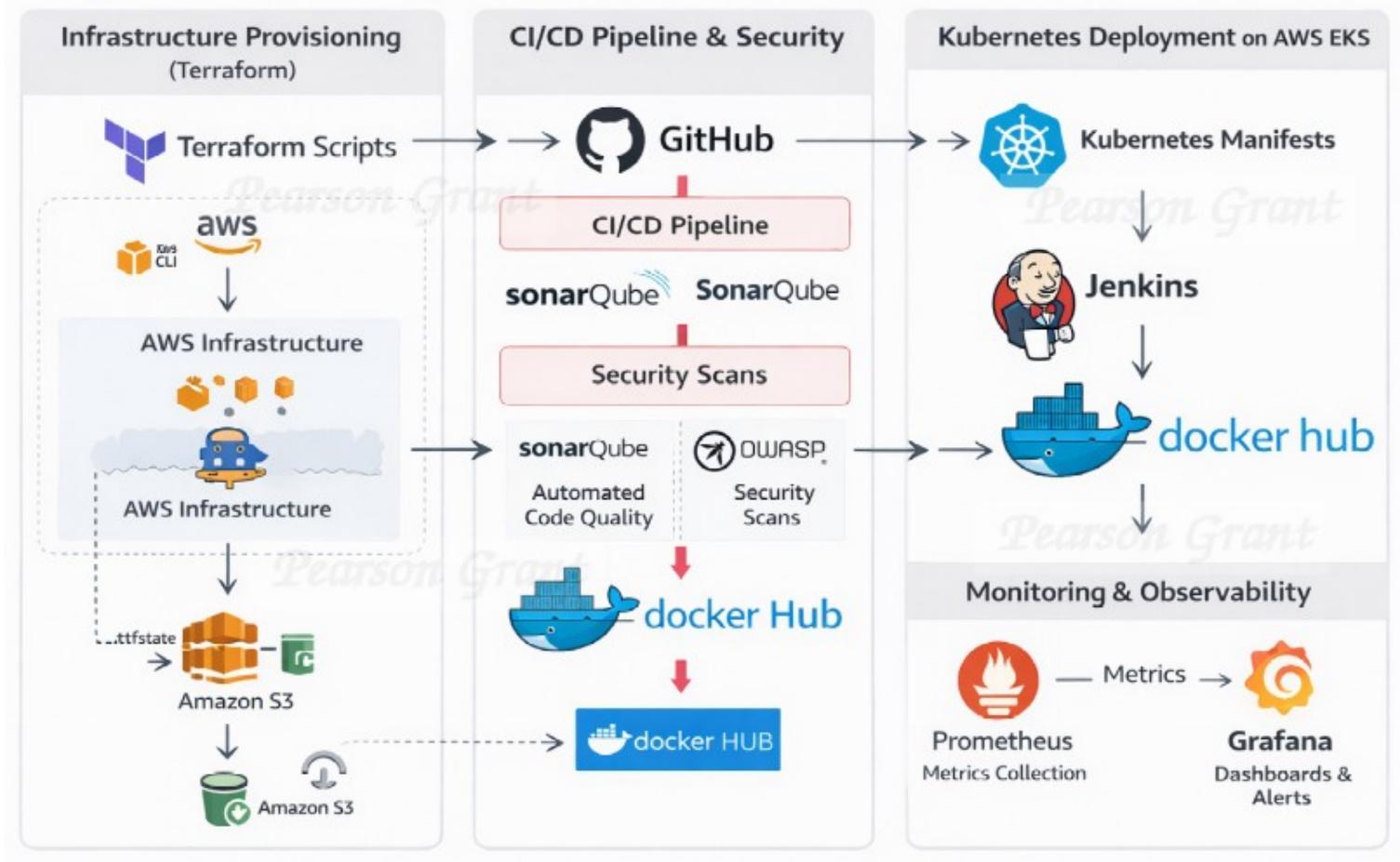


Deploying a ChatGPT Clone on AWS EKS Using Terraform, Jenkins, and Prometheus/Grafana



Introduction

In this project, I designed and implemented an end-to-end DevOps solution to deploy a ChatGPT-like application on AWS EKS. The implementation covers infrastructure provisioning with Terraform, CI/CD automation using Jenkins, containerization with Docker, Kubernetes-based deployment, security and quality enforcement, and full observability using Prometheus and Grafana.

This article documents a complete, step-by-step DevOps implementation, focusing not only on what was deployed, but on *how and why* each decision was made. The structure reflects real-world DevOps problem-solving workflows and is intended for technical readers, hiring managers, and recruiters evaluating practical cloud and DevOps experience.

Project Objectives

- Provision cloud infrastructure using Infrastructure as Code (Terraform)
 - Implement a secure CI/CD pipeline with Jenkins
 - Deploy the application on Kubernetes (EKS)
 - Apply code quality and security scanning practices
 - Implement monitoring and observability using Prometheus and Grafana
 - Ensure proper resource cleanup to control cloud costs
-

Technology Stack

- **Cloud Provider:** AWS
 - **IaC:** Terraform
 - **CI/CD:** Jenkins
 - **Containerization:** Docker
 - **Orchestration:** Kubernetes (Amazon EKS)
 - **Code Quality:** SonarQube
 - **Security:** OWASP Dependency Check, Trivy
 - **Monitoring:** Prometheus, Node Exporter, Grafana
 - **Application:** Node.js-based ChatGPT UI
-

Prerequisites

- Active AWS account
 - Basic knowledge of Terraform, Jenkins, Docker, and Kubernetes
 - AWS CLI installed locally
 - clone this repo: <https://github.com/PearsonGrant24/Chat-gpt-clone-deployment.git>
-

PHASE 1: Local Setup + AWS Foundations

Step 1.1 – AWS Account Basics (Very Important)

You need:

- An AWS account
- IAM user (not root)
- Programmatic access

IAM Permissions (Minimum Required)

Your IAM user should have:

- AdministratorAccess (for learning projects)

If you don't know how to create an IAM user:

A: Log in as ROOT (One Time Only)

1. Go to **AWS Console**
2. Login using **root email + password**
3. Complete MFA if enabled

This is the **only time** we'll use root.

B: Create IAM User

Navigate

AWS Console → IAM → Users → Create IAM user

The screenshot shows the AWS IAM service page. On the left, there's a sidebar with links for Services, Features, Resources (New), Documentation, Knowledge articles, Marketplace, Blog posts, Events, and Tutorials. The main content area has a heading 'Services' with a 'Show more' link. It lists three services: 'IAM' (Manage access to AWS resources), 'IAM Identity Center' (Manage workforce user access to multiple AWS accounts and cloud applications), and 'Resource Access Manager' (Share AWS resources with other accounts or AWS Organizations). Below this, there's a section titled 'Features' with a 'Show more' link, listing 'Groups' (IAM feature), 'Roles' (IAM feature), and 'Identity providers' (IAM feature). At the bottom, there's a link 'Resources / for a focused search' and a note 'Were these results helpful?'. A vertical scroll bar is visible on the right side of the page.

User Details

- **Enter user name:** Pears;
- Check: **Provide user access to the AWS Management Console**

Click **Next**

us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create

Search [Alt+S]

Global ▾

Users > Create user

Step 1 Specify user details

Step 2 Set permissions

Step 3 Review and create

Step 4 Retrieve password

Specify user details

User details

User name Pears

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = . _ - (hyphen)

Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

Are you providing console access to a person?

Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keypairs, or a backup credential for emergency account access.

Console password

Autogenerated password
You can view the password after you create the user.

Custom password
Enter a custom password for the user.

- * Must be at least 8 characters long
- * Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # \$ % ^ & * () _ + - (hyphen) = [] { } | `

Users must create a new password at next sign-in - Recommended
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user.

C: Permissions (IMPORTANT)

Choose:

Attach policies directly

Select:

AdministratorAccess

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

- Add user to group
- Copy permissions
- Attach policies directly

Permissions policies (1/1354)

Choose one or more policies to attach to your new user.

Policy name	Type	Attached entities
<input type="checkbox"/> AccessAnalyzerServiceRolePolicy	AWS managed	0
<input checked="" type="checkbox"/> AdministratorAccess	AWS managed - job function	0
<input type="checkbox"/> AdministratorAccess-Amplify	AWS managed	0
<input type="checkbox"/> AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
<input type="checkbox"/> AIOpsAssistantPolicy	AWS managed	0
<input type="checkbox"/> AIOpsConsoleAdminPolicy	AWS managed	0
<input type="checkbox"/> AIOpsOperatorAccess	AWS managed	0
<input type="checkbox"/> AIOpsReadOnlyAccess	AWS managed	0

[Create policy](#)

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name Pears	Console password type Custom password	Require password reset Yes
--------------------	--	-------------------------------

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy
IAMUserChangePassword	AWS managed	Permissions policy

Tags - optional

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create user](#)

This is OK for learning projects
(Production would be more restricted)

Click **Next → Create user**

D: Create Access Keys (Programmatic Access)

1. Open devops-user , we named Pears
2. Go to Security credentials

3. Click **Create access key**

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more [\[?\]](#)

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. Learn more [\[?\]](#)

[Create access key](#)

4. Choose:

Command Line Interface (CLI)

Step 1
Access key best practices & alternatives

Step 2 - optional
Set description tag

Step 3
Retrieve access keys

Access key best practices & alternatives [Info](#)

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

5. Click **Create**

SAVE THESE IMMEDIATELY:

- Access Key ID
- Secret Access Key

You will NOT see the secret again.

E: Configure AWS CLI on Linux

On your Linux terminal:

```
aws configure
```

```
pearson@Pear:~/Desktop/Chat-gpt-deployment$ aws configure
AWS Access Key ID [*****OIN2]: AKIAXLGHPIWCGJDN7EEZ
AWS Secret Access Key [*****Y3Rm]: +U00EeBrEM+8ixpl9BMxF9dpq2VfsjMzLC
At0dHk
Default region name [us-east-1]:
Default output format [None]
pearson@Pear:~/Desktop/Chat-gpt-deployment$
```

Enter:

```
AWS Access Key ID:      <your-access-key>
AWS Secret Access Key: <your-secret-key>
Default region name:  us-east-1
Default output format: json
```

F: Verify IAM Setup

Run:

```
aws sts get-caller-identity
```

Expected Output

```
{
  "Account": "123456789012",
  "Arn": "arn:aws:iam::123456789012:user/devops-user",
  "UserId": "AIDA..."
}
```

This confirms:

- IAM user works

- Credentials are correct
 - AWS CLI is connected
-

Step 1.2 – Install Required Tools (Local Machine)

AWS CLI

Check:

```
aws --version
```

If not installed:

```
# Linux  
sudo apt install awscli -y
```

Terraform

Check:

```
terraform version
```

If not installed:

```
# Linux  
sudo apt install terraform -y
```

kubectl (We won't use it yet, but install now)

```
kubectl version --client
```

Install if missing:

```
sudo apt install kubectl -y
```

Step 1.3 – Configure AWS Credentials

This connects **your machine → AWS**

Run:

```
aws configure
```

Enter:

```
AWS Access Key ID:      <your-key>
AWS Secret Access Key: <your-secret>
Default region name:   us-east-1
Default output format: json
```

Verify Connection

```
aws sts get-caller-identity
```

Expected output:

```
{
  "Account": "123456789",
  "Arn": "arn:aws:iam::123456789:user/your-user",
  "UserId": "AIDA..."
}
```

PHASE 2: Create Infrastructure as Code using Terraform

Step 2.1 – Terraform Code Review (CRITICAL)

Now let's make sure your Terraform code is safe.

Go to your terraform folder

```
cd terraform
ls
```

Run Terraform Init

```
terraform init
```

The screenshot shows a Visual Studio Code (VS Code) interface with a dark theme. On the left is the Explorer sidebar, which lists files and folders for a project named "CHAT-GPT-DEPLOYMENT". The "main.tf" file is open in the main editor area. The code in "main.tf" defines an AWS instance resource:

```
cidr_blocks = ["0.0.0.0/0"]
ipv6_cidr_blocks = []
prefix_list_ids = []
security_groups = []
self = false

egress {
    from_port = 0
    to_port = 0
    protocol = "-1"
    cidr_blocks = ["0.0.0.0/0"]
}

tags = {
    Name = "Jenkins-sg"
}
```

The terminal window at the bottom right shows the output of running Terraform commands:

```
pearson@Pear:~/Desktop/Chat-gpt-deployment/Instance-terraform$ cd instance-terraform
bash: cd: instance-terraform: No such file or directory
pearson@Pear:~/Desktop/Chat-gpt-deployment$ cd Instance-terraform
pearson@Pear:~/Desktop/Chat-gpt-deployment/Instance-terraform$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.98.0

Terraform has been successfully initialized!
```

A tooltip message is visible in the terminal area: "You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work."

