

## Chat Bot deployment in Kubernetes

### **Launch a EC2 Ubuntu Instance:**

Launch a Ubuntu(22.04) T2 Large Instance with 30 GB storage. Add HTTP and HTTPS settings in the security group and open the nessasary ports.

### **Create a IAM Role:**

Search “IAM” in AWS console and click on “Rules”. Click on “Create rule” button.

Select “AWS service” in Trusted entity type.

Select “EC2” in Use case and click next.

Select “Administrator Access” and click next.

Give a name to the role as “chatbot” and create the role.

### **Attach the role to the EC2 instance:**

Select the ec2 instance. Go to “Actions” -> “Security” -> “Modify IAM Role”.

Select the role that is newly created and click “Update IAM role” button.

Now connect to the EC2 instance using Mobaxterm.

### **Installation of Jenkins:**

In the EC2 console, do the below:

```
vi jenkins.sh
```

```
#!/bin/bash

sudo apt update -y

wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public | tee
/etc/apt/keyrings/adoptium.asc

echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/deb
$(awk -F= '/^VERSION_CODENAME/{print$2}' /etc/os-release) main" | tee
/etc/apt/sources.list.d/adoptium.list

sudo apt update -y

sudo apt install temurin-17-jdk -y

/usr/bin/java --version

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

```
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \  
      https://pkg.jenkins.io/debian-stable binary/ | sudo tee \  
      /etc/apt/sources.list.d/jenkins.list > /dev/null  
  
sudo apt-get update -y  
  
sudo apt-get install jenkins -y  
  
sudo systemctl start jenkins
```

Give execution permission to the jenkins.sh file:

```
sudo chmod 777 jenkins.sh  
  
sudo su  
  
.jenkins.
```

### Install Docker:

```
sudo apt-get update  
  
sudo apt-get install docker.io -y  
  
sudo usermod -aG docker $USER #my case is ubuntu  
  
newgrp docker  
  
sudo chmod 777 /var/run/docker.sock
```

### Install Sonarqube using a docker container:

```
docker run -d --name sonar -p 9000:9000 sonarqube:its-community
```

### Install Trivy, Kubectl, Terraform:

```
vi script.sh  
  
sudo apt-get install wget apt-transport-https gnupg lsb-release -y  
  
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee  
/usr/share/keyrings/trivy.gpg > /dev/null  
  
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb  
$(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list  
  
sudo apt-get update  
  
sudo apt-get install trivy -y  
  
# Install Terraform  
  
sudo apt install wget -y  
  
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o  
/usr/share/keyrings/hashicorp-archive-keyring.gpg
```

```

echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com ${lsb_release -cs} main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list

sudo apt update && sudo apt install terraform

# Install kubectl

sudo apt update

sudo apt install curl -y

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

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

kubectl version --client

# Install AWS CLI

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

sudo apt-get install unzip -y

unzip awscliv2.zip

sudo ./aws/install

```

Give permission and run script:

