Name: Manav Jawrani

Roll No.: 19

Subject: Advanced DevOps

Experiment No.: 7

Experiment 7.

Aim: To understand static analysis SAST process and learn to integrate Jenkins SAST to SonarQube /GitLab.

Theory:

- What is SonarQube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications It supports 25+ major programming languages throug built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube -

1. Sustainability- Reduces complexity, possible vulnerabilities and code duplications, optimising the applications.
2. Increase productivity- Redules the scale, cost of maintenance and risk of the application.
3. Quaity code - Code Quality control is an inseparable part of the process of software development.
4. Detect Errors- Detect Errors in the code and alerts developers to fix them automatically before submitting them for output.

- What is SAST?

Static Application Security ~~Tess~~ Testing (SAST), or Static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your application susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

- What are the key steps to run effectively?

1. Finalize the tool
2. Create the scanning infrastructure and deploy the tool.
3. Customize the tool.
4. Prioritize and onboard applications.
5. Analyze scan results.
6. Provide governance and training

Implementation:

Prerequisites:

- Docker Installed

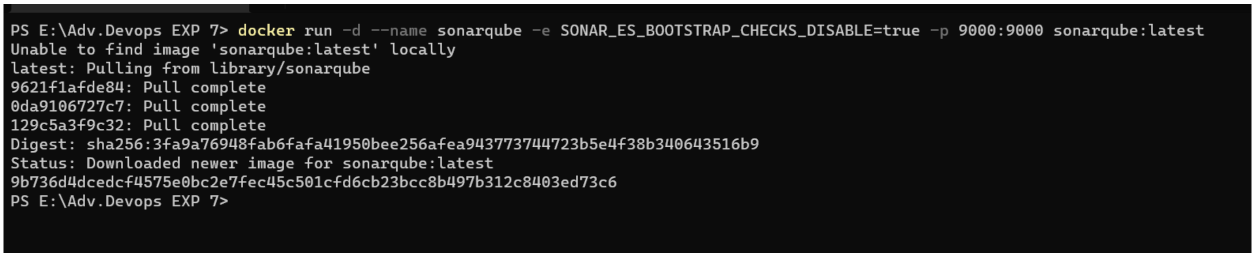
Download from here: https://www.docker.com

- Jenkins

Download from here: https://www.jenkins.io/download/

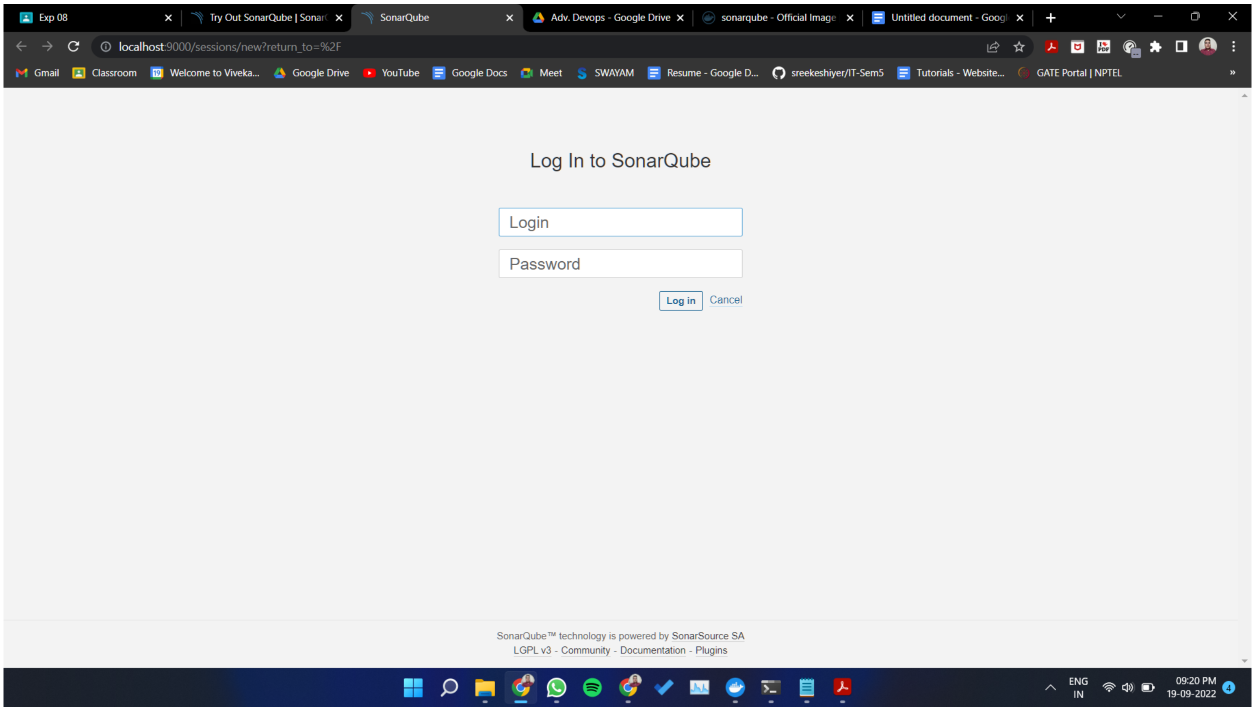
Step 1: Installing SonarQube from the Docker Image

$ docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest



```
PS E:\Adv.Devops EXP 7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
9621f1afde84: Pull complete
0da9106727c7: Pull complete
129c5a3f9c32: Pull complete
Digest: sha256:3fa9a76948fab6fafa41950bee256afea943773744723b5e4f38b340643516b9
Status: Downloaded newer image for sonarqube:latest
9b736d4dcedcf4575e0bc2e7fec45c501cfd6cb23bcc8b497b312c8403ed73c6
PS E:\Adv.Devops EXP 7>
```

Step 2: After installation of SonarQube, go to the SonarQube page by typing: http://localhost:9000/ on your browser. If you see such page then you have successfully installed it.



localhost:9000/sessions/new?return_to=%2F

Log In to SonarQube

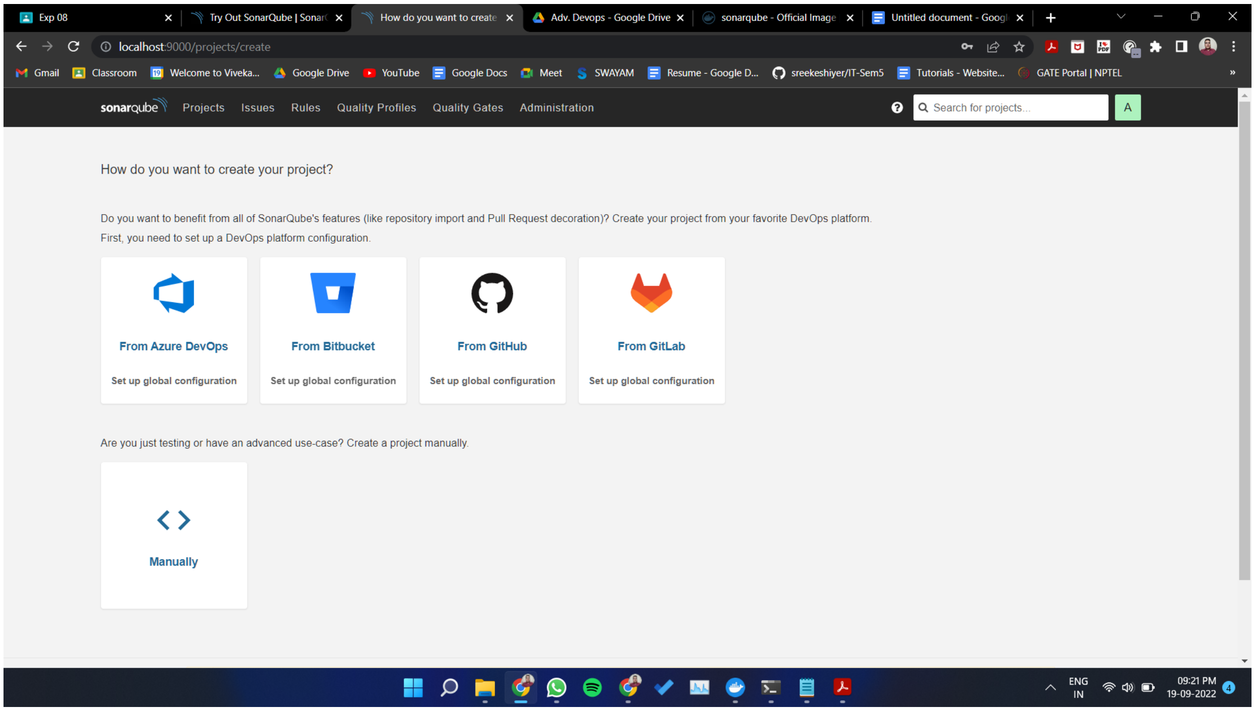
Login

Password

Log in Cancel

SonarQube™ technology is powered by SonarSource SA
LGPL v3 - Community - Documentation - Plugins

Step 3: Login using the username as “admin” and password as “admin”. And then you will see the home page of SonarQube.



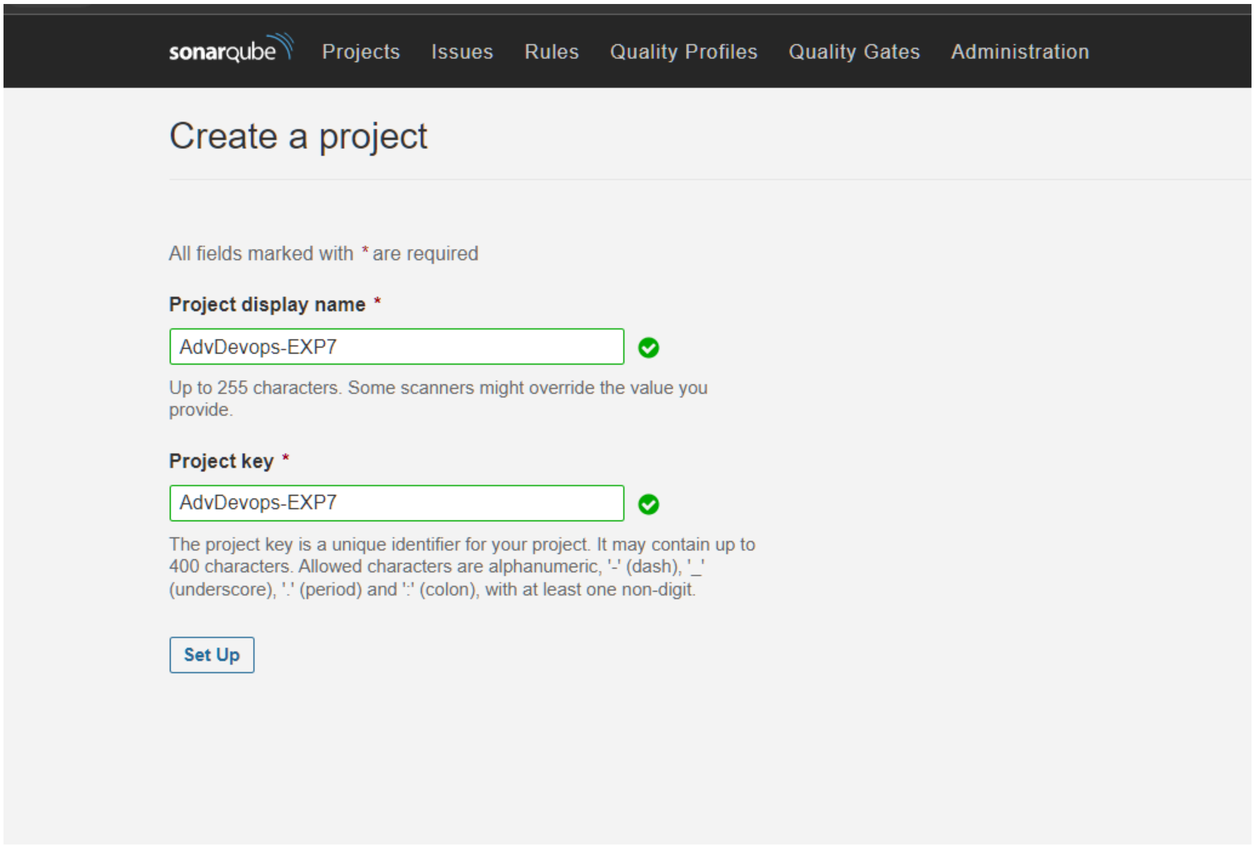
How do you want to create your project?