This downloads providers

Run Terraform Plan

```
pearson@Pear: ~/Desktop/Chat-gpt-deployment/Instance-te...
+ {
+   cidr_blocks      = [
+     "+ \"0.0.0.0/0\",
+   ]
+   description      = "TLS from VPC"
+   from_port        = 9090
+   ipv6_cidr_blocks = []
+   prefix_list_ids = []
+   protocol         = "tcp"
+   security_groups  = []
+   self              = false
+   to_port           = 9090
},
+ {
+   cidr_blocks      = [
+     "+ \"0.0.0.0/0\",
+   ]
+   description      = "TLS from VPC"
+   from_port        = 9100
+   ipv6_cidr_blocks = []
+   prefix_list_ids = []
+   protocol         = "tcp"
+   security_groups  = []
+   self              = false
+   to_port           = 9100
},
]
+ name                  = "Jenkins-Security Group"
+ name_prefix          = (known after apply)
+ owner_id              = (known after apply)
+ revoke_rules_on_delete = false
+ tags                 = {
+   + "Name" = "Jenkins-sg"
}
+ tags_all             = {
+   + "Name" = "Jenkins-sg"
}
+ vpc_id                = (known after apply)
}

Plan: 3 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't
guarantee to take exactly these actions if you run "terraform apply" now.
pearson@Pear:~/Desktop/Chat-gpt-deployment/Instance-terra$
```

This shows:

- What AWS resources will be created
- How many EC2s, subnets, IAM roles

Terraform apply --auto-approve

gpt instance got created by terraform with the following configuration

Find Instance by attribute or tag (case-sensitive)									
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
<input type="checkbox"/>	Monitoring via...	i-07a85399bbbd9d8a6	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-13-218-203-53.co...	13.218.203.53
<input type="checkbox"/>	gpt clone	i-005ca4075a706d398	Running	t2.large	Initializing	View alarms +	us-east-1d	ec2-18-215-236-219.co...	18.215.236.219

S

Name	Security group rule ID	Port range	Protocol	Source	Security groups	Description
sgr-044ce032a715b2766	8080	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-04d18c0df9f9ec85	5000	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-09d362f3e165cf11	443	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-09e33d1ff18ab8b341c	9000	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-09e621adadae59a128	9100	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-02ab6376b148bfcd1	22	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-0d139775c80757e0b	9090	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC
sgr-0e5e9c0dded468cf9	80	TCP	0.0.0.0/0	Jenkins-Security Group	Jenkins-Security Group	TLS from VPC

t
o

v
i
e

Attach IAM role with your EC2

1.go to EC2 section

2.click on actions → security → modify IAM role option

a
g

Instances (1/2) Info									
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Actions
<input checked="" type="checkbox"/>	Monitoring via...	i-07a85399bbbd9d8a6	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1b	ec2-13-218-203-53.co...	Launch instances
<input type="checkbox"/>	gpt clone	i-005ca4075a706d398	Running	t2.large	2/2 checks passed	View alarms +	us-east-1d	ec2-18-215-236-219.co...	

i-07a85399bbbd9d8a6 (Monitoring via grafana)

[Details](#) | [Status and alarms](#) | [Monitoring](#) | [Security](#) | [Networking](#) | [Storage](#) | [Tags](#)

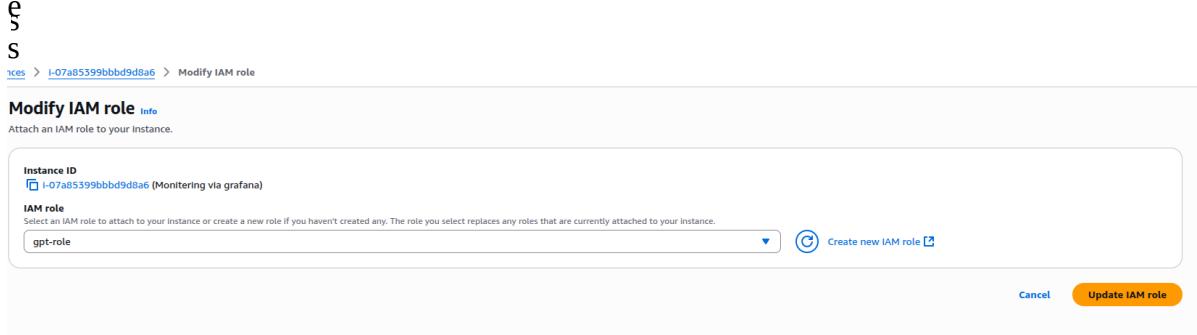
l
l

s
i
z
e

P

P

3. choose the role from dropdown and click on update IAM role



Phase 3 : Setup Sonarqube and Jenkins

STEP 3.1 Sonarqube

copy the public ip of your machine

Access sonarqube

<http://publicip:9000>

Username: admin

Password: admin

Update your password

Welcome to Sonarqube interface

STEP 3.2 — Setup Jenkins

Access Jenkins

`http://<public_ip>:8080`

Unlock Jenkins:

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

Paste your password on jenkins

P

r

e

s

e

n

t

e

o

r

c

l

i

c

k

t

o

v

i



A screenshot of the Jenkins Dashboard. The top navigation bar shows a browser tab for 'Not secure 13.220.231.156:8080'. The dashboard header includes the Jenkins logo and a 'Dashboard' link. On the left, there are links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. A 'Build Queue' section indicates 'No builds in the queue.' Below the queue is a 'Build Executor Status' section showing '0/2'. The main content area features a 'Welcome to Jenkins!' message, a 'Start building your software project' section with a 'Create a job' button, and a 'Set up a distributed build' section with links for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'.

S

i

Z

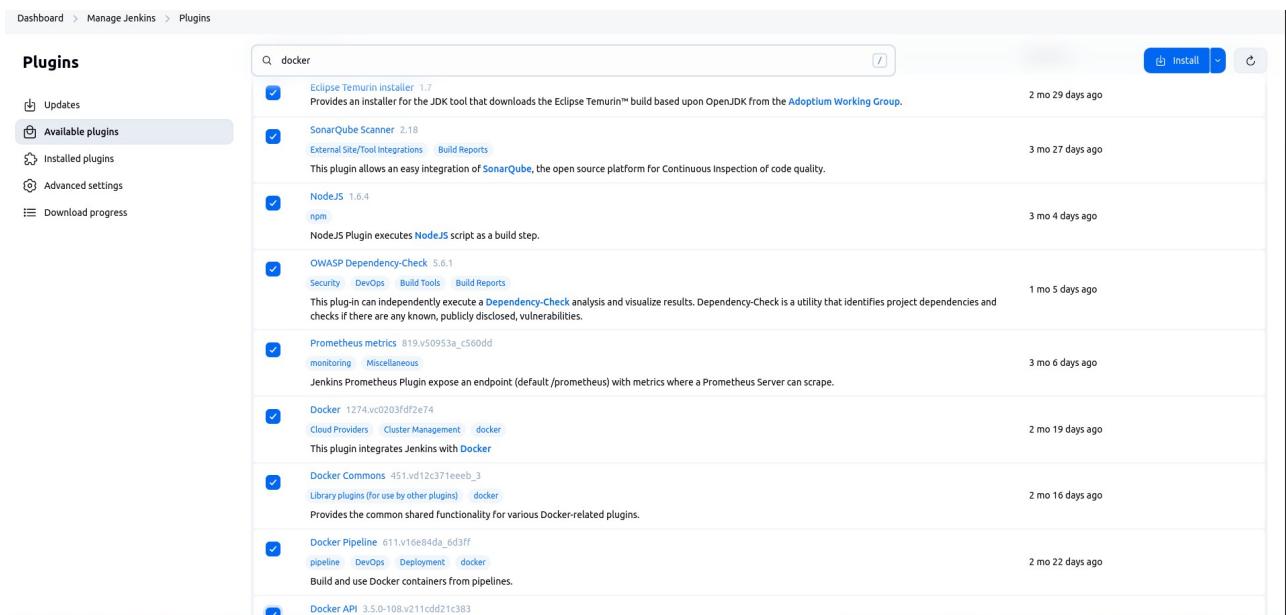
e

Welcome to Jenkins dashboard

Phase 4 → CI-CD pipeline

STEP 4.1 — Install Required Tools on Jenkins Server

All the Docker plugins



The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. A search bar at the top contains the text 'docker'. Below it, a table lists several Jenkins plugins:

Plugin	Description	Last Updated
Eclipse Temurin Installer	Provides an installer for the JDK tool that downloads the Eclipse Temurin™ build based upon OpenJDK from the Adoptium Working Group .	2 mo 29 days ago
SonarQube Scanner	External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	3 mo 27 days ago
NodeJS	npm NodeJS Plugin executes NodeJS script as a build step.	3 mo 4 days ago
OWASP Dependency-Check	Security DevOps Build Tools Build Reports This plugin can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.	1 mo 5 days ago
Prometheus metrics	monitoring Miscellaneous Jenkins Prometheus Plugin expose an endpoint (default /prometheus) with metrics where a Prometheus Server can scrape.	3 mo 6 days ago
Docker	Cloud Providers Cluster Management docker This plugin integrates Jenkins with Docker	2 mo 19 days ago
Docker Commons	Library plugins (for use by other plugins) docker Provides the common shared functionality for various Docker-related plugins.	2 mo 16 days ago
Docker Pipeline	pipeline DevOps Deployment docker Build and use Docker containers from pipelines.	2 mo 22 days ago
Docker API		

Terraform

Eclipse Temurin Installer (Install without restart)

SonarQube Scanner (Install without restart)

NodeJs Plugin (Install Without restart)

Owasp → The OWASP Plugin in Jenkins is like a “security assistant” that helps you find and fix security issues in your software. It uses the knowledge and guidelines from the Open Web Application Security Project (OWASP) to scan your web applications and provide suggestions on how to make them more secure. It’s a tool to ensure that your web applications are protected against common security threats and vulnerabilities.

Prometheus metrics → to monitor Jenkins on Grafana dashboard

Download all the Docker related plugins

Kubernetes

Kubernetes CLI

Kubernetes Client API

Kubernetes Pipeline DevOps steps

AWS CLI

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. The left sidebar has 'Available plugins' selected. A search bar at the top contains the text 'kuber'. Below the search bar, there is a list of available plugins:

- Docker API** (version 3.5.0-108.v211cdd21c383): Library plugins (for use by other plugins) → docker. This plugin provides docker-java API for other plugins. Last updated 1 mo 24 days ago.
- Kubernetes Client API** (version 1.10.0-251.v556f5f100500): kubernetes → Library plugins (for use by other plugins). Kubernetes Client API plugin for use by other Jenkins plugins. Last updated 2 mo 23 days ago.
- Kubernetes Credentials** (version 192.v4d5b_1c429d17): kubernetes → credentials. Common classes for Kubernetes credentials. Last updated 2 mo 23 days ago.
- Kubernetes** (version 4340.v345364d31a_2a_): Cloud Providers → Cluster Management → kubernetes → Agent Management. This plugin integrates Jenkins with Kubernetes. Last updated 11 days ago.
- Kubernetes CLI** (version 1.36.4.vadef8cb8b823): kubernetes. Configure kubectl for Kubernetes. Last updated 2 mo 5 days ago.
- Kubernetes Credentials Provider** (version 1.276.v99a_de03cb_076): kubernetes → credentials. Provides a read only credentials store backed by Kubernetes. Last updated 2 mo 19 days ago.
- Kubernetes :: Pipeline :: DevOps Steps** (version 1.6): pipeline → kubernetes. Last updated 6 yr 3 mo ago.
- GitLab Credentials - Kubernetes Integration** (version 424.vd4c848a_61813): kubernetes → gitlab. Last updated 3 mo 2 days ago.

The URL <https://plugins.jenkins.io/kubernetes-pipeline-devops-steps> is visible at the bottom of the page.

Dashboard > Manage Jenkins > Plugins

Plugins

Updates Available plugins Installed plugins Advanced settings Download progress

Plugin	Status
Dark Theme	Success
Loading plugin extensions	Success
Eclipse Temurin installer	Success
SonarQube Scanner	Success
Config File Provider	Success
NodeJS	Success
OWASP Dependency-Check	Success
Pipeline: REST API	Success
Prometheus metrics	! prometheus plugin doesn't support dynamic loading. Jenkins needs to be restarted for the update to take effect.
Cloud Statistics	Success
Authentication Tokens API	Success
Docker Commons	Success
Apache HttpComponents Client 5.x API	Success
Commons Compress API	Success
Docker API	Success
Docker	Success
Docker Commons	Success
Docker Pipeline	Success
Docker API	Success
Kubernetes Client API	Success
Kubernetes Credentials	Success
Kubernetes	Success
Kubernetes CLI	Success
Oracle Java SE Development Kit Installer	Success
SSH server	Success
Command Agent Launcher	Success
JavaMail API	Success
Kubernetes :: Pipeline :: DevOps Steps	Success
Loading plugin extensions	Success

→ Go back to the top page

STEP 4.2 — Add SonarQube credentials to Jenkins

WE FIRST GENERATE A TOKEN FOR SONARQUBE TO USE IT IN JENKINS CREDENTIALS AS SECRETE TEXT

Go to **SonarQube Server**:

- URL: `http://publicip:9000`

security → users → token → generate token

- Token name: `jenkins`

Administration

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

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 another database.

SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.8 (build 100196) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins - Web API](#)

Get the most out of SonarQube!
Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

r

Administration

Tokens of Administrator

Generate Tokens

Name	Expires in	Generate
Enter Token Name	30 days	

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

Copy squ_86466ca3a31828fedc52923a7a5b8499841cb8f

Name	Type	Project	Last use	Created	Expiration
chatgpt	User		Never	July 6, 2025	August 5, 2025

Done

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 another database.

SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.8 (build 100196) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins - Web API](#)

Get the most out of SonarQube!
Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

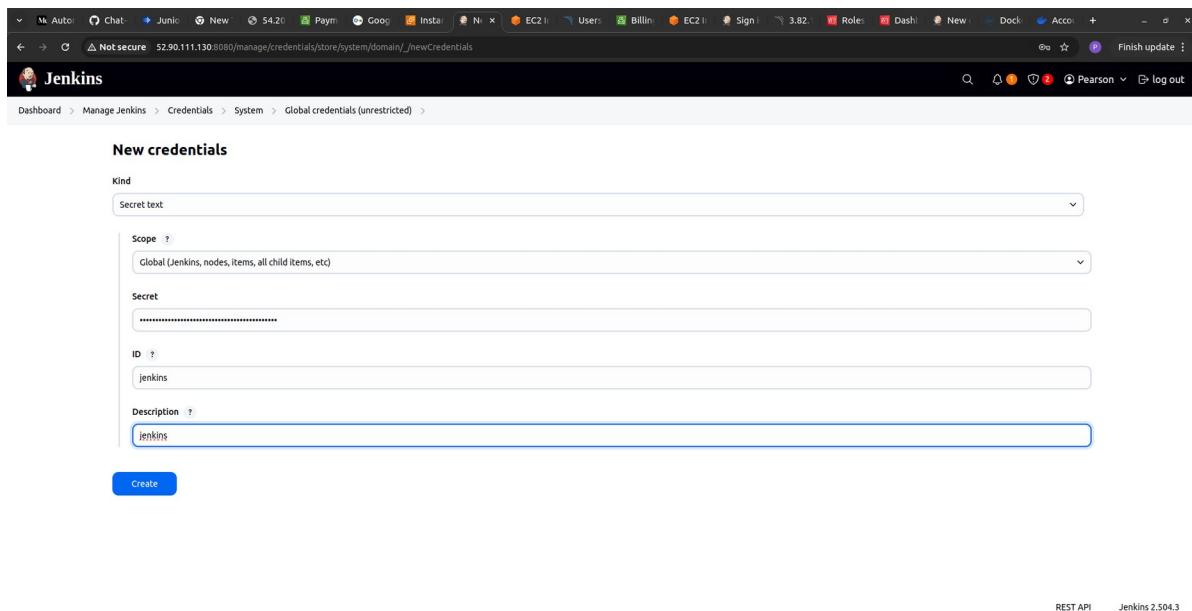
III

a
g
e

- i• **Copy the token and go to your Jenkins → manage Jenkins**
- n → **credentials → global → add credentials**
- f• Select secret text from dropdown
- u
- l• **secret text: your token**
- l• **id: jenkinsn**

s
i
z
e

- Click: create



STEP 4.3 – Setup projects in Sonarqube for Jenkins

- Go to your Sonarqube server
- Click on projects
- In the name field type gpt
- Click on set up

All fields marked with * are required

Project display name *
git

Up to 255 characters. Some scanners might override the value you provide.