```

sudo chmod 777 script.sh
./script.sh

```

### **Jenkins Dashboard:**

Now login to the Jenkins Dashboard by using <ec2 public ip address>:8080 in a browser.

Select “Install suggested plugins”.

Create First Admin User and create a user click on save and continue.

### **Sonarqube Dashboard:**

Enter username as “admin” & password as “admin”

### **In Jenkins Dashboard:**

Install the below Plugins: Go to “Manage Jenkins” -> “Plugins” -> “Available Plugins

1. Eclipse Temurin Installer
2. SonarQube Scanner
3. Nodejs
4. Docker
5. Docker commons
6. Docker pipeline
7. Docker API
8. Docker build step
9. QWASP dependency check
10. Terraform
11. Kubernetes
12. Kubernetes CLI
13. Kubernetes client API
14. Kubernetes :: Pipeline :: Devops steps
15. Kubernetes Credentials
16. Kubernetes credentials provider
17. Blue Ocean

Configure Java and nodejs in Global Tool Configuration:

Go to “Manage Jenkins” -> “Tools”

Click “Add JDK”



Click “Add NodeJS”

## NodeJS installations

Add NodeJS

### NodeJS

Name

node19

Install automatically ?

#### Install from nodejs.org

Version

NodeJS 19.0.0

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'

Click “Add SonarQube”

## SonarQube Scanner

Name

sonar-scanner

! Required

Install automatically ?

#### Install from Maven Central

Version

SonarQube Scanner 5.0.1.3006

Add Installer ▾

Click “Add Dependency-Check”

Add Dependency-Check

## Dependency-Check

Name

DP-Check

Install automatically ?

#### Install from github.com

Version

dependency-check 9.0.9

Add Installer ▾

Click “Add Docker”

## Docker installations

Add Docker

**Docker**

Name  
docker

Install automatically ?

**Download from docker.com**

Docker version ?  
latest

Add Installer ▾



Click “Add Terraform”

## Terraform installations

Add Terraform

**Terraform**

Name  
terraform

Install directory  
/usr/bin

Install automatically ?



Click Apply and Save.

In Sonarqube Dashboard:

Click on “Administration” -> “Security” -> “Users”

Administration

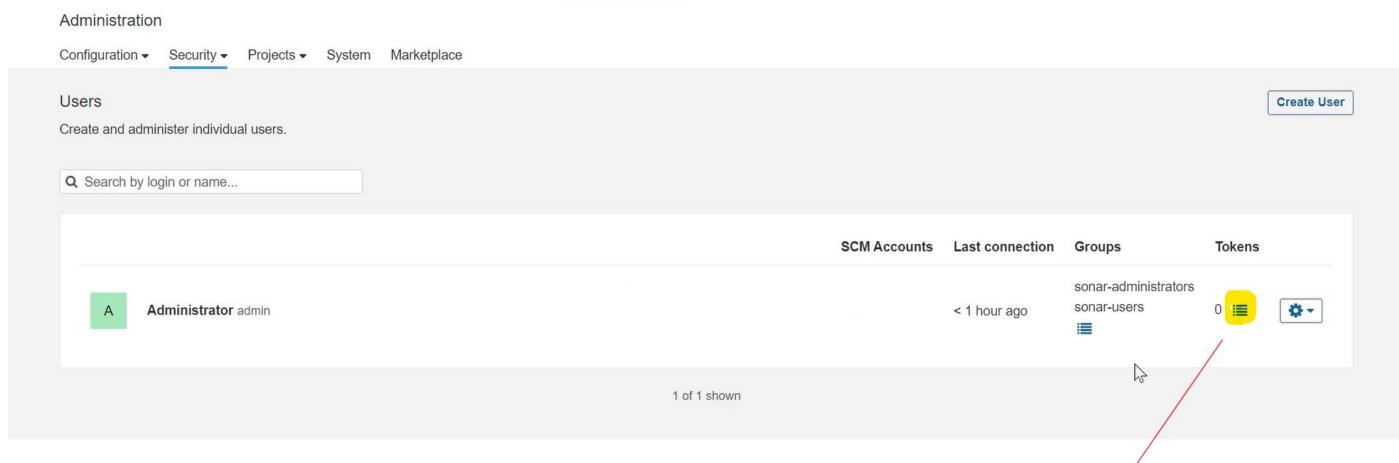
Configuration ▾ Security ▾ Projects ▾ System Marketplace

Users  
Create and administer individual users.

Search by login or name...

	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	0

1 of 1 shown



Click Tokens

Give Token name and create. Save the token in a notepad.

The screenshot shows the 'Tokens of Administrator' page. At the top, there's a 'Generate Tokens' section with fields for 'Name' (Enter Token Name) and 'Expires in' (30 days). A 'Generate' button is present. Below this, a message says 'New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!' with a 'Copy' button and the token value 'squ\_0f3c9d6b2212402b2b0310d2f9fa142722be8065'. A table lists tokens: Jenkins (User, Never, April 21, 2024, May 21, 2024, Revoke button). A 'Done' button is at the bottom right.

### Add Quality Gate in Sonarqube Dashboard:

Go to “Administration” -> “Configuration” -> “Webhooks”. Add the below and click “Create”.

The screenshot shows the 'Create Webhook' dialog. It has fields for 'Name \*' (jenkins), 'URL \*' (http://13.233.140.233:8080/sonarqube-webhook/), and a 'Secret' field (empty). Below the URL field is a note: '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"'.

At the bottom are 'Create' and 'Cancel' buttons, with 'Create' being highlighted by a cursor icon.

### Add Sonarqube secret text in Jenkins Credentials:

Go to the Jenkins Dashboard -> “Manage Jenkins” -> “Credentials”. Add sonar token in credentials.

**Jenkins**

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

### New credentials

Kind: Secret text

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

Secret: Sonar-token

ID: Sonar-token

Description: Sonar-token

**Create**

### Add DockerHub Username and Password:

Username with password

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

Username: rkiruthiga

Treat username as secret

Password: Sonar-token

ID: docker

Description: docker

**Create**

### Configure system tools for sonarqube:

Go to “Manage Jenkins” -> “System”. Click “Add SonarQube”.

Name: sonar-server

Server URL: http://13.233.140.233:9000

Server authentication token: Sonar-token

**+ Add**

**Advanced**

Click Apply and Save.

## Create EKS Cluster from Jenkins using Terraform.

Go to legacy Branch in GitHub repository.

There is a folder named “Eks-terraform”. Here backend.tf file is available. (If needed: change the S3 bucket name as per the preference).

### Create a Pipeline for EKS cluster creation:

Click “+New Item”. Enter the Name as “Terraform-EKS” and select as “Pipeline”. Click OK.

The screenshot shows the Jenkins 'New Item' creation interface. In the top bar, there is a placeholder text 'Enter an item name'. Below it, a red box highlights the input field where 'Terraform-EKS' has been typed. A small note '» Required field' is visible below the input field. Below the input field, there are four project types listed with their icons: 'Freestyle project', 'Maven project', 'Pipeline', and 'Multi-configuration project'. The 'Pipeline' option is also highlighted with a red box. Each option has a brief description below it. The 'Pipeline' description states: '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.'

Select “Discard Old Builds” and mention 3 in “Max # of builds to keep”.

Select “This project is parameterized”. Click “Add parameter” button and Select “Choice Parameter”.

This project is parameterized ?

The screenshot shows the 'Choice Parameter' configuration dialog. It has a header 'Choice Parameter ?' with a close button 'X'. Below the header, there is a 'Name' field with a question mark icon, which contains the value 'action'. The entire dialog is enclosed in a dashed border.

Choices ?

apply  
destroy

Description ?

use apply to create & use destroy to delete

Plain text [Preview](#)

Add the below script in the pipeline. [Click here to get the Script](#)

```
pipeline{  
    agent any  
    stages {  
        stage('Checkout from Git'){  
            steps{  
                git branch: 'legacy', url: ' https://github.com/Krishpluto/chatbot-ui.git'  
            }  
        }  
        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'  
                }  
            }  
        }  
    }  
}
```

