Create one folder to create terraform project with any name and add below files like

Install.sh

Main.tf

Refer https://github.com/nareshdevopscloud/devops-terraform/tree/main/project-terraform-devops-tools-install

Create file install.sh and copy paste below scrips into install.sh

```
#!/bin/bash

sudo yum update -y



#---------------git install ---------------



sudo yum install git -y





#-------java dependency for jenkins------------



sudo dnf install java-11-amazon-corretto -y



#------------jenkins install-------------

sudo wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-
2023.key

sudo yum install jenkins -y

sudo systemctl enable jenkins

sudo systemctl start jenkins
```

```
#---------------------------------install tomcat-----------------

#sudo wget url https://dlcdn.apache.org/tomcat/tomcat-
9/v9.0.83/bin/apache-tomcat-9.0.83.tar.gz

#sudo tar -xvzf apache-tomcat-9.0.83.tar.gz #untar

#cd apache-tomcat-9.0.83

#cd bin

#chmod +x startup.sh
```

```
#---------------------------Maven install -------------

sudo yum install maven -y
```

```
#---------------------------kubectl install ---------------

sudo curl -o kubectl https://amazon-eks.s3.us-west-
2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl

sudo chmod +x ./kubectl

sudo mv ./kubectl /usr/local/bin

# -----------------------------eksctl install--------------------------
------

sudo curl --silent --location
"https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$
(uname -s)_amd64.tar.gz" | tar xz -C /tmp

sudo mv /tmp/eksctl /usr/local/bin
```

```
#----------------------Trivy install---------------

sudo rpm -ivh
https://github.com/aquasecurity/trivy/releases/download/v0.48.3/trivy_0
.48.3_Linux-64bit.rpm




#----------------------sonarQube install------------------------------
----

sudo yum -y install wget nfs-utils

sudo wget -O /etc/yum.repos.d/sonar.repo
http://downloads.sourceforge.net/project/sonar-pkg/rpm/sonar.repo

sudo yum -y install sonar

#-----------------------JFROg-----------------------------

sudo wget https://releases.jfrog.io/artifactory/artifactory-
rpms/artifactory-rpms.repo -O jfrog-artifactory-rpms.repo;



sudo mv jfrog-artifactory-rpms.repo /etc/yum.repos.d/;



sudo yum update && sudo yum install jfrog-artifactory-oss -y



sudo systemctl start artifactory.service



#-------------------terraform install-------------------------



sudo wget
https://releases.hashicorp.com/terraform/1.7.2/terraform_1.7.2_linux_am
d64.zip
```

```
sudo unzip terraform_1.7.2_linux_amd64.zip

sudo mv terraform /usr/local/bin




#------------------Docker install-------------

#sudo amazon-linux-extras install docker #linux 2022

sudo yum install docker -y #linux 2023

sudo usermod -aG docker ec2-user

sudo usermod -aG docker jenkins

newgrp docker

sudo chmod 777 /var/run/docker.sock

sudo service docker start

#------------------sonar install by using docker---------------

docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

docker run -d --name tomcat -p 8089:8080 tomcat:lts-community
```

main.tf

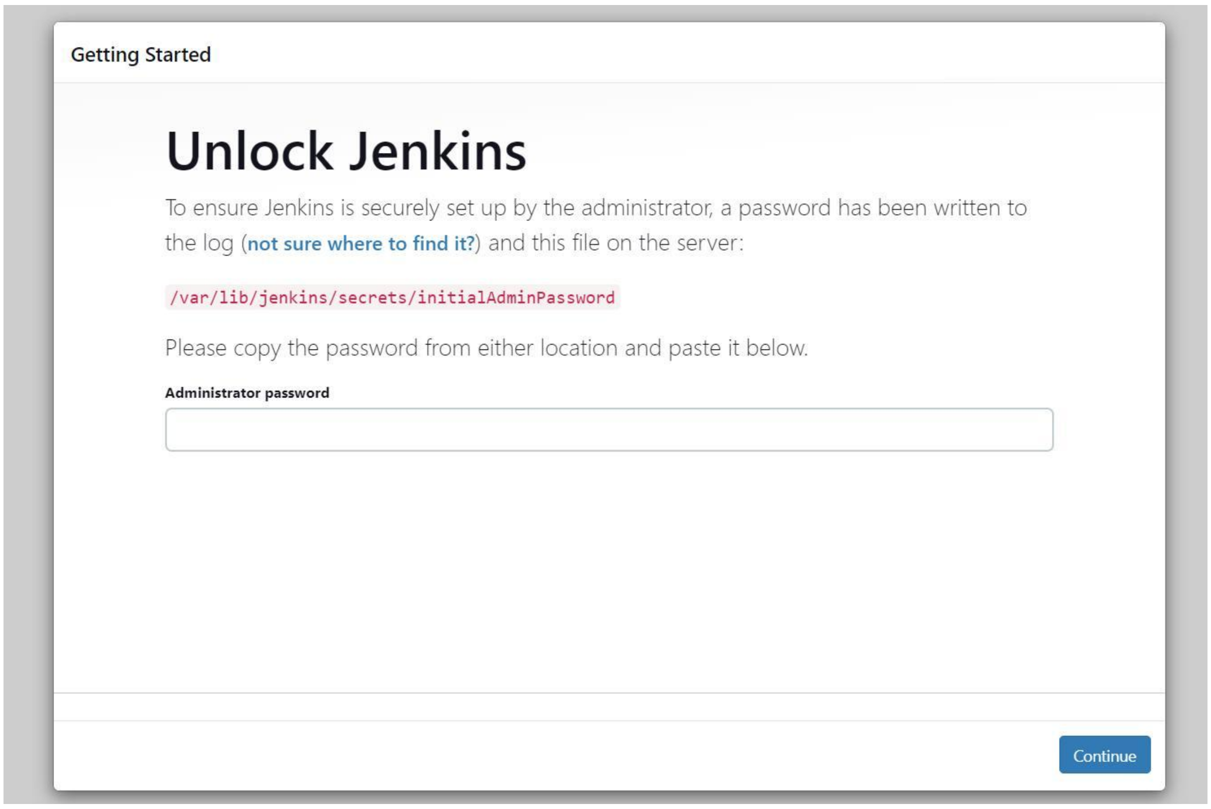
```
resource "aws_instance" "web" {

  ami                    = "ami-0277155c3f0ab2930"      #change ami id
for different region

  instance_type          = "t2.large"

  key_name               = "vscode"              #change key name as
per your setup
```

```
  vpc_security_group_ids = [aws_security_group.devops-project-veera.id]

  user_data              = templatefile("./install.sh", {})


  tags = {

    Name = "project-MainEc2"

  }


  root_block_device {

    volume_size = 40

  }

}


resource "aws_security_group" "devops-project-veera" {

  name        = "devops-project-veera"

  description = "Allow TLS inbound traffic"


  ingress = [

    for port in [22, 80, 443, 8080, 9000, 3000, 8082, 8081] : {

      description      = "inbound rules"

      from_port        = port

      to_port          = port

      protocol         = "tcp"
      cidr_blocks      = ["0.0.0.0/0"]

      ipv6_cidr_blocks = []

      prefix_list_ids  = []

      security_groups  = []

      self             = false

    }

  ]



  egress {

    from_port   = 0

    to_port     = 0

    protocol    = "-1"

    cidr_blocks = ["0.0.0.0/0"]

  }



  tags = {

    Name = "devops-project-veera"

  }

}
```

(Optional

If you want to configure ubuntu server refer below link for install.sh scripts to call from terraform https://github.com/nareshdevopscloud/devops-terraform/blob/main/ubuntu.sh)

```
<Ec2-ip:8080> #you will Jenkins login page
```


Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

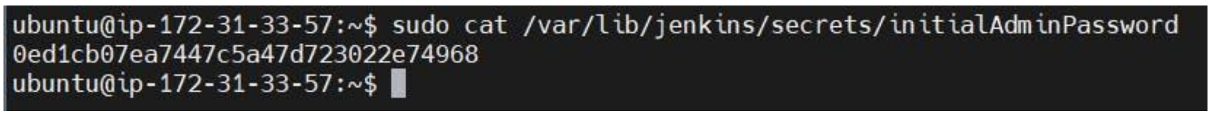
/var/lib/jenkins/secrets/initialAdminPassword

Please copy the password from either location and paste it below.

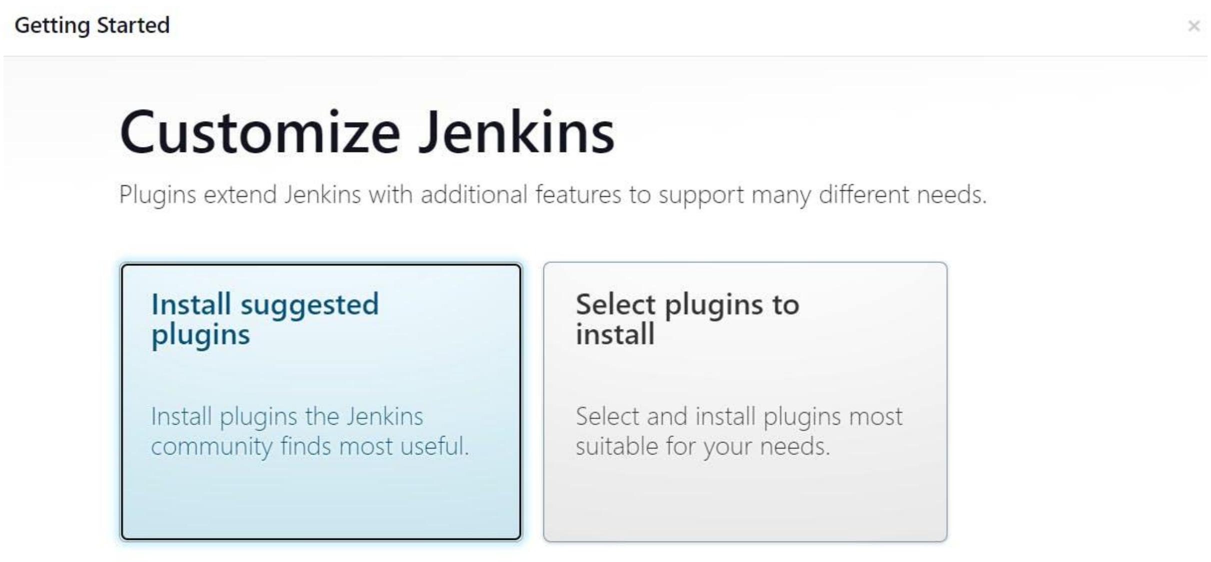
Administrator password

Connect your Instance to Putty or Mobaxtreme and provide the below command for the Administrator password

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```


```
ubuntu@ip-172-31-33-57:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
0ed1cb07ea7447c5a47d723022e74968
ubuntu@ip-172-31-33-57:~$
```

Now, install the suggested plugins.


Getting Started

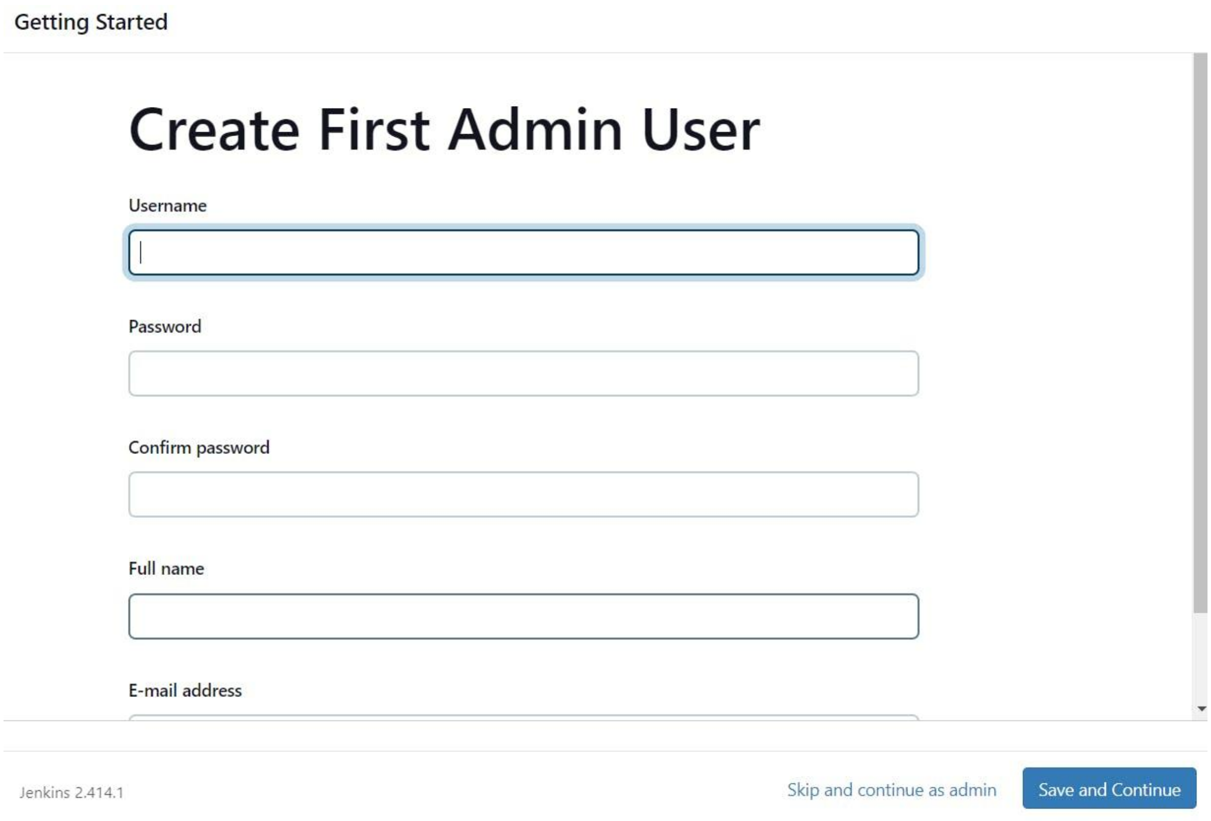
Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.

Jenkins will now get installed and install all the libraries.