Project key *
git

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Main branch name *
main

The name of your project's default branch [Learn More](#)

[Set Up](#)

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 in the future.

SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.9.8 [build 100196] - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)

Get the most out of SonarQube!
+ Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow.
- Connect SonarLint to SonarQube to sync rule sets and issue states.

[Learn More](#) [Dismiss](#)

All fields marked with * are required

Project display name *
gpt

Up to 255 characters. Some scanners might override the value you provide.

Project key *
gpt

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Main branch name *
main

The name of your project's default branch [Learn More](#)

[Set Up](#)

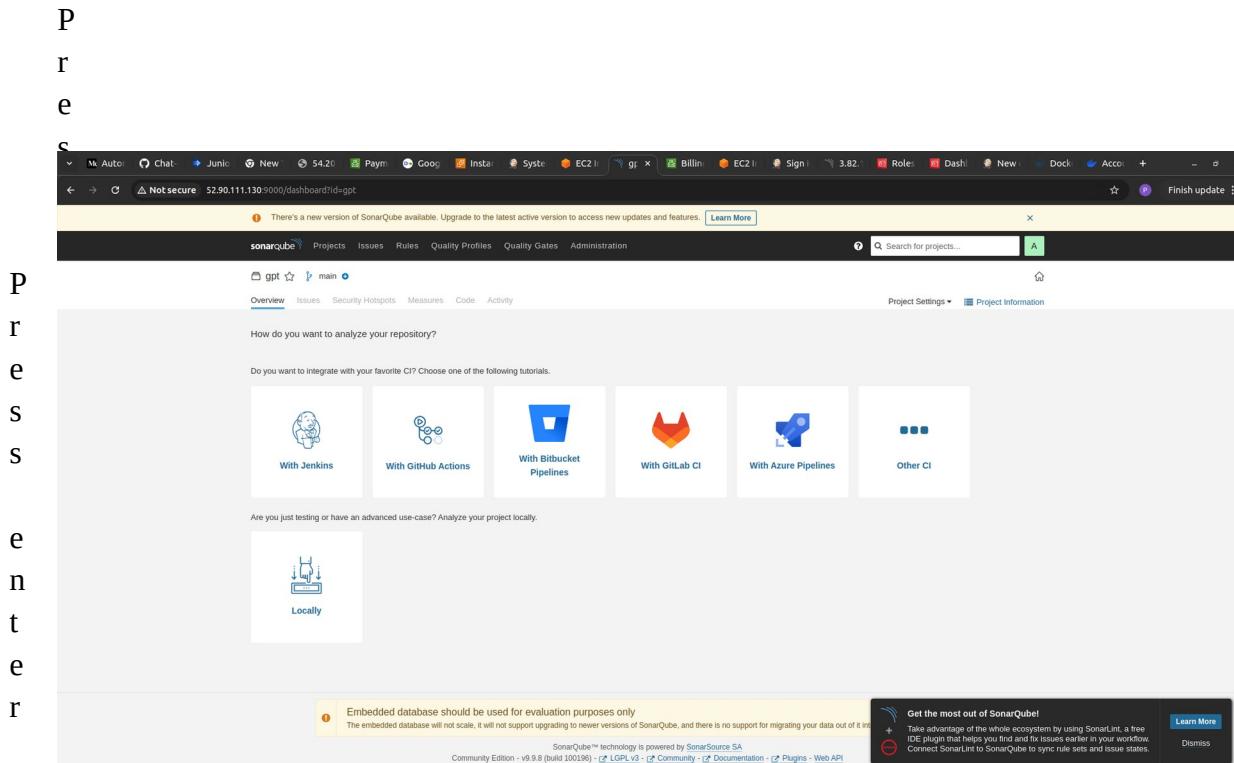
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it in the future.

SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.9.8 [build 100196] - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)

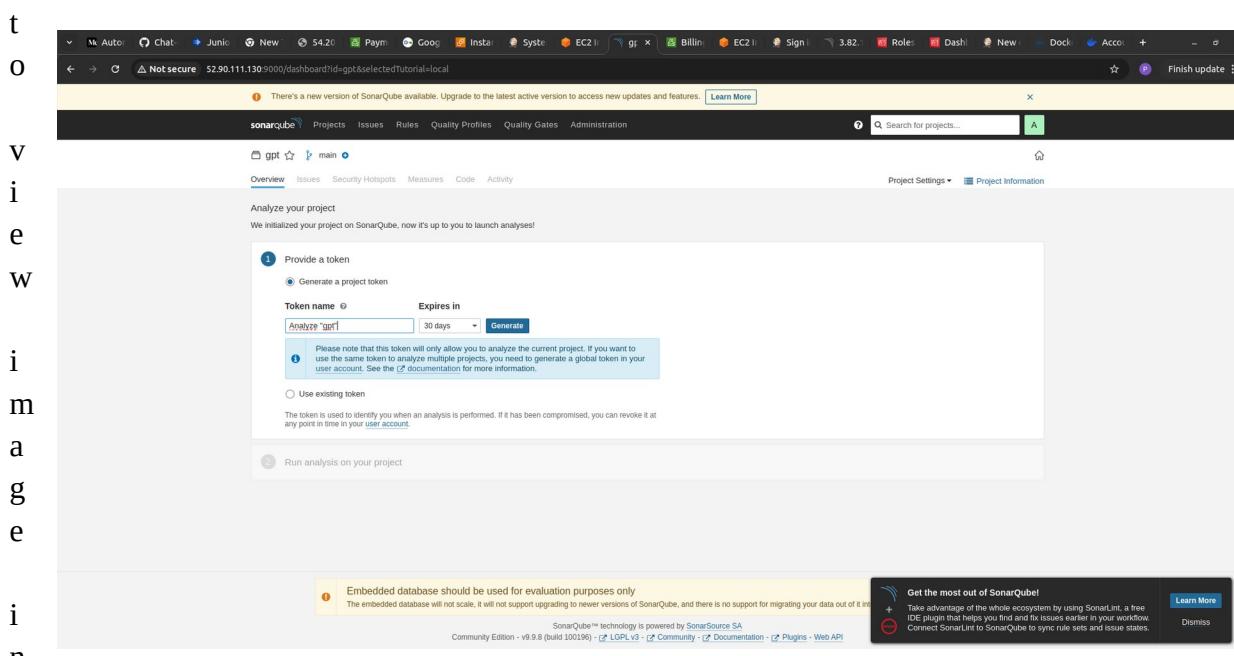
Get the most out of SonarQube!
+ Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow.
- Connect SonarLint to SonarQube to sync rule sets and issue states.

[Learn More](#) [Dismiss](#)

click on locally

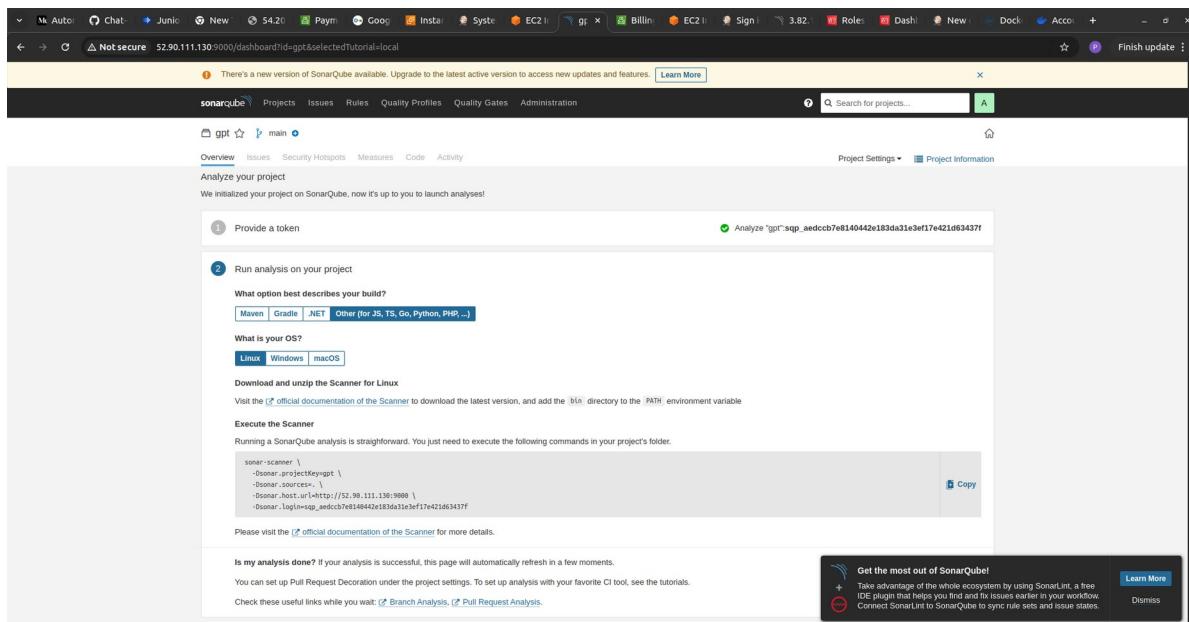


c
l
i
·
c
k click on generate



f
u
l
l

s
i
z
e



click on continue

Sonarqube project for Jenkins is setup now

STEP 4.4 — DockerHub Credentials (REQUIRED)

Create DockerHub credentials in Jenkins

Jenkins → Manage Jenkins → Credentials → Global → Add Credentials

- Kind: Username with password
- ID: docker
- Username: <your-dockerhub-username>
- Password: <dockerhub-password>

Save.

The screenshot shows the Jenkins Global credentials (unrestricted) page. It lists two entries:

ID	Name	Kind	Description
jenkins	jenkins	Secret text	jenkins
docker	pearpear23/*****	Username with password	

At the bottom right, there are links for REST API and Jenkins 2.504.3.

STEP 4.5 – Setup tools for jenkins

**go to:
jenkins → tools**

a. Add jdk

1. click on add jdk and select installer adoptium.net
2. choose jdk 17.0.8.1+1version and in name section enter jdk 17

The screenshot shows the Jenkins management interface under the 'Tools' section. It displays configurations for Java Development Kits (JDK) and Git installations.

JDK installations:

- Add JDK:** A new entry named "jdk 17 jdk 17.0.8+1" is listed.
- Install automatically:** The checkbox is checked.
- Install from adoptium.net:** The dropdown shows "jdk-17.0.8+1+1".
- Add Installer:** A button to add more installers.

Git installations:

- Add Git:** A new entry named "git" is listed.

Buttons: Save and Apply.

b. Add node js

1. click on add nodejs
2. enter node16 in name section
3. choose version nodejs 16.2.0

The screenshot shows the Jenkins management interface under the 'Tools' section. It displays configurations for NodeJS installations.

Add NodeJS:

- Name:** node16
- Install automatically:** The checkbox is checked.
- Install from nodejs.org:** The dropdown shows "NodeJS 16.2.0".
- Force 32bit architecture:** An unchecked checkbox.
- Global npm packages to install:** A text input field containing "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'".
- Global npm packages refresh hours:** A text input field containing "72".

Buttons: Save and Apply.

c. Add Docker

1. click on add docker
2. name==docker
3. add installer ==download from docker.com

The screenshot shows the Jenkins interface at 52.90.111.130:8080/manage/configureTools/. The page title is "Manage Jenkins > Tools". Under "Docker installations", there is a section titled "Docker" with a "Name" field containing "docker" and an "Install automatically" checkbox checked. Below it is a "Download from docker.com" section with a "Docker version" field containing "latest". At the bottom are "Save" and "Apply" buttons.

d. Add Sonarqube

1. add sonar scanner
2. name ==sonar-scanner

The screenshot shows the Jenkins management interface for tools. Under the 'SonarQube Scanner installations' section, a new configuration is being added. The 'Name' field is set to 'sonar-scanner'. The 'Install automatically' checkbox is checked. Under 'Install from Maven Central', the 'Version' dropdown is set to 'SonarQube Scanner 5.0.0.2966'. Below this, there is a 'Save' button and an 'Apply' button.

e. Add owasp dependency check

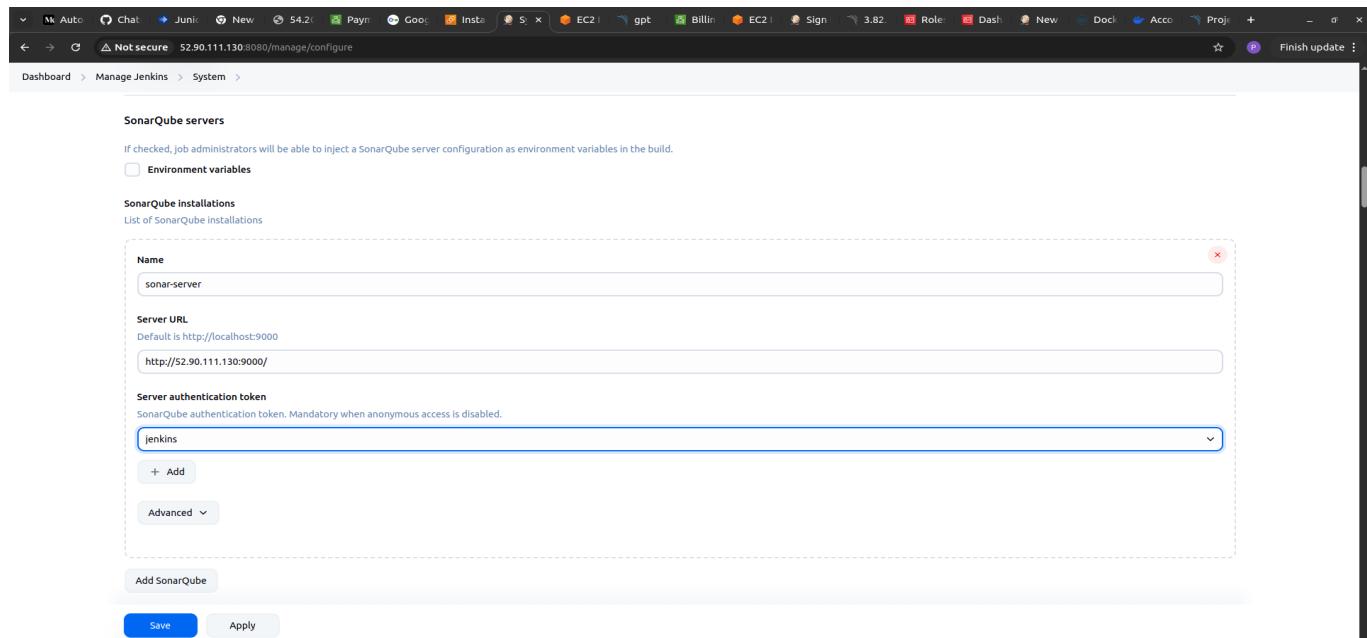
1. add dependency check
2. name == DP-Check
3. from add installer select install from github.com

The screenshot shows the Jenkins management interface for tools. Under the 'Dependency-Check installations' section, a new configuration is being added. The 'Name' field is set to 'DP-Check'. The 'Install automatically' checkbox is checked. Under 'Install from github.com', the 'Version' dropdown is set to 'dependency-check 8.4.2'. Below this, there is a 'Save' button and an 'Apply' button.

