EXPERIMENT 7

Name: Anmol Agnihotri

Class: TE6

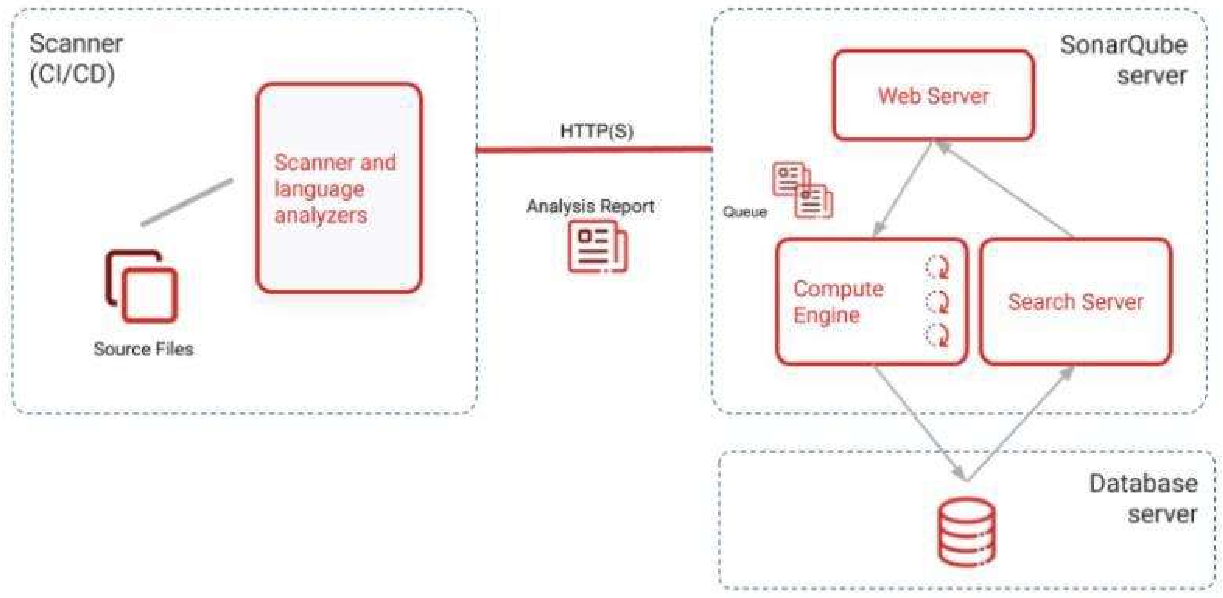
Roll: 02

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Lab Outcome: To use Continuous Monitoring Tools to resolve any system errors (low memory, unreachable server etc.) before they have any negative impact on the business productivity

Theory:

SonarQube is a Code Quality Assurance tool that collects and analyzes source code, and provides reports for the code quality of your project. It combines static and dynamic analysis tools and enables quality to be measured continually over time. Everything from minor styling choices, to design errors are inspected and evaluated by SonarQube. This provides users with a rich searchable history of the code to analyze where the code is messing up and determine whether or not it is styling issues, code defeats, code duplication, lack of test coverage, or excessively complex code. The software will analyze source code from different aspects and drills down the code layer by layer, moving module level down to the class level, with each level producing metric values and statistics that should reveal problematic areas in the source code that needs improvement. SonarQube also ensures code reliability, Application security, and reduces technical debt by making your code base clean and maintainable. SonarQube also provides support for 27 different languages, including C, C++, Java, JavaScript, PHP, GO, Python, and much more. SonarQube also provides Ci/CD integration, and gives feedback during code review with branch analysis and pull request decoration.



Installation:

Pre-requisites

The only prerequisite for running SonarQube is to have Java (Oracle JRE 11 or OpenJDK 11) installed on your machine.

Hardware Requirements

1. A small-scale (individual or small team) instance of the SonarQube server requires at least 2GB of RAM to run efficiently and 1GB of free RAM for the OS. If you are installing an instance for a large teams or Enterprise, please consider the additional recommendations below.
2. The amount of disk space you need will depend on how much code you analyze with SonarQube.
3. SonarQube must be installed on hard drives that have excellent read & write performance. Most importantly, the "data" folder houses the Elasticsearch indices on which a huge amount of I/O will be done when the server is up and running. Great read & write hard drive performance will therefore have a great impact on the overall SonarQube server performance.
4. SonarQube does not support 32-bit systems on the server side. SonarQube does, however, support 32-bit systems on the scanner side.

Download "SonarQube" (download community edition)
https://www.sonarqube.org/downloads/
Download "SonarQube-Scanner" (download as per your machine OS)
https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/

Setup for SonarQube Server:

1. Unzip both the downloaded files and keep it at a common place.

2. Add following to path in "System Variable"

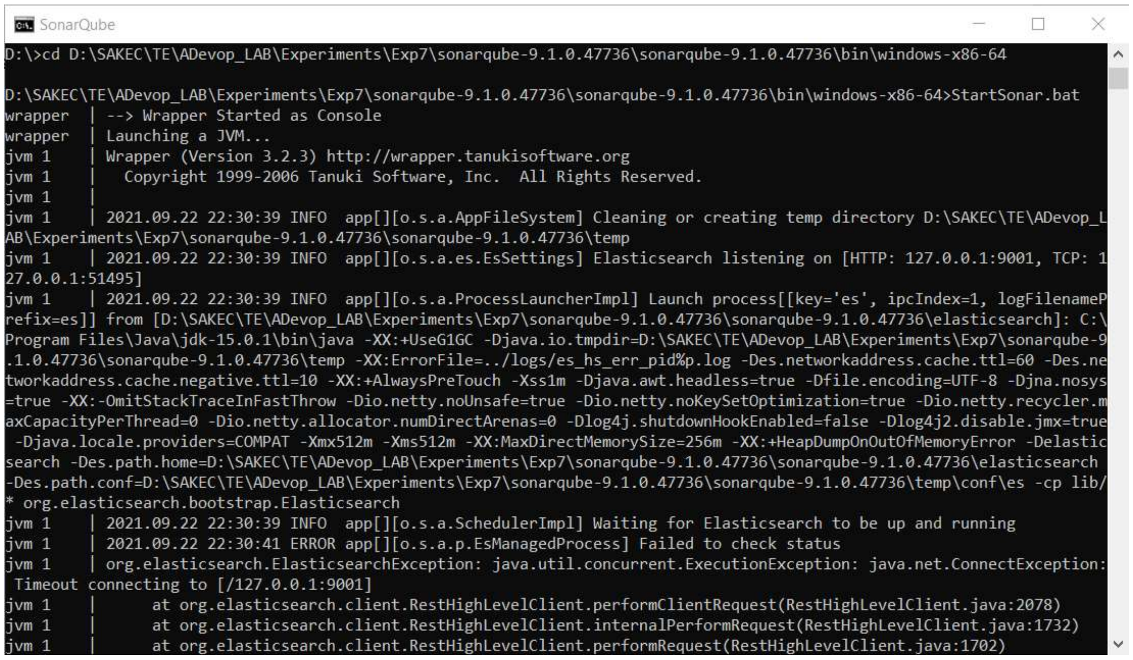
a) <location>\sonar-scanner-cli-4.6.2.2472-windows\sonar-scanner-4.6.2.2472-windows\bin

3. Set some configuration inside “sonarqube-scanner” config file

Inside your “sonarqube-scanner” folder, go to “conf” folder and find “sonar-scanner.properties” file. Open it in edit mode.
Add these two basic properties in “sonar-scanner.properties” file, or if it’s already there but commented, then uncomment it.
sonar.host.url=http://localhost:9000 sonar.sourceEncoding=UTF-8

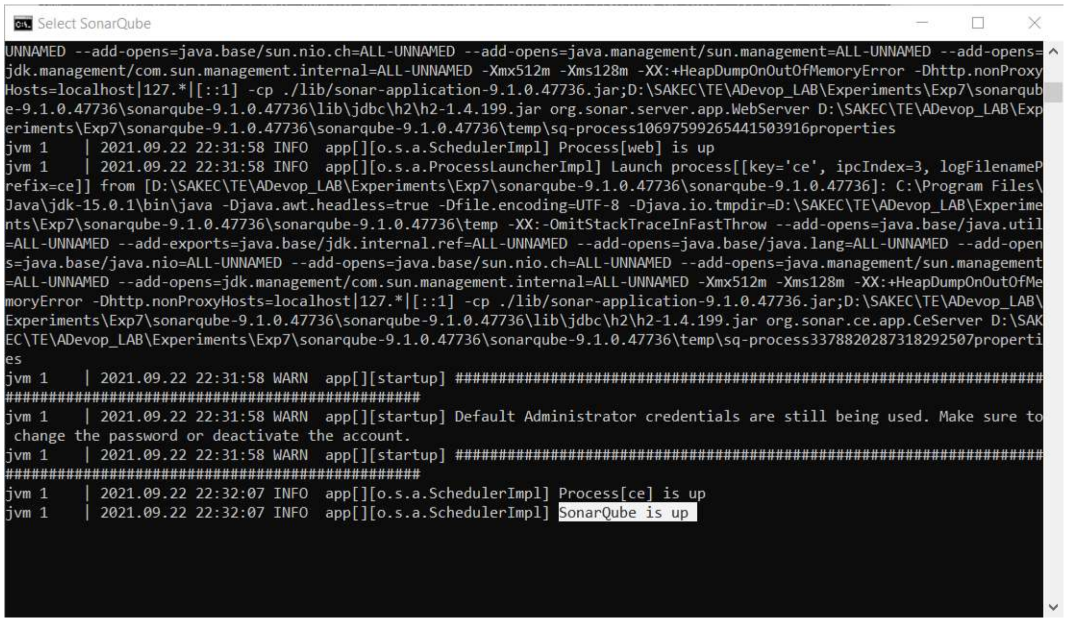
4. Start the sonarqube server Open “Command prompt”, and from terminal itself, go to same folder path where we kept the 1st unzipped folder, i.e., sonarqube folder > bin > respective OS folder.

