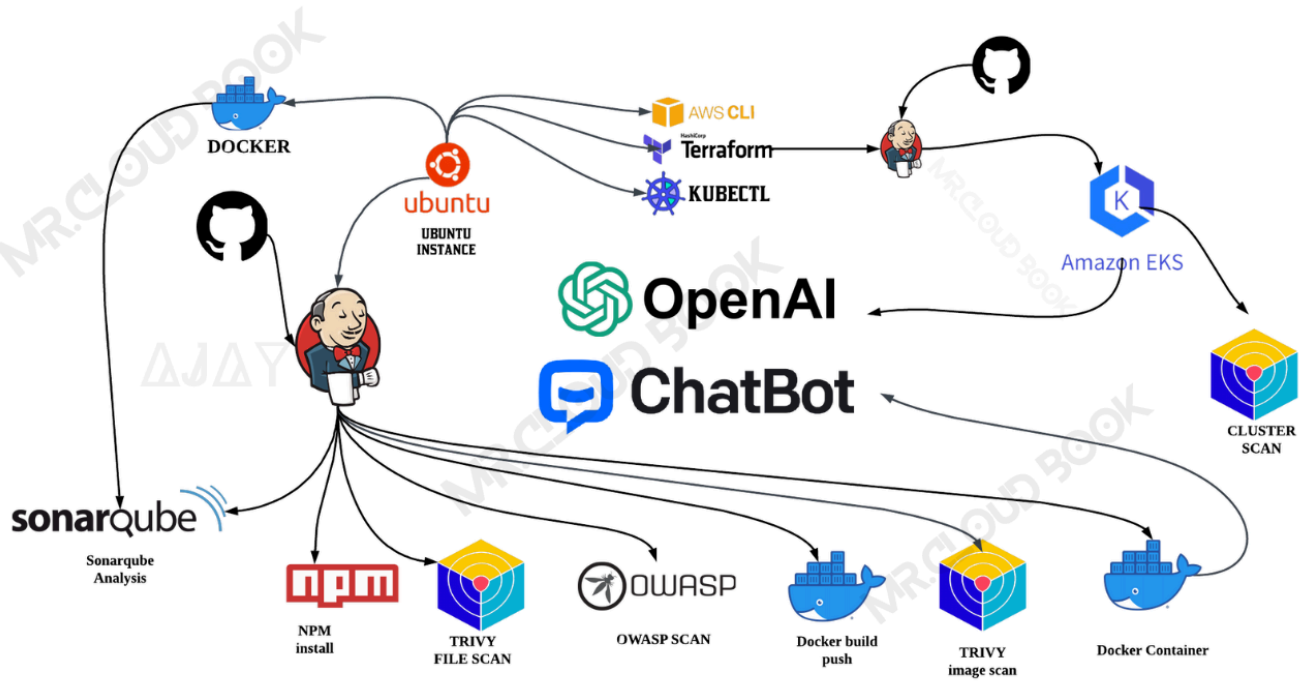
for OpenAI Chatbot UI Deployment | DevSecOps

mrcloudbook.com · 6 March 2024



In today's digital landscape, user engagement is key to the success of any application. From websites to mobile apps, providing users with interactive and personalized experiences is essential. That's why we're thrilled to announce our latest endeavor:

What is ChatBOT?

ChatBOT is an AI language model trained on vast amounts of human conversation data. It's capable of generating human-like text responses based on the input it receives. From answering questions to engaging in conversation, ChatBOT can simulate natural language interactions with users, making it an invaluable tool for enhancing user engagement.

Why ChatBOT?

1. Personalized Interactions: ChatBOT can understand and respond to user queries in a natural and conversational manner, creating personalized interactions that keep users engaged.
2. 24/7 Availability: Unlike human agents, ChatBOT is available 24/7 to assist users, providing instant responses to their queries and ensuring a seamless user experience.
3. Scalability: With ChatBOT deployed in our application, we can handle a large volume of user interactions without compromising performance, ensuring scalability as our user base grows.

How We're Deploying ChatBOT

To deploy ChatBOT on our EKS, we're leveraging Jenkins as our CICD (Continuous Integration/Continuous Deployment) tool and deploying the chatbot within a Docker container. This approach allows us to automate the deployment process, ensuring efficient and reliable delivery of updates and enhancements to our users.

CLICK HERE FOR GITHUB REPO

HERE IS MY REPO WITH TERRAFORM FILES IN LEGACY BRANCH EKS_TERRAFORM

Special thanks to McKay Wrigley, the owner of Open Source, for his dedication to fostering innovation and collaboration in the open-source community.

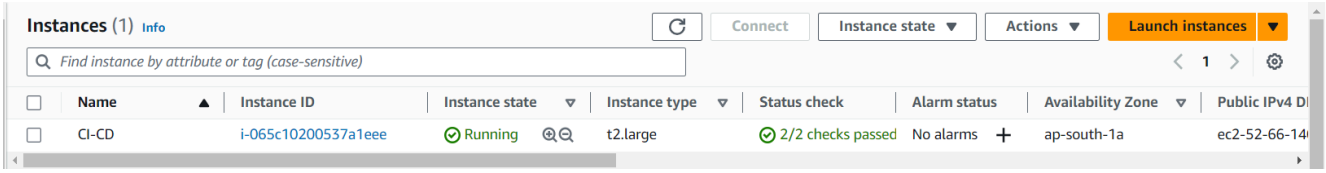
We are grateful for his contributions to the development of technology and for making projects like ChatBOT UI possible.

Now, let's get started and dig deeper into each of these steps:-

Contents [hide]

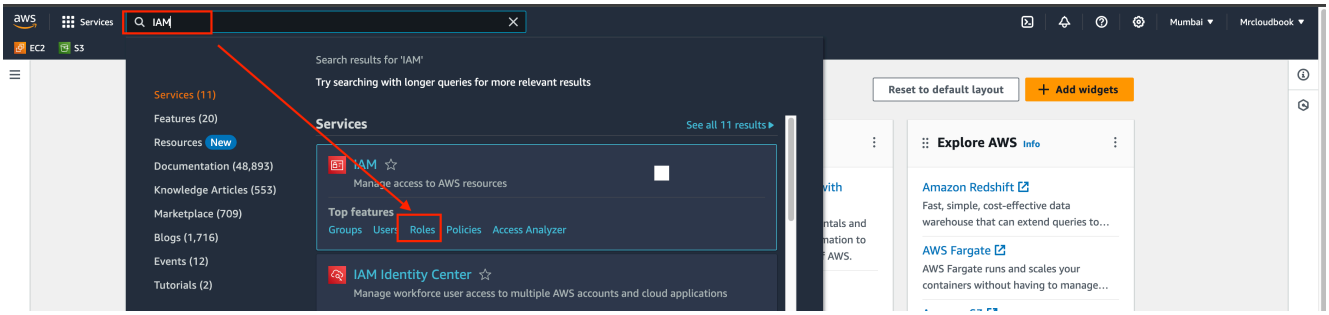
Launch an Ubuntu(22.04) T2 Large Instance

Launch an AWS T2 Large Instance. Use the image as Ubuntu. You can create a new key pair or use an existing one. Enable HTTP and HTTPS settings in the Security Group and open all ports (not best case to open all ports but just for learning purposes it's okay).

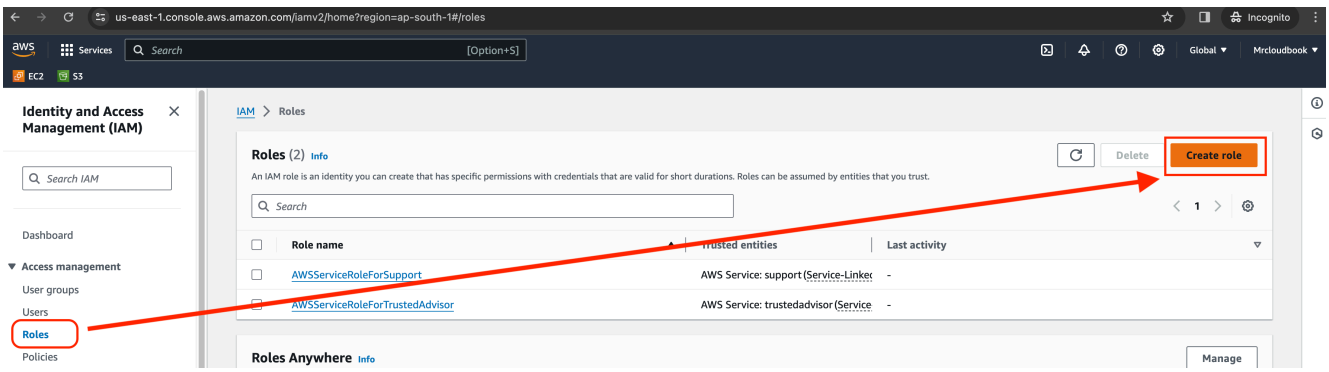


Create IAM role

Search for IAM in the search bar of AWS and click on roles.

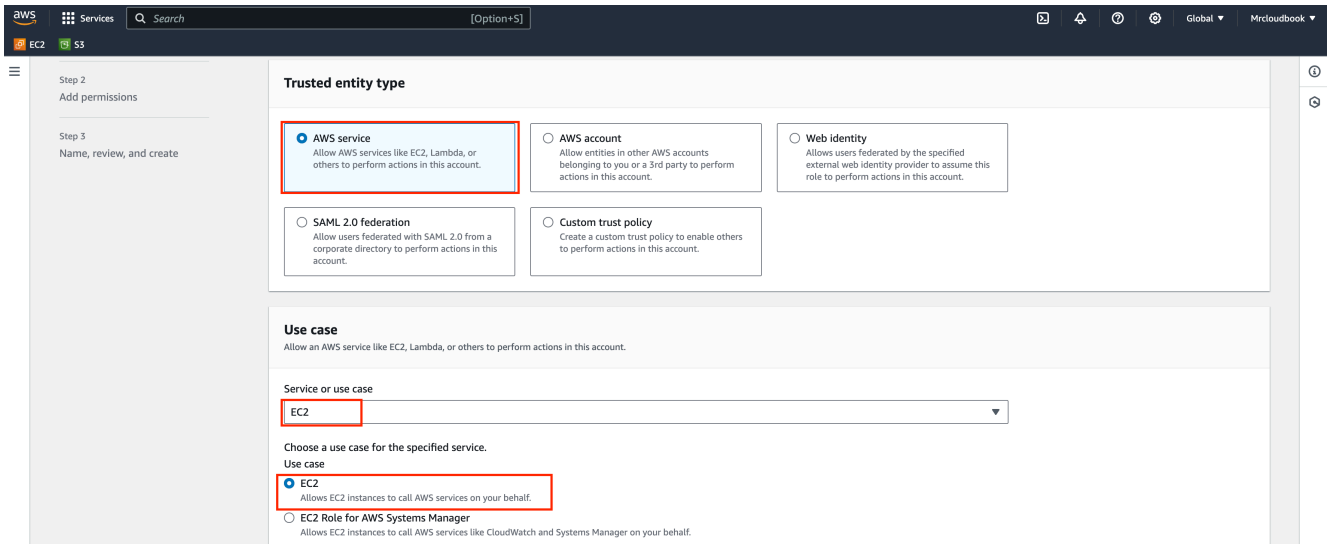


Click on Create Role

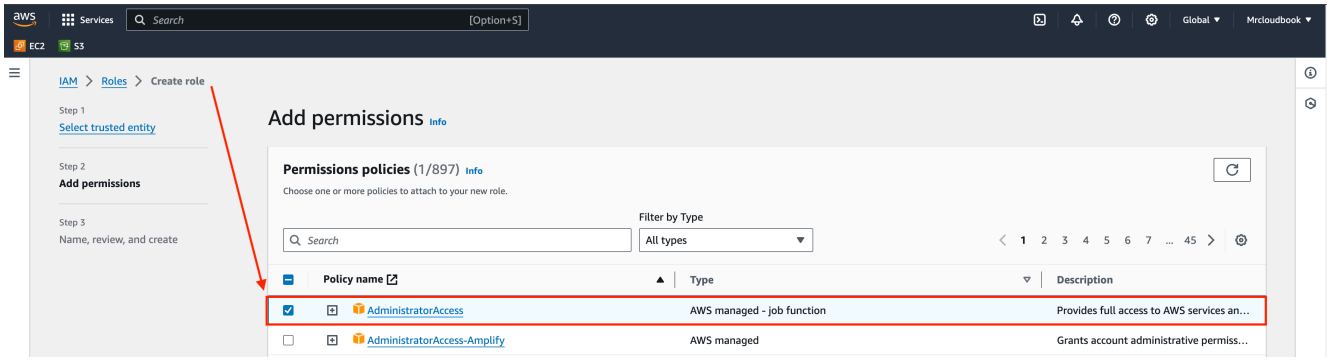


Select entity type as AWS service

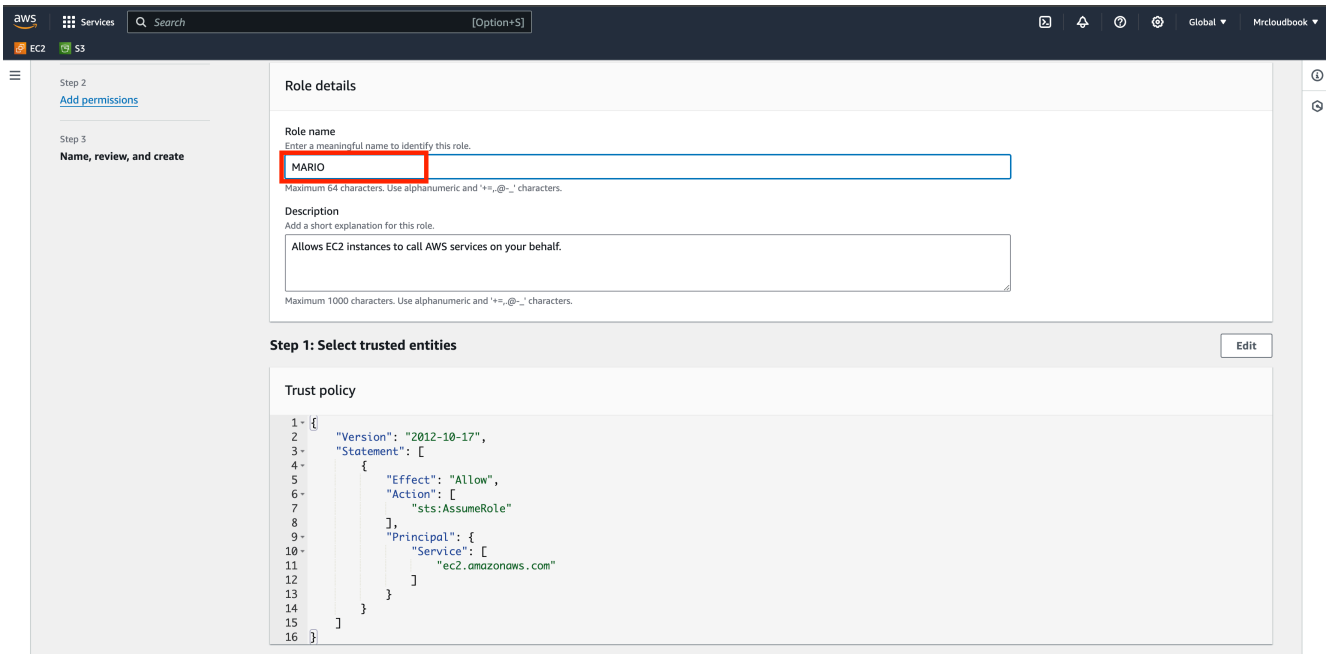
Use case as EC2 and click on Next.