STEP 4.6 – Configure global settings for sonarqube and webhooks

jenkins → configure global setting → Add Sonarqube servers

- Jenkins → New Item
- Name: sonar-server
- server_url: http://public_ip:9000
- server authentication token: jenkins (its created in sonarqube security configurations)



Setup webhooks

In sonarqube server:

administration → configuration → webhooks

1. create webhook
2. provide and in the url section enter →
3. <http://jenkins-public-ip:8080/sonarqube-webhook/>

4. click create

The screenshot shows the SonarQube Administration settings page. The left sidebar lists various configuration categories: General Settings (selected), Configuration, Security, Projects, System, Marketplace, Analysis Scope, Authentication, DevOps Platform Integrations, External Analyzers, General, Housekeeping, JaCoCo, Languages, New Code, SCM, Security, and Technical Debt. The main content area is titled "General Settings" and contains sections for "Duplications" (with a note about cross-project duplication detection being deprecated) and "Email". Under "Email", there is a "SMTP host" field set to "smtp.gmail.com" and a "SMTP port" field set to "587". A note at the bottom of the page states: "SonarQube™ technology is powered by SonarSource SA".

The screenshot shows the SonarQube Administration webhooks page. The left sidebar shows the same navigation as the previous page. The main content area is titled "Webhooks" and contains a "Create" button. A note at the top of the page says: "Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the [Webhooks documentation](#)". Below this, it says "No webhook defined." A note at the bottom of the page states: "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." At the very bottom, it says: "SonarQube™ technology is powered by SonarSource SA. Community Edition - v9.8.8 (build 100196) - [LGPL V3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)".

STEP 4.8 – Run the pipeline in Jenkins

In Jenkins server

new item → select pipeline → type: gpt-pipeline

scroll down to the pipeline script and paste the script below

run the pipeline

```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('Checkout from Git'){
            steps{
                git branch: 'legacy', url: 'https://github.com/PearsonGrant24/Chat-gpt-deployment.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', odclInstallation: '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 #your_dockerhub_id/chatbot:latest "
                sh "docker push #your_dockerhub_id/chatbot:latest "
            }
        }
    }
}

```

```

    }

    stage("TRIVY"){
        steps{
            sh "trivy image #your_dockerhub_id/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
#your_dockerhub_id/chatbot:latest'
        }
    }
}

}

```

The Jenkins pipeline configuration page shows the Groovy script above. The 'Pipeline' tab is selected in the sidebar. The 'Advanced' button is visible at the bottom.

S
i
z
e

P
r
e
s
s
e
n
t
e
r

o
r

```
[Pipeline] stage
[Pipeline] { (TRIVY FS SCAN)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ trivy fs .
2025-07-06T18:23:38Z INFO  [vulndb] Need to update DB
2025-07-06T18:23:38Z INFO  [vulndb] Downloading vulnerability DB...
2025-07-06T18:23:38Z INFO  [vulndb] Downloading artifact... repo="mirror.gcr.io/aquasec/trivy-db:2"
26.36 MIB / 66.39 MIB [----->] 39.70% ? p/s 751.75 MIB / 66.39 MIB [----->]
> [----->] 77.94% ? p/s 766.39 MIB / 66.39 MIB [----->] 100.00% ? p/s 766.39 MIB / 66.39 MIB [----->]
-----> 100.00% 66.63 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 66.63 MIB p/s ETA 0s66.39 MIB / 66.39 MIB
ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 66.63 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 62.33 MIB p/s ETA 0s66.39 MIB / 66.39 MIB
-----> 100.00% 62.33 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 58.31 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 58.31 MIB p/s ETA 0s66.39 MIB / 66.39 MIB
-----> 100.00% 54.54 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 54.54 MIB p/s ETA 0s66.39 MIB / 66.39 MIB [----->] 100.00% 21.65 MIB p/s 3.3s2025-07-06T18:23:42Z INFO  [vulndb]
Artifact successfully downloaded! repo="mirror.gcr.io/aquasec/trivy-db:2"
2025-07-06T18:23:42Z INFO  [vuln] Vulnerability scanning is enabled
2025-07-06T18:23:42Z INFO  [secret] Secret scanning is enabled
2025-07-06T18:23:42Z INFO  [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-07-06T18:23:42Z INFO  [secret] Please see also https://trivy.dev/v0.84/docs/scanner/secret/recommendation for faster secret detection
2025-07-06T18:23:42Z WARN  [secret] The size of the scanned file is too large. It is recommended to use '-skip-files' for this file to avoid high memory consumption.
file_path="dependency-check-report.xml" size (MB)=12
```

• • •

REST API Jenkins 2.504.3

C
l
i
k
t
o

V
i
e
w

i
m
a
g
e

f
u
l
l

s
i
z
e

Jenkins

Dashboard > gpt-jenkins >

Status: gpt-jenkins

SonarQube Quality Gate

- Chatbot: Passed
- server-side processing: Success

Latest Dependency-Check

Permalinks

- Last build (#6), 15 min ago
- Last stable build (#6), 15 min ago
- Last successful build (#6), 15 min ago
- Last failed build (#5), 25 min ago
- Last unsuccessful build (#5), 25 min ago
- Last completed build (#6), 15 min ago

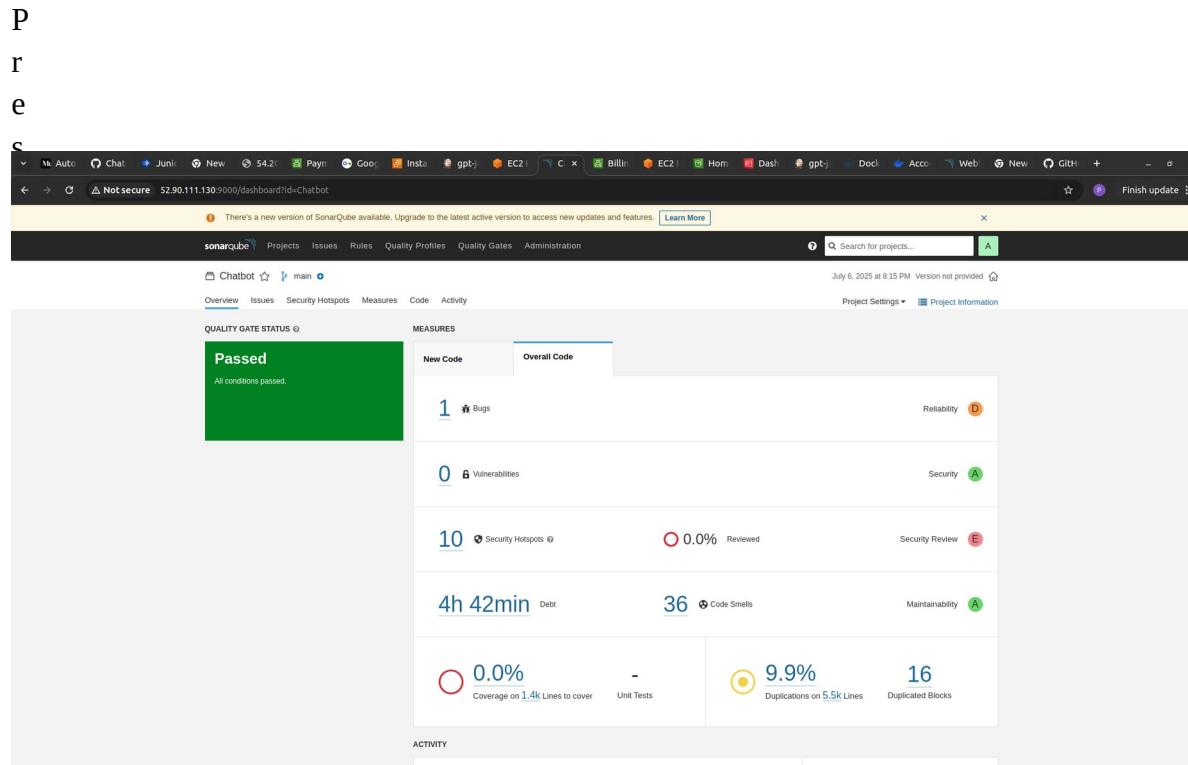
Builds

Build	Started	Duration
#6	18:13	15 min ago
#5	18:04	25 min ago
#4	18:03	25 min ago
#3	18:03	25 min ago
#2	18:01	25 min ago
#1	17:59	25 min ago

Today

Builds: #6 18:13, #5 18:04, #4 18:03, #3 18:03, #2 18:01, #1 17:59

REST API Jenkins 2.504.3



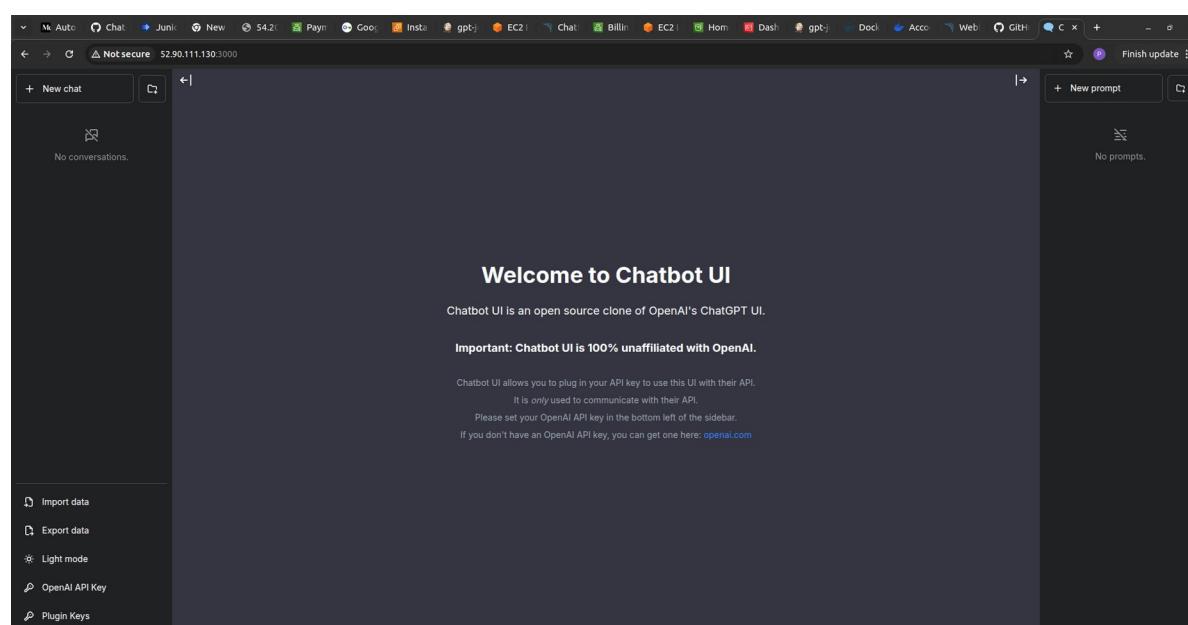
C
k

t
o

Application is successfully deployed

Checkout your dockerhub and to access your application

Go to your browser and type



[
]
]

[
]
]
]

STEP 4.8 – Get API to connect your app to Openai

- Click or search: openai.com
- Click: create a new secret access key
- Provide: name and click on create new secret key
- Copy the secret access key and comeback to your application
- Click: on openai api key present on bottom left corner and paste your api key and now your application is ready to be used

The screenshot shows the OpenAI Platform interface. On the left, there's a sidebar with various project management and AI-related tabs like Dashboard, Prompts, Logs, Traces, Assistants, Batches, Evaluations, Fine-tuning, Storage, Usage, and API keys. The API keys tab is selected. In the main content area, there's a table of existing API keys, one of which is named "my-gpt". A modal window titled "Save your key" is open over the table. It contains instructions to save the key securely, a note about OpenAI's security practices, and a "copy" button next to the key value "sk-proj-eDSgn4CXk6PKYJw7siunQaGgw". Below the key is a "Permissions" section with the text "Read and write API resources". At the bottom of the modal is a "Done" button.

The screenshot shows the Chatbot UI interface. At the top, there's a header with a "New chat" button, a search bar, and a "New prompt" button. The main area is titled "Chatbot UI" and features a "Model" dropdown set to "Default (GPT-3.5)". Below the model selection is a "View Account Usage" link and a "System Prompt" box containing the text: "You are ChatGPT, a large language model trained by OpenAI. Follow the user's instructions carefully. Respond using markdown." On the left side, there's a sidebar with options for "Import data", "Export data", "Light mode", "OpenAI API Key", and "Plugin Keys". At the bottom, there's a message input field with the placeholder "Type a message or type '/' to select a prompt..." and a note: "ChatBot UI: Chatbot UI is an advanced chatbot kit for OpenAI's chat models aiming to mimic ChatGPT's interface and functionality."