Create an admin user


Getting Started

Create First Admin User

Username

Password

Confirm password

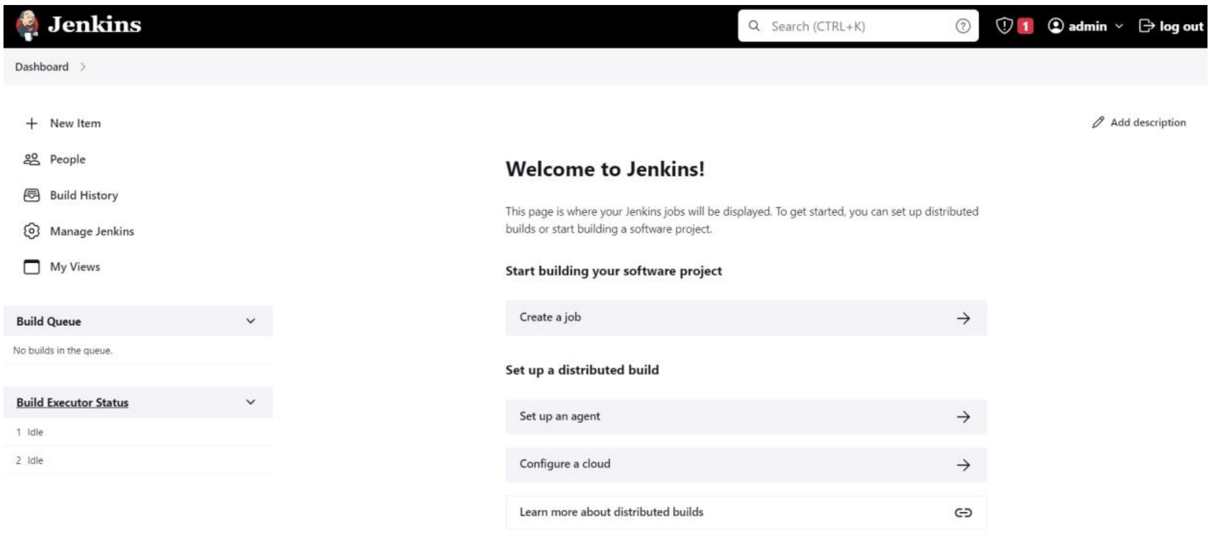
Full name

E-mail address

Jenkins 2.414.1

Click on save and continue.

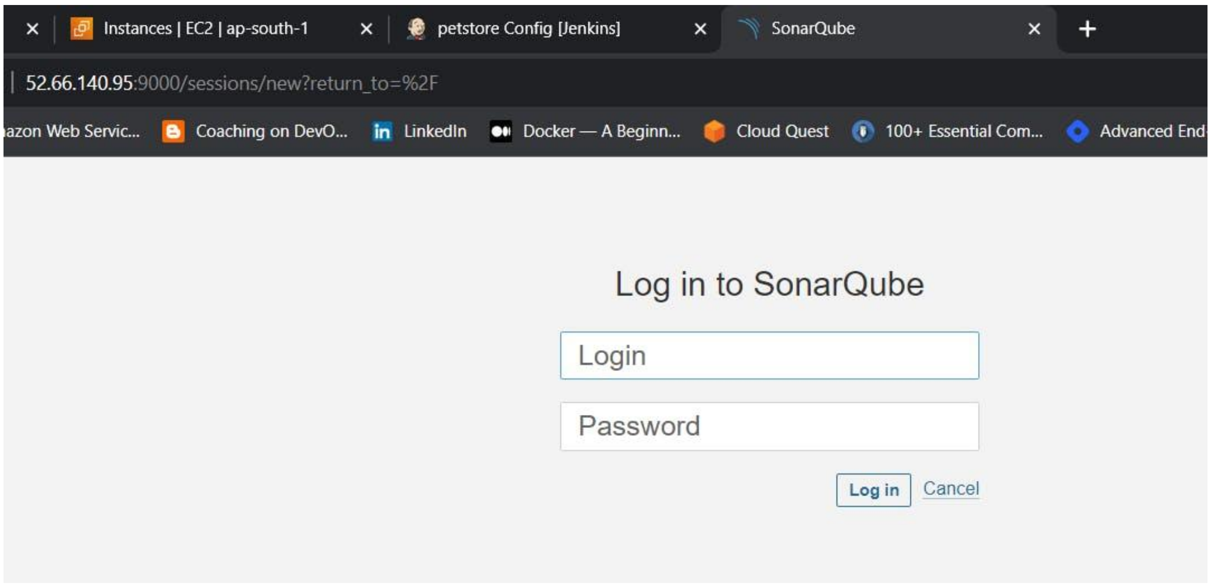
Jenkins Dashboard


Jenkins
Search (CTRL+K)
admin
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds

Sonar configuration Process

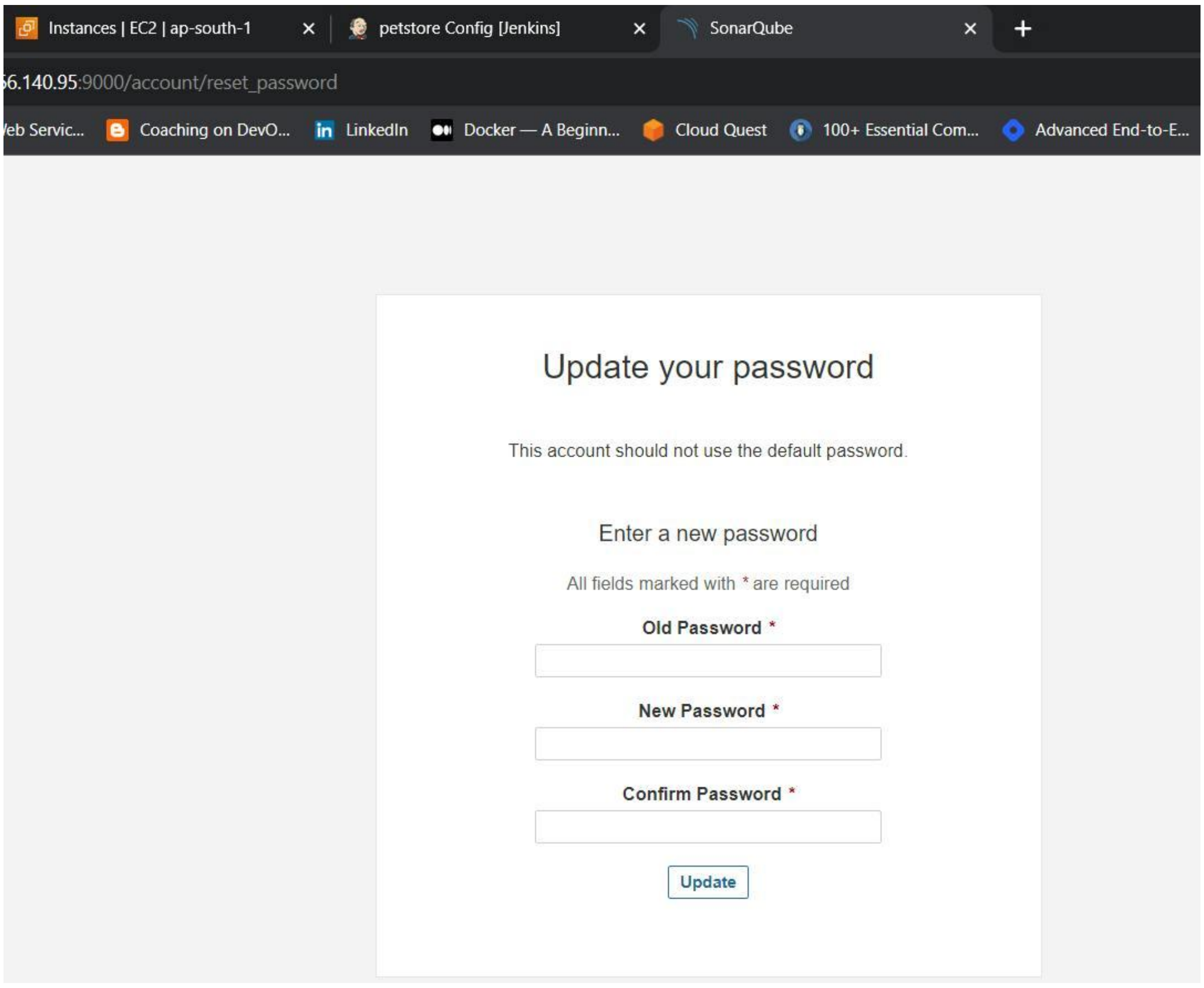
Now Copy the public IP again and paste it into a new tab in the browser with 9000

```
<ec2-ip:9000>  #runs sonar container
```



Enter username and password, click on login and change password

```
username admin

password admin
```


Instances | EC2 | ap-south-1
petstore Config [Jenkins]
SonarQube
6.140.95:9000/account/reset_password
/eb Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E...

Update your password

This account should not use the default password.

Enter a new password

All fields marked with * are required

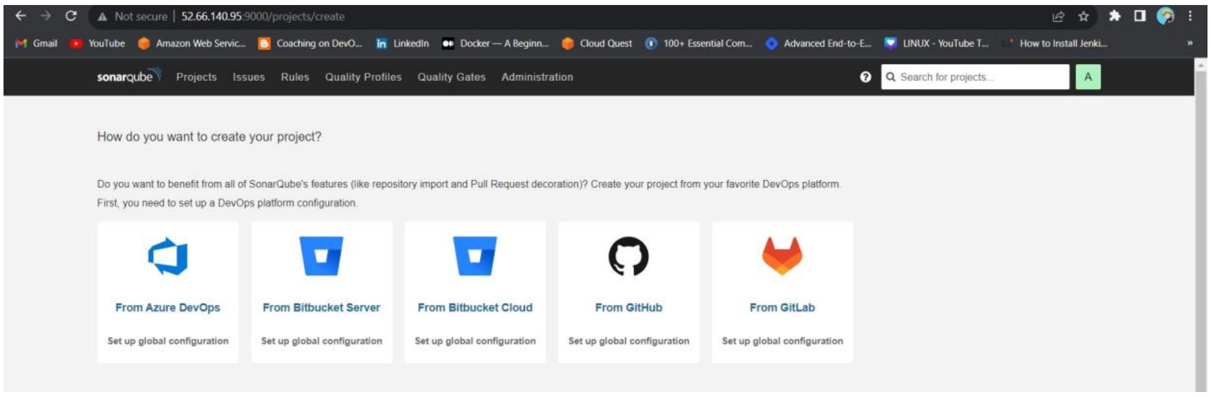
Old Password *

New Password *

Confirm Password *

Update

Update New password, This is Sonar Dashboard.


Not secure | 52.66.140.95:9000/projects/create
Gmail YouTube Amazon Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E... LINUX - YouTube T... How to Install Jenki...
sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration
Search for projects...
A

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform.
First, you need to set up a DevOps platform configuration.

From Azure DevOps
Set up global configuration

From Bitbucket Server
Set up global configuration

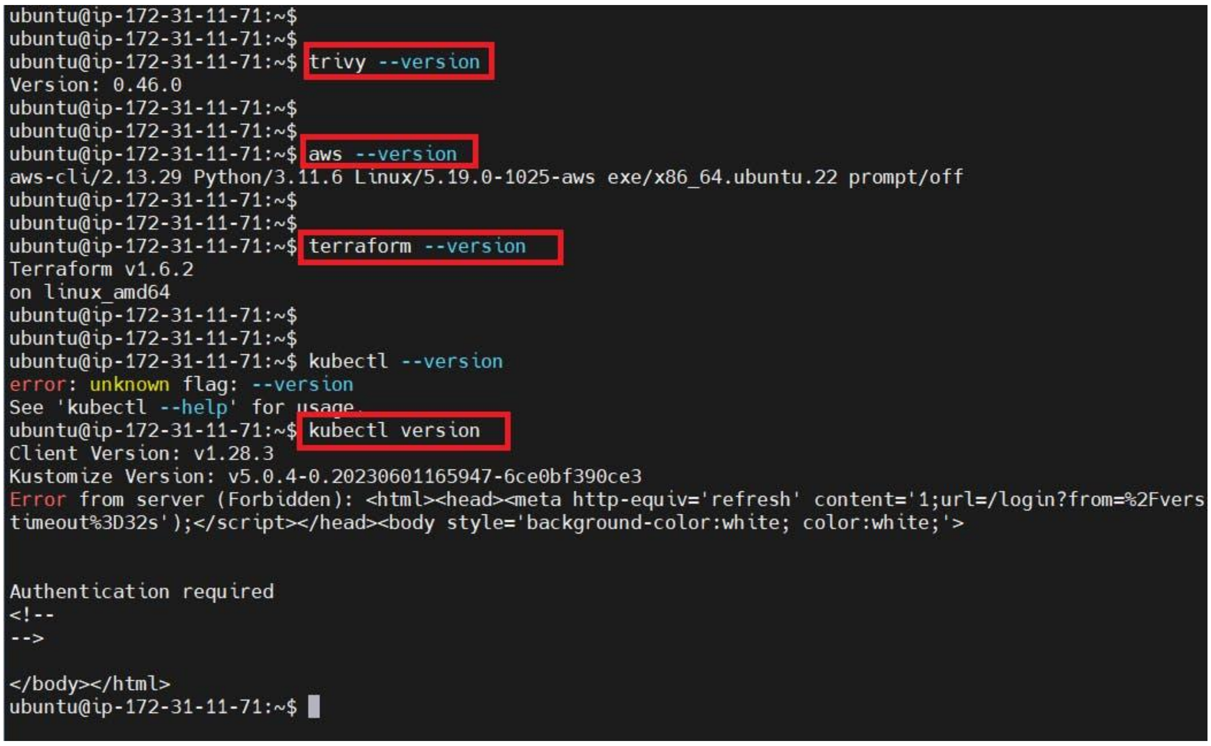
From Bitbucket Cloud
Set up global configuration

From GitHub
Set up global configuration

From GitLab
Set up global configuration

Now go to terminal and see whether it's installed docker, Terraform, Aws cli, Kubectl or not.

```
docker --version

aws --version

terraform --version

kubectl version
```



```
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ trivy --version
Version: 0.46.0
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ aws --version
aws-cli/2.13.29 Python/3.11.6 Linux/5.19.0-1025-aws exe/x86_64.ubuntu.22 prompt/off
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ terraform --version
Terraform v1.6.2
on linux_amd64
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ kubectl --version
error: unknown flag: --version
See 'kubectl --help' for usage.
ubuntu@ip-172-31-11-71:~$ kubectl version
Client Version: v1.28.3
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
Error from server (Forbidden): <html><head><meta http-equiv='refresh' content='1;url=/login?from=%2Fvers
timeout%3D32s');</script></head><body style='background-color:white; color:white;'>


Authentication required
<!--
-->

</body></html>
ubuntu@ip-172-31-11-71:~$
```

Step 3: Jenkins Job Configuration

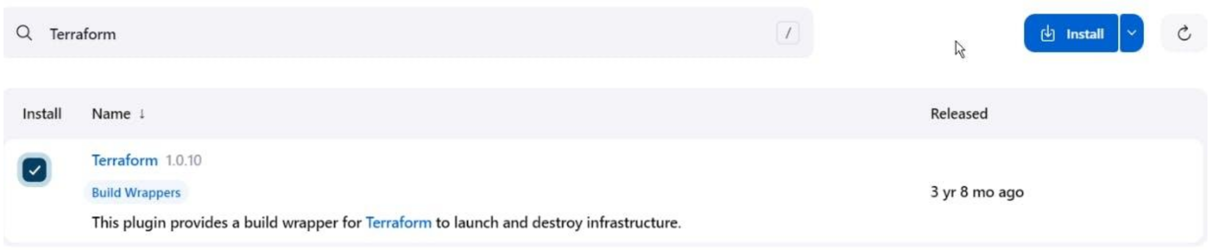
Step 3: EKS Provision job

Note: before keep it ready for EKS script we are going to run EKS by using Jenkins pipeline

That is done now go to Jenkins and add a terraform plugin to provision the AWS EKS using the Pipeline Job.