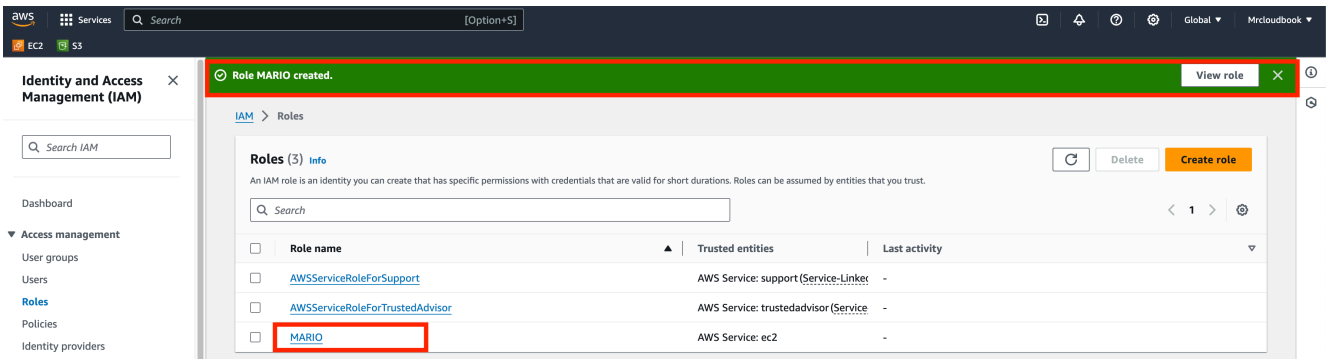
For permission policy select Administrator Access (Just for learning purpose), click Next.



Provide a Name for Role and click on Create role,you can use any name for role.



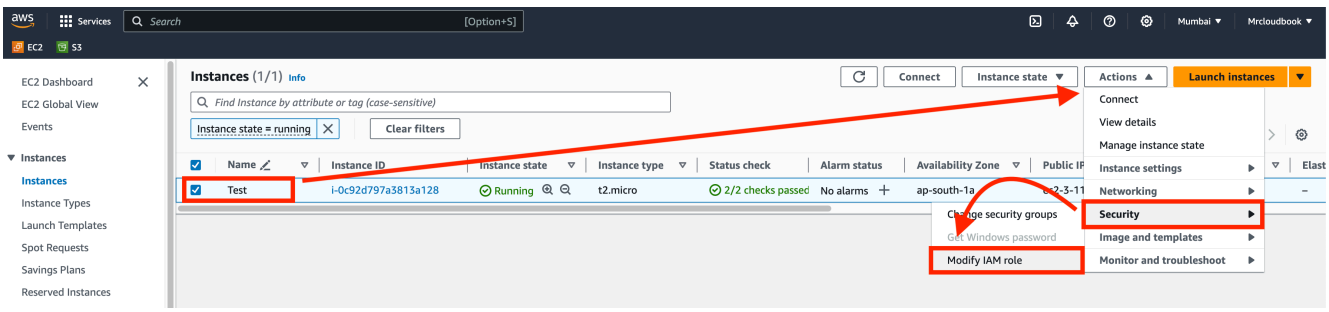
Role is created.



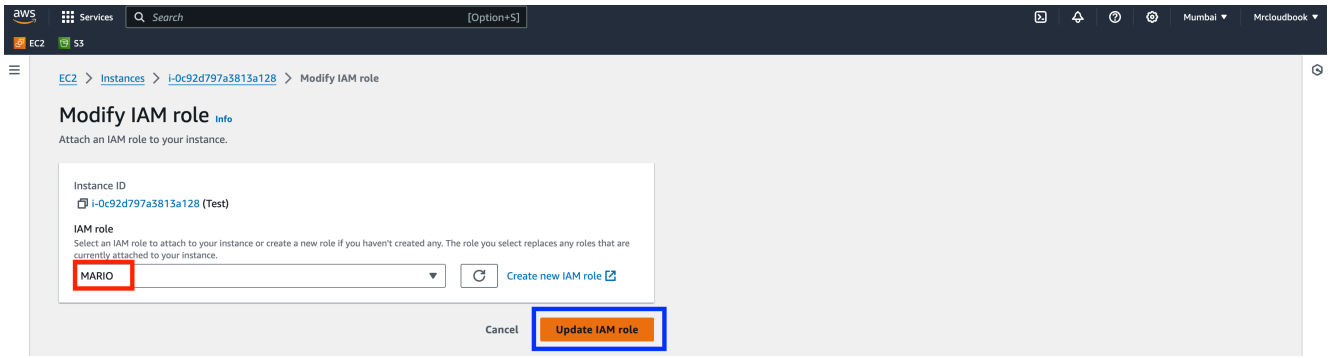
Now Attach this role to Ec2 instance that we created earlier, so we can provision cluster from that instance.

Go to EC2 Dashboard and select the instance.

Click on Actions –> Security –> Modify IAM role.



Select the Role that created earlier and click on Update IAM role.



Connect the instance to Mobaxtreme or Putty

Install Jenkins, Docker and Trivy

To Install Jenkins

Connect to your console, and enter these commands to Install Jenkins

```
vi jenkins.sh
```

```
#!/bin/bash
sudo apt update -y
wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.ad
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 | su
                  /usr/share/keyrings/jenkins-keyring.asc &gt; /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
                  https://pkg.jenkins.io/debian-stable binary/ | sudo te
                              /etc/apt/sources.list.d/jenkins.list &gt;
sudo apt-get update -y
```

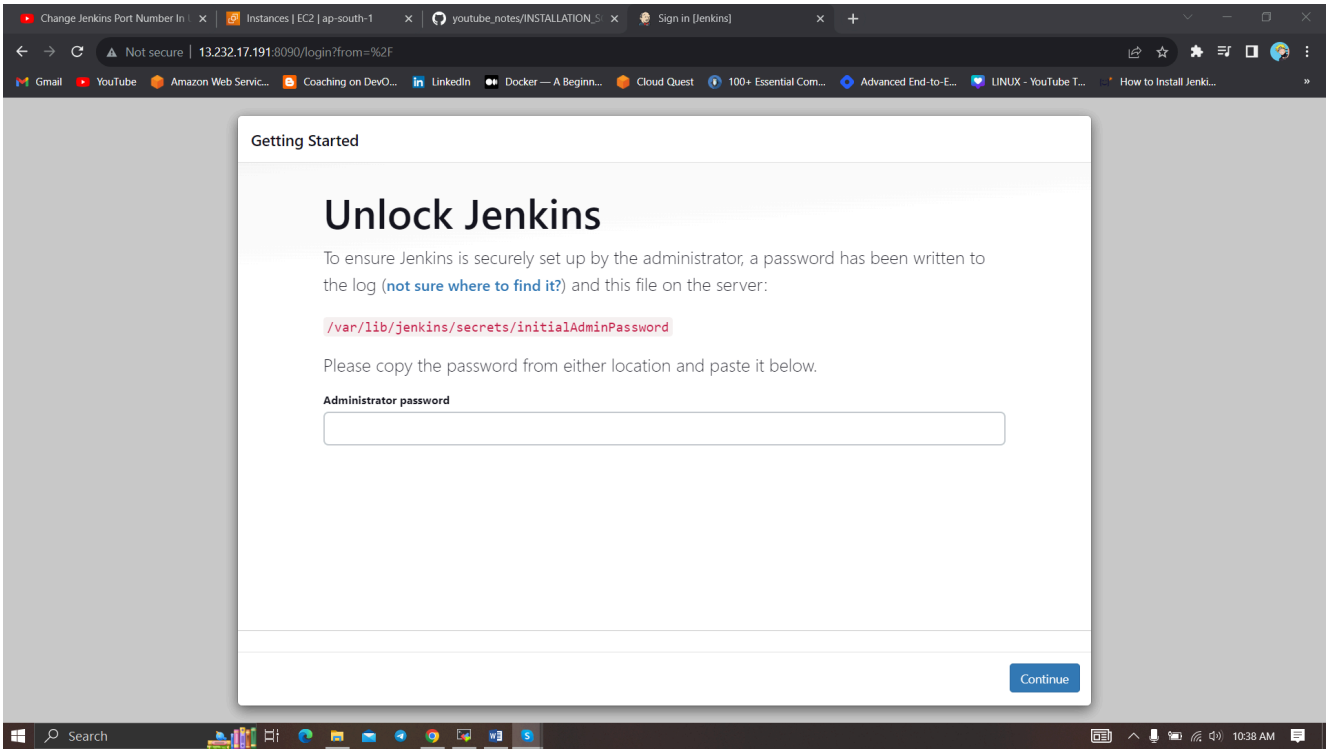
```
sudo apt-get install jenkins -y
sudo systemctl start jenkins
```

```
sudo chmod 777 jenkins.sh
sudo su   #move into root and run
./jenkins.sh    # this will installl jenkins
```

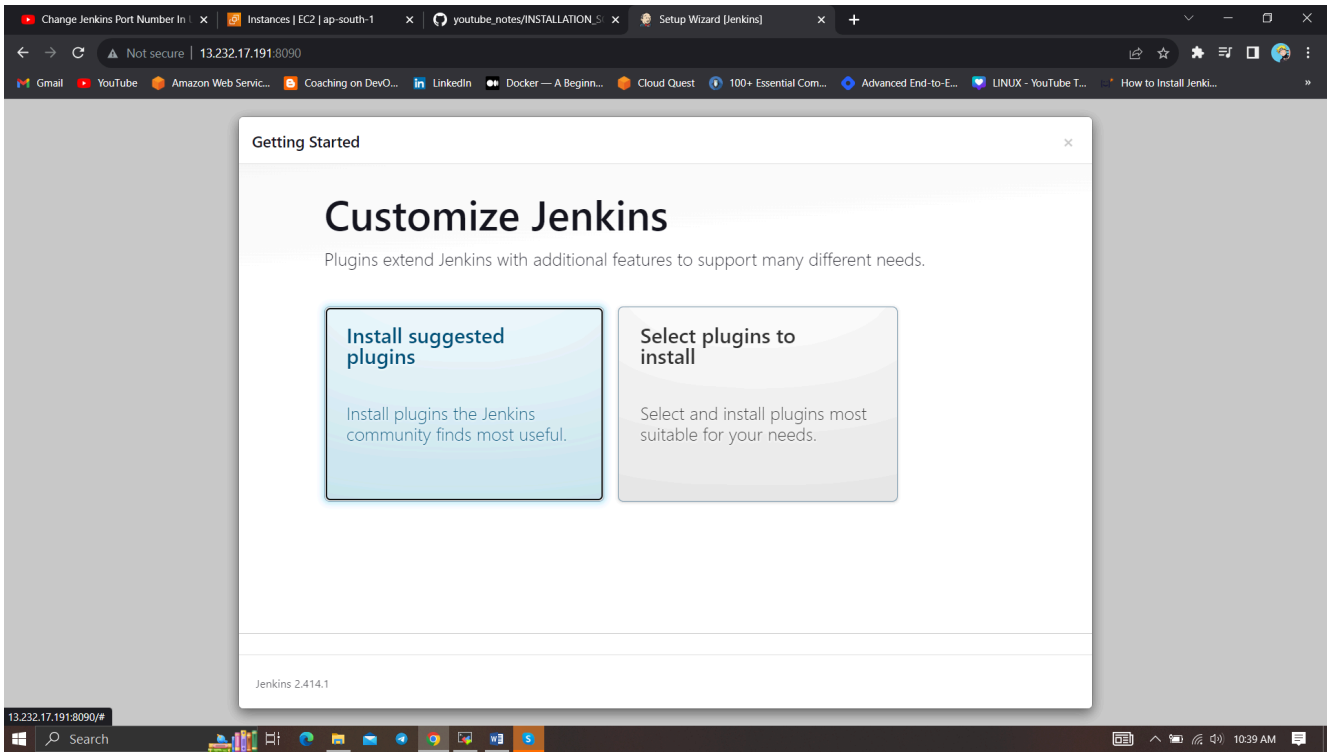
Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

Now [redacted]

