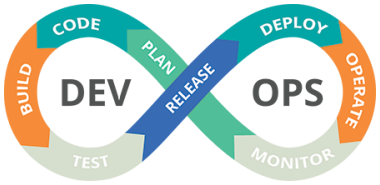


Name : S.Guhan Balaji

Complete CICD implementation using Jenkins

Introduction:

In this project we are going to build an application using Jenkins and deploy to the Kubernetes cluster. In this process allows for automated and streamlined software development and deployment, ensuring that your applications are built, tested, and deployed consistently and efficiently. So we are going to use AWS, Jenkins, Maven, SonarQube, Trivy, Docker and Kubernetes in this project.

Steps Involved:

Step 1: AWS

- Login to your AWS account.
- Launch an ec2 instance with instance type as Large. So it can handle many resources easily.
- Here Iam using Mobaxterm, to connect to my instance. Ensure that port is 22.
- Copy the public ip of your aws instance and launch it in your terminal using private key you have. Finally login as ec2-user.

Step 2: INSTALLATION

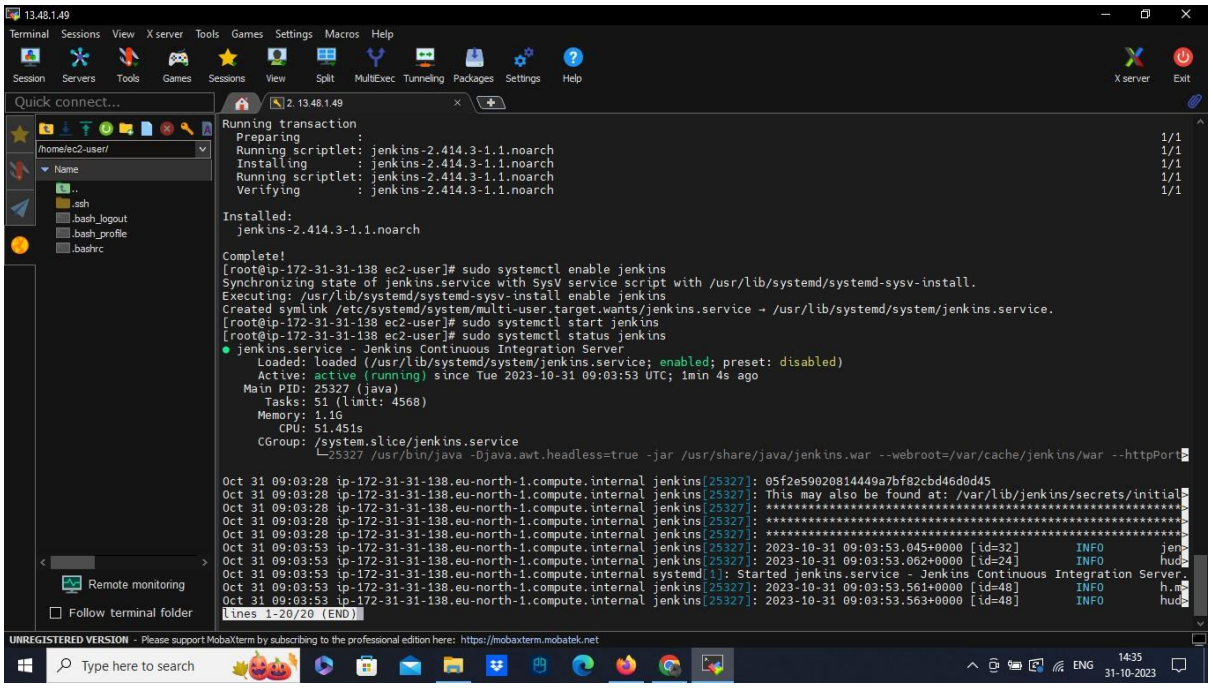
Here we have to install necessary tools and resources to our prod server to work properly.

- Jenkins
    1. Ensure that your software packages are up to date on your instance by using the following command to perform a quick software update:
       [ec2-user ~]$ sudo yum update –y
    2. Add the Jenkins repo using the following command:
       [ec2-user ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
       https://pkg.jenkins.io/redhat-stable/jenkins.repo
    3. Import a key file from Jenkins-CI to enable installation from the package:

[ec2-user ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

[ec2-user ~]$ sudo yum upgrade

4. Install Java (Amazon Linux 2023):
   [ec2-user ~]$ sudo dnf install java-17-amazon-corretto -y
5. Install Jenkins:
   [ec2-user ~]$ sudo yum install jenkins -y
6. Enable the Jenkins service to start at boot:
   [ec2-user ~]$ sudo systemctl enable jenkins
7. Start Jenkins as a service:
   [ec2-user ~]$ sudo systemctl start jenkins
8. You can check the status of the Jenkins service using the command:
   [ec2-user ~]$ sudo systemctl status Jenkins



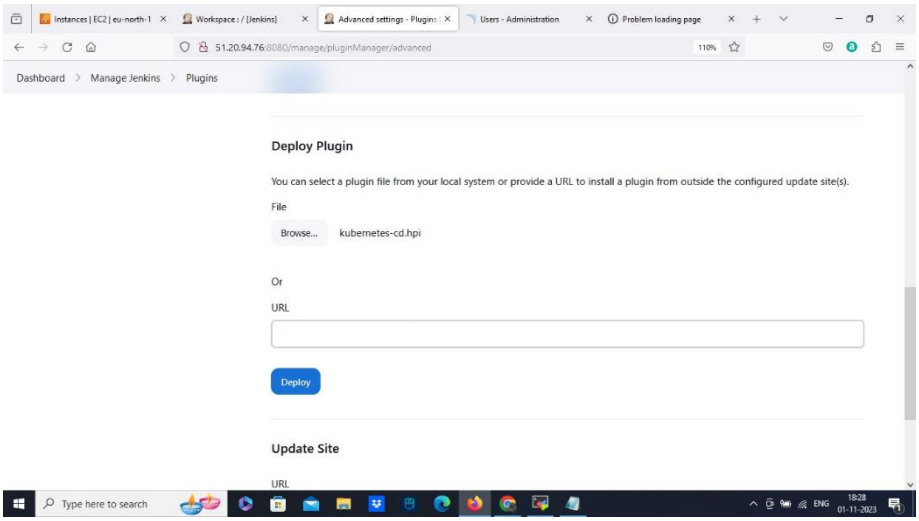
- Jenkins is now installed and running on your EC2 instance. Now to open Jenkins.
- Copy the public ip and paste it in the browser as http:// :8080.
- [ec2-user ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword – use this command to get the password
- Login in with the initial password. Create user and new password.
- And you will get login. Now back to the server
- Install Git:
    1. sudo yum install git -y

- Install Trivy (to secure a container) :
    1. Download the Trivy binary package for your architecture (either trivy_0.12.0_Linux-64bit.tar.gz or trivy_0.12.0_Linux-32bit.tar.gz) from the GitHub releases page: https://github.com/aquasecurity/trivy/releases
    2. Unpack the archive by running tar xvf trivy_0.12.0_Linux-64bit.tar.gz
    3. Move the binary to a directory in your PATH, for example /usr/local/bin/ by running mv trivy /usr/local/bin/
    4. Verify that trivy is installed by running trivy –version

- Install Docker:
    1. yum install docker -y
    2. systemctl start docker
    3. systemctl enable docker
- Pull Docker image from sonarqube and run the container:
    1. docker pull sonarqube:lts-community
    2. docker run -d --name sonarqube -p 9000:9000 sonarqube:lts-community
    3. docker ps to check the container status.
    4. To grant permission between Jenkins and docker use: sudo usermod -aG docker Jenkins
- now restart the Jenkins and login in to Jenkins in browser.

Step 3: Installing plugins in Jenkins:

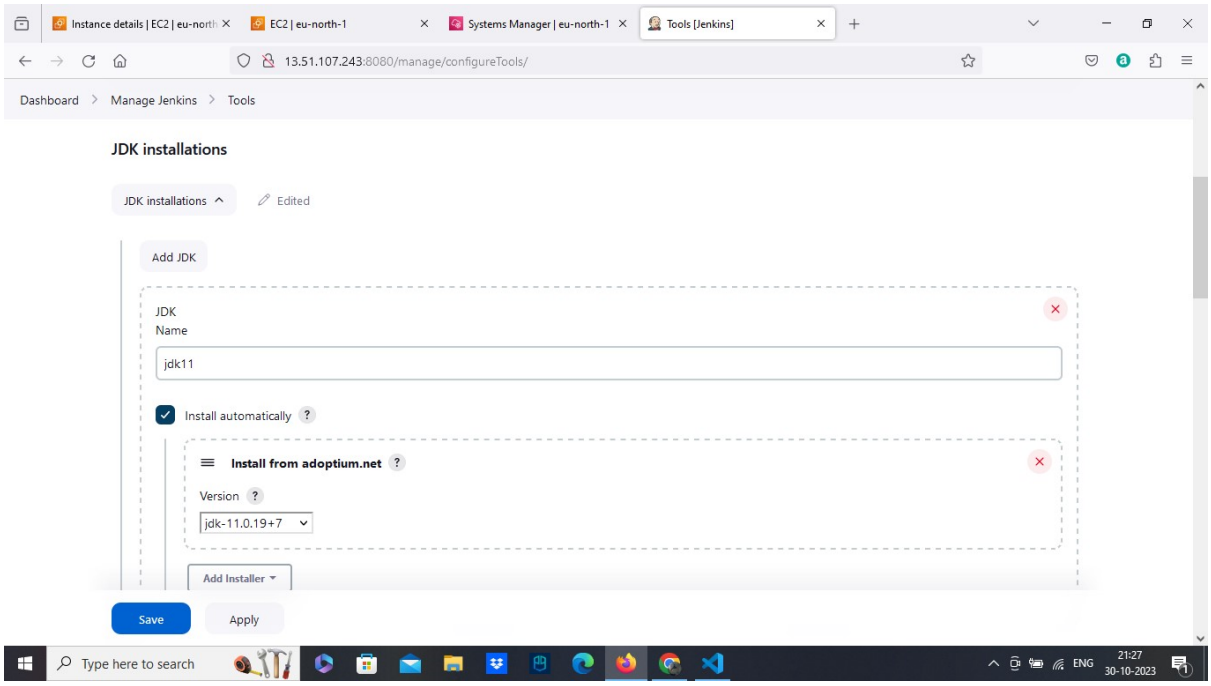
- Here we need to install additional plugin in our Jenkins
- Go to manage Jenkins and go to available plugins. The plugins are:
    1. Jdk from adoptium.net
    2. Sonarqube Scanner
    3. Docker
    4. Kubernetes
- Sometimes an important Kubernetes CD plugin wont be available. In that case we need to install it manually to our Jenkins:
        a. Download Kubernetes cd.hpi (plugin) from browser

