Jenkins-POC

POC Overview

1. Setting up Jenkins: Continuous Integration server setup.

2. Installing Docker: Containerization tool installation for environment consistency.

3. Setting up SonarQube: Code quality and security analysis.

4. Setting up EC2 for deployment

5. Pipeline and Deployments: Automating builds, tests, and deployments.

Github repository link:

https://github.com/rushikeshmj/hello-world

Launch Virtual Machine using AWS EC2

Here is a detailed list of the basic requirements and setup for the EC2 instance i have used for running Jenkins, including the specifics of the instance type, AMI, and security groups.

EC2 Instance Requirements and Setup:

1. Instance Type

- Instance Type: `t2.large`

- vCPUs: 2

- Memory: 8 GB

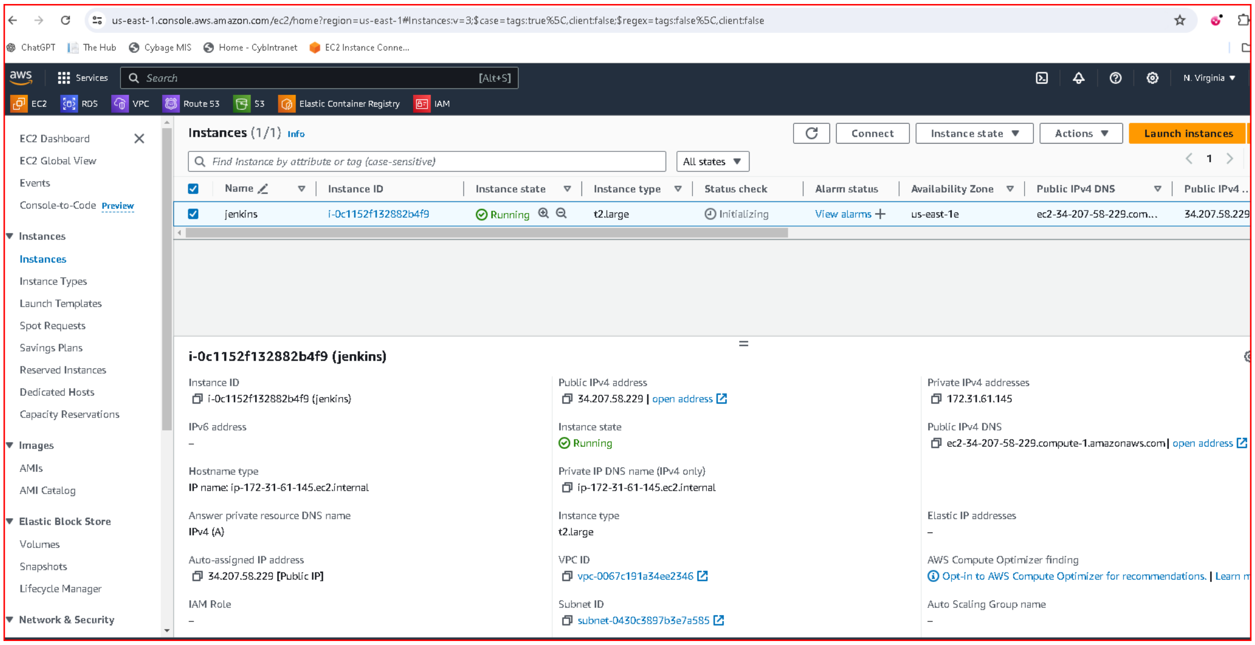
- Network Performance: Moderate

2. Amazon Machine Image (AMI)

- AMI: Ubuntu Server 20.04 LTS (Focal Fossa)

3. Security Groups

Security groups act as a virtual firewall for your instance to control inbound and outbound traffic.



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ..
☑	jenkins	i-0c1152f132882b4f9	Running	t2.large	Initializing	View alarms +	us-east-1e	ec2-34-207-58-229.com...	34.207.58.229

i-0c1152f132882b4f9 (jenkins)

Instance ID
i-0c1152f132882b4f9 (jenkins)

IPv6 address
–

Hostname type
IP name: ip-172-31-61-145.ec2.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
34.207.58.229 [Public IP]

IAM Role
–

Public IPv4 address
34.207.58.229 | open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-61-145.ec2.internal