Go to Jenkins dashboard –> Manage Jenkins –> Plugins

Available Plugins, Search for Terraform and install it.

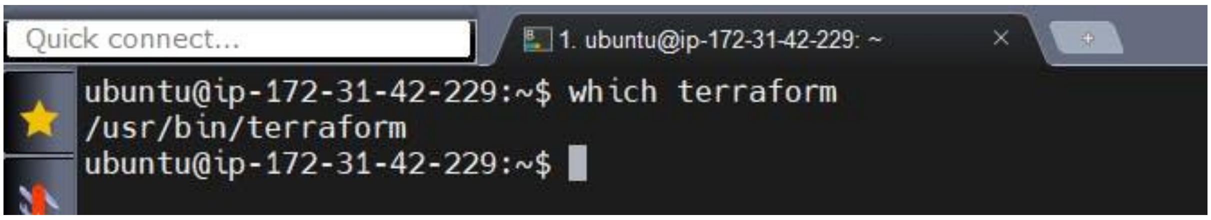


Terraform 1.0.10
Build Wrappers
This plugin provides a build wrapper for Terraform to launch and destroy infrastructure.
3 yr 8 mo ago

Go to Terminal and use the below command

let's find the path to our Terraform (we will use it in the tools section of Terraform)

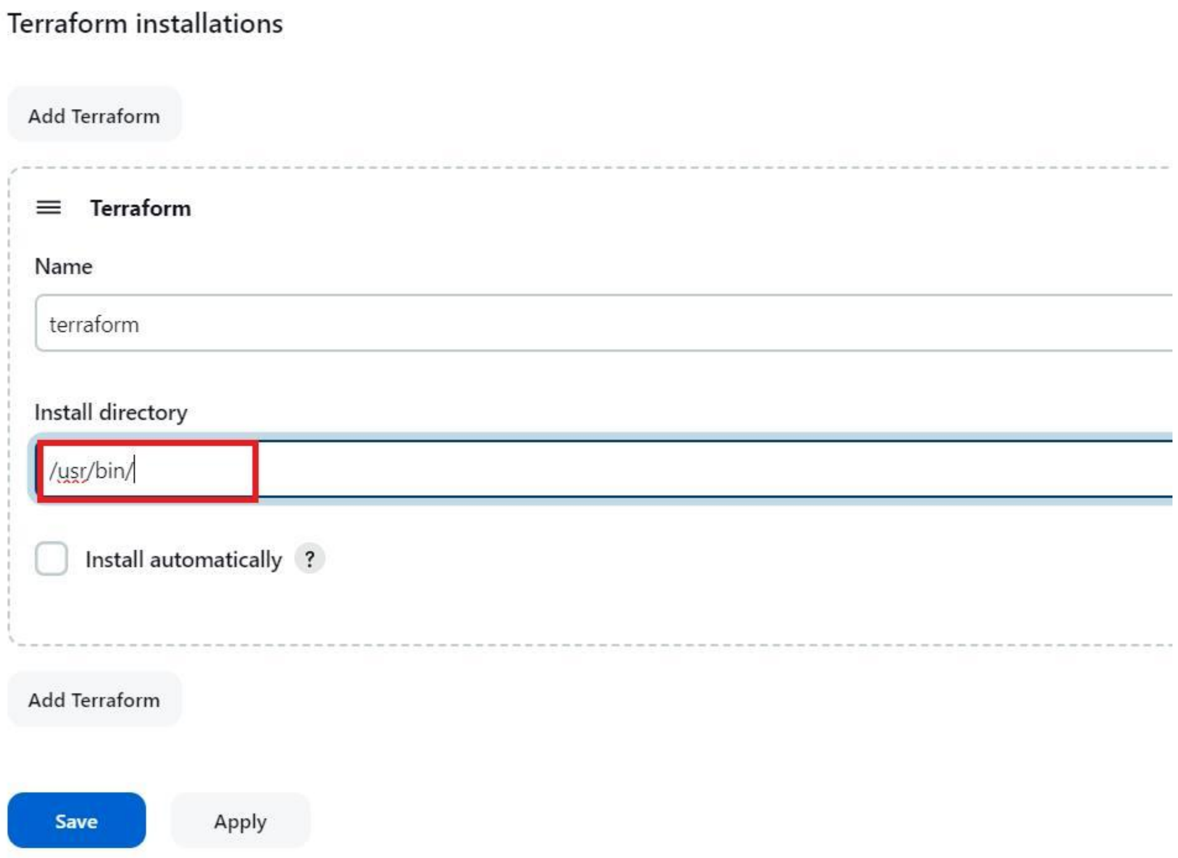
```
which terraform
```



```
ubuntu@ip-172-31-42-229:~$ which terraform
/usr/bin/terraform
ubuntu@ip-172-31-42-229:~$
```

Now come back to Manage Jenkins –> Tools

Add the terraform in Tools


Terraform installations

Terraform

Name: terraform

Install directory: /usr/bin/

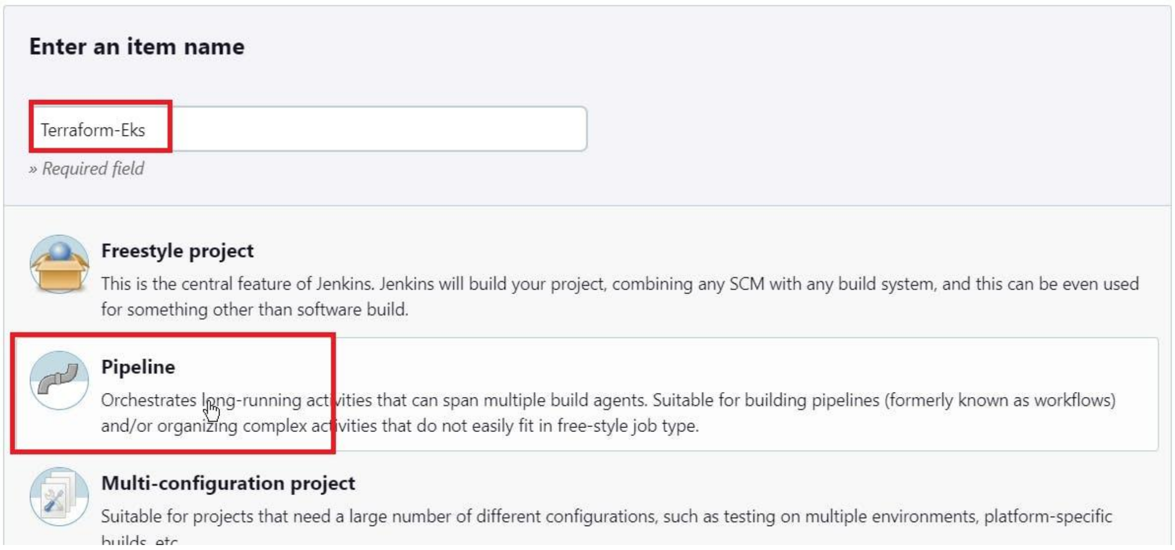
Install automatically ?

Apply and save.

Now in our EKS script we have configured backeen.tf (to maintain state file remote(s3)

GIVE YOUR S3 BUCKET NAME IN THE BACKEND.TF

Now create a new job for the Eks provision


Enter an item name

Terraform-Eks

» Required field

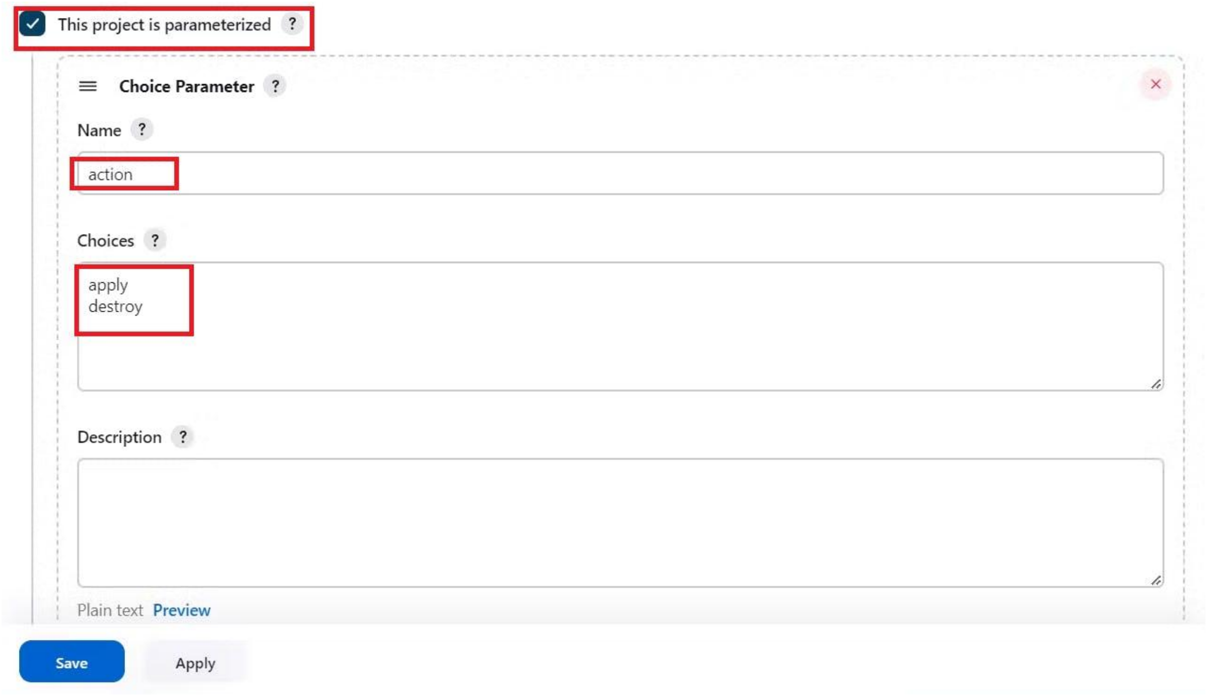
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

I want to do this with build parameters to apply and destroy while building only.

you have to add this inside job like the below image


✓ This project is parameterized

Choice Parameter

Name: action

Choices:
apply
destroy

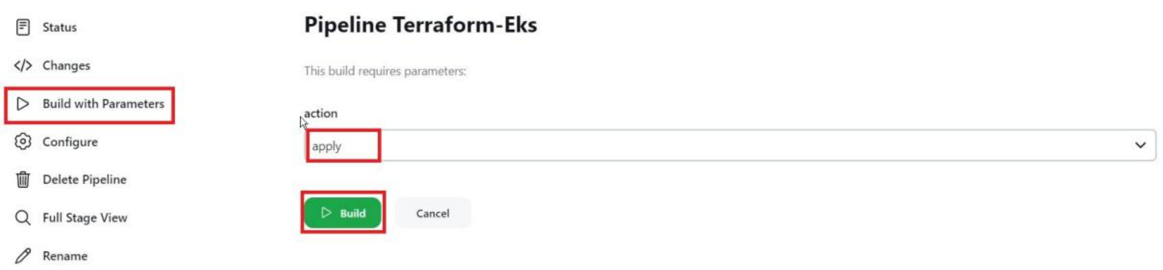
Description:

Let's add a pipeline

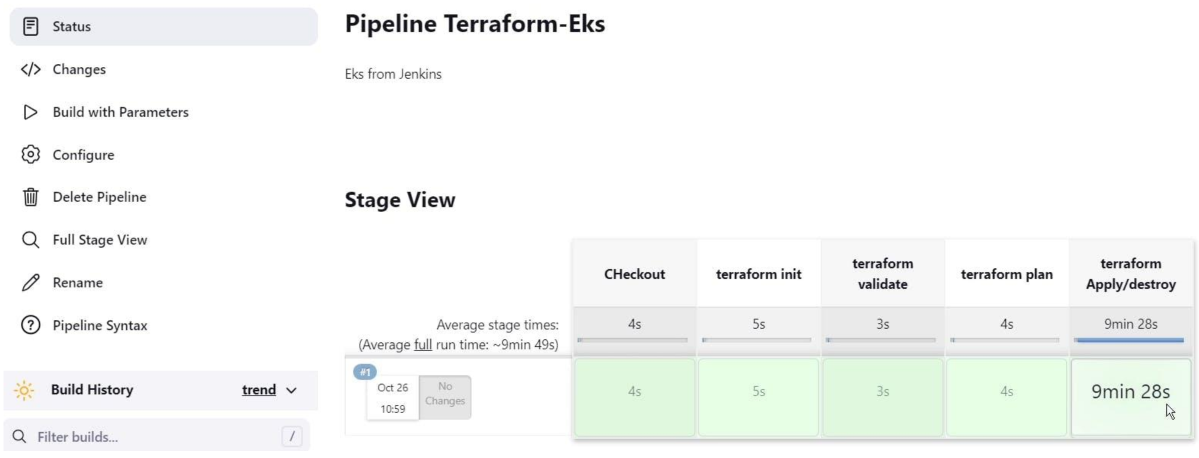
```
pipeline{

    agent any
    stages {
        stage('Checkout from Git'){
            steps{
                <githuburl>
            }
        }
        stage('Terraform version'){
             steps{
                 sh 'terraform --version'
             }
        }
        stage('Terraform init'){
             steps{
                 dir('EKS_TERRAFORM') {
                      sh 'terraform init --reconfigure'
                   }
             }
        }
        stage('Terraform validate'){
             steps{
                 dir('EKS_TERRAFORM') {
                      sh 'terraform validate'
```

```
                    }
                }
            }
            stage('Terraform plan'){
                steps{
                    dir('EKS_TERRAFORM') {
                        sh 'terraform plan'
                    }
                }
            }
            stage('Terraform apply/destroy'){
                steps{
                    dir('EKS_TERRAFORM') {
                        sh 'terraform ${action} --auto-approve'
                    }
                }
            }
        }
}
```

