JENKINS – CI/CD PIPELINE FOR CUSTOMER SERVICE APPLICATION MICROSERVICES (CAPSTONE PROJECT)

- **Harsini A M(135963)**

JENKINS

- Jenkins is an open-source automation server used to automate the building, testing, and deploying of software projects.
- It is widely used for Continuous Integration (CI) and Continuous Delivery (CD).

Key Features:

- **Automation**: Automates repetitive tasks like building and testing code.
- **Extensibility**: Supports numerous plugins to extend functionality (e.g., Git, Docker, Maven).
- **Distributed Builds**: Can run builds on multiple nodes (machines) to distribute workload.

WEB HOOK AND Poll SCM

- **Webhook and Poll SCM (Source Code Management)** are two different methods in Jenkins for triggering builds based on changes in a version control system like Git. Here's a detailed comparison:

1. Webhook:

- A webhook is an HTTP callback that triggers a build in Jenkins immediately when there's a change in the repository (like a new commit or a pull request).
- The webhook is set up on the version control system's side (e.g., GitHub, GitLab) and sends a POST request to Jenkins when changes occur.

How It Works:

1. When a commit is pushed to the repository, the webhook sends an HTTP request to the specified URL in Jenkins.
2. Jenkins receives the request and triggers a build for the relevant job.

Advantages:

- Real-Time Triggers: Builds are triggered almost instantly when changes are pushed, reducing the delay between commits and builds.
- Efficient: No need for Jenkins to repeatedly check for changes, saving resources.

Setup:

Requires configuration on both Jenkins and the version control system (e.g., GitHub settings to add a webhook pointing to Jenkins).

2. Poll SCM:

- Poll SCM is a method where Jenkins periodically checks the version control system for changes in the repository.
- Jenkins uses a cron-like syntax to determine how often it should poll the repository for changes.

How It Works:

- Jenkins periodically checks the repository (e.g., every minute, every hour) to see if there have been any new commits since the last build.
- If changes are detected, Jenkins triggers a build.

Advantages:

- Simplicity: Easier to set up because it only requires configuration on the Jenkins side; no changes needed on the repository side.
- Control: You can control the frequency of checks, which might be beneficial if you don't need real-time builds.

Disadvantages:

- Latency: Builds may be delayed depending on the polling frequency. For example, if you poll every 5 minutes, there could be up to a 5-minute delay.
- Resource-Intensive: Frequent polling can be resource-intensive, especially for large repositories or multiple jobs.

Setup:

Configured directly in Jenkins under the job settings with a specific polling schedule.

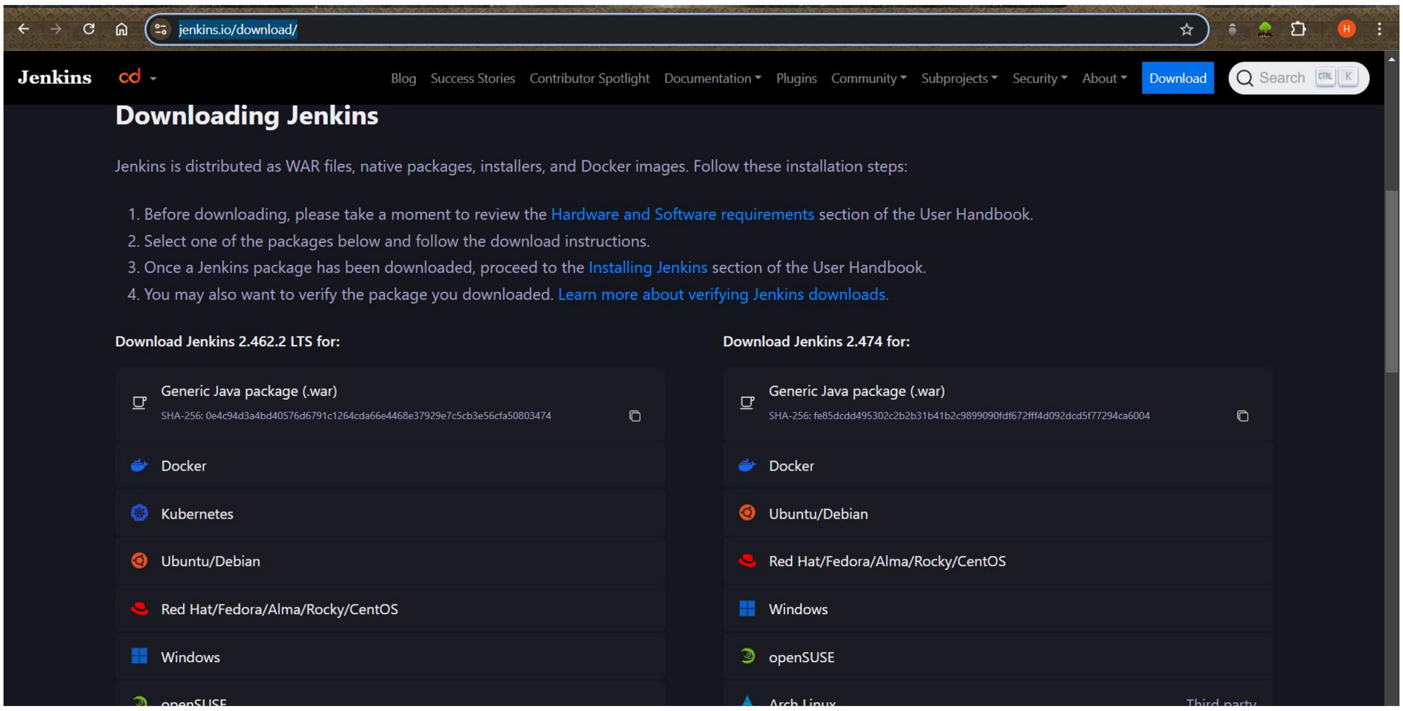
JENKINS INSTALLATION

PREREQUISITES

Java jdk-17

STEPS:

1. Download the Jenkins.war file from the link https://www.jenkins.io/download/



Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images. Follow these installation steps:

1. Before downloading, please take a moment to review the Hardware and Software requirements section of the User Handbook.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the Installing Jenkins section of the User Handbook.
4. You may also want to verify the package you downloaded. Learn more about verifying Jenkins downloads.

Download Jenkins 2.462.2 LTS for:

Generic Java package (.war)
SHA-256: 0e4c94d3a4bd40576d6791c1264cda66e4468e37929e7c5cb3e56cfa50803474

Docker

Kubernetes

Ubuntu/Debian

Red Hat/Fedora/Alma/Rocky/CentOS

Windows

openSUSE

Download Jenkins 2.474 for:

Generic Java package (.war)
SHA-256: fe85dcdd495302c2b2b31b41b2c9899090fdf672fff4d092dcd5f77294ca6004

Docker

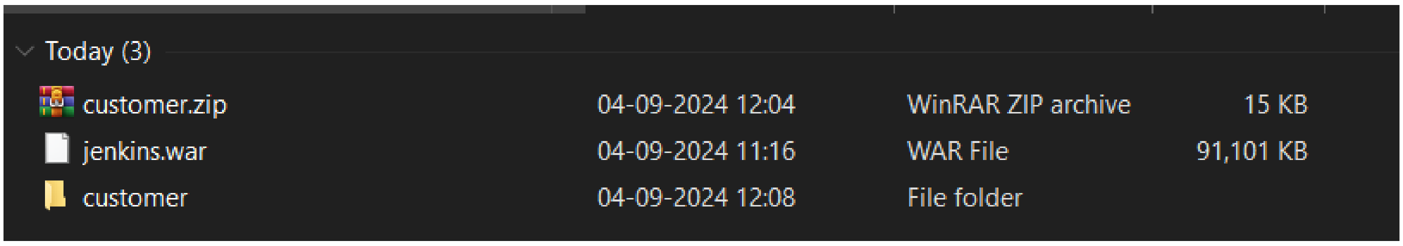
Ubuntu/Debian

Red Hat/Fedora/Alma/Rocky/CentOS

Windows

openSUSE

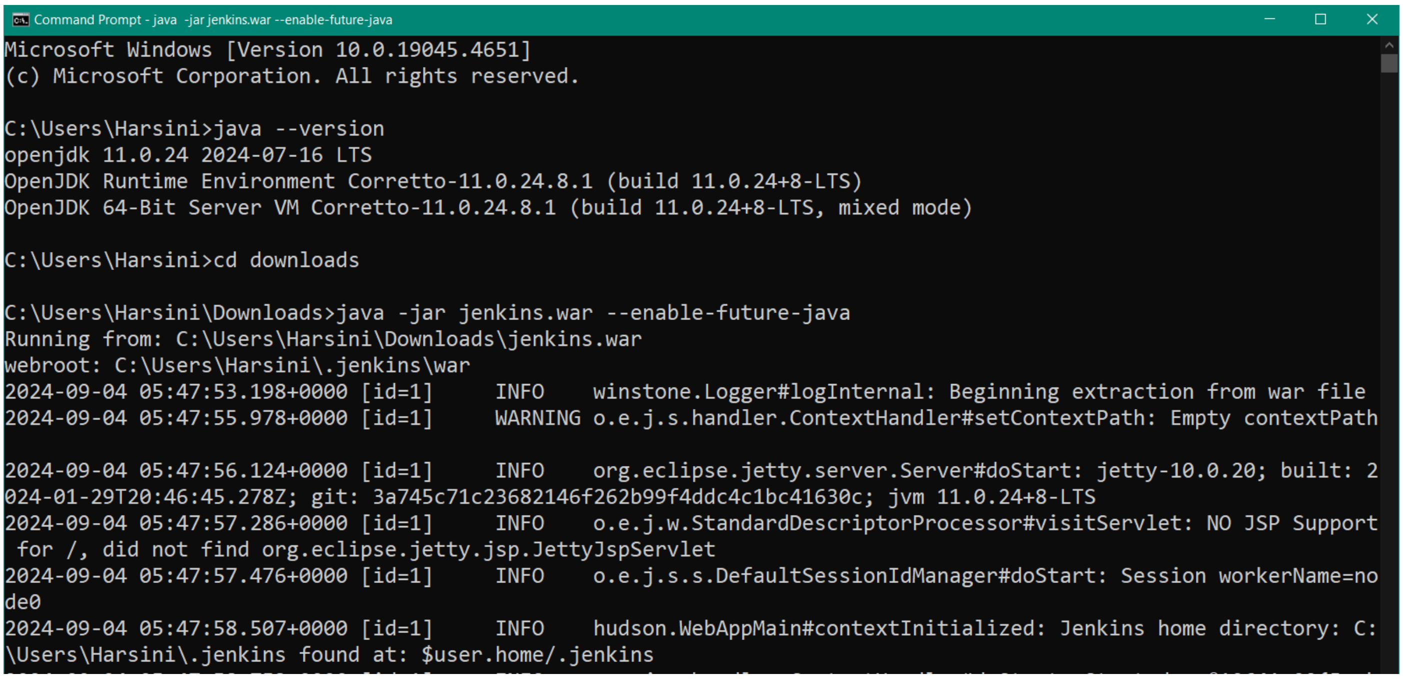
Arch Linux
Third party



```
Today (3)
customer.zip    04-09-2024 12:04    WinRAR ZIP archive    15 KB
jenkins.war     04-09-2024 11:16    WAR File          91,101 KB
customer        04-09-2024 12:08    File folder
```

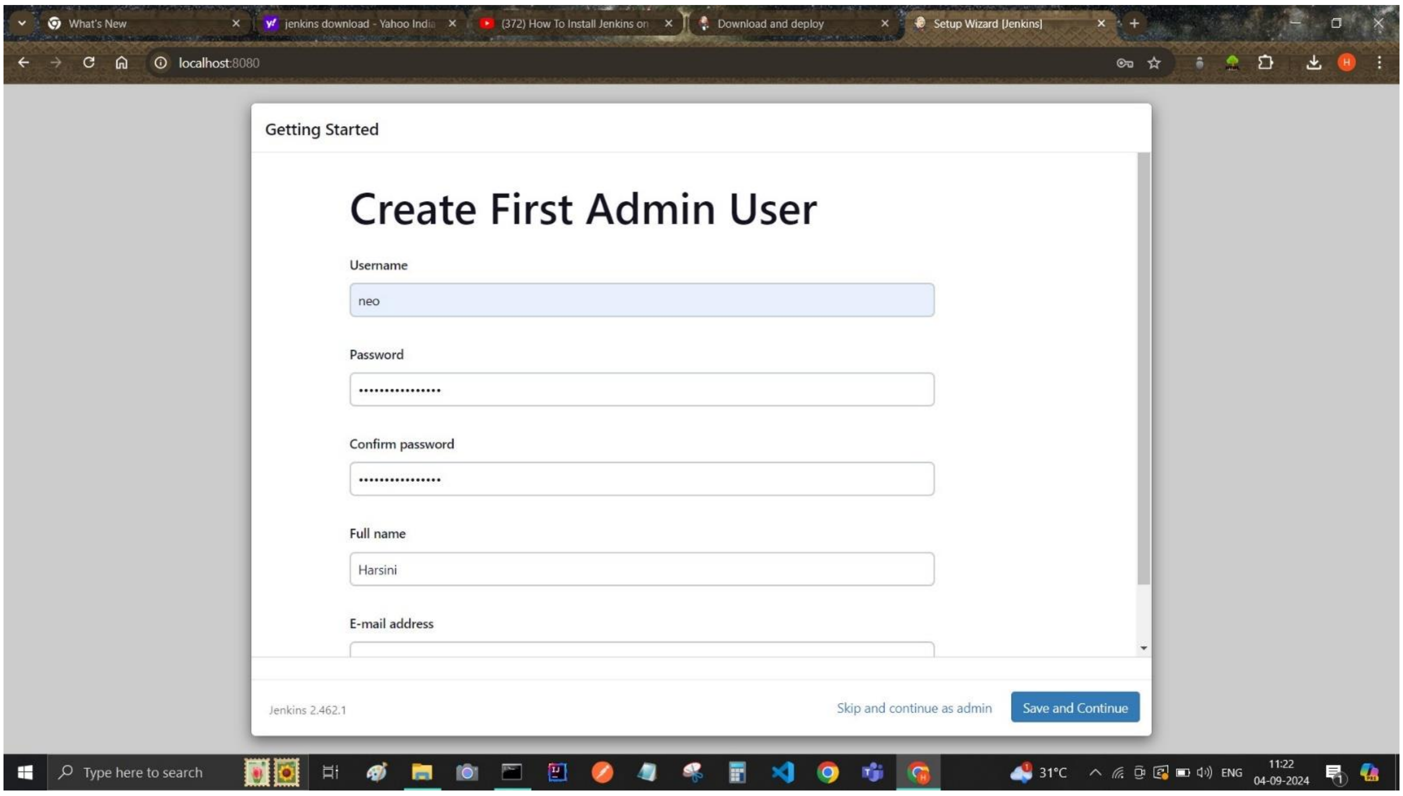
2. Open a command prompt and move to the war file location and run the command

➢ **java -jar Jenkins.war –enable-future-java**



```
Microsoft Windows [Version 10.0.19045.4651]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Harsini>java --version
openjdk 11.0.24 2024-07-16 LTS
OpenJDK Runtime Environment Corretto-11.0.24.8.1 (build 11.0.24+8-LTS)
OpenJDK 64-Bit Server VM Corretto-11.0.24.8.1 (build 11.0.24+8-LTS, mixed mode)

C:\Users\Harsini>cd downloads

C:\Users\Harsini\Downloads>java -jar jenkins.war --enable-future-java
Running from: C:\Users\Harsini\Downloads\jenkins.war
webroot: C:\Users\Harsini\.jenkins\war
2024-09-04 05:47:53.198+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2024-09-04 05:47:55.978+0000 [id=1]     WARNING o.e.j.s.handler.ContextHandler#setContextPath: Empty contextPath

2024-09-04 05:47:56.124+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-10.0.20; built: 2
024-01-29T20:46:45.278Z; git: 3a745c71c23682146f262b99f4ddc4c1bc41630c; jvm 11.0.24+8-LTS
2024-09-04 05:47:57.286+0000 [id=1]     INFO    o.e.j.w.StandardDescriptorProcessor#visitServlet: NO JSP Support
 for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
2024-09-04 05:47:57.476+0000 [id=1]     INFO    o.e.j.s.s.DefaultSessionIdManager#doStart: Session workerName=no
de0
2024-09-04 05:47:58.507+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: C:
\Users\Harsini\.jenkins found at: $user.home/.jenkins
```

3. Go to web browser and visit port 8080(Jenkins usually runs in this port).
4. Click on Installed Plugins to install the default plugins.
5. Give username, password, Full-name and email address.