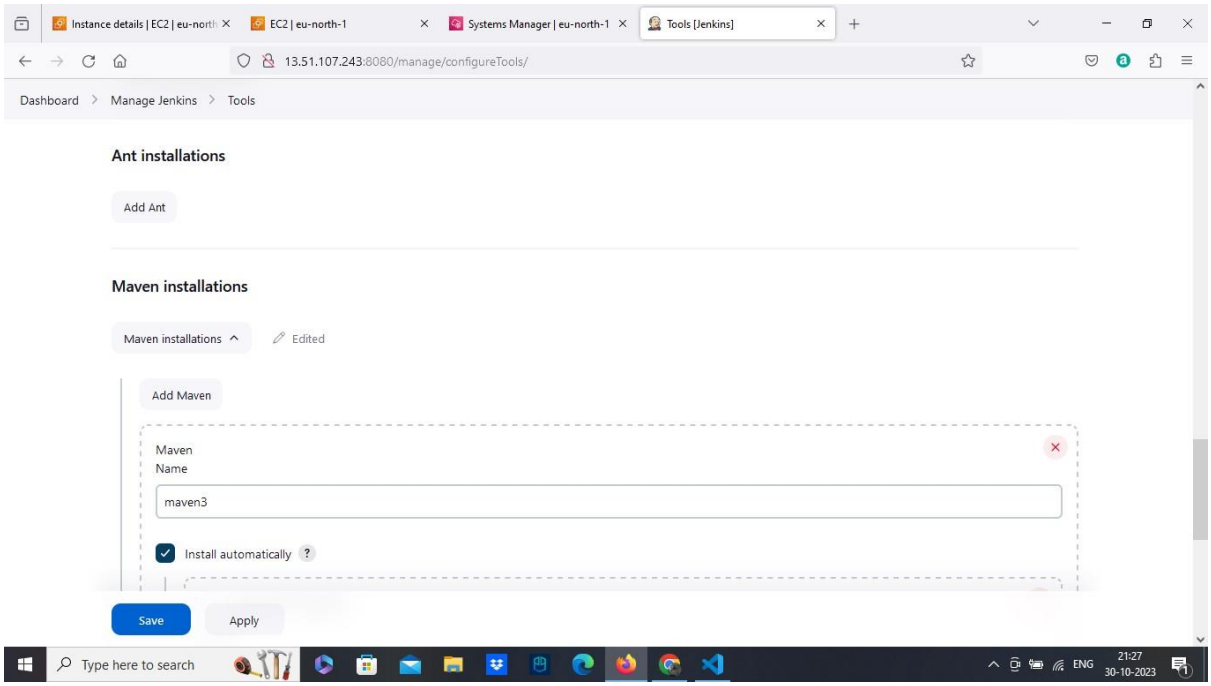


b. And install it manually in manage plugins

Step 4: Configure tools in Jenkins:

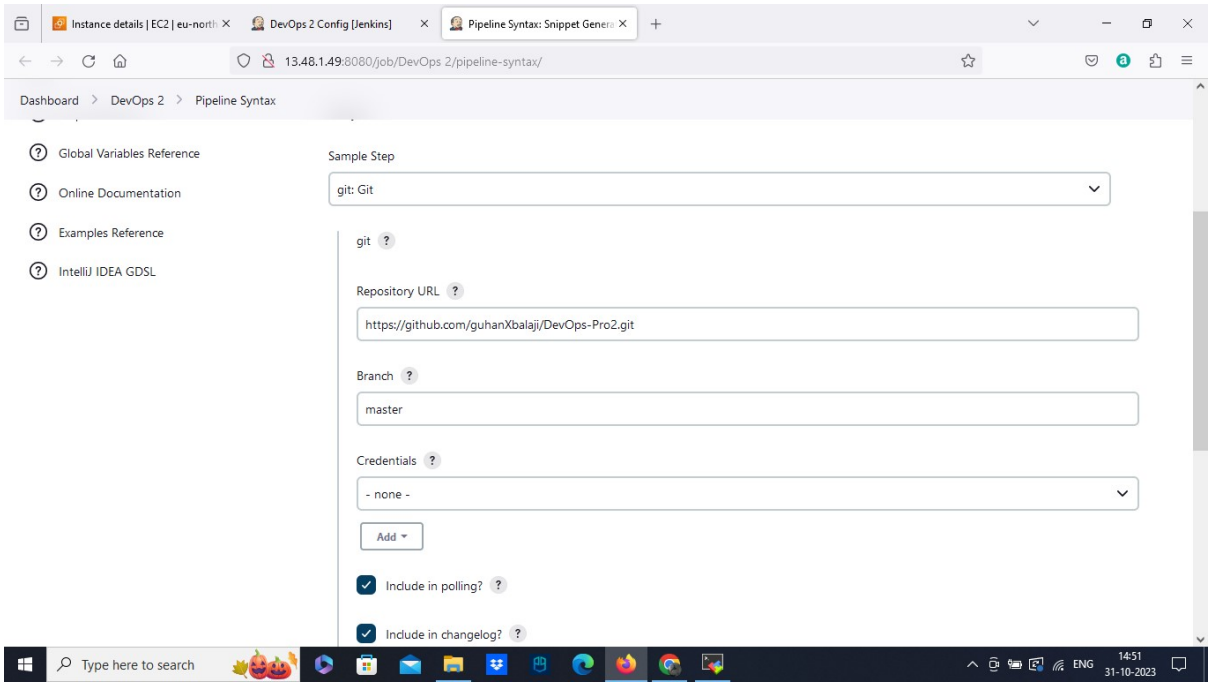
- Open configure tools in Jenkins
- Add jdk and name it jdk 11, with the installer adoptium.net and select the version
- Add maven and name it maven3, and select the version.
- Add docker with the latest version and add add sonarqube as well.

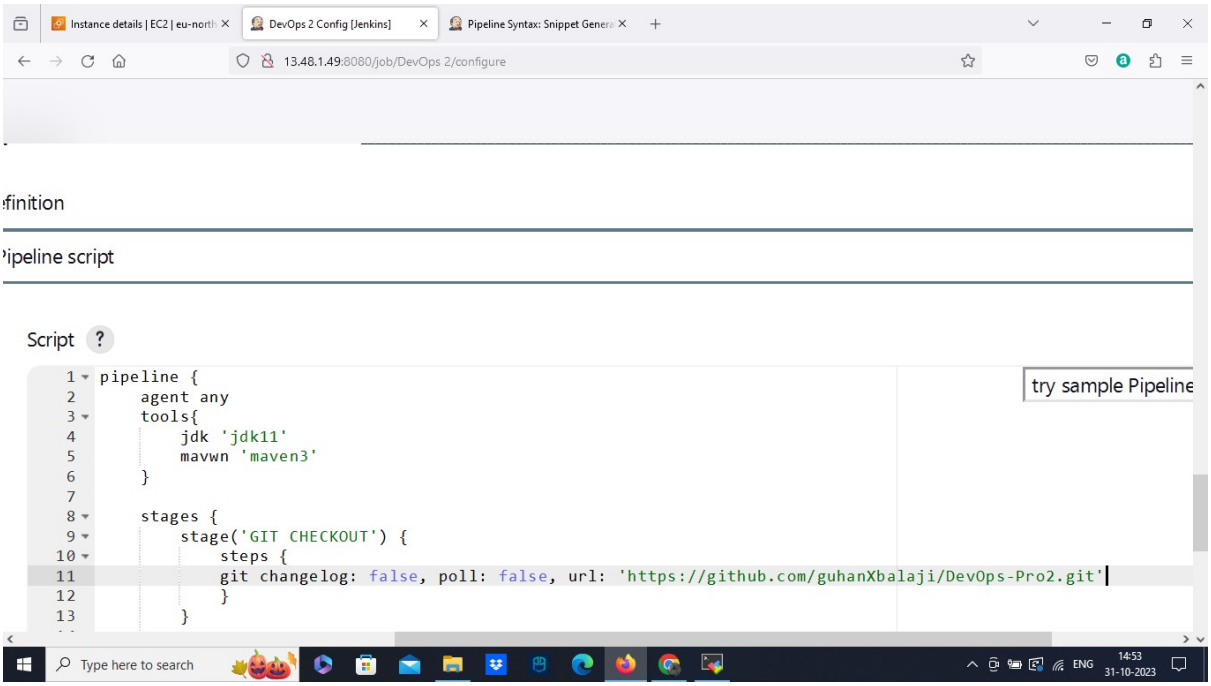




Step 5: Create a Pipeline:

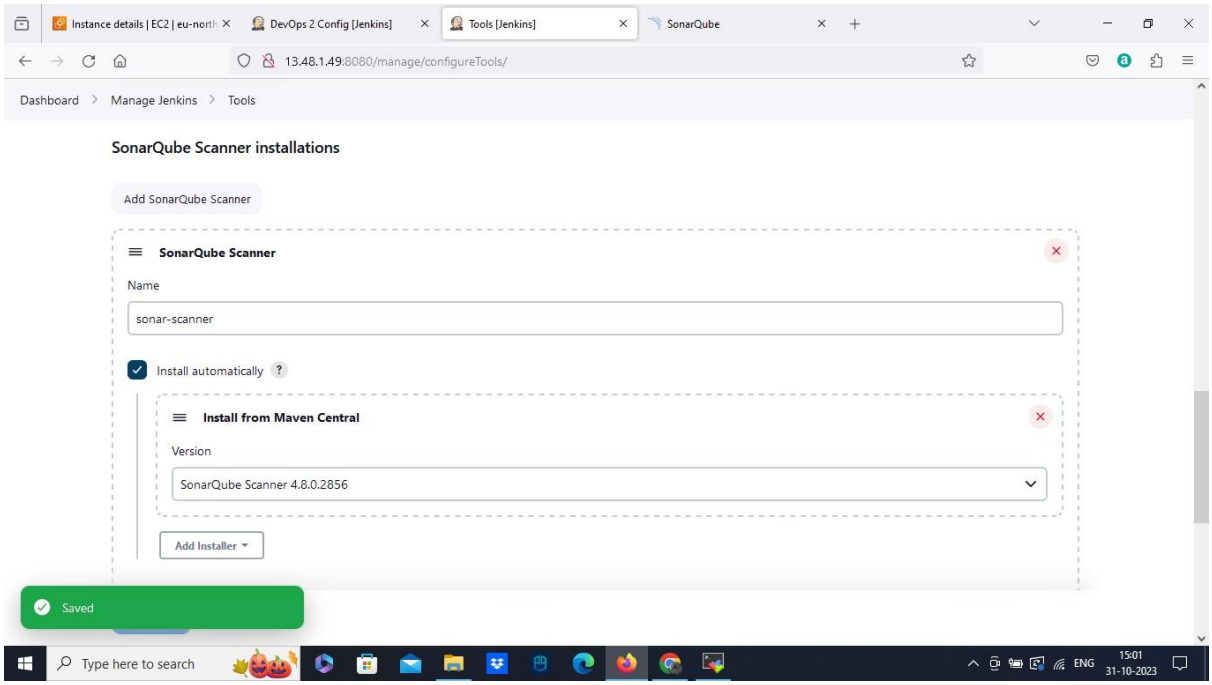
- Create a pipe line. Add a name and description.
- Write the script with any agent and use tools jdk11 and maven3.
- In the next stage of the script add the stage GIT CHECKOUT
- For this script use pipeline syntax for generate script
- Select git:Git as sample step and paste the repository URL. Then generate the script, copy it and paste it in the main script.
- In the next stage of the script will be code compile, and add the command sh “mvn clean compile”.