let’s apply and save and Build with parameters and select action as apply

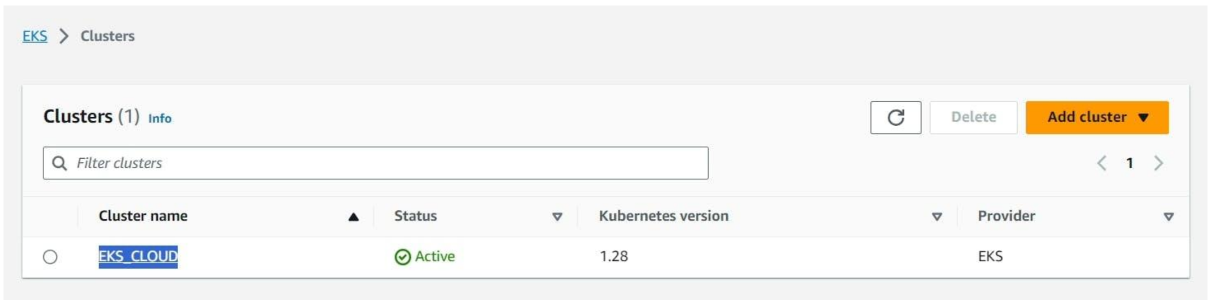


Stage view it will take max 10mins to provision



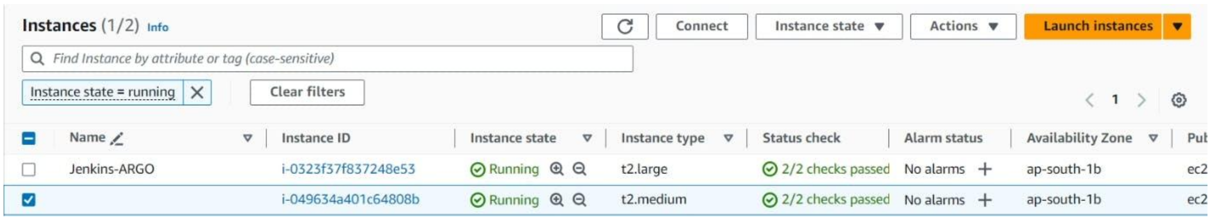
	CHeckout	terraform init	terraform validate	terraform plan	terraform Apply/destroy
Average stage times: (Average full run time: ~9min 49s)	4s	5s	3s	4s	9min 28s
#1 Oct 26 10:59 No Changes	4s	5s	3s	4s	9min 28s

Check in Your Aws console whether it created EKS or not.



Cluster name	Status	Kubernetes version	Provider
EKS_CLOUD	Active	1.28	EKS

Ec2 instance is created for the Node group



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Pul
Jenkins-ARGO	i-0323f37f837248e53	Running	t2.large	2/2 checks passed	No alarms	ap-south-1b	ec2
	i-049634a401c64808b	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b	ec2

Step 3: Hotstar job

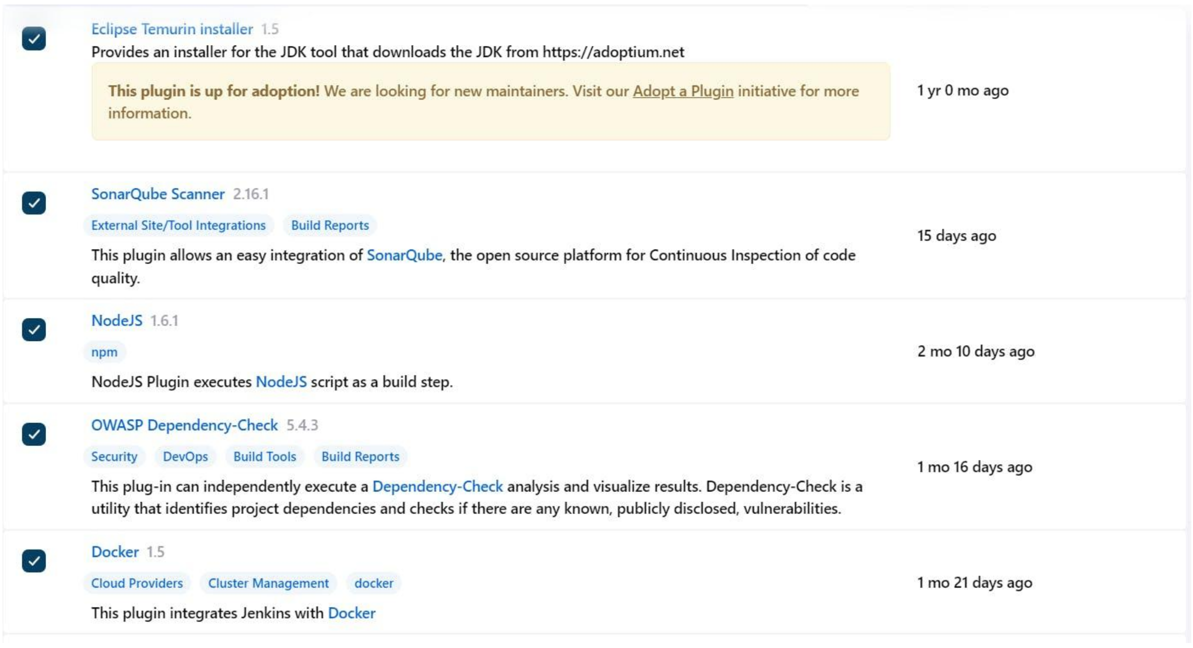
Plugins installation & setup (Java, Sonar, Nodejs, owasp, Docker)

Go to Jenkins dashboard

Manage Jenkins –> Plugins –> Available Plugins

Search for the Below Plugins

```
Eclipse Temurin installer

Sonarqube Scanner

NodeJs

Docker

Docker Commons

Docker Pipeline

Docker API

Docker-build-step
```



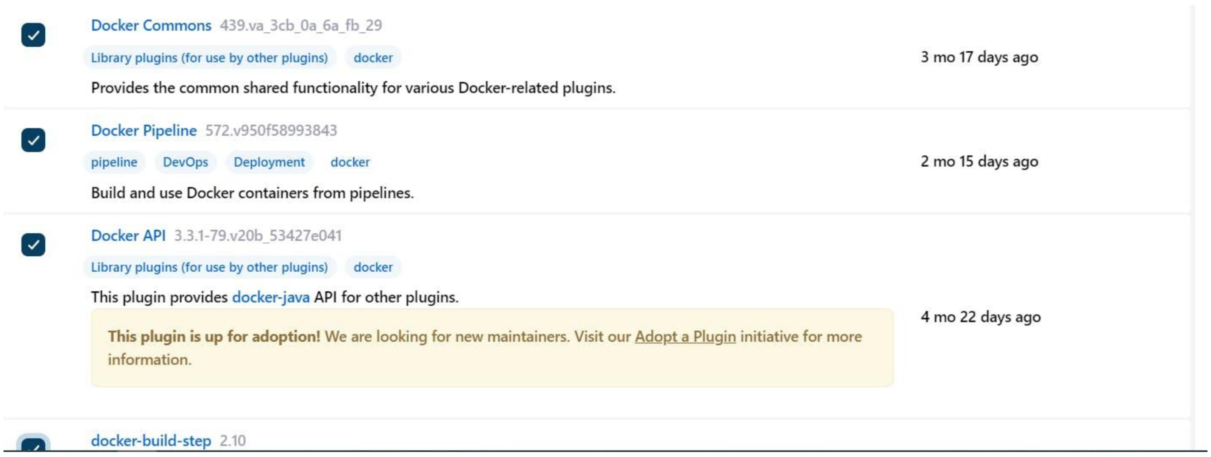
Eclipse Temurin installer 1.5
Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
1 yr 0 mo ago

SonarQube Scanner 2.16.1
External Site/Tool Integrations Build Reports
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
15 days ago

NodeJS 1.6.1
npm
NodeJS Plugin executes NodeJS script as a build step.
2 mo 10 days ago

OWASP Dependency-Check 5.4.3
Security DevOps Build Tools Build Reports
This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.
1 mo 16 days ago

Docker 1.5
Cloud Providers Cluster Management docker
This plugin integrates Jenkins with Docker
1 mo 21 days ago



Docker Commons 439.va_3cb_0a_6a_fb_29
Library plugins (for use by other plugins) docker
Provides the common shared functionality for various Docker-related plugins.
3 mo 17 days ago

Docker Pipeline 572.v950f58993843
pipeline DevOps Deployment docker
Build and use Docker containers from pipelines.
2 mo 15 days ago

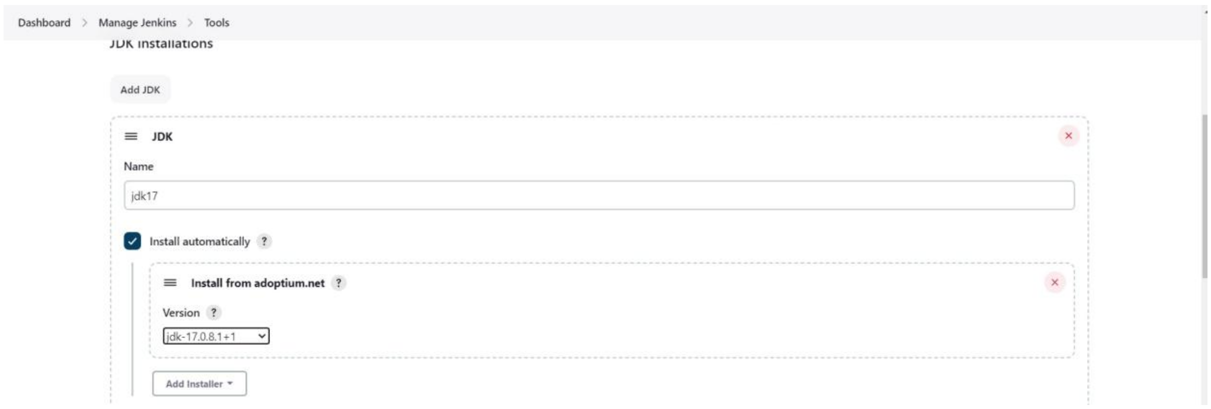
Docker API 3.3.1-79.v20b_53427e041
Library plugins (for use by other plugins) docker
This plugin provides docker-java API for other plugins.
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
4 mo 22 days ago

docker-build-step 2.10

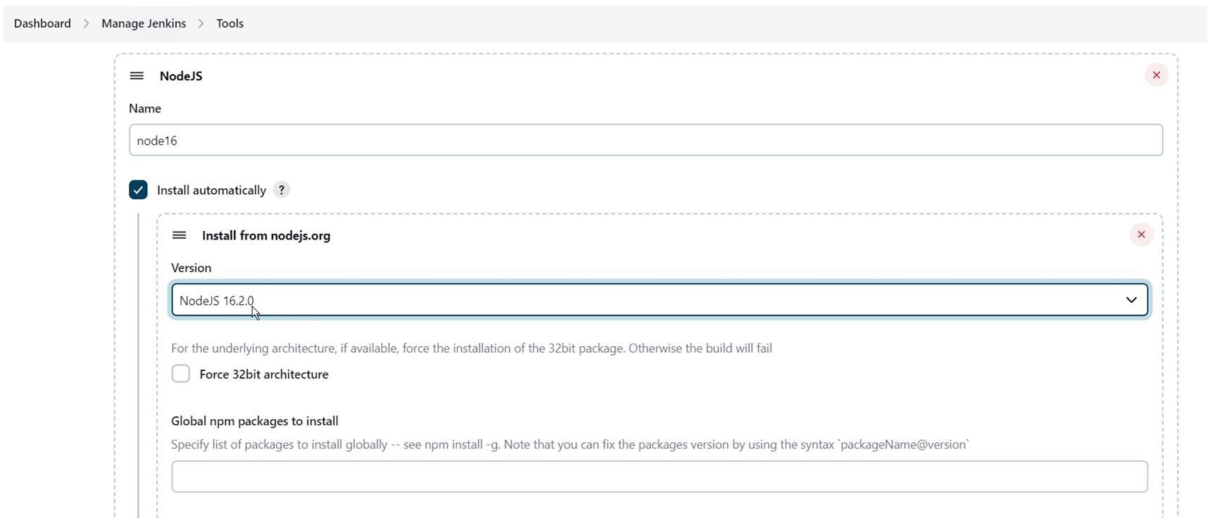
Configure in Global Tool Configuration

Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(16)→ Click on Apply and Save

NOTE: USE ONLY NODE JS 16

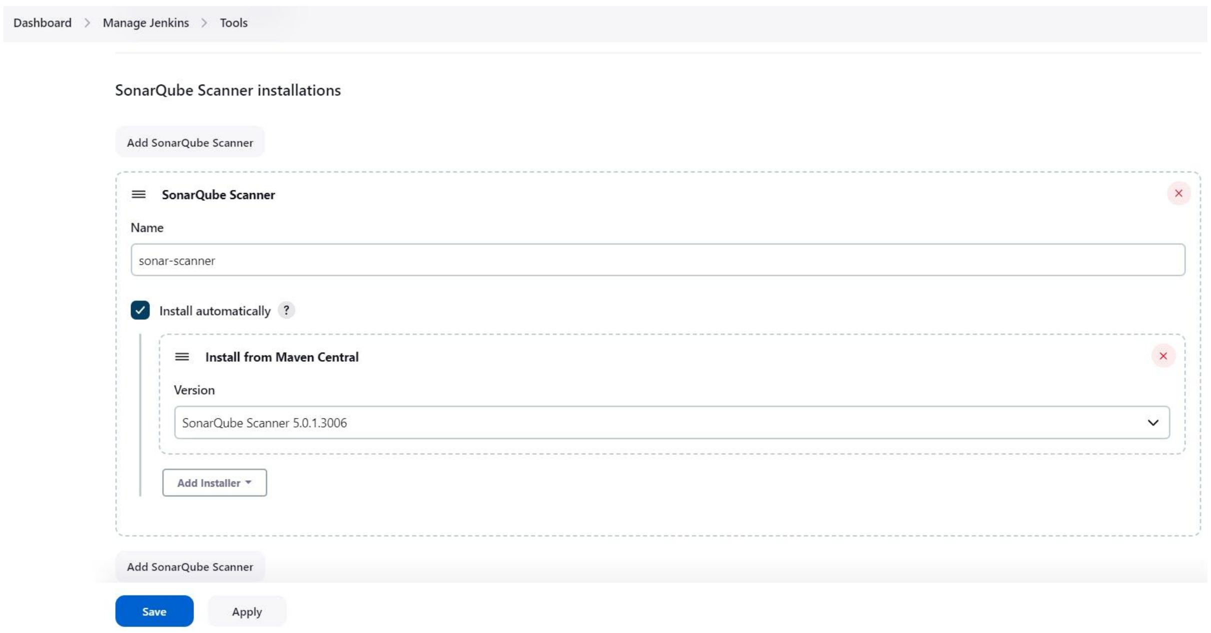


Dashboard > Manage Jenkins > Tools
JDK installations
Add JDK
JDK
Name: jdk17
Install automatically ✓
Install from adoptium.net
Version: jdk-17.0.8.1+1
Add Installer