P
r
e
s

PHASE 5 – Kubernetes cluster

e
n
t
e
r

Step 5.1 — What is Kubernetes (Simple Explanation)

r
Think of Kubernetes as:

- **Manager** → schedules apps
- **Pods** → smallest runnable unit
- **Nodes** → EC2 machines
- **Deployment** → desired state
- **Service** → network access

Y
ou already have:

- k
 - EKS control plane (AWS-managed)

t
o

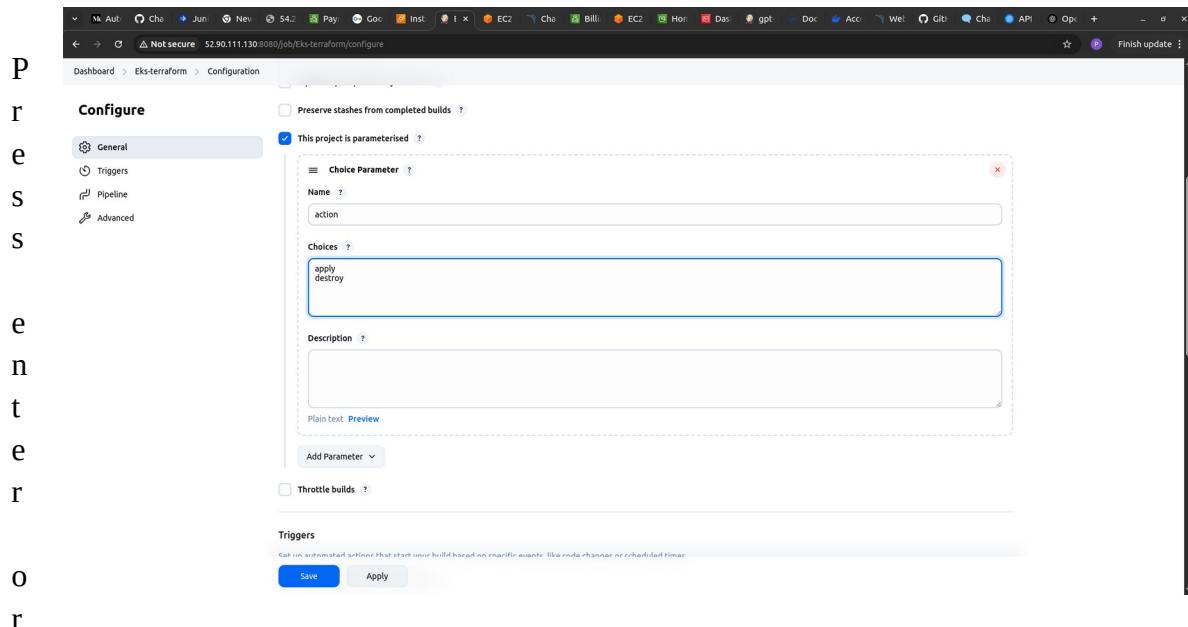
STEP 5.2 — Creating kubernetes cluster

- v
i
e
w
 - create a new pipeline named as eks-terraform
 - scroll down and select this project as parameterized
 - choice: apply and destroy

i
n

f
u
l
l

s
i
z
e



C
3. scroll down to the pipeline script and paste the below pipeline script:

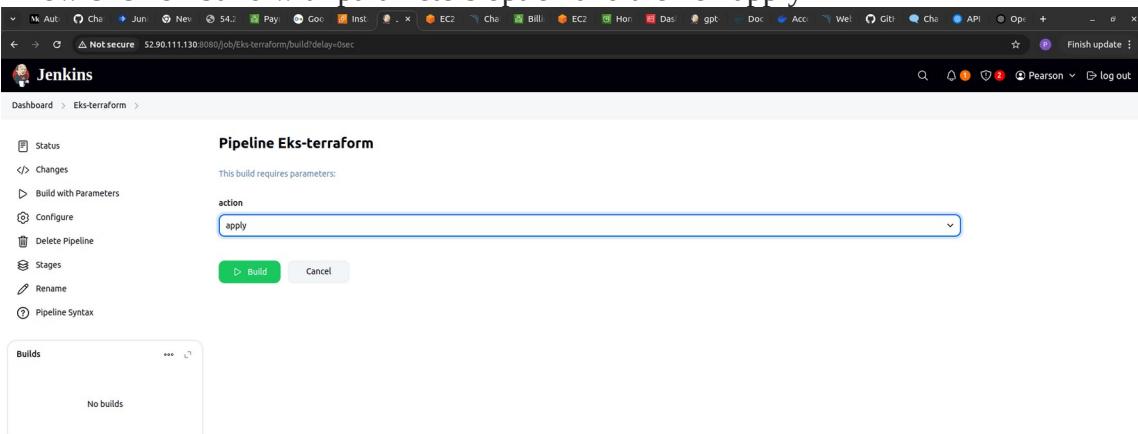
```
t    pipeline{
o        agent any
m        stages {
v            stage('Checkout from Git'){
i                steps{
e                    git branch: 'legacy', url:
w                    'https://github.com/PearsonGrant24/Chat-gpt-deployment.git'
i                }
m            stage('Terraform version'){
a                steps{
g                    sh 'terraform --version'
e                }
m            stage('Terraform init'){
i                steps{
n                    dir('Eks-terraform') {
f                        sh 'terraform init'
u                }
l            stage('Terraform validate'){


```

s
i
z
e

```
P  
r  
e  
s  
s  
e  
n  
t  
e  
r  
o  
r  
c  
l  
i  
c  
_  
} 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'  
        }  
    }  
}  
}
```

- click on apply and then save
 - now click on build with parameters option and then on apply



```
P  
r  
e  
s  
s  
  
e  
n  
t  
e  
r  
  
o  
[  ]  ↗  ⚡ Not secure 52.90.111.130:8080/job/Eks-terraform/12/console  
Dashboard > Eks-terraform > #12  
  
[1m[1maws iam user: Refreshing state... [id=eks-user][0m][0m  
[0m[1maws iam policy document_assume_role: Reading...[0m][0m  
[0m[1maws iam role.example: Refreshing state... [id=eks-node-group-cloud][0m  
[0m[1maws iam policy document_assume_role: Read complete after 0s [id=3552664922][0m  
[0m[1maws iam role.example: Refreshing state... [id=eks-cluster-cloud][0m  
[0m[1maws iam role policy attachment.example:AmazonEKSTaskWorkerNodePolicy: Refreshing state... [id=eks-node-group-cloud-20250706174858761200000002][0m  
[0m[1maws iam role policy attachment.example:AmazonECSContainerRegistryReadOnly: Refreshing state... [id=eks-node-group-cloud-20250706174859102200000003][0m  
[0m[1maws iam role policy attachment.example:AmazonEKS_CNI_Policy: Refreshing state... [id=eks-node-group-cloud-20250706174859510700000004][0m  
[0m[1maws iam role policy attachment.example:AmazonEKSClusterPolicy: Refreshing state... [id=eks-cluster-cloud-20250706174859837000000001][0m  
[0m[1mdata.aws_vpc.default: Read complete after 1s [id=vpc-00536421e9fe1eb][0m  
[0m[1mdata.aws_subnets.public: Read complete after 1s [id=ap-south-1][0m  
[0m[1maws_ebs_cluster.example: Refreshing state... [id=EKS_CLOUD][0m  
[0m[1maws_eks_node_group.example: Refreshing state... [id=EKS_CLOUD:Node-cloud][0m  
  
[0m[1m[32mNo changes.[0m[1m Your infrastructure matches the configuration.[0m  
  
[0mTerraform has compared your real infrastructure against your configuration  
and found no differences, so no changes are needed.  
[0m[1m[32m  
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.  
[0m  
[Pipeline]   
[Pipeline] // dir  
[Pipeline]   
[Pipeline] // withCredentials  
[Pipeline]   
[Pipeline] // stage  
[Pipeline]   
[Pipeline] // node  
[Pipeline] End of Pipeline  
Finished: SUCCESS  
  
REST API Jenkins 2.504.3
```

0 REST API Jenkins 2.504.3

C C
l l
j j

c c Its gonna take some time (like 5 – 10 minutes):

k k : Once its done, you will see :

The screenshot shows the AWS Amazon Elastic Kubernetes Service (EKS) Clusters page. The left sidebar includes navigation links for Dashboard, Clusters (selected), Settings, Amazon EKS Anywhere, and Related services (Amazon ECR, AWS Batch). The main content area displays a table titled "Clusters (1) Info" with one entry: "EKS_CLOUD" (Status: Active, Kubernetes version: 1.33, Support period: Standard support until July 29, 2026, Upgrade policy: Extended, Created: 3 hours ago, Provider: EKS). A "Delete" button is available for the cluster.

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provider
EKS_CLOUD	Active	1.33	Standard support until July 29, 2026	Extended	3 hours ago	EKS

i 0 / 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookies preferences

f f
u u
l l

8. check for the node groups → go to your ec2 instances

P
r
e
s

P e
r n
e t
s e
s r

e o **STEP 5.3 — Deploy on kubernetes**
n r
t e
r l

- go to your gpt instance ,
i . Run this (replace cluster name if needed):

o c
r aws eks update-kubeconfig --region <ap-south-1> \
c --name <clustername>

c l This creates:

i ~/.kube/config

```
ubuntu@ip-172-31-86-211:~$ aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1  
Added new context arn:aws:eks:ap-south-1:120569617612:cluster/EKS_CLOUD to /home/ubuntu/.kube/config  
ubuntu@ip-172-31-86-211:~$
```

0 1

v

i

Verify Connection

kubectl get nodes

i Expected:

NAME	STATUS	ROLES	AGE	VERSION
ip-192-168-x-x.ec2.internal	Ready	<none>	5m	v1.xx

Clone the repository on your ec2

Go to k8s folder

f

u

l

l

s

i

z

e

```

ubuntu@ip-172-31-86-211:~$ aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
Added new context arn:aws:eks:ap-south-1::cluster/EKS_CLOUD to /home/ubuntu/.kube/config
ubuntu@ip-172-31-86-211:~$ git clone https://github.com/PearsonGrant24/Chat-gpt-clone-deployment.git
Cloning into 'Chat-gpt-clone-deployment'...
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (16/16), done.
remote: Writing objects: 100% (29/29), done.
Receiving objects: 100% (2037/2037) 1.84 MiB | 26.57 MiB/s, done.
Resolving deltas: 100% (1000/1000), done.
ubuntu@ip-172-31-86-211:~/Chat-gpt-clone-deployment$ ls
CONTRIBUTING.md  Instance-terraform  tests  docs  next-i18nnext.config.js  package.json  prettier.config.js  tailwind.config.js  utils
Dockerfile        Makefile          tests  docs  next.config.js    pages      public      tsconfig.json  vite-test.config.ts
Eks-Terraform    README.md        docker-compose.yml  license  package-lock.json  postcss.config.js  styles      types
ubuntu@ip-172-31-86-211:~/Chat-gpt-clone-deployment$ cd Eks-Terraform
ubuntu@ip-172-31-86-211:~/Chat-gpt-clone-deployment$ ls
CONTRIBUTING.md  Instance-terraform  tests  docs  next-i18nnext.config.js  package.json  prettier.config.js  tailwind.config.js  utils
Dockerfile        Makefile          tests  docs  next.config.js    pages      public      tsconfig.json  vite-test.config.ts
Eks-Terraform    README.md        docker-compose.yml  license  package-lock.json  postcss.config.js  styles      types
ubuntu@ip-172-31-86-211:~/Chat-gpt-clone-deployment$ cd k8s
ubuntu@ip-172-31-86-211:~/Chat-gpt-clone-deployment/k8s$ ls

```

i-016ade854064f2d9d (gpt)
PublicIPs: 52.90.111.130 PrivateIPs: 172.31.86.211
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Expected:

You will find the deployment file for Kubernetes chatbot
run the following command to deploy application

```
kubectl apply -f chatbot-ui.yaml

kubectl get all
```

```

contributing.md  Dockerfile  components  next-linter-config.js  package.json  prettier.config.js  tailwind.config.js  utils
docs          README.md  .gitignore  docker-compose.yml  license  package-lock.json  postcss.config.js  styles
infra        terraform  .gitmodules  next-next-config.js  package-lock.json  postcss.config.js  scss/globals.json  types
main          .gitignore  .travis.yml  package.json  public  public  scss/globals.json  types
scripts      .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
src          .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
tests        .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
utils        .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
  
```

```

ubuntu@ip-172-31-86-211:~/Chat-GPT-clone-deployment$ ls
  
```

```

CONTRIBUTING.md  Dockerfile  README.md  .gitignore  .travis.yml  .travis.yml  .travis.yml  .travis.yml  .travis.yml  .travis.yml
docs            Dockerfile  README.md  .gitignore  .travis.yml  .travis.yml  .travis.yml  .travis.yml  .travis.yml  .travis.yml
infra          terraform  .gitmodules  next-linter-config.js  package.json  prettier.config.js  tailwind.config.js  utils
main           .gitignore  .travis.yml  next-next-config.js  package-lock.json  postcss.config.js  styles
scripts        .travis.yml  .travis.yml  package.json  public  public  scss/globals.json  types
src            .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
tests          .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
utils          .travis.yml  .travis.yml  package-lock.json  public  public  scss/globals.json  types
  
```

```

ubuntu@ip-172-31-86-211:~/Chat-GPT-clone-deployment$ kubectl get all
  
```

```

NAME                                     TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/chatbot-service   ClusterIP  10.100.0.1    <none>        80/TCP   17m
deployment.apps/chatbot     READY   1/1   1/1   0/0   32s
  
```

```

ubuntu@ip-172-31-86-211:~/Chat-GPT-clone-deployment$ kubectl get svc
  
```

```

NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/chatbot-service   ClusterIP  10.100.0.1    <none>        80/TCP   17m
  