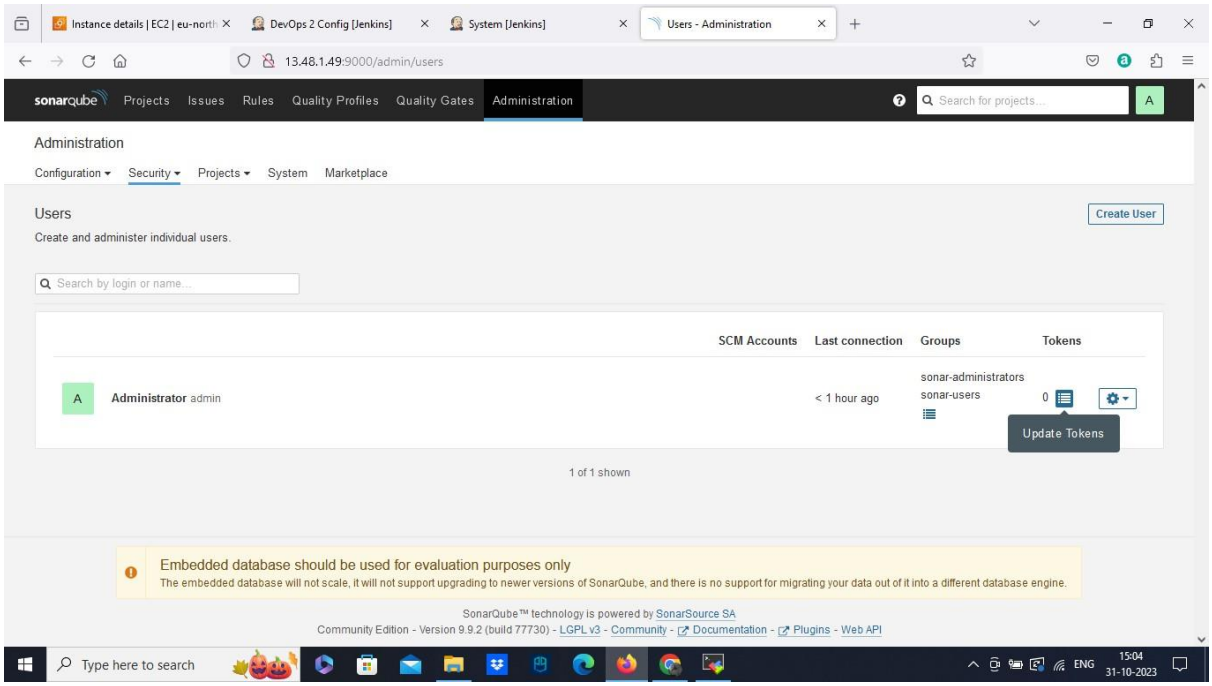


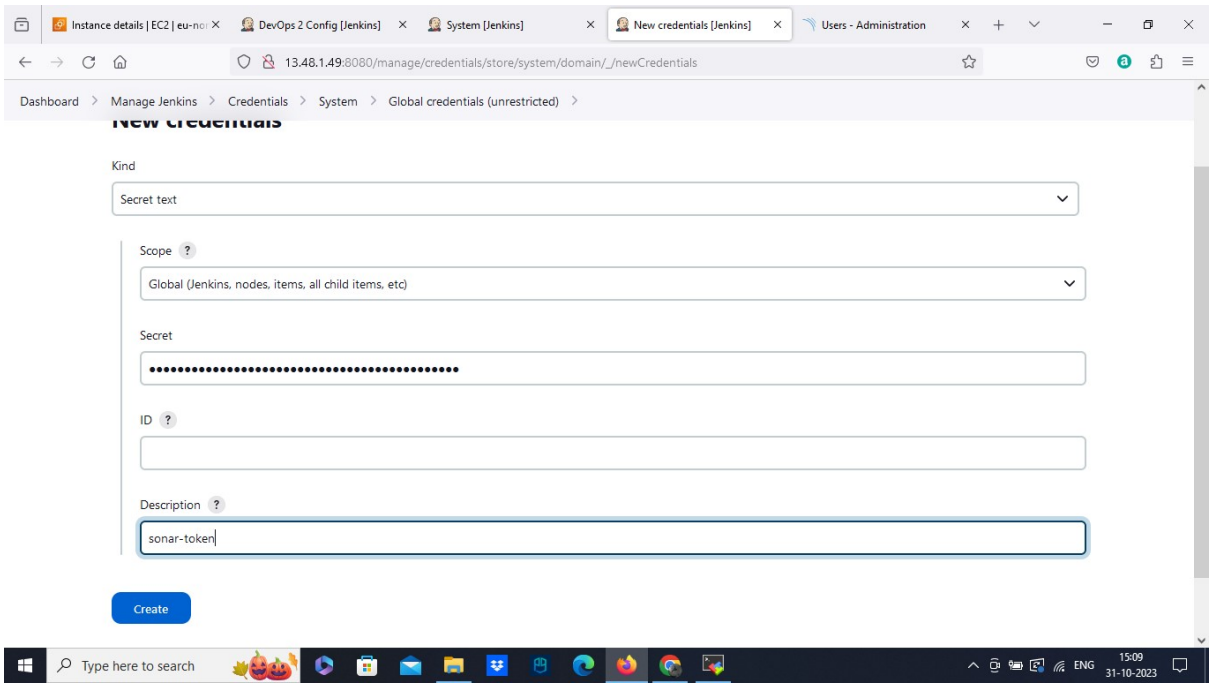


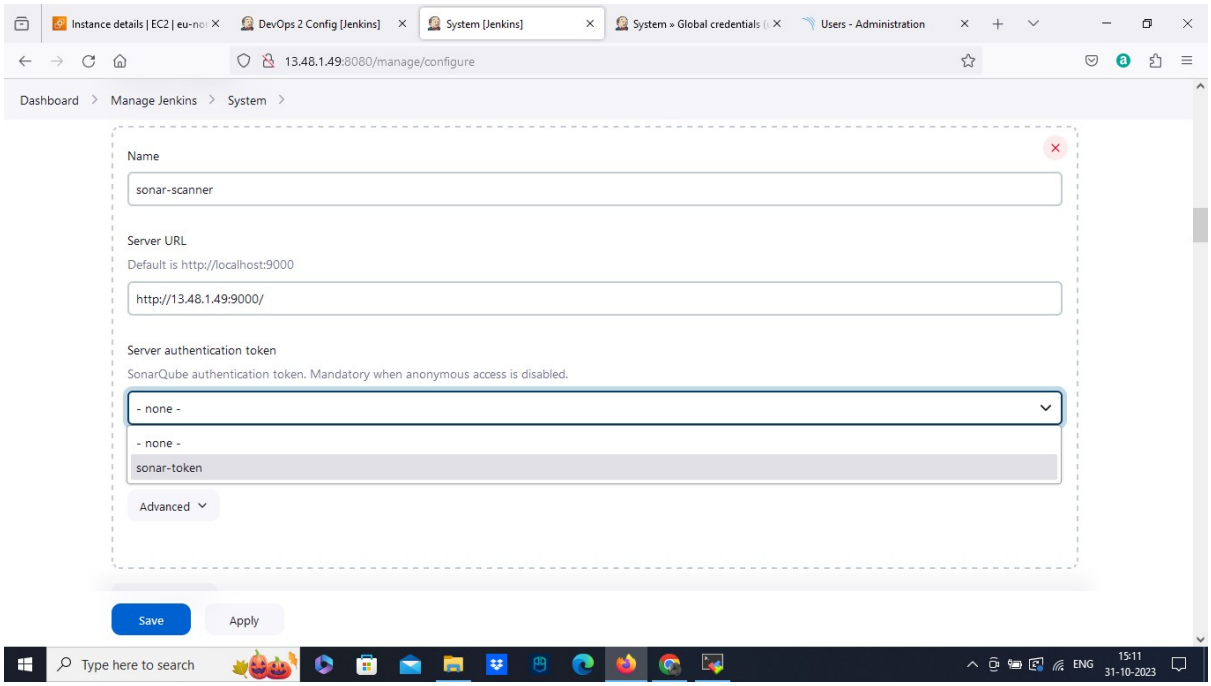
Step 6: Sonarqube analysis:

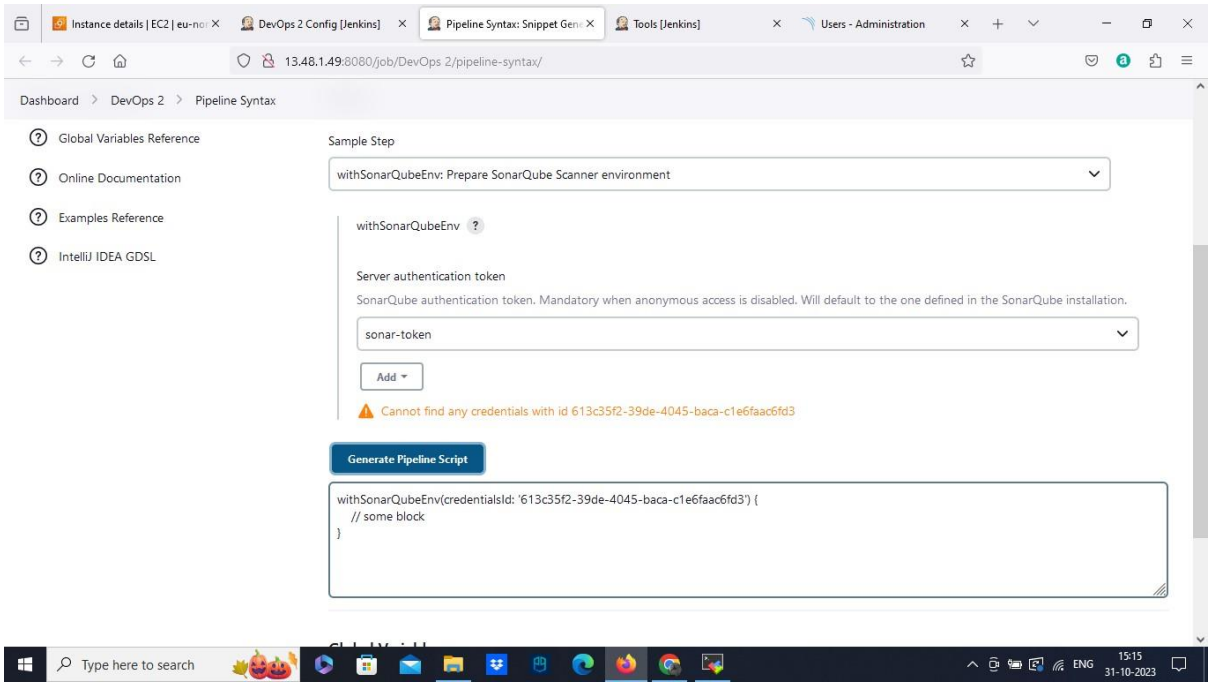
- The next stage of the script will be sonar qube analysis
- We already started the sonarqube in our server using docker container. Copy the public ip and paste it in browser http:// :9000.
- Login the sonarqube and go to administration/security.
- Generate token.
- Copy the token. Create a credentials which is secretext. Add the token in secret field and save the credentials with id in your Jenkins.
- Go to configure system in Jenkins. Select sonarqube installations, add name, server URL and authentication token we created as a secret text.
- And now go back to the script we created. Using pipeline syntax again, generate a script .
- Add sonarqube env as sample step, and also add sonar token.
- Generate the script,copy it and paste it to our main script.