Instance type
t2.large

VPC ID
vpc-0067c191a34ee2346

Subnet ID
subnet-0430c3897b3e7a585

Private IPv4 addresses
172.31.61.145

Public IPv4 DNS
ec2-34-207-58-229.compute-1.amazonaws.com| open address

Elastic IP addresses
–

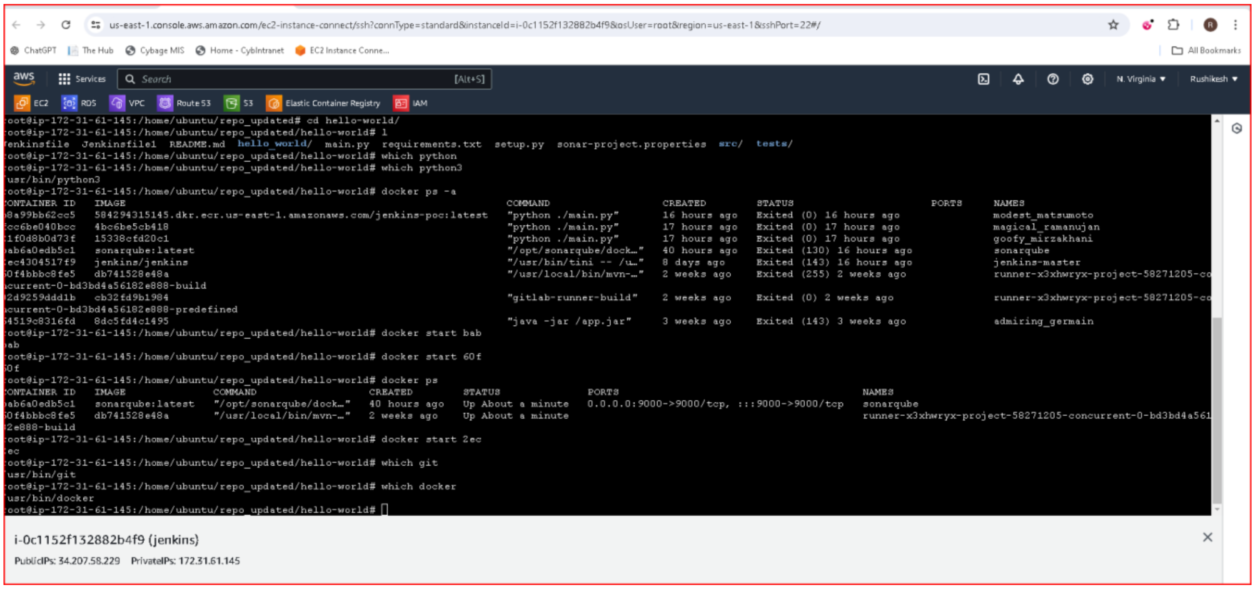
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn m

Auto Scaling Group name
–

After Launching your Virtual machine ,SSH into the Server.

Install below tools to EC2

1. **Jenkins**
   I am launching the container of Jenkins because 8080 port is not accessible on my machine
2. **Docker**
   Sudo apt-get install docker.io
   Run this command to give access to docker
   sudo chmod 666 /var/run/docker.sock
3. **Trivy**
   sudo apt-get install wget apt-transport-https gnupg lsb-release
   wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo apt-key add -
   echo deb https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main | sudo tee -a /etc/apt/sources.list.d/trivy.list
   sudo apt-get update
   sudo apt-get install trivy
4. **Git**
5. **Aws cli**


us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0c1152f132882b4f9&osUser=root&region=us-east-1&sshPort=22#/
ChatGPT
The Hub
Cybage MIS
Home - CybIntranet
EC2 Instance Conne...
All Bookmarks
aws
Services
Search
[Alt+S]
N. Virginia
Rushikesh
EC2
RDS
VPC
Route 53
S3
Elastic Container Registry
IAM

```
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated# cd hello-world/
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# l
Tenkinsfile  Jenkinsfile1  README.md  hello_world/  main.py  requirements.txt  setup.py  sonar-project.properties  src/  tests/
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# which python
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# which python3
'usr/bin/python3
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# docker ps -a
:ONTAINER ID   IMAGE                                                      COMMAND                  CREATED         STATUS                        PORTS     NAMES
o8a99bb62cc5   584294315145.dkr.ecr.us-east-1.amazonaws.com/jenkins-poc:latest   "python ./main.py"       16 hours ago    Exited (0) 16 hours ago                 modest_matsumoto
:cc6be040bcc   4bc6be5cb418                                               "python ./main.py"       17 hours ago    Exited (0) 17 hours ago                 magical_ramanujan
:1f0d8b0d73f   15338cfd20c1                                               "python ./main.py"       17 hours ago    Exited (0) 17 hours ago                 goofy_mirzakhani
oab6a0edb5c1   sonarqube:latest                                           "/opt/sonarqube/dock…"   40 hours ago    Exited (130) 16 hours ago               sonarqube
:ec4304517f9   jenkins/jenkins                                            "/usr/bin/tini -- /u…"   8 days ago      Exited (143) 16 hours ago               jenkins-master
50f4bbbc8fe5   db741528e48a                                               "/usr/local/bin/mvn-…"   2 weeks ago     Exited (255) 2 weeks ago                runner-x3xhwryx-project-58271205-co
ncurrent-0-bd3bd4a56182e888-build
)2d9259ddd1b   cb32fd9b1984                                               "gitlab-runner-build"    2 weeks ago     Exited (0) 2 weeks ago                  runner-x3xhwryx-project-58271205-co
ncurrent-0-bd3bd4a56182e888-predefined
54519c8316fd   8dc5fd4c1495                                               "java -jar /app.jar"     3 weeks ago     Exited (143) 3 weeks ago                admiring_germain
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# docker start bab
oab
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# docker start 60f
50f
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# docker ps
:ONTAINER ID   IMAGE              COMMAND                  CREATED        STATUS              PORTS                                       NAMES
oab6a0edb5c1   sonarqube:latest   "/opt/sonarqube/dock…"   40 hours ago   Up About a minute   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   sonarqube
50f4bbbc8fe5   db741528e48a       "/usr/local/bin/mvn-…"   2 weeks ago    Up About a minute                                               runner-x3xhwryx-project-58271205-concurrent-0-bd3bd4a561
32e888-build
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# docker start 2ec
:ec
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# which git
'usr/bin/git
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# which docker
'usr/bin/docker
:oot@ip-172-31-61-145:/home/ubuntu/repo_updated/hello-world# 
```

i-0c1152f132882b4f9 (jenkins)

PublicIPs: 34.207.58.229   PrivateIPs: 172.31.61.145

Configure Jenkins

Access Jenkins Dashboard:

Open a web browser and navigate to your Jenkins instance

Log in with your Jenkins credentials. (cat address provided on Jenkins)

Install Plugins:

-Go to Manage Jenkins > Manage Plugins.

