

AWS Deployment Project:

Project Name: Starbucks Coffee Shop Project

Name : S.Venkateswara Reddy



→ Launch EC2 Instance with t2.xlarge

Security Group: Allow All Traffic (For Practice Purpose)

→ Connect Virtual Machine Through SSH

EC2 Instance Connect Session Manager SSH client EC2 serial console

Instance ID i-087476c71ef34c8d8 (Amazon-Prime-Video)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is Amazon.pem.
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "Amazon.pem"
- Connect to your instance using its Public DNS:
ec2-3-108-56-203.ap-south-1.compute.amazonaws.com

Example:
ssh -i "Amazon.pem" ubuntu@ec2-3-108-56-203.ap-south-1.compute.amazonaws.com

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

→ Connect through Git Bash or MobaXterm
→ Sudo apt update -y
→ Connect the server and git clone <https://github.com/Singareddy-Venkatesh/starbucks-kubernetes.git>

→ cd scripts install all Packages Jenkins, Docker, SonarQube, Trivy, Docker Scout
→ cat permissionexecut.sh (chmod +x *.sh)
→ chmod +x permissionexecut.sh (All get Execute Permissions)
→ sh permissionexecute.sh

Jenkins Packages:

```
#!/bin/bash
# jenkins installation on ubuntu
sudo apt update -y
sudo apt install fontconfig openjdk-17-jre -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl enable jenkins
sudo systemctl start Jenkins
```

=>sh Jenkins.sh

⇒ <Public IP>8080 (You Will Open Jenkins Dashboard)
⇒ In Your Terminal (cat /var/lib/jenkins/secrets/initialAdminPassword)
Here generates a password for to Login into the Jenkins Dashboard and Install Plugins

Install Docker Packages:

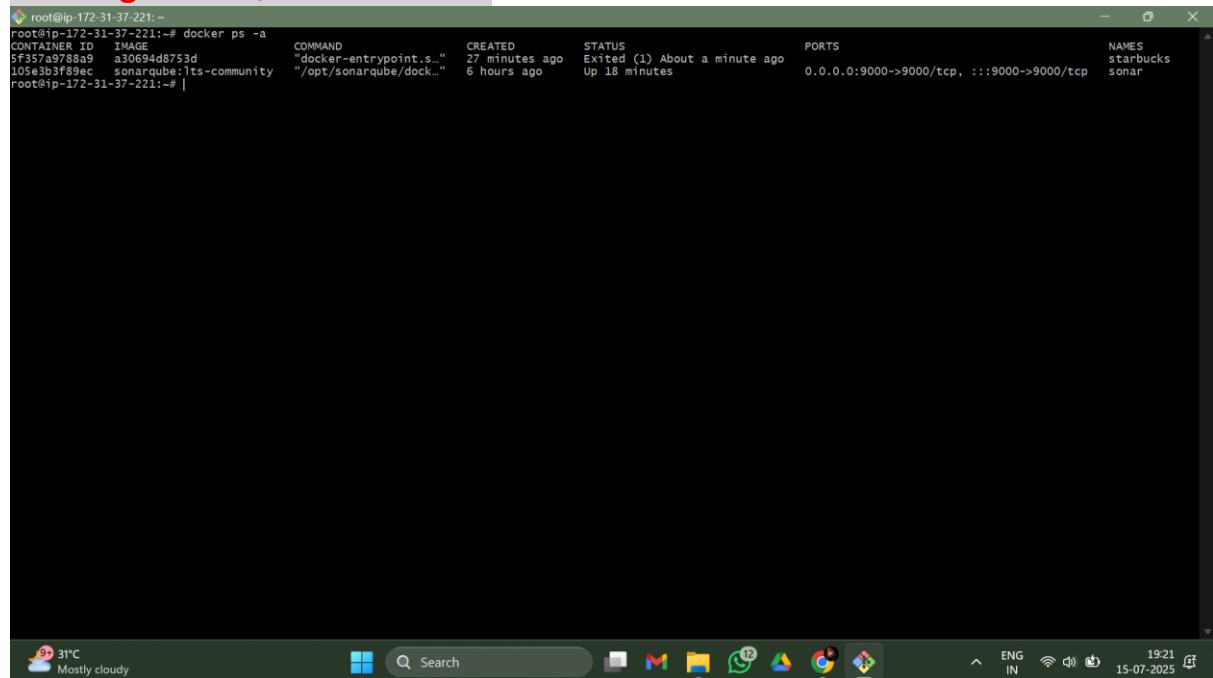
→ Install docker (sh docker.sh) by apt install docker.io -y

```
#!/bin/bash
sudo apt-get update -y
sudo apt-get install docker.io -y
sudo usermod -aG docker ubuntu
```

```
sudo usermod -aG docker jenkins  
newgrp docker  
sudo chmod 660 /var/run/docker.sock  
sudo chown root:docker /var/run/docker.sock  
sudo systemctl restart docker  
docker -version
```