Dashboard > Manage Jenkins > Tools
NodeJS
Name: node16
Install automatically ✓
Install from nodejs.org
Version: NodeJS 16.2.0
For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail
Force 32bit architecture
Global npm packages to install
Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

For Sonarqube use the latest version


Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

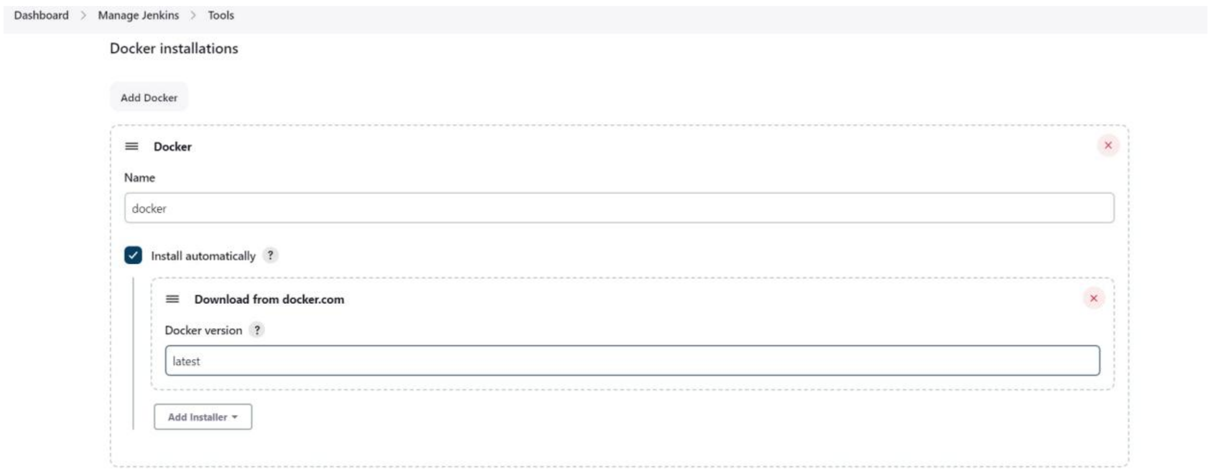
Name: sonar-scanner

Install automatically ✓

Install from Maven Central

Version: SonarQube Scanner 5.0.1.3006

Use the latest version of Docker


Dashboard > Manage Jenkins > Tools

Docker installations

Name: docker

Install automatically ✓

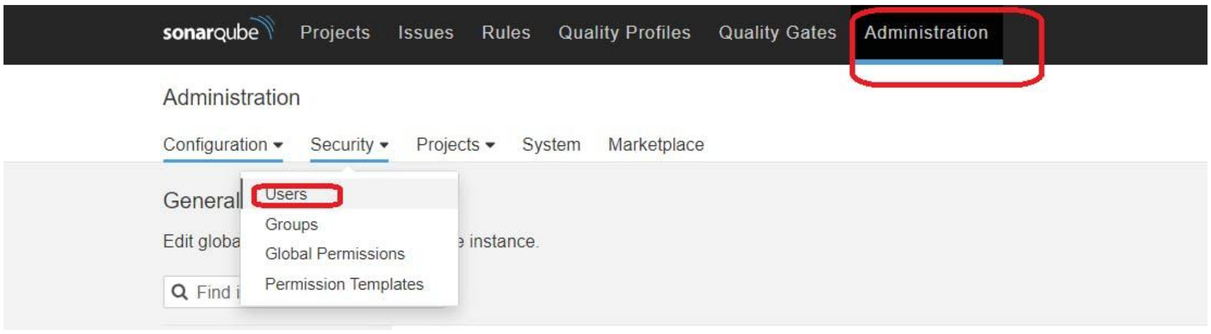
Download from docker.com

Docker version: latest

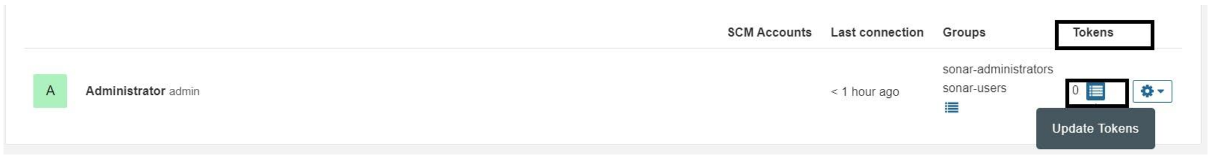
Click apply and save.

Configure Sonar Server in Manage Jenkins

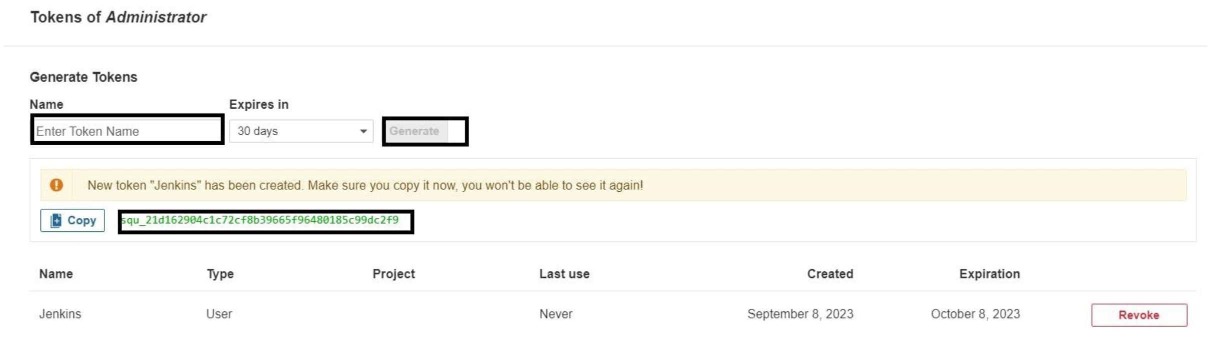
Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server. Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token



click on update Token



Create a token with a name and generate


Tokens of *Administrator*

Generate Tokens

Name: Enter Token Name | Expires in: 30 days | Generate

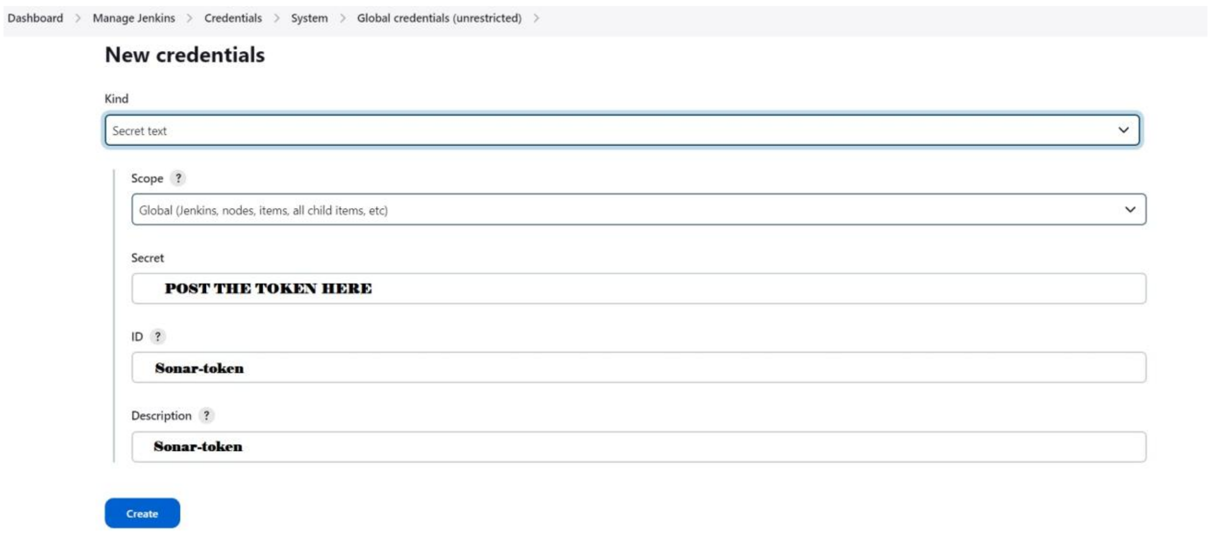
New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

Copy `squ_21d162904c1c72cf8b39665f96480185c99dc2f9`

Name	Type	Project	Last use	Created	Expiration	
Jenkins	User		Never	September 8, 2023	October 8, 2023	Revoke

copy Token

Goto Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this


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

New credentials

Kind: Secret text

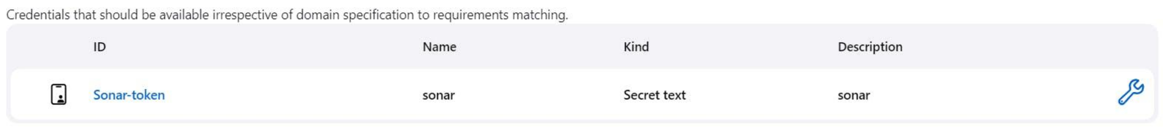
Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: POST THE TOKEN HERE

ID: Sonar-token

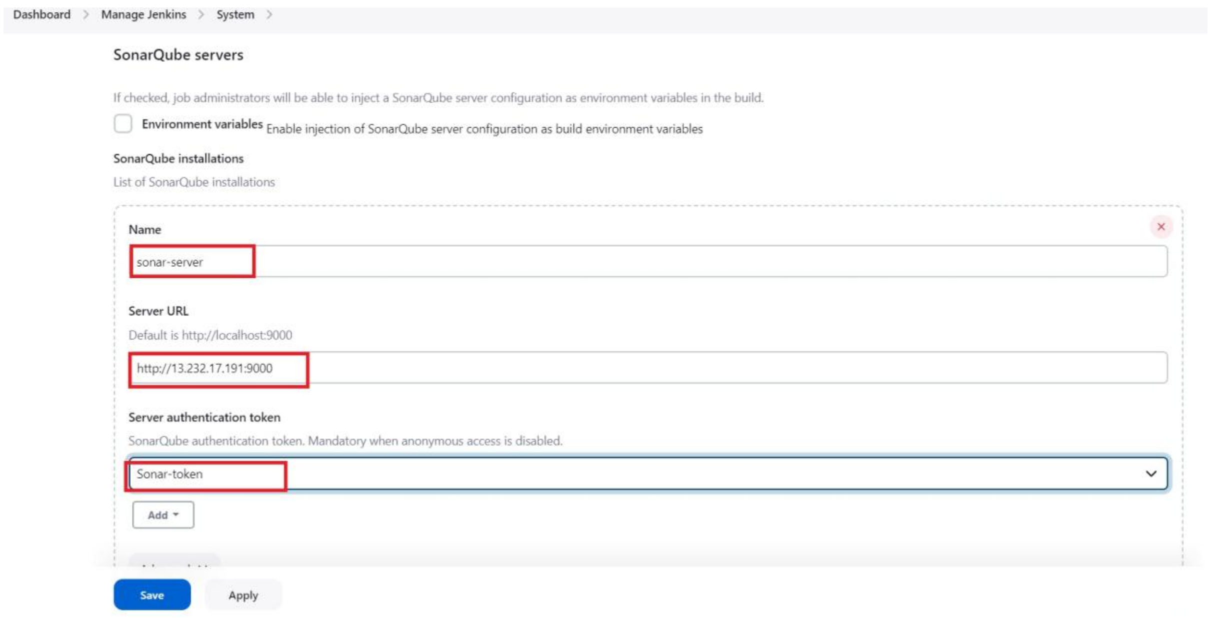
Description: Sonar-token

You will this page once you click on create


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

ID	Name	Kind	Description
Sonar-token	sonar	Secret text	sonar

Now, go to Dashboard → Manage Jenkins → System and Add like the below image.


Dashboard > Manage Jenkins > System >

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

Name: sonar-server

Server URL
Default is http://localhost:9000
http://13.232.17.191:9000

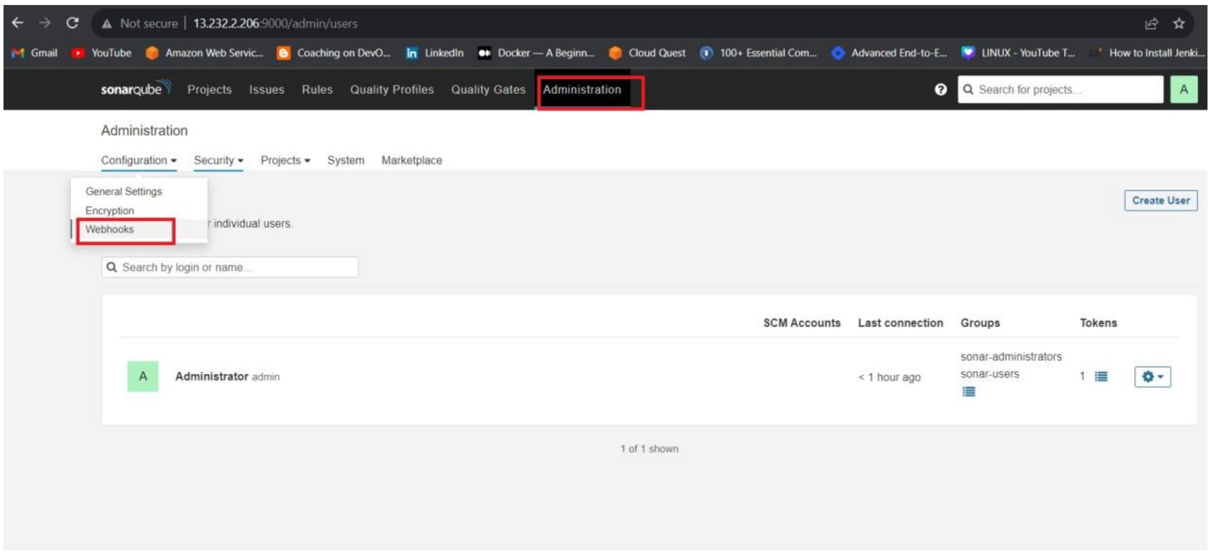
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
Sonar-token