```
        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'  
        }  
    }  
}  
}
```

Click Apply and Save.

Now Select “Build with Parameters”, select “apply” and click “Build”.

**Jenkins**

Dashboard > Terraform-EKS >

**Pipeline Terraform-EKS**

This build requires parameters:

**action**  
use apply to create & use destroy to delete

apply

**Build** **Cancel**

- Status
- </> Changes
- > Build with Parameters
- Configure
- Delete Pipeline
- Full Stage View
- Favorite
- Open Blue Ocean
- Rename
- Pipeline Syntax

## Output:

**Terraform-EKS**

Average stage times:  
(Average full run time: ~8min 52s)

	Checkout from Git	Terraform version	Terraform init	Terraform validate	Terraform plan	Terraform apply/destroy
#1 Apr 21 19:36 No Changes	2s	953ms	6s	4s	4s	8min 28s

**Permalinks**

- Last build (#1), 81 ms ago

**Terraform-EKS 1**

Branch: – Commit: – Started by user kiruthiga

8m 52s 5 minutes ago No changes

Checkout from Git Terraform version Terraform init Terraform validate Terraform plan Terraform apply/destroy End

Terraform apply/destroy - 8m 29s

Restart Terraform apply/destroy

terraform \${action} --auto-approve - Shell Script

Check in your AWS Console, if the EKS cluster is created or not.

The screenshot shows the AWS EKS Clusters page. At the top, there's a navigation bar with 'EKS' and 'Clusters'. Below it, a table titled 'Clusters (1) Info' displays one cluster: 'EKS\_CLOUD' (Status: Active, Kubernetes version: 1.29, Support type: Standard support until March 23, 2025, Provider: EKS). There are buttons for 'Delete' and 'Add cluster'.

Check if EC2 instance is created or not.

The screenshot shows the AWS Instances page. It lists two EC2 instances: 'Main\_Server' (Instance ID: i-08436607f1985e605, State: Running, Type: t2.large) and another instance (Instance ID: i-07a50c3800dbdd436, State: Running, Type: t2.medium). There are buttons for 'Connect', 'Actions', and 'Launch instances'.

Create New pipeline for deploying application.

Go to Jenkins Dashboard -> click "+New Item".

The screenshot shows the Jenkins 'Enter an item name' screen with 'Chatbot-clone' entered. Below it, three project types are listed: 'Freestyle project' (classic job type), 'Maven project' (for Maven projects), and 'Pipeline' (orchestrates long-running activities). The 'Pipeline' option is highlighted with a red box.

Select "Discard Old Builds" and mention 3 in "Max # of builds to keep".

Add the below script in the pipeline script. ([SCRIPT](#))

```
pipeline{
```

```
    agent any
```

```
    tools{
```

```
        jdk 'jdk17'
```

```
        nodejs 'node19'
```

```
}

environment {
    SCANNER_HOME=tool 'sonar-scanner'
}

stages {

    stage('Checkout from Git'){
        steps{
            git branch: 'legacy', url: ' https://github.com/Krishpluto/chatbot-ui.git'
        }
    }

    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }

    stage("Sonarqube Analysis ") {
        steps{
            withSonarQubeEnv('sonar-server') {
                sh """ $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Chatbot \
-Dsonar.projectKey=Chatbot """
            }
        }
    }

    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
            }
        }
    }

    stage('OWASP FS SCAN') {
        steps {

```

```
        dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'

        dependencyCheckPublisher pattern: '**/dependency-check-report.xml'

    }

}

stage('TRIVY FS SCAN') {

    steps {

        sh "trivy fs . > trivyfs.json"

    }

}

stage("Docker Build & Push"){

    steps{

        script{

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

                sh "docker build -t chatbot ."

                sh "docker tag chatbot rkiruthiga/chatbot:latest"

                sh "docker push rkiruthiga/chatbot:latest"

            }

        }

    }

}

stage("TRIVY"){

    steps{

        sh "trivy image rkiruthiga/chatbot:latest > trivy.json"

    }

}

stage ("Remove container") {

    steps{
        sh "docker stop chatbot | true"
        sh "docker rm chatbot | true"
    }
}

stage('Deploy to container'){

    steps{

        sh 'docker run -d --name chatbot -p 3000:3000 rkiruthiga/chatbot:latest'

    }

}
```

```
}
```

```
}
```

```
}
```

Not secure 13.233.140.233:8080/job/Chatbot-clone/

## Jenkins

Dashboard > Chatbot-clone >

Status

Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Favorite

Open Blue Ocean

Rename

Pipeline Syntax

Chatbot-clone

Stage View

Average stage times: (Average full run time: ~36min 14s)

Declarative: Tool Install	Checkout from Git	Install Dependencies	Sonarqube Analysis	quality gate	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Remove container	Deploy to container
3min 33s	1s	32s	41s	674ms	26min 32s	32s	3min 38s	27s	811ms	799ms
Apr 21 19:52	No Changes				26min 32s (paused for 7s)					

Build History trend Filter... April 21, 2024, 2:22 PM Atom feed for all Atom feed for failures

Permalinks

- Last build (#1), 2 min 41 sec ago

REST API Jenkins 2.440.3

Not secure 13.233.140.233:8080/blue/organizations/jenkins/Chatbot-clone/detail/Chatbot-clone/1/pipeline/

## ✓ Chatbot-clone 1

Branch: - 36m 14s No changes Started by user kiruthiga

Commit: - 4 minutes ago

Pipeline Changes Tests Artifacts

Logout

Start Checkout from Git Install Dependencies Sonarqube Analysis quality gate OWASP FS SCAN TRIVY FS SCAN Docker Build & Push TRIVY Remove container Deploy to container End

Deploy to container - <1s

- > jdk17 — Use a tool from a predefined Tool Installation
- > Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.
- > node19 — Use a tool from a predefined Tool Installation
- > Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.
- > docker run -d --name chatbot -p 3000:3000 rkiruthiga/chatbot:latest — Shell Script

Restart Deploy to container

## Dependency Check Reports:

The screenshot shows the Jenkins interface for a dependency check. On the left, there's a sidebar with various build-related links. The main content area has a title "Dependency-Check Results". Below it is a "SEVERITY DISTRIBUTION" chart with three bars: one red bar at level 1, one orange bar at level 9, and one yellow bar at level 8. To the right is a table titled "Dependency-Check Results" with columns for "File Name", "Vulnerability", "Severity", and "Weakness". The table lists 14 entries, each with a color-coded severity indicator (green for OSSINDEX, blue for NVD). The last entry is "postcss:8.4.21" with a severity of "Medium" (yellow) and a CWE ID of "CWE-74". At the bottom of the table is a navigation bar with icons for back, forward, and search.

Jenkins 2.44.0.3

## Trivy Reports:

The screenshot shows the Trivy execution report for a container image. The URL in the address bar is "13.233.140.233:8080/job/Chatbot-clone/lastCompletedBuild/execution/node/3/ws/trivy.json". The table below lists vulnerabilities found in various packages. The columns are: Library, Vulnerability, Severity, Status, Installed Version, Fixed Version, and Title. The table shows multiple entries for "busybox" and "openssl" across different CVE numbers and versions. For example, "busybox" has a critical vulnerability (CVE-2022-48174) in version 1.36.0-r9, fixed in 1.36.1-r1. "openssl" has numerous vulnerabilities across different versions from 3.1.0-r4 to 3.1.4-r0, with severity levels ranging from HIGH to CRITICAL. Each row includes a link to the corresponding NVD entry.

← → ⌂ Not secure 13.233.140.233:8080/job/Chatbot-clone/lastCompletedBuild/execution/node/3/ws/trivyfs.json

Pretty-print