→ Install SonarQube (docker run --name sonar -d -p 9000:9000 sonarqube:lts-community)

Running SonarQube Container:



```
root@ip-172-31-37-221:~# docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
5f357a9768a9 a30694d6753d "docker-entrypoint.s..." 27 minutes ago Exited (1) About a minute ago 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp starbucks  
103e3b3f3f9ec sonarqube:lts-community "/opt/sonarqube/dock... 6 hours ago Up 18 minutes  
root@ip-172-31-37-221:~# |
```

→ Install trivy (sh trivy.sh)

```
#!/bin/bash  
# Install necessary dependencies  
sudo apt-get install wget apt-transport-https gnupg lsb-release -y
```

```
# Add the Trivy repository key  
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --  
dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
```

```
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg]  
https://aquasecurity.github.io/trivy-repo/deb ${lsb_release -sc} main" | sudo  
tee -a /etc/apt/sources.list.d/trivy.list
```

```
sudo apt-get update -y  
sudo apt-get install trivy -y
```

In Jenkins Install Plugins:

Eclipse Temurin Installer

SonarQube Scanner

Sonar Quality Gates

Pipeline Stage View

NodeJs

Docker

Docker Compose

Docker Pipeline

Docker API

Kubernetes

Kubernetes Client API

Kubernetes Credentials

Kubernetes CLI

Kubernetes Credentials Provider

Kubernetes:Pipeline:DevOps Steps

Blue Ocean (We Start Pipeline in Different Stages)

Create Pipeline Job Project in Jenkins:

The screenshot shows the Jenkins dashboard with a single pipeline job listed. The job is named 'Amazon_Deploy' and has the following details:

S	W	Name	Last Success	Last Failure	Last Duration
✓	cloud	Amazon_Deploy	19 min #3	23 min #2	2 min 27 sec

Below the job list, there are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (0 of 7 executors busy). At the bottom right, there are links for 'REST API' and 'Jenkins 2.504.3'.

⇒ In Browser <Public IP>:9000 You can Open SonarQube

⇒ Username: admin

⇒ Password: admin

Integrate SonarQube with Jenkins:

In SonarQube → Go to Administrator → User → Create Token

The screenshot shows the SonarQube 'Users' administration page. The 'Administration' tab is selected. A single user, 'Administrator' (login: admin), is listed. The 'Tokens' column indicates 2 tokens for groups 'sonar-administrators' and 'sonar-users'. A note at the bottom 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.'

⇒ In Configuration → Webhook → Name (Jenkins) → <JenkinsURL>sonarqube-webhooks/ → Create

Ex: <http://3.108.56.203:8080/sonarqube-webhook/>

Update Webhook

All fields marked with * are required

Name *

jenkins



URL *

<http://3.110.87.41:8080/sonarqube-webhook/>

Server endpoint that will receive the webhook payload, for example: "http://my_server/foo". If HTTP Basic authentication is used, HTTPS is recommended to avoid man in the middle attacks. Example: "https://myLogin:myPassword@my_server/foo"

Secret

Hidden for security reasons. [Click here to update the secret](#)

If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header. If blank, any secret previously configured will be removed. If not set, the secret will remain unchanged.

The screenshot shows the SonarQube Administration interface. In the top navigation bar, the 'Administration' tab is selected. Below it, the 'Webhooks' section is visible. A table lists a single webhook entry:

Name	URL	Has secret?	Last delivery	Actions
Jenkins	http://3.108.56.203:8080/sonarqube-webhook/	No	✓ July 15, 2025 at 6:46 PM	

A note at the bottom of the section 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."

⇒ To Integrate SonarQube with Jenkins → Go to Manage Jenkins → System
→ SonarQube Server → Select SonarQube Installations → Name (SonarQube) → SonarQube URL → Select Server AuthenticationToken

The screenshot shows the Jenkins 'Manage Jenkins > System' configuration page. Under the 'SonarQube installations' section, a new server is being configured with the following details:

- Name:** SonarQube
- Server URL:** Default is http://localhost:9000
http://3.108.56.203:9000
- Server authentication token:** SonarQube authentication token. Mandatory when anonymous access is disabled.
Sonar-token

At the bottom of the form are 'Save' and 'Apply' buttons.