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.

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

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

Step 4: Create a manual project in SonarQube with the name “AdvDevops-EXP7” (in my case) and set up the project.



Create a project

All fields marked with * are required

Project display name *

AdvDevops-EXP7

Up to 255 characters. Some scanners might override the value you provide.

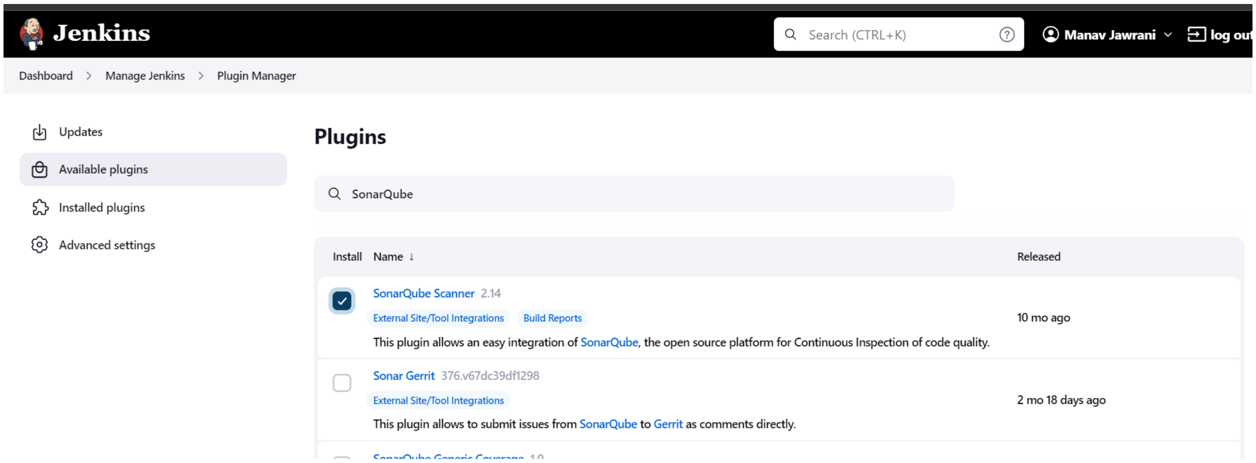
Project key *

AdvDevops-EXP7

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Now open the Jenkins Dashboard in the new tab of the browser.

Step 5: Go to Dashboard > Manage Jenkins > Plugin Manager and search for SonarQube Scanner under Available plugins for Jenkins and install it.

