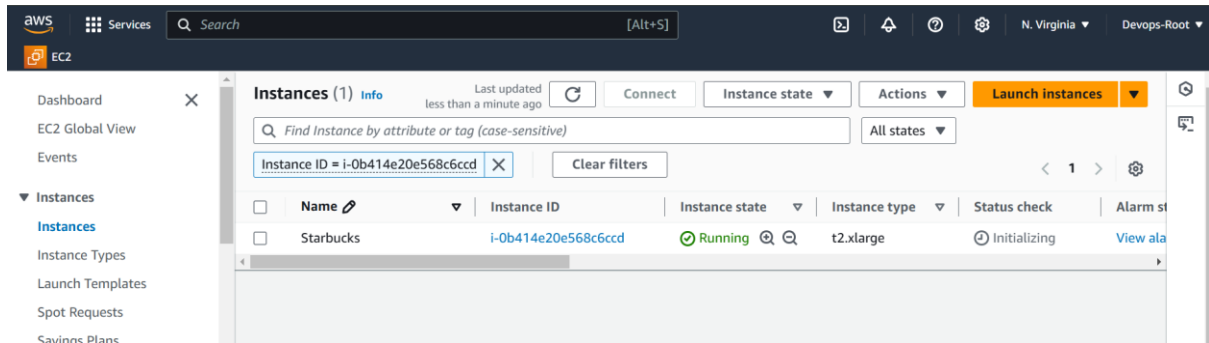


Starbucks-Clone

Step-1: Create EC2 Instance

- Ubuntu
- t2.xlarge with 30GB storage



Step-2: Install Dependencies

- AWS CLI
- Jenkins
- Docker

```
root@ip-172-31-34-82:~#  
root@ip-172-31-34-82:~# aws --version  
aws-cli/2.19.4 Python/3.12.6 Linux/6.8.0-1016-aws exe/x86_64.ubuntu.24  
root@ip-172-31-34-82:~#  
root@ip-172-31-34-82:~# jenkins --version  
2.479.1  
root@ip-172-31-34-82:~#  
root@ip-172-31-34-82:~# docker --version  
Docker version 27.3.1, build ce12230  
root@ip-172-31-34-82:~#  
root@ip-172-31-34-82:~#
```

- Trivy

```
root@ip-172-31-34-82:~# trivy --version  
Version: 0.57.0  
root@ip-172-31-34-82:~#
```

Step-3: Install Docker Scout

- To Install Docker Scout 1st we need to login to DockerHub using credentials

Docker Login

```

root@ip-172-31-34-82:~#
root@ip-172-31-34-82:~# docker login -u vamsi3203
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

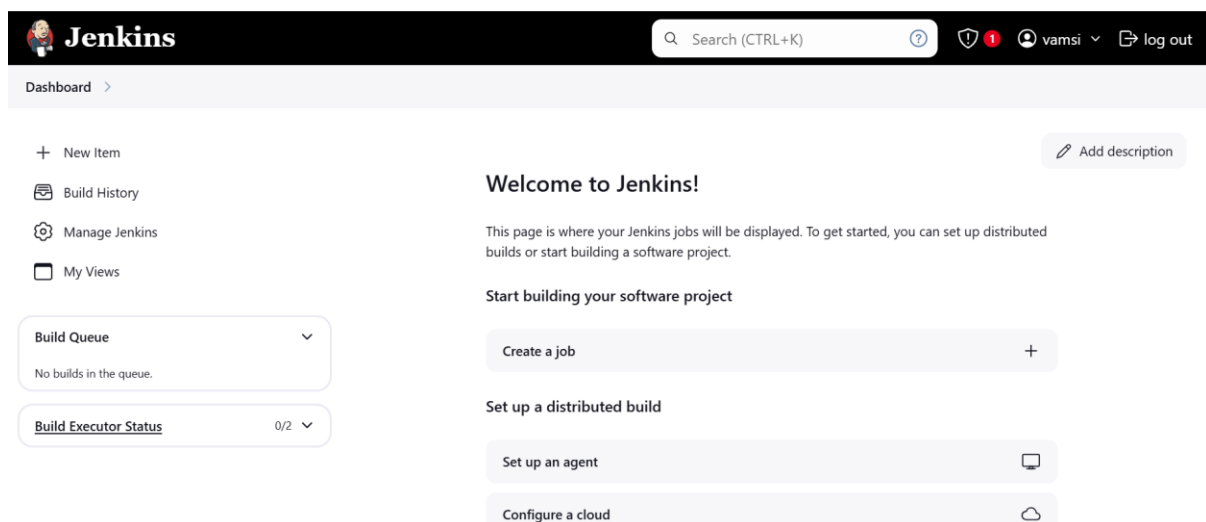
Login Succeeded
root@ip-172-31-34-82:~# █