⇒ In Jenkins → Go to Credentials → In Global → Select Secret Text → Paste Token → Give Same Name As per the Pipeline (Sonar-token) → Create

⇒ Create Credentials of (SonarQube, Docker, Gmail)

The screenshot shows the Jenkins 'Credentials' management interface. It displays two sections: 'Stores scoped to Jenkins' and 'Stores shared across Jenkins'. In the 'Stores scoped to Jenkins' section, there are entries for 'System' and 'Kubernetes'. In the 'Stores shared across Jenkins' section, there are entries for 'SonarQube' (ID: Sonar-token, Name: Sonar-token), 'Docker' (ID: docker, Name: venkatesh09***** (docker)), and 'Gmail' (ID: smtp-token, Name: singareddyv91@gmail.com***** (smtp-token)).

- ⇒ In Jenkins → In tools → SonarQube Scanner Installations → sonar-scanner → Select **SonarQube Scanner 7.1.0.4889** → Apply

⇒ Docker Credentials in Jenkins:

- Go to Manage Jenkins → System → Global → Select Username & Password
→ Give Docker Username & Password → Give Same Name As per the Pipeline (docker) → Create

→ **Install Docker-Scout** → go to Terminal docker login with Username & Password Install

```
curl -sSfL https://raw.githubusercontent.com/docker/scout-cli/main/install.sh | sh -s -- -b /usr/local/bin sh install-scout.sh
```

→ In Jenkins → Manage Jenkins → tools → Select Docker Installations

The screenshot shows the 'Docker installations' configuration page. A new installation named 'docker' is being created. The 'Installation root' is set to 'docker'. A warning message states: 'docker is not a directory on the Jenkins controller (but perhaps it exists on some agents)'. The 'Install automatically' checkbox is checked. Under the 'Download from docker.com' section, the 'Docker version' is set to 'latest'. At the bottom, there are 'Save' and 'Apply' buttons.

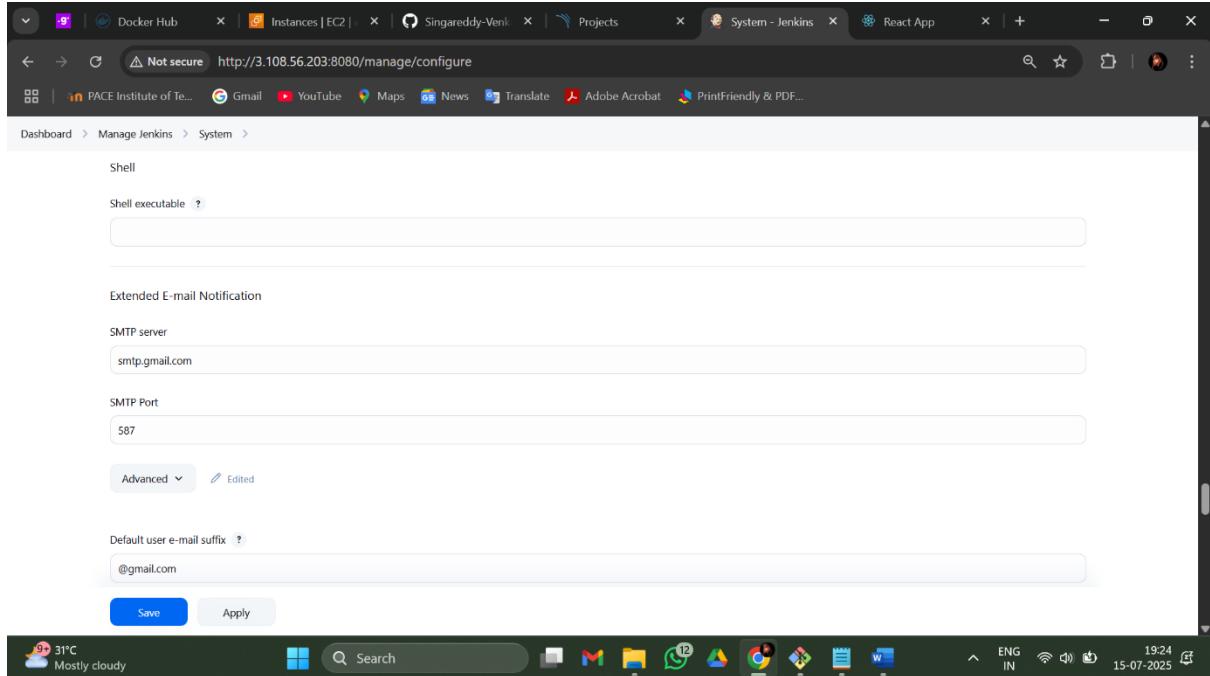
- ⇒ In Jenkins → Go to Manage Jenkins → Tools → JDK Installations → jdk → Select Install from Adoptium.net (Version jdk-17.0.1+12)