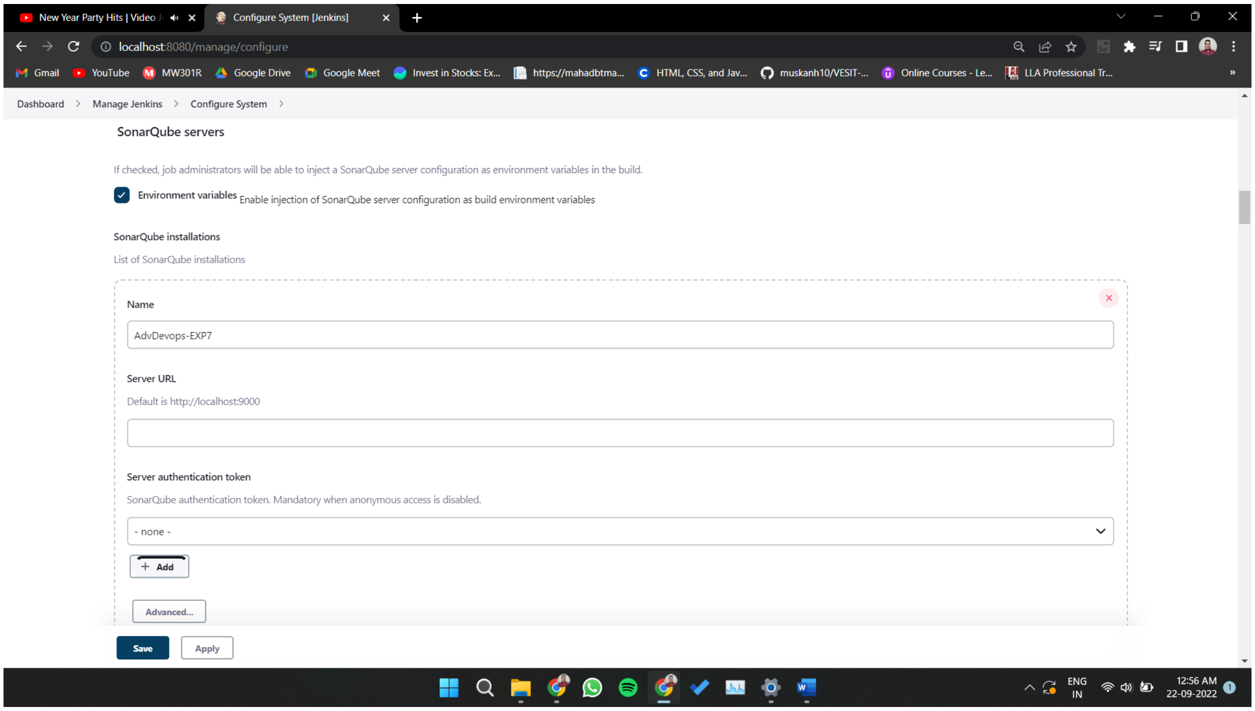


Plugins

SonarQube

Install	Name ↓	Released
☑	SonarQube Scanner 2.14 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	10 mo ago
□	Sonar Gerrit 376.v67dc39df1298 External Site/Tool Integrations This plugin allows to submit issues from SonarQube to Gerrit as comments directly.	2 mo 18 days ago
□	SonarQube Generic Coverage 1.0	

Step 6: Under Jenkins , Dashboard > Manage Jenkins > Configure System , look for SonarQube Servers and enter the details. Enter the Server Authentication Token if needed.