//for example, this is my path D: Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\bin\windows-x86-64
Here, you will find “sonar.sh” Bash file.
5. “SartSonar.bat” run this command to start SonarQube server



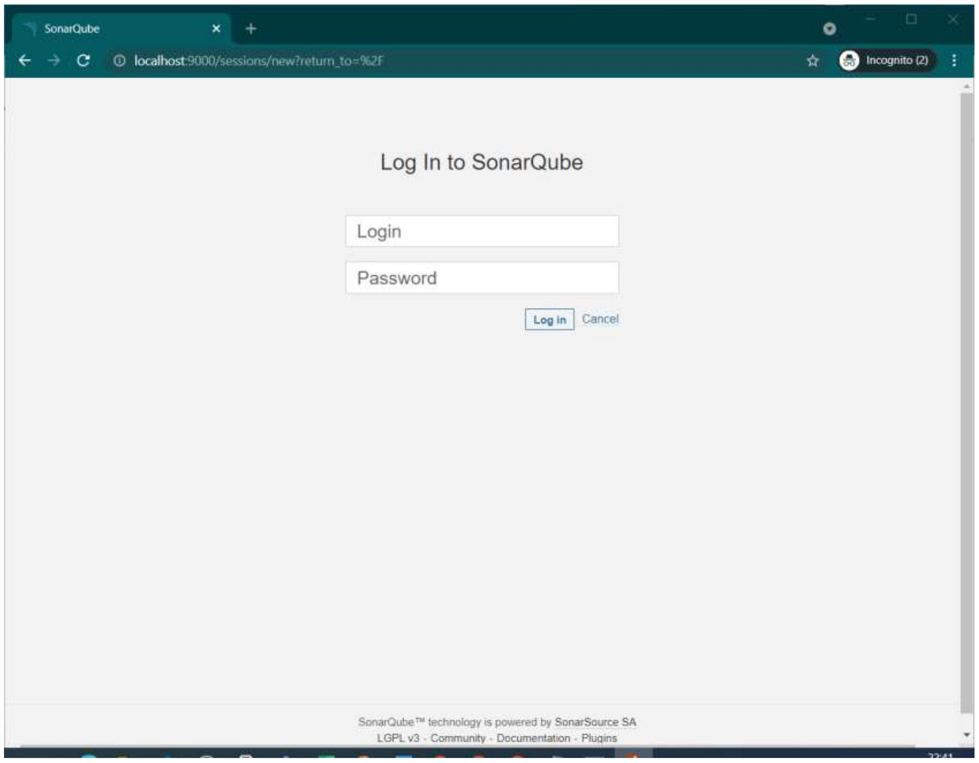
```
D:\>cd D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\bin\windows-x86-64

D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\bin\windows-x86-64>StartSonar.bat
wrapper  | --> Wrapper Started as Console
wrapper  | Launching a JVM...
jvm 1    | Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
jvm 1    |   Copyright 1999-2006 Tanuki Software, Inc.  All Rights Reserved.
jvm 1    |
jvm 1    | 2021.09.22 22:30:39 INFO  app[][o.s.a.AppFileSystem] Cleaning or creating temp directory D:\SAKEC\TE\ADevop_L
AB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\temp
jvm 1    | 2021.09.22 22:30:39 INFO  app[][o.s.a.es.EsSettings] Elasticsearch listening on [HTTP: 127.0.0.1:9001, TCP: 1
27.0.0.1:51495]
jvm 1    | 2021.09.22 22:30:39 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='es', ipcIndex=1, logFilenameP
refix=es]] from [D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\elasticsearch]: C:\
Program Files\Java\jdk-15.0.1\bin\java -XX:+UseG1GC -Djava.io.tmpdir=D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9
.1.0.47736\sonarqube-9.1.0.47736\temp -XX:ErrorFile=../logs/es_hs_err_pid%p.log -Des.networkaddress.cache.ttl=60 -Des.ne
tworkaddress.cache.negative.ttl=10 -XX:+AlwaysPreTouch -Xss1m -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djna.nosys
=true -XX:-OmitStackTraceInFastThrow -Dio.netty.noUnsafe=true -Dio.netty.noKeySetOptimization=true -Dio.netty.recycler.m
axCapacityPerThread=0 -Dio.netty.allocator.numDirectArenas=0 -Dlog4j.shutdownHookEnabled=false -Dlog4j2.disable.jmx=true
 -Djava.locale.providers=COMPAT -Xmx512m -Xms512m -XX:MaxDirectMemorySize=256m -XX:+HeapDumpOnOutOfMemoryError -Delastic
search -Des.path.home=D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\elasticsearch
-Des.path.conf=D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\temp\conf\es -cp lib/
* org.elasticsearch.bootstrap.Elasticsearch
jvm 1    | 2021.09.22 22:30:39 INFO  app[][o.s.a.SchedulerImpl] Waiting for Elasticsearch to be up and running
jvm 1    | 2021.09.22 22:30:41 ERROR app[][o.s.a.p.EsManagedProcess] Failed to check status
jvm 1    | org.elasticsearch.ElasticsearchException: java.util.concurrent.ExecutionException: java.net.ConnectException:
 Timeout connecting to [/127.0.0.1:9001]
jvm 1    |      at org.elasticsearch.client.RestHighLevelClient.performClientRequest(RestHighLevelClient.java:2078)
jvm 1    |      at org.elasticsearch.client.RestHighLevelClient.internalPerformRequest(RestHighLevelClient.java:1732)
jvm 1    |      at org.elasticsearch.client.RestHighLevelClient.performRequest(RestHighLevelClient.java:1702)
```

6. After step 5, if your terminal shows this output, that means your SonarQube Server is up and running.

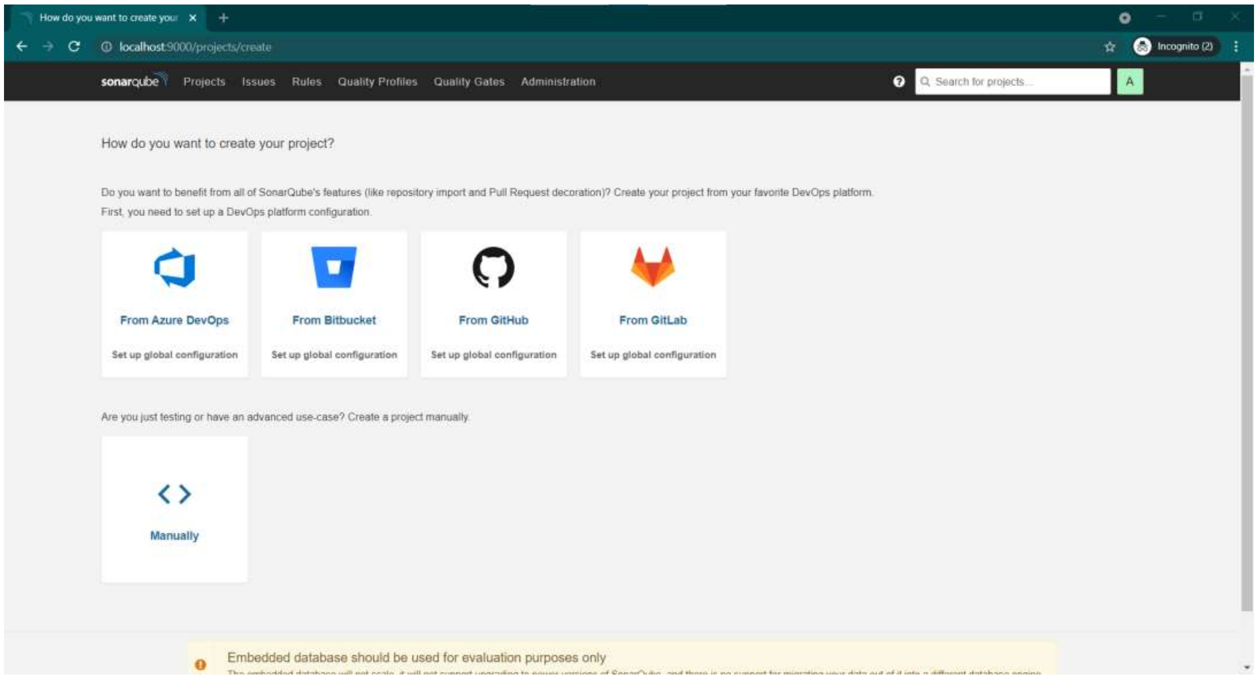


```
Select SonarQube
UNNAMED --add-opens=java.base/sun.nio.ch=ALL-UNNAMED --add-opens=java.management/sun.management=ALL-UNNAMED --add-opens=
jdk.management/com.sun.management.internal=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryError -Dhttp.nonProxy
Hosts=localhost|127.*|[::1] -cp ./lib/sonar-application-9.1.0.47736.jar;D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqub
e-9.1.0.47736\sonarqube-9.1.0.47736\lib\jdbc\h2\h2-1.4.199.jar org.sonar.server.app.WebServer D:\SAKEC\TE\ADevop_LAB\Exp
eriments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\temp\sq-process10697599265441503916properties
jvm 1    | 2021.09.22 22:31:58 INFO  app[][o.s.a.SchedulerImpl] Process[web] is up
jvm 1    | 2021.09.22 22:31:58 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='ce', ipcIndex=3, logFilenameP
refix=ce]] from [D:\SAKEC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736]: C:\Program Files\
Java\jdk-15.0.1\bin\java -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=D:\SAKEC\TE\ADevop_LAB\Experime
nts\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\temp -XX:-OmitStackTraceInFastThrow --add-opens=java.base/java.util
=ALL-UNNAMED --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED --add-open
s=java.base/java.nio=ALL-UNNAMED --add-opens=java.base/sun.nio.ch=ALL-UNNAMED --add-opens=java.management/sun.management
=ALL-UNNAMED --add-opens=jdk.management/com.sun.management.internal=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMe
moryError -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/sonar-application-9.1.0.47736.jar;D:\SAKEC\TE\ADevop_LAB\
Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\lib\jdbc\h2\h2-1.4.199.jar org.sonar.ce.app.CeServer D:\SAK
EC\TE\ADevop_LAB\Experiments\Exp7\sonarqube-9.1.0.47736\sonarqube-9.1.0.47736\temp\sq-process3378820287318292507properti
es
jvm 1    | 2021.09.22 22:31:58 WARN  app[][startup] ####################################################################
################################################
jvm 1    | 2021.09.22 22:31:58 WARN  app[][startup] Default Administrator credentials are still being used. Make sure to
 change the password or deactivate the account.
jvm 1    | 2021.09.22 22:31:58 WARN  app[][startup] ####################################################################
################################################
jvm 1    | 2021.09.22 22:32:07 INFO  app[][o.s.a.SchedulerImpl] Process[ce] is up
jvm 1    | 2021.09.22 22:32:07 INFO  app[][o.s.a.SchedulerImpl] SonarQube is up
```