Click on Apply and Save

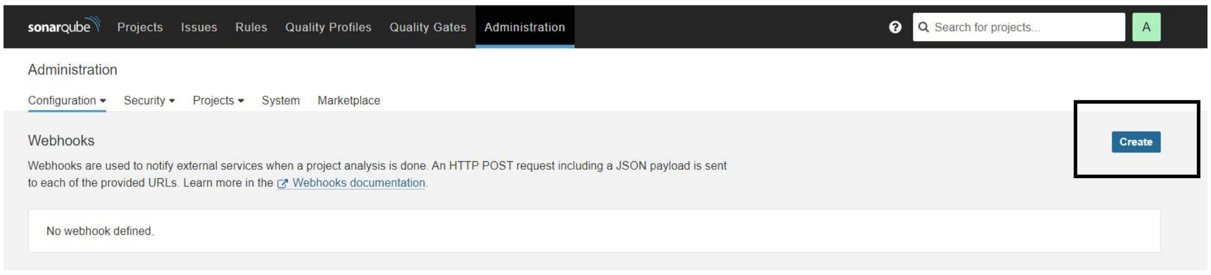
Process to Adding Quality Gates In the SonarQube Dashboard

Quality gate allows to next process if code is pass for quality checks

Administration–> Configuration–>Webhooks



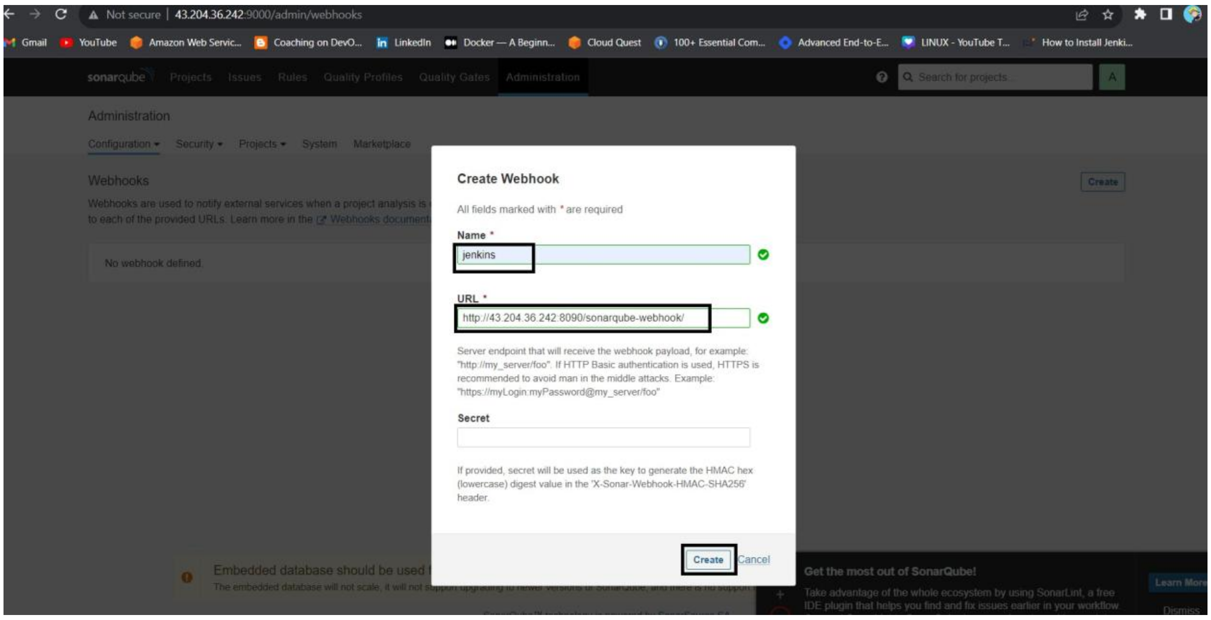
Click on Create



Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the Webhooks documentation.

Add details

```
#in url section of quality gate

<http://jenkins-public-ip:8080>/sonarqube-webhook/>
```



Create Webhook

All fields marked with * are required

Name * jenkins

URL * http://43.204.36.242:8090/sonarqube-webhook/

Server endpoint that will receive the webhook payload, for example: "http://my_server/foo". If HTTP Basic authentication is used, HTTPS is recommended to avoid man in the middle attacks. Example: "https://myLogin:myPassword@my_server/foo"

Secret

If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

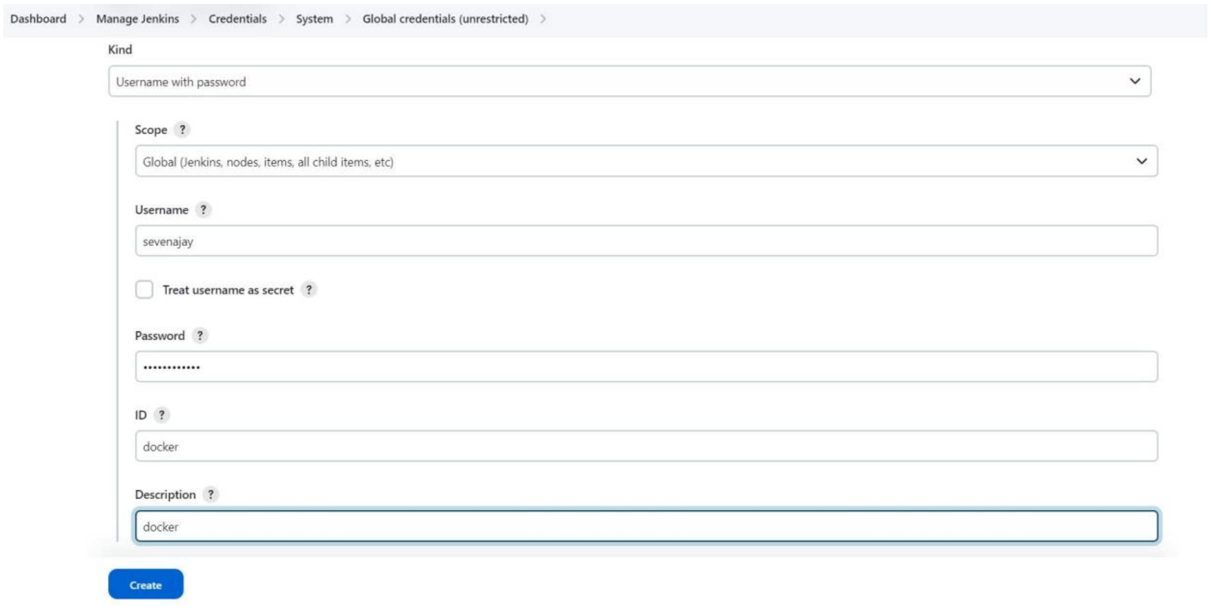
Docker hub process or ECR process

Considering Docker hub

Now add Docker credentials to the Jenkins to log in and push the image

Manage Jenkins –> Credentials –> global –> add credential

Add DockerHub Username and Password under Global Credentials



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

Kind: Username with password

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

Username: sevenajay

☐ Treat username as secret

Password: ••••••••••••

ID: docker

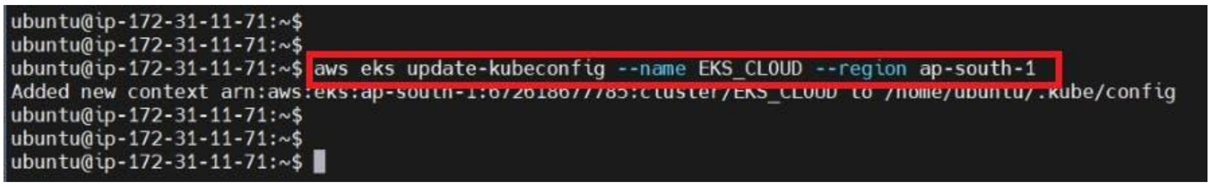
Description: docker

Deployment Process on EKS

Go to terminal of your Jenkins and enter the below command

```
aws eks update-kubeconfig --name <CLUSTER NAME> --region <CLUSTER
REGION>
```

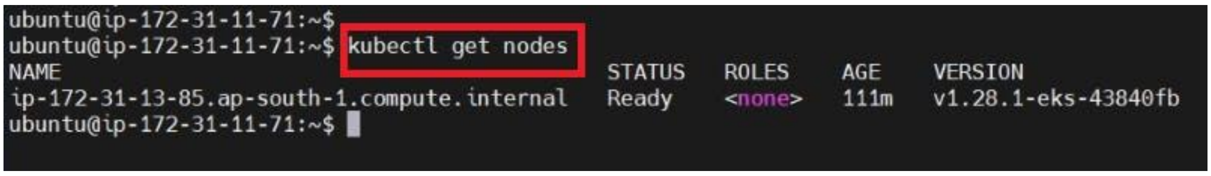
```
aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
```



```
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
Added new context arn:aws:eks:ap-south-1:672618677785:cluster/EKS_CLOUD to /home/ubuntu/.kube/config
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$
```

Let's see the nodes

```
kubectl get nodes
```



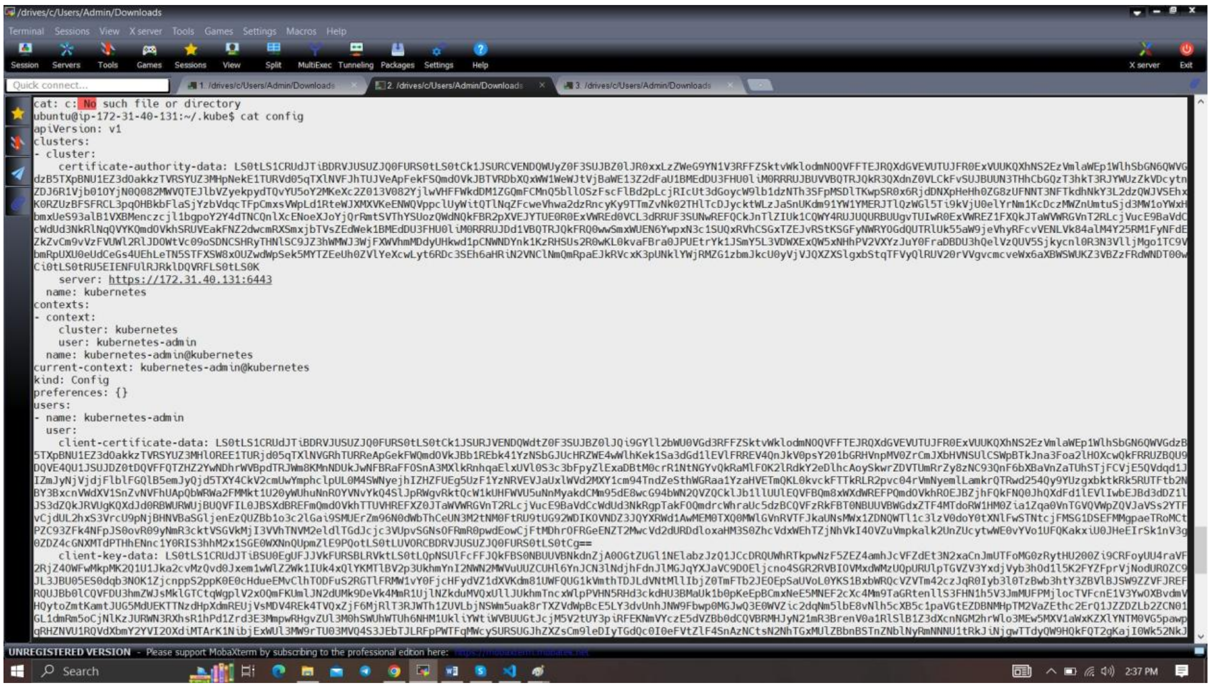
```
ubuntu@ip-172-31-11-71:~$
ubuntu@ip-172-31-11-71:~$ kubectl get nodes
NAME                                         STATUS   ROLES    AGE    VERSION
ip-172-31-13-85.ap-south-1.compute.internal  Ready    <none>   111m   v1.28.1-eks-43840fb
ubuntu@ip-172-31-11-71:~$
```