```

- Install docker-scout

Step-4: Login to Jenkins using EC2 IP

<http://54.204.138.70:8080/>



The screenshot shows the Jenkins dashboard. At the top, there's a header with the Jenkins logo, a search bar, and user information (vamsi) with a log out button. Below the header, the main content area is divided into two columns. The left column contains a sidebar with links: 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. Below these links are two status boxes: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '0/2'). The right column features a 'Welcome to Jenkins!' message, followed by instructions on how to get started. Below this, there are three main action buttons: 'Create a job', 'Set up a distributed build' (which includes 'Set up an agent' and 'Configure a cloud' sub-buttons), and 'Add description'.

Step-5: Install below jenkins plugins

- Eclipse Temurin installer
- SonarQube Scanner
- NodeJS

- Docker
- Docker Commons
- Docker Pipeline
- Docker API
- docker-build-step
- OWASP Dependency-Check
- Email Extension Template
- Blue Ocean
- Pipeline: Stage View

Step-6: Install SonarQube

Using Docker Container:

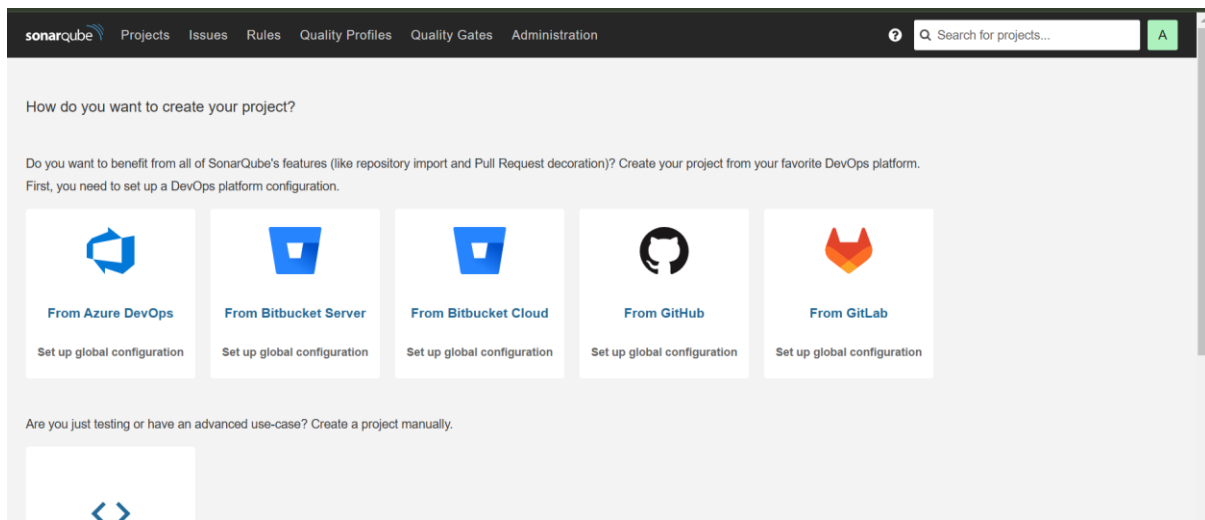
```
docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
```

```

root@ip-172-31-34-82:~# docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
6414378b6477: Pull complete
17da8ec43a12: Pull complete
d12988e90d61: Pull complete
f4d133ca2b7f: Pull complete
143733ae87a4: Pull complete
d021fe0fda1e: Pull complete
bc52ff982b3d: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:aac4b3bdd2ae29c95e1a4c07bc50f0cc81d1914019bc609e92dee25e72ff2e6d
Status: Downloaded newer image for sonarqube:lts-community
afe743886bb0c3f37594d6bddf7388b659ecd0a0771a508a7cb17e2c71b8627e
root@ip-172-31-34-82:~#

```

- Login with admin ID and password and change password



- Generate SonarQube Token for Jenkins

Tokens of Administrator

Generate Tokens

Name Expires in

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

[Copy](#) `squ_031f1db636988ca7139390f7539e045f6947172d`

Name	Type	Project	Last use	Created	Expiration	
jenkins	User		Never	November 9, 2024	December 9, 2024	<input type="button" value="Revoke"/>

Step-7: Add credentials in Jenkins

- Add SonarQube Credential

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

Secret text

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

Secret
.....