7. Open any browser, add the following address into address bar, and hit Enter.
http://localhost:9000


SonarQube
localhost:9000/sessions/new?return_to=%2F
Incognito (2)
Log In to SonarQube
Login
Password
Log in
Cancel
SonarQube™ technology is powered by SonarSource SA
LGPL v3 - Community - Documentation - Plugins

8. Default login and Password is admin. (you can change the password later)


How do you want to create your
localhost:9000/projects/create
Incognito (2)
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
A
From Azure DevOps
Set up global configuration
From Bitbucket
Set up global configuration
From GitHub
Set up global configuration
From GitLab
Set up global configuration
Manually

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform.
First, you need to set up a DevOps platform configuration.

Are you just testing or have an advanced use-case? Create a project manually.

Embedded database should be used for evaluation purposes only

Setup for SonarQube-Scanner

1. Go to your project folder which you want to scan.

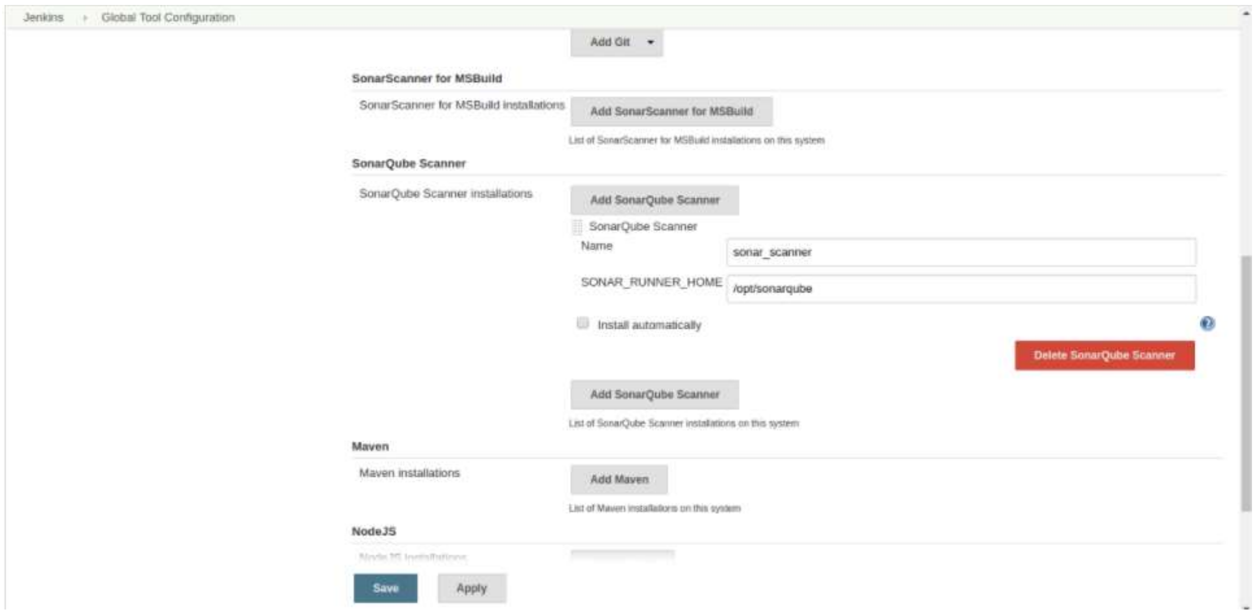
Create one new file inside your project's root folder path with name "sonar-project". The extension of the file will be ".properties".
sonar-project.properties
Add the following basic configurations inside "sonar-project.properties" file.
sonar.projectKey="any unique name" sonar.projectName="any unique name" sonar.sourceEncoding=UTF-8 sonar.sources="list of folders which will scan" sonar.exclusion="list of folders which will exclude from scan"
"sonar.sources" & "sonar.exclusion" property values will be the list of folders or files which you wants to scan or exclude from scan. The list must be separated by comma(,). If you want to include all files or folders, then just mention Dot(.)
2. Run SonarQube Scanner on your project.

Now, you are all set for your scanning your code. "Command prompt", and from Command prompt, go to the folder path where your project code resides.
// for example, I kept my test project on this path D:\JavaTest\src\main
Run this command to scan your code.
sonar-scanner // start scann *sonar-scanner -h* // to see other commands
Once the scanning ends, it will show you the output of scanning with the path where you can see the scanning details with dashboard data.

Integrate SonarQube with Jenkins

1. Login into Jenkins and install the **SonarQube scanner** plugin Go to **Manage Jenkins** –> **Manage Plugins** > **Available** –> **SonarQube scanner** And also add credentials plugins to store your credentials in Jenkins

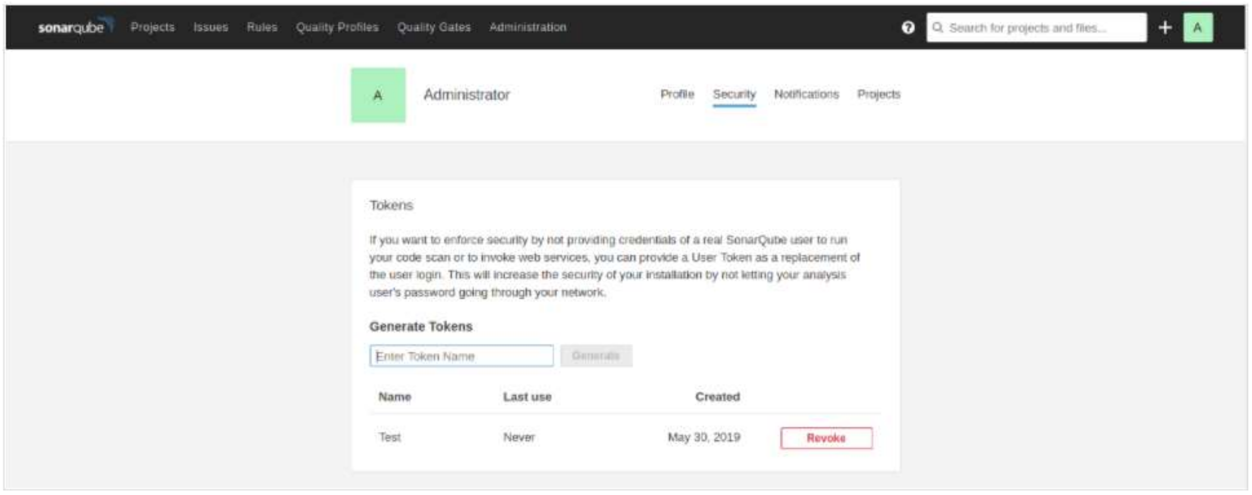
2. Configure SonarQube home path Go to **Manage Jenkins –> Global Tool Configuration –> SonarQube Scanner** 1. Name : **sonar_scanner**
2. SONAR_RUNNER_HOME: **/opt/sonarqube** (Your directory path of SonarQube)



Jenkins › Global Tool Configuration
Add Git
SonarScanner for MSBuild
SonarScanner for MSBuild installations
Add SonarScanner for MSBuild
List of SonarScanner for MSBuild installations on this system
SonarQube Scanner
SonarQube Scanner installations
Add SonarQube Scanner
SonarQube Scanner
Install automatically
Delete SonarQube Scanner
Add SonarQube Scanner
List of SonarQube Scanner installations on this system
Maven
Maven installations
Add Maven
List of Maven installations on this system
NodeJS
Save
Apply