```
package-lock.json (npm)
=====
Total: 10 (UNKNOWN: 0, LOW: 1, MEDIUM: 9, HIGH: 0, CRITICAL: 0)
```

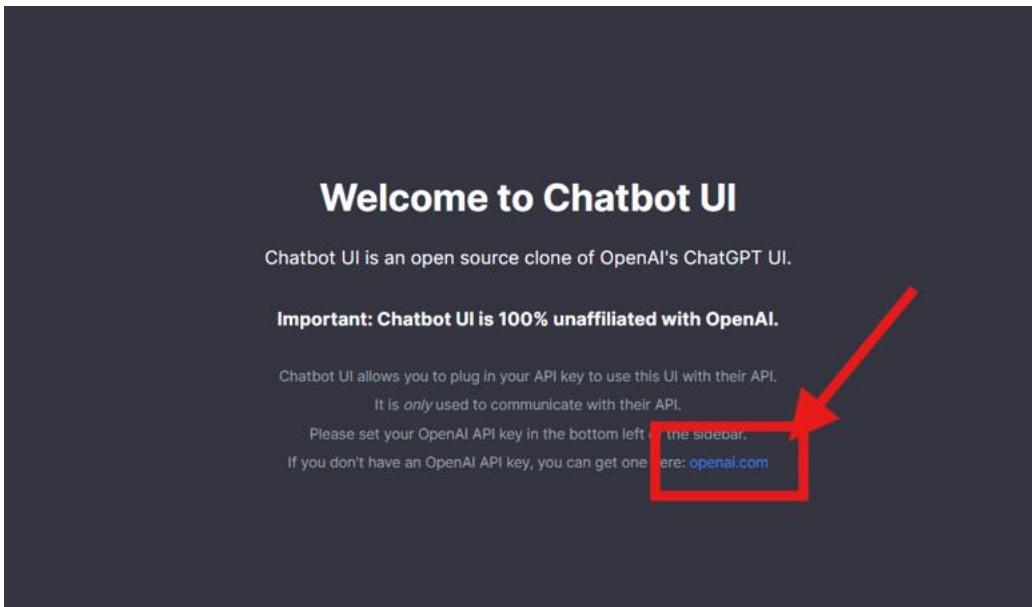
Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title	
axios	CVE-2023-45857	MEDIUM	fixed	0.26.1	1.6.0, 0.28.0	axios: exposure of confidential data stored in cookies <a href="https://avd.aquasec.com/nvd/cve-2023-45857">https://avd.aquasec.com/nvd/cve-2023-45857</a>	
follow-redirects	CVE-2023-26159			1.15.2	1.15.4	follow-redirects: Improper Input Validation due to the improper handling of URLs by... <a href="https://avd.aquasec.com/nvd/cve-2023-26159">https://avd.aquasec.com/nvd/cve-2023-26159</a>	
	CVE-2024-28849				1.15.6	follow-redirects: Possible credential leak <a href="https://avd.aquasec.com/nvd/cve-2024-28849">https://avd.aquasec.com/nvd/cve-2024-28849</a>	
katex	CVE-2024-28243	MEDIUM		0.13.24	0.16.10	KaTeX is a JavaScript library for TeX math rendering on the web.... <a href="https://avd.aquasec.com/nvd/cve-2024-28243">https://avd.aquasec.com/nvd/cve-2024-28243</a>	
	CVE-2024-28245					KaTeX is a JavaScript library for TeX math rendering on the web.... <a href="https://avd.aquasec.com/nvd/cve-2024-28245">https://avd.aquasec.com/nvd/cve-2024-28245</a>	
	CVE-2024-28246					KaTeX is a JavaScript library for TeX math rendering on the web.... <a href="https://avd.aquasec.com/nvd/cve-2024-28246">https://avd.aquasec.com/nvd/cve-2024-28246</a>	
next	CVE-2023-46298	LOW		13.2.4	13.4.20-canary.13	Next.js missing cache-control header may lead to CDN caching empty reply <a href="https://avd.aquasec.com/nvd/cve-2023-46298">https://avd.aquasec.com/nvd/cve-2023-46298</a>	
postcss	CVE-2023-44270	MEDIUM		8.4.14	8.4.31	An issue was discovered in PostCSS before 8.4.31. The vulnerability af .....	
tough-cookie	CVE-2023-26136			4.1.2	4.1.3	tough-cookie: prototype pollution in cookie memstore <a href="https://avd.aquasec.com/nvd/cve-2023-26136">https://avd.aquasec.com/nvd/cve-2023-26136</a>	
word-wrap	CVE-2023-26115		1.2.3	1.2.4	word-wrap: ReDoS <a href="https://avd.aquasec.com/nvd/cve-2023-26115">https://avd.aquasec.com/nvd/cve-2023-26115</a>		

Now go to browser and enter <public ip of jenkins server ip>:3000. You will see the below output

← → ⌂ Not secure 13.233.140.233:3000

The screenshot shows a dark-themed user interface for a Chatbot. At the top, there are navigation icons for back, forward, and search. The URL bar shows the address 13.233.140.233:3000. Below the header, there are two input fields for 'New chat' and 'New prompt'. The main area features a large 'Welcome to Chatbot UI' heading. Below it, a paragraph states: 'Chatbot UI is an open source clone of OpenAI's ChatGPT UI.' A bold note below reads: 'Important: Chatbot UI is 100% unaffiliated with OpenAI.' Further down, instructions for setting an OpenAI API key are provided, along with a link to openai.com. On the left side, there is a sidebar with several options: Import data, Export data, Light mode, OpenAI API Key, and Plugin Keys.

Click on openai.com to generate the API token.



Click on Create new secret key.

A screenshot of the OpenAI Project API keys page. The URL is platform.openai.com/api-keys. The page shows a sidebar with options: Usage (disabled), API keys (selected and highlighted in green), Settings, and Docs. The main content area has a purple banner about introducing projects. A message states "Project API keys have replaced user API keys. We recommend using project based API keys for more granular control over your resources." A "View user API keys" button is available. Below this, instructions say "As an owner of this project, you can view and manage all API keys in this project." and "Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that has leaked publicly." A "Usage page" link is provided. At the bottom, there is a lock icon and a button labeled "+ Create new secret key", which is also highlighted with a red box. Navigation icons for back, forward, and search are at the top right.

Give Name and click on create.

The screenshot shows the 'Project API keys' interface. A modal window titled 'Create new secret key' is open. It has fields for 'Name' (set to 'chatbot'), 'Project' (set to 'Default Project'), and 'Permissions' (set to 'All'). At the bottom are 'Cancel' and 'Create secret key' buttons. The background shows a purple sidebar with 'Introducing projects' and a message about replacing user API keys.

Save the Token and use.

The screenshot shows the 'Project API keys' interface. A modal window titled 'Save your key' is open. It displays the generated token 'sk-proj-Vu2neUVy4zGkXUNC7cceT3B1bkFJIAYdaip' with a 'Copy' button. Below it, 'Permissions' are listed as 'Read and write API resources'. At the bottom are 'Done' and 'Close' buttons. The background shows the same project settings and sidebar as the previous screenshot.

Navigate back to Chatbot UI that we deployed and bottom left you will see “OpenAI API Key”. Click that and give the generated key and save.

**Now in Jenkins MobaXterm instance**, Give the below command:

```
aws eks update-kubeconfig --name <clustername> --region <region>
```

```
2. 13.233.140.233 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
ubuntu@ip-172-31-35-209:~$ aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
Added new context arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD to /home/ubuntu/.kube/config
ubuntu@ip-172-31-35-209:~$ ls -la
total 109332
drwxr-x--- 6 ubuntu ubuntu 4096 Apr 21 15:58 .
drwxr-xr-x 3 root root 4096 Apr 21 13:25 ..
-rw----- 1 ubuntu ubuntu 62 Apr 21 15:55 .xauthority
-rw----- 1 ubuntu ubuntu 280 Apr 21 15:54 .bash_history
-rw-r--r-- 1 ubuntu ubuntu 220 Jan 6 2022 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Jan 6 2022 .bashrc
drwxr-x--- 2 ubuntu ubuntu 4096 Apr 21 13:26 .cache
drwxrwxr-x 2 ubuntu ubuntu 4096 Apr 21 15:58 .kube
-rw-r--r-- 1 ubuntu ubuntu 807 Jan 6 2022 .profile
drwxr-x--- 2 ubuntu ubuntu 4096 Apr 21 13:26 .ssh
-rw-r--r-- 1 ubuntu ubuntu 0 Apr 21 13:28 sudo_as_admin_successful
-rw-r--r-- 1 ubuntu ubuntu 3179 Apr 21 13:45 .vmlinuinfo
-rw-r--r-- 1 ubuntu ubuntu 176 Apr 21 13:33 wget-hsts
drwxr-xr-x 3 root root 4096 Apr 19 16:31 aws
-rw-r--r-- 1 root root 60434506 Apr 21 13:47 awscli2.zip
-rw-rwxrwx 1 ubuntu ubuntu 899 Apr 21 13:38 jenkins.sh
-rw-r--r-- 1 root root 51454104 Apr 21 13:47 kubectl
-rw-rwxrwx 1 ubuntu ubuntu 1251 Apr 21 13:45 script.sh
ubuntu@ip-172-31-35-209:~$ cd .kube/
ubuntu@ip-172-31-35-209:~/ kube$ ls
config
ubuntu@ip-172-31-35-209:~/ kube$ cat config
```

Remote monitoring

Follow terminal folder

ip-172-31-35-209 0% 4.64 GB / 7.74 GB 0.01 Mb/s 0.00 Mb/s 153 min ubuntu /: 35% /boot/efi: 6%

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Copy the contents of the config file and save it in text file in local machine.