```

```

ubuntu@ip-172-31-86-211:~/Chat-GPT-clone-deployment$ kubectl get pods
  
```

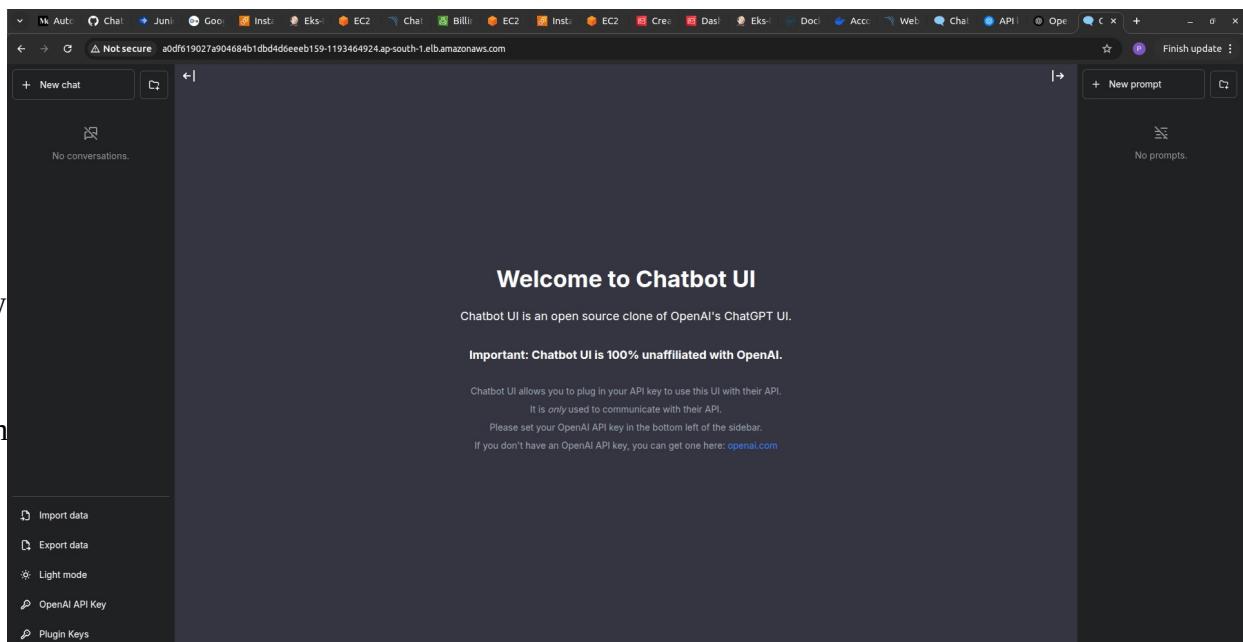
```

NAME          READY   STATUS    RESTARTS   AGE
chatbot-66f69b8d55-z876b   0/1     Pending   0          32s
  
```

STEP 3.5 – Expose the App

- o c r l Copy the load balancer external ip
- i i c c Open browser: paste external ip

YOUR CHATGPT APP IS LIVE ON EKS



Load Balancer Ingress →

It is a mechanism that helps distribute incoming internet traffic among multiple servers or services, ensuring efficient and reliable delivery of requests.

It's like having a receptionist at a busy office building entrance who guides visitors to different floors or departments, preventing overcrowding at any one location. In the digital world, a Load Balancer Ingress helps maintain a smooth user experience, improves application performance, and ensures that no single server becomes overwhelmed with too much traffic.

Service.yaml

`service.yaml` file is like a set of rules that helps computers find and talk to each other within a software application. It's like a directory that says, "Hey, this is how you can reach different parts of our application." It specifies how different parts of your application communicate and how other services or users can connect to them.

PHASE 6 – Prometheus And Grafana on EKS

This is the **final technical phase**.

- **Prometheus** is like a detective that constantly watches your software and gathers data about how it's performing. It's good at collecting metrics, like how fast your software is running or how many users are visiting your website.
- **Grafana**, on the other hand, is like a dashboard designer. It takes all the data collected by Prometheus and turns it into easy-to-read charts and graphs. This helps you see how well your software is doing at a glance and spot any problems quickly.

In other words, Prometheus collects the information, and Grafana makes it look pretty and understandable so you can make decisions about your software. They're often used together to monitor and manage applications and infrastructure.

STEP 6.1 – Use EC2 Server for monitoring

- Go to ec2 your console and launch an instance having a base image of Ubuntu and with t2.medium specs because **Minimum Requirements to Install Prometheus :**

Expected:

- 2 CPU cores.
 - 4 GB of memory.
 - 20 GB of free disk space.
-

STEP 6.2 — Install Prometheus

- Paste the following script to your ec2 instance

```
sudo useradd -- system -- no-create-home -- shell /bin/false prometheus  
  
wget  
https://github.com/prometheus/prometheus/releases/download/v2.47.1/  
prometheus-2.47.1.linux-amd64.tar.gz
```

- Extract Prometheus files, move them, and create directories:

```
tar -xvf prometheus-2.47.1.linux-amd64.tar.gz  
  
cd prometheus-2.47.1.linux-amd64/  
  
sudo mkdir -p /data /etc/prometheus  
  
sudo mv prometheus promtool /usr/local/bin/  
  
sudo mv consoles/ console_libraries/ /etc/prometheus/  
  
sudo mv prometheus.yml /etc/prometheus/prometheus.yml
```

- Set ownership for directories:

```
useradd prometheus  
  
sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
```

- Create a systemd unit configuration file for Prometheus:

```
sudo nano /etc/systemd/system/prometheus.service
```

- Add the following code to the prometheus.servicefile:

```
[Unit]
```

```
Description=Prometheus
```

```
Wants=network-online.target
```

```
After=network-online.target
```

```
StartLimitIntervalSec=500
```

```
StartLimitBurst=5[Service]
```

```
User=prometheus
```

```
Group=prometheus
```

```
Type=simple
```

```
Restart=on-failure
```

```
RestartSec=5s
```

```
ExecStart=/usr/local/bin/prometheus \
```

```
    --config.file=/etc/prometheus/prometheus.yml \
```

```
    --storage.tsdb.path=/data \
```

```
    --web.console.templates=/etc/prometheus/consoles \
```

```
    --web.console.libraries=/etc/prometheus/console_libraries \
```

```
--web.listen-address=0.0.0.0:9090 \
--web.enable-lifecycle[Install]
WantedBy=multi-user.target
```

press →**ctrl+o** #for save and then **ctrl+x** #for exit from the file

STEP 6.3 — Enable and start Prometheus

```
sudo systemctl enable prometheus
sudo systemctl start prometheus
sudo systemctl status prometheus
```

```

drwxr-xr-x 2 prometheus 4999 Jul  6 20:58 /data
drwxr-xr-x 4 prometheus prometheus 4096 Jul  6 20:58 /etc/prometheus
ubuntu@ip-172-31-82-64:~/prometheus$ sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: failed (Result: exit-code) since Sun 2025-07-06 21:21:39 UTC; 9min ago
       Process: 11037 ExecStart=/usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/data --web.console.templates=/etc/prometheus/consoles --web.console.libraries=/etc/prometheus/console_libraries --web.console.templates=/etc/prometheus/consoles --web.console.libraries=/etc/prometheus/console_libraries
      Main PID: 11037 (code=exited, status=217/USER)
        CPU: 673us

Jul 06 21:21:39 ip-172-31-82-64 systemd[1]: prometheus.service: Scheduled restart job, restart counter is at 5.
Jul 06 21:21:39 ip-172-31-82-64 systemd[1]: Stopped Prometheus.
Jul 06 21:21:39 ip-172-31-82-64 systemd[1]: prometheus.service: Start request repeated too quickly.
Jul 06 21:21:39 ip-172-31-82-64 systemd[1]: prometheus.service: Failed with result 'exit-code'.
Jul 06 21:21:39 ip-172-31-82-64 systemd[1]: Failed to start Prometheus.

ubuntu@ip-172-31-82-64:~/prometheus$ # Copy the corrected service file
sudo nano /etc/systemd/system/prometheus.service
# Paste the corrected contents, save and exit

# Reload systemd to recognize the new service
sudo systemctl daemon-reexec
sudo systemctl daemon-reload

# Enable the service on boot
sudo systemctl enable prometheus

# Start it now
sudo systemctl start prometheus

# Check status
sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-07-06 21:32:32 UTC; 19ms ago
       Main PID: 11538 (prometheus)
          Tasks: 4 (limit: 4674)
         Memory: 4.6M
            CPU: 15ms
           CGroup: /system.slice/prometheus.service
                   └─11538 /usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/data --web.console.templates=/etc/prometheus/consoles --web.console.libraries=/etc/prometheus/console_libraries --web.console.templates=/etc/prometheus/consoles --web.console.libraries=/etc/prometheus/console_libraries

Jul 06 21:32:32 ip-172-31-82-64 systemd[1]: Started Prometheus.
Lines 1-11/11 (END)

```

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Copy public ip of the EC2 your prometheus is running at

go to → http://public_ip:9090 to see the webpage of Prometheus

The screenshot shows the Prometheus web interface with the following details:

- URL:** http://18.212.199.0:9090/graph?g.expr=g0.tab=1&g0.stackd=0&g0.show_exemplars=0&g0.range_input=1h
- Panel Type:** Graph
- Query:** Expression (press Shift+Enter for newlines)
- Buttons:** Use local time, Enable query history, Enable autocomplete, Enable highlighting, Enable linter, Execute
- Table/Graph Selection:** Table
- Time Range:** Evaluation time
- Message:** No data queried yet
- Buttons:** Add Panel, Remove Panel

STEP 6.1 — Install Node Exporter

What Node Exporter Does

- Runs on **every node**

- Collects:

- CPU
- Memory
- Disk
- Network

How Node Exporter is Deployed

Node Exporter is usually:

- A **DaemonSet**

- One pod per node

Create a system user for Node Exporter and download Node Exporter:

```
sudo useradd -- system -- no-create-home -- shell /bin/false  
node_exporter  
  
wget  
https://github.com/prometheus/node\_exporter/releases/download/v1.6.1/  
node_exporter-1.6.1.linux-amd64.tar.gz
```

Extract Node Exporter files, move the binary, and clean up:

```
tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz  
sudo mv node_exporter-1.6.1.linux-amd64/node_exporter /usr/local/bin/  
rm -rf node_exporter*
```

Create a systemd unit configuration file for Node Exporter:

```
sudo nano /etc/systemd/system/node_exporter.service
```

Add the following code to the node_exporter.service file:

```
[Unit]
Description=Node Exporter
After=network.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=default.target
```

Enable and start Node Exporter

```
sudo useradd -m -s /bin/bash node_exporter
sudo chown node_exporter:node_exporter /usr/local/bin/node_exporter
```

```

sudo systemctl daemon-reload

sudo systemctl start node_exporter

sudo systemctl enable node_exporter

sudo systemctl status node_exporter

```

Node Exporter service is now running

```

ubuntu@ip-172-31-82-64:~$ sudo mv node_exporter-1.6.1.linux-amd64/node_exporter /usr/local/bin/
ubuntu@ip-172-31-82-64:~$ rm -rf node_exporter*
ubuntu@ip-172-31-82-64:~$ node_exporter-1.6.1.linux-amd64/ --version
node_exporter-1.6.1.linux-amd64/node_exporter
node_exporter-1.6.1.linux-amd64/LICENSE
ubuntu@ip-172-31-82-64:~$ /prometheus-2.47.1.linux-amd64$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
ubuntu@ip-172-31-82-64:~$ cp node_exporter /etc/systemd/system/node_exporter.service
ubuntu@ip-172-31-82-64:~$ sudo chmod +x /bin/bash node_exporter
ubuntu@ip-172-31-82-64:~$ /prometheus-2.47.1.linux-amd64$ sudo chown node_exporter:node_exporter /usr/local/bin/node_exporter
ubuntu@ip-172-31-82-64:~$ /prometheus-2.47.1.linux-amd64$ sudo systemctl daemon-reload
ubuntu@ip-172-31-82-64:~$ sudo systemctl start node_exporter
ubuntu@ip-172-31-82-64:~$ sudo systemctl enable node_exporter
ubuntu@ip-172-31-82-64:~$ sudo systemctl status node_exporter
Created symlink /etc/systemd/system/multi-user.target.wants/node_exporter.service → /etc/systemd/system/node_exporter.service.
● node_exporter.service - node exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-07-06 21:40:50 UTC; 247ms ago
       PID: 11883 (node_exporter)
      Tasks: 5 (limit: 4674)
     Memory: 1.6M
        CPU: 8ms
       CGroup: /system.slice/node_exporter.service
               └─11883 /usr/local/bin/node_exporter

Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=thermal_zone
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=time
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=timevar
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=queues
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=uname
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=vmstat
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=cpu
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=node_exporter.go:117 level=info collector=mem
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.492Z caller=tls_config.go:274 level=info msg="listening on" address=[::]:9100
Jul 06 21:40:50 ip-172-31-82-64 node_exporter[11883]: ts=2025-07-06T21:40:50.493Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9100
ubuntu@ip-172-31-82-64:~$ /prometheus-2.47.1.linux-amd64

```

Check metrics

Node Exporter metrics is now accessible in Prometheus.

```

Not secure 18.12.199.0:5100/metrics
# HELP gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0
go_gc_duration_seconds{quantile="2"} 0
go_gc_duration_seconds count 0
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 9
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.20.6"} 1
# HELP memstats_heap_bytes Number of bytes allocated and still in use.
# TYPE memstats_heap_bytes gauge
go_memstats_heap_bytes 946536
# HELP memstats_heap_bytes_total Total number of bytes allocated, even if freed.
# TYPE memstats_heap_bytes_total counter
go_memstats_heap_bytes_total 946536
# HELP memstats_heap_bytes_sys Number of bytes used by the profiling bucket hash table.
# TYPE memstats_heap_bucket_hash_sys_bytes gauge
go_memstats_heap_bucket_hash_sys_bytes 1.44508e+06
# HELP memstats_frees_total Total number of frees.
# TYPE memstats_frees_total counter
go_memstats_frees_total 719
# HELP memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 7.196104e+06
# HELP memstats_heap_allocated Number of heap bytes allocated and still in use.
# TYPE memstats_heap_allocated counter
go_memstats_heap_allocated 946536
# HELP memstats_heap_head_alloc_bytes_gauge Number of heap bytes allocated and still in use.
# TYPE memstats_heap_head_alloc_bytes_gauge
go_memstats_heap_head_alloc_bytes 946536
# HELP memstats_heap_idles_bytes_gauge Number of heap bytes waiting to be used.
# TYPE memstats_heap_idles_bytes_gauge
go_memstats_heap_idles_bytes 1.605432e+06
# HELP memstats_heap_inuse_bytes_gauge Number of heap bytes that are in use.
# TYPE memstats_heap_inuse_bytes_gauge
go_memstats_heap_inuse_bytes 2.269992e+06
# HELP memstats_heap_objects_gauge Number of heap objects.
# TYPE memstats_heap_objects_gauge
go_memstats_heap_objects 8493
# HELP memstats_heap_released_bytes_gauge Number of heap bytes released to OS.
# TYPE memstats_heap_released_bytes_gauge
go_memstats_heap_released_bytes 1.605632e+06
# HELP memstats_heap_stale_bytes_gauge Number of heap bytes obtained from system.
# TYPE memstats_heap_stale_bytes_gauge
go_memstats_heap_stale_bytes 0
# HELP memstats_mallocs_total Total number of mallocs.
# TYPE memstats_mallocs_total counter
go_memstats_mallocs_total 9212
# HELP memstats_mcache_inuse_bytes_gauge Number of bytes in use by mcache structures.
# TYPE memstats_mcache_inuse_bytes_gauge
go_memstats_mcache_inuse_bytes 1200