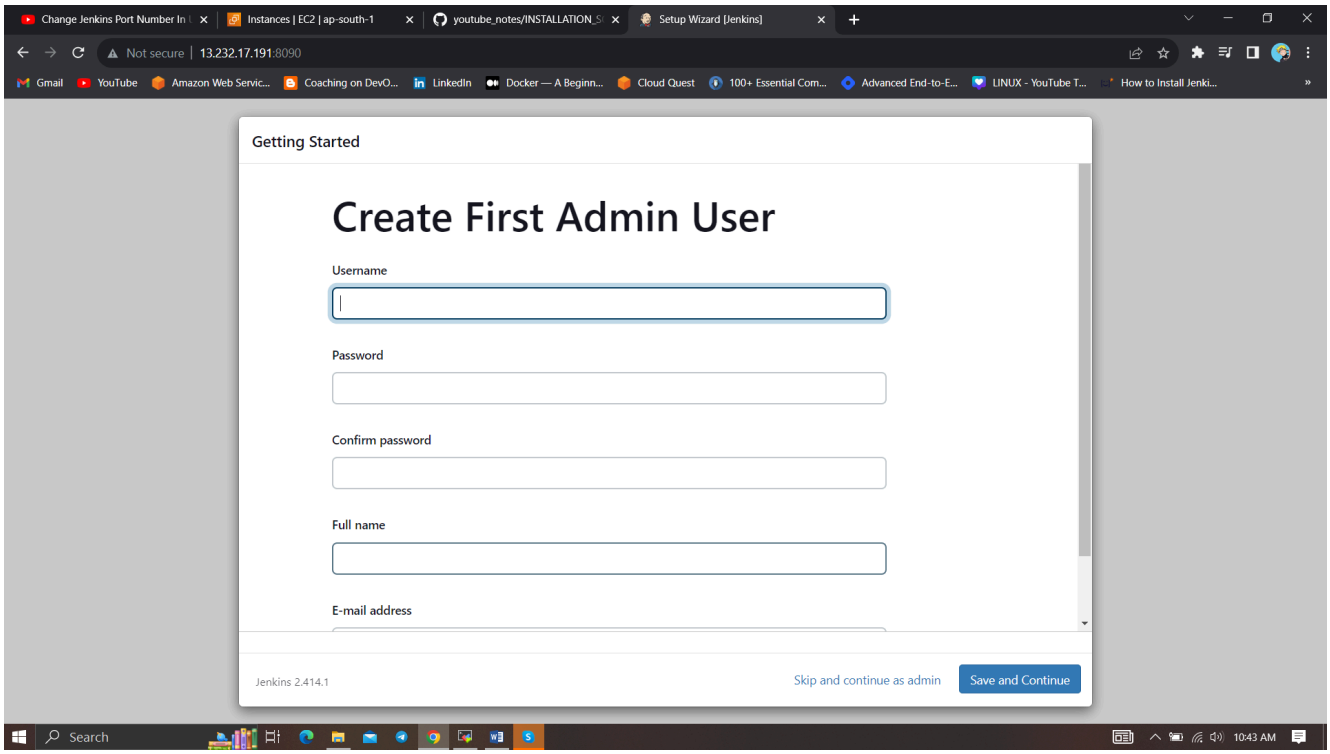
```
EC2 Public IP Address:8080&
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



Unlock Jenkins using an administrative password and install the suggested plugins.

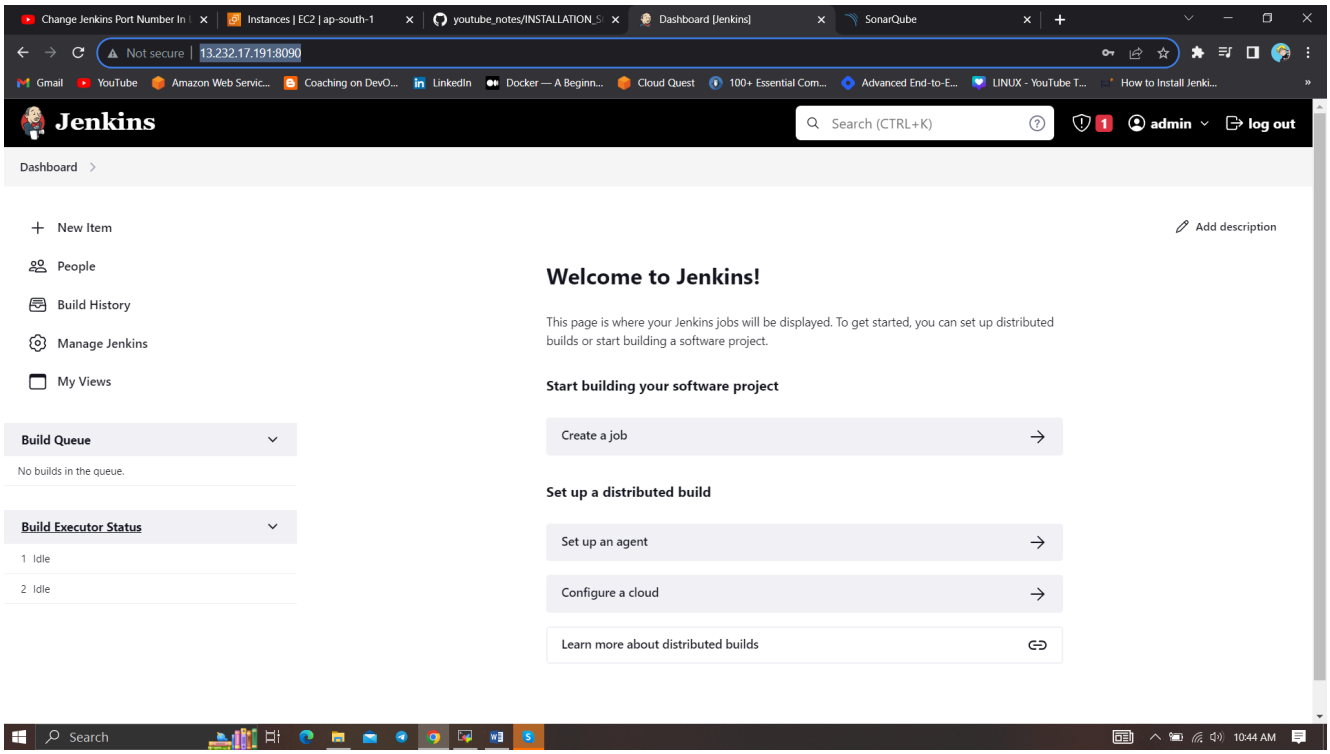


Jenkins will now get installed and install all the libraries.



Create a user click on save and continue.

Jenkins Getting Started Screen.



Install Docker

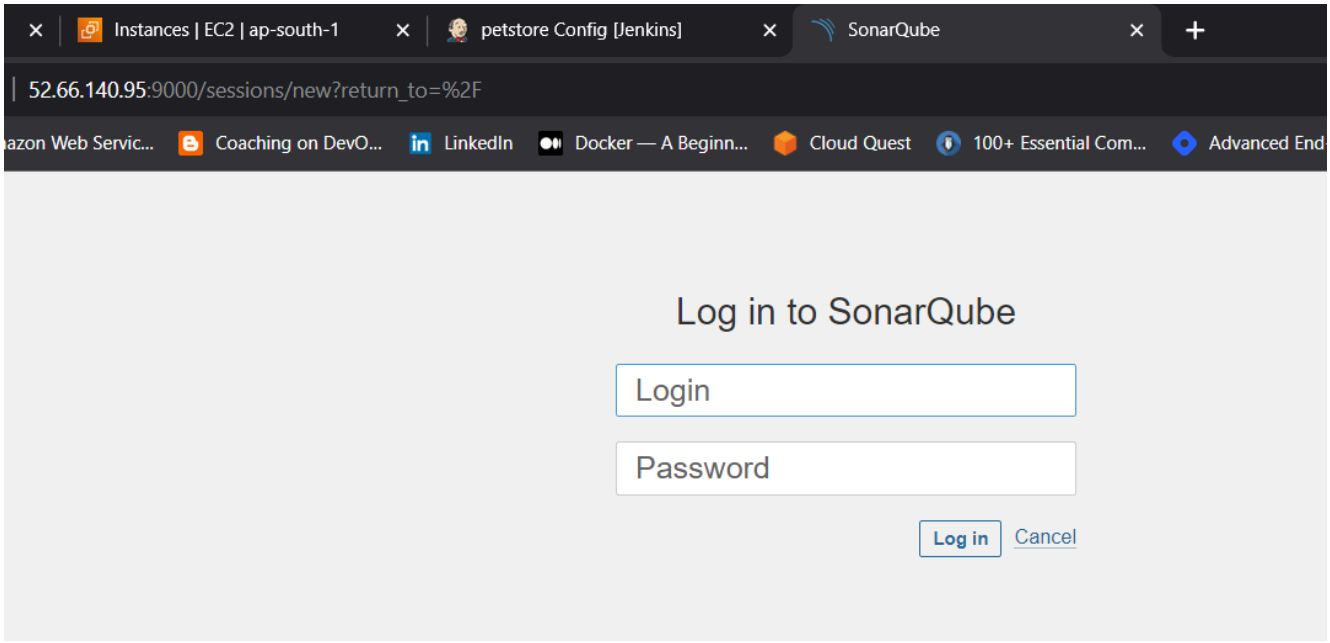
```
sudo apt-get update
sudo apt-get install docker.io -y
sudo usermod -aG docker $USER   #my case is ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
```

After the docker installation, we create a sonarqube container (Remember to add 9000 ports in the security group).

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

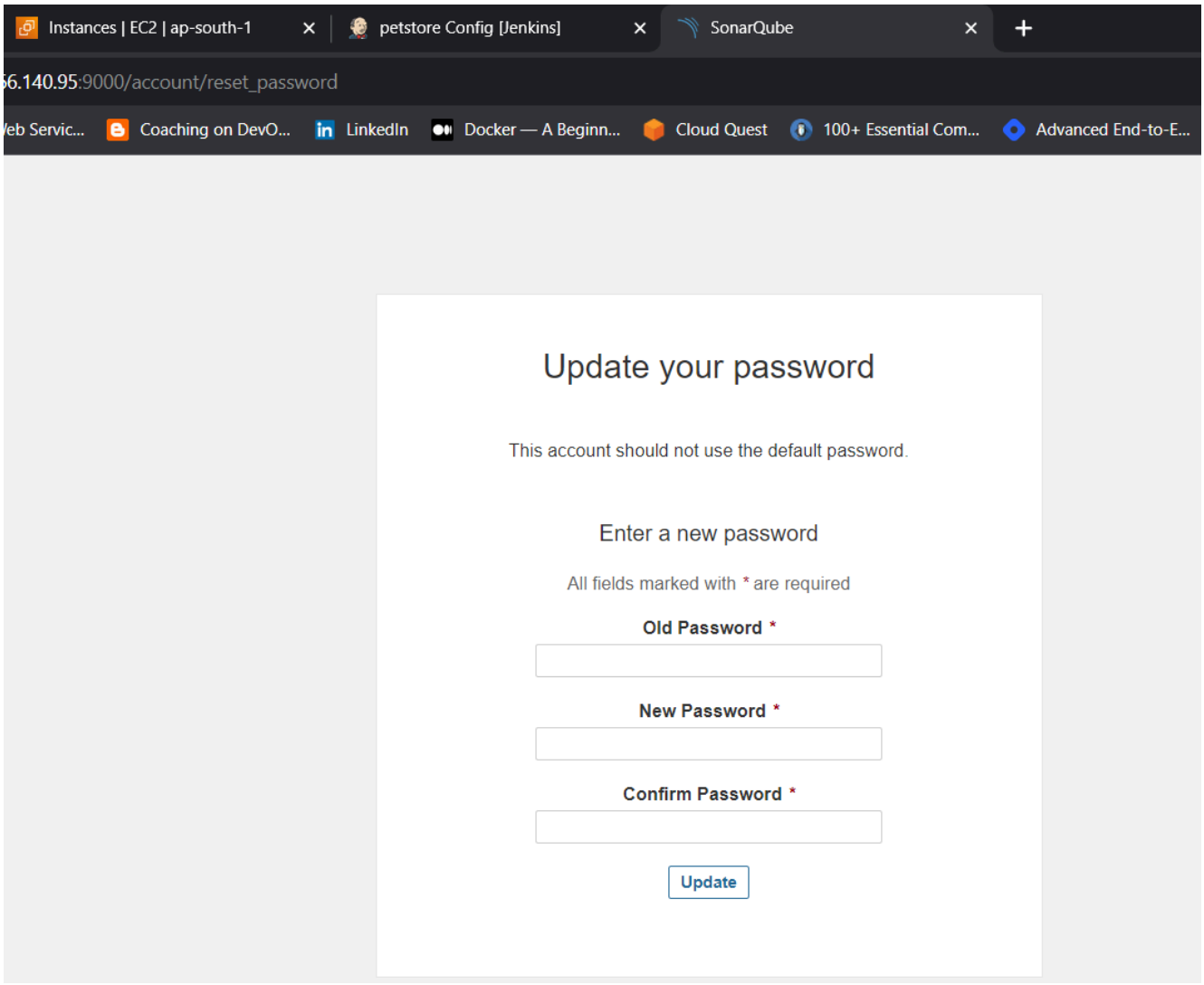
```
ubuntu@ip-172-31-42-253:~$ sudo chmod 777 /var/run/docker.sock
ubuntu@ip-172-31-42-253:~$ docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
44ba2882f8eb: Pull complete
2cabec57fa36: Pull complete
c20481384b6a: Pull complete
bf7b17ee74f8: Pull complete
38617faac714: Pull complete
706f20f58f5e: Pull complete
65a29568c257: Pull complete
Digest: sha256:1a118f8ab960d6c3d4ea8b4455a5a6560654511c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b60c96bf9ad3d62289436af7f752fdb04993092d0ca3065e2f2e32301b50139
ubuntu@ip-172-31-42-253:~$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED         STATUS         PORTS                                       NAMES
4b60c96bf9ad   sonarqube:lts-community   "/opt/sonarqube/dock…"   9 seconds ago   Up 5 seconds   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   sonar
ubuntu@ip-172-31-42-253:~$
```

Now our Sonarqube is up and running

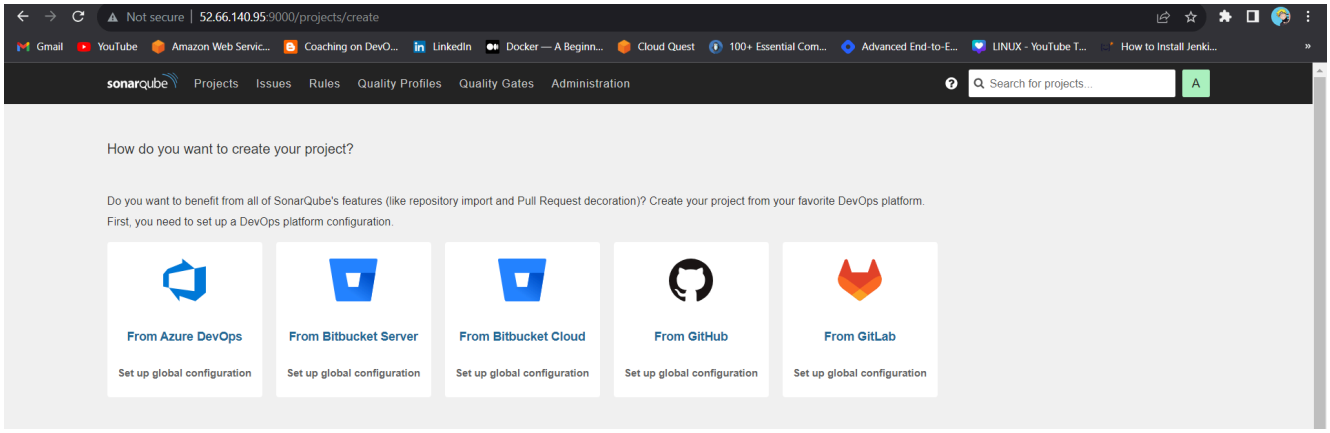


Enter username and password, click on login and change password

```
username admin
password admin
```



Update New password, This is Sonar Dashboard.



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 | gp
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity
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
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
sudo apt update &amp;&amp; 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/releas
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 "awsc
sudo apt-get install unzip -y
unzip awscliv2.zip
sudo ./aws/install
```