ID ?
Sonar-token

Description ?
Sonar-token

Create

- Add DockerHub Credential

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

Username with password

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

Username ?
vamsi3203

☐ Treat username as secret ?

Password ?
.....

ID ?
docker

Create

Step-8: Create a webhook in SonarQube for Jenkins

Configuration ▾ Security ▾ Projects ▾ System Marketplace

Webhooks

Webhooks are used to notify external services when a project analysis is completed. You can configure webhooks to send notifications to each of the provided URLs. Learn more in the [Webhooks document](#).

No webhook defined.

Create Webhook

All fields marked with * are required

Name *
jenkins ✓

URL *
http://54.204.138.70:8080/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:

Step-9: Setup tools in Jenkins

JDK

JDK installations

Add JDK

≡

JDK

Name

jdk17

☒ Install automatically ?

≡

Install from adoptium.net ?

Version ?

jdk-17.0.8.1+1 ▾

Add Installer X

NodeJS

NodeJS

Name

node16

☒ Install automatically ?

≡

Install from nodejs.org

Version

NodeJS 16.20.0

- Add other tools as well

Step-10: configure SonarQube in Jenkins

Dashboard > Manage Jenkins > System >

☐ Environment variables

SonarQube installations

List of SonarQube installations

Name

sonar-server

Server URL

Default is <http://localhost:9000>

<http://54.204.138.70:9000/>

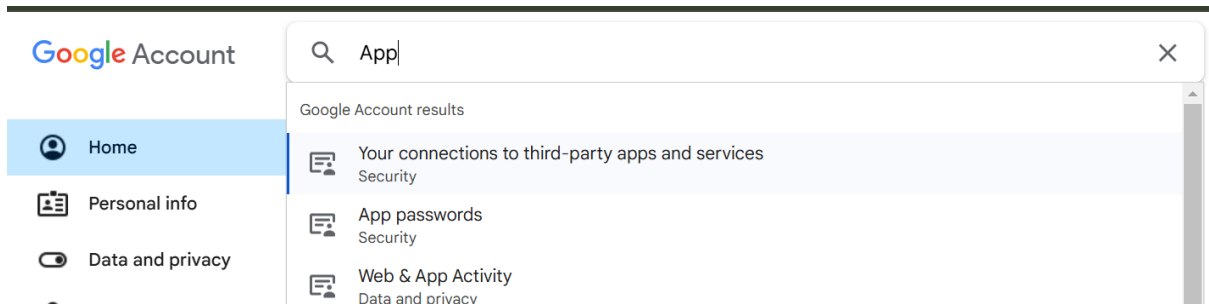
Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

Sonar-token

Step-11: Configure Email Notification to know the build details over Email

- Login to gmail
- Select Manage your google account
- Search for App passwords



- It will ask for login to email ID
- Enter App Name and click on create

You don't have any app passwords.

To create a new app-specific password, type a name for it below...

App name

jenkins

Create

- Add this password under Jenkins credentials



mail-cred

learning.vamsi3203@gmail.com/****** (mail-cred)

Username with password

mail-cred



- SetUp Email Notification in Jenkins

Dashboard → Manage Jenkins → System → Extended E-mail Notification

Extended E-mail Notification

SMTP server

smtp.gmail.com

SMTP Port

465

- Click in Advance, Select email credentials
- Select **Use SSL** and **Use OAuth 2.0**

Advanced ^

 Edited

Credentials

learning.vamsi3203@gmail.com/***** (mail-cred)

+ Add

- ☒ Use SSL
- ☐ Use TLS
- ☒ Use OAuth 2.0

Under E-mail Notification

- Select SMTP server: smtp.gmail.com
- Click on advanced and Enter User name as email and password is the app password which we created for gmail
- Click on **Use SSL and** enter port as 465

☒ Use SMTP Authentication ?