Dashboard > Manage Jenkins > Configure System >

SonarQube servers

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

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

SonarQube installations

List of SonarQube installations

Name

AdvDevops-EXP7

Server URL

Default is http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

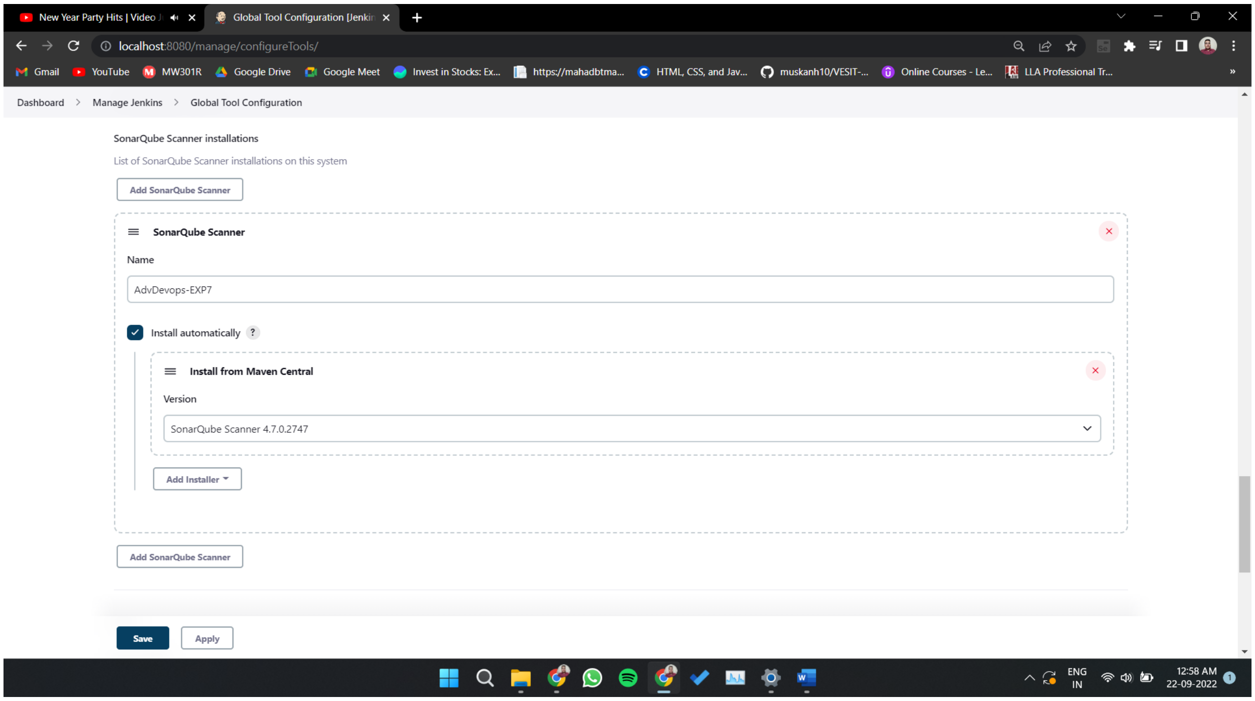
- none -

+ Add

Advanced...

Save Apply

Step 7: Search SonarQube Scanner under Dashboard > Manage Jenkins > Global Tool Configuration. Choose the latest configuration and choose Install Automatically.



Dashboard > Manage Jenkins > Global Tool Configuration

SonarQube Scanner installations

List of SonarQube Scanner installations on this system

Add SonarQube Scanner

SonarQube Scanner

Name

AdvDevops-EXP7

☑ Install automatically ?