Create First Admin User

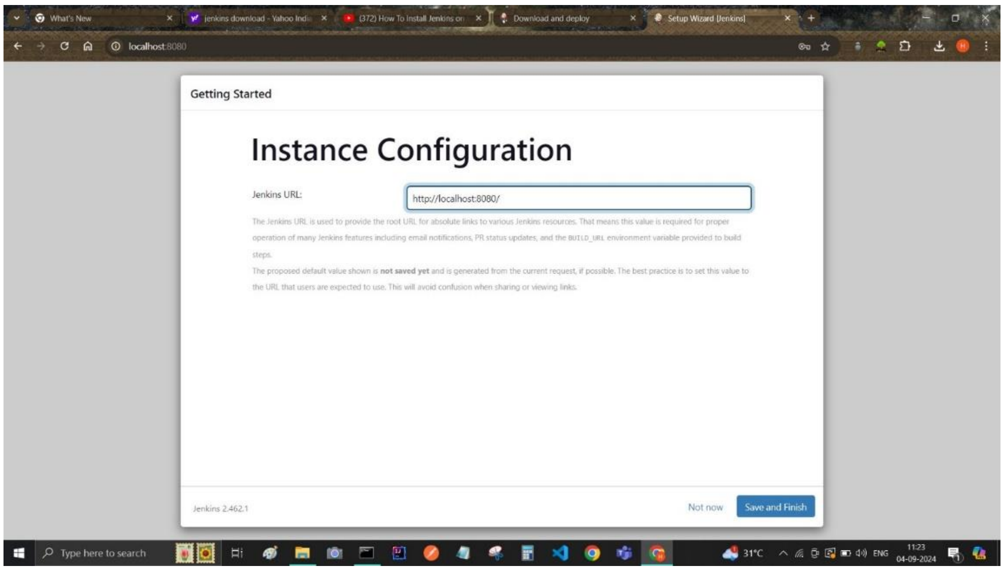
Username: neo

Password: ••••••••••••••••

Confirm password: ••••••••••••••••

Full name: Harsini

E-mail address:


Getting Started

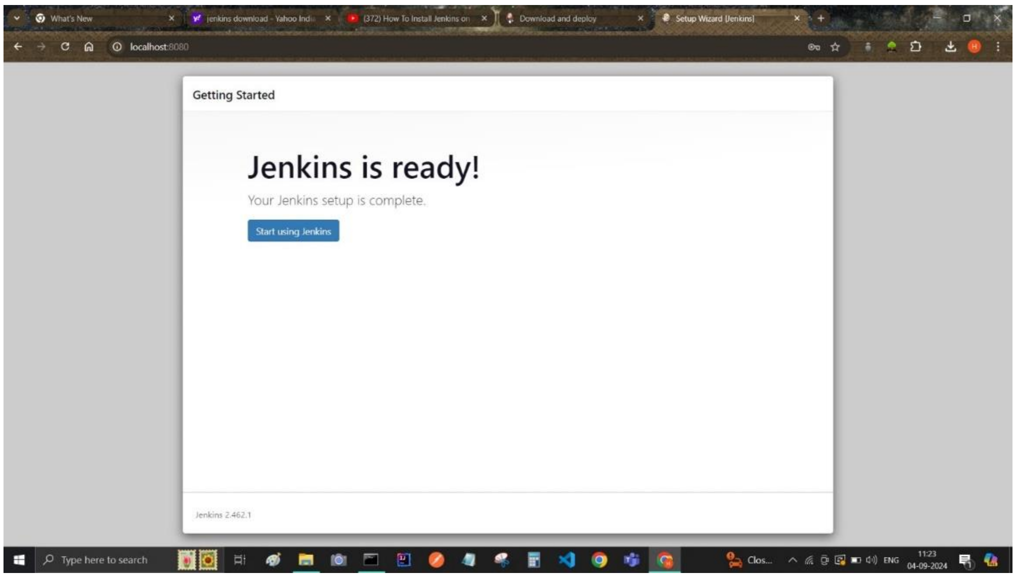
Instance Configuration

Jenkins URL: http://localhost:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.462.1

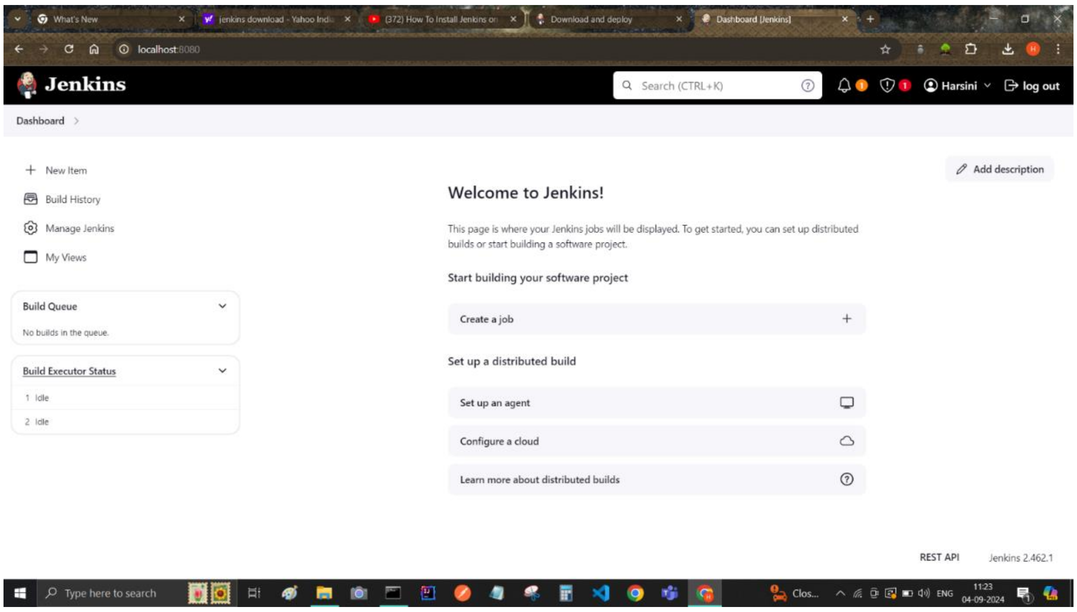

Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

Jenkins 2.462.1

6. Click on Start using Jenkins to complete the setup.


Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

REST API Jenkins 2.462.1

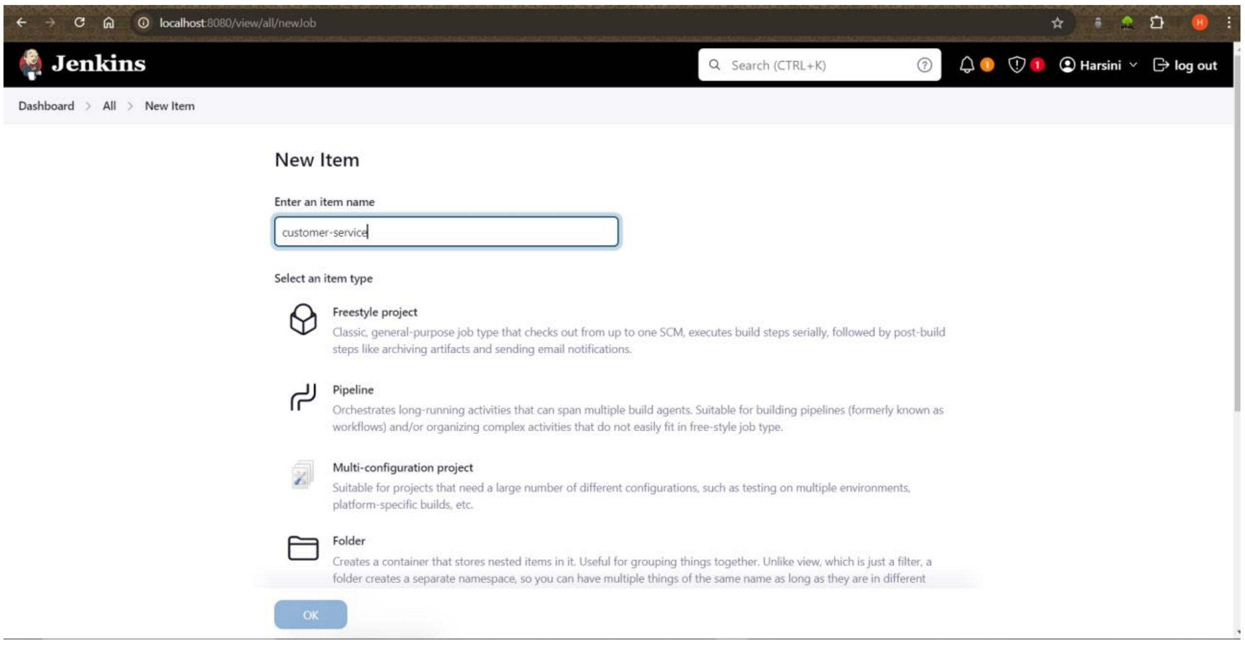
IMPLEMENTATION JENKINS JOB FOR OUR CAPSTONE PROJECT – EMPLOYEE MICROSERVICE

1. **Freestyle Project in Jenkins:**

- Simple & Basic: A Freestyle Project is a basic project type in Jenkins.
- Configuration: You manually configure build steps, like running shell commands, compiling code, or running tests.
- Ease of Use: It's easy to set up and suitable for simple, straightforward jobs.
- Limited Flexibility: While it's quick to configure, it lacks the advanced control and flexibility of a pipeline.
- Best For: Small or single-step tasks that don't require complex workflows or multiple stages.

Using Poll SCM

1. Click on Add New Item to create a new Job.
2. Give a name, click on freestyle project.



New Item

Enter an item name

customer-service

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

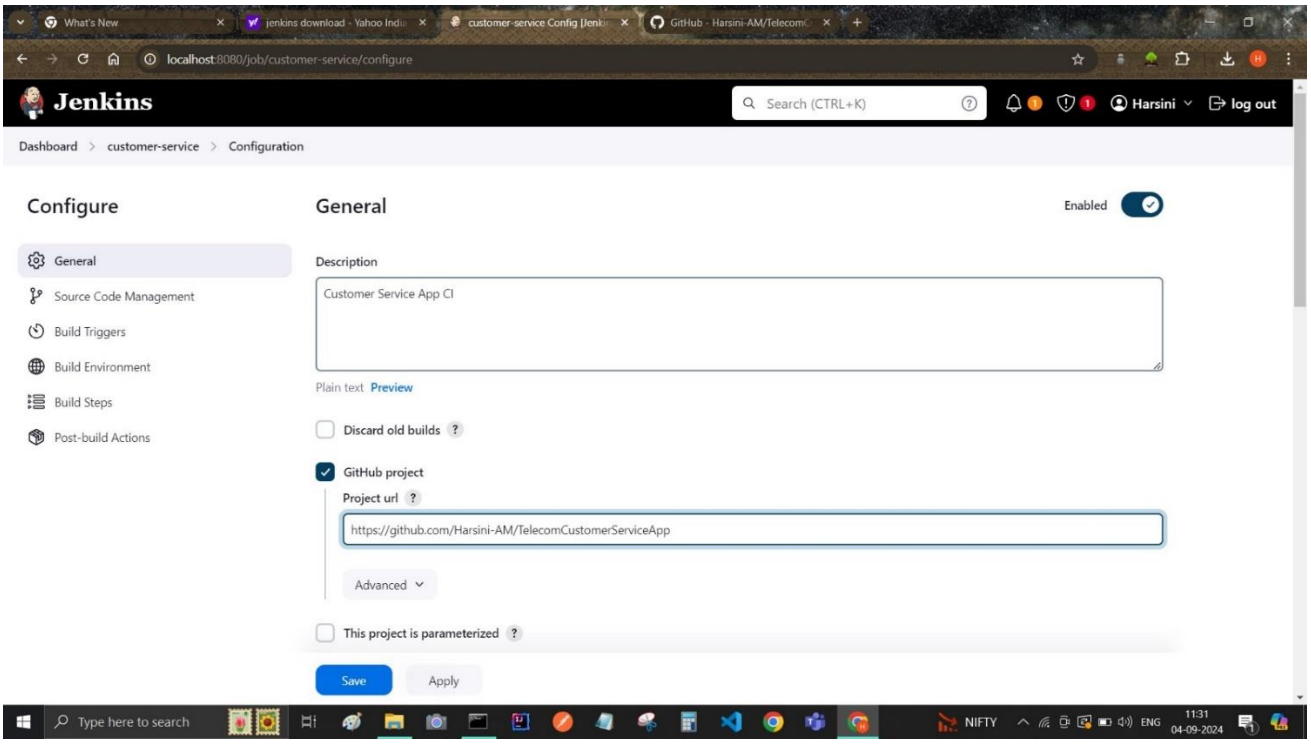
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different