-Click on the Available tab.

Search for and install the following plugins:

1. Docker: Enables Jenkins to use Docker containers.
2. Sonar Scanner: For Scanning Vulnerabilities.
3. Docker Pipeline: Allows Jenkins to use Docker containers in pipeline jobs.
4. ECR
5. Cobertura
6. Code scanner plugin
7. Github
8. Pipeline stage view
9. Ssh
10. Suggested plugins

Configuring SonarQube Scanner Plugin

1. Manage Jenkins: Go to "Manage Jenkins".

2. Global Tool Configuration: Click on "Global Tool Configuration".

3. SonarQube Scanner:

- Scroll down to the "SonarQube Scanner" section.
- Click "Add SonarQube Scanner".
- Provide a name (e.g., Sonar scanner).
- Optionally, check "Install automatically" to let Jenkins handle the installation.
- Save the configuration.

4. Manage Jenkins: Go back to "Manage Jenkins".

5. Configure System:

- Scroll down to the "SonarQube servers" section.
- Click "Add SonarQube".
- Provide a name for the server (e.g., SonarQube).
- Set the "Server URL" to the URL of your SonarQube instance.
- Add a "Server Authentication Token".

Creating a Token on SonarQube

1. Log in to SonarQube: Open your SonarQube instance in a web browser and log in.

2. My Account: Click on your user profile at the top-right corner and select "My Account".

3. Security: Navigate to the "Security" tab.

4. Generate Token: Under "Generate Tokens", provide a name for the token (e.g., JenkinsToken).

5. Generate: Click on "Generate" and copy the token.

Adding SonarQube Token to Jenkins

1. Manage Jenkins: Go to "Manage Jenkins".

2. Configure System: Scroll to the "SonarQube servers" section.

3. Add Token:

- Under the "Server Authentication Token" section, click "Add" next to "Credentials".

- Select "Jenkins" and then "Secret text".

- Paste the token you copied from SonarQube.

- Provide an ID (e.g., sonarqube-token).

- Save the credentials.

- Select the newly added token from the dropdown list.

Configuring Docker Plugin

1. Manage Jenkins: Go to "Manage Jenkins".

2. Global Tool Configuration: Click on "Global Tool Configuration".

3. Docker:

- Scroll down to the "Docker" section.

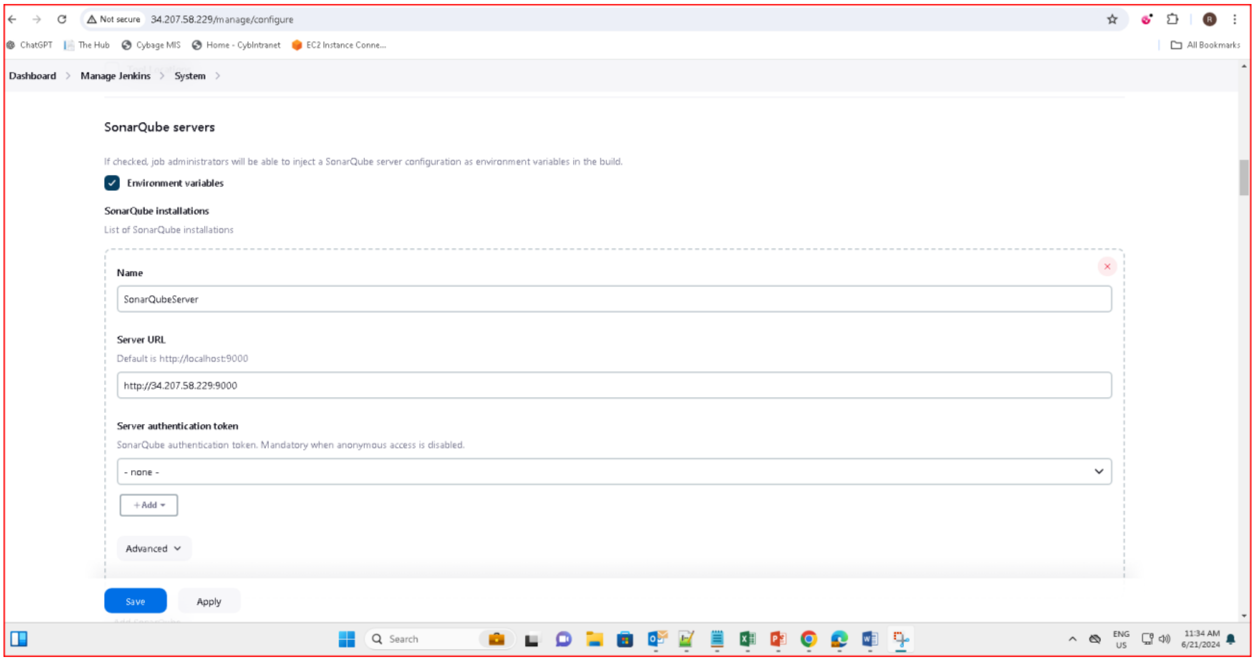
- Click "Add Docker Tool".

- Provide a name (e.g., docker).

- Optionally, check "Install automatically" to let Jenkins handle the installation.

- Save the configuration.