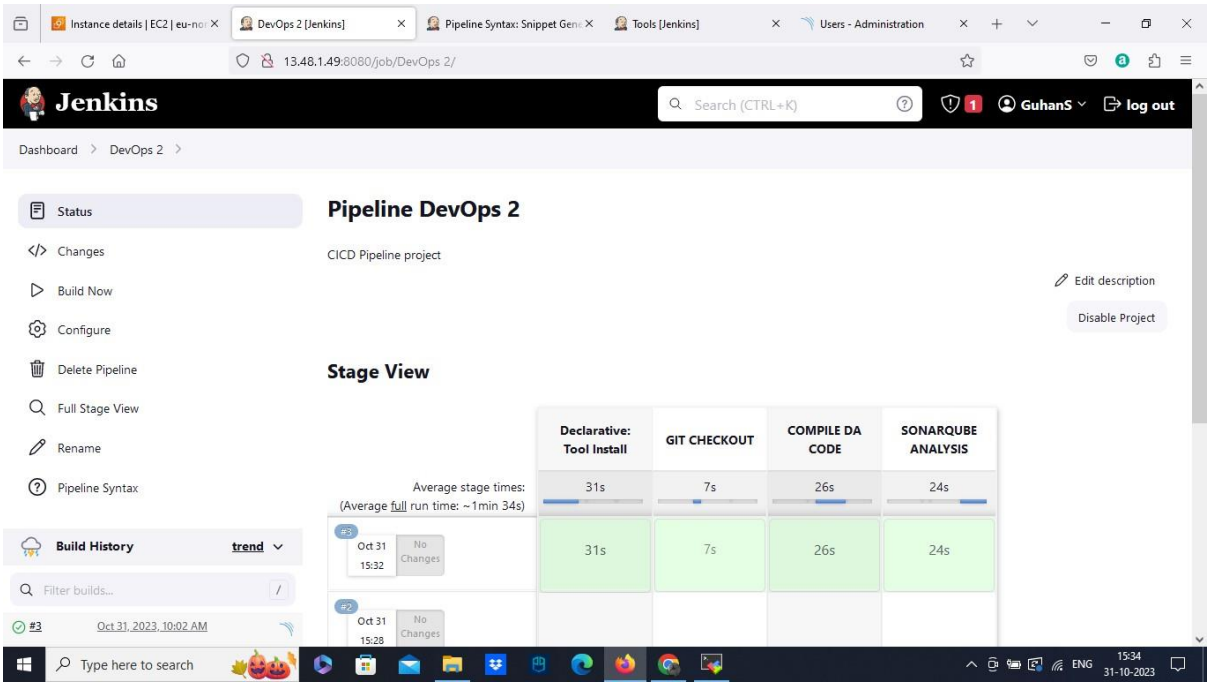

Instance details | EC2 | eu-north
DevOps 2 Config [Jenkins]
System [Jenkins]
Users - Administration
13.48.1.49:9000/admin/users
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
A
Administration
Configuration
Security
Projects
System
Marketplace
Users
Create and administer individual users.
Create User
Search by login or name...
SCM Accounts
Last connection
Groups
Tokens
A
Administrator admin
< 1 hour ago
sonar-administrators
sonar-users
0
Update Tokens
1 of 1 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.9.2 (build 77730) - LGPL v3 - Community - Documentation - Plugins - Web API
Type here to search
ENG
15:04
31-10-2023

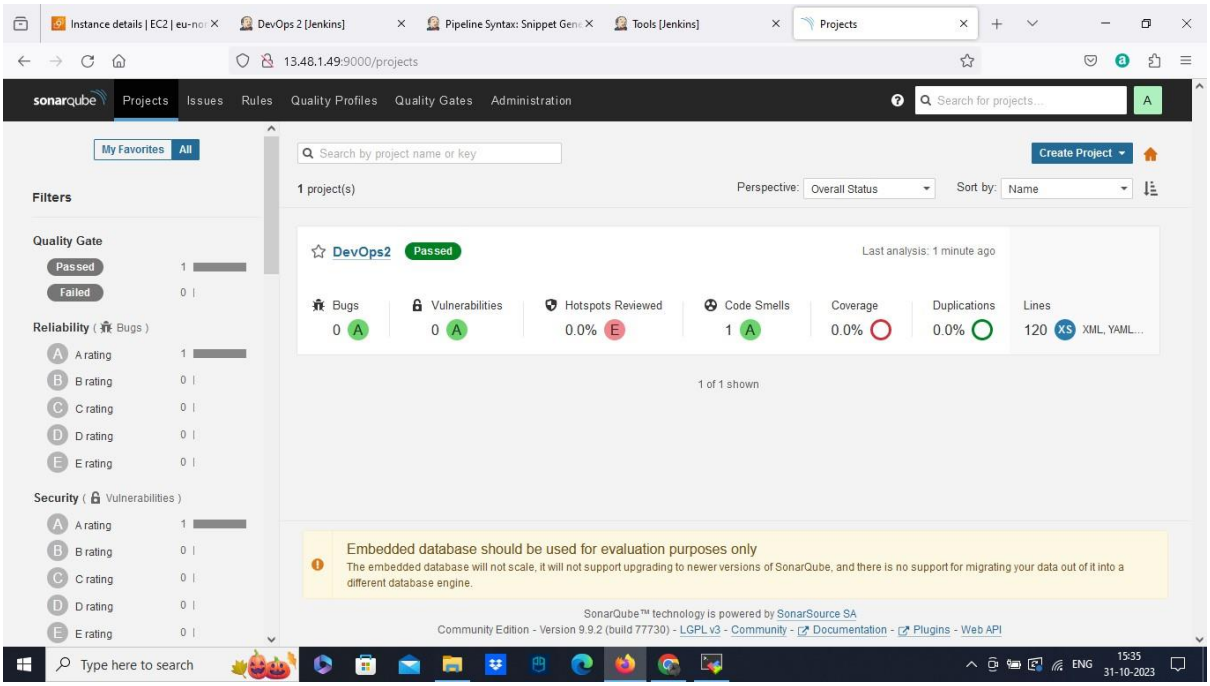

Instance details | EC2 | eu-no
DevOps 2 Config [Jenkins]
System [Jenkins]
New credentials [Jenkins]
Users - Administration
13.48.1.49:8080/manage/credentials/store/system/domain/_/newCredentials
Dashboard
Manage Jenkins
Credentials
System
Global credentials (unrestricted)
New credentials
Kind
Secret text
Scope
Global (Jenkins, nodes, items, all child items, etc)
Secret
ID
Description
sonar-token
Create
Type here to search
ENG
15:09
31-10-2023





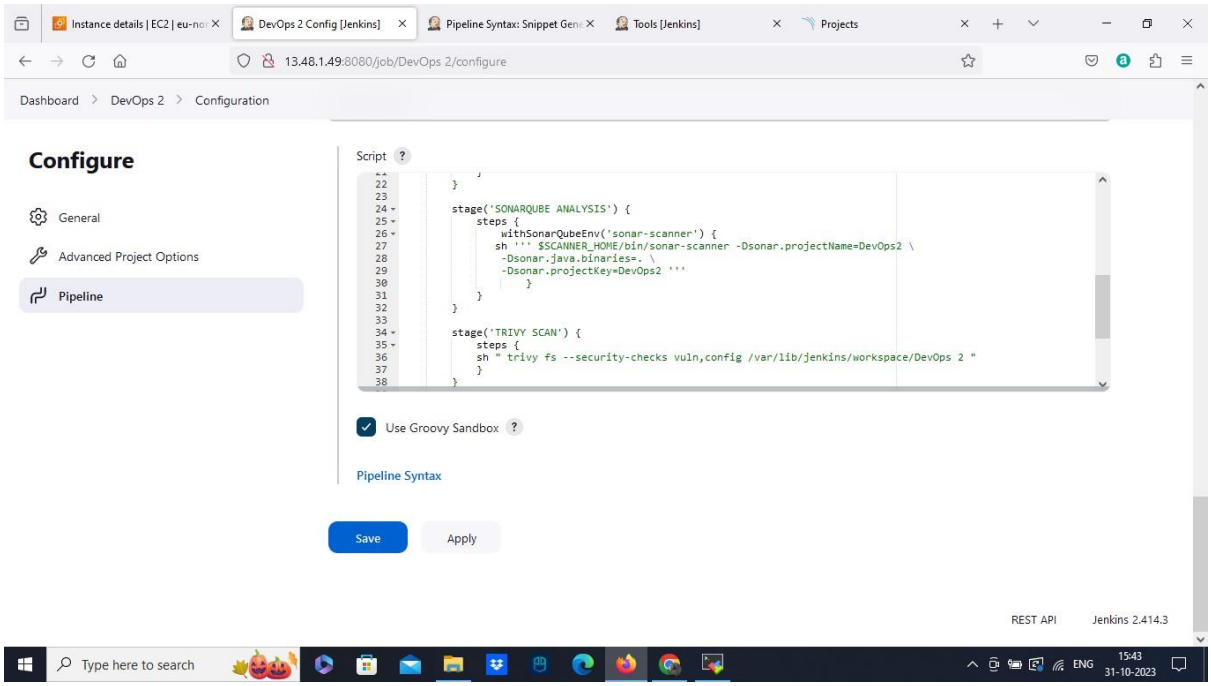
- Save the script and select build now to build the script we done so far.
- After the built we can see that it got passed in sonarqube as you can see

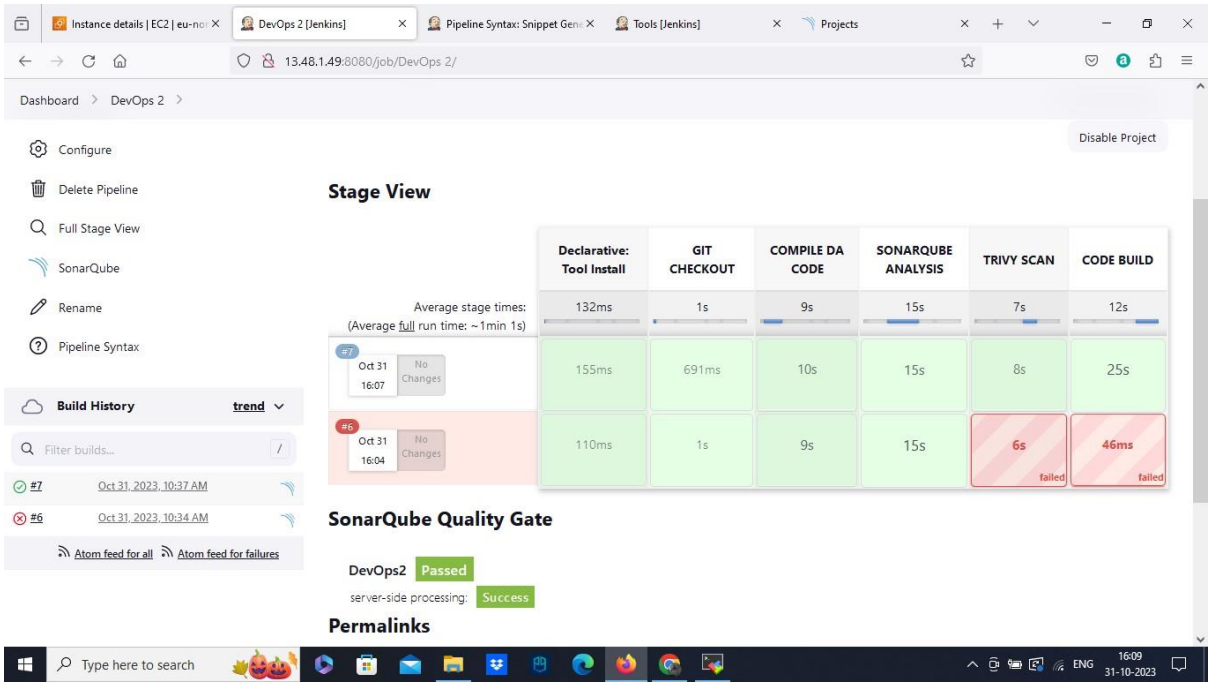

Instance details | EC2 | eu-no
DevOps 2 [Jenkins]
Pipeline Syntax: Snippet Gen
Tools [Jenkins]
Users - Administration
13.48.1.49:8080/job/DevOps 2/
Jenkins
Search (CTRL+K)
GuhanS
log out
Dashboard
DevOps 2
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Pipeline DevOps 2
CICD Pipeline project
Edit description
Disable Project
Stage View
Declarative: Tool Install
GIT CHECKOUT
COMPILE DA CODE
SONARQUBE ANALYSIS
Average stage times:
(Average full run time: ~1min 34s)
31s
7s
26s
24s
Oct 31 15:32
No Changes
Oct 31 15:28
Build History
trend
Filter builds...
#3
Oct 31, 2023, 10:02 AM
Type here to search
ENG
15:34
31-10-2023


Instance details | EC2 | eu-no
DevOps 2 [Jenkins]
Pipeline Syntax: Snippet Gen
Tools [Jenkins]
Projects
13.48.1.49:9000/projects
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
My Favorites
All
Filters
Search by project name or key
Create Project
1 project(s)
Perspective:
Overall Status
Sort by:
Name
Quality Gate
Passed
Failed
Reliability (Bugs)
A rating
B rating
C rating
D rating
E rating
Security (Vulnerabilities)
DevOps2
Passed
Last analysis: 1 minute ago
Bugs
Vulnerabilities
Hotspots Reviewed
Code Smells
Coverage
Duplications
Lines
0
0
0.0%
1
0.0%
120
XML, YAML...
1 of 1 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.9.2 (build 77730) - LGPL v3 - Community - Documentation - Plugins - Web API
Type here to search
ENG
15:35
31-10-2023