```

STEP 6.3 — Configure Prometheus Plugin Integration:

- go to your EC2 and run →

```
cd /etc/prometheus
```

- you have to edit the prometheus.yml file to monitor anything
- add the code with proper indentation like this:

```
scrape_configs:

  - job_name: 'node_exporter'

static_configs:

  - targets: ['localhost:9100']

  - job_name: 'jenkins'

metrics_path: '/prometheus'

static_configs:

  - targets: ['<your-jenkins-ip>:<your-jenkins-port>']
```

press esc+:wq to save and exit

Check the validity of the configuration file

```
promtool check config /etc/prometheus/prometheus.yml
```

o/p : success

Reload the Prometheus configuration without restarting

```
curl -X POST http://localhost:9090/-/reload
```

Prometheus -> status - targets

you will find three targets present as we entered in the yaml file for monitoring

Prometheus targets dashboard

STEP 6.3 — Setup Grafana

Install Dependencies:

```
sudo apt-get update
```

```
sudo apt-get install -y apt-transport-https software-properties-common
```

Add the GPG Key for Grafana:

```
wget -q -O — https://packages.grafana.com/gpg.key | sudo apt-key add -
```

Add the repository for Grafana stable releases:

```
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee  
— a /etc/apt/sources.list.d/grafana.list
```

```
apt-get update  
(Reading database ... 88981 files and directories currently installed.)  
Preparing to unpack .../grafana_12.0.2_amd64.deb ...  
Unpacking grafana (12.0.2) ...  
Selecting previously unselected package grafana.  
Preparing to unpack .../grafana_12.0.2_amd64.deb ...  
Unpacking grafana (12.0.2) ...  
Setting up grafana (12.0.2) ...  
Adding system user 'grafana' (UID 115)  
Adding system user 'grafana' (UID 115) with group 'grafana' ...  
Not creating home directory '/usr/share/grafana'.  
#### NOT starting on installation, please execute the following statements to configure grafana to start automatically using systemd  
sudo systemctl daemon-reload  
sudo /bin/systemctl enable grafana-server  
#### You can start grafana-server by executing  
sudo /bin/systemctl start grafana-server  
Processing triggers for man-db (2.10.2-1) ...  
Done.  
Scanning linux images...  
Running kernel seems to be up-to-date.  
No services need to be restarted.  
No containers need to be restarted.  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
ubuntu@ip-172-31-82-64:~$ sudo systemctl enable grafana-server  
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemd-sysv-install.  
Executing: /lib/systemd/systemd-sysv-install enable grafana-server  
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /lib/systemd/system/grafana-server.service.  
ubuntu@ip-172-31-82-64:~$
```

Update the package list, install and start Grafana:

```
sudo apt-get update
```

```
sudo apt-get -y install grafana
```

```
sudo systemctl enable grafana-server
```

```
sudo systemctl start grafana-server
```

```
sudo systemctl status grafana-server
```

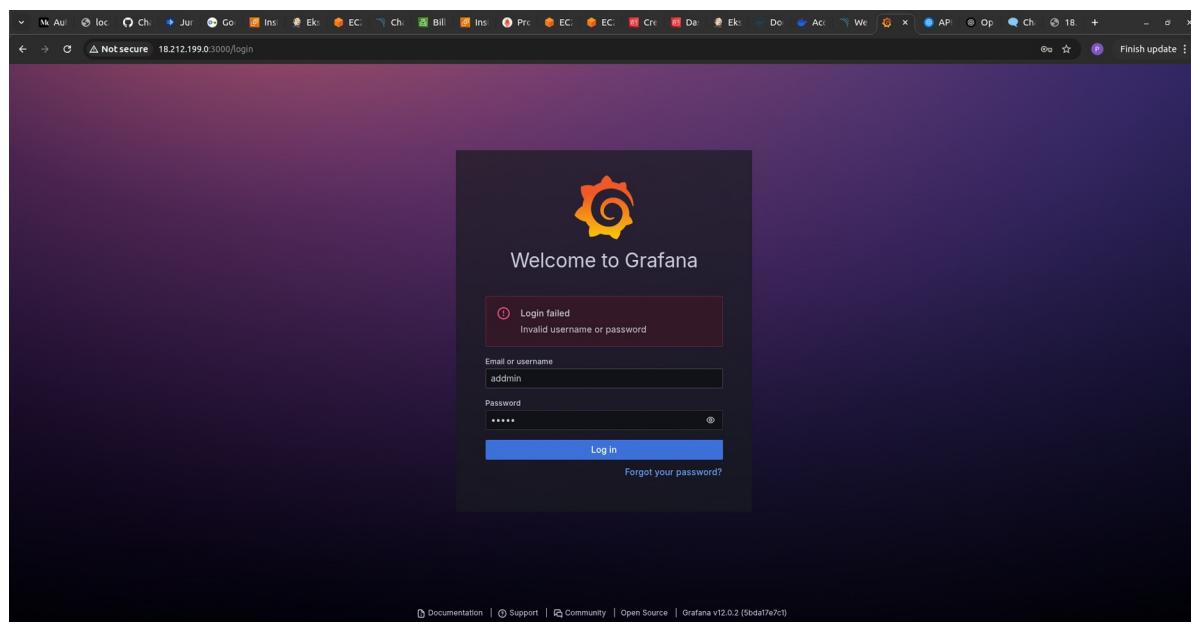
```
No containers need to be restarted.  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (Qemu) binaries on this host.  
ubuntu@ip-172-31-82-64:~$ sudo systemctl enable grafana-server  
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemv-install.  
Executing: /lib/systemd/systemv-install enable grafana-server  
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service -- /lib/systemd/system/grafana-server.service.  
ubuntu@ip-172-31-82-64:~$ sudo systemctl start grafana-server  
ubuntu@ip-172-31-82-64:~$ sudo systemctl status grafana-server  
● grafana-server.service - Grafana instance  
   Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)  
     Active: active (running) since Sun 2025-07-06 22:32:57 UTC; 14s ago  
       Docs: http://docs.grafana.org  
 Main PID: 14893 (grafana)  
    Tasks: 15 (limit: 4074)  
   Memory: 129.3M  
      CPU: 4.790s  
 CGroup: /system.slice/grafana-server.service  
        └─14893 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packaging=deb cfg=default.paths.logs=/var/log/grafana cfg=default.paths[  
  
Jul 06 22:33:04 ip-172-31-82-64 grafana[14893]: logger=plugin.backgroundinstaller t=2025-07-06T22:33:04.963827444Z level=info msg="Installing plugin" pluginId=grafana-exploretraces-app version=1.0.0  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=installer.fs t=2025-07-06T22:33:05.171336892Z level=info msg="Downloaded and extracted grafana-exploretraces-app v1.1.1 successfully to /var/lib/grafana"  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=plugins.registration t=2025-07-06T22:33:05.214029471Z level=info msg="Plugin registered" pluginId=grafana-exploretraces-app  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=plugin.backgroundinstaller t=2025-07-06T22:33:05.214095967Z level=info msg="Plugin successfully installed" pluginId=grafana-exploretraces-app version=durable  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=plugin.backgroundinstaller t=2025-07-06T22:33:05.214095967Z level=info msg="Plugin successfully installed" pluginId=grafana-metricsdrilldown-app version=durable  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=installer.fs t=2025-07-06T22:33:05.395441352Z level=info msg="Downloaded and extracted grafana-metricsdrilldown-app v1.0.4 zip successfully to /var/lib/grafana"  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=plugins.registration t=2025-07-06T22:33:05.437533882Z level=info msg="Plugin registered" pluginId=grafana-metricsdrilldown-app  
Jul 06 22:33:05 ip-172-31-82-64 grafana[14893]: logger=plugin.backgroundinstaller t=2025-07-06T22:33:05.437594797Z level=info msg="Plugin successfully installed" pluginId=grafana-metricsdrilldown-app version=durable  
TIME 1-27/21 (END)
```

Open Grafana on port 3000

`http://<public_ip>:3000`

username: admin

password: admin

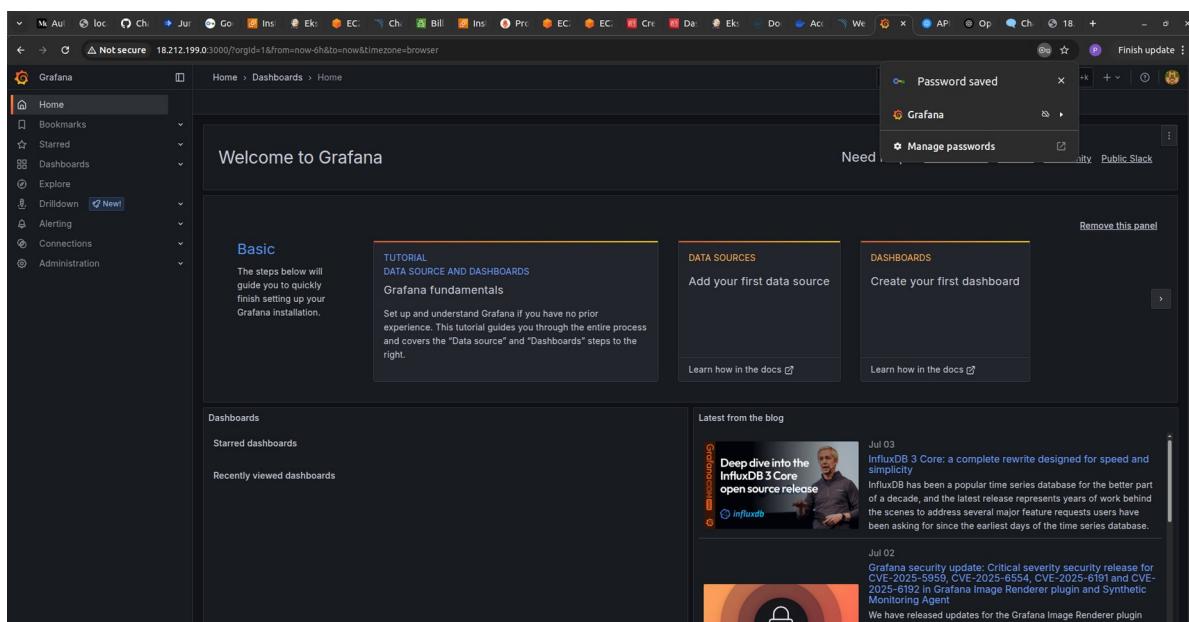


STEP 6.3 — Add Prometheus as Data Source

To visualize metrics, you need to add a data source.

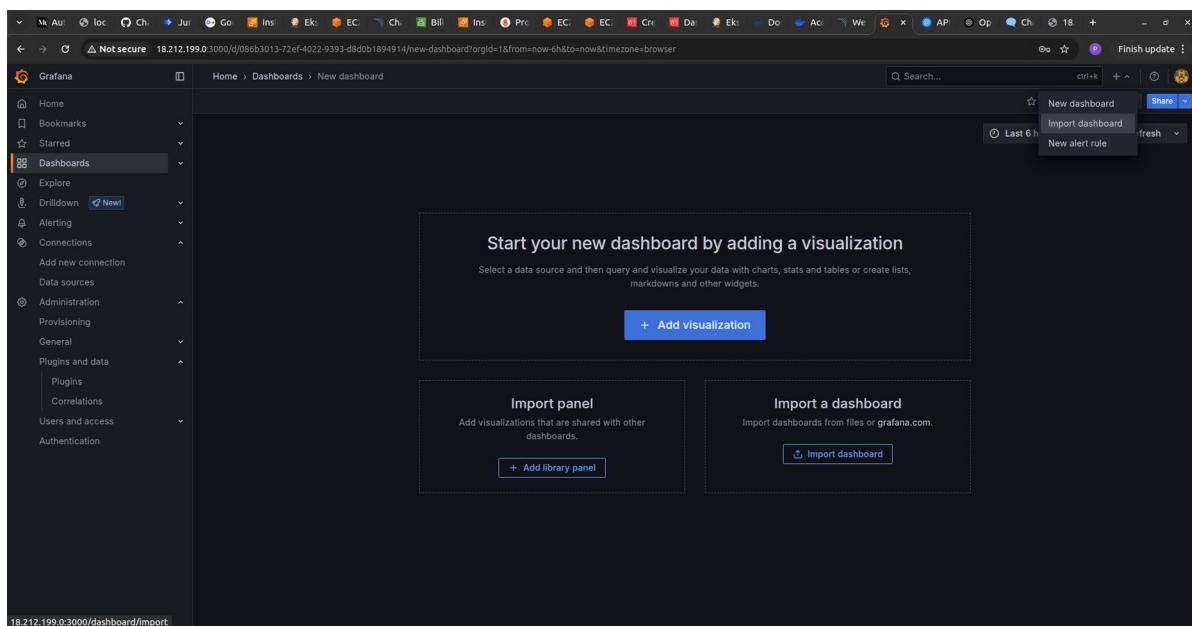
Follow these steps:

- Click on the gear icon () in the left sidebar to open the “Configuration” menu.
- Select “Data Sources.”
- Click on the “Add data source” button.