Configure system with their setting and path



Not secure 34.207.58.229/manage/configure
ChatGPT | The Hub | Cybage MIS | Home - CybIntranet | EC2 Instance Conne... | All Bookmarks
Dashboard > Manage Jenkins > System >

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

☑ Environment variables

SonarQube installations

List of SonarQube installations

Name: SonarQubeServer

Server URL

Default is http://localhost:9000

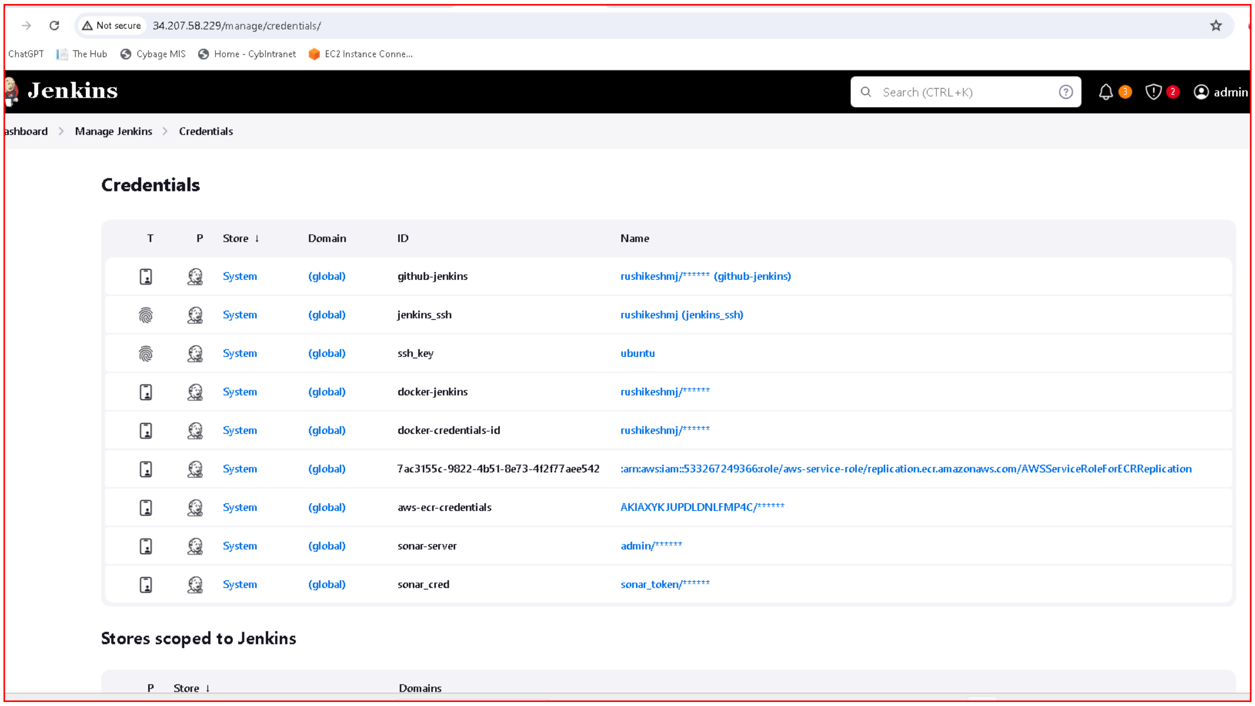
http://34.207.58.229:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

Save the credentials



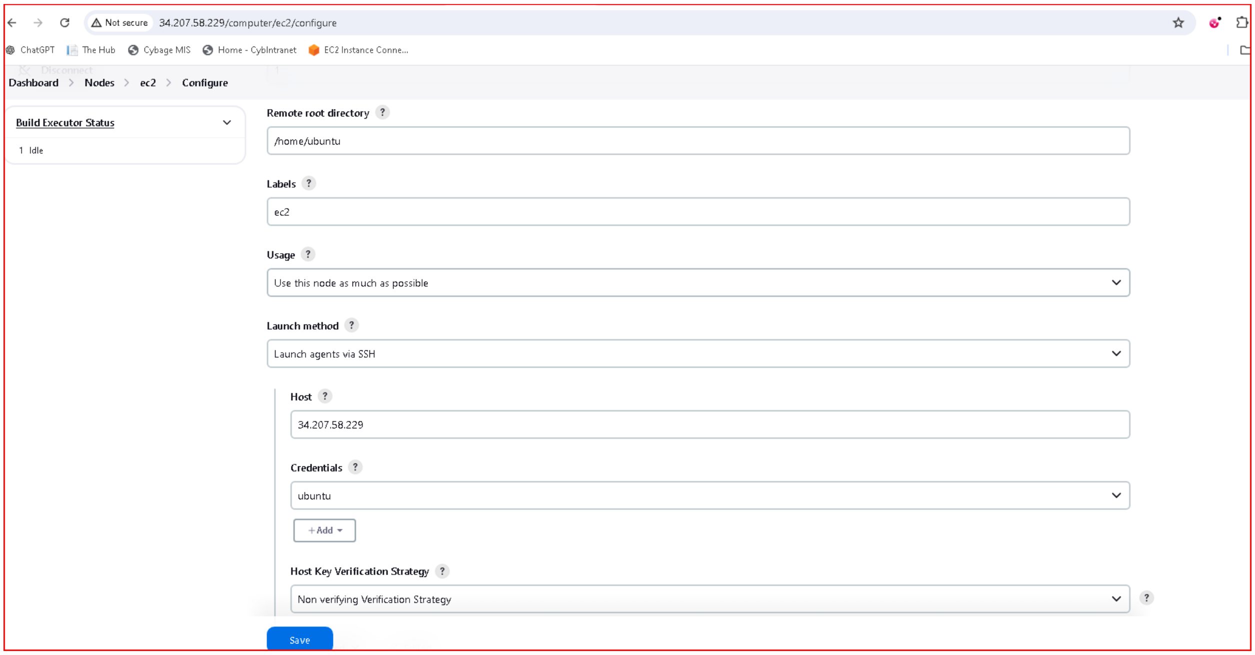
Not secure 34.207.58.229/manage/credentials/
ChatGPT | The Hub | Cybage MIS | Home - CybIntranet | EC2 Instance Conne...
Jenkins
Search (CTRL+K)
admin
ashboard > Manage Jenkins > Credentials

Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	github-jenkins	rushikeshmj/****** (github-jenkins)
		System	(global)	jenkins_ssh	rushikeshmj (jenkins_ssh)
		System	(global)	ssh_key	ubuntu
		System	(global)	docker-jenkins	rushikeshmj/******
		System	(global)	docker-credentials-id	rushikeshmj/******
		System	(global)	7ac3155c-9822-4b51-8e73-4f2f77aee542	:arn:aws:iam::533267249366:role/aws-service-role/replication.ecr.amazonaws.com/AWSServiceRoleForECRReplication
		System	(global)	aws-ecr-credentials	AKIAXYKJUPDLDNLFMP4C/******
		System	(global)	sonar-server	admin/******
		System	(global)	sonar_cred	sonar_token/******

Stores scoped to Jenkins

Create Node and configure



Remote root directory: /home/ubuntu

Labels: ec2

Usage: Use this node as much as possible

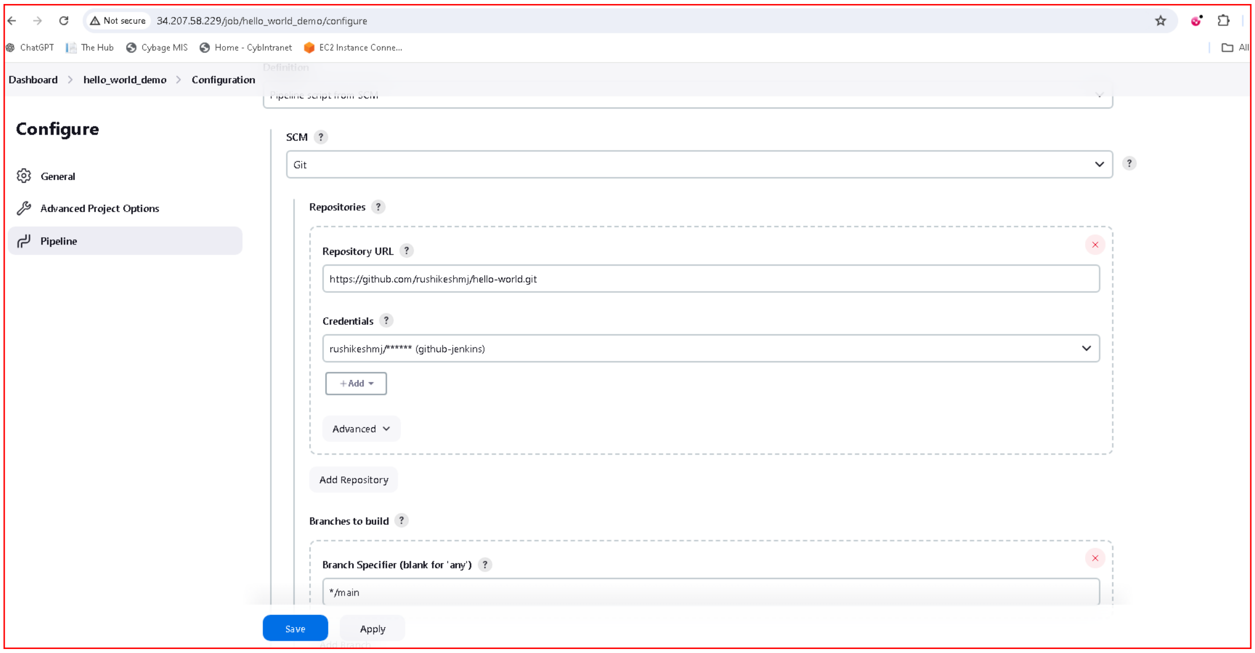
Launch method: Launch agents via SSH

Host: 34.207.58.229

Credentials: ubuntu

Host Key Verification Strategy: Non verifying Verification Strategy