Name: sonar_scanner
SONAR_RUNNER_HOME: /opt/sonarqube

3. Now, Configure SonarQube server in Jenkins 1. For integration, you need a SonarQube Server authentication token in Jenkins Log in into your SonarQube Server and find the following under the user bar Go to **My Account –> Security –> Generate Token**

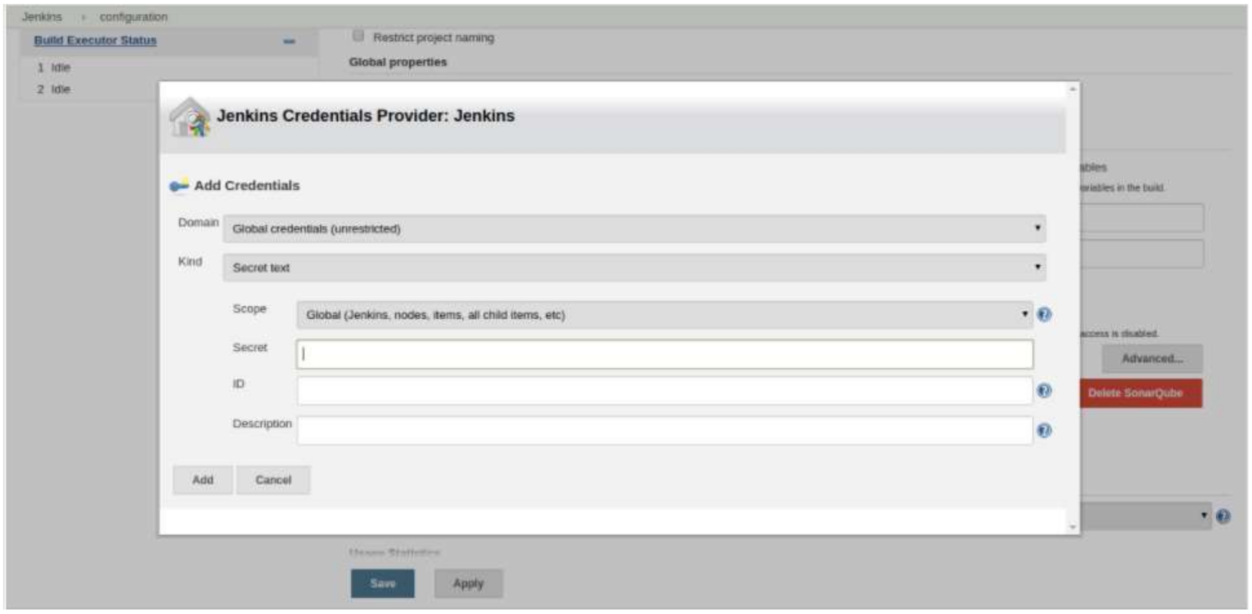


Tokens

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

Name	Last use	Created	
Test	Never	May 30, 2019	Revoke

2. Go to **Manage Jenkins** –> **Configure Systems** –> **SonarQube Servers** Name: SonarQube Server URL: Not Required is the same as the default Server authentication token : Add server authentication token as following



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Secret text

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

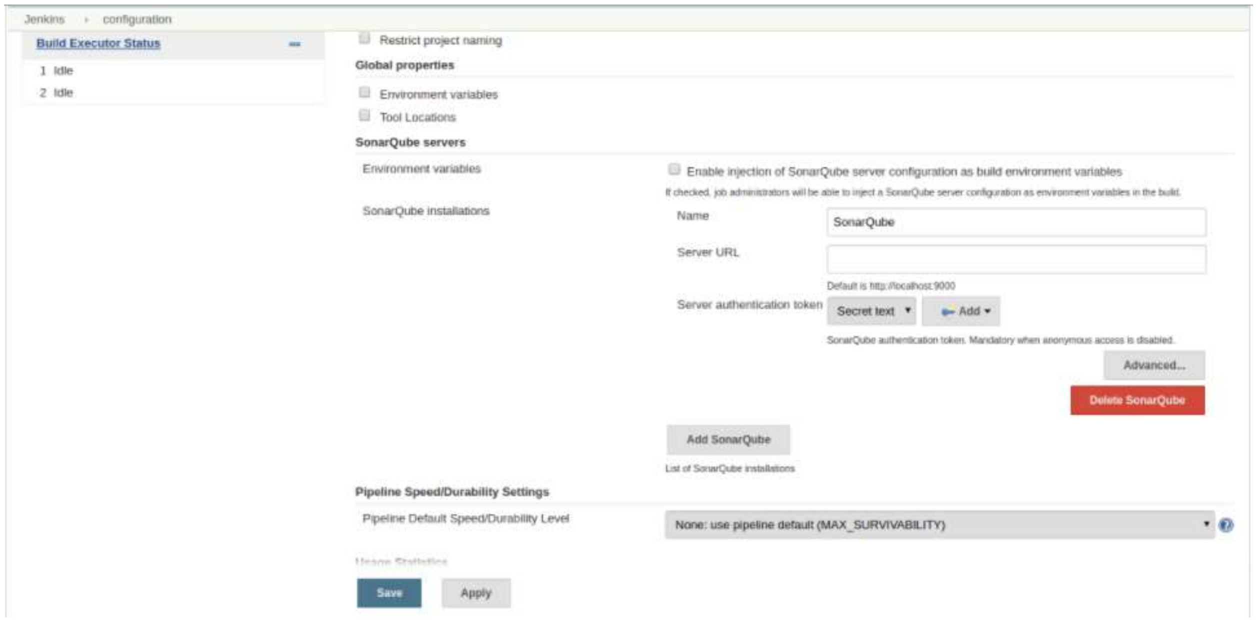
Secret:

ID:

Description:

Add Cancel

Select it as a server authentication token. Save



Restrict project naming

Global properties

Environment variables

Tool Locations

SonarQube servers

Environment variables: Enable injection of SonarQube server configuration as build environment variables

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

SonarQube installations

Name: SonarQube

Server URL:

Default is http://localhost:9000

Server authentication token: Secret text

SonarQube authentication token. Mandatory when anonymous access is disabled.

List of SonarQube installations

Pipeline Speed/Durability Settings

Pipeline Default Speed/Durability Level: None: use pipeline default (MAX_SURVIVABILITY)

Save it. Now, your SonarQube integration is completed with Jenkins. Create a job (Follow Jenkins – Continuous Integration System) to test SonarQube and generate a report of your project.

Conclusion:

SonarQube and SonarScanner is successfully installed and integrated with Jenkins.

Mahavir Education Trust's

Shah & Anchor Kutchhi Engineering College,

Chembur, Mumbai 400 088

UG Program in Information Technology

Experiment No.: 8					
Date of Performance:	24/09/2021				
Date of Submission:	01/10/2021				
Program formation/ Execution/ ethical practices (07)	Documentation (02)	Timely Submission (03)	Viva Answer (03)	Experiment Marks (15)	Teacher Signature with date
7	2	3	2	14	[illegible]

EXPERIMENT 8

Name: Anmol Agnihotri

Class: TE6

Roll: 02

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 Web / Java / Python application

Lab Outcome: To use Continuous Monitoring Tools to resolve any system errors (low memory, unreachable server etc.) before they have any negative impact on the business productivity

Theory:

Jenkins is a free and open-source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat. It supports version control tools, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, ClearCase and RTC, and can execute Apache Ant, Apache Maven and sbt based projects as well as arbitrary shell scripts and Windows batch commands.

SonarQube is an automatic code review tool to detect bugs, vulnerabilities, and code smells in your code. It can integrate with your existing workflow to enable continuous code inspection across your project branches and pull requests.

What is Jenkins Pipeline?

Jenkins Pipeline (or simply "Pipeline") is a suite of plugins which supports implementing and integrating *continuous delivery pipelines* into Jenkins.

A *continuous delivery (CD) pipeline* is an automated expression of your process for getting software from version control right through to your users and customers. Every change to your software (committed in source control) goes through a complex process on its way to being released. This process involves building the software in a reliable and repeatable manner, as well as progressing the built software (called a "build") through multiple stages of testing and deployment.

Pipeline provides an extensible set of tools for modeling simple-to-complex delivery pipelines "as code" via the Pipeline domain-specific language (DSL) syntax.

The definition of a Jenkins Pipeline is written into a text file (called a Jenkinsfile) which in turn can be committed to a project's source control repository. This is the foundation of "Pipelineas-code"; treating the CD pipeline a part of the application to be versioned and reviewed like any other code.

Creating a Jenkins file and committing it to source control provides a number of immediate benefits:

1. Automatically creates a Pipeline build process for all branches and pull requests.
2. Code review/iteration on the Pipeline (along with the remaining source code).
3. Audit trail for the Pipeline.
4. Single source of truth for the Pipeline, which can be viewed and edited by multiple members of the project.

While the syntax for defining a Pipeline, either in the web UI or with a Jenkinsfile is the same, it is generally considered best practice to define the Pipeline in a Jenkinsfile and check that in to source control.

What is SonarQube ?