Now Give this command in CLI

```
cat /root/.kube/config
```

copy content from api version -- to --command aws

Copy the config file to Jenkins master or the local file manager and save it

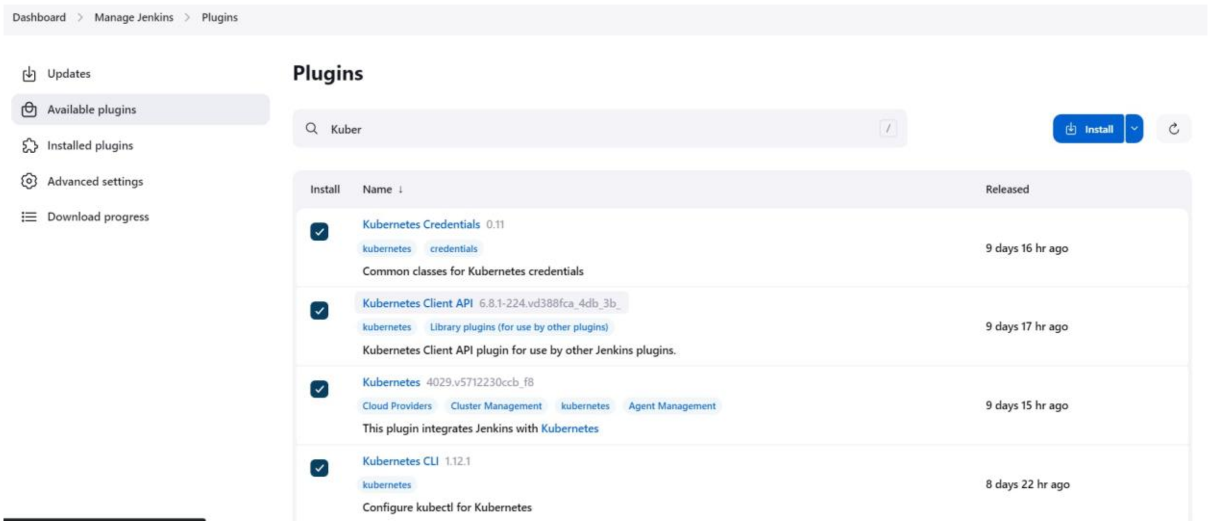


```
cat: c: No such file or directory
ubuntu@ip-172-31-40-131:~/.kube$ cat config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJR0xxLzZWeG9YN1V3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVG
dzB5TXpBNU1EZ3dOakkzTVRSYUZ3MHpNekE1TURVd05qTXlNVFJhTUJVeApFekFSQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0liM0RRRUJBUVVBQTRJQkR3QXdnZ0VLCkFvSUJBUUN3THhCbGQzT3hkT3RJYWUzZkVDcytn
ZDJ6R1Vjb01OYjN0Q082MWVQTEJlbVZyekpydTQvYU5oY2MKeXc2Z013V082YjlwVHFFWkdDM1ZGQmFCMnQ5bllOSzFscFlBd2pLcjRIcUt3dGoycW9lb1dzNTh3SFpMSDlTKwpSR0x6RjdDNXpHeHh0ZG8zUFNNT3NFTkdhNkY3L2dzQWJVSEhx
K0RZUzBFSFRCL3pqOHBkbFlaSjYzbVdqcTFpCmxsVWpLd1RteWJXMXVKeENWQVppclUyWitQTlNqZFcweVhwa2dzRncyKy9TTmZvNk02THlTcDJycktWLzJaSnUKdm91YW1YMERJTlQzWGl5Ti9kVjU0elYrNm1KcDczMWZnUmtuSjd3MW1oYWxH
bmxUeS93alB1VXBMenczcjl1bgpoY2Y4dTNCQnlXcENoeXJoYjQrRmtSVThYSUozQWdNQkFBR2pXVEJYTUE0R0ExVWREd0VCL3dRRUF3SUNwREFQCkJnTlZIUk1CQWY4RUJUQURBUUgvTUIwR0ExVWREZ1FXQkJTaWVWRGVnT2RLcjVucE9BaVdC
cWdUd3NkRlNqQVYKQmdOVkhSRUVEakFNZ2dwcmRXSmxjbTVsZEdWek1BMEdDU3FHU0liM0RRRUJDd1VBQTRJQkFRQ0wwSmxWUEN6YwpxN3c1SUQxRVhCSGxTZEJvRStKSGFyNWRYOGdQUTRlUk55aW9jeVhyRFcvVENLVk84alM4Y25RM1FyNFdE
ZkZvCm9vVzFVUWl2RlJDOWtVc09oSDNCSHRyTHNlSC9JZ3hWMWJ3WjFXWVhmMDdyUHkwd1pCNWNDYnk1KzRHSUs2R0wKL0kvaFBra0JPUEtrYk1JSmY5L3VDWXExQW5xNHhPV2VXYzJuY0FraDBDU3hQelVzQUV5Sjkycnl0R3N3VlljMgo1TC9V
bmRpUXU0eUdCeGs4UEhLeTN5STFXSW8xOUZwdWpSek5MYTZEeUh0ZVlYeXcwLyt6RDc3SEh6aHRiN2VNClNmQmRpaEJkRVcxK3pUNklYWjRMZG1zbmJkcU0yVjVJQXZXSlgxbStqTFVyQlRUV20rVVgvcmcveWx6aXBWSWUKZ3VBZzFRdWNDT00w
Ci0tLS0tRU5EIENFUlRJRklDQVRFLS0tLS0K
    server: https://172.31.40.131:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURJVENDQWdtZ0F3SUJBZ0lJQi9GYll2bWU0VGd3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB
5TXpBNU1EZ3dOakkzTVRSYUZ3MHlOREE1TURjd05qTXlNVGRhTURReApGekFWQmdOVkJBb1REbk41YzNSbGJUcHRZWE4wWlhKek1Sa3dGd1lEVlFRREV4QnJkV0psY201bGRHVnpMV0ZrCmJXbHVNSUlCSWpBTkJna3Foa2lHOXcwQkFRRUZBQU9
DQVE4QU1JSUJDZ0tDQVFFQTZHZ2YwNDhrWVBpdTRJWm8KMnNDUkJwNFBRaFFOSnA3MXlkRnhqaElxUVl0S3c3bFpyZlExaDBtM0crR1NtNGYvQkRaMlFOK2lRdkY2eDlhcAoySkwrZDVTUmRrZy8zNC93QnF6bXBaVnZaTUhSTjFCVjE5QVdqd1J
IZmJyNjVjdjFlblFGQlB5emJyQjd5TXY4CkV2cmUwYmphclpUL0M4SWNyejhIZHZFUEg5UzF1YzNRVEVJaUxlWVd2MXY1cm94TndZeSthWGRaa1YzaHVETmQKL0kvckFTTkRLR2pvc04rVmNyemlLamkrQTRwd254Qy9YUzgxbktkRk5RUTFtb2N
BY3BxcnVWdXV1SnZvNVFhUApQbWRWa2FMMkt1U20yWUhuNnROYVNvYkQ4SlJpRWgvRktQcW1kUHFWVU5uNnMyakdCMm95dE8wcG94bWN2QVZQCklJb1llUUlEQVFBQm8xWXdWREFPQmdOVkhROEJBZjhFQkFNQ0JhQXdFd1lEVlIwbEJBd3dDZ1l
JS3dZQkJRVUgKQXdJd0RBWURWUjBUQVFIL0JBSXdBREFmQmdOVkhTTUVHREFXZ0JTaWVWRGVnT2RLcjVucE9BaVdCcWdUd3NkRgpTakFOQmdrcWhraUc5dzBCQVFzRkFBT0NBUUVBWGdxZTF4MTdoRW1HM0Zia1Zqa0VnTGVQVWpZQVJaVSs2YTF
vCjdUL2hxS3VrcU9pNjBHNVBaSGljenEzQUZBb1o3c2lGai9SMUErZm96N0dWbThCeUN3M2tNM0FtRU9tUG92WDIKOVNDZ3JQYXRWd1AwMEM0TXQ0MWlGVnRVTFJkaUNsMWx1ZDNQWTl1c3lzV0doY0tXNlFwSTNtcjFMSG1DSEFMMgpaeTRoMCt
PZC93ZFk4NFpJS0ovR09yNmR3cktVSGVkMjI3VVhTNVM2eldlTGdJcjc3VUpvSGNsOFRmR0pwdEowCjFtMDhrOFRGeENZT2MwcVd2dURDdloxaHM3S0ZhcVdxWEhTZjNhVkI4OVZuVmpkalk2UnZUcytwWE0vYVo1UFQKakxiU0JHeEIrSk1nV3g
0ZDZ4cGNXMTdPTHhENnc1Y0RIS3hhM2x1SGE0WXNnQUpmZlE9PQotLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0tCg==
    client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFcFFJQkFBS0NBUUVBNkdnZjA0OGtZUGl1NElabzJzQ1JCcDRQUWhRTkpwNzF5ZEZ4amhJcVFZdEt3N2xaCnJmUTFoMG0zRytHU200Zi9CRFoyUU4raVF
2RjZ4OWFwMkpMK2Q1U1Jka2cvMzQvd0Jxem1wWlZ2Wk1IUk4xQlYKMTlBV2p3UkhmYnI2NWN2MWVuUUZCUHl6YnJCN3lNdjhFdnJlMGJqYXJaVC9DOEljcno4SGR2RVBIOVMxdWMzUQpURUlpTGVZV3Yxdjd2b3hOd1l5K2FYZFprVjNodUROZC9
JL3JBU05ES0dqb3NOK1ZjcnppS2ppK0E0cHdueEMvClhTODFuS2RGTlFRMW1vY0FjcHFydVZ1dXVKdm81UWFQUG1kVmthTDJLdVNtMllIbjZ0TmFTb2JEOEpSaUVoL0YKS1BxbWRQcVZVTm42czJqR0Iyb3l0TzBwb3htY3ZBVlBJSW9ZZVFJREF
RQUJBb0lCQVFDU3hmZWJsMklGTCtqWgplV2xOQmFKUmlJN2dUMk9DeVk4MmR1UjlNZkduMVQxUllJUkhmTncxWlpPVHN5RHd3ckdHU3BMaUk1b0pKeEpBCmxNeE5MNEF2cXc4Mm9TaGRtenllS3FHN1h5V3JmMUFPMjlocTVFcnE1V3YwOXBvdmV
HQytoZmtKamtJUG5MdUEKTTNzdHpXdmREUjVsMDV4REk4TVQxZjF6MjRlT3RJWTh1ZUVLbjNSWm5uak8rTXZVdWpBcE5LY3dvUnhJNW9Fbwp0MGJwQ3E0WVZic2dqNm5lbE8vNlh5cXB5c1paVGtEZDBNMHpTM2VaZEthc2ErQ1JZZDZLb2ZCN01
GL1dmRm5oCjNlKzJURWN3RXhsR1hPd1Zrd3E3MmpwRHgvZUl3M0hSWUhWTUh6NHM1UkliYWtiWVBUUGtJcjM5V2tUY3piRFEKNmVYczE5dVZBb0dCQVBRMHJyN21mR3BrenV0a1RlSlB1Z3dXcnNGM2hrWlo3MEw5MXV1aWxKZXlYNTM0VG5hawp
qRHZNVU1RQVdXbmY2YVI2OXdiMTArK1NibjExWUl3MW9rTU03MVQ4S3JEbTJLRFpPWTFqMWcySURSUGJhZXZsCm9leDIyTGdQc0I0eFVtZlF4SnAzNCtsN2NhTGxMUlZBbnBSTnZNblNyRmNNNU1tRkJiNjgwTTdyQW9HQkFQT2gKajI0Wk52NkJ
```

copy it and save it in documents or another folder save it as secret-file.txt

Note: create a secret-file.txt in your file explorer save the config in it and use this at the kubernetes credential section.

Install Kubernetes Plugin to give the saved file



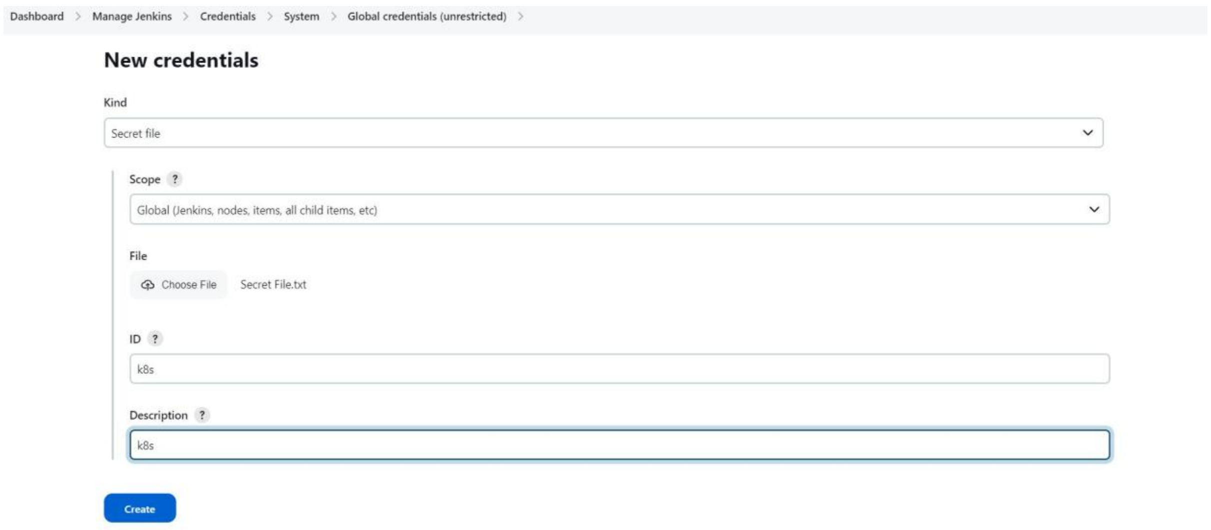
Dashboard > Manage Jenkins > Plugins

Plugins

Search: Kuber

Install	Name ↓	Released
✓	Kubernetes Credentials 0.11 kubernetes · credentials Common classes for Kubernetes credentials	9 days 16 hr ago
✓	Kubernetes Client API 6.8.1-224.vd388fca_4db_3b_ kubernetes · Library plugins (for use by other plugins) Kubernetes Client API plugin for use by other Jenkins plugins.	9 days 17 hr ago
✓	Kubernetes 4029.v5712230ccb_f8 Cloud Providers · Cluster Management · kubernetes · Agent Management This plugin integrates Jenkins with Kubernetes	9 days 15 hr ago
✓	Kubernetes CLI 1.12.1 kubernetes Configure kubectl for Kubernetes	8 days 22 hr ago

goto manage Jenkins –> manage credentials –> Click on Jenkins global –> add credentials


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

New credentials

Kind: Secret file

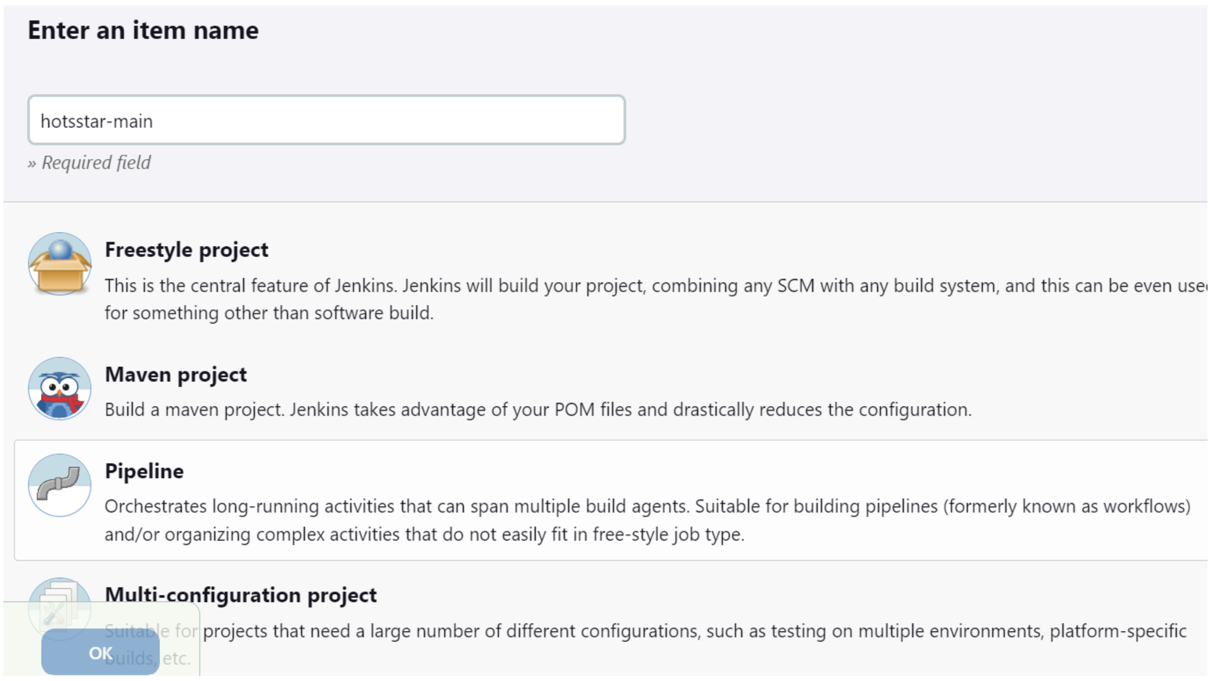
Scope: Global (Jenkins, nodes, items, all child items, etc)