Create Job and it to github repo and cred



SCM: Git

Repositories

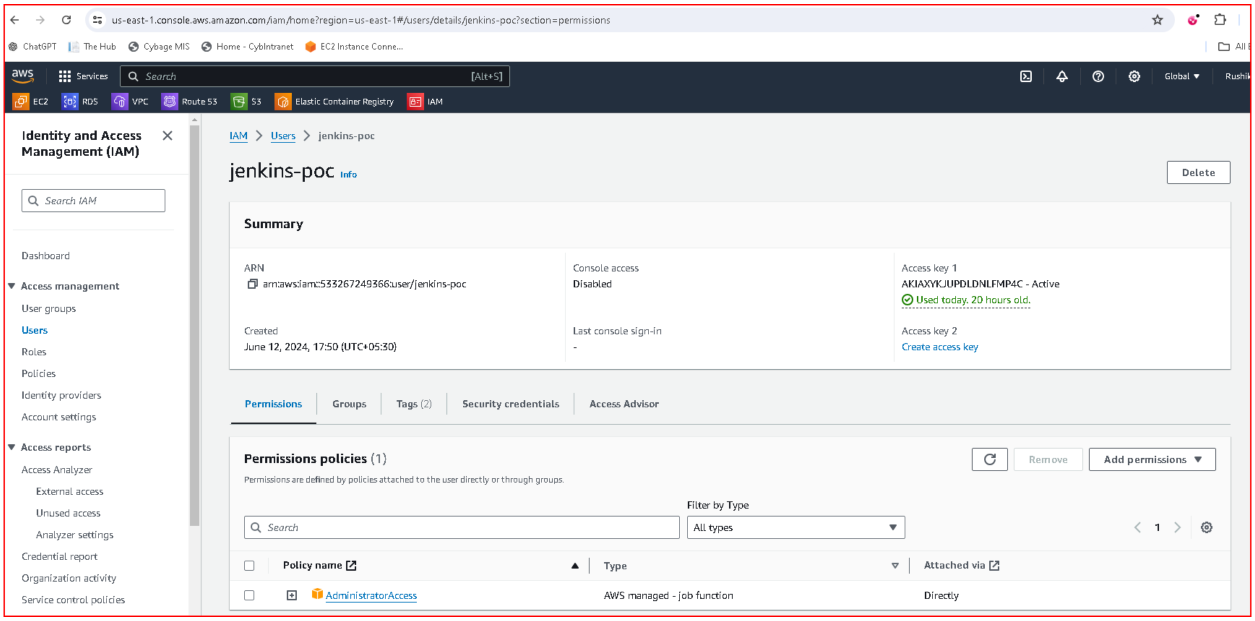
Repository URL: https://github.com/rushikeshmj/hello-world.git

Credentials: rushikeshmj/****** (github-jenkins)

Branches to build

Branch Specifier (blank for 'any'): */main

Create IAM user



us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/details/jenkins-poc?section=permissions

IAM > Users > jenkins-poc

jenkins-poc

Summary

ARN arn:aws:iam::533267249366:user/jenkins-poc	Console access Disabled	Access key 1 AKIAXYKJUPDLDNLFMP4C - Active Used today. 20 hours old.
Created June 12, 2024, 17:50 (UTC+05:30)	Last console sign-in -	Access key 2 Create access key

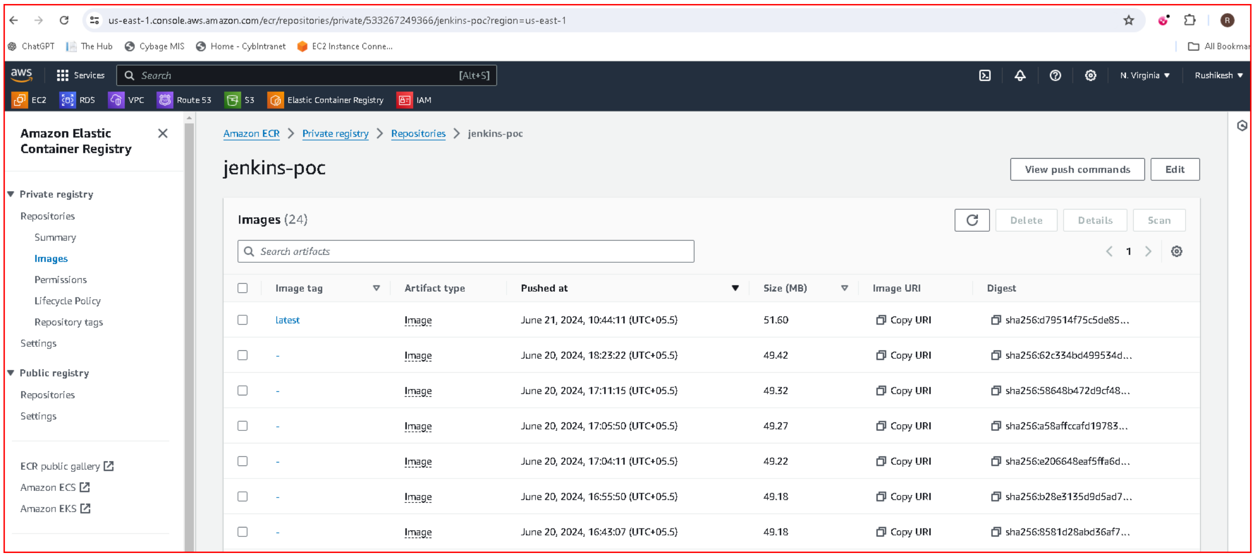
Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

□	Policy name	Type	Attached via
□	AdministratorAccess	AWS managed - job function	Directly

Create ECR repo



us-east-1.console.aws.amazon.com/ecr/repositories/private/533267249366/jenkins-poc?region=us-east-1