SonarQube is a Code Quality Assurance tool that collects and analyzes source code, and provides reports for the code quality of your project. It combines static and dynamic analysis tools and enables quality to be measured continually over time. Everything from minor styling choices, to design errors are inspected and evaluated by SonarQube. This provides users with a rich searchable history of the code to analyze where the code is messing up and determine whether or not it is styling issues, code defeats, code duplication, lack of test coverage, or excessively complex code. The software will analyze source code from different aspects and drills down the code layer by layer, moving module level down to the class level, with each level producing metric values and statistics that should reveal problematic areas in the source code that needs improvement.

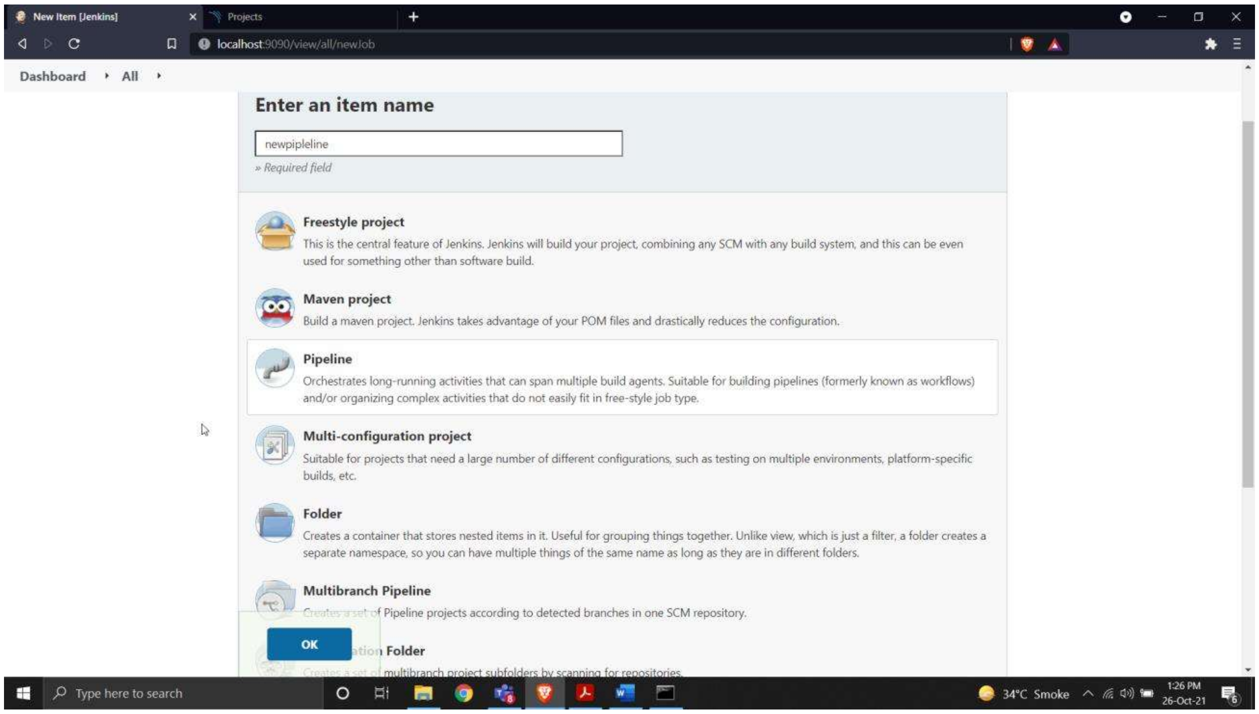
Sonarqube also ensures code reliability, Application security, and reduces technical debt by making your code base clean and maintainable. Sonarqube also provides support for 27 different languages, including C, C++, Java, Javascript, PHP, GO, Python, and much more.SonarQube also provides Ci/CD integration, and gives feedback during code review with branch analysis and pull request decoration.

Pre-requisites:

- Make sure SonarQube is up and running and do the below steps:
- Make sure SonarQube plug-in installed in Jenkins.

Stepwise Procedure

Step 1: Create a Pipleline project



Enter an item name

newpipleline

» 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.

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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 folders.

Multibranch Pipeline

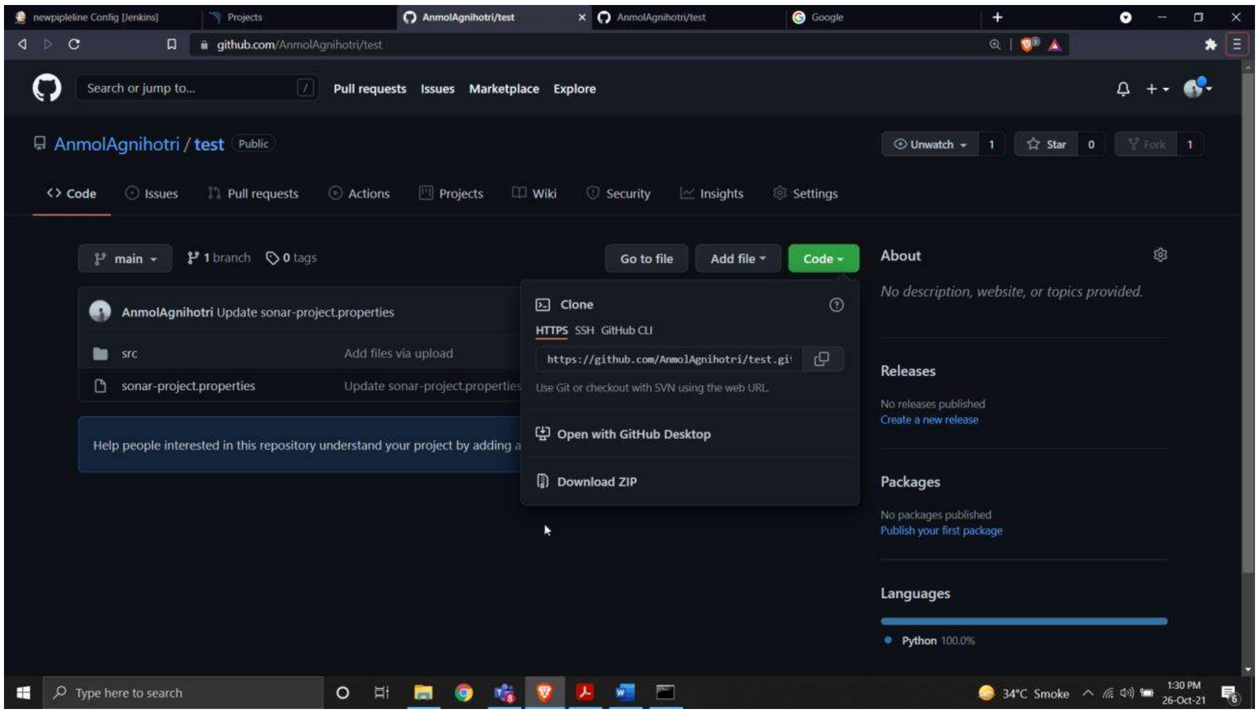
Creates a set of Pipeline projects according to detected branches in one SCM repository.

ation Folder

Creates a set of multibranch project subfolders by scanning for repositories.

Step 2: Select your project from github repository mine is

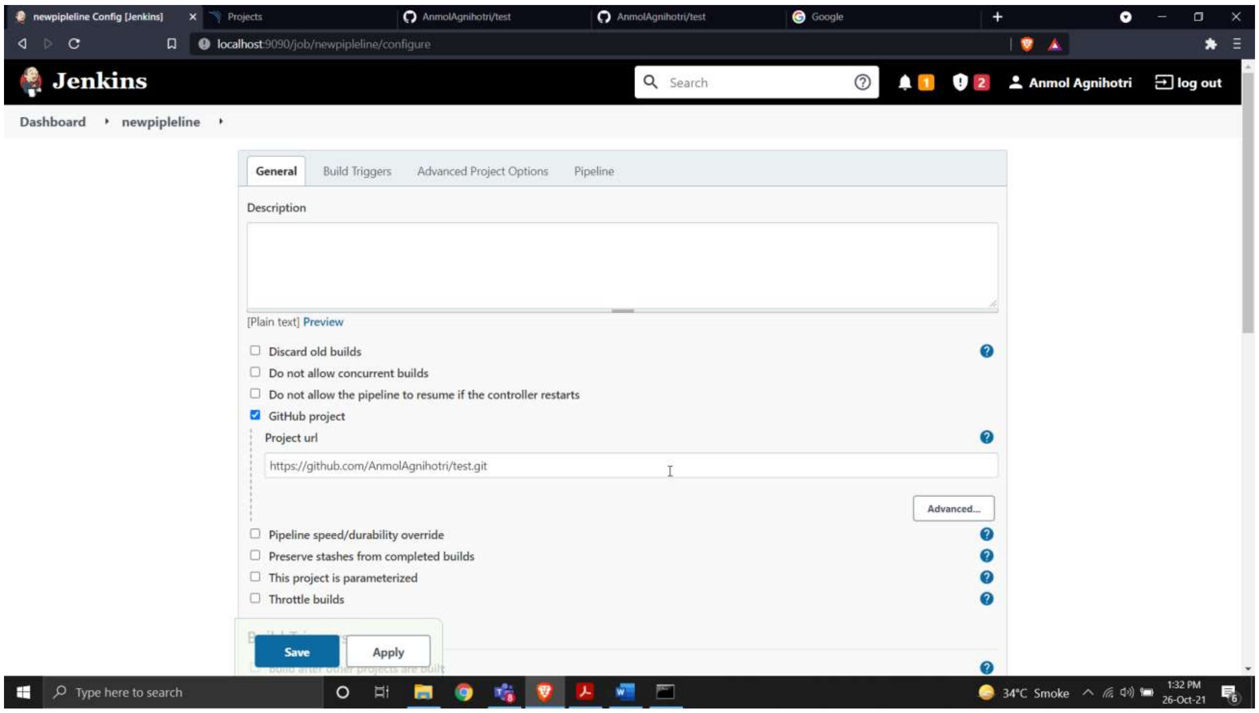
https://github.com/AnmolAgnihotri/test.git



https://github.com/AnmolAgnihotri/test.git

Use Git or checkout with SVN using the web URL.