Step 7: Trivy scan:

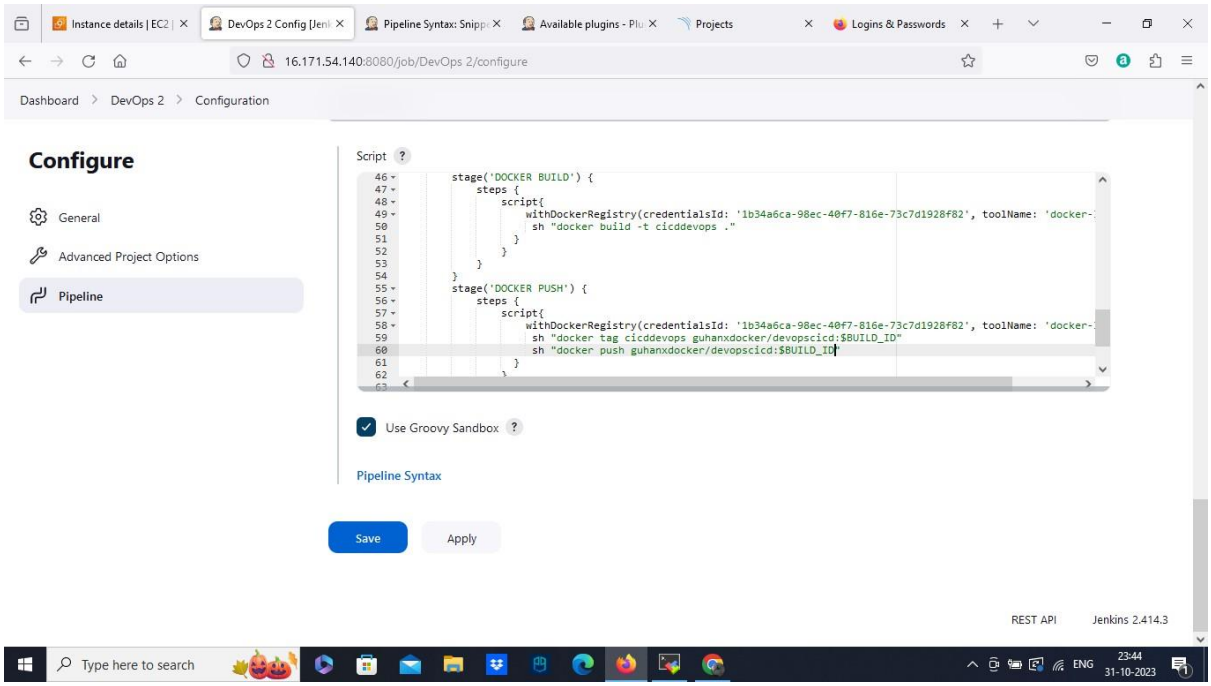
- The next stage of our script will be trivy scan.
- And add the command sh "trivy fs --security-checks vuln,config /var/lib/jenkins/workspace/DevOps\\ 2" in the script
- Then add the next stage as CODE BUILD and add the command sh "mvn clean install" in the step.
- Save it and build now.

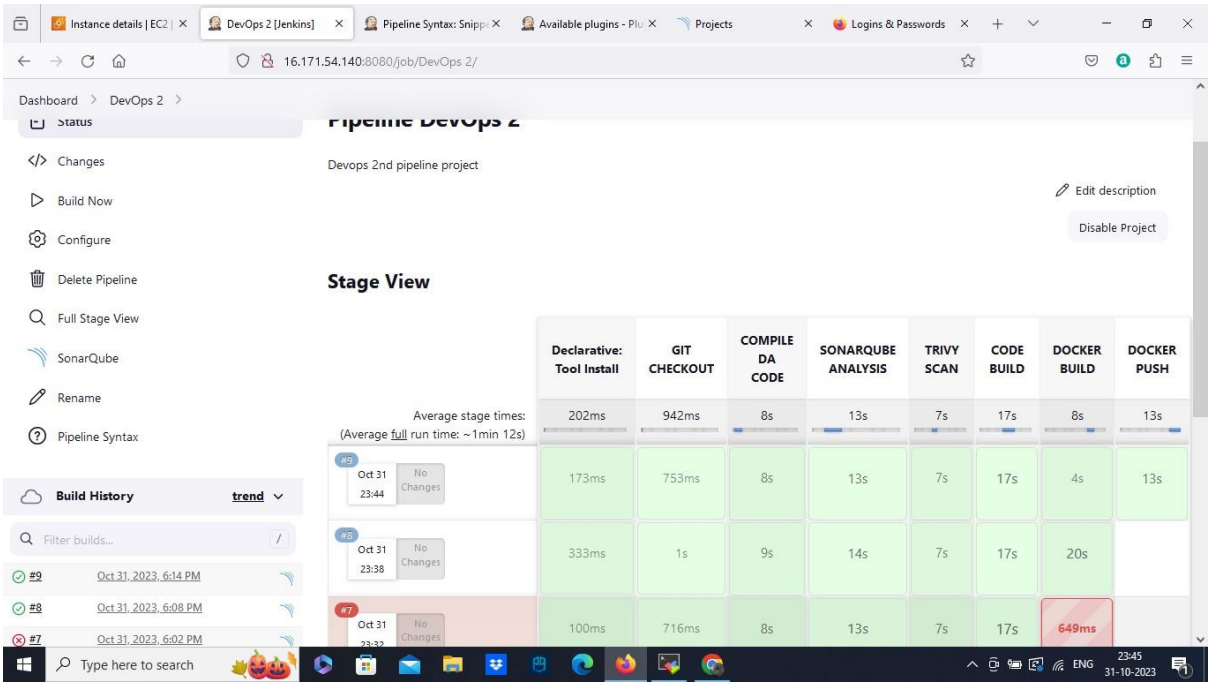




Step 8: Docker build and push:

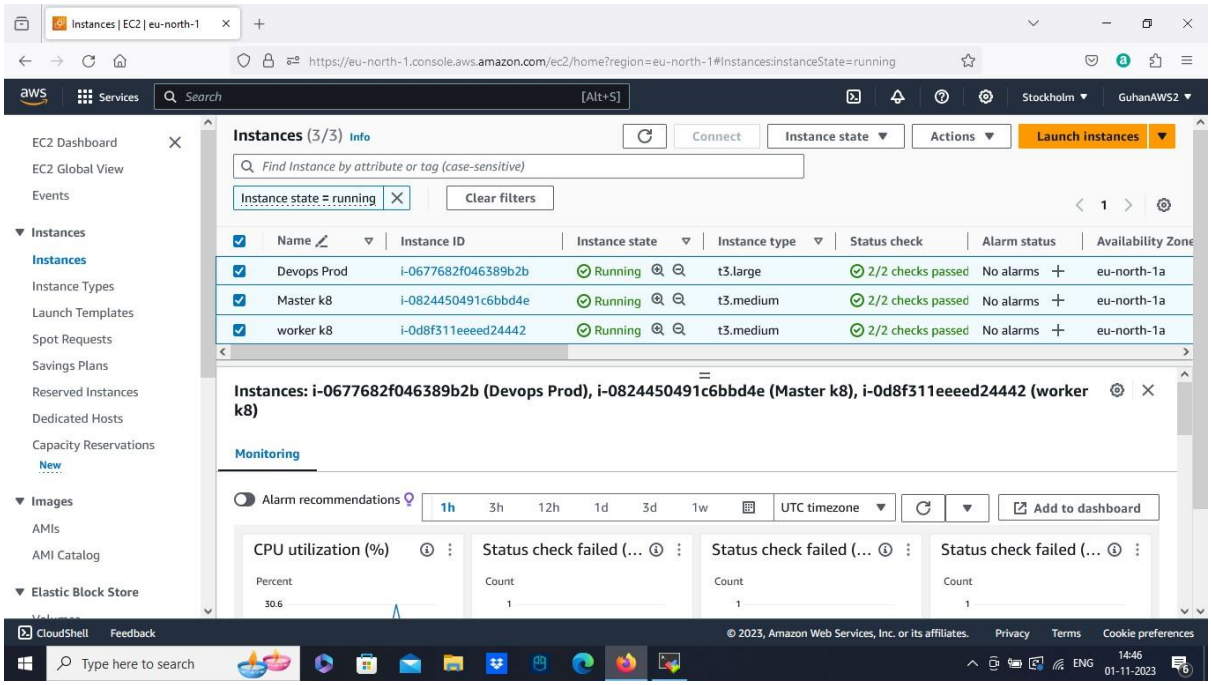
- The next stage will be docker build in our script
- We can use pipeline syntax here with the dokerregistry, add the username and password of your docker hub. And generate the syntax.
- Copy to the main script and add the command sh "docker build -t cicddevops ." The image will be build
- The next stage will be docker push copy the same script and add docker tag and pushin the script.
- Apply it and build now





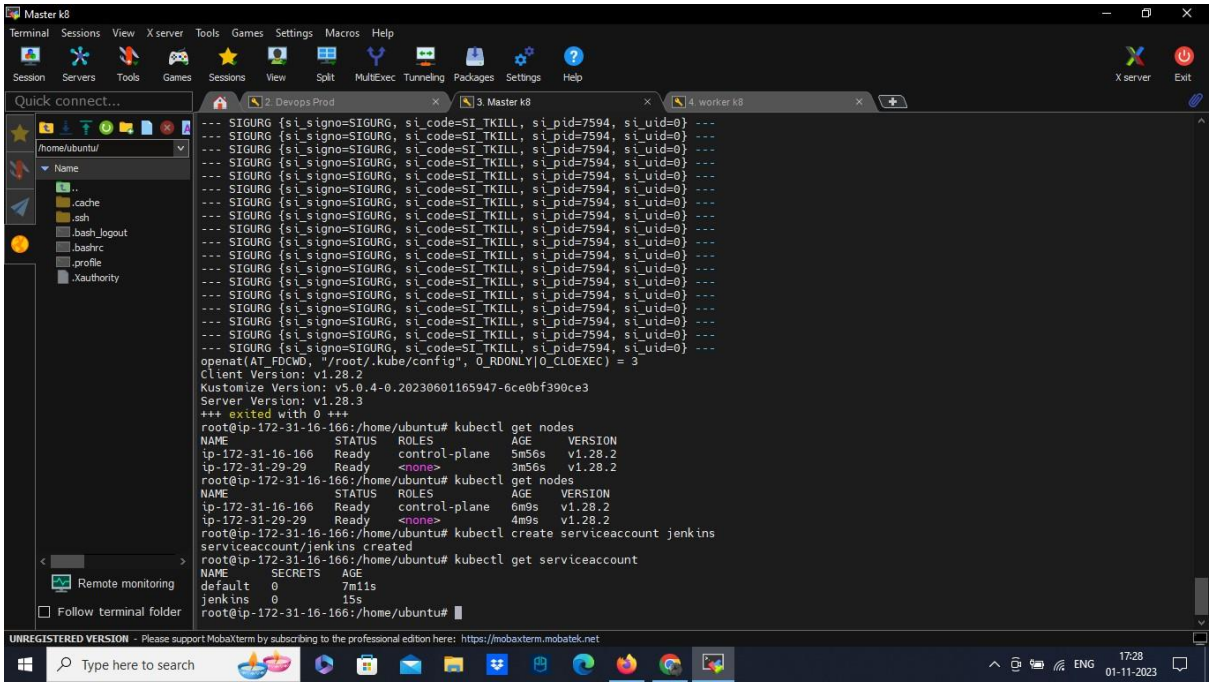
Step 9: Deploy to Kubernetes:

- Now we are going to deploy to the Kubernetes.
- First we will create Kubernetes cluster using AWS.
- Go to AWS, launch 2 instance with AMI type UBUNTU and instance type as medium.
- Make sure to create in same subnet
- Then rename one instance as master and other as worker.
- Same here iam going to use Mobaxterm as my both master and worker terminal.
- And login user as Ubuntu and with our private key.