Give permissions and run script

```
sudo chmod 777 script.sh
./script.sh
```

Next, we will log in to Jenkins and start to configure our Pipeline in Jenkins

Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

Install Plugin

Goto Manage Jenkins →Plugins → Available Plugins →

Install below plugins

Blue ocean

1 → Eclipse Temurin Installer

2 → SonarQube Scanner

3 → NodeJs Plugin

4 → Docker

5 → Docker commons

6 → Docker pipeline

7 → Docker API

8 → Docker Build step

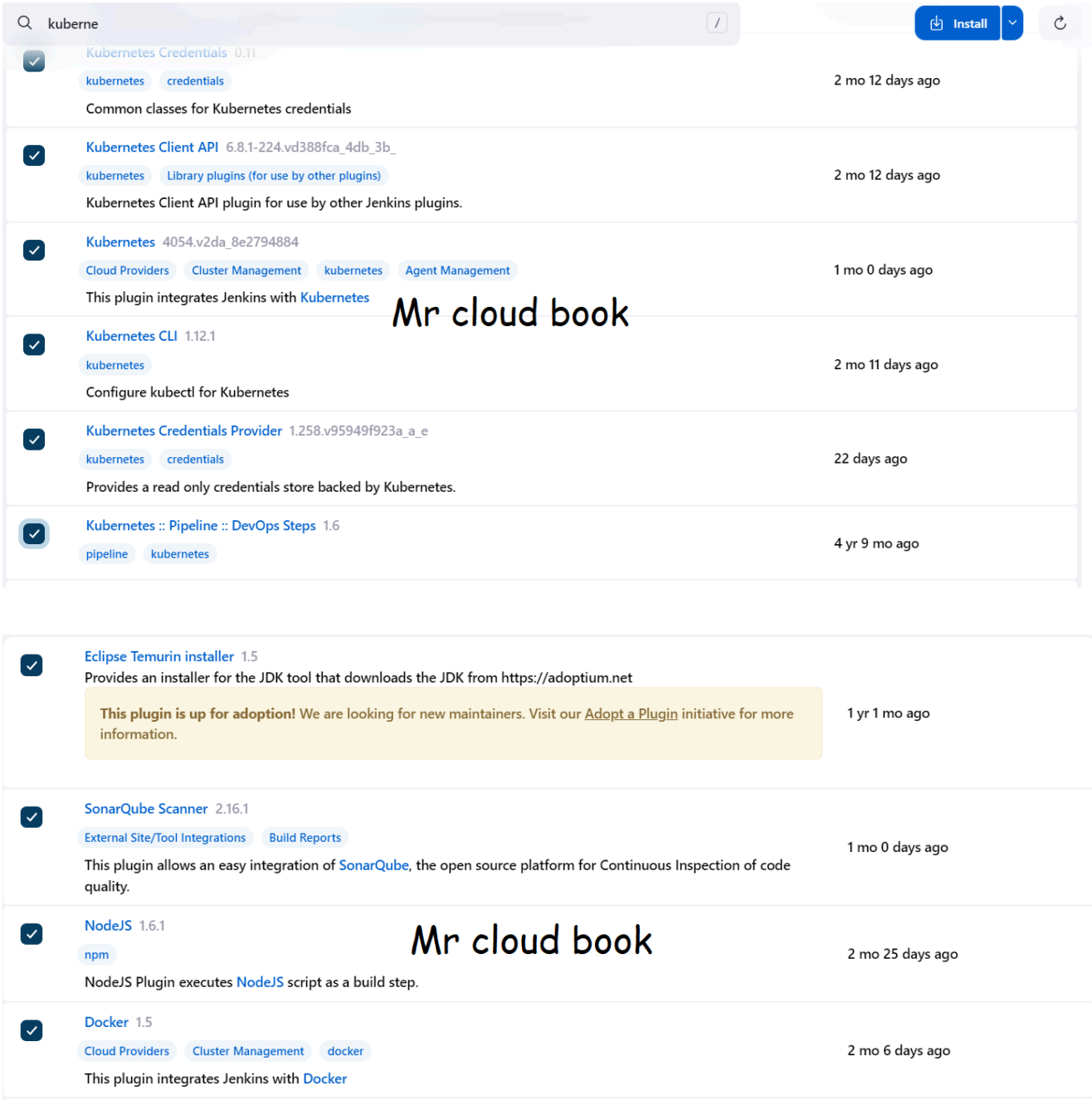
9 → Owasp Dependency Check

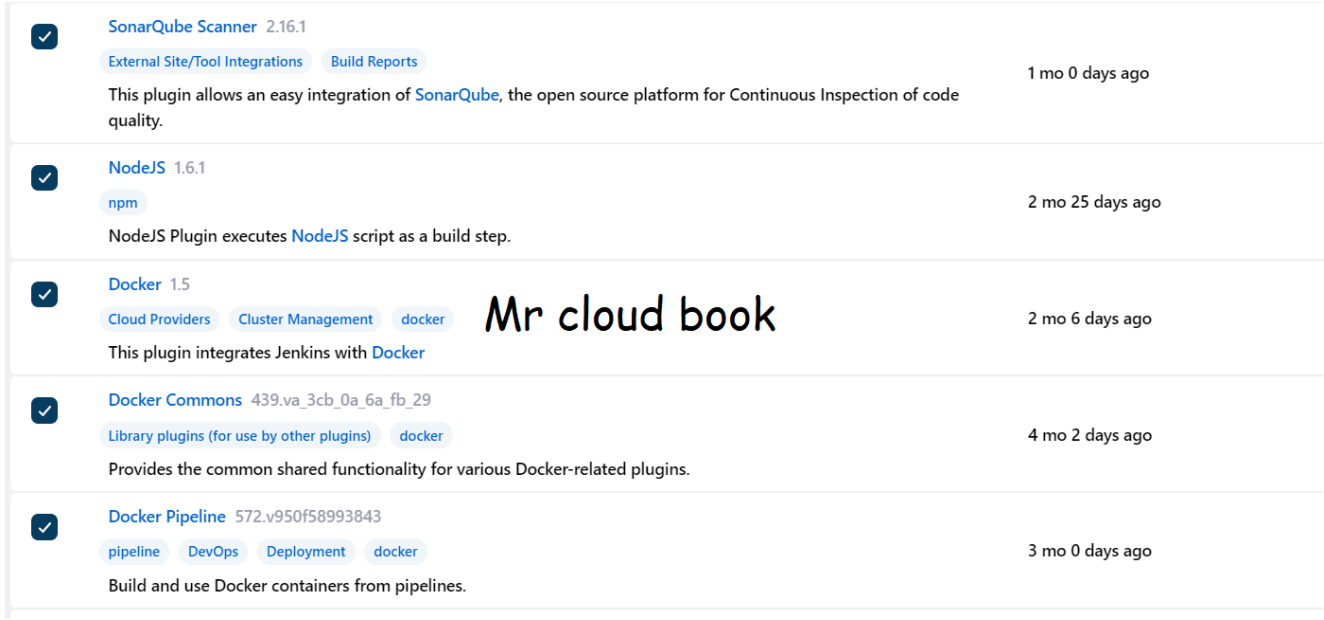
10 → Kubernetes

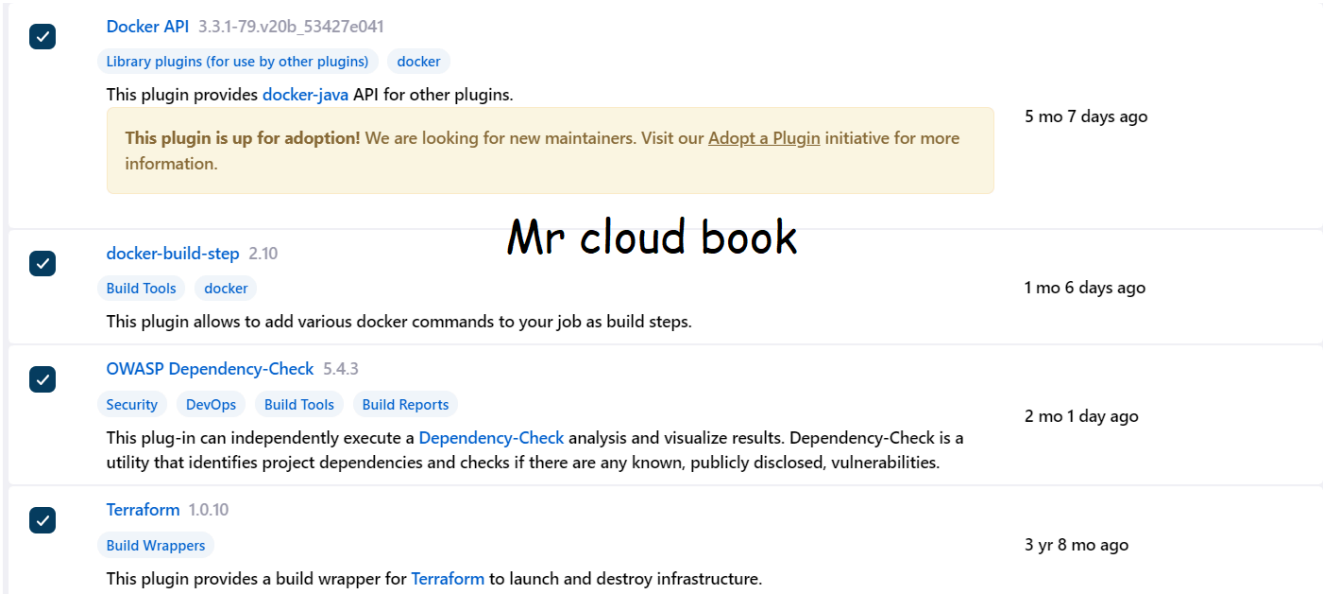
11 → Kubernetes CLI

12 → Kubernetes Client API

13 → Kubernetes Pipeline DevOps steps

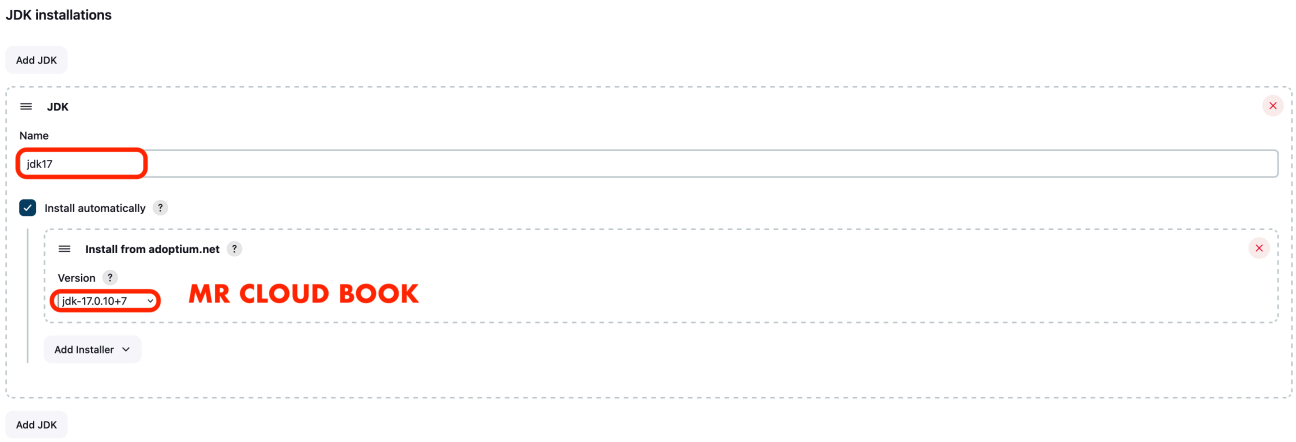


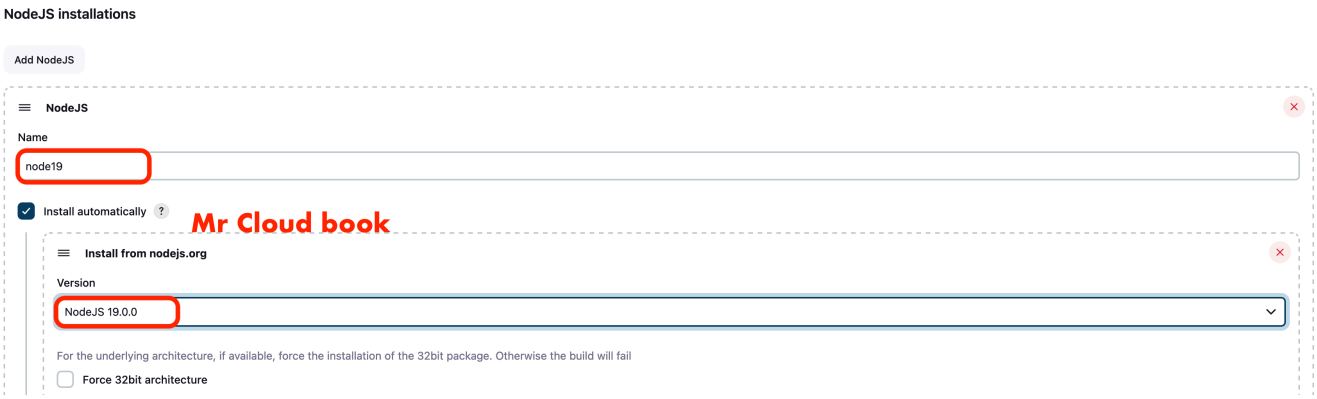




Configure Java and Nodejs in Global Tool Configuratio

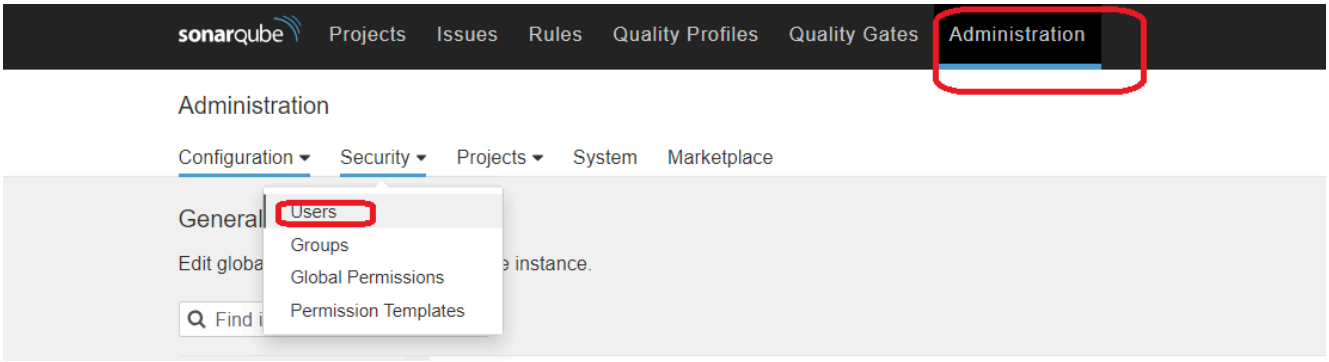
Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(19)→ Click on Apply and Save



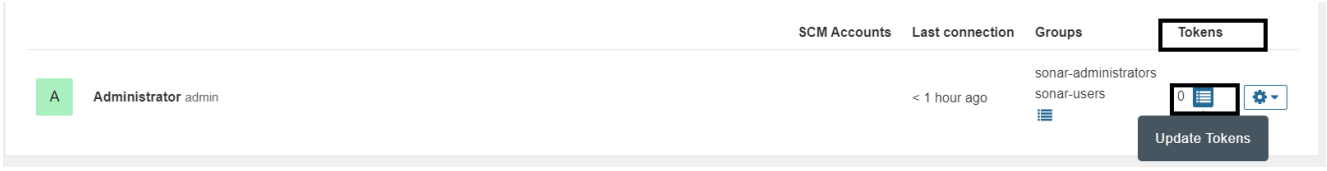


Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server.

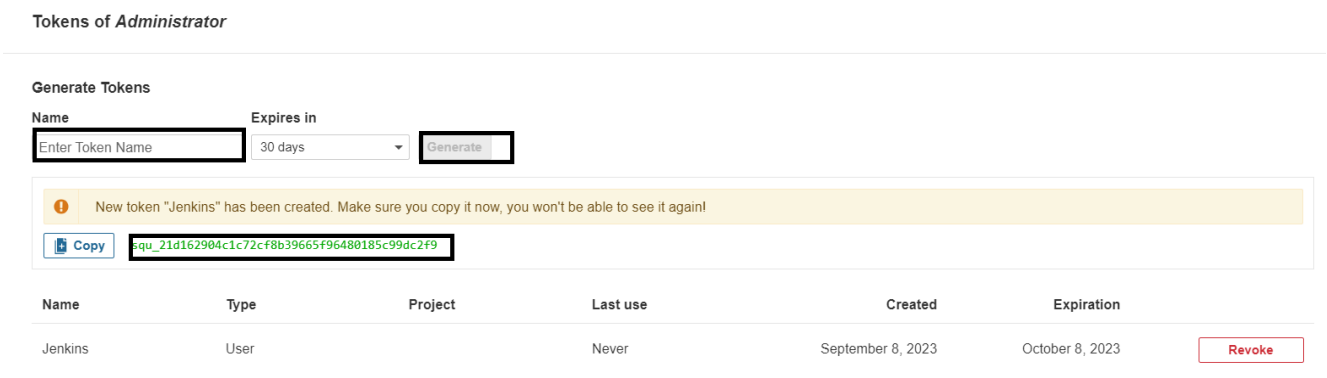
Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token



click on update Token

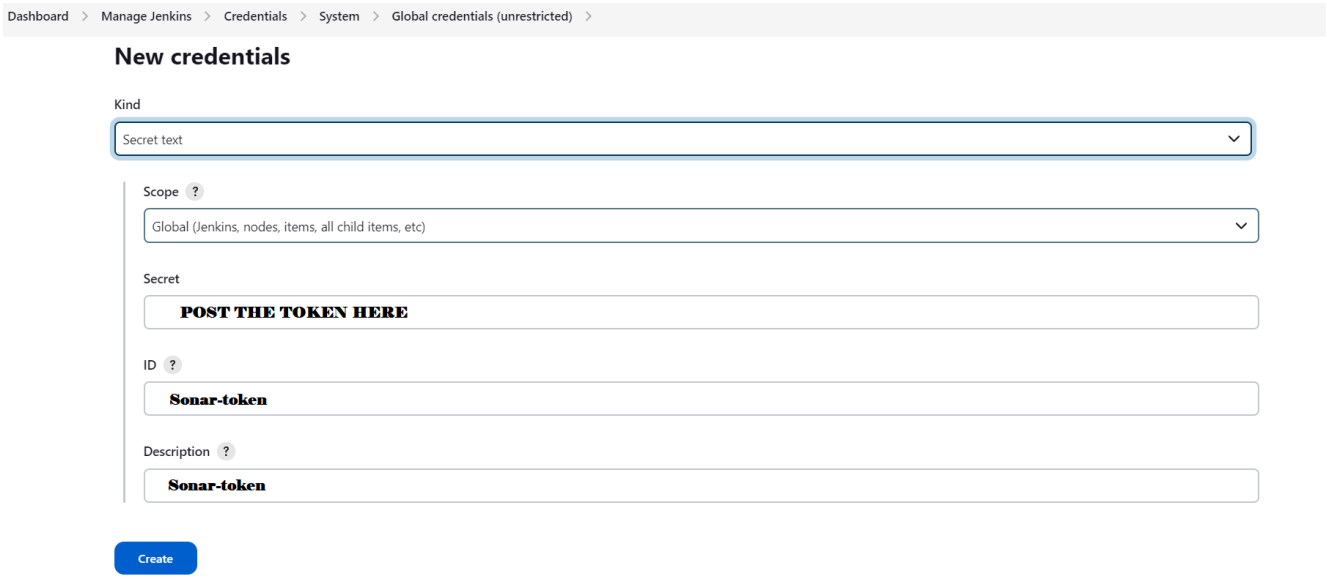


Create a token with a name and generate

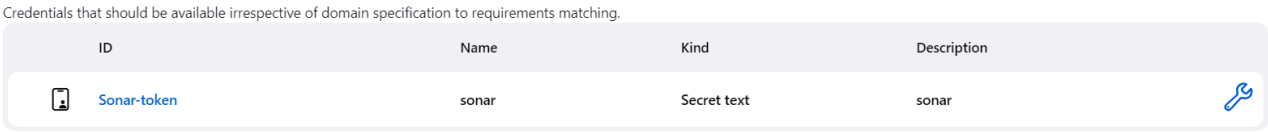


copy Token

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



You will this page once you click on create



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

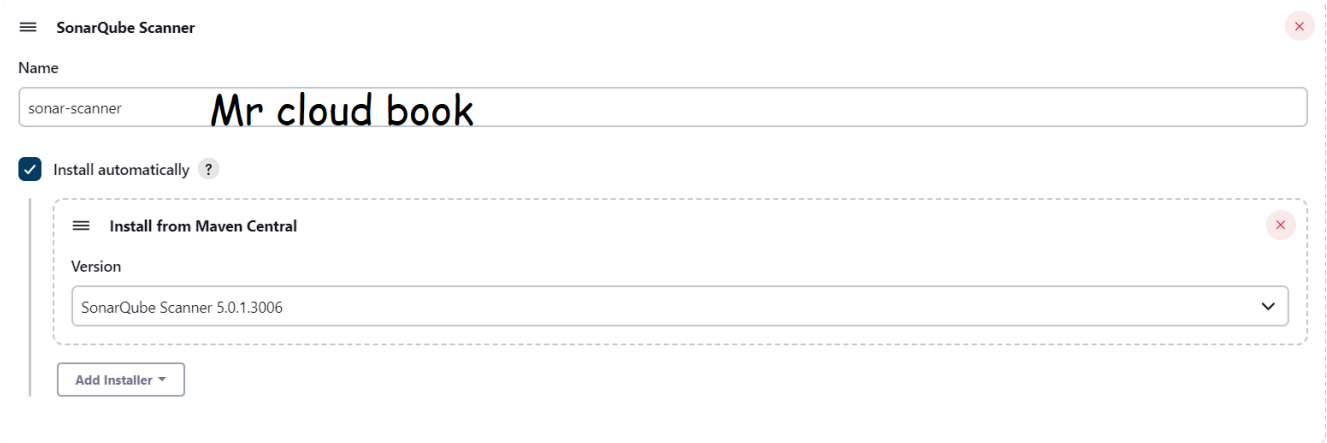
Click on Apply and Save

The Configure System option is used in Jenkins to configure different server

Global Tool Configuration is used to configure different tools that we install using Plugins