Step 3: Go to General → Select GitHub Project → Paste the github repository URL in Jenkins



localhost:9090/job/newpipleline/configure

Dashboard ▸ newpipleline ▸

Description

[Plain text] Preview

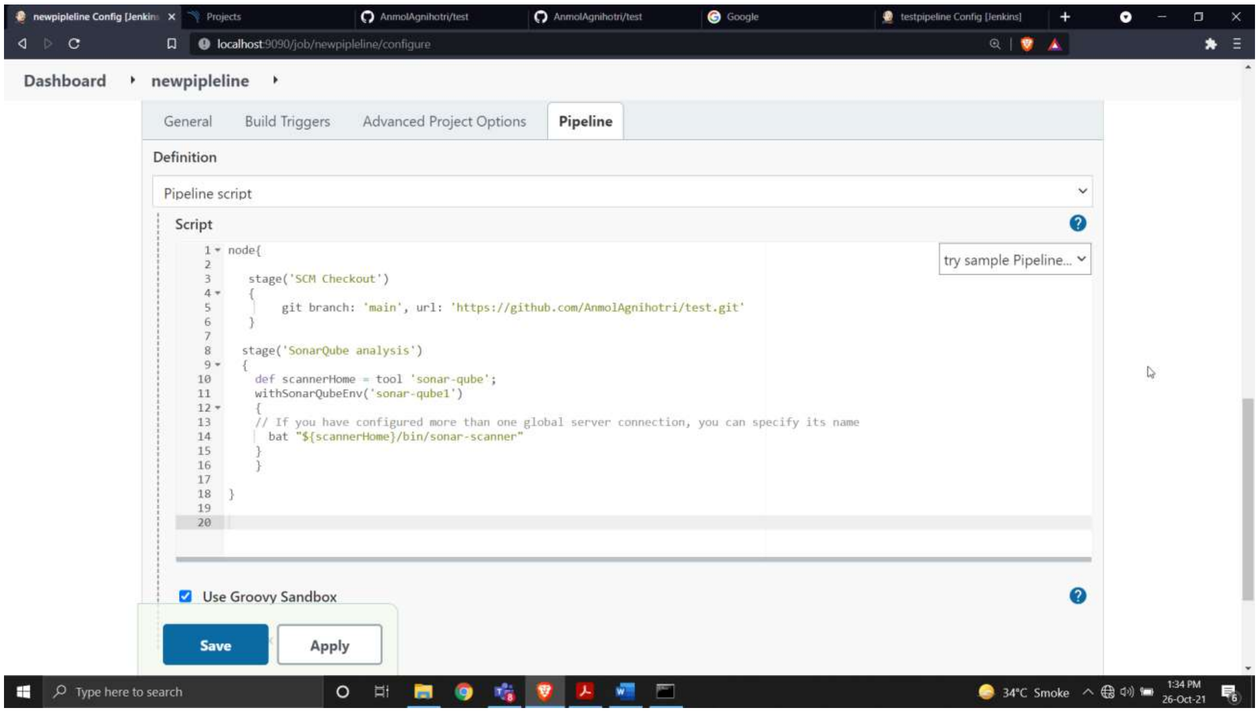
- ☐ Discard old builds
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts
- ☑ GitHub project

Project url: https://github.com/AnmolAgnihotri/test.git

- ☐ Pipeline speed/durability override
- ☐ Preserve stashes from completed builds
- ☐ This project is parameterized
- ☐ Throttle builds

Step 4: Go to **advance project options** → **pipeline** → paste the pipeline script

```
node{

   stage('SCM Checkout')
   {
        git branch: 'main', url: 'https://github.com/AnmolAgnihotri/test.git'
   }

  stage('SonarQube analysis')
  {
    def scannerHome = tool 'sonar-qube';
    withSonarQubeEnv('sonar-qube1')
    {
    // If you have configured more than one global server connection, you can
specify its name
      bat "${scannerHome}/bin/sonar-scanner"
    }
    }

}
```



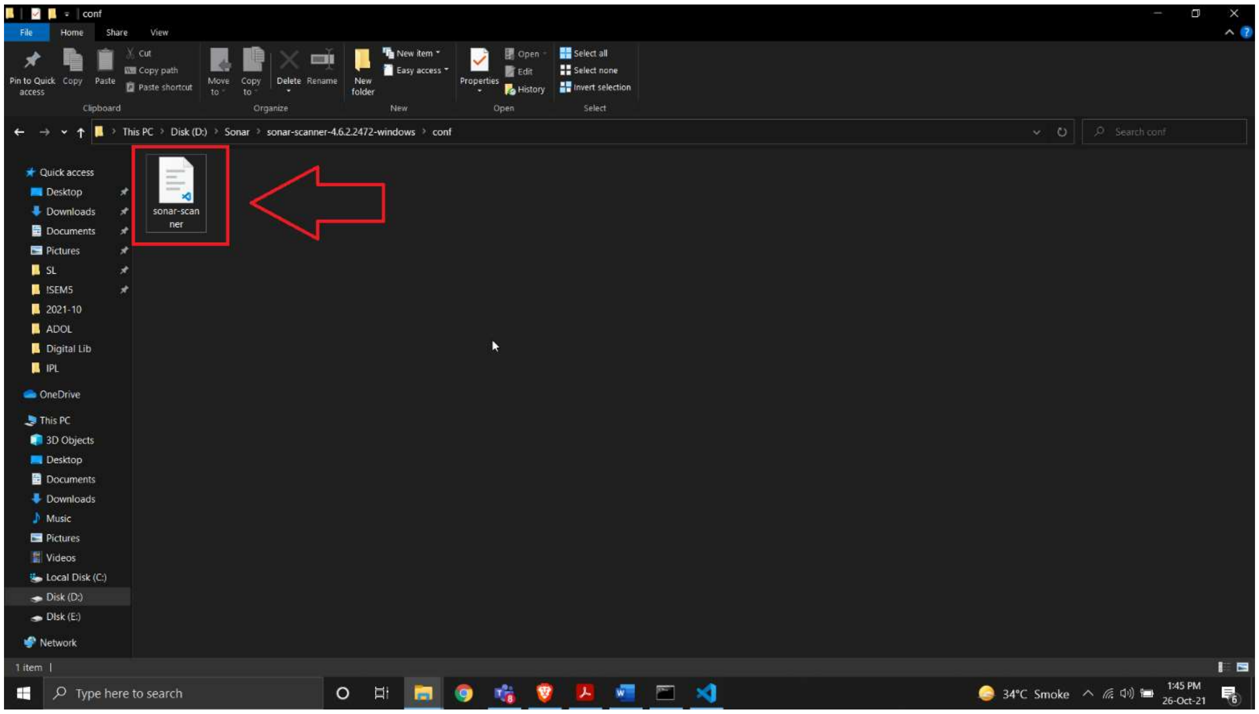
```
node{

   stage('SCM Checkout')
   {
        git branch: 'main', url: 'https://github.com/AnmolAgnihotri/test.git'
   }

  stage('SonarQube analysis')
  {
    def scannerHome = tool 'sonar-qube';
    withSonarQubeEnv('sonar-qube1')
    {
    // If you have configured more than one global server connection, you can specify its name
      bat "${scannerHome}/bin/sonar-scanner"
    }
    }

}
```