```
2. 13.233.140.233 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
ubuntu@ip-172-31-35-209:~$ wxxrwxrwx 1 ubuntu ubuntu 1251 Apr 21 13:45 script.sh
ubuntu@ip-172-31-35-209:~$ cd .kube/
ubuntu@ip-172-31-35-209:~/ kube$ ls
config
ubuntu@ip-172-31-35-209:~/ kube$ cat config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJZHFrRTNhdmdoY2d3RFFZ5ktvWklodmN0QVFTEJR0XdGVEVUTUJFR0ExViUKQXhblq2EzmlawEp1Wshb6N6QWVGd2B5TRBMsqXmREEyTURWUZ3MhpOREEwTVRwE5ERXhNRFZhtTUJVeApFeFkFS0md0VjkBTVRDbxQxWm1WeJ7tvBaWE1322fdaf18MEDdU3FHU01lMR0RRJIBUUVBQTJ0kR3QXdn0VPLCKfVsUJBUMvYjE3cV10H1SwUdLQXBUsE85c1Z5WjBUtR1Qz2tZD1wbHdBa2hHYWlyNDVNkVDcHN5mtF53MjdUrPmVnPkttefdxRUUew0Xp3U1dYdxMbhN5QTRoS1oVTC9xJ3pPZLxz0vdewyDa3ahmU1By0C9RNmV0L3jocAo1M0l0UljkZ1FwZU1YZkyzaWvrlVldV1Bd1JuDUdkUyQ03BIZFU1Nze32jE00kxyVmTT1ASOSFVLM22CSWZUZVY1C1h0WFRUSXVvVU1XrJfaC3dLmTHmGN0ThdHmcjV3V2NTazVnZGo4RmczMEIzeFxgTxLPYOrVRBNeKnhkMFZ2KzcWLAuJ0hamG1HbfdoA0G1JyMf6dbA4VTY2nnU20VjGWHNPuWTS3VHT1lwR1JMMK5aTfAn2hWRw0vXm1X3dvLvp4RvdlDbVBt2d233eWQYKnmcmh1b0g2z2VXEjYJTE0R0ExwRdE0VCL3dRUFU3SUwNREF0Ck1nT1ZIUk1C0W4RUJUQURBUlgvTU1wR0ExwRNEZ1FX0kQJTFVf13bEn1b1pOVKyK2t2VjY1RFNMWkdkVkd6QYK0md0VkhSRUVEakFNZ2dwmcmRXSmxjbTvsZedWek1BMEd0lJFU01lM0RRRJ0d1vB0TR30kFR0ThTTR0LTKz5tAp4LyITkRaSjczV2JtN1JLUnYSEo0VlhneStCc2ZKN2FkT12p0VhqcjkCsV40UcrYndhb0wyT3dPR3ZCKzhcZjnSd2CB5FKNUFLU2d1FR0b0m5vXHmrM2fia014Ukx4QnEywVrsWm16dXpMohwSU8xNluVyn2NUOKYw4xam1PUF5bKnDaU9ReDA2h2RjK25hbl2TzDEV2QVZhKFZn0TRh51LBSTy2b1TcnPB2h0g0Sj3TEx5cmtrQwpkVdhs1L2LnOfLzys2V0zjcvbUhoeWVNSExCUm9jU1R1HzNmL3dnThmVtm9ENG15ttE5nVdpnvk5CUhpMcNrRxSKhIVmQ4b045cGovtDNRNVEnRw01VmR2RkdEzsR0UNGmkhsjBjYY1l0TvxSmExoVfJUlnHOHK3ewXMSkL0f6ckhyhdDdcQstQci0tLs0tRUEIENFUlrJRK1ldQvRLs0tLs0k
      server: https://ap-south-1.eks.amazonaws.com
      name: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
contexts:
- context:
    cluster: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
    user: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
    name: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
current-context: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
kind: Config
preferences: {}
users:
- name: arn:aws:eks:ap-south-1:150187756870:cluster/EKS_CLOUD
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1beta1
      args:
        - --region
        - ap-south-1
        - eks
        - get-token
        - --cluster-name
        - EKS_CLOUD
        - --output
        - json
      command: aws
ubuntu@ip-172-31-35-209:~/ kube$
```

Remote monitoring

Follow terminal folder

ip-172-31-35-209 0% 4.64 GB / 7.74 GB 0.01 Mb/s 0.00 Mb/s 154 min ubuntu /: 35% /boot/efi: 6%

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Now save this in a file and use this file to create a “Secret file” in “Manage Jenkins” -> “Credentials”

The screenshot shows the Jenkins 'New credentials' configuration page. The 'Kind' dropdown is set to 'Secret file'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'File' field contains 'k8s.txt'. The 'ID' field is 'k8s'. The 'Description' field is 'k8s'. A 'Create' button is at the bottom.

Now add the below stage in the pipeline as a last stage:

```
stage('Deploy to kubernets'){\n    steps{\n        script{\n            withKubeConfig(caCertificate: "",\n                clusterName: "",\n                contextName: "",\n                credentialsId: 'k8s',\n                namespace: "",\n                restrictKubeConfigAccess: false,\n                serverUrl: "") {\n                sh 'kubectl apply -f k8s/chatbot-ui.yaml'\n            }\n        }\n    }\n}
```

Configure

General Advanced Project Options Pipeline