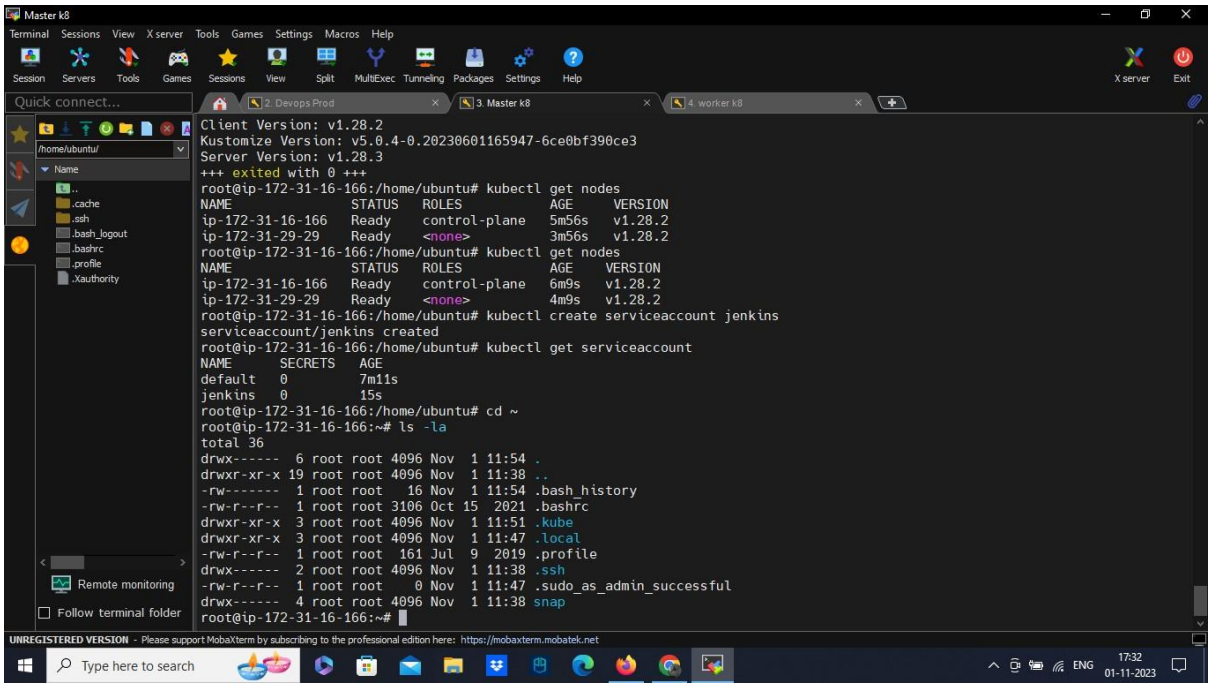
- Kubernetes cluster installation (need to run these commands in both master and worker instance).
  - Switch to root user
    sudo su -
  - Update the System
    apt-get update
  - Install http package
    apt-get install apt-transport-https
- Install Docker
  - apt install docker.io -y
  - docker --version
  - systemctl start docker
  - systemctl enable docker
- Setup open GPG Key
  - sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
- Edit source list file
  nano /etc/apt/sources.list.d/kubernetes.list

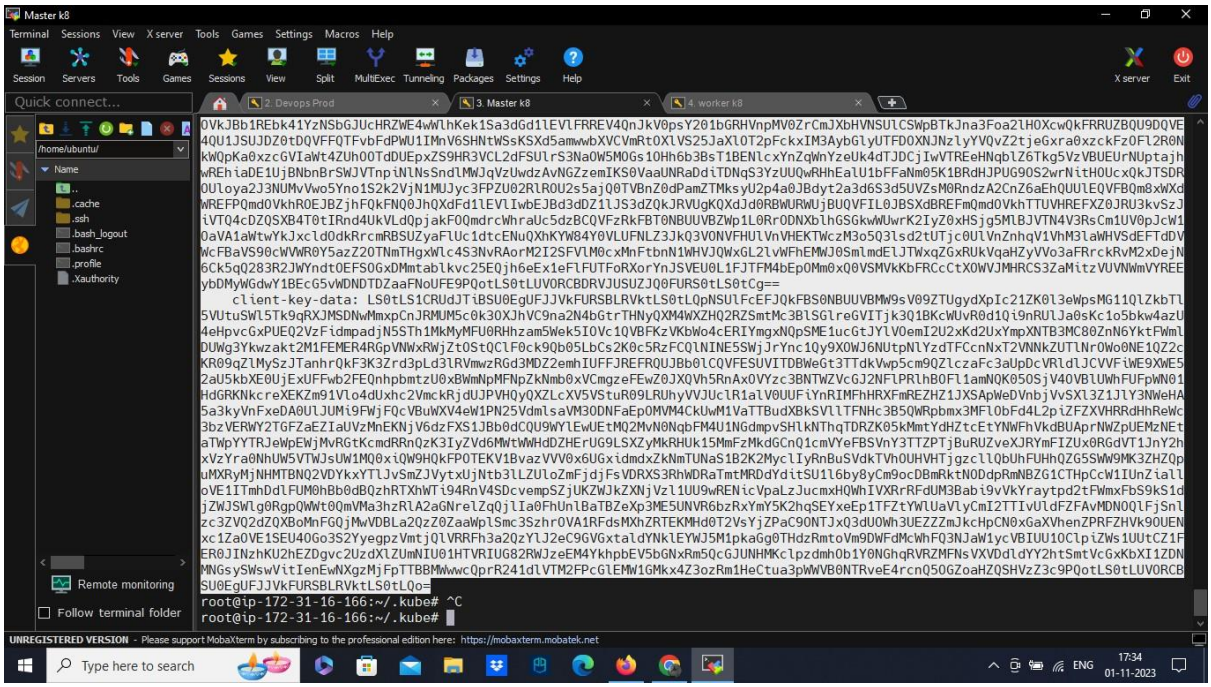
- Add below line in the above file
  deb http://apt.kubernetes.io/ kubernetes-xenial main
- Install the Kubernetes packages
  - apt-get update
  - apt-get install -y kubelet kubeadm kubectl kubernetes-cni
- BOOTSTRAPPING THE MASTER NODE (Only in MASTER Node) - kubeadm init
- Copy the steps of mkdir.kube which wil be above and run it in master instance.
- Deploy flannel node network - kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
- Configure Worker Nodes - kubeadm join 172.31.14.32:6443 --token mcvd2n.aqvyp59vq3inhtks --discovery-token-ca-cert-hash sha256:5ab6e6d695d31c05a4e4cde8f6e7b14ff0d0b450383e9b255270be90904de128 (copy this in the worker instance only).
- Go to master and run the command kubectl get nodes

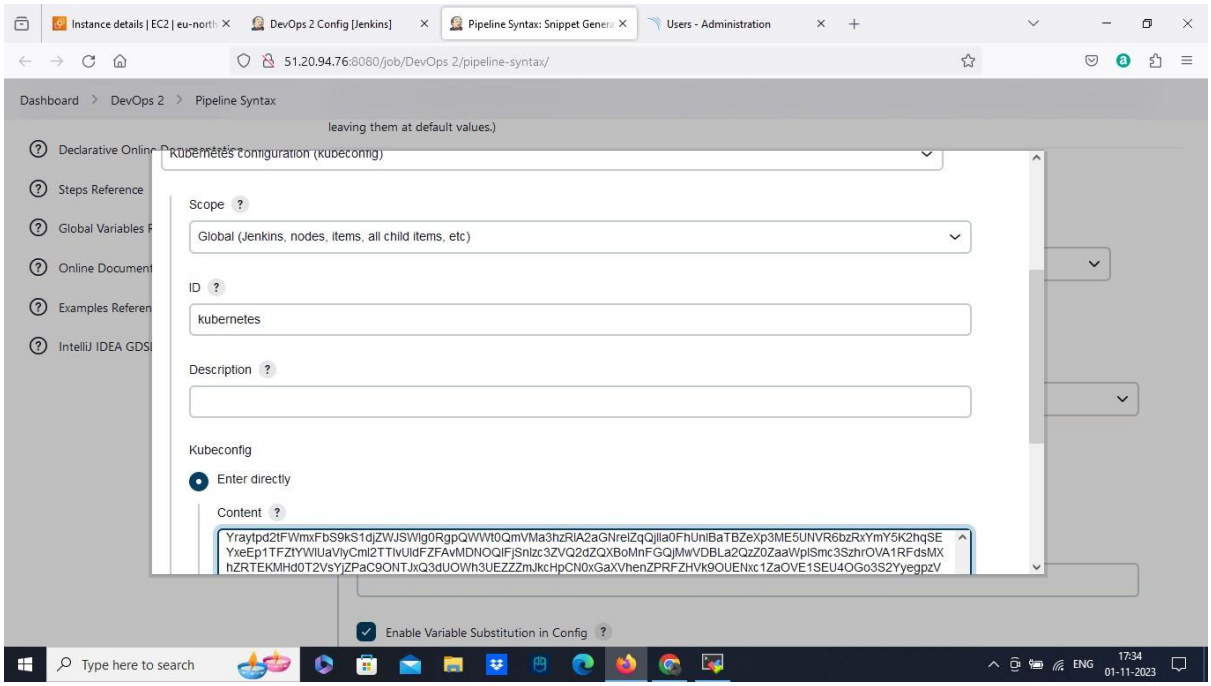


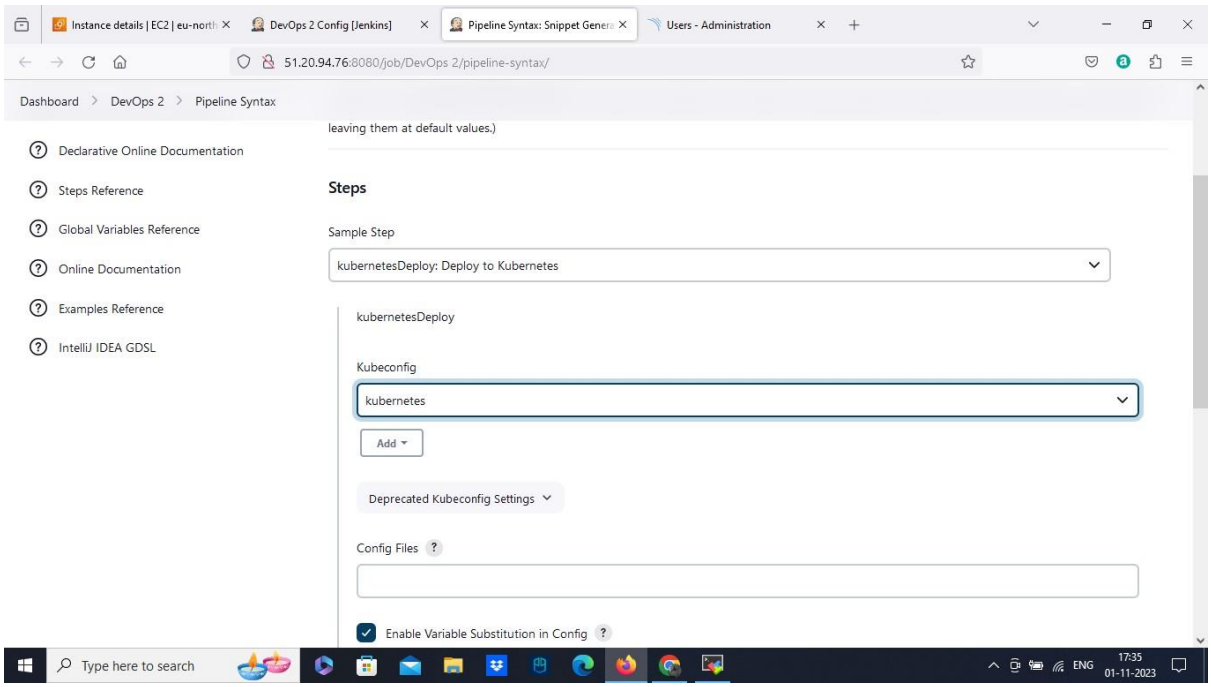
- Create a service account named Jenkins. With the command kubectl create serviceaccount Jenkins, and kubectl get serviceaccount to check it.
- Now go to our scriptline configure. And add Deploy to k8 stage.
- With the help of pipeline syntax. We are using Kubernetes Deploy.
- Here we are going to add kubeconfig file as a credential format.
- Go to our master instance view the config file from .kube/config, copy that and paste it in the kubeconfig credentials content and save it. Generate the pipeline script.