Step 5: Apply → Save

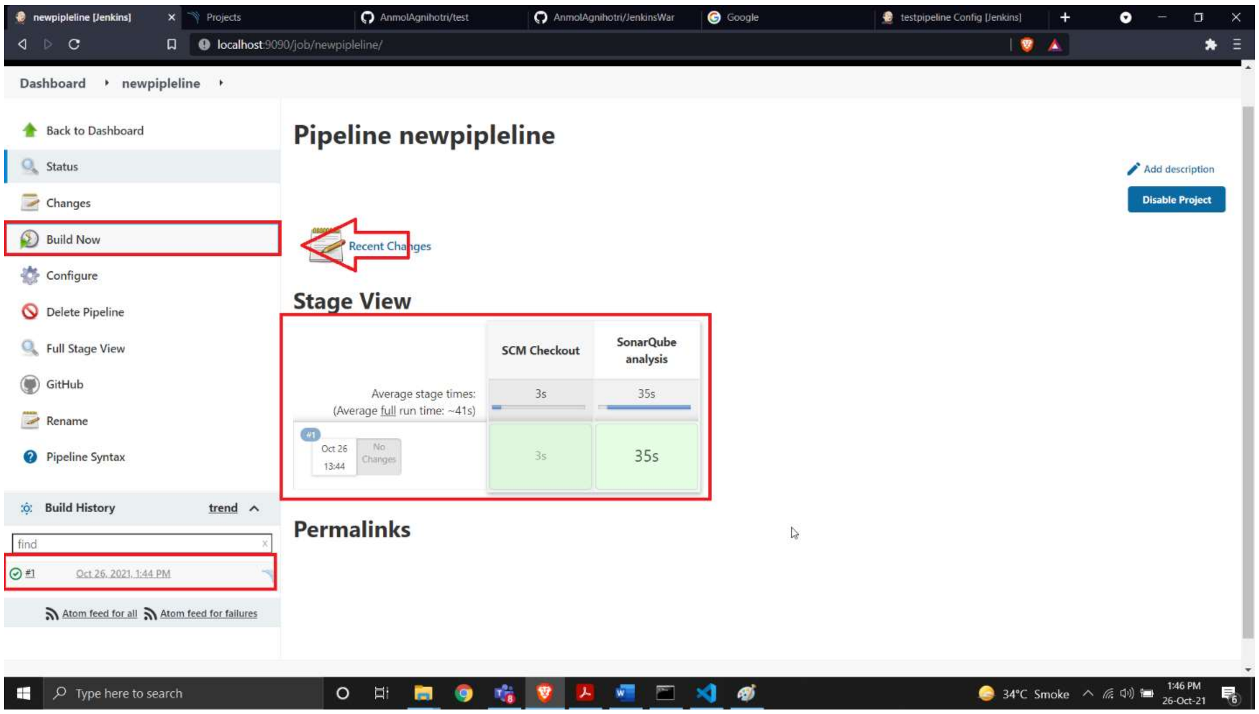
Step 6: Go to **SonarScanner(folder) → conf → sonar-scanner.properties** open this file and update source code for scanner

In my case path was D:\Sonar\sonar-scanner-4.6.2.2472-windows\conf\sonar-scanner.properties

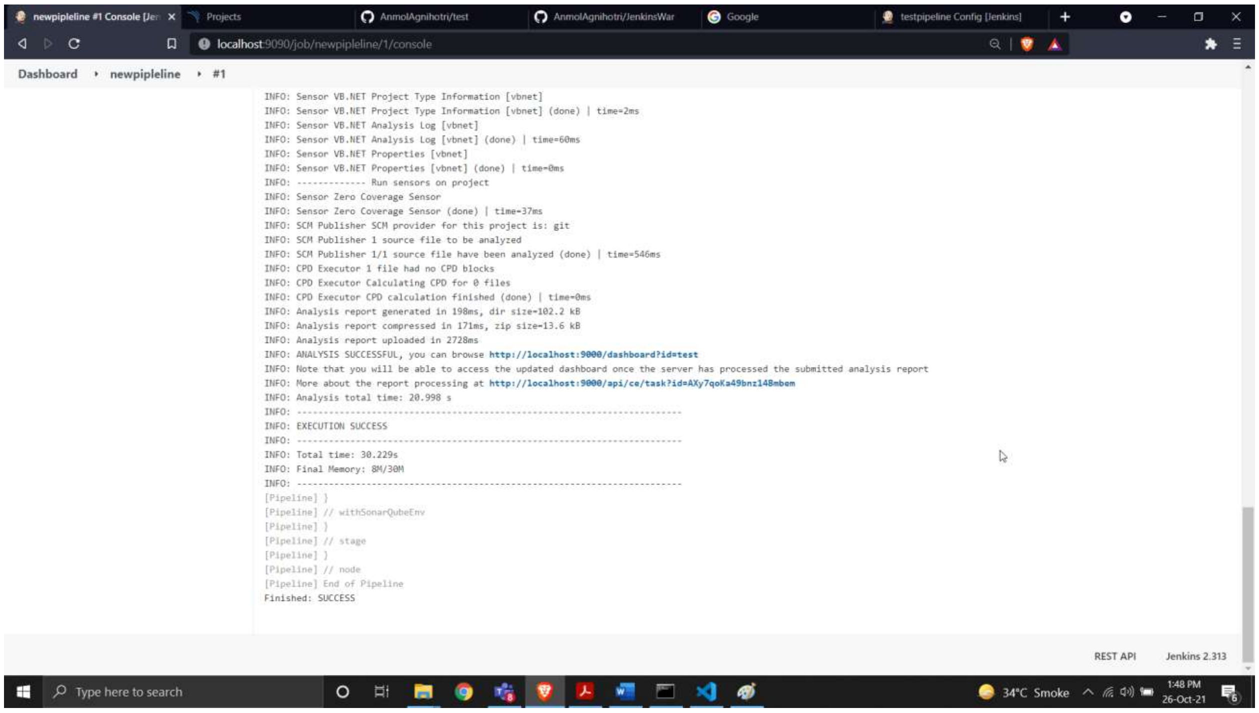
```
#Configure here general information about the environment, such as SonarQube
server connection details for example
#No information about specific project should appear here

#----- Default SonarQube server
sonar.host.url=http://localhost:9000

#----- Default source code encoding
sonar.sourceEncoding=UTF-8

sonar.projectKey=test
sonar.projectName=test
```


conf
File
Home
Share
View
This PC › Disk (D:) › Sonar › sonar-scanner-4.6.2.2472-windows › conf
sonar-scanner
1 item

Step 7: Go to Jenkins project and select build now



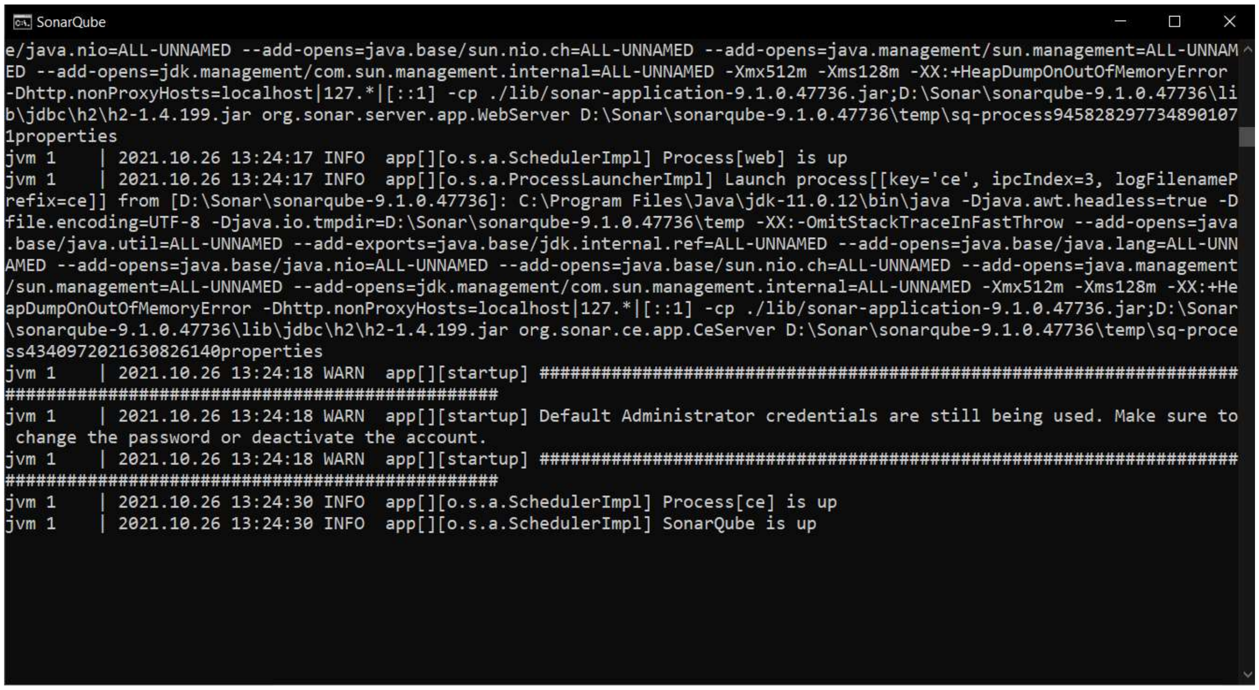
Go to build and open console output you will see
`Finished: SUCCESS`



```
INFO: Sensor VB.NET Project Type Information [vbnet]
INFO: Sensor VB.NET Project Type Information [vbnet] (done) | time=2ms
INFO: Sensor VB.NET Analysis Log [vbnet]
INFO: Sensor VB.NET Analysis Log [vbnet] (done) | time=60ms
INFO: Sensor VB.NET Properties [vbnet]
INFO: Sensor VB.NET Properties [vbnet] (done) | time=0ms
INFO: ------------- Run sensors on project
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=37ms
INFO: SCM Publisher SCM provider for this project is: git
INFO: SCM Publisher 1 source file to be analyzed
INFO: SCM Publisher 1/1 source file have been analyzed (done) | time=546ms
INFO: CPD Executor 1 file had no CPD blocks
INFO: CPD Executor Calculating CPD for 0 files
INFO: CPD Executor CPD calculation finished (done) | time=0ms
INFO: Analysis report generated in 198ms, dir size=102.2 kB
INFO: Analysis report compressed in 171ms, zip size=13.6 kB
INFO: Analysis report uploaded in 2728ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard?id=test
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AXy7qoKa49bnz148mbem
INFO: Analysis total time: 20.998 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 30.229s
INFO: Final Memory: 8M/30M
INFO: ------------------------------------------------------------------------
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

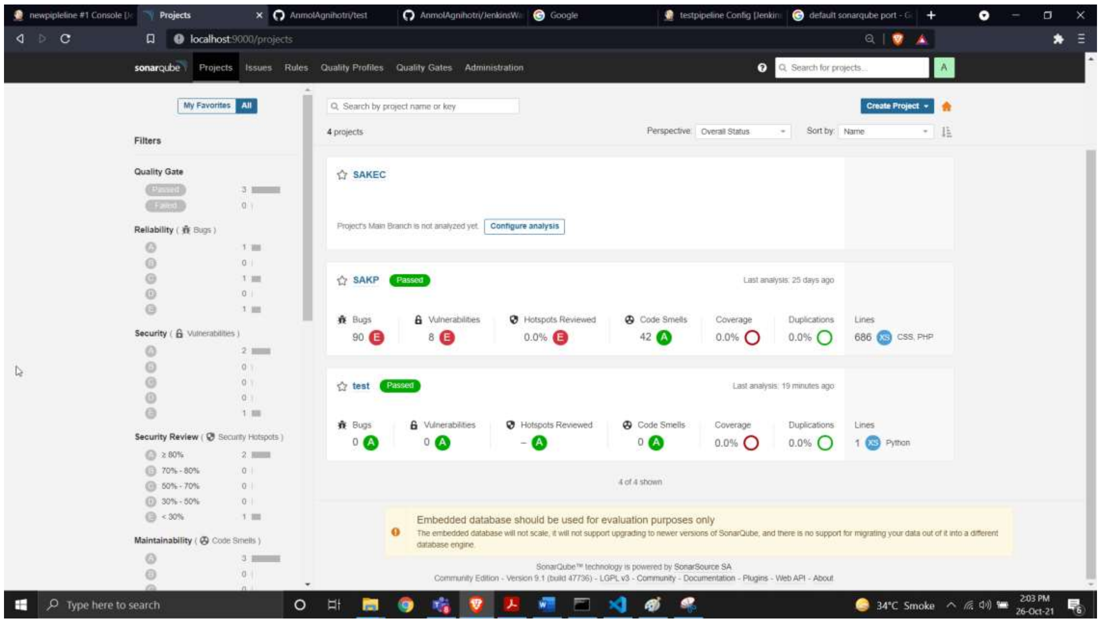
Step 8: Start sonar cube
To start sonar cube, go to

sonar-cube(folder) → bin → windows-x86-64 → StartSonar.bat
Run StartSonar.bat file → sonar server will start