Amazon ECR > Private registry > Repositories > jenkins-poc

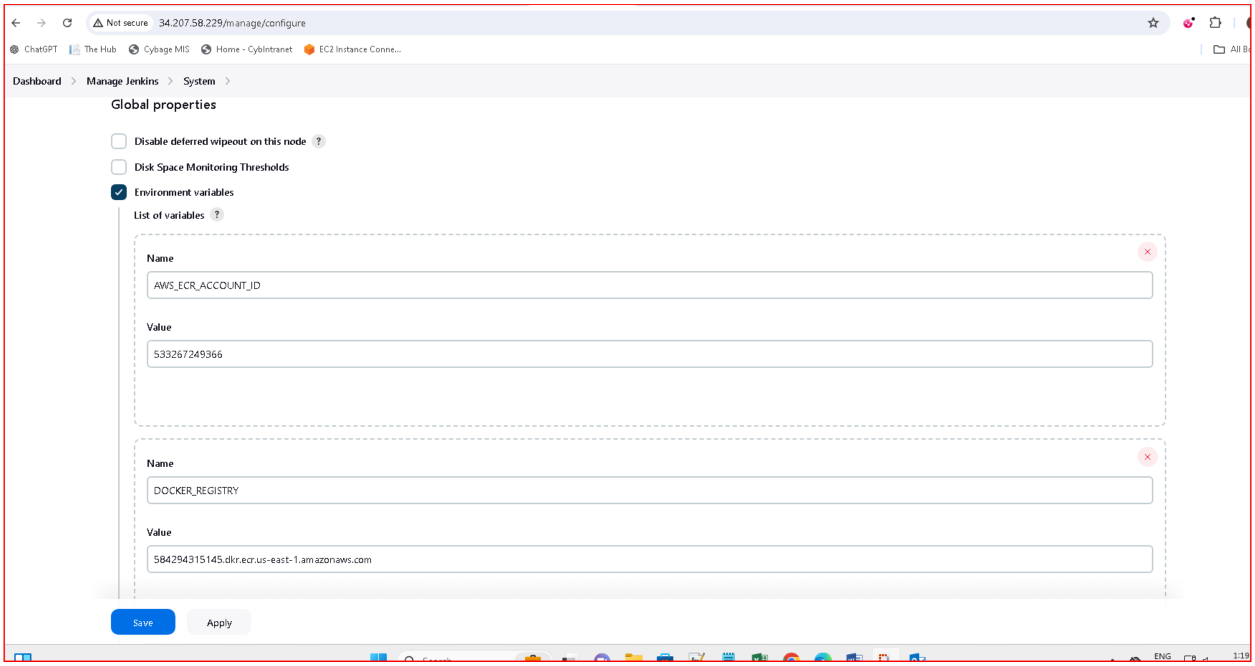
jenkins-poc

Images (24)

□	Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest
□	latest	Image	June 21, 2024, 10:44:11 (UTC+05.5)	51.60	Copy URI	sha256:d79514f75c5de85...
□	-	Image	June 20, 2024, 18:23:22 (UTC+05.5)	49.42	Copy URI	sha256:62c334bd499534d...
□	-	Image	June 20, 2024, 17:11:15 (UTC+05.5)	49.32	Copy URI	sha256:58648b472d9cf48...
□	-	Image	June 20, 2024, 17:05:50 (UTC+05.5)	49.27	Copy URI	sha256:a58affccafd19783...
□	-	Image	June 20, 2024, 17:04:11 (UTC+05.5)	49.22	Copy URI	sha256:e206648eaf5ffa6d...
□	-	Image	June 20, 2024, 16:55:50 (UTC+05.5)	49.18	Copy URI	sha256:b28e3135d9d5ad7...
□	-	Image	June 20, 2024, 16:43:07 (UTC+05.5)	49.18	Copy URI	sha256:8581d28abd36af7...