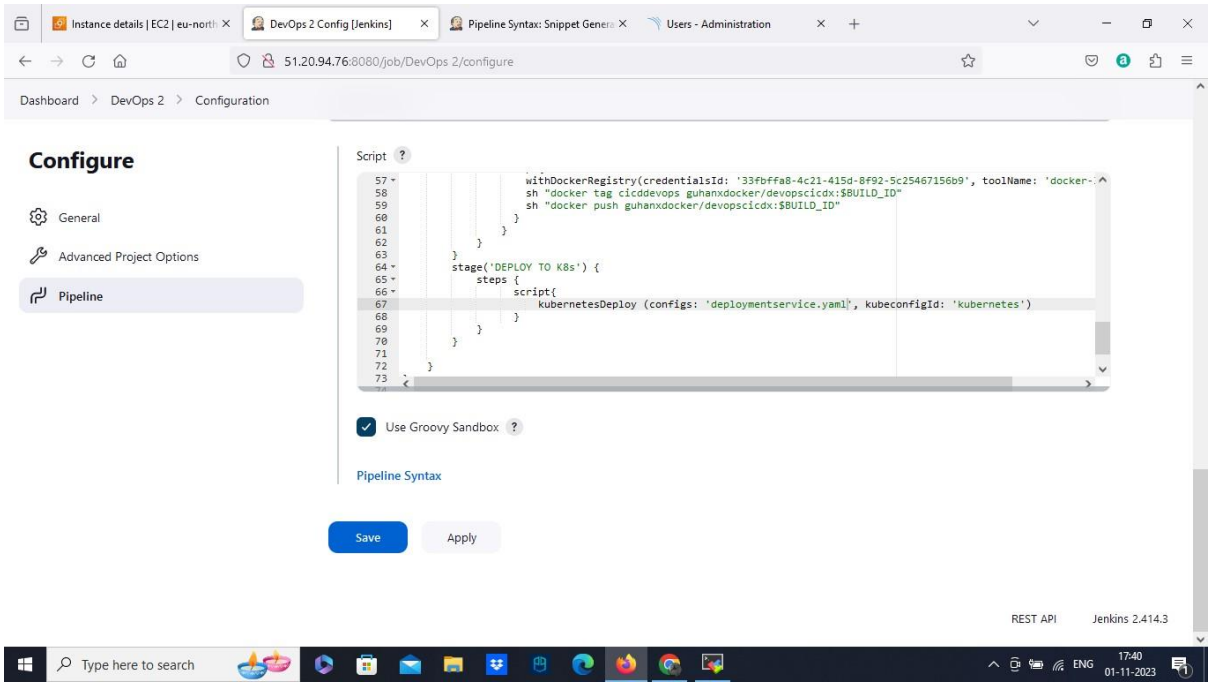
- Copy the generated script to the main script under the stage deploy to k8 and save it.
- Need to change a thing, that is open your git repository, locate the deploymentservice.yaml.
- Open the file from git and change the docker image to our docker image we created and pushed to the hub.
- That’s it save it. And build now.