```

43      > trivy.json
44    }
45  }
46  }
47  }
48  }
49  }
50  }
51  }
52  }
53  }
54  }
55  }
56  }
57  }
58  }
59  }
60  }
61  }
62  }
63  }
64  }
65  }
66  }
67  }
68  }
69  }
70  }
71  }
72  }
73  }
74  }
75  }
76  }
77  }
78  }
79  }
80  }
81  }
82  }
83  }
84  }

```

Use Groovy Sandbox

Pipeline Syntax

Save Apply

REST API Jenkins 2.440.3

[CLICK THIS LINK TO GET THE FULL SCRIPT](#)

Update the Image in the chatbot-ui yml file in GitHub repository.

Click Apply and Save, Run the build to deploy to EKS.

Dashboard > Chatbot-clone > [Chatbot-clone](#)

Status Changes Build Now Configure Delete Pipeline Full Stage View Favorite SonarQube Open Blue Ocean Rename Pipeline Syntax

Add description Disable Project

### Stage View

Declarative: Tool Install	Checkout from Git	Install Dependencies	Sonarqube Analysis	quality gate	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Remove container	Deploy to container	Deploy to kubernetes
Average stage times: (Average full run time: ~23min 2s)	1min 46s	3s	21s	49s	640ms	15min 53s	30s	2min 32s	42s	920ms	918ms
Apr 21 21:37 1 commit	179ms	4s	10s	56s	606ms (passed for 24s)	5min 14s	27s	1min 25s	57s	1s	1s
Apr 21 19:52 No changes	3min 33s	1s	32s	41s	674ms (passed for 7s)	26min 32s	32s	3min 38s	27s	811ms	799ms

### SonarQube Quality Gate

Chatbot Passed server-side processing Success

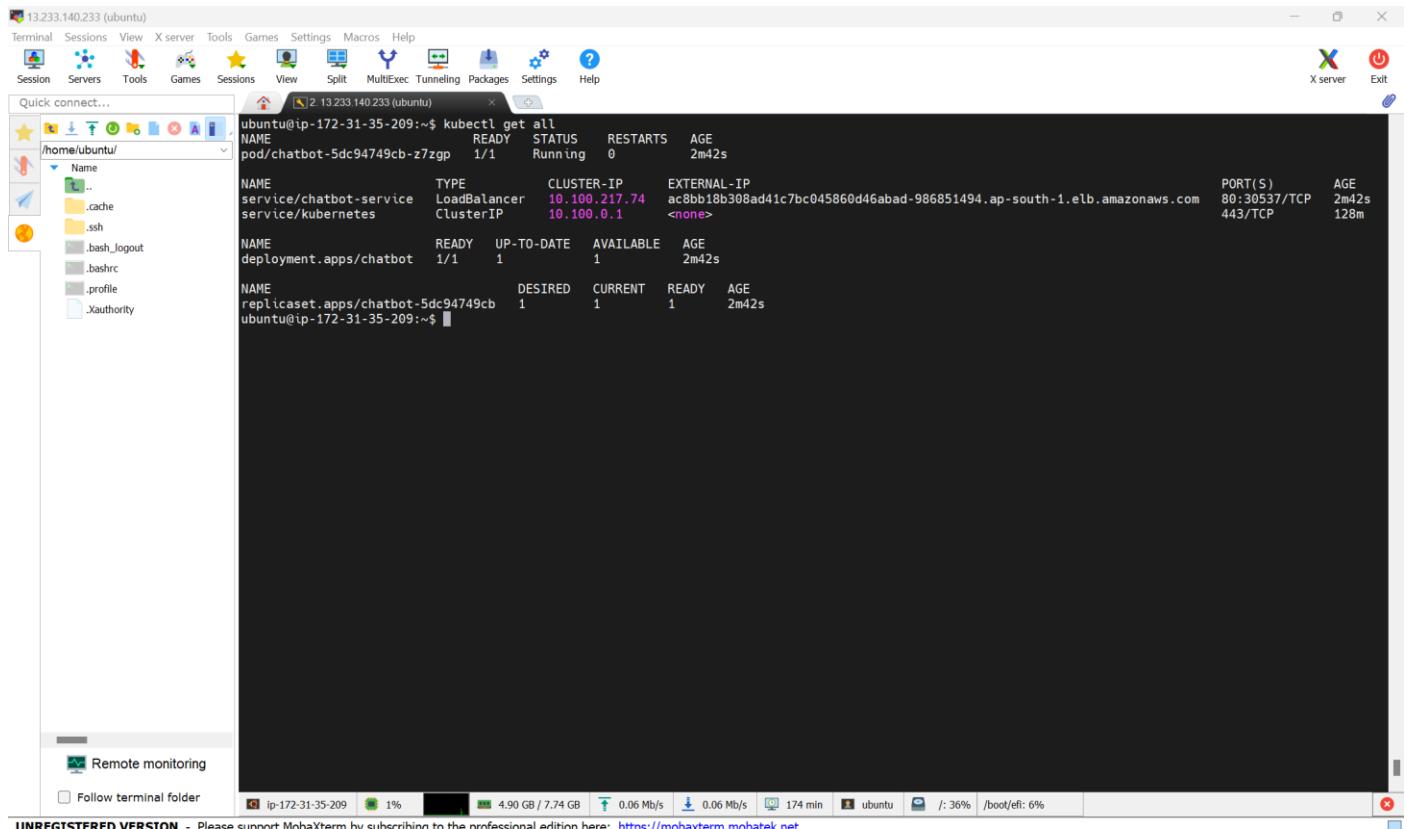
Latest Dependency-Check

Permalinks

- Last build (#1), 1 hr 44 min ago
- Last stable build (#1), 1 hr 44 min ago
- Last successful build (#1), 1 hr 44 min ago
- Last completed build (#1), 1 hr 44 min ago

DECT API Jenkins 2.440.3

Now in Jenkins Mobaxterm instance, Give the below command:



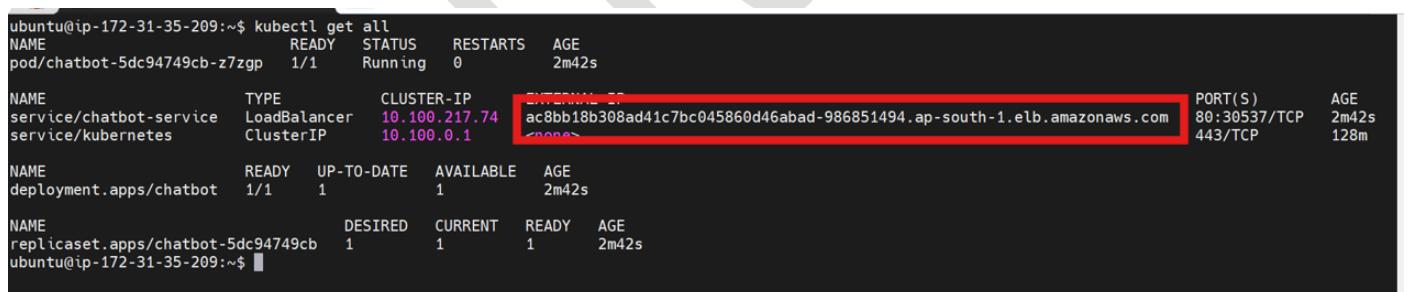
```
ubuntu@ip-172-31-35-209:~$ kubectl get all
NAME           READY   STATUS    RESTARTS   AGE
pod/chatbot-5dc94749cb-z7zgp   1/1     Running   0          2m42s