ADD ENV variables

Manage Jenkins → system → Global properties → ENV varibales



Global properties

□ Disable deferred wipeout on this node

□ Disk Space Monitoring Thresholds

☑ Environment variables

List of variables

Name: AWS_ECR_ACCOUNT_ID

Value: 533267249366

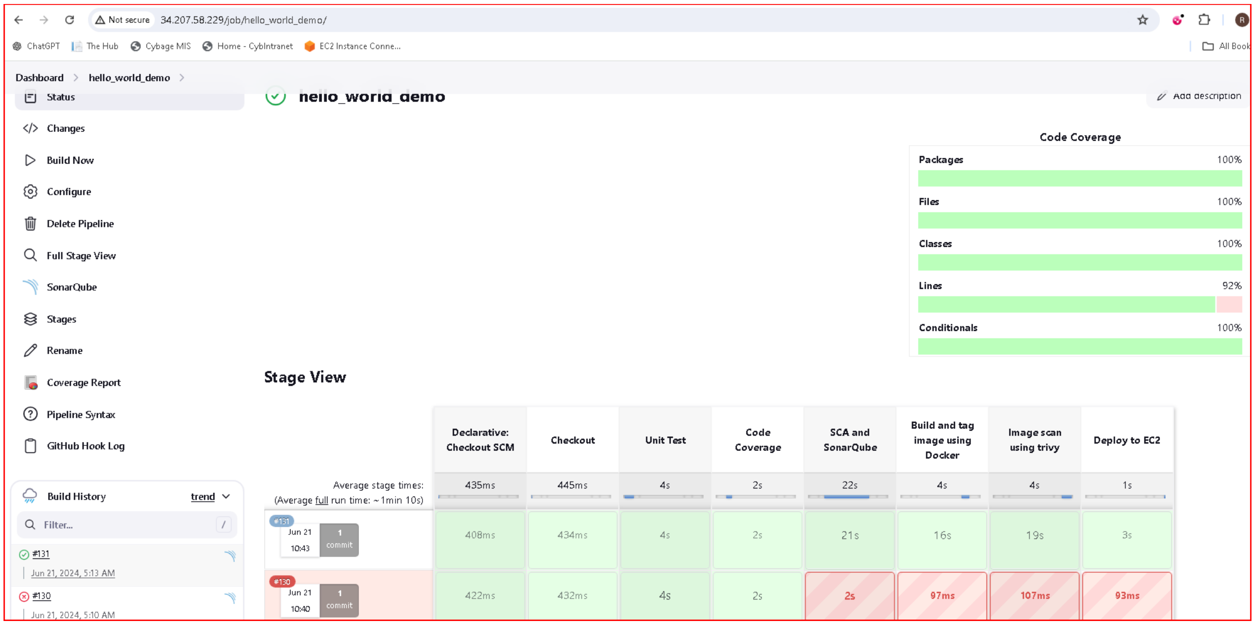
Name: DOCKER_REGISTRY

Value: 584294315145.dkr.ecr.us-east-1.amazonaws.com

Create Pipeline

```
pipeline {
    agent {
        label 'ec2'
    }

    environment {
        SONARQUBE_SERVER = "http://$PUBLIC_IP:9000/"
        AWS_ECR_REGION = "us-east-1"
        AWS_ECR_REPOSITORY = "jenkins-poc"
        DOCKER_IMAGE_NAME = "hello-world-app"
    }

    stages {
        stage('Checkout') {
            steps {
                retry(3) {
                    checkout([$class: 'GitSCM', branches: [[name: '*/main']], userRemoteConfigs: [[url: 'https://github.com/shekharbo/hello-world.git']]])
                }
            }
        }

        stage('Unit Test') {
            steps {
                sh 'docker --version'
                sh 'docker pull python:3.9'
                sh 'pip install -r requirements.txt'
                sh 'python3 -m venv ~/myenv'
                sh """
                set +x
                . /home/ubuntu/myenv/bin/activate
                """
                sh """
                /home/ubuntu/.local/bin/pytest /home/ubuntu/workspace/hello_world_demo/tests/test_main.py
                """
            }
        }
    stage('Code Coverage') {
      steps {
        sh 'pip install coverage'
        sh """
        /home/ubuntu/.local/bin/coverage run -m pytest /home/ubuntu/workspace/hello_world_demo/tests/test_main.py
        """
        sh '/home/ubuntu/.local/bin/coverage report'
        sh '/home/ubuntu/.local/bin/coverage xml -o coverage.xml'
        cobertura coberturaReportFile: 'coverage.xml'
      }
    }

    stage('SCA and SonarQube') {
      steps {
        withSonarQubeEnv('SonarQubeServer') {
          script {
            def scannerHome = tool 'SonarQubeScanner'
            if (scannerHome) {
              sh "/home/hello-world-demo-python/hello-world/sonar-scanner-5.0.1.3006-linux/bin/sonar-scanner \
                -Dsonar.projectKey=hello-world \
                -Dsonar.sources=src \
                -Dsonar.host.url=${SONARQUBE_SERVER} \
                -Dsonar.login=${SONARQUBE_LOGIN_TOKEN}"
            } else {
              error "SonarQube Scanner not configured."
            }
          }
        }
      }
    }

    stage('Build and tag image using Docker') {
      steps {
        script {
          dir('/home/ubuntu/hello-world-demo-python/hello-world') {
            sh 'pwd'
            sh 'ls -l Dockerfile'
            sh 'docker build -t hello-world-app .'
            sh "docker tag hello-world-app ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com/${AWS_ECR_REPOSITORY}:latest"
            withCredentials([usernamePassword(credentialsId: 'aws-ecr-credentials', usernameVariable: 'AWS_ACCESS_KEY_ID', passwordVariable: 'AWS_SECRET_ACCESS_KEY')]) {
              sh "aws ecr get-login-password --region ${AWS_ECR_REGION} | docker login --username AWS --password-stdin ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com"
              sh "docker push ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com/${AWS_ECR_REPOSITORY}:latest"
            }
          }
        }
      }
    }
    stage('Image scan using trivy') {
      steps {
        sh "trivy image ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com/${AWS_ECR_REPOSITORY}:latest"
      }
    }

    stage('Deploy to EC2') {
      steps {
        sshagent(['ssh_key']) {
          sh "ssh -o StrictHostKeyChecking=no ubuntu@$PUBLIC_IP aws ecr get-login-password --region ${AWS_ECR_REGION} | docker login --username AWS --password-stdin ${DOCKER_REGISTRY}"
          sh "ssh -o StrictHostKeyChecking=no ubuntu@$PUBLIC_IP docker pull ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com/${AWS_ECR_REPOSITORY}:latest"
          sh "ssh -o StrictHostKeyChecking=no ubuntu@$PUBLIC_IP docker run -d -p 8081:8080 ${AWS_ECR_ACCOUNT_ID}.dkr.ecr.${AWS_ECR_REGION}.amazonaws.com/${AWS_ECR_REPOSITORY}:latest"
        }
      }
    }
  }
}
```

Successful Deployment



hello_world_demo

Code Coverage

Packages	100%
Files	100%
Classes	100%
Lines	92%
Conditionals	100%

Build History

- #131 Jun 21, 2024, 5:13 AM
- #130 Jun 21, 2024, 5:10 AM

Stage View

	Declarative: Checkout SCM	Checkout	Unit Test	Code Coverage	SCA and SonarQube	Build and tag image using Docker	Image scan using trivy	Deploy to EC2
Average stage times: (Average full run time: ~1min 10s)	435ms	445ms	4s	2s	22s	4s	4s	1s
#131 Jun 21 10:43 1 commit	408ms	434ms	4s	2s	21s	16s	19s	3s
#130 Jun 21 10:40 1 commit	422ms	432ms	4s	2s	2s	97ms	107ms	93ms