PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY
Department of Information Technology
(NBA Accredited)

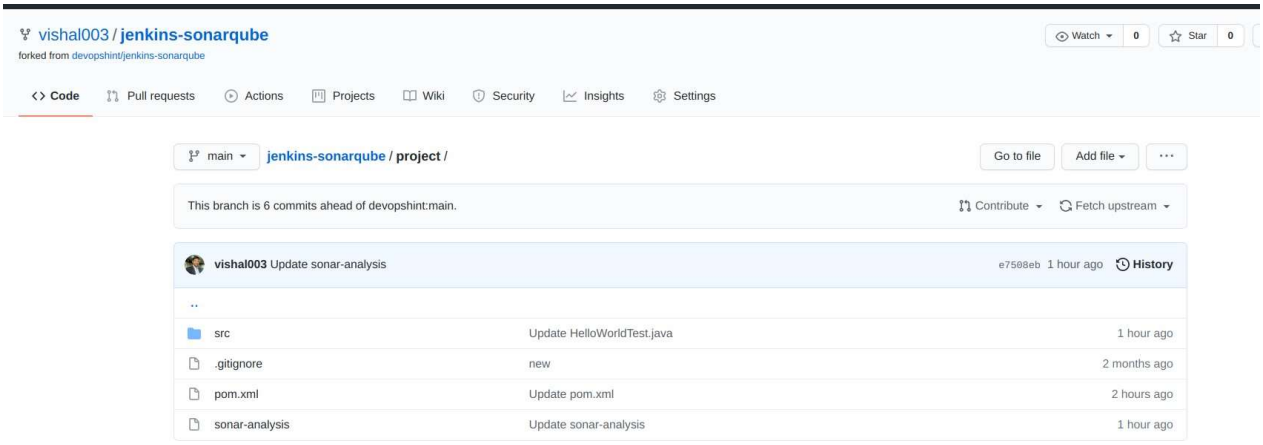
Semester: V
Academic Year: 2022-23
Class / Branch: TE IT
Subject: Advanced Devops Lab (ADL)
Name of Instructor:Prof. Manjusha K.

Name of StudentSoham Dalvi
Student ID: 21104010

EXPERIMENT NO. 08

Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Java application.

Screenshots:



main ▾ **jenkins-sonarqube** / project / **src** /

This branch is 6 commits ahead of devopshint:main.

vishal003 Update HelloWorldTest.java

..	
main/java	Update HelloWorld.java
test/java	Update HelloWorldTest.java

Enter an item name

SonarQube.

» *Required field*

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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.

Bitbucket Team/Project
Scans a Bitbucket Cloud Team (or Bitbucket Server Project) for all repositories matching some defined markers.

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 folders.

OK

General | Build Triggers | Advanced Project Options | Pipeline

Description

Hello Pipeline job

[Plain text] Preview

- [ ] Discard old builds
- [ ] Do not allow concurrent builds
- [ ] Do not allow the pipeline to resume if the controller restarts
- [x] GitHub project