OK

3. Give job – a description, provide the gihub repository URL, select the branch.



Dashboard > customer-service > Configuration

Configure

General

Enabled

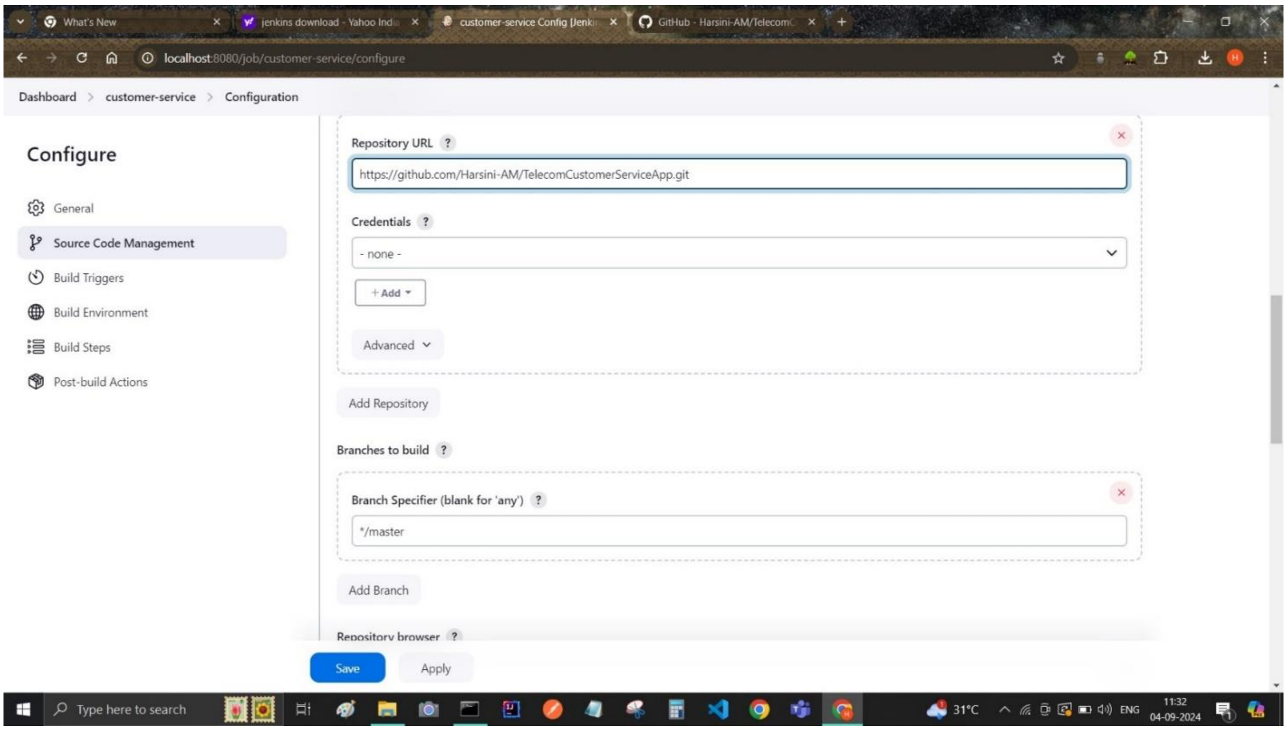
Description: Customer Service App CI

Discard old builds

GitHub project: checked

Project url: https://github.com/Harsini-AM/TelecomCustomerServiceApp

This project is parameterized



Dashboard > customer-service > Configuration

Configure

Repository URL: https://github.com/Harsini-AM/TelecomCustomerServiceApp.git

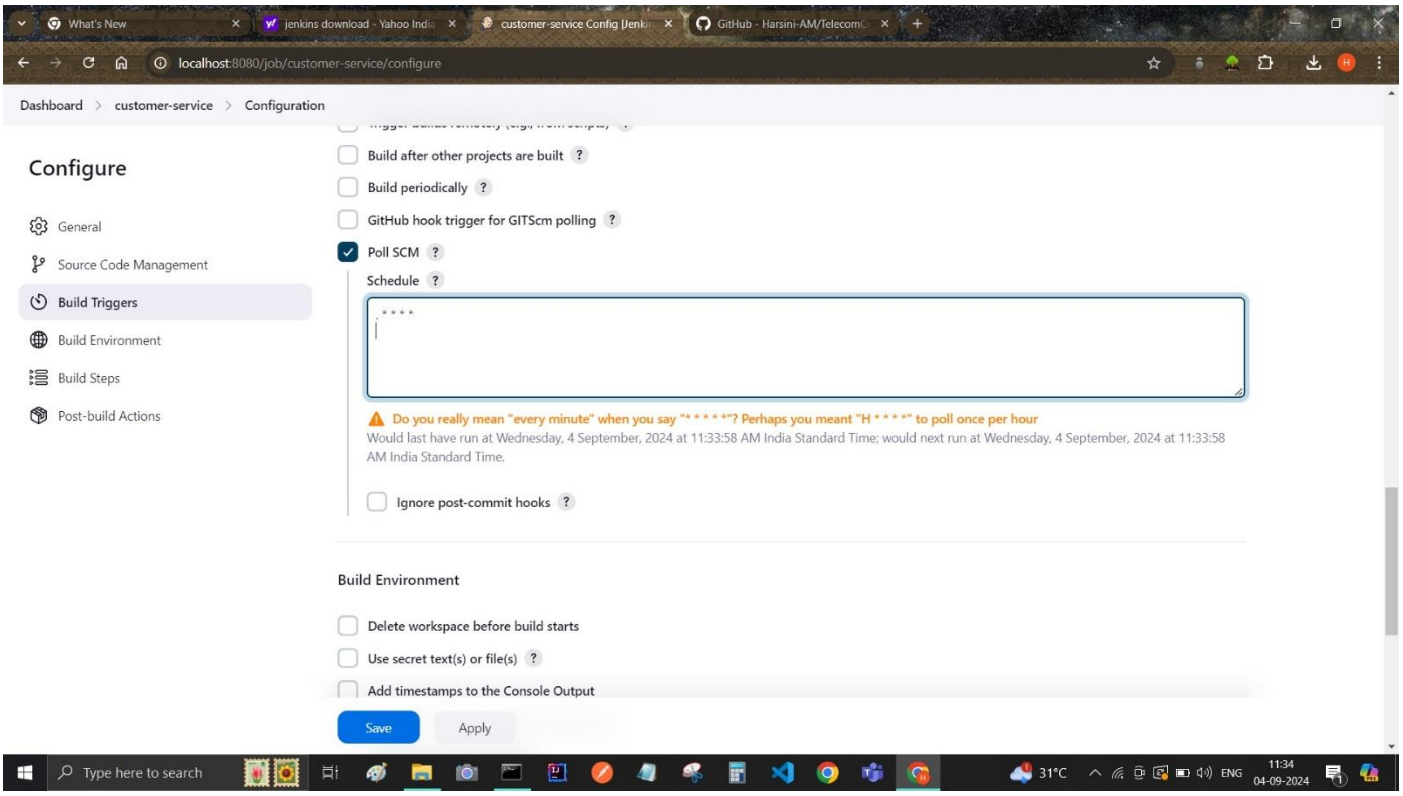
Credentials: - none -

Branches to build

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

Repository browser

4. In build triggers, click on Poll SCM. And provide * * * * * - that will check the github repo for every one minute, if there is any new commit, build will be triggered.


localhost:8080/job/customer-service/configure

Dashboard > customer-service > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [✓] Poll SCM

Schedule

```
* * * *
```

⚠ Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour
Would last have run at Wednesday, 4 September, 2024 at 11:33:58 AM India Standard Time; would next run at Wednesday, 4 September, 2024 at 11:33:58 AM India Standard Time.

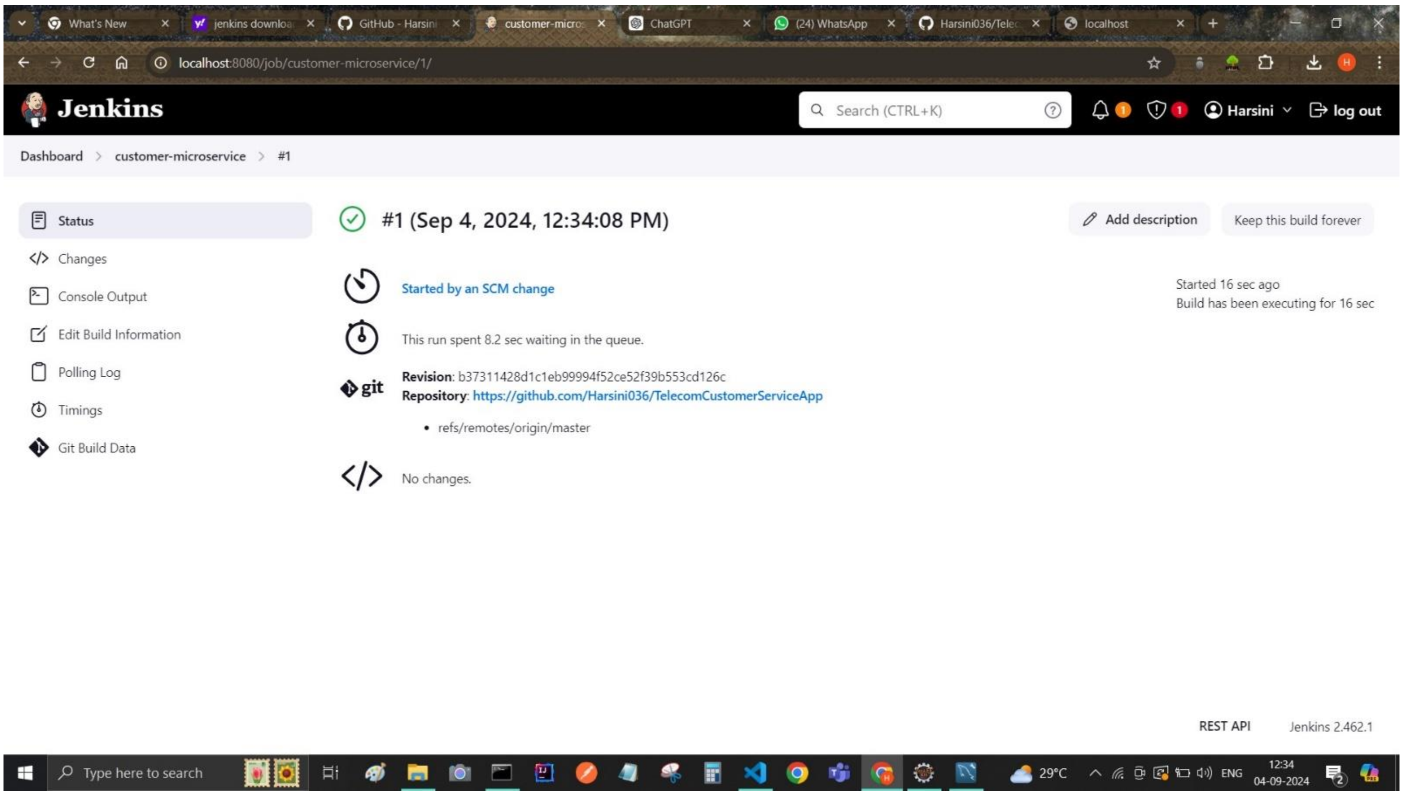
- [ ] Ignore post-commit hooks

Build Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Add timestamps to the Console Output

Save Apply

5. Click on Apply to save the job.
6. Now click on Build Now in the Sidebar or do new commit to the gihub repository.
7. Build will be triggered.
8. The build status – Success or failure will be displayed in the status and details messages can be viewed in console output.


localhost:8080/job/customer-microservice/1/

Jenkins

Dashboard > customer-microservice > #1

- Status
- Changes
- Console Output
- Edit Build Information
- Polling Log
- Timings
- Git Build Data

#1 (Sep 4, 2024, 12:34:08 PM)

Add description Keep this build forever

Started by an SCM change

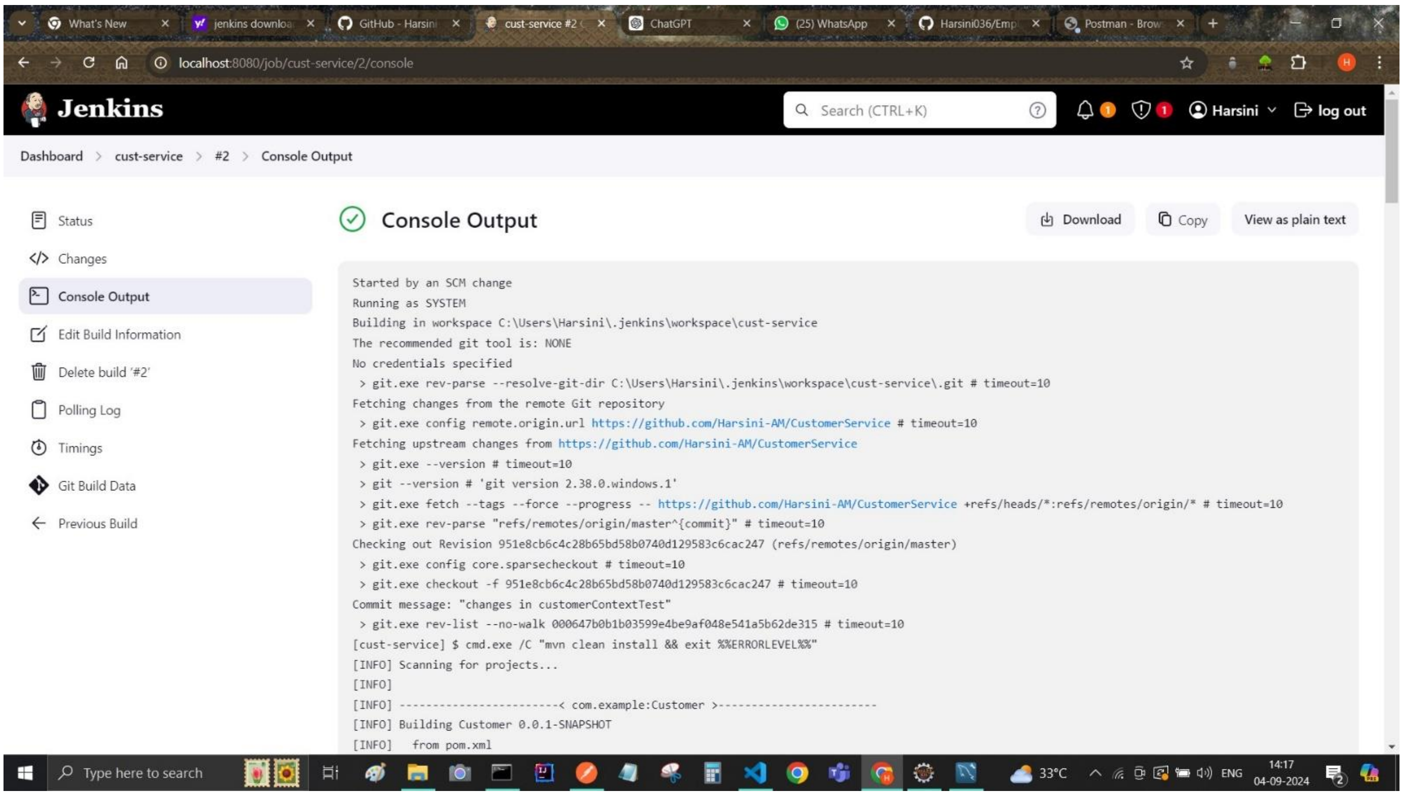
Started 16 sec ago
Build has been executing for 16 sec

This run spent 8.2 sec waiting in the queue.

git
Revision: b37311428d1c1eb99994f52ce52f39b553cd126c
Repository: https://github.com/Harsini036/TelecomCustomerServiceApp
- refs/remotes/origin/master

No changes.