Nodejs Integrate with Jenkins:

- ⇒ In Jenkins → Manage Jenkins → NodeJs Installations → node17 → Version(NodeJS 17.0.0)

- ⇒ Email Notification → go to google Account → Manage Google → Security → Search App password → Create Token
- ⇒ After go to Jenkins Credentials → Username & Password → Your Gmail & Token which we created → ID ([smtp-token](#)) → Create
- ⇒ In Jenkins → Manage Jenkins → System → Email Notification
 → Name([smtp.gmail.com](#)) → Default user email ([@gmail.com](#)) → Advanced
 → Use SMTP Authentication (Your Gmail) → Give Token → Select Use SSL
 → Port:[465](#) → Reply-to-Address (your Gmail) → select Email (Your Gmail)

- ⇒ In Jenkins → Manage Jenkins → Go to Extended E-mail Notification
 → smtp.gmail → Port:587 → Select Credentials → Select Use TLS → Default User email Suffix(@gmail.com) → Apply



→ In Jenkins write Declarative Pipeline

Pipeline Syntax → Snippet Generator → Select Git → Paste the Git Repository → you will get the Generate Pipeline Script Paste into your Pipeline Groovy Script.

Jenkins Pipeline:

```
pipeline{
    agent any
    tools{
        jdk 'jdk'
        nodejs 'node17'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
    }
}
```

```
        }
    }
stage('Checkout from Git'){
    steps{
        git branch: 'main', url: 'https://github.com/Singareddy-
Venkatesh/starbucks-kubernetes.git'
    }
}
stage("Sonarqube Analysis"){
    steps{
        withSonarQubeEnv('SonarQube') {
            sh """ $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=starbucks \
-Dsonar.projectKey=starbucks """
        }
    }
}
stage("quality gate"){
    steps {
        script {
            waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
        }
    }
}
stage('Install Dependencies') {
    steps {
        sh "npm install"
    }
}
stage('TRIVY FS SCAN') {
    steps {
        sh "trivy fs . > trivyfs.txt"
    }
}
stage('Free Disk Space') {
    steps {
        echo "Running cleanup to free disk space..."
        sh 'df -h'
        sh 'docker system prune -af || true'
        sh 'docker volume prune -f || true'
    }
}
```

```

        // DON'T delete the current workspace manually here!
        sh 'df -h'
    }
}

stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {
                sh "docker build -t starbucks ."
                sh "docker tag starbucks venkatesh09/starbucks:latest"
                sh "docker push venkatesh09/starbucks:latest"
            }
        }
    }
}

stage('App Deploy to Docker container'){
    steps{
        sh 'docker run -d --name starbucks -p 3000:3000
venkatesh09/starbucks:latest'
    }
}

}

post {
always {
script {
    def buildStatus = currentBuild.currentResult
    def buildUser =
currentBuild.getBuildCauses('hudson.model.Cause$UserIdCause')[0]?.userId ?:
'Github User'

emailext (
    subject: "Pipeline ${buildStatus}: ${env.JOB_NAME}
#${env.BUILD_NUMBER}",
    body: """
        <p>This is a Jenkins starbucks CI/CD pipeline status.</p>
        <p>Project: ${env.JOB_NAME}</p>
        <p>Build Number: ${env.BUILD_NUMBER}</p>
        <p>Build Status: ${buildStatus}</p>
        <p>Started by: ${buildUser}</p>
    """
}
}
}

```

```

<p>Build URL: <a href="${env.BUILD_URL}">${env.BUILD_URL}</a></p>
      '',
      to: 'singaredyv91@gmail.com',
      from: 'singareddyv91@gmail.com',
      replyTo: 'singareddyv91@gmail.com',
      mimeType: 'text/html',
      attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
    }
  }
}

}