Project url

https://github.com/vishal003/jenkins-sonarqube/

Pipeline

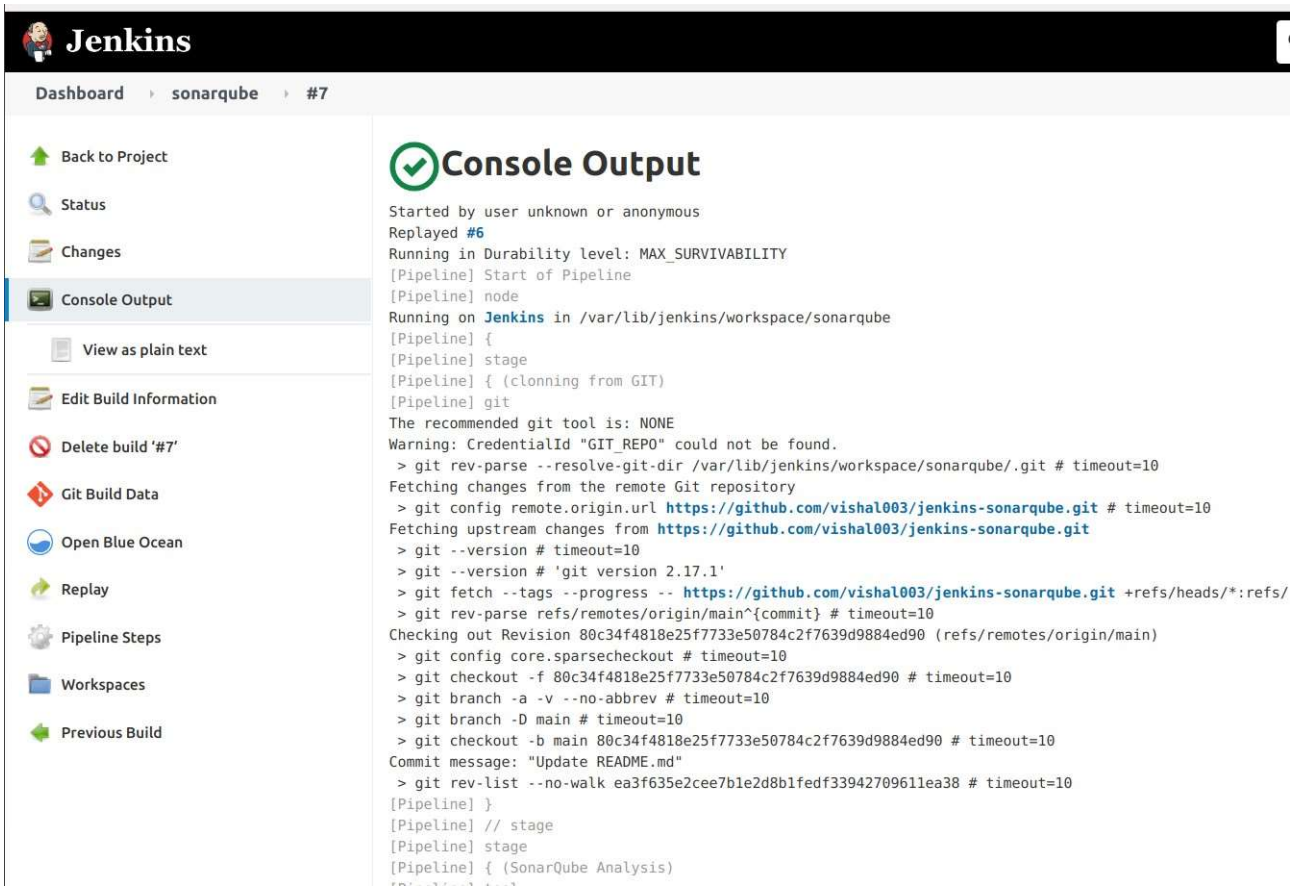
Definition

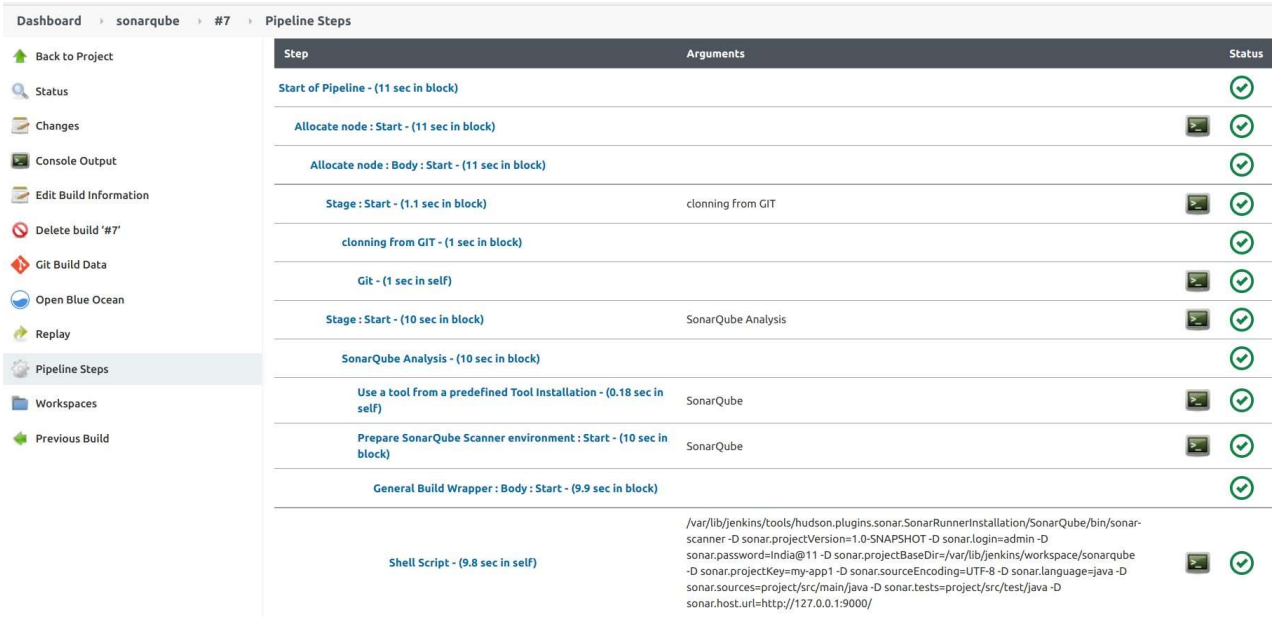
Pipeline script

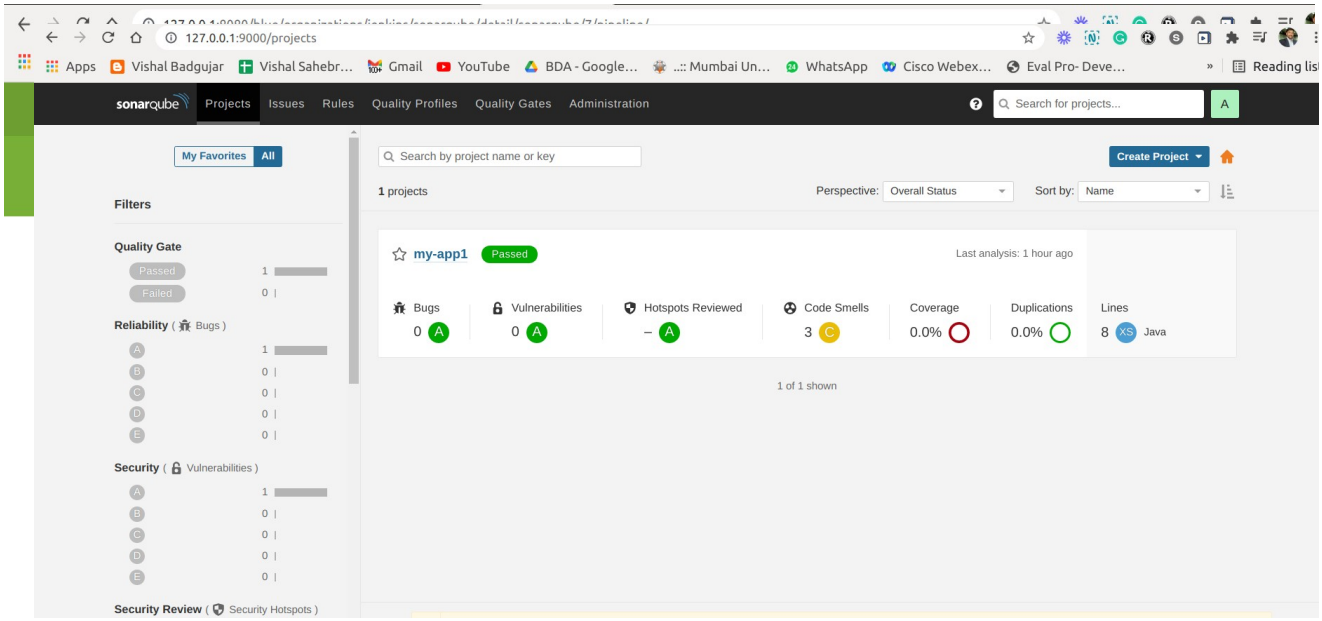
Script

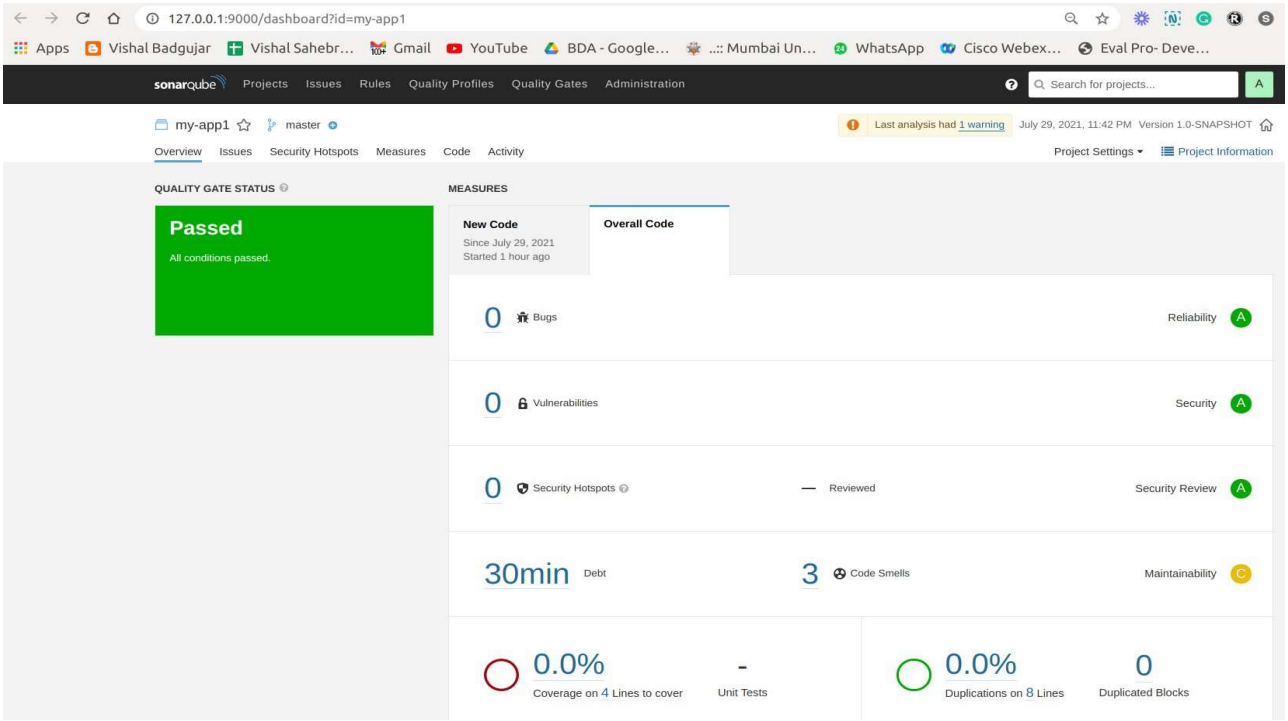
```
node
{
    stage('clonning from GIT'){
git branch: 'main', credentialsId: 'GIT_REPO', url: 'https://github.com/vishal003/jenkins-sonarqube.git'
    }

stage('SonarQube Analysis') {
    def scannerHome = tool 'SonarQube'
      withSonarQubeEnv('SonarQube') {
      sh """/var/lib/jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/SonarQube/bin/sonar-scanner \
      -D sonar.projectVersion=1.0-SNAPSHOT \
        -D sonar.login=admin \
       -D sonar.password=India@11 \
       -D sonar.projectBaseDir=/var/lib/jenkins/workspace/sonarqube \
         -D sonar.projectKey=my-app1 \
         -D sonar.sourceEncoding=UTF-8 \
         -D sonar.language=java \
         -D sonar.sources=project/src/main/java \
         -D sonar.tests=project/src/test/java \
         -D sonar.host.url=http://127.0.0.1:9000/"""
         }
```