File: Choose File Secret File.txt

ID: k8s

Description: k8s

Now let's create a new job for our pipeline


Enter an item name

hotsstar-main

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even use
for something other than software build.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Past below script

```
pipeline{

    agent any
```

```
   tools{

      jdk 'jdk17'

      nodejs 'node16'

   }

   environment {

      SCANNER_HOME=tool 'sonar-scanner'

   }

   stages {

      stage('clean workspace'){

         steps{

            cleanWs()

         }

      }

      stage('Checkout from Git'){

         steps{

            git branch: 'main', url: 'https://github.com/Aj7Ay/Hotstar-
Clone.git'

         }

      }

      stage("Sonarqube Analysis "){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -
Dsonar.projectName=Hotstar \
                    -Dsonar.projectKey=Hotstar'''
                }
            }
        }
        stage("quality gate"){
           steps {
                script {
                    waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-
token'
                }
            }
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
```

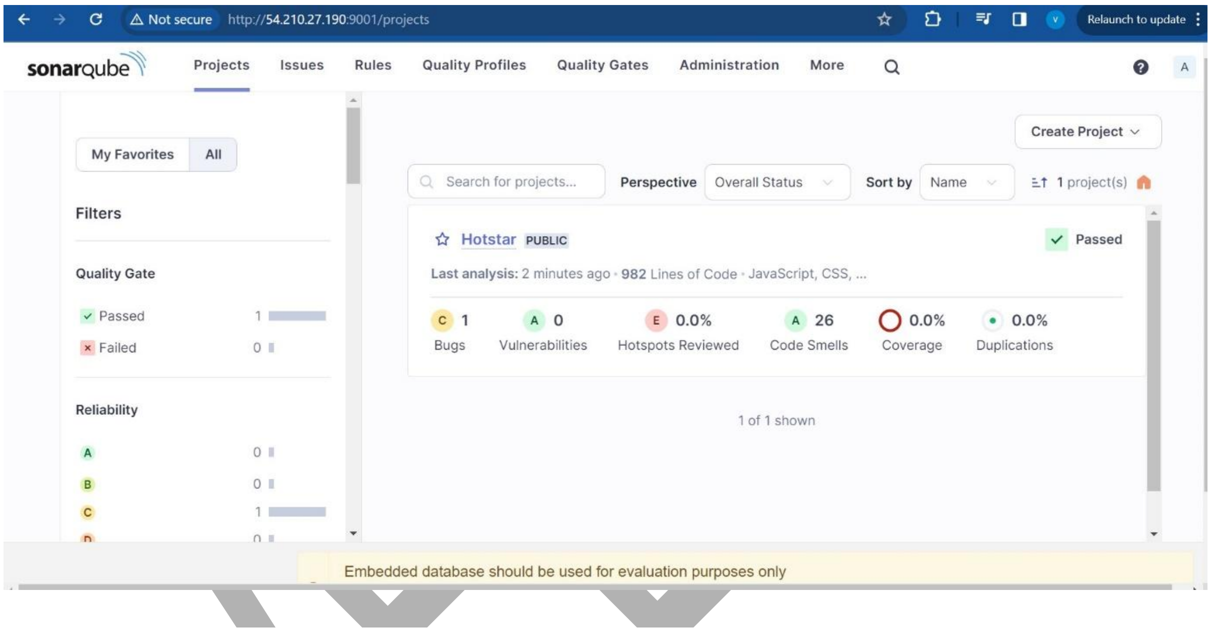
```
stage("Docker Build & Push"){

    steps{

        script{

           withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){

               sh "docker build -t hotstar ."

               sh "docker tag hotstar veeranarni/hotstar:latest "

               sh "docker push veeranarni/hotstar:latest"

            }

        }

    }

}


stage('Image scanner') {

    steps {

        sh "trivy image hoststar"

    }

}

stage("deploy_docker"){
```

```
            steps{
                sh "docker run -d --name hotstar -p 3000:3000
veeranarni/hotstar:latest"
            }
        }
    }
}


        stage('Deploy to kubernets'){
            steps{
                script{
                    dir('K8S') {
                        withKubeConfig(caCertificate: '', clusterName: '',
contextName: '', credentialsId: 'k8s', namespace: '',
restrictKubeConfigAccess: false, serverUrl: '') {
                                sh 'aws eks update-kubeconfig --name EKS_CLOUD --
region ap-south-1'
                                sh 'kubectl apply -f deployment.yml'
                                sh 'kubectl apply -f service.yml'
                        }
                    }
                }
        }
    }
```

After Run

We can observe

Sonar project will be created for reference below



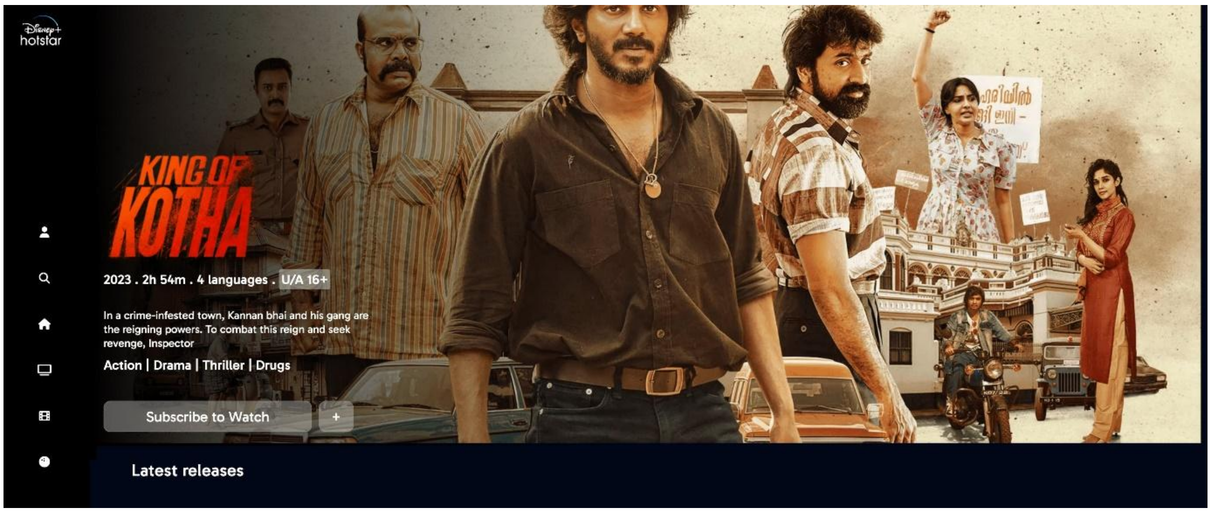
When you log in to Dockerhub, you will see a new image is created

Recommendations

Deploy to Container check manually with docker container also by using below command

```
<ec2-ip:3000>  Note :“NodeJs runs port 3000”
```

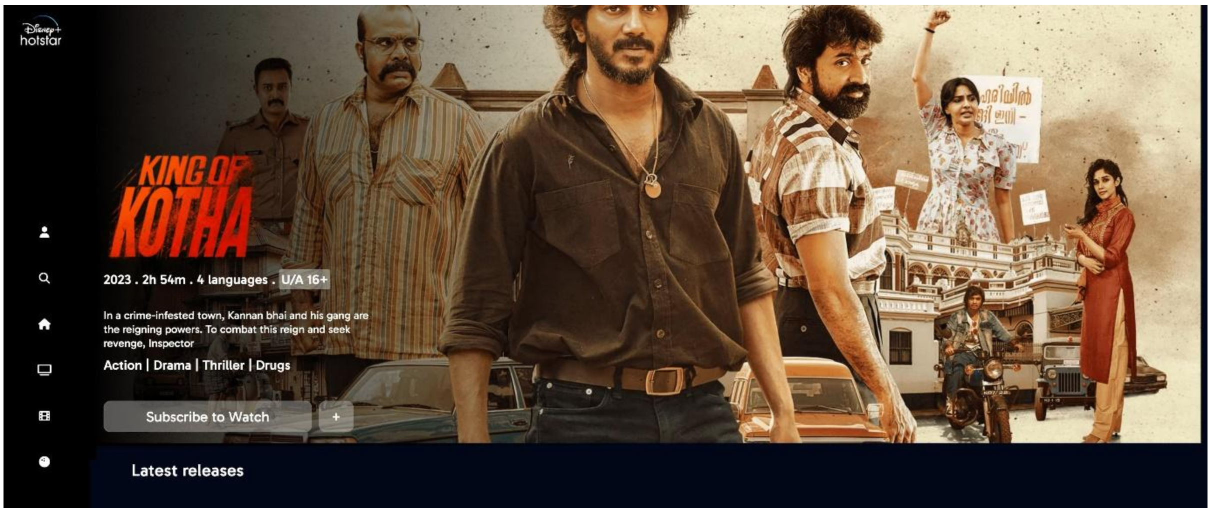
Output


Disney+ hotstar
KING OF KOTHA
2023 . 2h 54m . 4 languages . U/A 16+
In a crime-infested town, Kannan bhai and his gang are the reigning powers. To combat this reign and seek revenge, Inspector
Action | Drama | Thriller | Drugs
Subscribe to Watch
+
Latest releases

```
kubectl get all
```

Add Load balancer IP address to cluster ec2 instance security group and copy load balancer Link and open in a browser

You will see output like this.

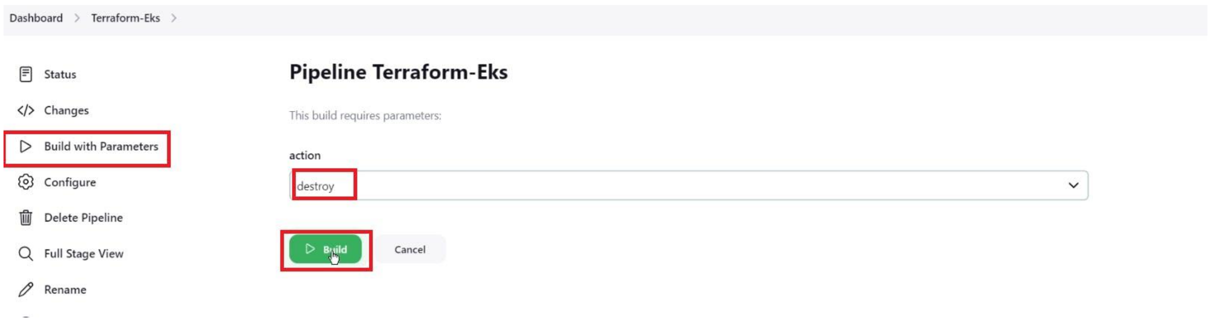

Disney+ hotstar
KING OF KOTHA
2023 . 2h 54m . 4 languages . U/A 16+
In a crime-infested town, Kannan bhai and his gang are the reigning powers. To combat this reign and seek revenge, Inspector
Action | Drama | Thriller | Drugs
Subscribe to Watch
+
Latest releases

Step 4: Destruction

Now Go to Jenkins Dashboard and click on Terraform-Eks job

And build with parameters and destroy action

It will delete the EKS cluster that provisioned

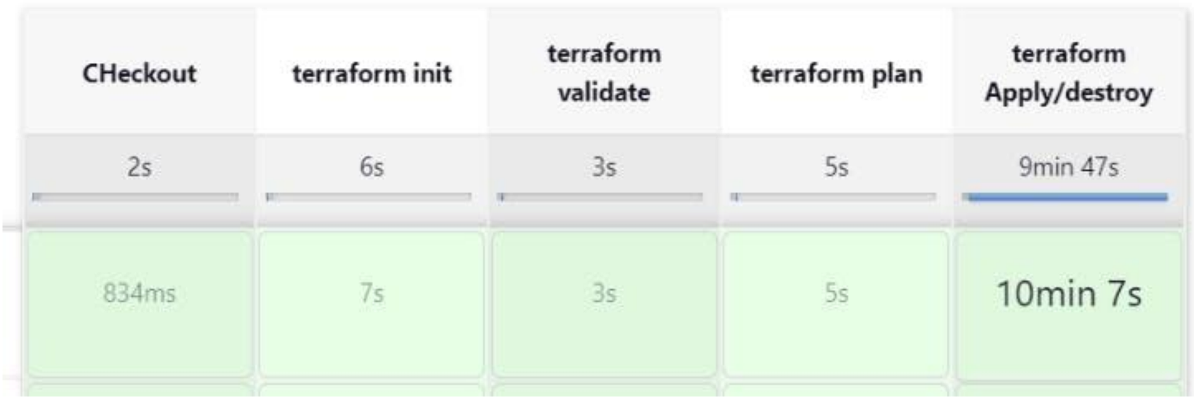

Dashboard > Terraform-Eks >

Pipeline Terraform-Eks

This build requires parameters:

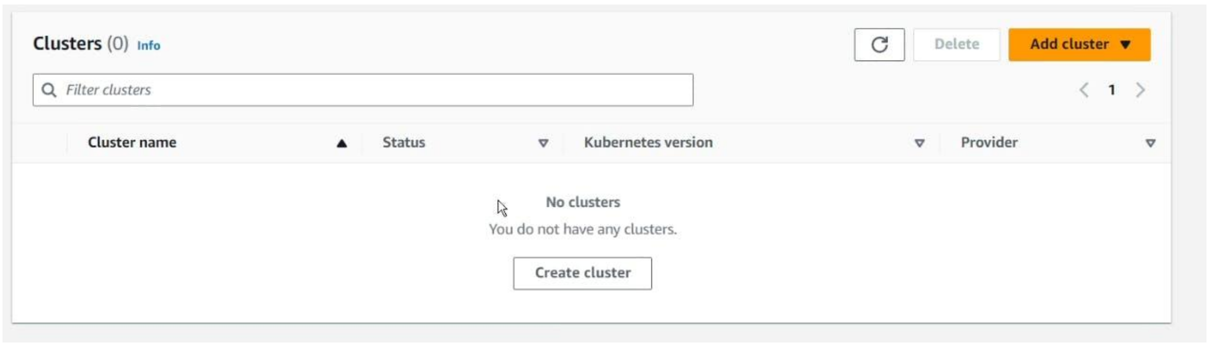
action: destroy

After 10 minutes cluster will delete and wait for it. Don't remove ec2 instance till that time.



CHeckout	terraform init	terraform validate	terraform plan	terraform Apply/destroy
2s	6s	3s	5s	9min 47s
834ms	7s	3s	5s	10min 7s

Cluster deleted


Clusters (0) Info

Cluster name	Status	Kubernetes version	Provider

No clusters
You do not have any clusters.

Delete the Ec2 instance & IAM role.

-------------------------------- Thanks --