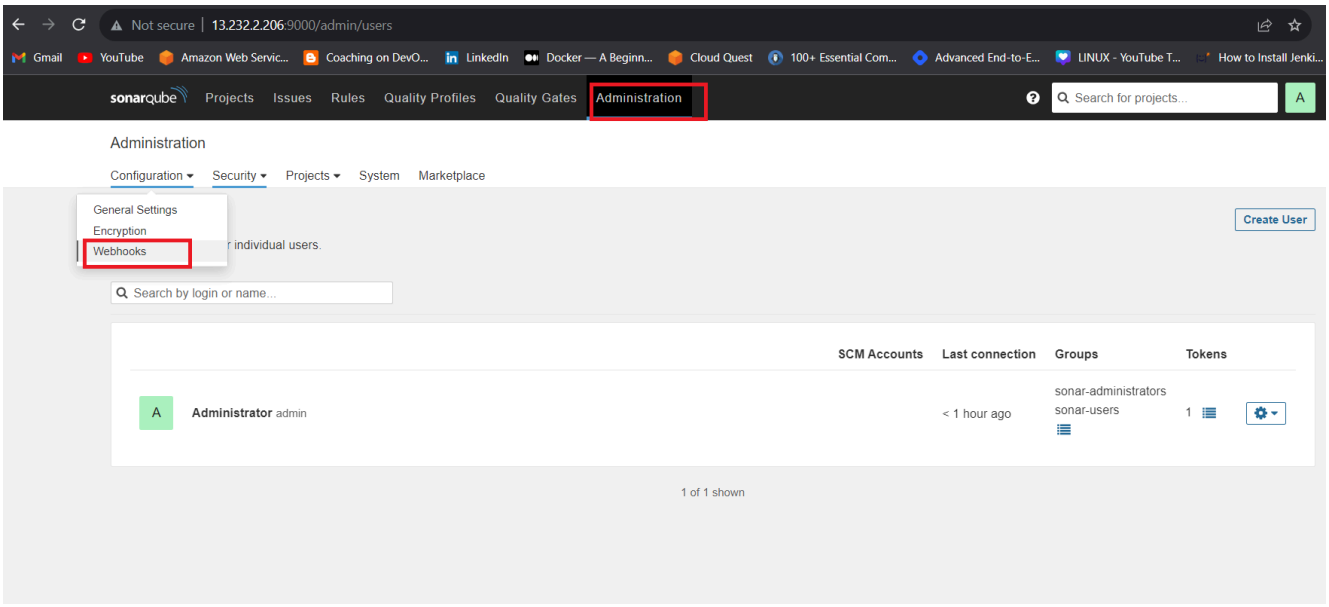
We will install a sonar scanner in the tools.

Manage Jenkins –> Tools –> SonarQube Scanner

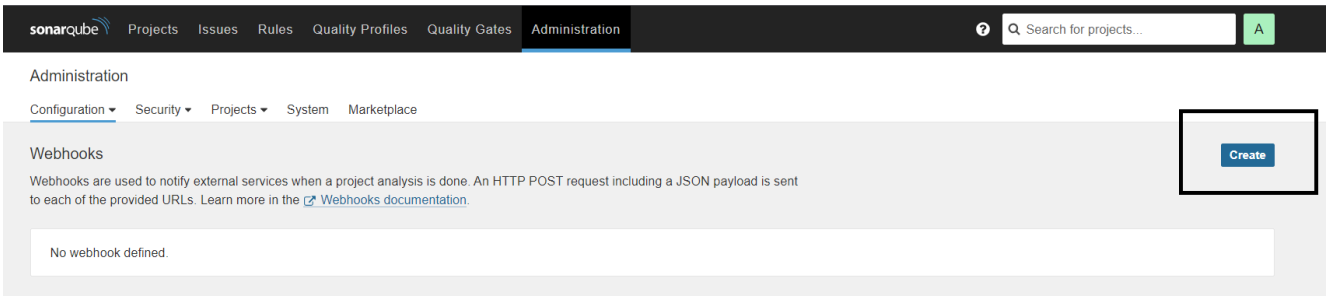


In the Sonarqube Dashboard add a quality gate also

Administration–> Configuration–>Webhooks

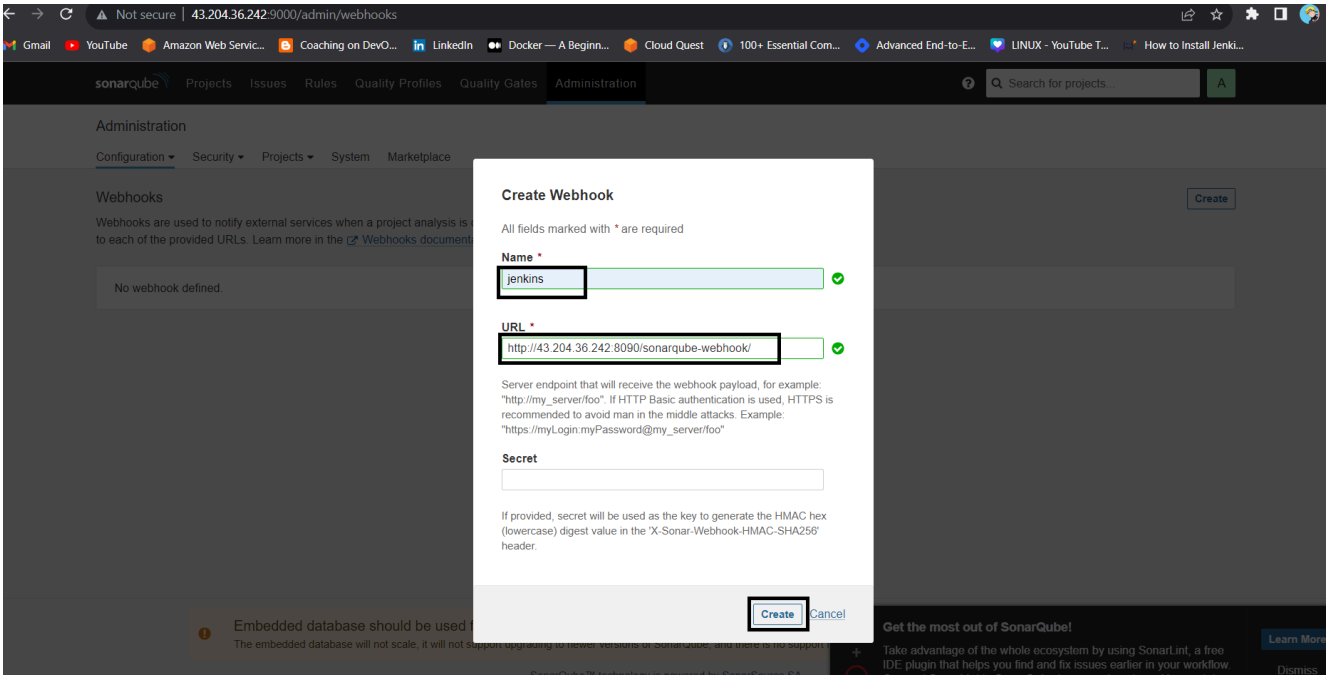


Click on Create



Add details

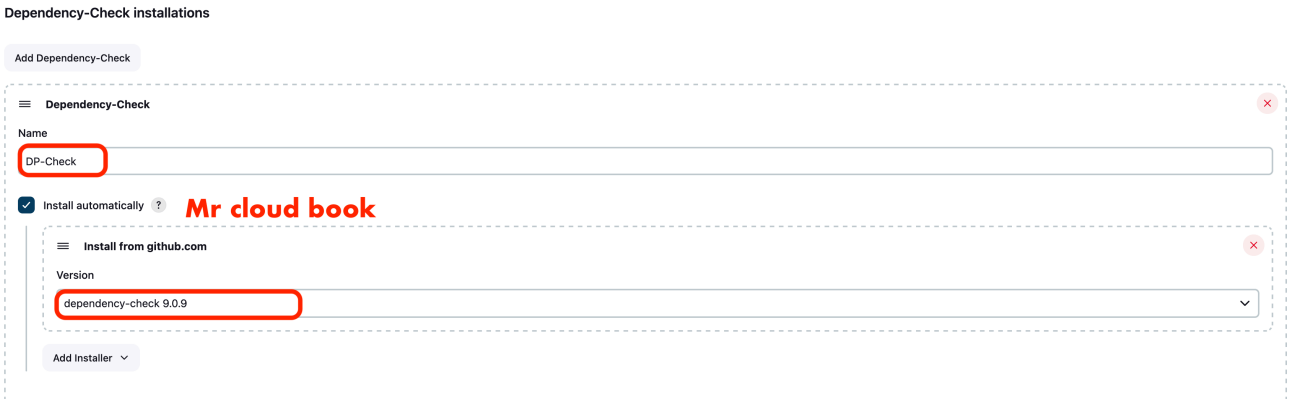
```
#in url section of quality gate
http://jenkins-public-ip:8080/sonarqube-webhook/
```



To see the report, you can go to Sonarqube Server and go to Projects.

First, we configured the Plugin and next, we had to configure the Tool

Goto Dashboard → Manage Jenkins → Tools →



Click on Apply and Save here.