Jenkins
Dashboard › sonarqube › #7
Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#7'
Git Build Data
Open Blue Ocean
Replay
Pipeline Steps
Workspaces
Previous Build
Console Output
Started by user unknown or anonymous
Replayed #6
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/sonarqube
[Pipeline] {
[Pipeline] stage
[Pipeline] { (clonning from GIT)
[Pipeline] git
The recommended git tool is: NONE
Warning: CredentialId "GIT_REPO" could not be found.
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/sonarqube/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/vishal003/jenkins-sonarqube.git # timeout=10
Fetching upstream changes from https://github.com/vishal003/jenkins-sonarqube.git
> git --version # timeout=10
> git --version # 'git version 2.17.1'
> git fetch --tags --progress -- https://github.com/vishal003/jenkins-sonarqube.git +refs/heads/*:refs/
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 80c34f4818e25f7733e50784c2f7639d9884ed90 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 80c34f4818e25f7733e50784c2f7639d9884ed90 # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git branch -D main # timeout=10
> git checkout -b main 80c34f4818e25f7733e50784c2f7639d9884ed90 # timeout=10
Commit message: "Update README.md"
> git rev-list --no-walk ea3f635e2cee7b1e2d8b1fedf33942709611ea38 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (SonarQube Analysis)


Dashboard › sonarqube › #7 › Pipeline Steps
Back to Project
Status
Changes
Console Output
Edit Build Information
Delete build '#7'
Git Build Data
Open Blue Ocean
Replay
Pipeline Steps
Workspaces
Previous Build
Step
Arguments
Status
Start of Pipeline - (11 sec in block)
Allocate node : Start - (11 sec in block)
Allocate node : Body : Start - (11 sec in block)
Stage : Start - (1.1 sec in block)
clonning from GIT
clonning from GIT - (1 sec in block)
Git - (1 sec in self)
Stage : Start - (10 sec in block)
SonarQube Analysis
SonarQube Analysis - (10 sec in block)
Use a tool from a predefined Tool Installation - (0.18 sec in self)
SonarQube
Prepare SonarQube Scanner environment : Start - (10 sec in block)
SonarQube
General Build Wrapper : Body : Start - (9.9 sec in block)
Shell Script - (9.8 sec in self)
/var/lib/jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/SonarQube/bin/sonar-scanner -D sonar.projectVersion=1.0-SNAPSHOT -D sonar.login=admin -D sonar.password=India@11 -D sonar.projectBaseDir=/var/lib/jenkins/workspace/sonarqube -D sonar.projectKey=my-app1 -D sonar.sourceEncoding=UTF-8 -D sonar.language=java -D sonar.sources=project/src/main/java -D sonar.tests=project/src/test/java -D sonar.host.url=http://127.0.0.1:9000/





Conclusion: Hence we have studied about who create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Java application.