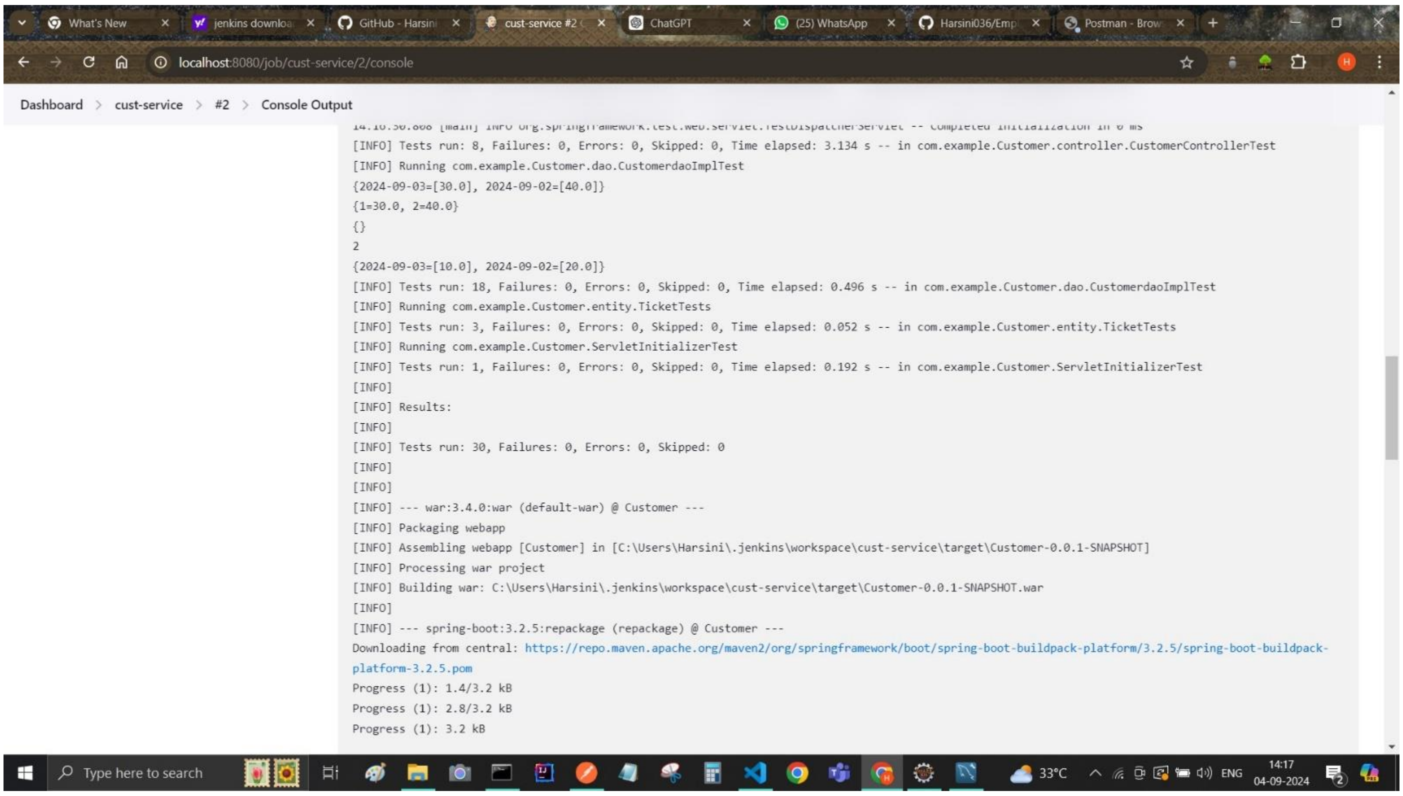
REST API Jenkins 2.462.1


Dashboard > cust-service > #2 > Console Output

Status
Changes
Console Output
Edit Build Information
Delete build '#2'
Polling Log
Timings
Git Build Data
Previous Build

Console Output

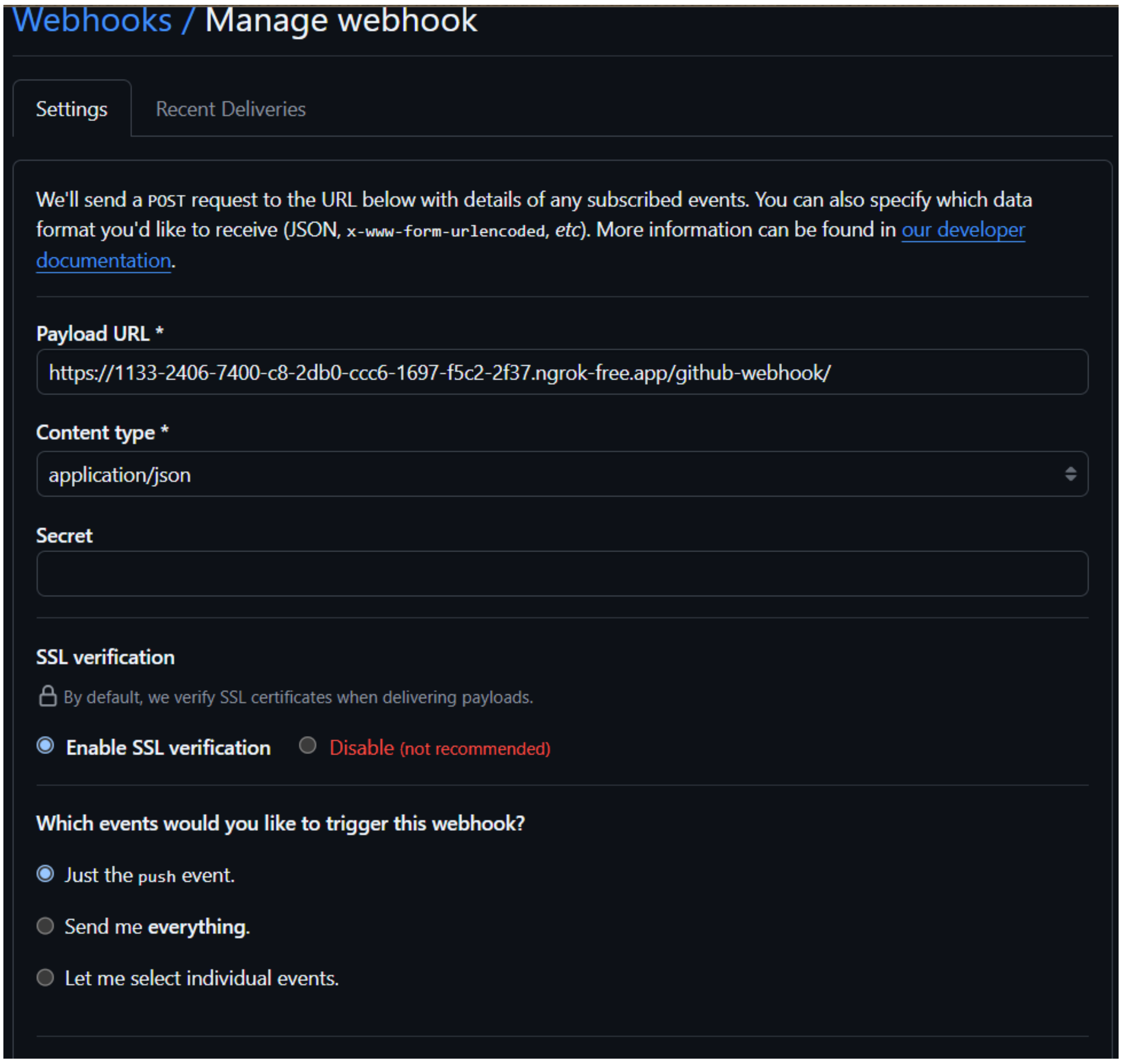
```
Started by an SCM change
Running as SYSTEM
Building in workspace C:\Users\Harsini\.jenkins\workspace\cust-service
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\Users\Harsini\.jenkins\workspace\cust-service\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/Harsini-AM/CustomerService # timeout=10
Fetching upstream changes from https://github.com/Harsini-AM/CustomerService
 > git.exe --version # timeout=10
 > git --version # 'git version 2.38.0.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/Harsini-AM/CustomerService +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision 951e8cb6c4c28b65bd58b0740d129583c6cac247 (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f 951e8cb6c4c28b65bd58b0740d129583c6cac247 # timeout=10
Commit message: "changes in customerContextTest"
 > git.exe rev-list --no-walk 000647b0b1b03599e4be9af048e541a5b62de315 # timeout=10
[cust-service] $ cmd.exe /C "mvn clean install && exit %%ERRORLEVEL%%"
[INFO] Scanning for projects...
[INFO]
[INFO] -----------------------< com.example:Customer >-----------------------
[INFO] Building Customer 0.0.1-SNAPSHOT
[INFO]   from pom.xml
```


Dashboard > cust-service > #2 > Console Output

```
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.134 s -- in com.example.Customer.controller.CustomerControllerTest
[INFO] Running com.example.Customer.dao.CustomerdaoImplTest
{2024-09-03=[30.0], 2024-09-02=[40.0]}
{1=30.0, 2=40.0}
{}
2
{2024-09-03=[10.0], 2024-09-02=[20.0]}
[INFO] Tests run: 18, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.496 s -- in com.example.Customer.dao.CustomerdaoImplTest
[INFO] Running com.example.Customer.entity.TicketTests
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.052 s -- in com.example.Customer.entity.TicketTests
[INFO] Running com.example.Customer.ServletInitializerTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.192 s -- in com.example.Customer.ServletInitializerTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 30, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- war:3.4.0:war (default-war) @ Customer ---
[INFO] Packaging webapp
[INFO] Assembling webapp [Customer] in [C:\Users\Harsini\.jenkins\workspace\cust-service\target\Customer-0.0.1-SNAPSHOT]
[INFO] Processing war project
[INFO] Building war: C:\Users\Harsini\.jenkins\workspace\cust-service\target\Customer-0.0.1-SNAPSHOT.war
[INFO]
[INFO] --- spring-boot:3.2.5:repackage (repackage) @ Customer ---
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-buildpack-platform/3.2.5/spring-boot-buildpack-
platform-3.2.5.pom
Progress (1): 1.4/3.2 kB
Progress (1): 2.8/3.2 kB
Progress (1): 3.2 kB
```

Using WEBHOOK

1. Create a freestyle project and follow same steps as above except, in the build trigger click on **GitHub hook trigger for GITScm polling**
2. In the github repository, go the settings -> WebHook -> Add WebHook
3. Add the Jenkins URL to which the HTTP POST request will be send.
4. Change the content type to application/json.
5. Click on ok.
6. For every commit to the repo, build will be automatically triggered in the Jenkins side.


Webhooks / Manage webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, *etc*). More information can be found in our developer documentation.

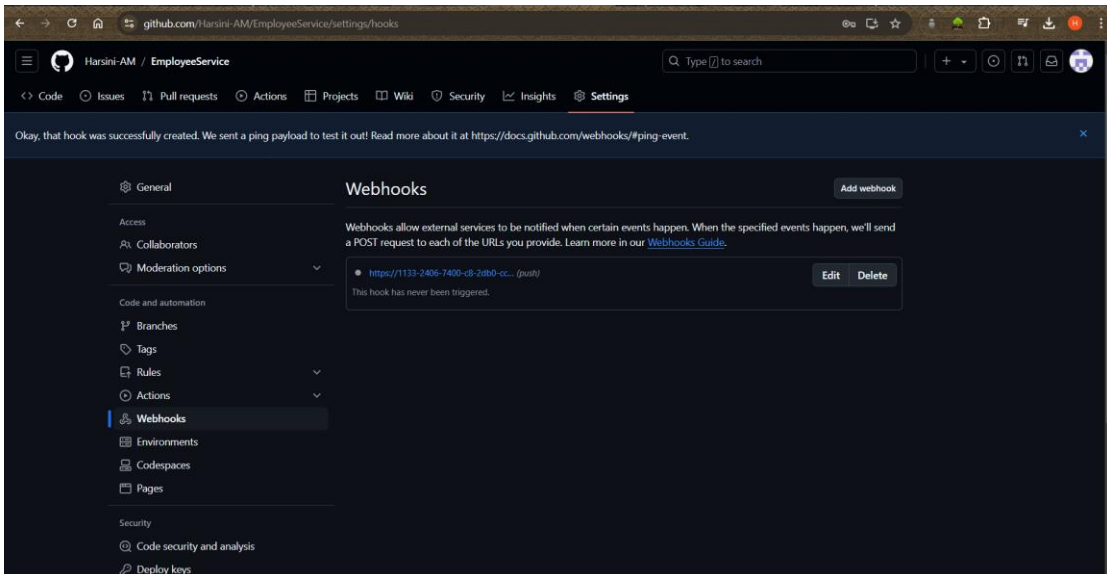
Payload URL *
https://1133-2406-7400-c8-2db0-ccc6-1697-f5c2-2f37.ngrok-free.app/github-webhook/

Content type *
application/json

Secret

SSL verification
By default, we verify SSL certificates when delivering payloads.
● Enable SSL verification ○ Disable (not recommended)

Which events would you like to trigger this webhook?
● Just the push event.
○ Send me everything.
○ Let me select individual events.


github.com/Harsini-AM/EmployeeService/settings/hooks

Okay, that hook was successfully created. We sent a ping payload to test it out! Read more about it at https://docs.github.com/webhooks/#ping-event.