Now, goto Dashboard → Manage Jenkins → Tools →

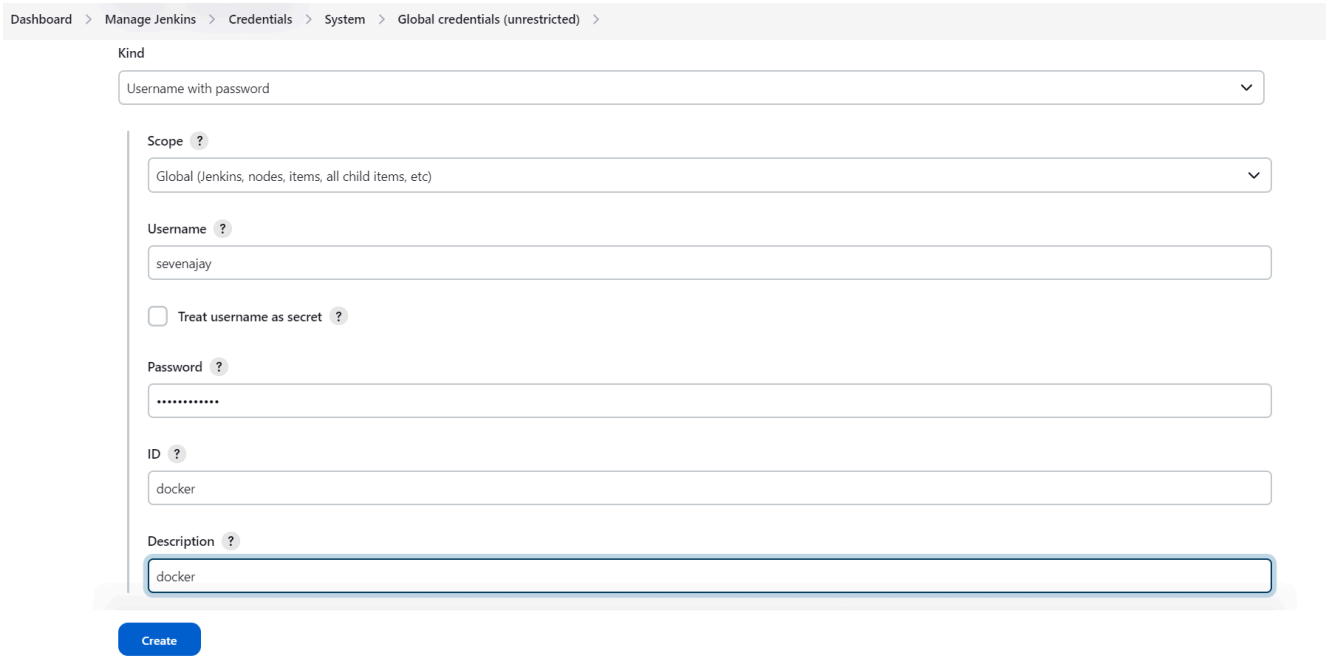
Tools –> Terraform add this

In Jenkins

```
which terraform
```

Go to manage Jenkins –> Credentials

Add DockerHub Username and Password under Global Credentials



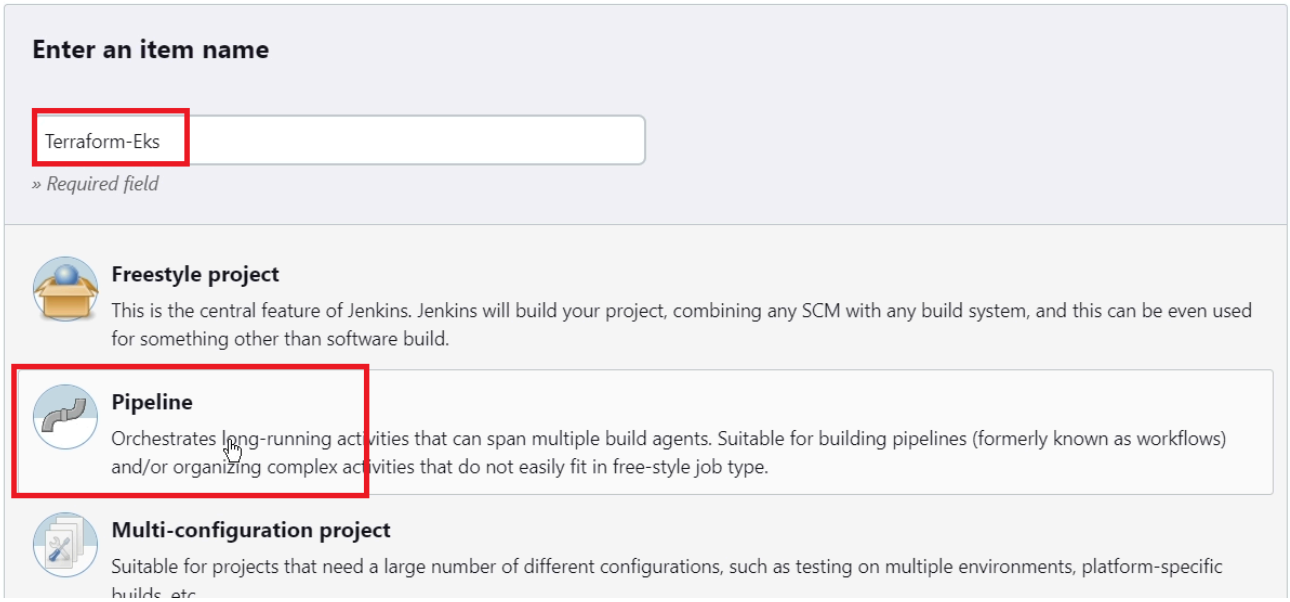
Create EKS Cluster from Jenkins

Go to Legacy Branch in Git Repository

There you can find folder called Eks-terrafrom there you can find backend file

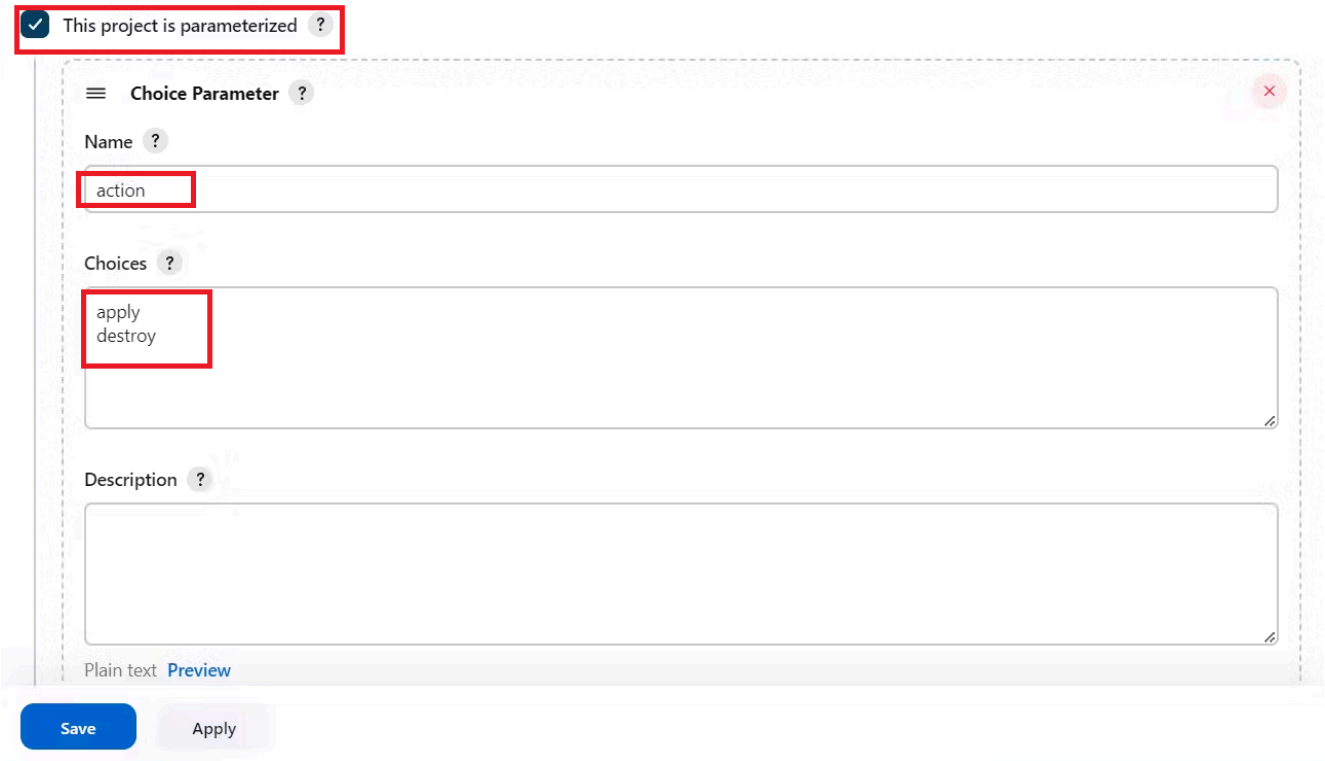
CHANGE YOUR S3 BUCKET NAME IN THE BACKEND.TF

Now create a new job for the Eks provision



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

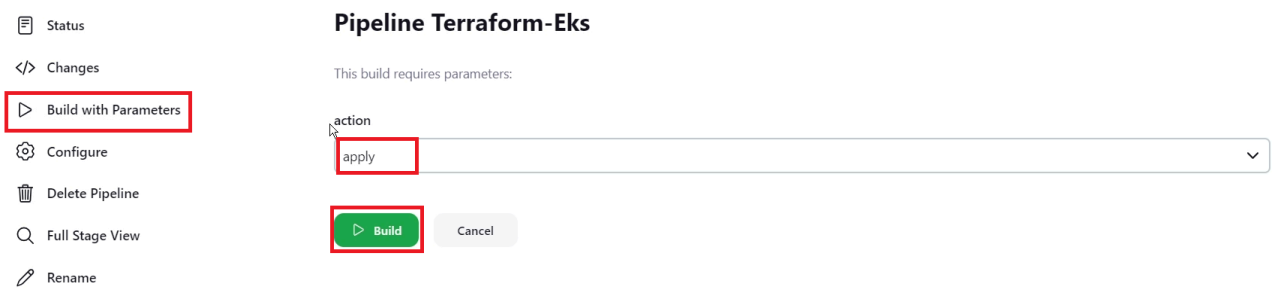


Let's add a pipeline

```
pipeline{
    agent any
    stages {
```

```
        stage('Checkout from Git'){
            steps{
                git branch: 'legacy', url: 'https://github.com/Aj7Ay/cha
            }
        }
        stage('Terraform version'){
             steps{
                 sh 'terraform --version'
             }
        }
        stage('Terraform init'){
             steps{
                 dir('Eks-terraform') {
                      sh 'terraform init'
                   }
             }
        }
        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

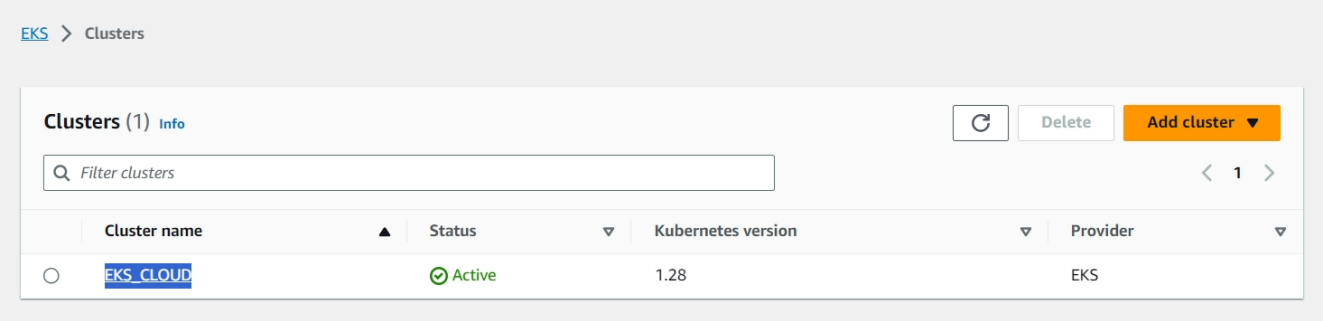


Stage view it will take max 10mins to provision

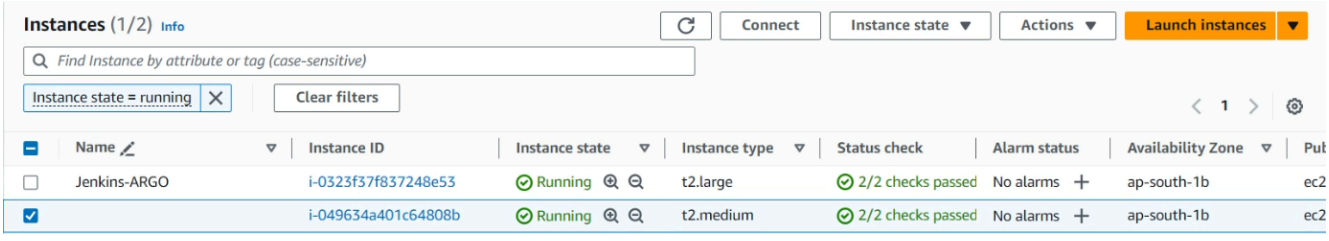
Blue ocean output

Mr Cloud book

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



Ec2 instance is created for the Node group



Create Job for chatbot clone

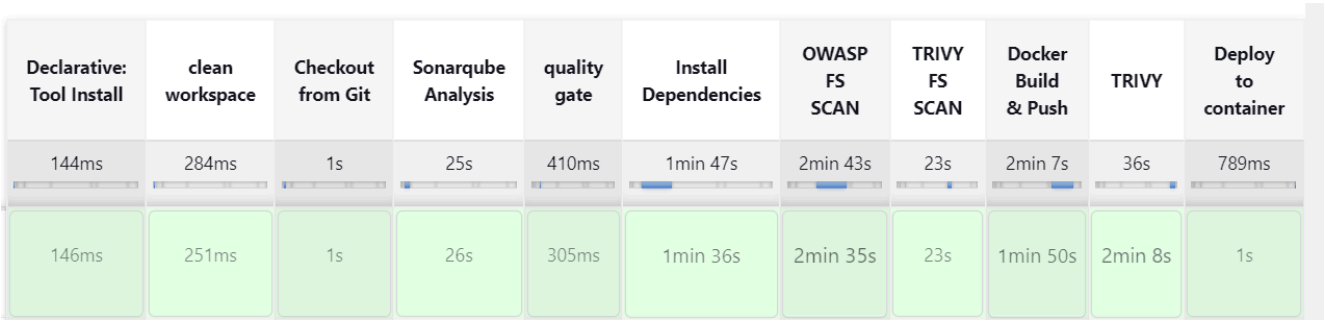
Add this stage to Pipeline Script

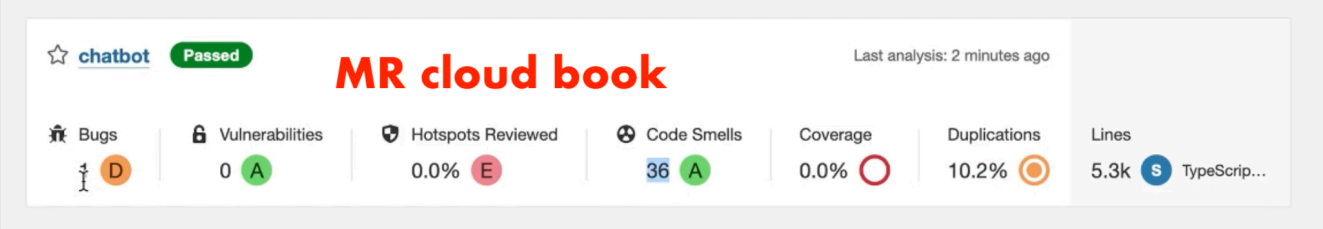
```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node19'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('Checkout from Git'){
            steps{
                git branch: 'legacy', url: 'https://github.com/Aj7Ay/cha
            }
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
        stage("Sonarqube Analysis "){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.proje
                    -Dsonar.projectKey=Chatbot '''
                }
            }
        }
        stage("quality gate"){
           steps {
                script {
                    waitForQualityGate abortPipeline: false, credentials
                }
            }
        }
        stage('OWASP FS SCAN') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --disabl
                dependencyCheckPublisher pattern: '**/dependency-check-r
            }
        }
        stage('TRIVY FS SCAN') {
            steps {
                sh "trivy fs . &gt; trivyfs.json"
```

```
            }
        }
        stage("Docker Build &amp;amp; Push"){
            steps{
                script{
                   withDockerRegistry(credentialsId: 'docker', toolName
                       sh "docker build -t chatbot ."
                       sh "docker tag chatbot sevenajay/chatbot:latest '
                       sh "docker push sevenajay/chatbot:latest "
                    }
                }
            }
        }
        stage("TRIVY"){
            steps{
                sh "trivy image sevenajay/chatbot:latest &amp;gt; trivy.jso
            }
        }
        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 sevenajay/
            }
        }
    }
```

Apply and Save and click on Build

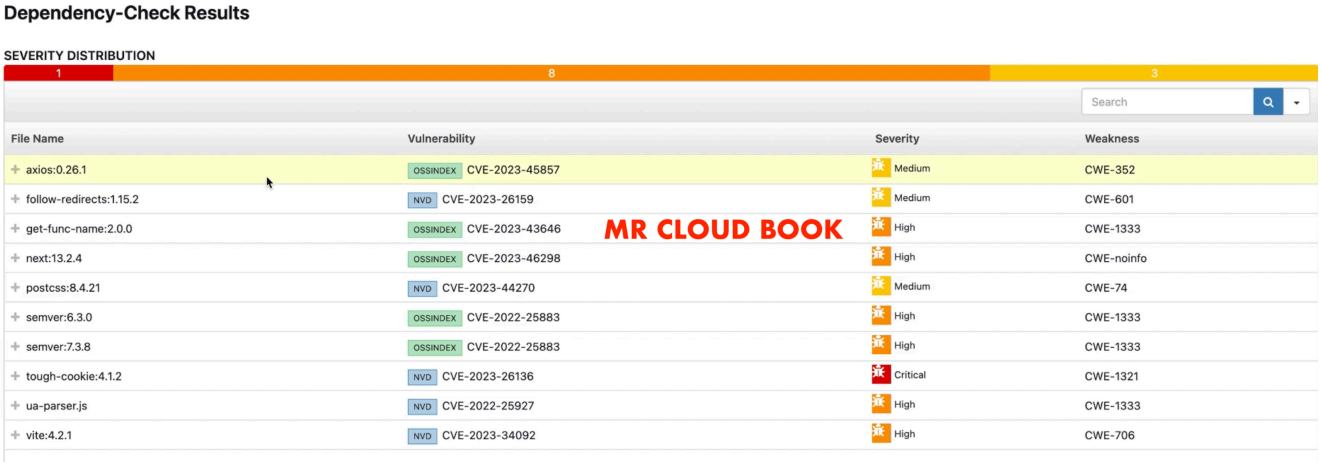
stage view





You can see the report has been generated and the status shows as passed. You can see that there are 5.3k lines it scanned. To see a detailed report, you can go to issues.

You will see that in status, a graph will also be generated and Vulnerabilities.



Trivy Container scan report

Save Copy

SyntaxError: JSON.parse: unexpected character at line 2 column 1 of the JSON data

```
sevenajay/chatbot:latest (alpine 3.18.0)
========================================
Total: 21 (UNKNOWN: 0, LOW: 0, MEDIUM: 16, HIGH: 2, CRITICAL: 3)
```

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
busybox	CVE-2022-48174	CRITICAL	fixed	1.36.0-r9	1.36.1-r1	stack overflow vulnerability in ash.c leads to arbitrary code execution https://avd.aquasec.com/nvd/cve-2022-48174
busybox-binsh						
libcrypto3	CVE-2023-5363	HIGH		3.1.0-r4	3.1.4-r0	openssl: Incorrect cipher key and IV length processing https://avd.aquasec.com/nvd/cve-2023-5363
	CVE-2023-2650	MEDIUM			3.1.1-r0	openssl: Possible DoS translating ASN.1 object identifiers https://avd.aquasec.com/nvd/cve-2023-2650
	CVE-2023-2975				3.1.1-r2	openssl: AES-SIV cipher implementation contains a bug that causes it to ignore... https://avd.aquasec.com/nvd/cve-2023-2975
	CVE-2023-3446				3.1.1-r3	openssl: Excessive time spent checking DH keys and parameters https://avd.aquasec.com/nvd/cve-2023-3446
	CVE-2023-3817				3.1.2-r0	OpenSSL: Excessive time spent checking DH q parameter value https://avd.aquasec.com/nvd/cve-2023-3817
	CVE-2023-5678				3.1.4-r1	openssl: Generating excessively long X9.42 DH keys or checking excessively long X9.42... https://avd.aquasec.com/nvd/cve-2023-5678
	CVE-2023-6129				3.1.4-r3	openssl: POLY1305 MAC implementation corrupts vector registers on PowerPC https://avd.aquasec.com/nvd/cve-2023-6129
	CVE-2023-6237				3.1.4-r4	openssl: Excessive time spent checking invalid RSA public keys https://avd.aquasec.com/nvd/cve-2023-6237
	CVE-2024-0727				3.1.4-r5	openssl: denial of service via null dereference https://avd.aquasec.com/nvd/cve-2024-0727
libssl3	CVE-2023-5363	HIGH			3.1.4-r0	openssl: Incorrect cipher key and IV length processing https://avd.aquasec.com/nvd/cve-2023-5363

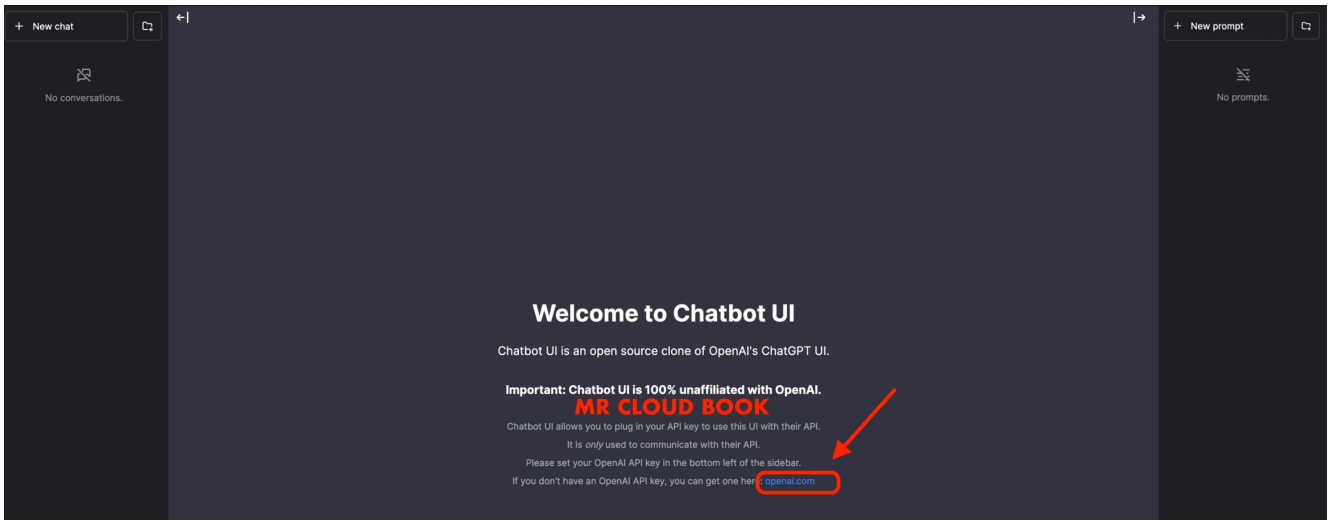
Trivy File scan report