```
SonarQube
e/java.nio=ALL-UNNAMED --add-opens=java.base/sun.nio.ch=ALL-UNNAMED --add-opens=java.management/sun.management=ALL-UNNAM
ED --add-opens=jdk.management/com.sun.management.internal=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryError
-Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/sonar-application-9.1.0.47736.jar;D:\Sonar\sonarqube-9.1.0.47736\li
b\jdbc\h2\h2-1.4.199.jar org.sonar.server.app.WebServer D:\Sonar\sonarqube-9.1.0.47736\temp\sq-process945828297734890107
1properties
jvm 1    | 2021.10.26 13:24:17 INFO  app[][o.s.a.SchedulerImpl] Process[web] is up
jvm 1    | 2021.10.26 13:24:17 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='ce', ipcIndex=3, logFilenameP
refix=ce]] from [D:\Sonar\sonarqube-9.1.0.47736]: C:\Program Files\Java\jdk-11.0.12\bin\java -Djava.awt.headless=true -D
file.encoding=UTF-8 -Djava.io.tmpdir=D:\Sonar\sonarqube-9.1.0.47736\temp -XX:-OmitStackTraceInFastThrow --add-opens=java
.base/java.util=ALL-UNNAMED --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNN
AMED --add-opens=java.base/java.nio=ALL-UNNAMED --add-opens=java.base/sun.nio.ch=ALL-UNNAMED --add-opens=java.management
/sun.management=ALL-UNNAMED --add-opens=jdk.management/com.sun.management.internal=ALL-UNNAMED -Xmx512m -Xms128m -XX:+He
apDumpOnOutOfMemoryError -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/sonar-application-9.1.0.47736.jar;D:\Sonar
\sonarqube-9.1.0.47736\lib\jdbc\h2\h2-1.4.199.jar org.sonar.ce.app.CeServer D:\Sonar\sonarqube-9.1.0.47736\temp\sq-proce
ss4340972021630826140properties
jvm 1    | 2021.10.26 13:24:18 WARN  app[][startup] ####################################################################
################################################
jvm 1    | 2021.10.26 13:24:18 WARN  app[][startup] Default Administrator credentials are still being used. Make sure to
 change the password or deactivate the account.
jvm 1    | 2021.10.26 13:24:18 WARN  app[][startup] ####################################################################
################################################
jvm 1    | 2021.10.26 13:24:30 INFO  app[][o.s.a.SchedulerImpl] Process[ce] is up
jvm 1    | 2021.10.26 13:24:30 INFO  app[][o.s.a.SchedulerImpl] SonarQube is up
```

Step 9: Open Sonar Cube in browser localhost
http://localhost:9000/
in my case port is 9000 (Your port may differ)



4 projects

SAKEC
Project's Main Branch is not analyzed yet.

SAKP Passed — Last analysis: 25 days ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
90 E	8 E	0.0% E	42 A	0.0%	0.0%	686 XS CSS, PHP

test Passed — Last analysis: 19 minutes ago

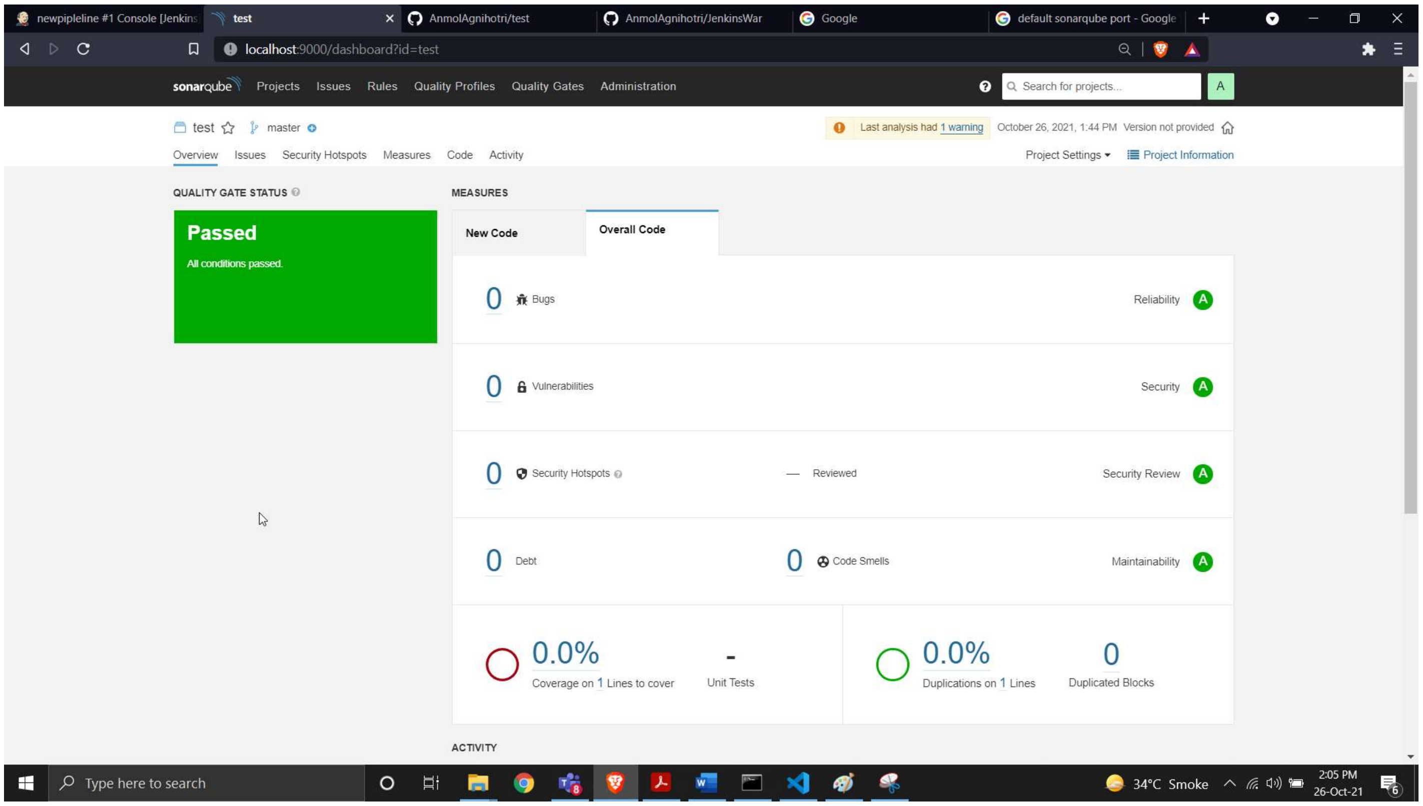
Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
0 A	0 A	– A	0 A	0.0%	0.0%	1 XS Python

4 of 4 shown

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.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.1 (build 47736) - LGPL v3 - Community - Documentation - Plugins - Web API - About

There will be project named on your GitHub repository in my case my repository name was **test → click on it**



This is the final analysis report for our code

Conclusion:

Created a Jenkins CICD pipeline using GitHub repository, successfully connected it with SonarQube and generated code analysis report