NAME            TYPE      CLUSTER-IP        EXTERNAL-IP          PORT(S)         AGE
service/chatbot-service   LoadBalancer   10.100.217.74   ac8bb18b308ad41c7bc045860d46abad-986851494.ap-south-1.elb.amazonaws.com   80:30537/TCP   2m42s
service/kubernetes   ClusterIP   10.100.0.1        <none>              443/TCP        128m

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/chatbot   1/1       1           1          2m42s

NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/chatbot-5dc94749cb   1         1       1      2m42s
ubuntu@ip-172-31-35-209:~$
```

Copy the Loadbalancer link and paste it in browser to see the output.

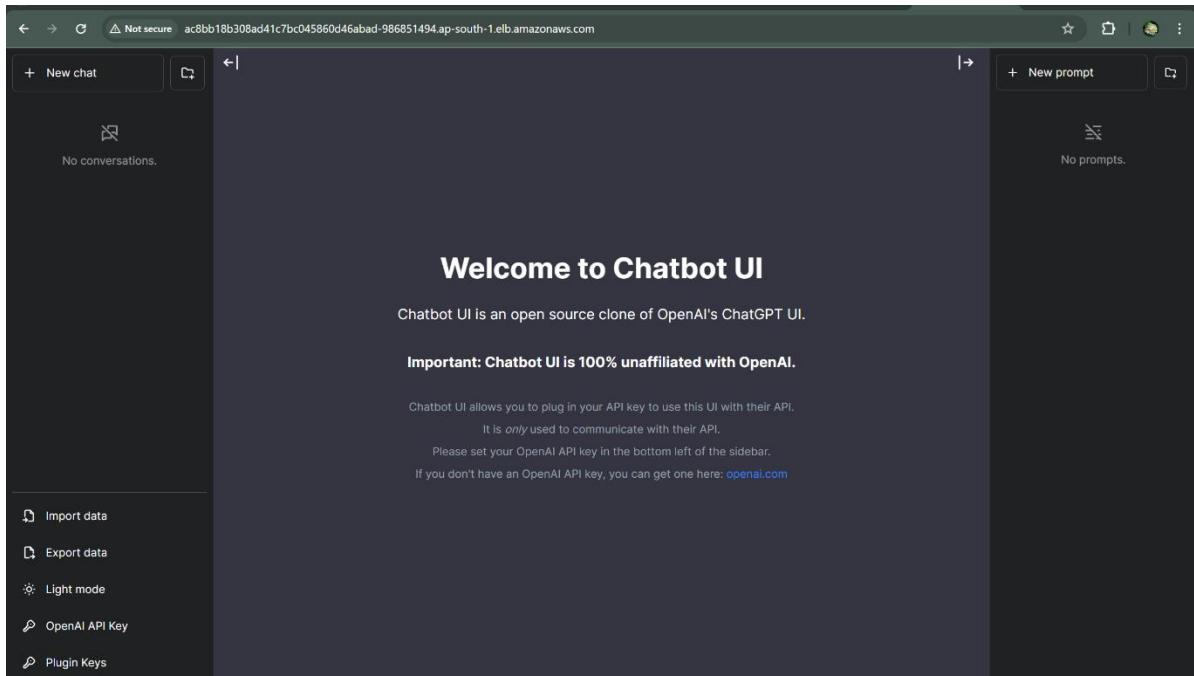


```
ubuntu@ip-172-31-35-209:~$ kubectl get all
NAME           READY   STATUS    RESTARTS   AGE
pod/chatbot-5dc94749cb-z7zgp   1/1     Running   0          2m42s

NAME            TYPE      CLUSTER-IP        EXTERNAL-IP          PORT(S)         AGE
service/chatbot-service   LoadBalancer   10.100.217.74   ac8bb18b308ad41c7bc045860d46abad-986851494.ap-south-1.elb.amazonaws.com   80:30537/TCP   2m42s
service/kubernetes   ClusterIP   10.100.0.1        <none>              443/TCP        128m

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/chatbot   1/1       1           1          2m42s

NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/chatbot-5dc94749cb   1         1       1      2m42s
ubuntu@ip-172-31-35-209:~$
```



## Delete the Resources:

Delete the Deployment by running the below script in the same pipeline (edit the script in the same pipeline)

[\(CLICK HERE\)](#)

```
pipeline{  
    agent any  
    stages {  
        stage ("Remove container") {  
            steps{  
                sh "docker stop chatbot | true"  
                sh "docker rm chatbot | true"  
            }  
        }  
        stage('Delete the Deployment in kubernets') {  
            steps{  
                script{  
                    withKubeConfig(caCertificate: "", clusterName: "", contextName: "", credentialsId: 'k8s', namespace: "",  
restrictKubeConfigAccess: false, serverUrl: "") {  
                        sh 'kubectl delete -f k8s/chatbot-ui.yaml'  
                    }  
                }  
            }  
        }  
    }  
}
```

```
stage('scan cluster'){
    steps{
        script{
            withKubeConfig(caCertificate: "", clusterName: "", contextName: "", credentialsId: 'k8s', namespace: "",
            restrictKubeConfigAccess: false, serverUrl: "") {
                sh 'trivy k8s --report summary cluster'
            }
        }
    }
}
```

This Script will remove the Loadbalancer from EC2 and removes the deployment.

Go to EKS pipeline to destroy the cluster:

Select the “Build with Parameters” and select “destroy” and build

Delete the EC2 instance used to install jenkins server and other packages.