```
package-lock.json (npm)
=======================
Total: 6 (UNKNOWN: 0, LOW: 1, MEDIUM: 5, 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 https://avd.aquasec.com/nvd/cve-2023-45857
follow-redirects	CVE-2023-26159			1.15.2	1.15.4	follow-redirects: Improper Input Validation due to the improper handling of URLs by... https://avd.aquasec.com/nvd/cve-2023-26159
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 https://avd.aquasec.com/nvd/cve-2023-46298
postcss	CVE-2023-44270	MEDIUM		8.4.14	8.4.31	An issue was discovered in PostCSS before 8.4.31. The vulnerability af https://avd.aquasec.com/nvd/cve-2023-44270
tough-cookie	CVE-2023-26136			4.1.2	4.1.3	tough-cookie: prototype pollution in cookie memstore https://avd.aquasec.com/nvd/cve-2023-26136
word-wrap	CVE-2023-26115			1.2.3	1.2.4	word-wrap: ReDoS https://avd.aquasec.com/nvd/cve-2023-26115

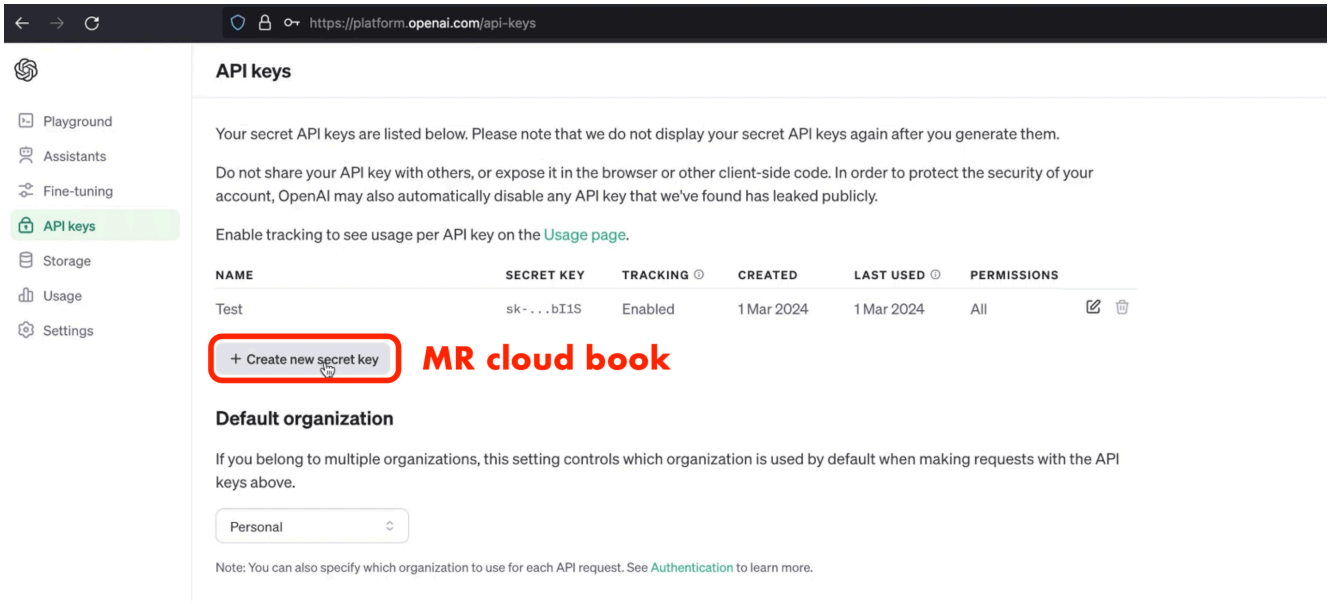
`<Jenkins-public-ip:3000>`

You will get this output

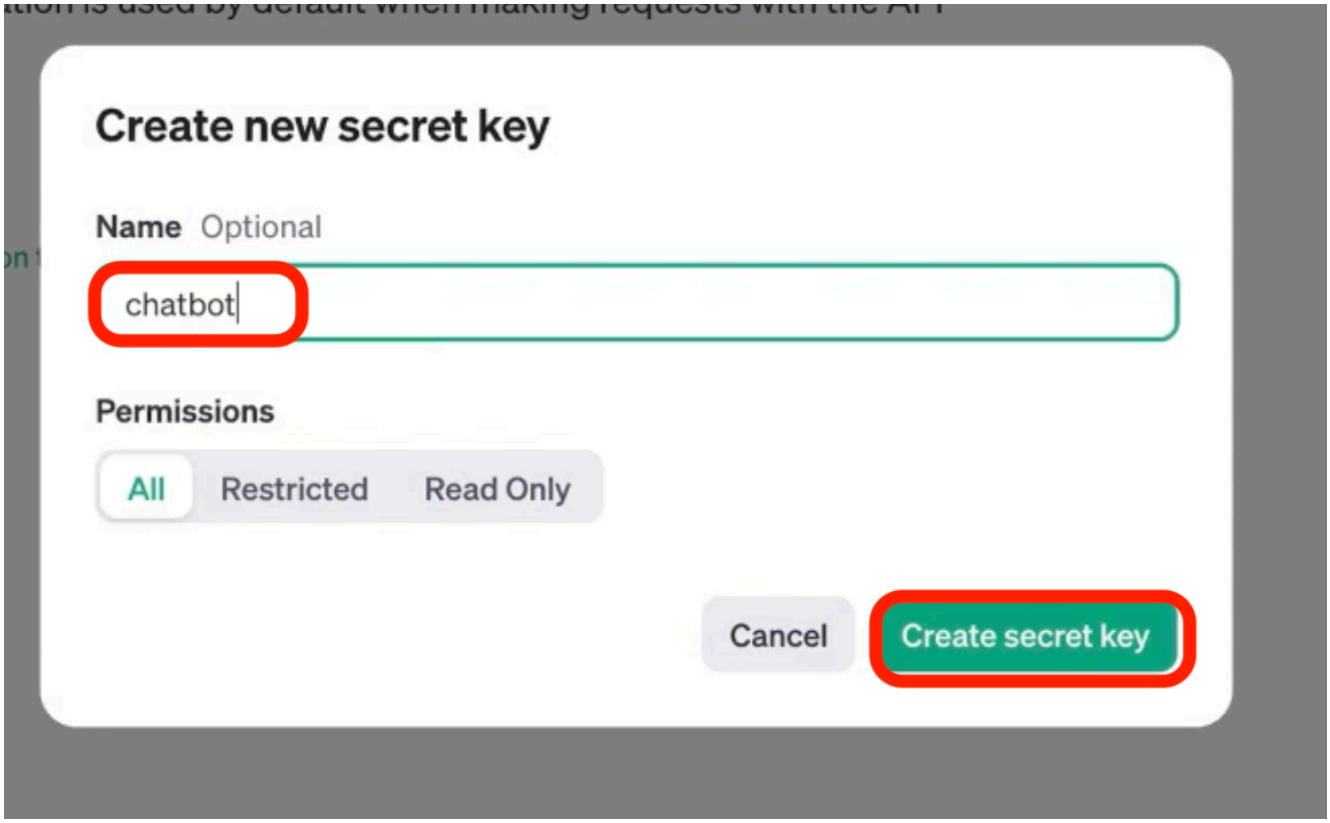


Click on openai.com to generate the API TOKEN

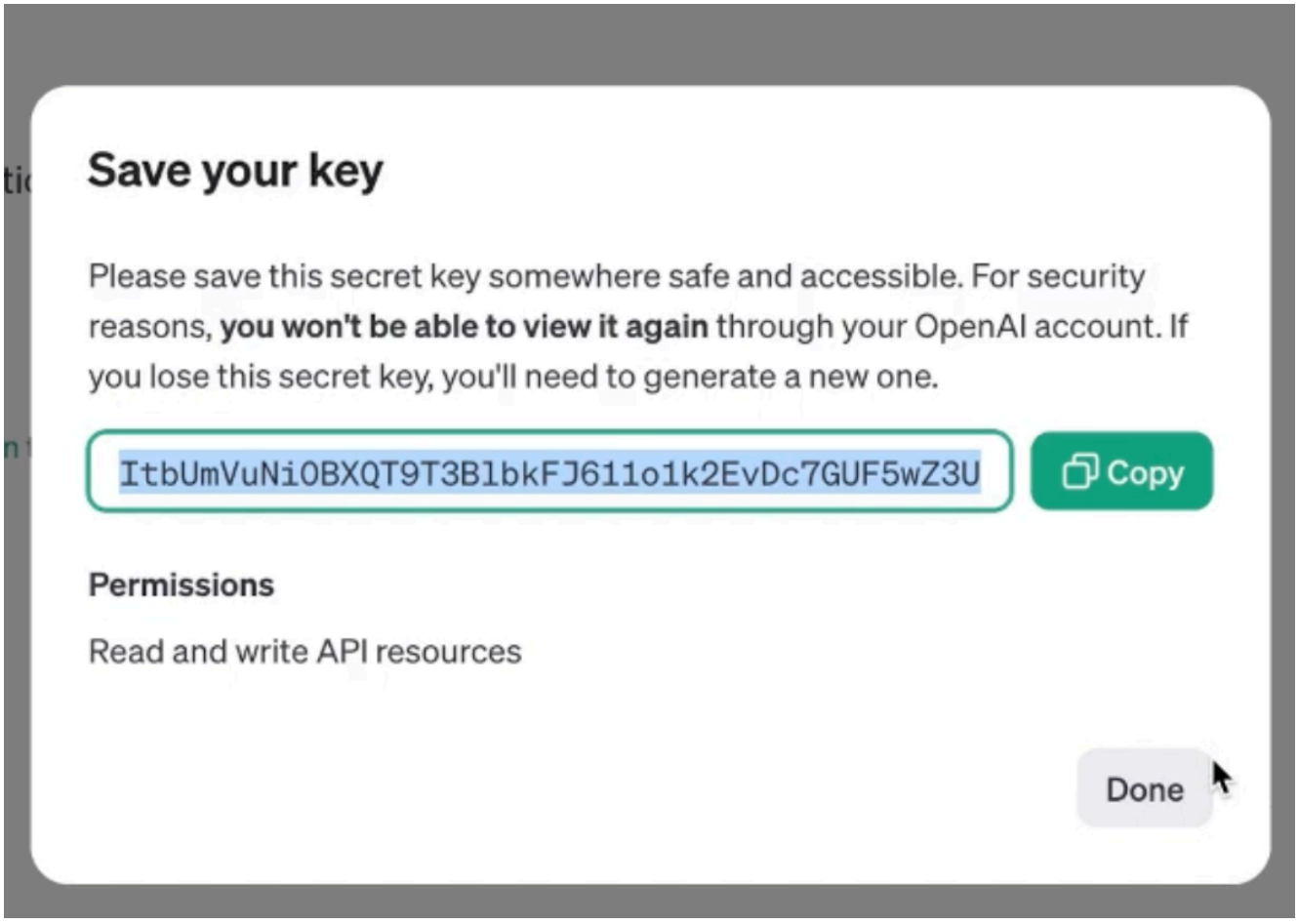
Click on Create new secret key



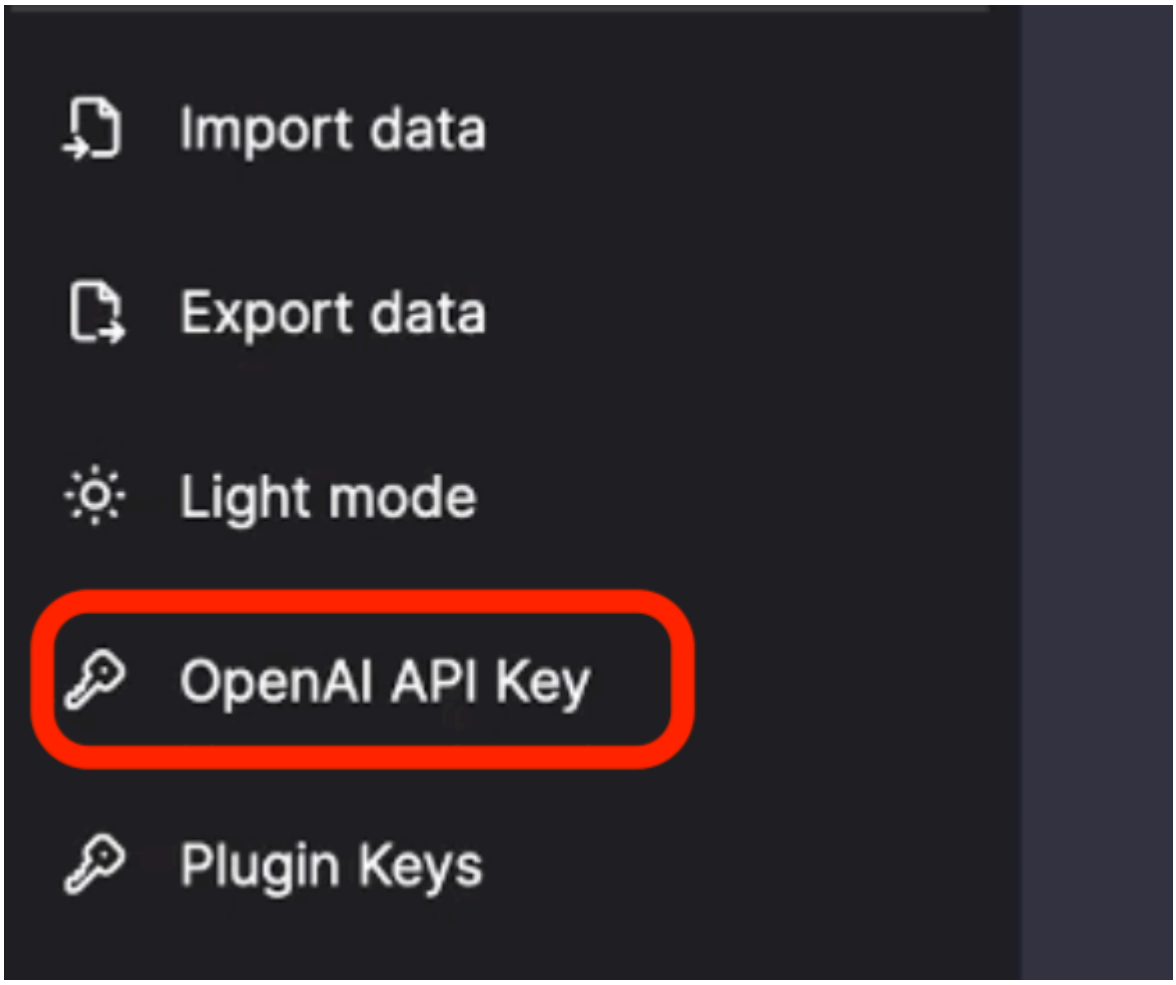
Give Name and click on Create



Copy Token and use

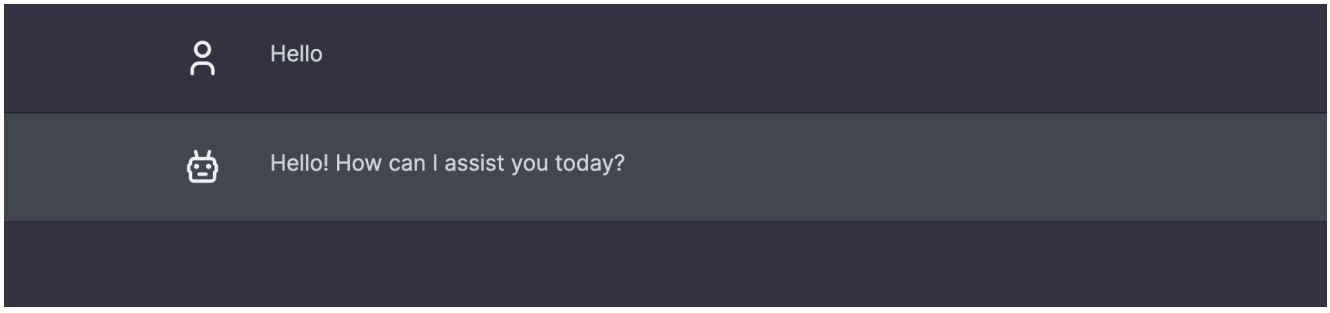


Come back to chatbot UI that we deployed and bottom of the page you will see OpenAI API key and give the Generated key and click on save (RIGHT MARK)