```

Build The Pipeline Successfully:

The screenshot shows the Jenkins pipeline interface for the 'Amazon Deploy' job. The pipeline consists of several stages:

- Declarative Tool Install:** Duration: 214ms
- clean workspace:** Duration: 345ms
- Checkout from Git:** Duration: 4s
- SonarQube Analysis:** Duration: 23s (status: In Progress)
- quality gate:** Duration: 390ms
- Install Dependencies:** Duration: 17s
- TRIVY FS SCAN:** Duration: 2s
- Free Disk Space:** Duration: 4s
- Docker Build & Push:** Duration: 1min 35s
- App Deploy to Docker container:** Duration: 717ms (status: Passed)
- Declarative Post Actions:** Duration: 4s

SonarQube Quality Gate:

- starbucks: Passed
- seneca-processing: Success

Build History:

- Today: #5 15:46 (Passed), #4 13:38 (Success), #3 13:16 (Success), #2 12:12 (Success), #1 12:12 (Success)

Permalinks:

- Last build (#5), 3 min 33 sec ago
- Last stable build (#3), 26 min ago
- Last successful build (#3), 26 min ago
- Last failed build (#4), 3 min 33 sec ago
- Last unsuccessful build (#4), 3 min 33 sec ago
- Last completed build (#4), 3 min 33 sec ago



SonarQube Dashboard with Starbucks Project:

The screenshot shows the SonarQube interface with the Starbucks project selected. The dashboard displays various metrics and analysis results for the Starbucks project.

Filters (Left Side):

- Quality Gate:** Passed (1), Failed (0)
- Reliability (Bugs):** A rating (0), B rating (0), C rating (0), D rating (1), E rating (0)
- Security (Vulnerabilities):** A rating (0), B rating (1), C rating (0), D rating (0), E rating (0)

Project Overview:

- starbucks (Passed):** Last analysis: 3 minutes ago
- Bugs:** 26 (D)
- Vulnerabilities:** 1 (B)
- Hotspots Reviewed:** 0.0% (E)
- Code Smells:** 8 (A)
- Coverage:** 0.0%
- Duplications:** 13.7%
- Lines:** 1.6k (JavaScript...)

Bottom Status Bar:

- 31°C Mostly cloudy
- Search bar
- System icons: ENG IN, 15-07-2025, 19:19 IST

Docker Hub Image Build:

The screenshot shows the Docker Hub repository page for the user 'venkatesh09'. The page lists several Docker images available in the 'venkatesh09' namespace.