- Type Prometheus
- URL:
 - <http://localhost:9090>(assuming Prometheus is running on the same server).
- Save & Test

STEP 6.5 — Import Node Exporter Dashboard



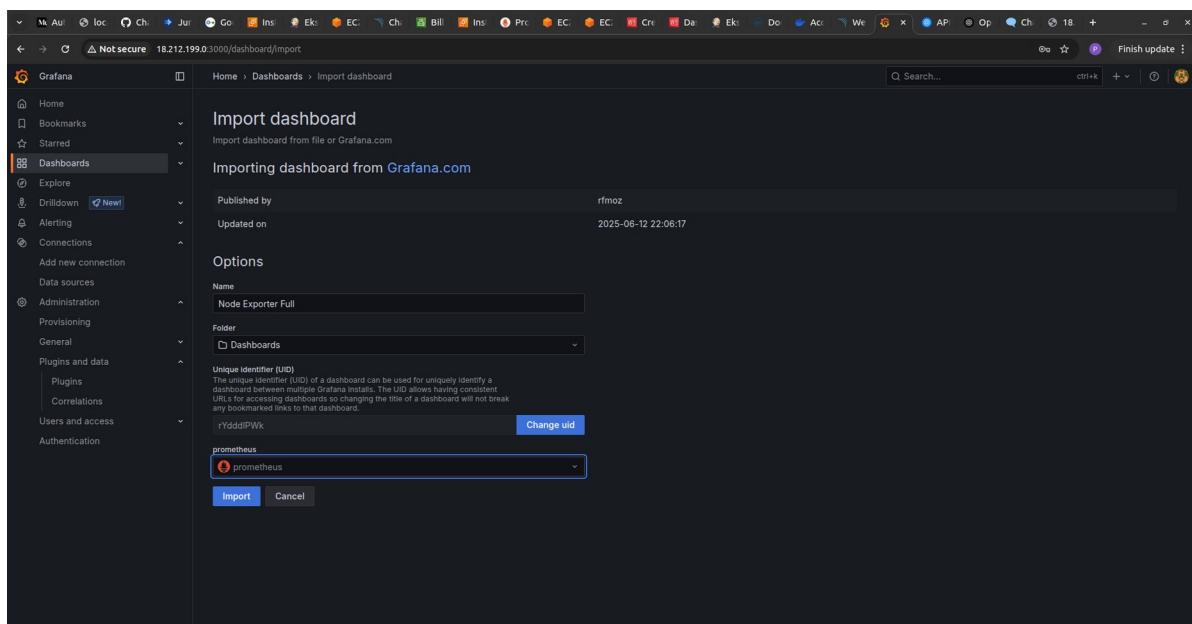
Grafana → Dashboards → Import

Use Dashboard ID:

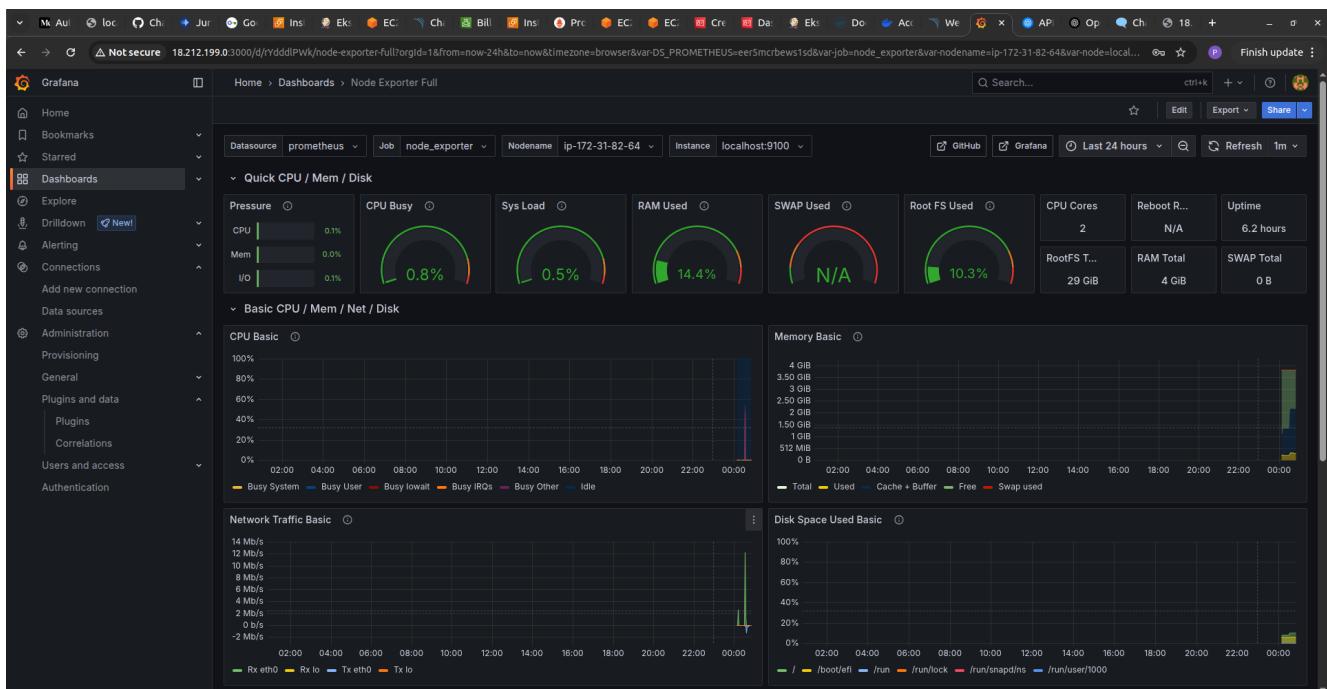
1860

You now see:

- CPU
- Memory
- Disk
- Network per node



- Click: Import
- Done



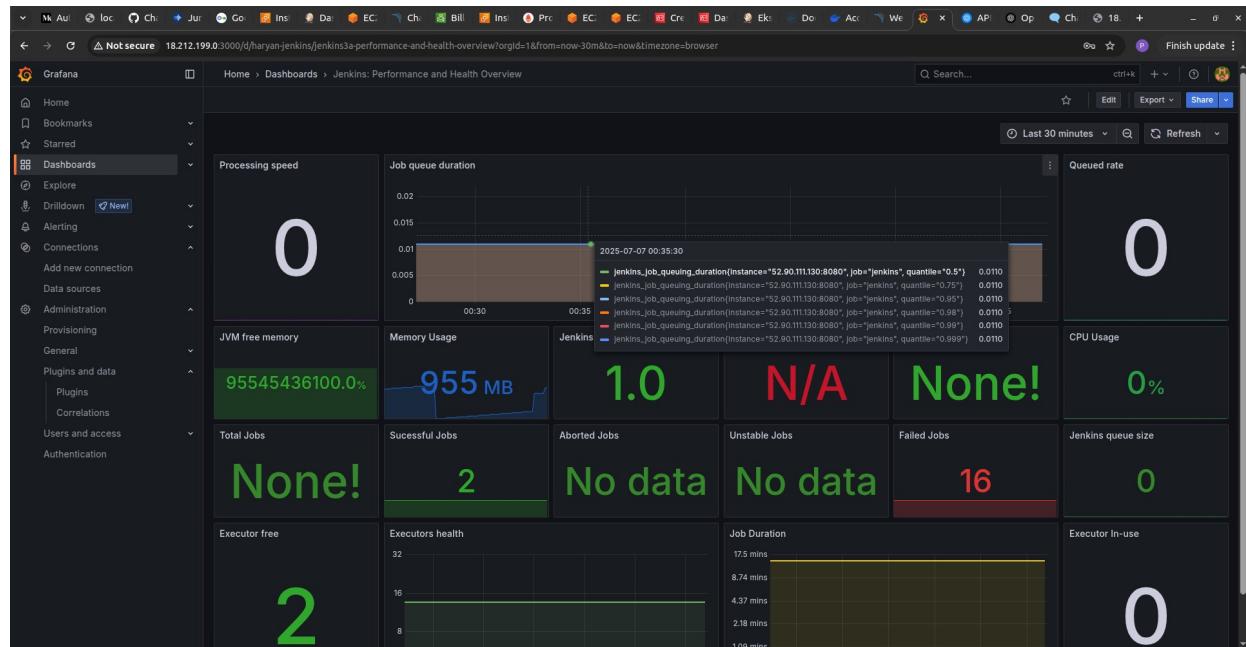
STEP 6.5 — Jenkins monitoring dashboard (ID: 9964)

manage Jenkins → system → search for Prometheus → apply → save

import a dashboard for Jenkins

+ (plus) icon → Dashboard → Import (dashboard option) → Enter Dashboard ID → Load Button

Select the data source you added (Prometheus) from the dropdown.



P r e s e **FINAL Phase 7→ Infrastructure Cleanup**

é
ñ

Destroy Infrastructure by:

kubectl **delete** deployment chatbot

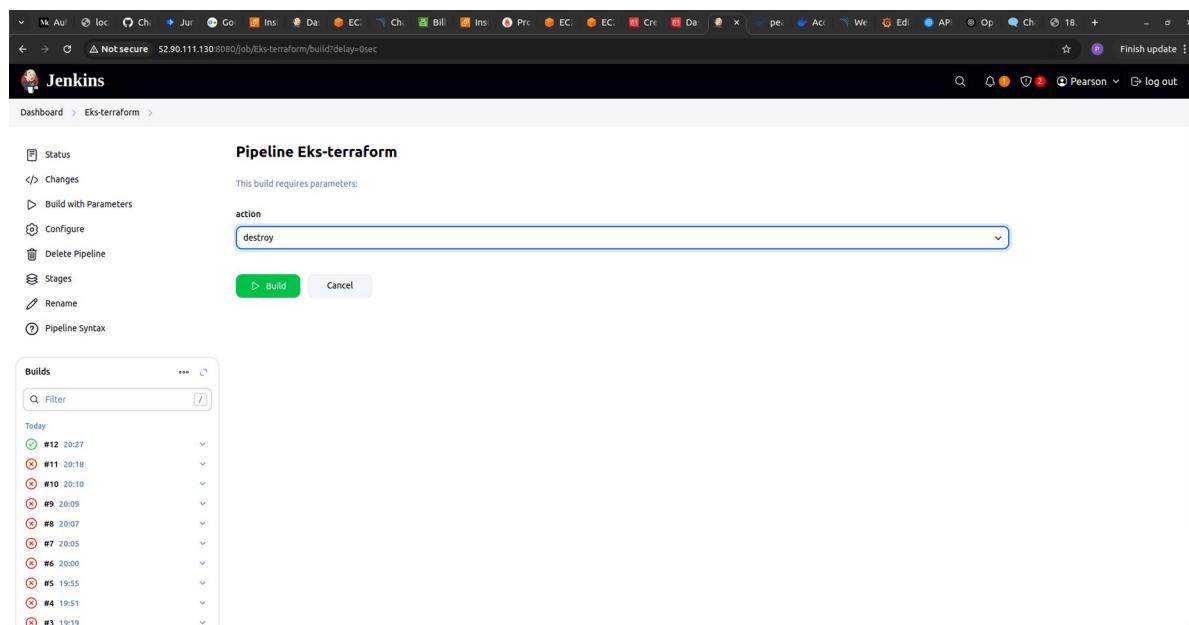
Go to your Jenkins and locate the eks-terraform pipeline Click on build with parameter and select: destroy and then click on build It will destroy your eks cluster and node group

i

3. After EKS cluster deletion let's delete the base instance → “gpt-instance and monitoring instance

Destroy : gpt-instance, monitoring instance together with all their supporting infrastructure

v
i
e



s
i
z
e

P
r
e
s

e
n
t
e
r

```
e  
n  
t      terraform destroy --auto-approve  
e  
r
```

```
O  
r          pearson@Pear: ~/Desktop/Chat-gpt-deployment/Instance-terraform  
C            
C          - to_port      = 9090  
C          },  
C          - {  
C          -   - cidr_blocks    = [  
C          -     - "0.0.0.0/0",  
C          -   ]  
C          -   - description    = "TLS from VPC"  
C          -   - from_port     = 9100  
C          -   - ipv6_cidr_blocks = []  
C          -   - prefix_list_ids = []  
C          -   - protocol       = "tcp"  
C          -   - security_groups = []  
C          -   - self           = false  
C          -   - to_port         = 9100  
O          },  
O          ] -> null  
V          - name          = "Jenkins-Security Group" -> null  
I          - owner_id       = "120569617612" -> null  
C          - revoke_rules_on_delete = false -> null  
K          - tags           = {  
I          -   - "Name" = "Jenkins-sg"  
E          } -> null  
W          - tags_all       = {  
I          -   - "Name" = "Jenkins-sg"  
E          } -> null  
V          - vpc_id         = "vpc-0e46122548ebe1b88" -> null  
I          # (1 unchanged attribute hidden)  
I          }  
M Plan: 0 to add, 0 to change, 3 to destroy.  
a  
g  
e  
Do you really want to destroy all resources?  
Terraform will destroy all your managed infrastructure, as shown above.  
There is no undo. Only 'yes' will be accepted to confirm.  
e  
Enter a value: yes  
aws_instance.web: Destroying... [id=i-016ade854064f2d9d]  
aws_instance.web2: Destroying... [id=i-08e1739952380b1f8]  
aws_instance.web: Still destroying... [id=i-016ade854064f2d9d, 10s elapsed]  
aws_instance.web2: Still destroying... [id=i-08e1739952380b1f8, 10s elapsed]  
aws_instance.web: Still destroying... [id=i-016ade854064f2d9d, 20s elapsed]  
aws_instance.web2: Still destroying... [id=i-08e1739952380b1f8, 20s elapsed]  
aws_instance.web: Still destroying... [id=i-016ade854064f2d9d, 30s elapsed]  
aws_instance.web2: Still destroying... [id=i-08e1739952380b1f8, 30s elapsed]  
aws_instance.web: Destruction complete after 33s  
aws_instance.web2: Destruction complete after 33s  
aws_security_group.Jenkins-sg: Destroying... [id=sg-0f15a057efca86318]  
aws_security_group.Jenkins-sg: Destruction complete after 2s  
Destroy complete! Resources: 3 destroyed.  
pearson@Pear:~/Desktop/Chat-gpt-deployment/Instance-terraform$
```

S
i
Z
e

Conclusion

This project showcases a **realistic DevOps implementation** combining Terraform, Jenkins, Kubernetes, and observability tooling. It highlights not just deployment success, but disciplined engineering practices around automation, security, monitoring, and cost control—core expectations for modern DevOps engineers.