You will see page like this

You can ask questions Now



Or you can generate from Rapidapi.com

https://rapidapi.com/hub

Now In the Jenkins Instance

Give this command

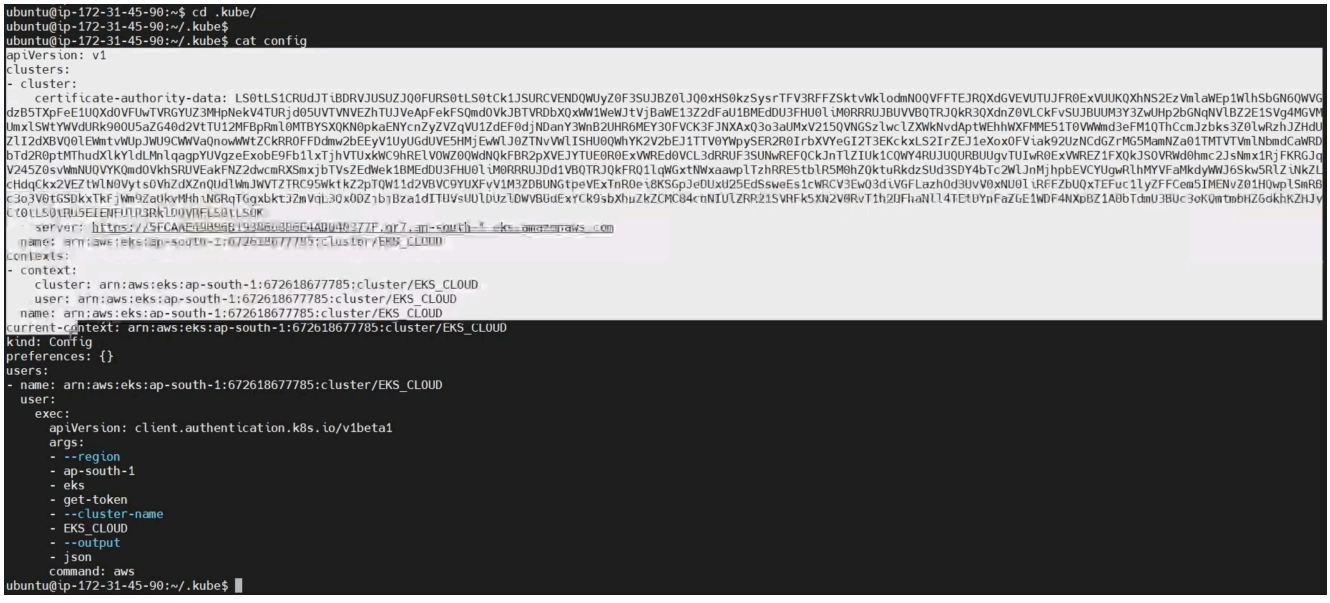
```
eks update-kubeconfig --name &lt;clustername&gt; --region &lt;region&gt
```

It will Generate an Kubernetes configuration file

Here is the path for config file

```
cd .kube
cat config

copy the file that generates
```

Save it in your local file explorer, at your desired location with any name as text file.



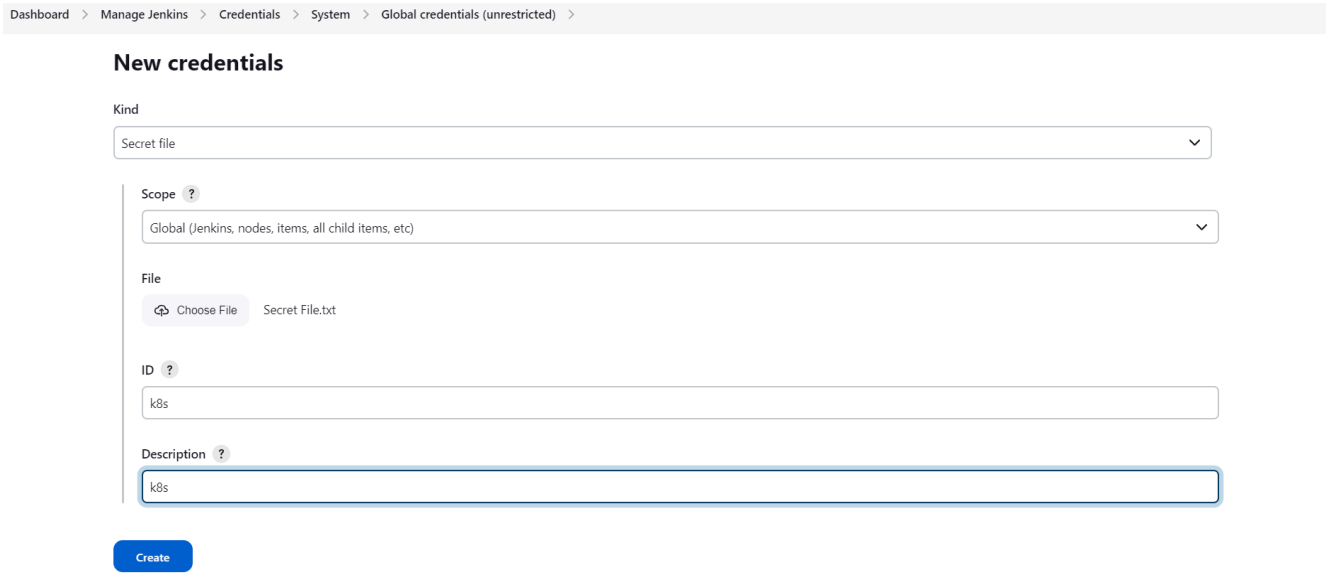
If you Have any doubts refer the YouTube video

https://youtu.be/aNHdarUeo5U time stamp 39m:20 secs

Let's add the Final script with Kubernetes

Go to manage Jenkins –> manage credentials –> Click on Jenkins global –> add credentials

Select Kind as Secret file and choose the file that you saved in your local for kubernetes configuration.




```
stage('Deploy to kubernets'){
            steps{
                script{
                    withKubeConfig(caCertificate: '', clusterName: '', c
                       sh 'kubectl apply -f k8s/chatbot-ui.yaml'
                  }
                }
            }
        }
```

Complete script

Dont forgot to update the Image in chatbot-ui yml file

```
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/Aj7Ay/cha
            }
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
        stage("Sonarqube Analysis "){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.proje
                    -Dsonar.projectKey=Chatbot '''
                }
            }
        }
        stage("quality gate"){
           steps {
                script {
                    waitForQualityGate abortPipeline: false, credentials
                }
            }
        }
        stage('OWASP FS SCAN') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --disabl
                dependencyCheckPublisher pattern: '**/dependency-check-r
            }
        }
        stage('TRIVY FS SCAN') {
            steps {
                sh "trivy fs . &gt; trivyfs.json"
```

```
            }
        }
        stage("Docker Build &amp;amp; Push"){
            steps{
                script{
                   withDockerRegistry(credentialsId: 'docker', toolName
                       sh "docker build -t chatbot ."
                       sh "docker tag chatbot sevenajay/chatbot:latest "
                       sh "docker push sevenajay/chatbot:latest "
                    }
                }
            }
        }
        stage("TRIVY"){
            steps{
                sh "trivy image sevenajay/chatbot:latest &amp;gt; trivy.jso
            }
        }
        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 sevenajay/
            }
        }
        stage('Deploy to kubernets'){
            steps{
                script{
                    withKubeConfig(caCertificate: '', clusterName: '', c
                       sh 'kubectl apply -f k8s/chatbot-ui.yaml'
                  }
                }
            }
        }
    }
```

Apply and save , Run the build to deploy to eks

Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Deploy to container	Deploy to kubernets
132ms	264ms	1s	25s	295ms	1min 49s	2min 38s	23s	1min 51s	1min 35s	1s	2s
133ms	261ms	1s	25s	284ms	1min 51s	2min 46s	23s	1min 23s	1min 52s	1s	1s

In the Jenkins give this command

```
kubectl get all
kubectl get svc #use anyone

Here it will generate loadbalancer DNS dont forgot
```

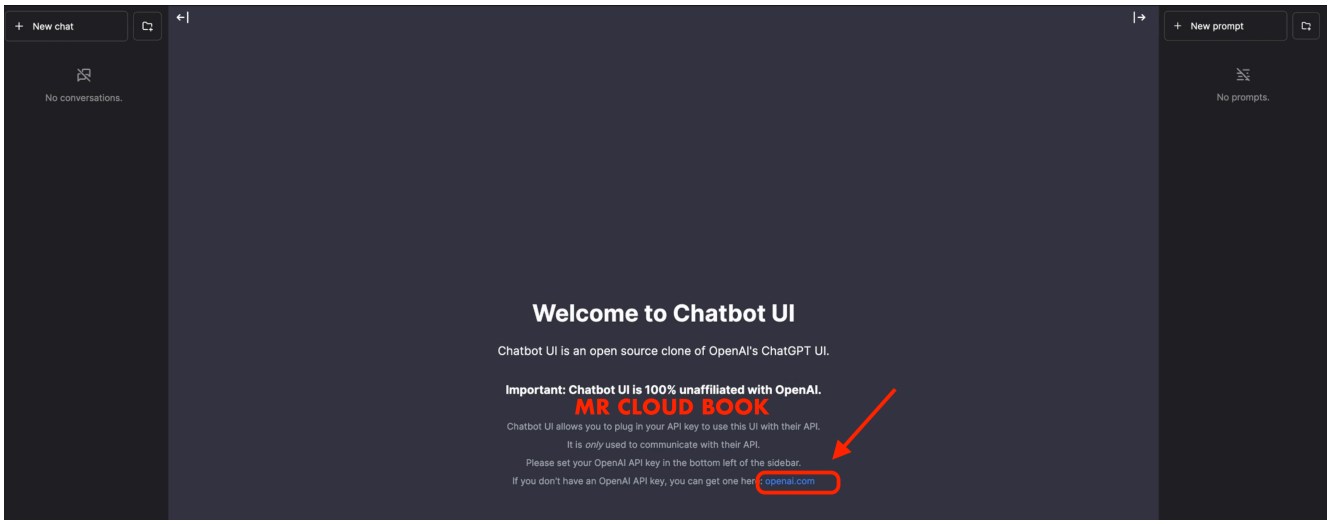
```
ubuntu@ip-172-31-40-131:~$ kubectl get all
NAME                          READY   STATUS    RESTARTS   AGE
pod/petshop-768578655f-kzcd9   1/1     Running   0          43s

NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
service/kubernetes   ClusterIP      10.96.0.1        <none>        443/TCP        58m
service/petshop      LoadBalancer   10.104.122.152   <pending>     80:30699/TCP   21m

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/petshop    1/1     1            1           43s

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/petshop-768578655f   1         1         1       43s
ubuntu@ip-172-31-40-131:~$
```

Open the load-balancer port to the Cluster EC2 instance

otherwise it wont give output

EXTERNAL IP IN browser gives output

output:



Do the same process and add key to get output

Here is sample query output i used

bash Copy code

```
docker network prune
```

5. Remove Docker Cache

To cleanup Docker cache or build cache, use:

bash Copy code

```
docker builder prune
```

- Remove all (include dangling) build cache:

bash Copy code

```
docker builder prune -a
```

6. Remove Everything

- To remove all unused data in one command:

bash Copy code

```
docker system prune -a
```

You can also specify `-f` or `--force` in the command to bypass the confirmation prompt.

Note: Be very careful when using these commands and ensure you don't need the data you're removing, especially when using the `-a` (all) or `-f` (force)

DESTRUCTION

Update the pipeline with this code and Run again to remove the deployment and container

```
pipeline{
    agent any
    stages {
        stage ("Remove container") {
            steps{
                sh "docker stop chatbot | true"
                sh "docker rm chatbot | true"
             }
        }
        stage('Deploy to kubernets'){
            steps{
                script{
                    withKubeConfig(caCertificate: '', clusterName: '', c
                       sh 'kubectl delete -f k8s/chatbot-ui.yaml'
                  }
                }
            }
        }
        stage('scan cluster'){
            steps{
                script{
                    withKubeConfig(caCertificate: '', clusterName: '', c
                       sh 'trivy k8s --report summary cluster'
                  }
                }
            }
        }
    }
```

The above Script will remove Loadbalancer from Ec2 and removes deployment.

Go to EKS job to destroy cluster

Select Build with parameters and select destory and build
It will Take 6 Minutes to remove the EKS

Delete the EC2 instance at End.

Together, we're shaping the future of user engagement and paving the way for innovative solutions in the world of technology.

Thanks for Reading the blog.

Aws awscli Chatbot ChatGpt devops DevSecOps Docker EKS Github kubectl kuberenetes Npm openAI Sonarqube Terraform Trivy Ubuntu

Ajay Kumar Yegireddi is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

Comments

12 responses to "Open Source Project: DevSecOps for OpenAI Chatbot UI Deployment | DevSecOps"

Snehal Sengar
15 March 2024

Really helpful and clear explanation to the project.

Reply

Snehal Sengar
15 March 2024

Really helpful and clear explanation to the project.Thank you

Reply

Kelvin
22 March 2024

E0322 07:54:41.409163 20196 memcache.go:265] couldn't get current server API group list: the server has asked for the client to provide credentials
E0322 07:54:42.248879 20196 memcache.go:265] couldn't get current server API group list: the server has asked for the client to provide credentials
E0322 07:54:43.063662 20196 memcache.go:265] couldn't get current server API group list: the server has asked for the client to provide credentials
E0322 07:54:43.891800 20196 memcache.go:265] couldn't get current server API group list: the server has asked for the client to provide credentials
E0322 07:54:44.700283 20196 memcache.go:265] couldn't get current server API group list: the server has asked for the client to provide credentials
error: You must be logged in to the server (the server has asked for the client to provide credentials)
————————————————————————————————
This a good project and I have followed the entire procedure but I'm getting the above error message.What could I be missing?
Thank you!

Reply

Kkrish
2 December 2024

Did u find a solution for it???

Reply

mrcloudbook.com
2 December 2024

which one ?

Reply

Zakład pogrzebowy Wola
7 June 2024

Your writing is both insightful and accessible. I really enjoyed reading your article.

Reply

Wiaty Fotowoltaiczne
15 June 2024

What a compelling read! Your points are well-articulated and thought-provoking.

Reply

Inwertery Solarne
16 June 2024

Your post is a goldmine of information. It's evident you've put a lot of research and effort into it.

Reply

Lot Balonem Cennik
8 July 2024

I'm always excited to read your posts, and this one exceeded my expectations. Excellent work!

Reply

Ile Kosztuje Lot Balonem

9 July 2024

Your post was not just informative, but also beautifully written. Your talent is evident.

Reply

pardeep kaur
15 October 2024

Thank you for amazing project. but in repo there is no Dockerfile is available sir. can you please provide that one.

Reply

Abhinav Shrivastava
19 November 2024

Thanks for the blog.

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

I'm not a robot

reCAPTCHA
Privacy - Terms

Post Comment

Uncategorized

How to Automate Incident Response : How Q Developer Helped Me Automate a Daily Pain Point

22 July 2025

AI

How to Run Docker Model Runner on Ubuntu 24.04

11 July 2025

AI, DevOps

How to Install docker-ai on Ubuntu 24.04

15 June 2025

Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.

Important Links

Privacy Policy

Terms & Conditions

Contact

Resources

Blog

YouTube Channel

Follow Us on YouTube