User Name

learning.vamsi3203@gmail.com

 For security when using authentication it is recommended to enable either TLS or SSL

Password

.....

☒ Use SSL ?

☐ Use TLS

SMTP Port ?

465

- Enter Email ID and test configuration

☒ Test configuration by sending test e-mail

Test e-mail recipient

learning.vamsi3203@gmail.com

Email was successfully sent

Test configuration

- Under Build Triggers select the following

Always, Failure – Any and Success

Default Triggers ?

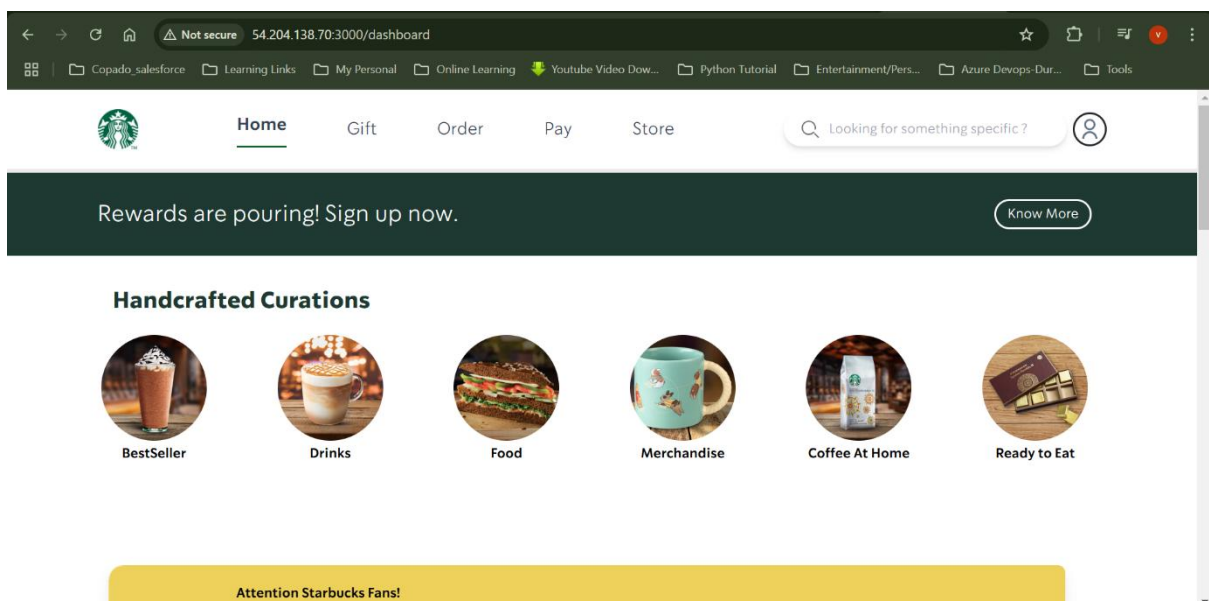
☐ Aborted

☒ Always

☐ Before Build

Step-12: Crate Jenkins job

Execute the build, post successful execution of build we will be able view the application



 Add description

Average stage times:
(Average full run time: ~29min
17s)

	Declarative: Tool Install	clean workspace	Git checkout	Sonarqube Analysis	quality gate	Install NPM Dependencies	OWASP FS SCAN	Trivy File Scan	Build Docker Image	Tag & Push to DockerHub	Docker Scout to Image	Deploy to Container	Declarative: Post Actions
Average stage times: (Average full run time ~29min 17s)	27s	412ms	1s	22s	444ms	27s	25min 1s	17s	47s	21s	1min 24s	810ms	435ms
<div> <div>40s</div> <div>Nov 08 14:41</div> <div> <div>Task</div> <div>Changes</div> </div> </div>	27s	412ms	1s	22s	444ms	27s	25min 1s	17s	47s	21s	1min 24s	810ms	435ms

starbucks **Passed**
server-side processing: **Success**

 Latest Dependency-Check

- Last build (#1), 39 min ago
- Last stable build (#1), 39 min ago
- Last successful build (#1), 39 min ago