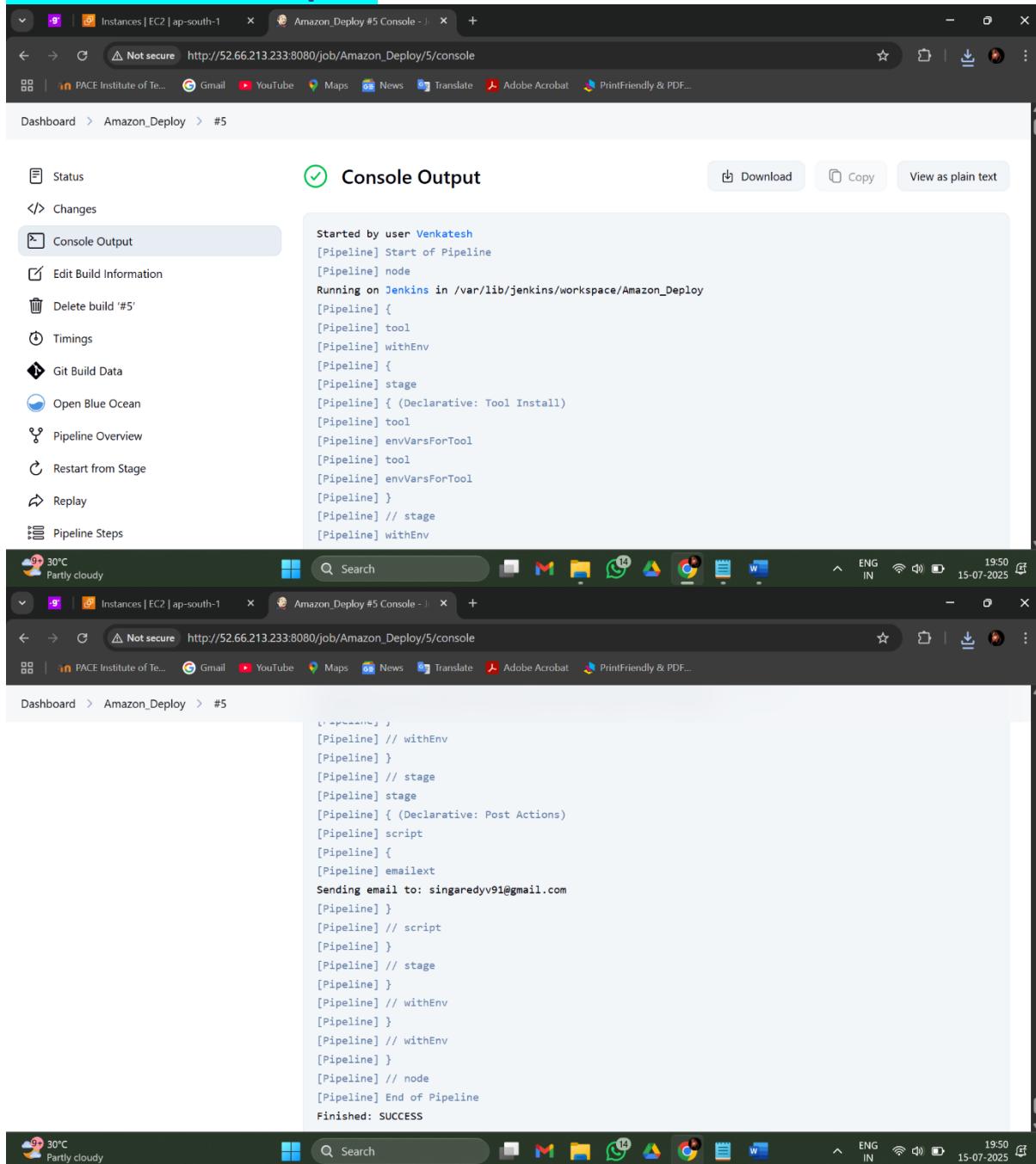
Repositories:

- venkatesh09/starbucks
- venkatesh09/recommendationservice
- venkatesh09/frontend
- venkatesh09/loadgenerator
- venkatesh09/productcatalogservice
- venkatesh09/emailservice
- venkatesh09/checkoutservice
- venkatesh09/adservice

Bottom Status Bar:

- 31°C Mostly cloudy
- Search bar
- System icons: ENG IN, 15-07-2025, 19:20 IST

Console Final Output:



The screenshot shows two identical browser windows displaying Jenkins console output for build #5. The top window shows the initial stages of the pipeline, while the bottom window shows the final stages, including sending an email to singaredyv91@gmail.com.

```
Started by user Venkatesh
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Amazon_Deploy
[Pipeline] {
[Pipeline] tool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Tool Install)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] }

[Pipeline] {
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] script
[Pipeline] {
[Pipeline] emailext
Sending email to: singaredyv91@gmail.com
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

→ Successfully Build All the Stages take <Public IP:3000> Check in your Browser its working or not and check the docker Hub Image → Check SonarQube
→ Email

Final Dashboard: Starbucks Coffee Shop Project

The screenshot shows a web browser window with the URL <http://3.108.56.203:3000/dashboard>. The page features a Starbucks logo in the top left. A yellow banner at the top says "Attention Starbucks Fans! Signature Milkshakes" with a picture of a milkshake. Below it, a text box says "Satisfy your sweet tooth with our signature milkshakes. Indulge today in these coffee-free versions." and "Starting From ₹ 330.00". A green "Order Now" button is visible. The main content area has a heading "Barista Recommends" and three items: "Hazelnut Triangle" (₹ 204.75), "Bhuna Chicken Puff" (₹ 194.25), and "Chocolate Brownie Cupcake" (₹ 288.75). Each item has a "Add Item" button. At the bottom of the screen, the Windows taskbar shows various icons and the date/time as 15-07-2025.

Email Notification Successfully Received:

The screenshot shows a Gmail inbox with 44 unread messages. An email from "singareddyv91@gmail.com" is selected, showing Jenkins pipeline status: "Project: Amazon_Deploy", "Build Number: 3", "Build Status: SUCCESS", "Started by: venkatesh", and a "Build URL: http://13.235.23.102:8080/job/Amazon_Deploy/3/". The email also mentions "One attachment · Scanned by Gmail" with a file named "trivyfs.txt". The Windows taskbar at the bottom shows the date/time as 15-07-2025.