Webhooks

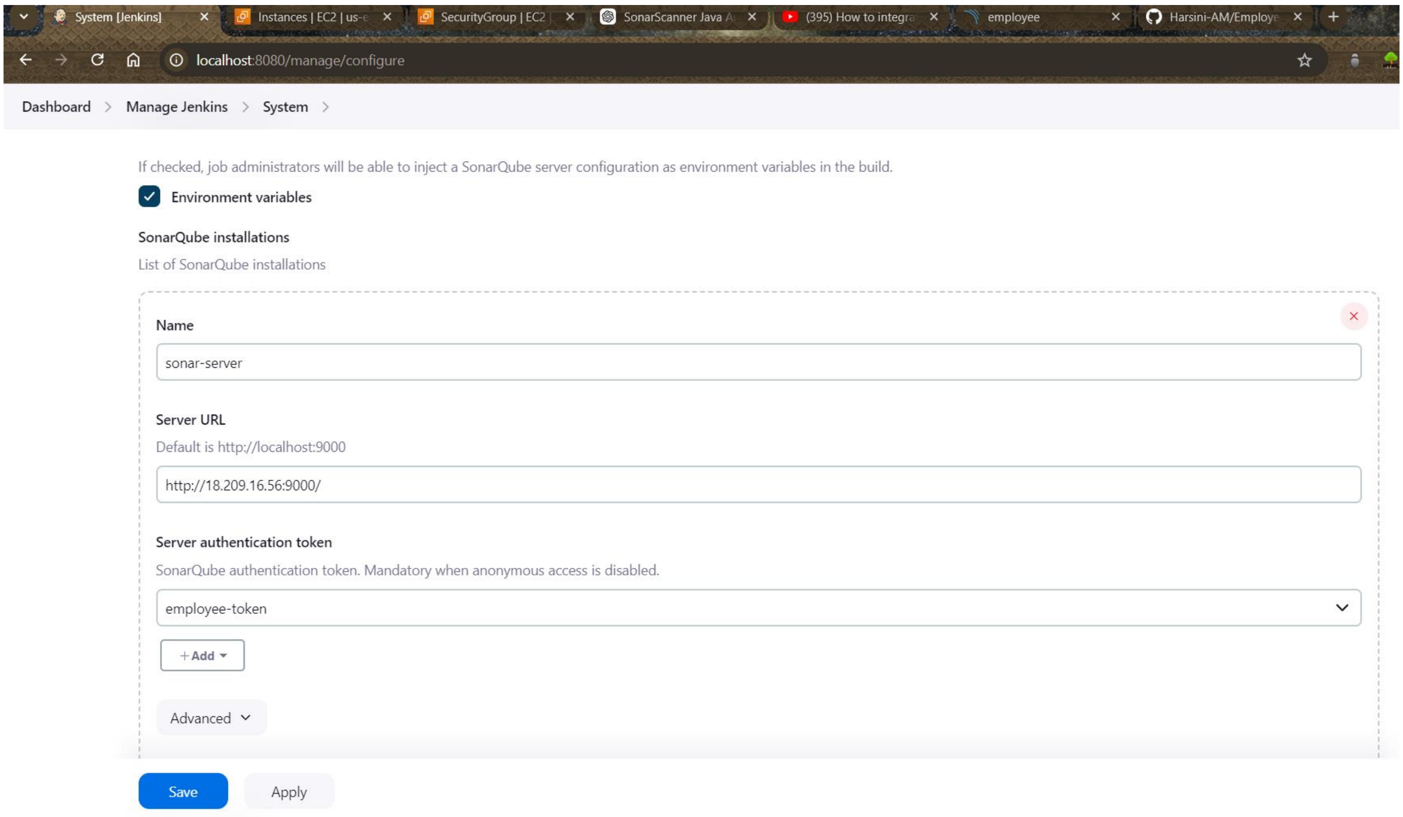
Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

https://1133-2406-7400-c8-2db0-cc... (push)
This hook has never been triggered.

JENKINS PIPELINE – To build, Run Junit test cases, Generate Sonar Report, Building Docker Image, and finally pushing to DockerHub.

Prerequisites:

1. Add Sonar server URL running in an amazon ec2-instance along with the project authentication token under **Manage Jenkins > System**



localhost:8080/manage/configure

Dashboard > Manage Jenkins > System

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

☑ Environment variables

SonarQube installations

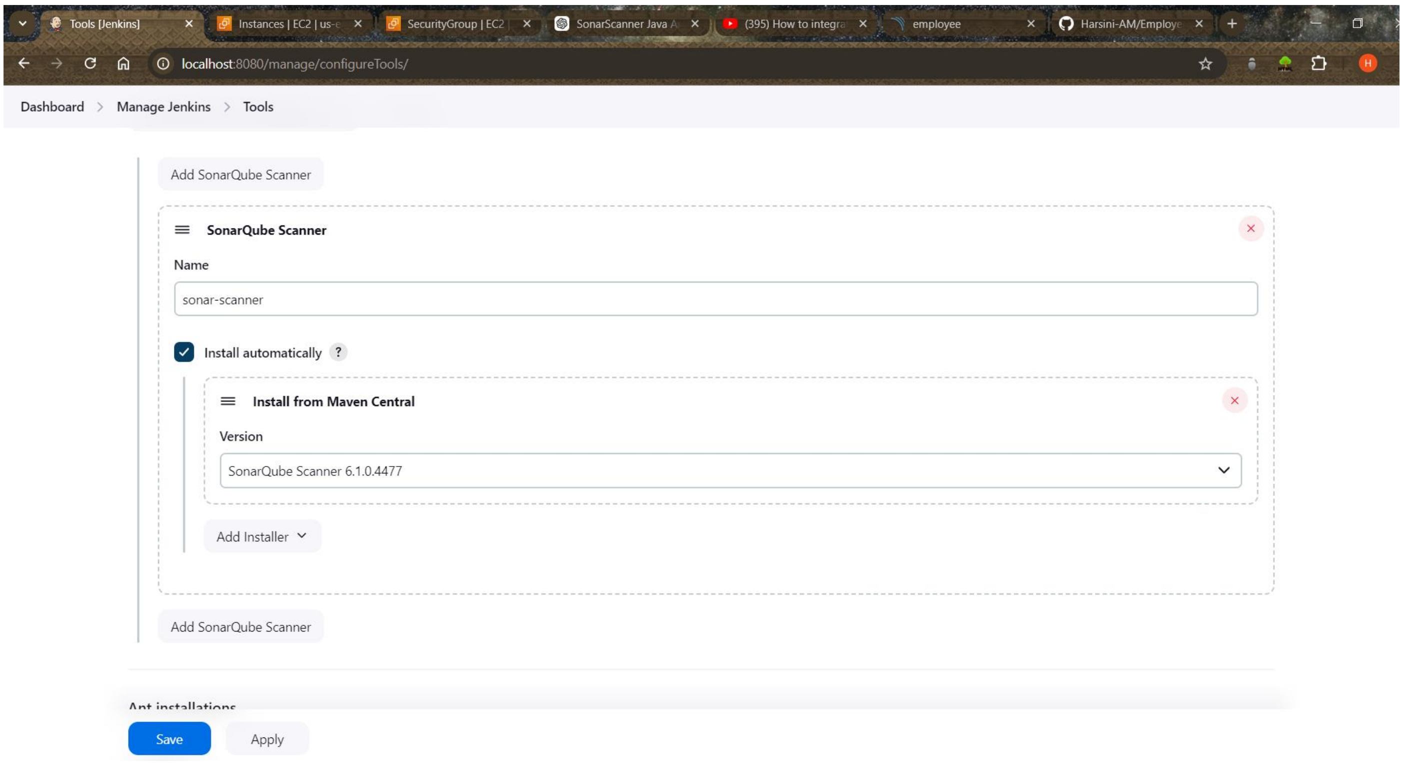
List of SonarQube installations

Name: sonar-server

Server URL (Default is http://localhost:9000): http://18.209.16.56:9000/

Server authentication token (SonarQube authentication token. Mandatory when anonymous access is disabled.): employee-token

2. Install SonarQube Scanner in Jenkins under **Manage Jenkins > Tools.**



localhost:8080/manage/configureTools/

Dashboard > Manage Jenkins > Tools

SonarQube Scanner

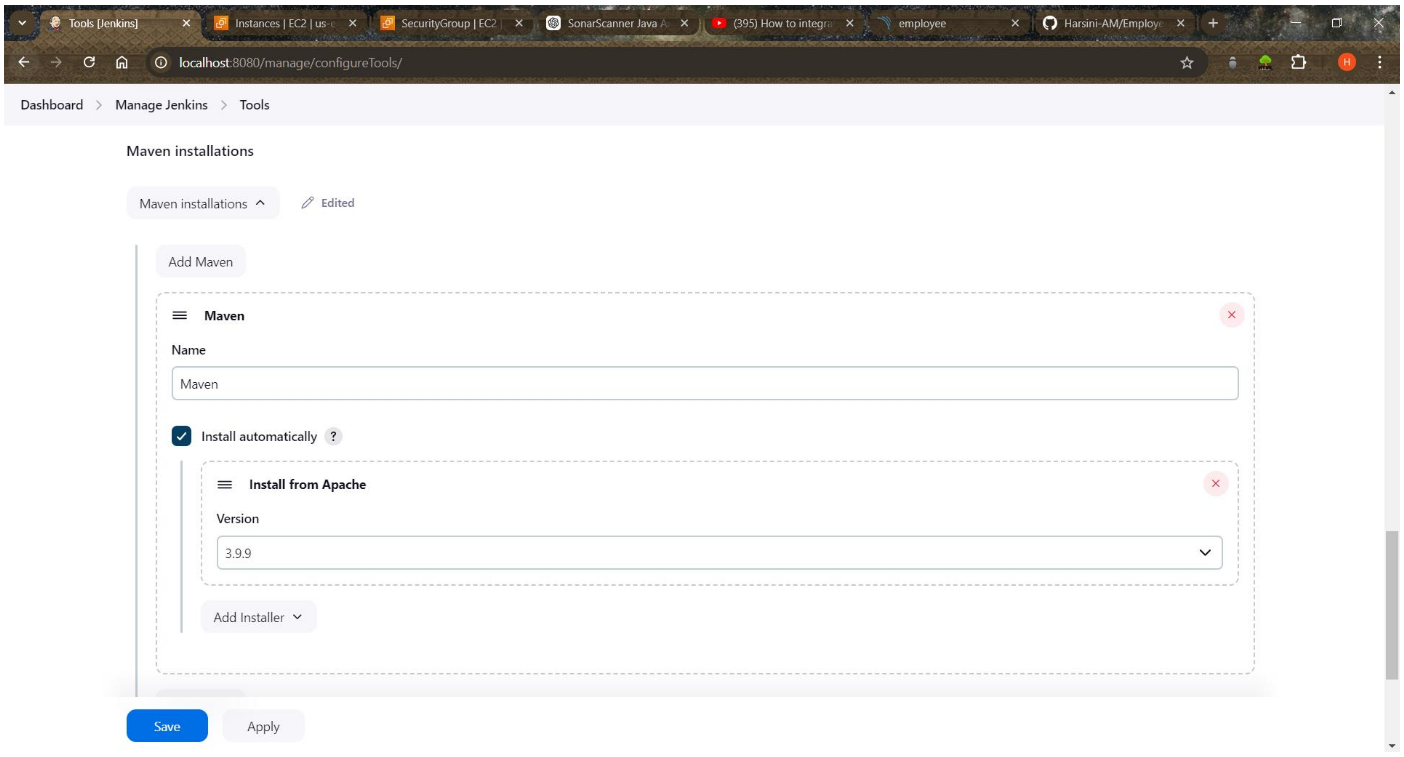
Name: sonar-scanner

☑ Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.1.0.4477

3. Install Maven in Jenkins under **Manage Jenkins > Tools.**


localhost:8080/manage/configureTools/

Dashboard > Manage Jenkins > Tools

Maven installations

Maven

Name: Maven

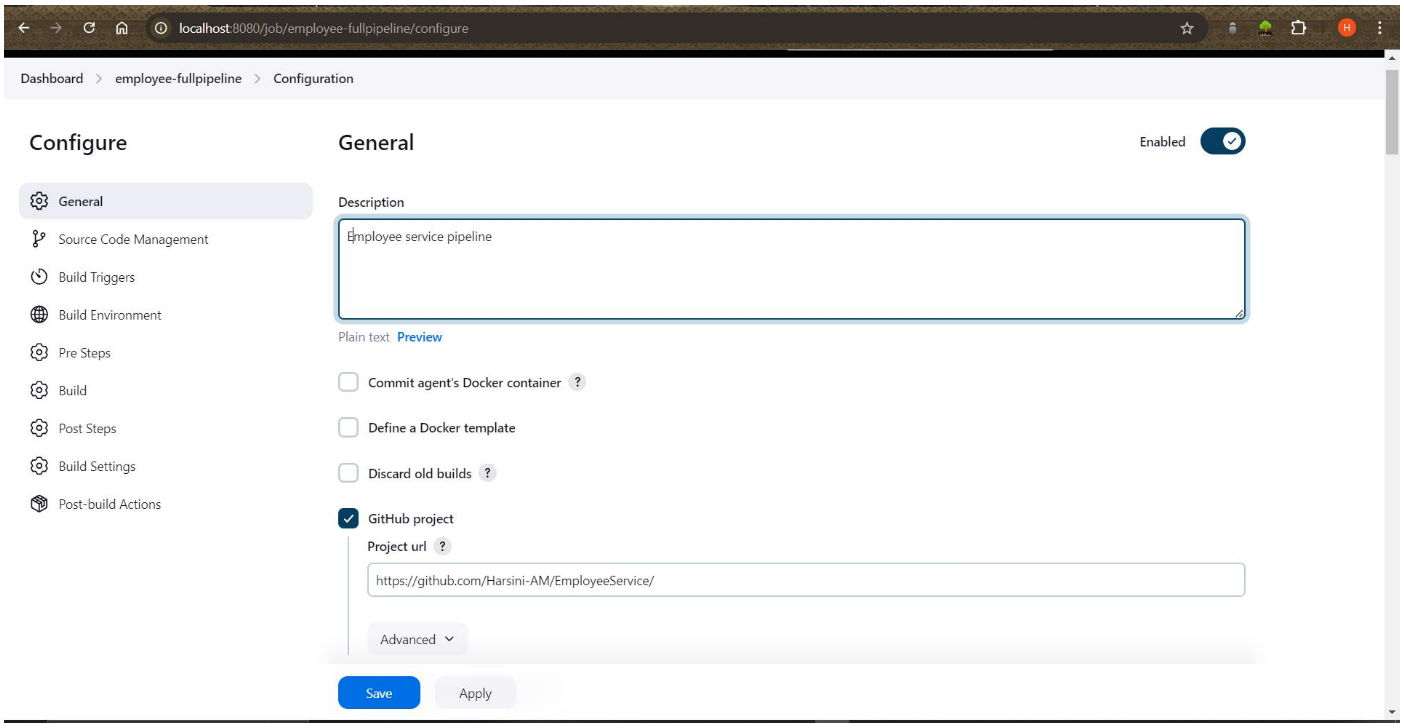
Install automatically ✓

Install from Apache

Version: 3.9.9

A. FREESTYLE PROJECT

1. Click on New Freestyle project


localhost:8080/job/employee-fullpipeline/configure

Dashboard > employee-fullpipeline > Configuration

General — Enabled ✓

Description: Employee service pipeline

Commit agent's Docker container ☐

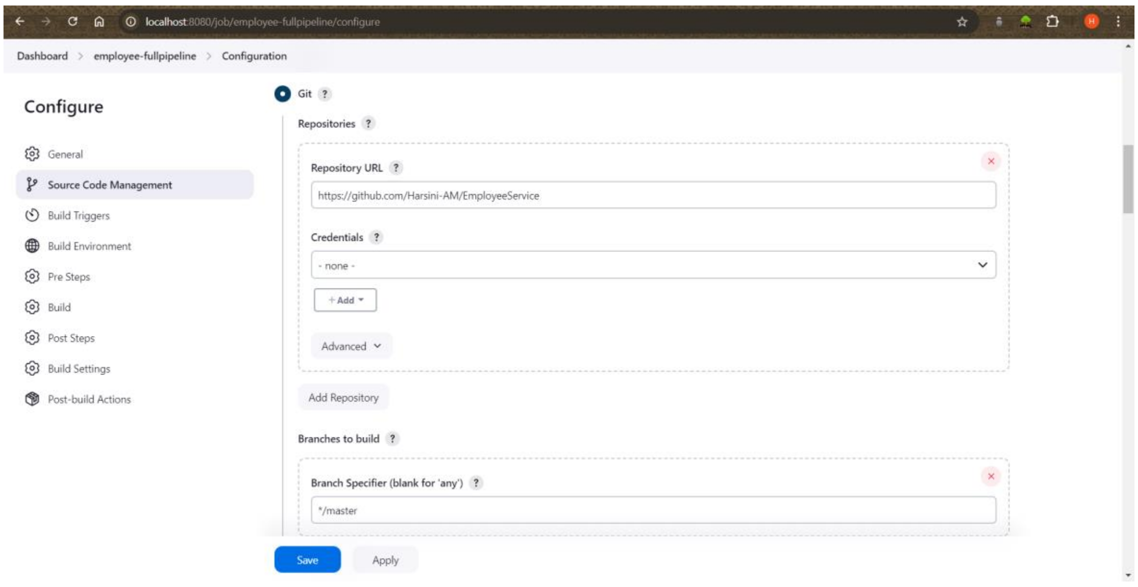
Define a Docker template ☐

Discard old builds ☐

GitHub project ✓

Project url: https://github.com/Harsini-AM/EmployeeService/

2. Give a name for the job, description, provide github URL.


localhost:8080/job/employee-fullpipeline/configure

Dashboard > employee-fullpipeline > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Pre Steps
- Build
- Post Steps
- Build Settings
- Post-build Actions