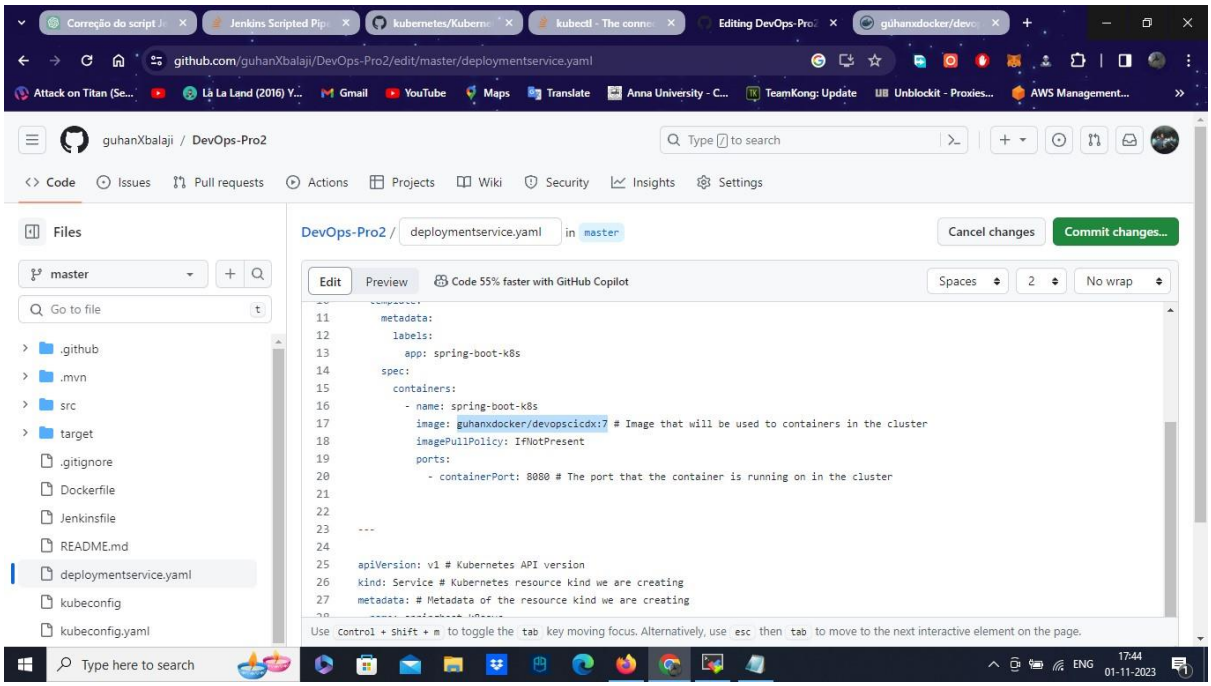


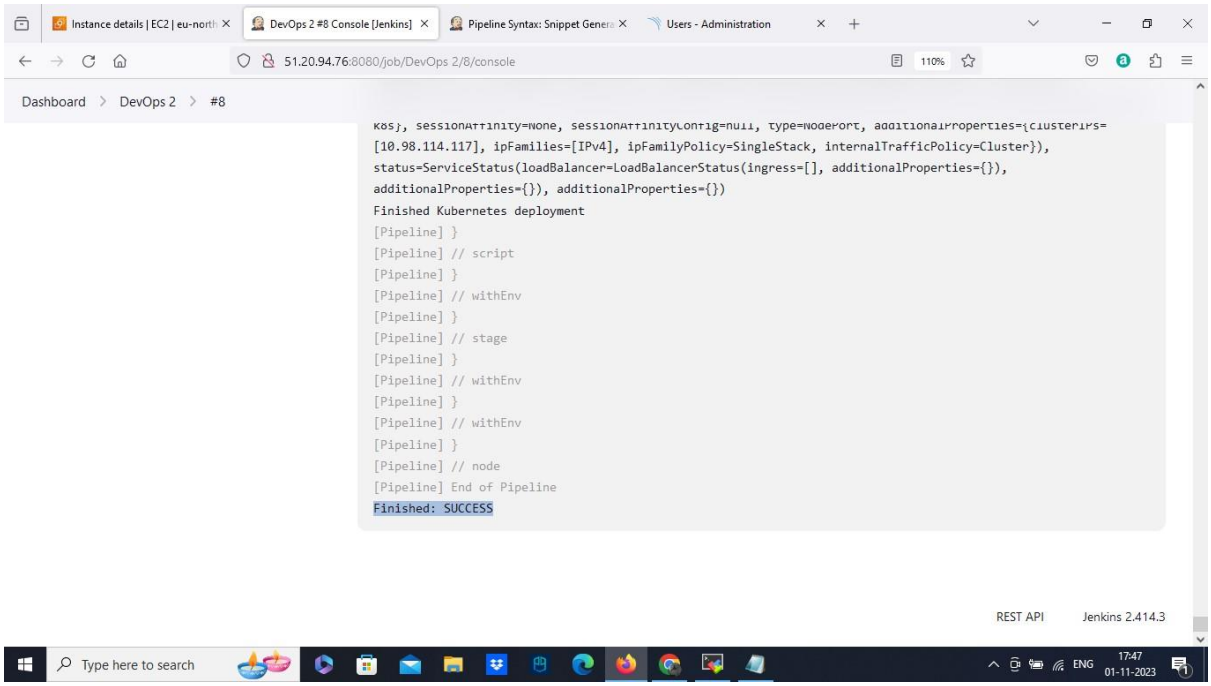


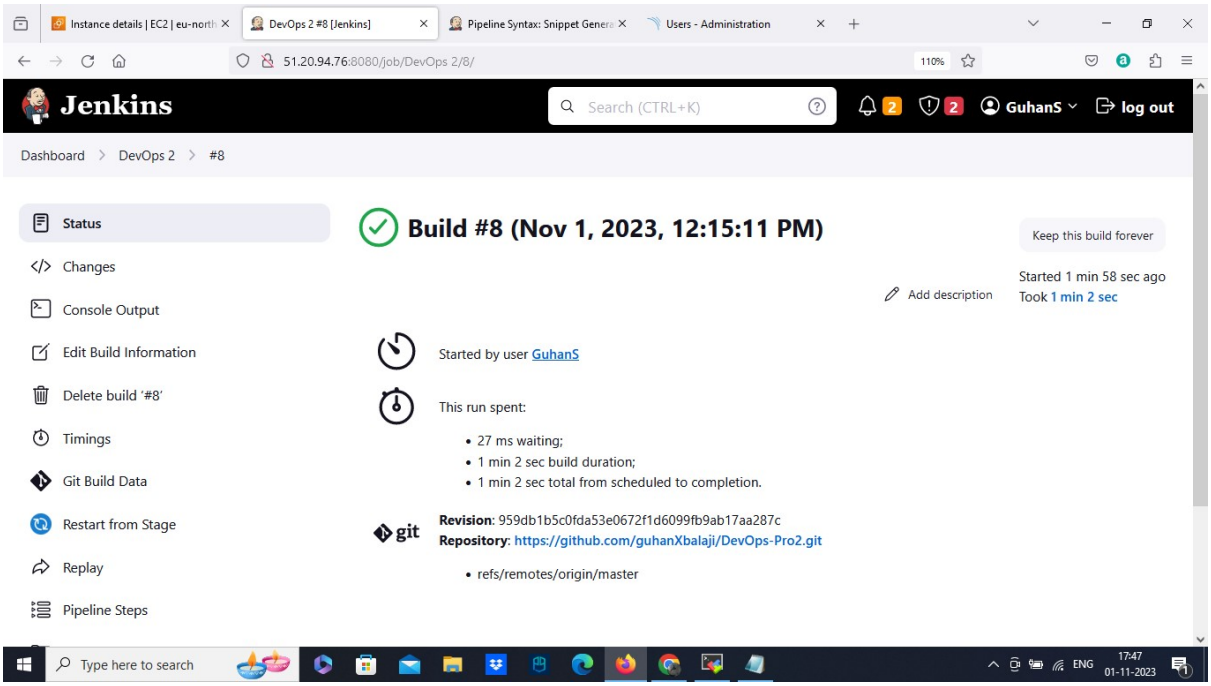

Instance details | EC2 | eu-north
DevOps 2 Config [Jenkins]
Pipeline Syntax: Snippet Gener
Users - Administration
51.20.94.76:8080/job/DevOps 2/pipeline-syntax/
Dashboard
DevOps 2
Pipeline Syntax
leaving them at default values.)
Declarative Online Documentation
Steps Reference
Global Variables R
Online Document
Examples Referen
IntelliJ IDEA GDSL
Kubernetes configuration (kubeconfig)
Scope
Global (Jenkins, nodes, items, all child items, etc)
ID
kubernetes
Description
Kubeconfig
Enter directly
Content
YraytpdtFWmxFbS9kS1djZWJSWlg0RgpQWWt0QmVMa3hzRlA2aGNrelZqQjlla0FhUnlBaTBZeXp3ME5UNVR6bzRxYmY5K2hqSE
YxeEp1TFZtYWlUaVlyCml2TTlvUldFZFAvMDNOQlFjSnlzc3ZVQ2dZQXBoMnFGQjMwVDBLa2QzZ0ZaaWplSmc3SzhrOVA1RFdsMX
hZRTEKMHd0T2VsYjZPaC9ONTJxQ3dUOWh3UEZZZmJkcHpCN0xGaXVhenZPRFZHVk9OUENxc1ZaOVE1SEU4OGo3S2YyegpzV
Enable Variable Substitution in Config
Type here to search
ENG
17:34
01-11-2023

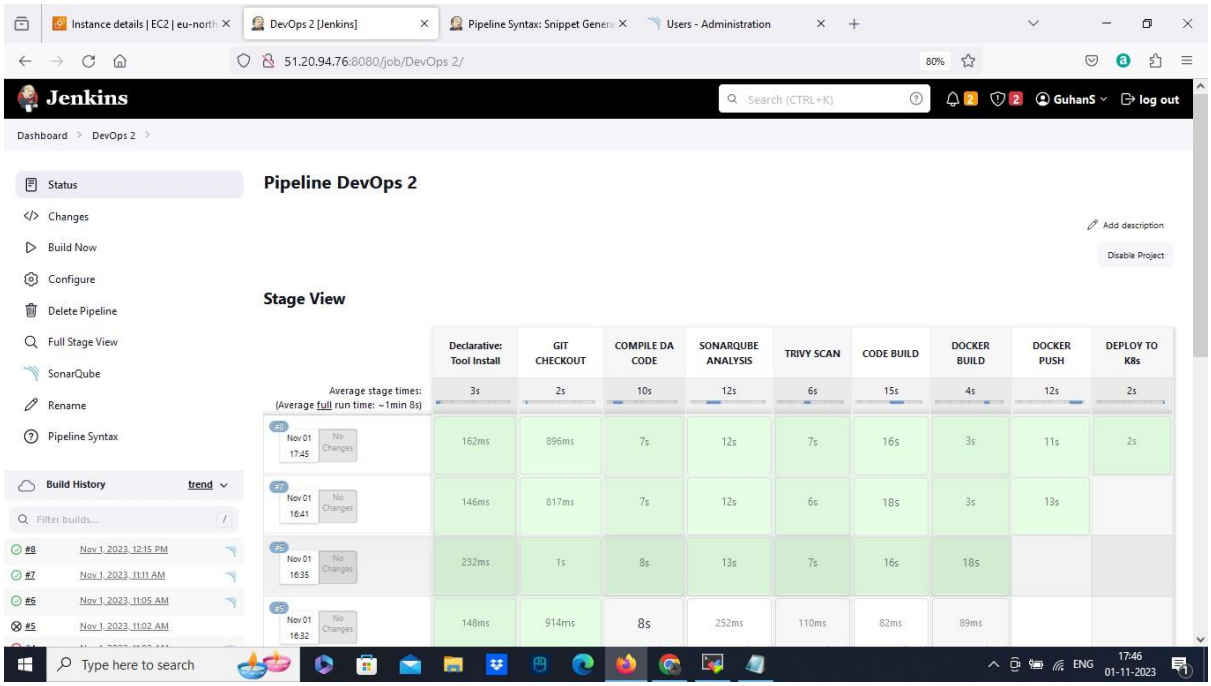

Instance details | EC2 | eu-north
DevOps 2 Config [Jenkins]
Pipeline Syntax: Snippet Gener
Users - Administration
51.20.94.76:8080/job/DevOps 2/pipeline-syntax/
Dashboard
DevOps 2
Pipeline Syntax
leaving them at default values.)
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Steps
Sample Step
kubernetesDeploy: Deploy to Kubernetes
kubernetesDeploy
Kubeconfig
kubernetes
Add
Deprecated Kubeconfig Settings
Config Files
Enable Variable Substitution in Config
Type here to search
ENG
17:35
01-11-2023

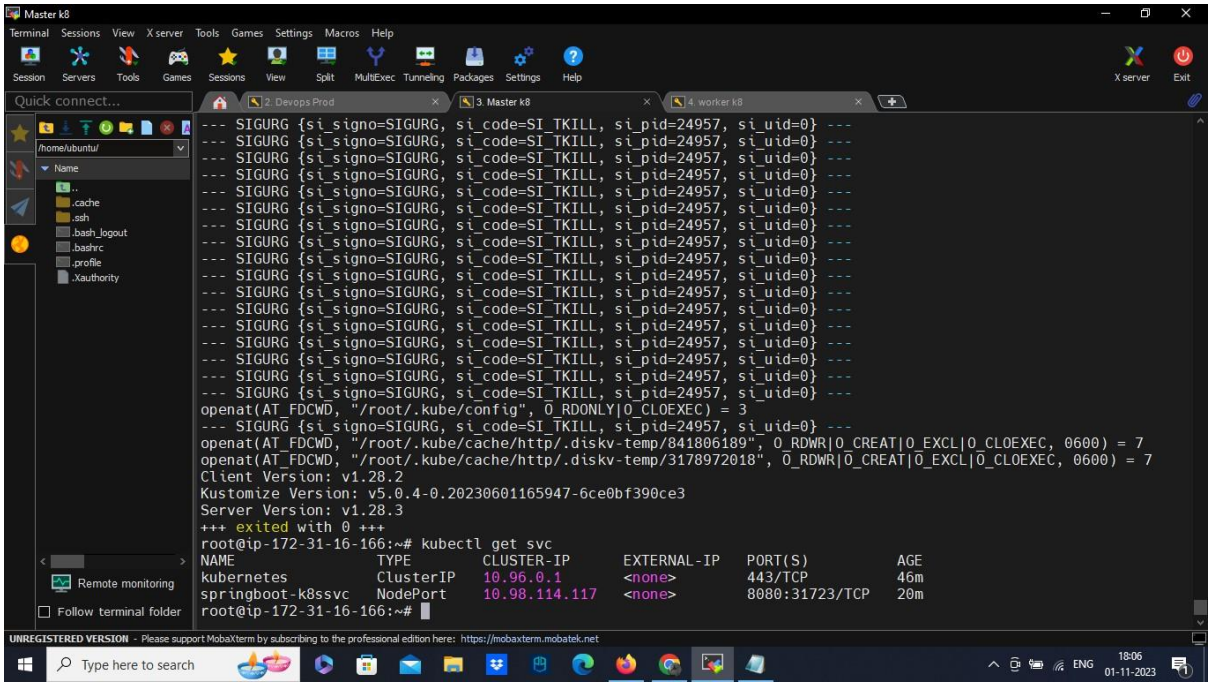

Instance details | EC2 | eu-north
DevOps 2 Config [Jenkins]
Pipeline Syntax: Snippet Gener
Users - Administration
51.20.94.76:8080/job/DevOps 2/configure
Dashboard > DevOps 2 > Configuration
Configure
General
Advanced Project Options
Pipeline
Script
withDockerRegistry(credentialsId: '33fbffa8-4c21-415d-8f92-5c25467156b9', toolName: 'docker-
sh "docker tag cicddevops guhanxdocker/devopscicdx:$BUILD_ID"
sh "docker push guhanxdocker/devopscicdx:$BUILD_ID"
stage('DEPLOY TO K8s') {
steps {
script{
kubernetesDeploy (configs: 'deploymentservice.yaml', kubeconfigId: 'kubernetes')
Use Groovy Sandbox
Pipeline Syntax
Save
Apply
REST API
Jenkins 2.414.3
Type here to search
ENG
17:40
01-11-2023


github.com/guhanXbalaji/DevOps-Pro2/edit/master/deploymentservice.yaml
guhanXbalaji / DevOps-Pro2
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
master
Go to file
.github
.mvn
src
target
.gitignore
Dockerfile
Jenkinsfile
README.md
deploymentservice.yaml
kubeconfig
kubeconfig.yaml
DevOps-Pro2 / deploymentservice.yaml in master
Cancel changes
Commit changes...
Edit
Preview
Code 55% faster with GitHub Copilot
Spaces
2
No wrap
metadata:
labels:
app: spring-boot-k8s
spec:
containers:
- name: spring-boot-k8s
image: guhanxdocker/devopscicdx:7 # Image that will be used to containers in the cluster
imagePullPolicy: IfNotPresent
ports:
- containerPort: 8080 # The port that the container is running on in the cluster
---
apiVersion: v1 # Kubernetes API version
kind: Service # Kubernetes resource kind we are creating
metadata: # Metadata of the resource kind we are creating
Use control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.
Type here to search
ENG
17:44
01-11-2023


Instance details | EC2 | eu-north
DevOps 2 #8 Console [Jenkins]
Pipeline Syntax: Snippet Gener
Users - Administration
51.20.94.76:8080/job/DevOps 2/8/console
110%
Dashboard > DevOps 2 > #8
kasj, sessionAffinity=None, sessionAffinityConfig=null, type=NodePort, additionalProperties={clusterIPs=
[10.98.114.117], ipFamilies=[IPv4], ipFamilyPolicy=SingleStack, internalTrafficPolicy=Cluster}),
status=ServiceStatus(loadBalancer=LoadBalancerStatus(ingress=[], additionalProperties={}),
additionalProperties={}), additionalProperties={})
Finished Kubernetes deployment
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.414.3
Type here to search
ENG
17:47
01-11-2023


Instance details | EC2 | eu-north
DevOps 2 #8 [Jenkins]
Pipeline Syntax: Snippet Gener
Users - Administration
51.20.94.76:8080/job/DevOps 2/8/
110%
Jenkins
Search (CTRL+K)
2
2
GuhanS
log out
Dashboard > DevOps 2 > #8
Status
Changes
Console Output
Edit Build Information
Delete build '#8'
Timings
Git Build Data
Restart from Stage
Replay
Pipeline Steps
Build #8 (Nov 1, 2023, 12:15:11 PM)
Keep this build forever
Add description
Started 1 min 58 sec ago
Took 1 min 2 sec
Started by user GuhanS
This run spent:
• 27 ms waiting;
• 1 min 2 sec build duration;
• 1 min 2 sec total from scheduled to completion.
git
Revision: 959db1b5c0fda53e0672f1d6099fb9ab17aa287c
Repository: https://github.com/guhanXbalaji/DevOps-Pro2.git
• refs/remotes/origin/master
Type here to search
ENG
17:47
01-11-2023





Now we can see our Application got build and successfully deployed to Kubernetes cluster. You can also copy the ip and paste it in the browser for checking the Application. This is how the complete CICD implementation works in DevOps.