Git

Repositories

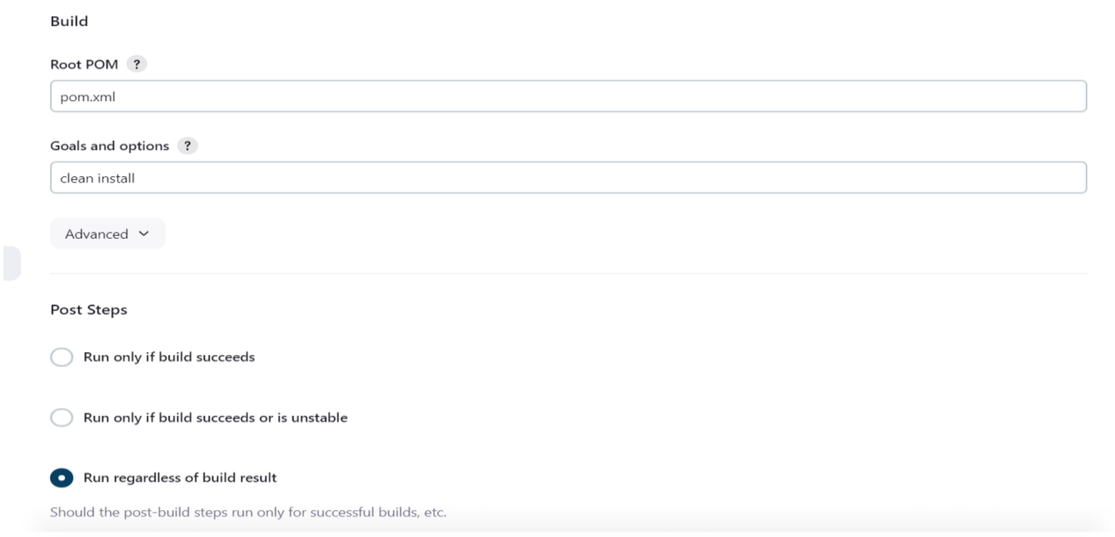
Repository URL: https://github.com/Harsini-AM/EmployeeService

Credentials: - none -

Branches to build

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

3. In the build option, under Goals and option give **clean install.** (to trigger maven build stages)


Build

Root POM: pom.xml

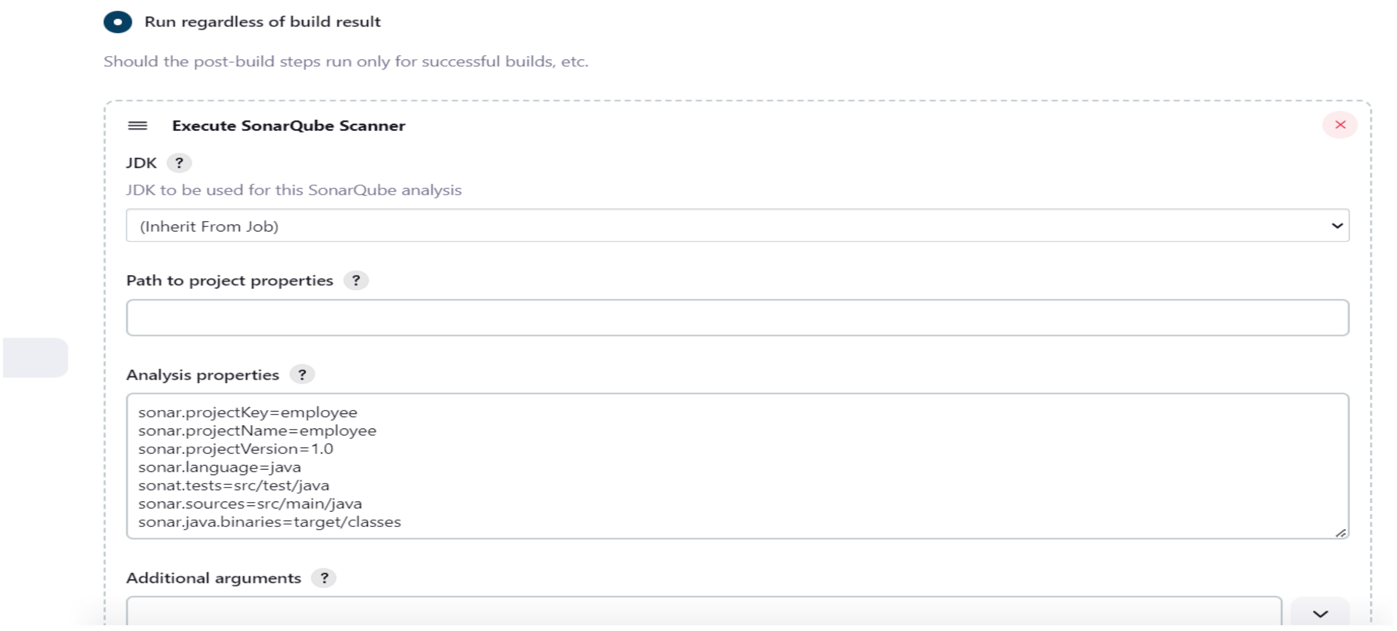
Goals and options: clean install

Post Steps

- Run only if build succeeds
- Run only if build succeeds or is unstable
- [✓ Run regardless of build result]

Should the post-build steps run only for successful builds, etc.

4. Under pre-build actions, click on **Execute SonarQube scanner** and in the **Analysis properties**, we have to provide sonar project name, key, version, test and source file directories.


[✓ Run regardless of build result]

Should the post-build steps run only for successful builds, etc.

Execute SonarQube Scanner

JDK: JDK to be used for this SonarQube analysis

(Inherit From Job)

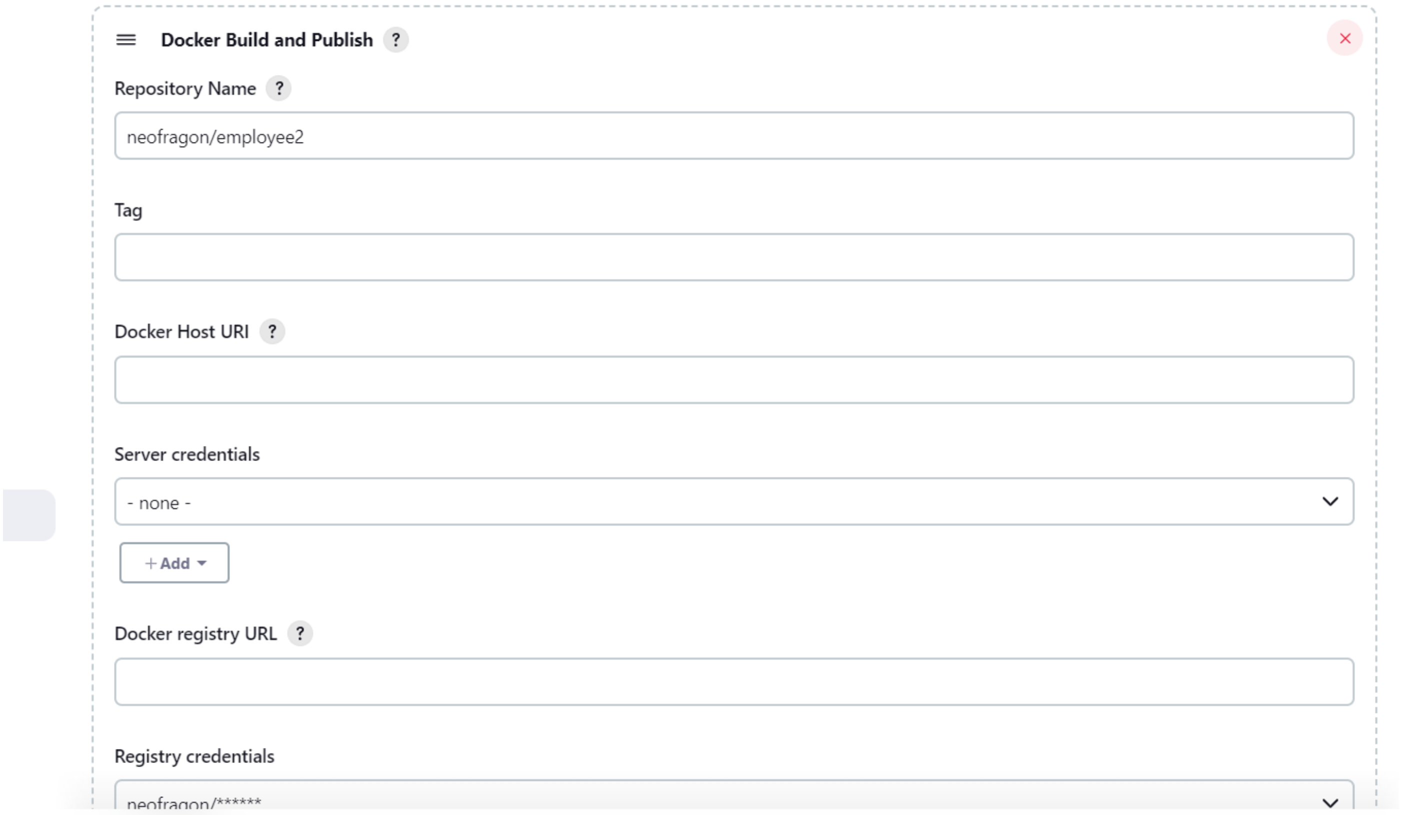
Path to project properties

Analysis properties

```
sonar.projectKey=employee
sonar.projectName=employee
sonar.projectVersion=1.0
sonar.language=java
sonat.tests=src/test/java
sonar.sources=src/main/java
sonar.java.binaries=target/classes
```

Additional arguments

5. To build and publish the docker image to Docker Hub, we should click on **Docker build and publish,** where we need to provide docker hub repository name and add our docker hub credentials.


Docker Build and Publish

Repository Name: neofragon/employee2

Tag:

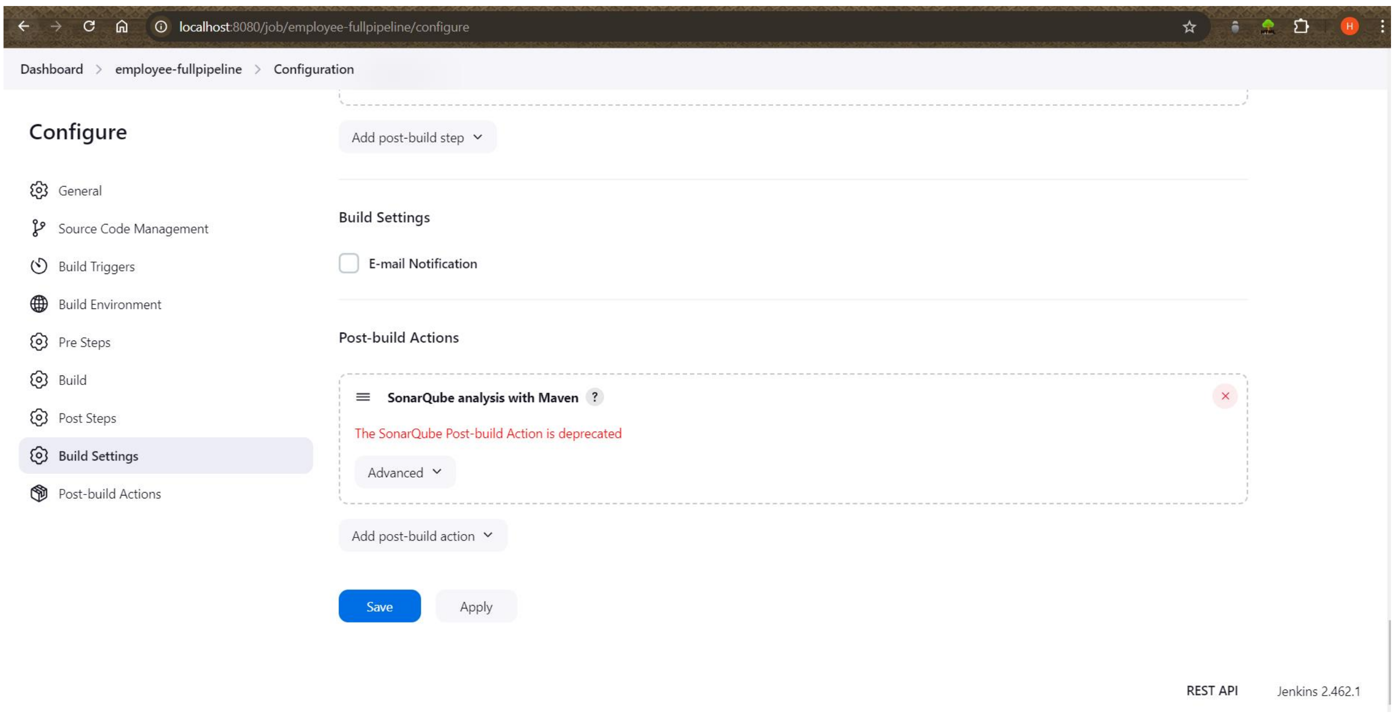
Docker Host URI:

Server credentials: - none -

Docker registry URL:

Registry credentials: neofragon/******

6. In the post build, we have to select SonarQube Analysis with Maven.


localhost:8080/job/employee-fullpipeline/configure

Dashboard > employee-fullpipeline > Configuration

Configure

Build Settings

E-mail Notification

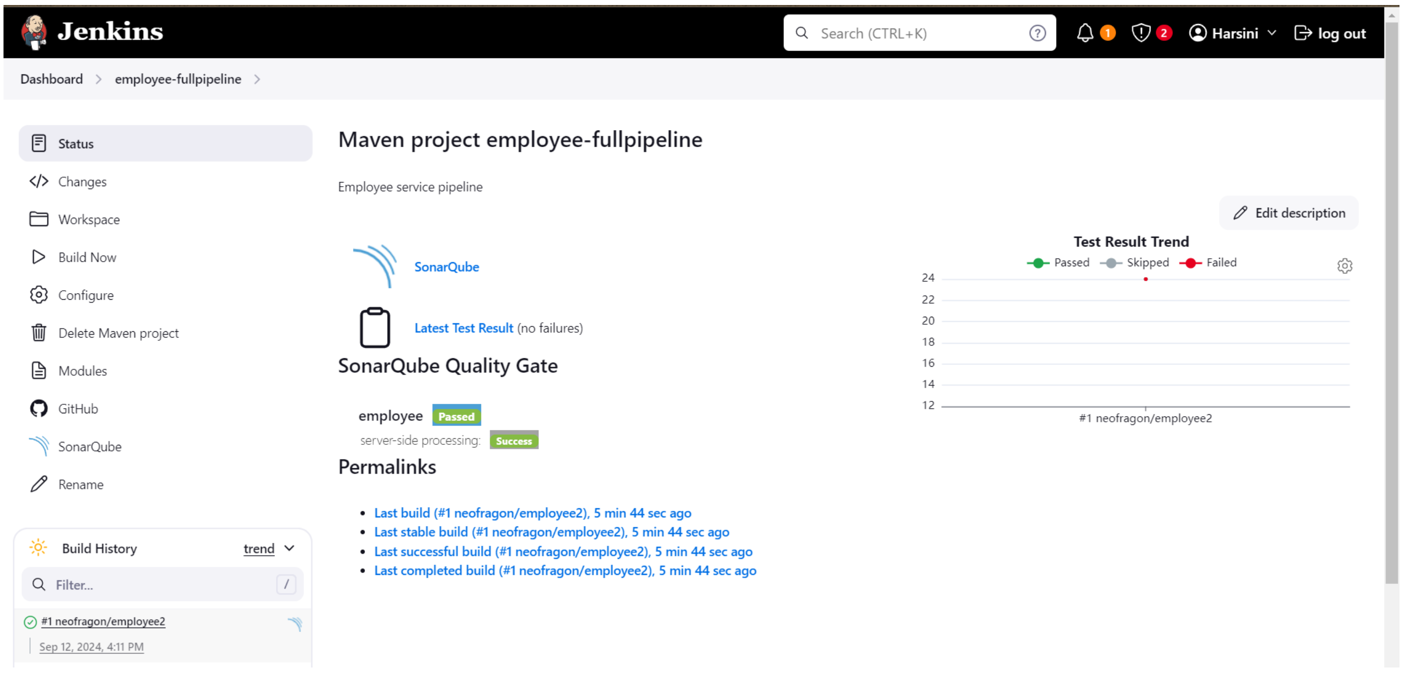
Post-build Actions

SonarQube analysis with Maven

The SonarQube Post-build Action is deprecated

REST API   Jenkins 2.462.1

BUILD OUTPUT


Dashboard > employee-fullpipeline >

Status
Changes
Workspace
Build Now
Configure
Delete Maven project
Modules
GitHub
SonarQube
Rename

Build History trend

#1 neofragon/employee2
Sep 12, 2024, 4:11 PM

Maven project employee-fullpipeline

Employee service pipeline

SonarQube

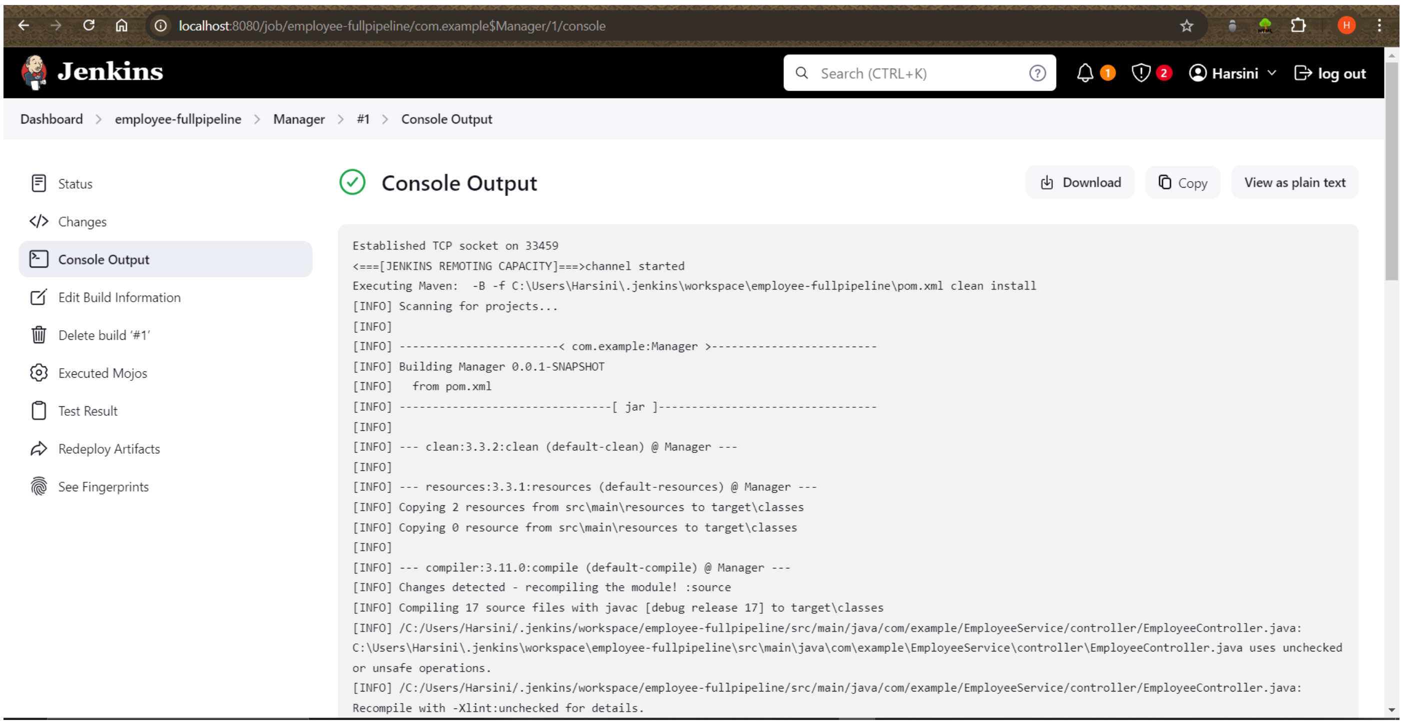
Latest Test Result (no failures)

SonarQube Quality Gate

employee Passed
server-side processing: Success

Permalinks

- Last build (#1 neofragon/employee2), 5 min 44 sec ago
- Last stable build (#1 neofragon/employee2), 5 min 44 sec ago
- Last successful build (#1 neofragon/employee2), 5 min 44 sec ago
- Last completed build (#1 neofragon/employee2), 5 min 44 sec ago


localhost:8080/job/employee-fullpipeline/com.example$Manager/1/console

Dashboard > employee-fullpipeline > Manager > #1 > Console Output

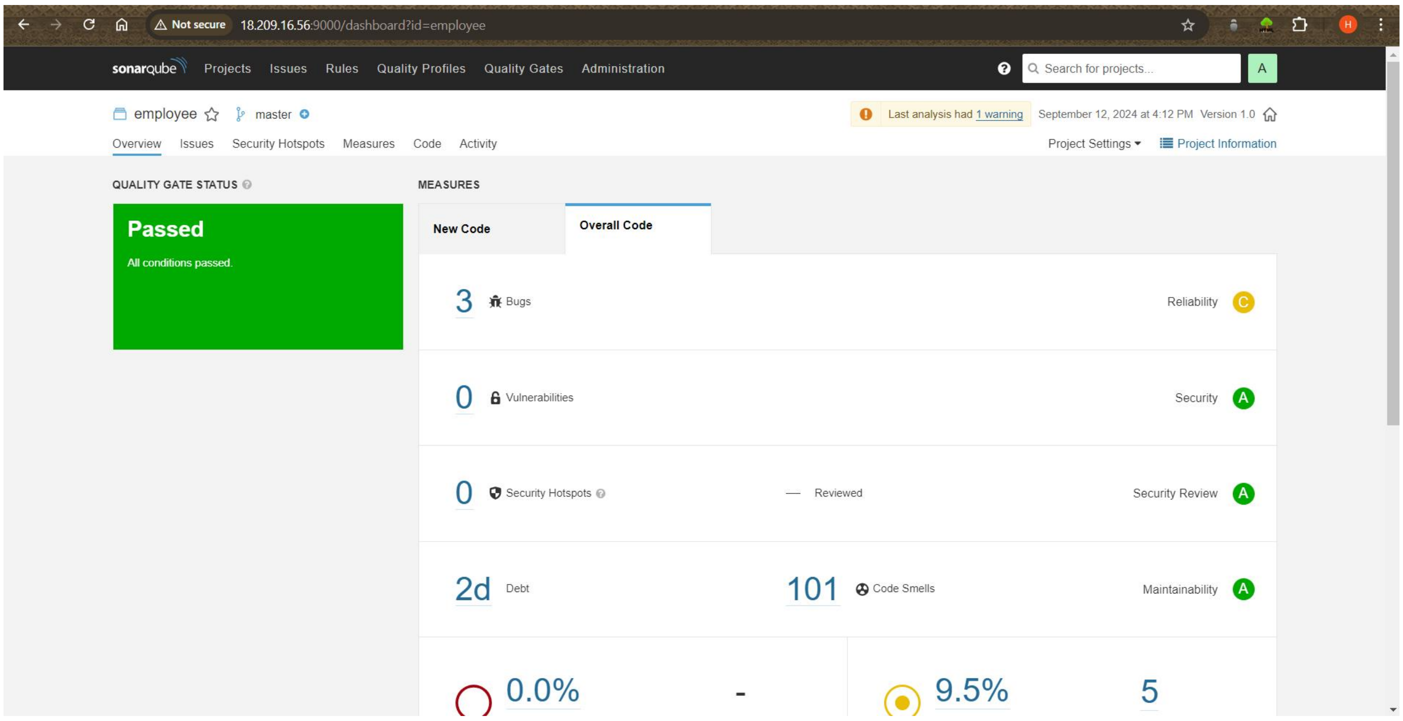
Status
Changes
Console Output
Edit Build Information
Delete build '#1'
Executed Mojos
Test Result
Redeploy Artifacts
See Fingerprints

Console Output

```
Established TCP socket on 33459
<===[JENKINS REMOTING CAPACITY]===>channel started
Executing Maven:  -B -f C:\Users\Harsini\.jenkins\workspace\employee-fullpipeline\pom.xml clean install
[INFO] Scanning for projects...
[INFO]
[INFO] -----------------------< com.example:Manager >------------------------
[INFO] Building Manager 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]--------------------------------
[INFO]
[INFO] --- clean:3.3.2:clean (default-clean) @ Manager ---
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ Manager ---
[INFO] Copying 2 resources from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ Manager ---
[INFO] Changes detected - recompiling the module! :source
[INFO] Compiling 17 source files with javac [debug release 17] to target\classes
[INFO] /C:/Users/Harsini/.jenkins/workspace/employee-fullpipeline/src/main/java/com/example/EmployeeService/controller/EmployeeController.java:
C:\Users\Harsini\.jenkins\workspace\employee-fullpipeline\src\main\java\com\example\EmployeeService\controller\EmployeeController.java uses unchecked
or unsafe operations.
[INFO] /C:/Users/Harsini/.jenkins/workspace/employee-fullpipeline/src/main/java/com/example/EmployeeService/controller/EmployeeController.java:
Recompile with -Xlint:unchecked for details.
```

```
16:12:37.569 INFO  Sensor VB.NET Project Type Information [vbnet] (done) | time=2ms
16:12:37.569 INFO  Sensor VB.NET Properties [vbnet]
16:12:37.571 INFO  Sensor VB.NET Properties [vbnet] (done) | time=2ms
16:12:37.579 INFO  ------------- Run sensors on project
16:12:37.605 INFO  Sensor Zero Coverage Sensor
16:12:37.661 INFO  Sensor Zero Coverage Sensor (done) | time=56ms
16:12:37.662 INFO  Sensor Java CPD Block Indexer
16:12:37.792 INFO  Sensor Java CPD Block Indexer (done) | time=131ms
16:12:37.796 INFO  SCM Publisher SCM provider for this project is: git
16:12:37.797 INFO  SCM Publisher 17 source files to be analyzed
16:12:38.292 INFO  SCM Publisher 17/17 source files have been analyzed (done) | time=494ms
16:12:38.302 INFO  CPD Executor 6 files had no CPD blocks
16:12:38.302 INFO  CPD Executor Calculating CPD for 11 files
16:12:38.352 INFO  CPD Executor CPD calculation finished (done) | time=49ms
16:12:38.620 INFO  Analysis report generated in 223ms, dir size=194.0 kB
16:12:39.014 INFO  Analysis report compressed in 393ms, zip size=66.6 kB
16:12:39.704 INFO  Analysis report uploaded in 686ms
16:12:39.708 INFO  ANALYSIS SUCCESSFUL, you can browse http://18.209.16.56:9000/dashboard?id=employee
16:12:39.708 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
16:12:39.710 INFO  More about the report processing at http://18.209.16.56:9000/api/ce/task?id=AZHl1A_1SYAcYk_HN1tR
16:12:39.822 INFO  Analysis total time: 22.403 s
16:12:39.826 INFO  EXECUTION SUCCESS
16:12:39.827 INFO  Total time: 26.406s
[employee-fullpipeline] $ docker build -t neofragon/employee2 --pull=true C:\Users\Harsini\.jenkins\workspace\employee-fullpipeline
#0 building with "desktop-linux" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 160B 0.0s done
#1 DONE 0.1s
```

SONAR RESULTS


Not secure 18.209.16.56:9000/dashboard?id=employee

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

employee master

Last analysis had 1 warning September 12, 2024 at 4:12 PM Version 1.0

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code | Overall Code

3 Bugs Reliability C

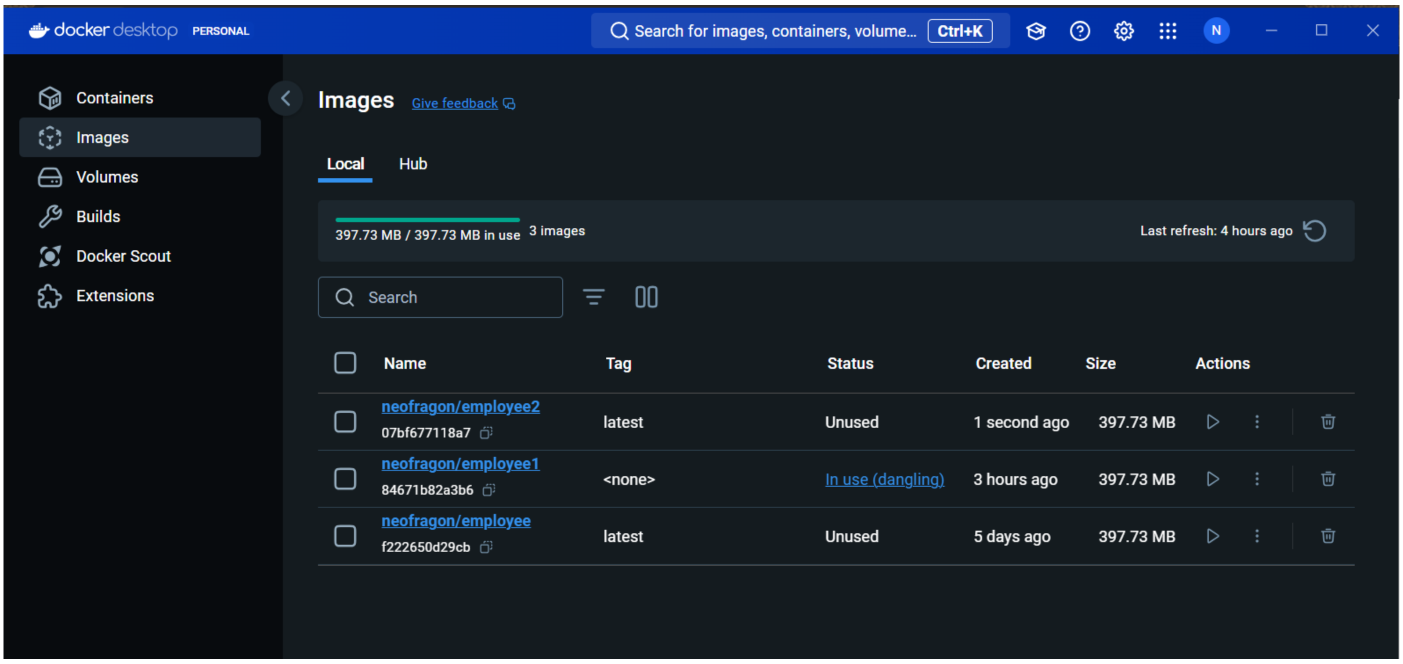
0 Vulnerabilities Security A

0 Security Hotspots — Reviewed Security Review A

2d Debt 101 Code Smells Maintainability A

0.0% - 9.5% 5

DOCKER IMAGE IN DOCKER HUB



397.73 MB / 397.73 MB in use 3 images

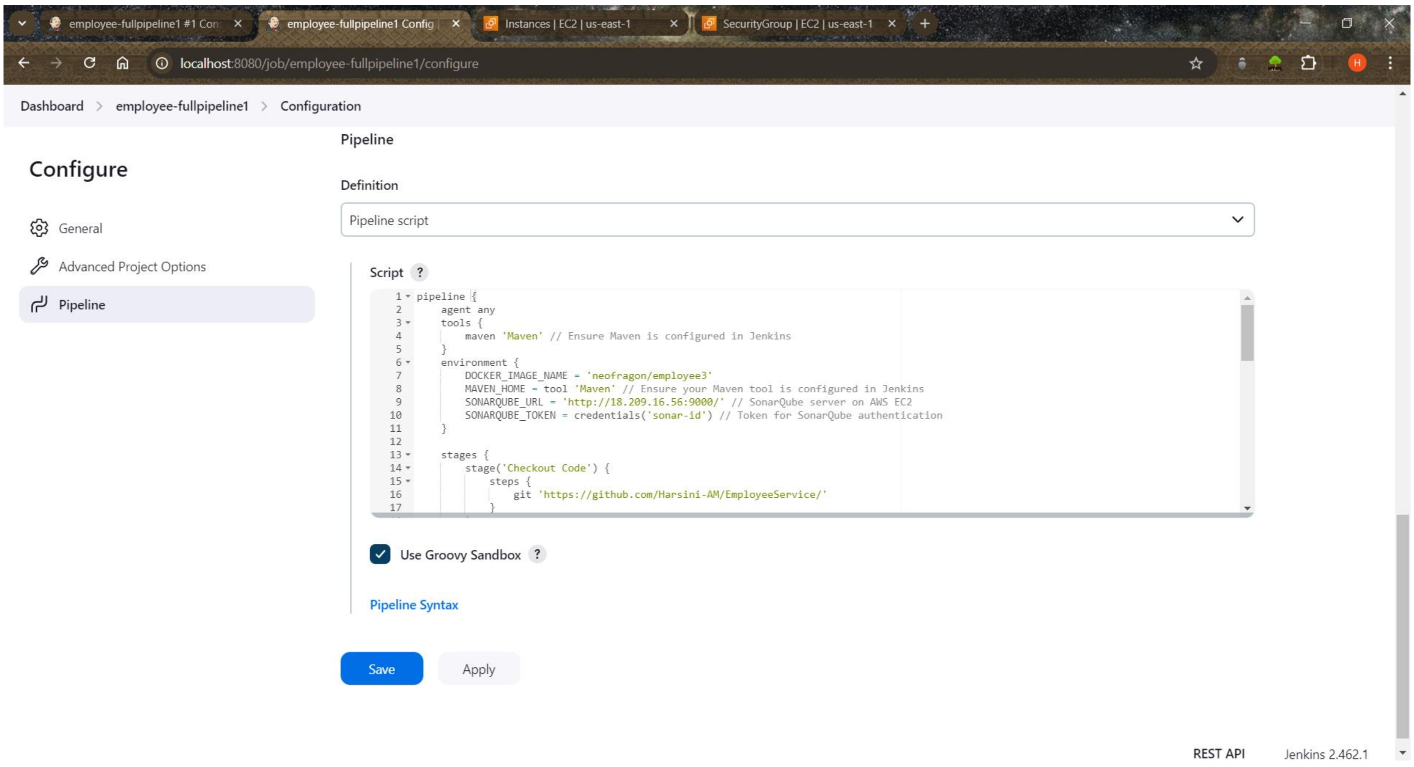
Last refresh: 4 hours ago

Name	Tag	Status	Created	Size	Actions
neofragon/employee2 07bf677118a7	latest	Unused	1 second ago	397.73 MB	
neofragon/employee1 84671b82a3b6	<none>	In use (dangling)	3 hours ago	397.73 MB	
neofragon/employee f222650d29cb	latest	Unused	5 days ago	397.73 MB	

B. PIPELINE PROJECT

- **Advanced & Flexible**: A **Pipeline Project** allows defining complex workflows as code in a Jenkinsfile.
- **Scripted/Declarative**: You write a series of steps or stages in Groovy, making it highly customizable.
- **Multi-Stage Builds**: Supports multiple stages (e.g., build, test, deploy), making it ideal for **Continuous Delivery (CD)**.
- **Code as Configuration**: The build process is defined in code, making it easy to version control and reproduce.
- **Best For**: Complex, multi-step processes where you need more control and automation.

PIPELINE THAT RUN JUNIT TESTS, GENERATE SONAR REPORTS, BUILD AND PUBLISH DOCKER IMAGE FOR EMPLOYEE MICROSERVICE TO DOCKER HUB.


Dashboard > employee-fullpipeline1 > Configuration

Configure

Pipeline

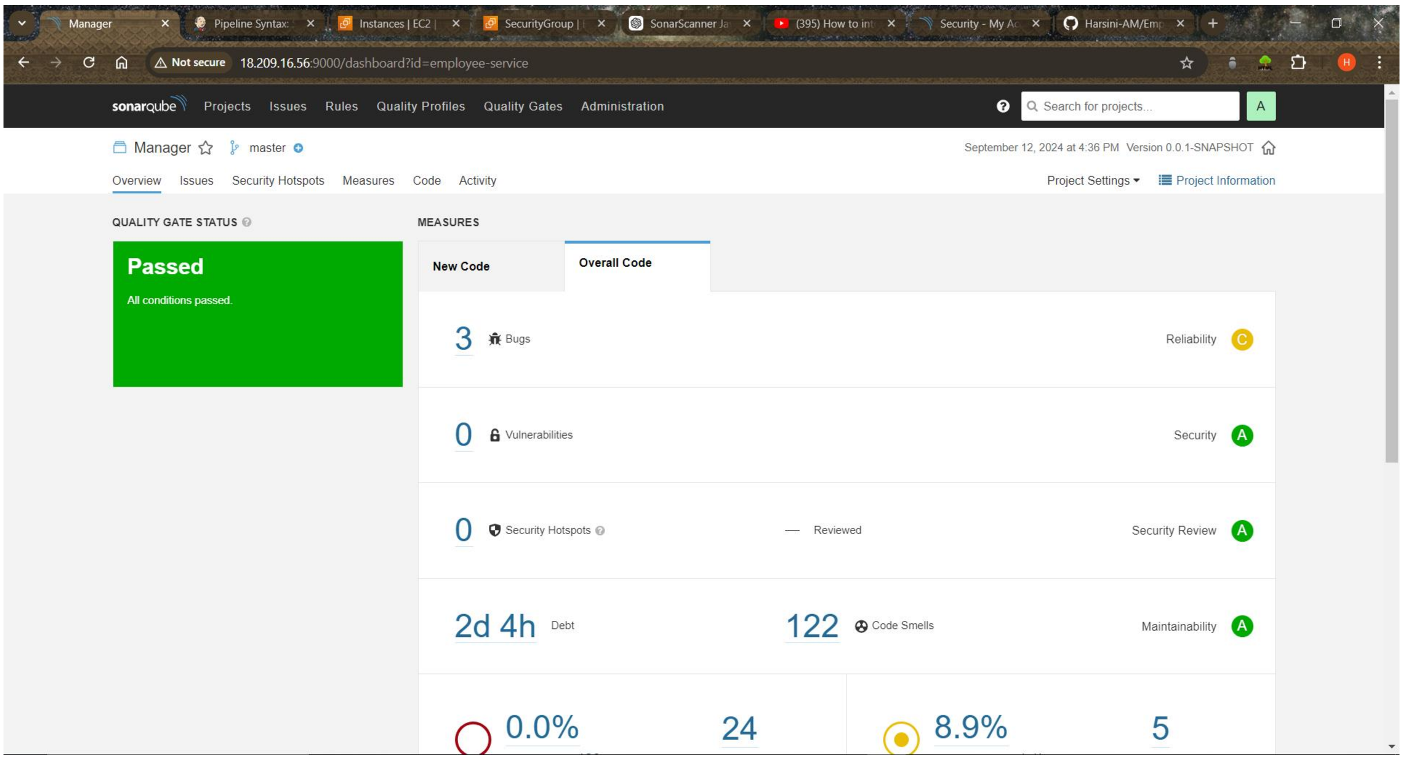
Definition

Pipeline script

Script

```
pipeline {
    agent any
    tools {
        maven 'Maven' // Ensure Maven is configured in Jenkins
    }
    environment {
        DOCKER_IMAGE_NAME = 'neofragon/employee3'
        MAVEN_HOME = tool 'Maven' // Ensure your Maven tool is configured in Jenkins
        SONARQUBE_URL = 'http://18.209.16.56:9000/' // SonarQube server on AWS EC2
        SONARQUBE_TOKEN = credentials('sonar-id') // Token for SonarQube authentication
    }

    stages {
        stage('Checkout Code') {
            steps {
                git 'https://github.com/Harsini-AM/EmployeeService/'
            }
```

SONARQUBE RESULTS


Manager · master

September 12, 2024 at 4:36 PM Version 0.0.1-SNAPSHOT

QUALITY GATE STATUS

Passed

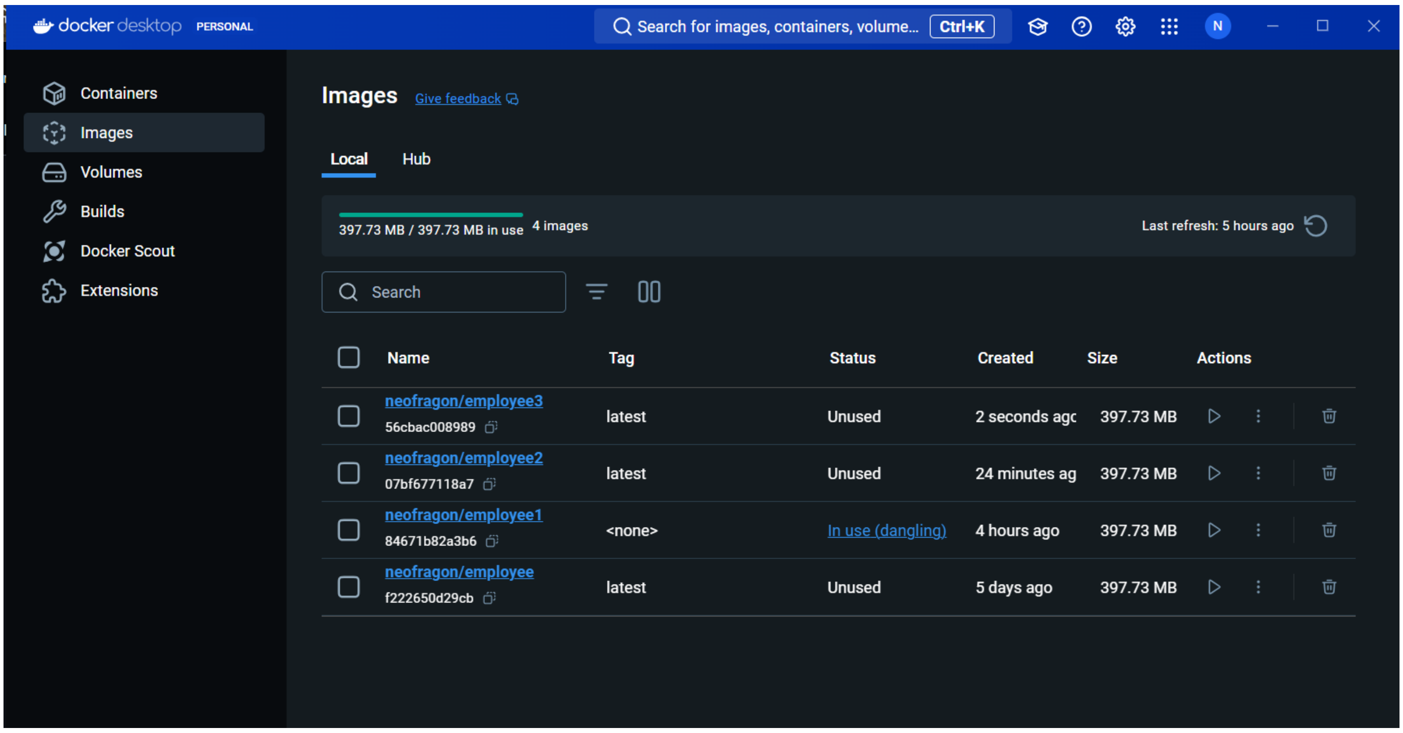
All conditions passed.

MEASURES

Overall Code

3 Bugs		Reliability C
0 Vulnerabilities		Security A
0 Security Hotspots	— Reviewed	Security Review A
2d 4h Debt	122 Code Smells	Maintainability A
0.0%	24	8.9% 5

DOCKER IMAGE OF EMPLOYEE MICROSERVICE



Images

397.73 MB / 397.73 MB in use 4 images

Last refresh: 5 hours ago

Name	Tag	Status	Created	Size	Actions
neofragon/employee3 56cbac008989	latest	Unused	2 seconds agc	397.73 MB	
neofragon/employee2 07bf677118a7	latest	Unused	24 minutes ag	397.73 MB	
neofragon/employee1 84671b82a3b6	<none>	In use (dangling)	4 hours ago	397.73 MB	
neofragon/employee f222650d29cb	latest	Unused	5 days ago	397.73 MB	