Install from Maven Central

Version

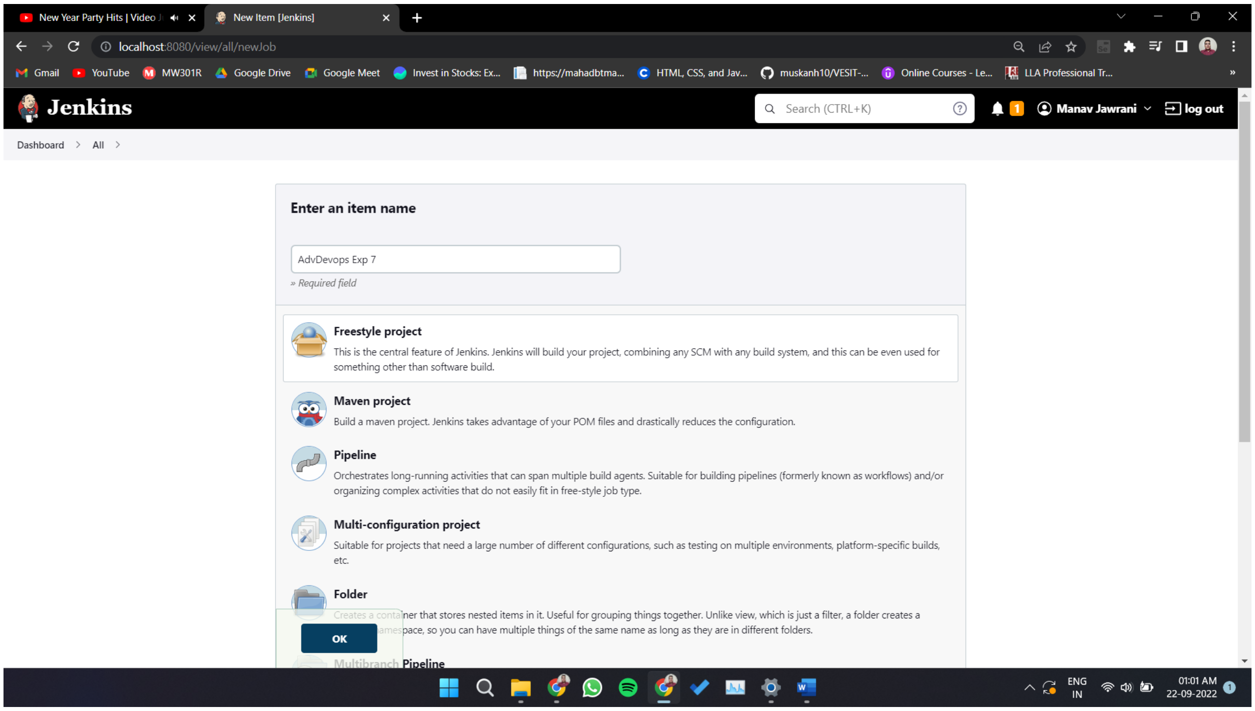
SonarQube Scanner 4.7.0.2747

Add Installer ▾

Add SonarQube Scanner

Save Apply

Step 8: After the configuration, create a New Item in Jenkins, choose a freestyle project.



Enter an item name

AdvDevops Exp 7

» 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.

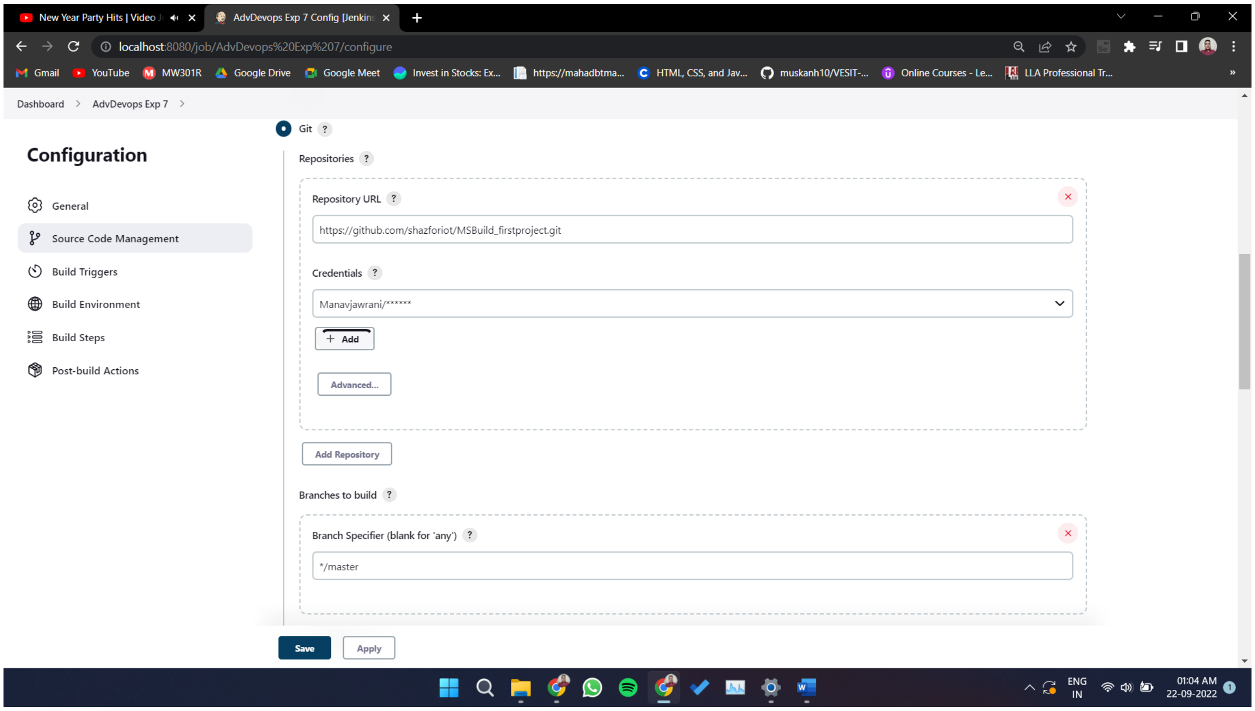
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

OK

Multibranch Pipeline

Step 9: Choose this GitHub repository in Source Code Management. https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



● Git

Repositories

Repository URL: https://github.com/shazforiot/MSBuild_firstproject.git

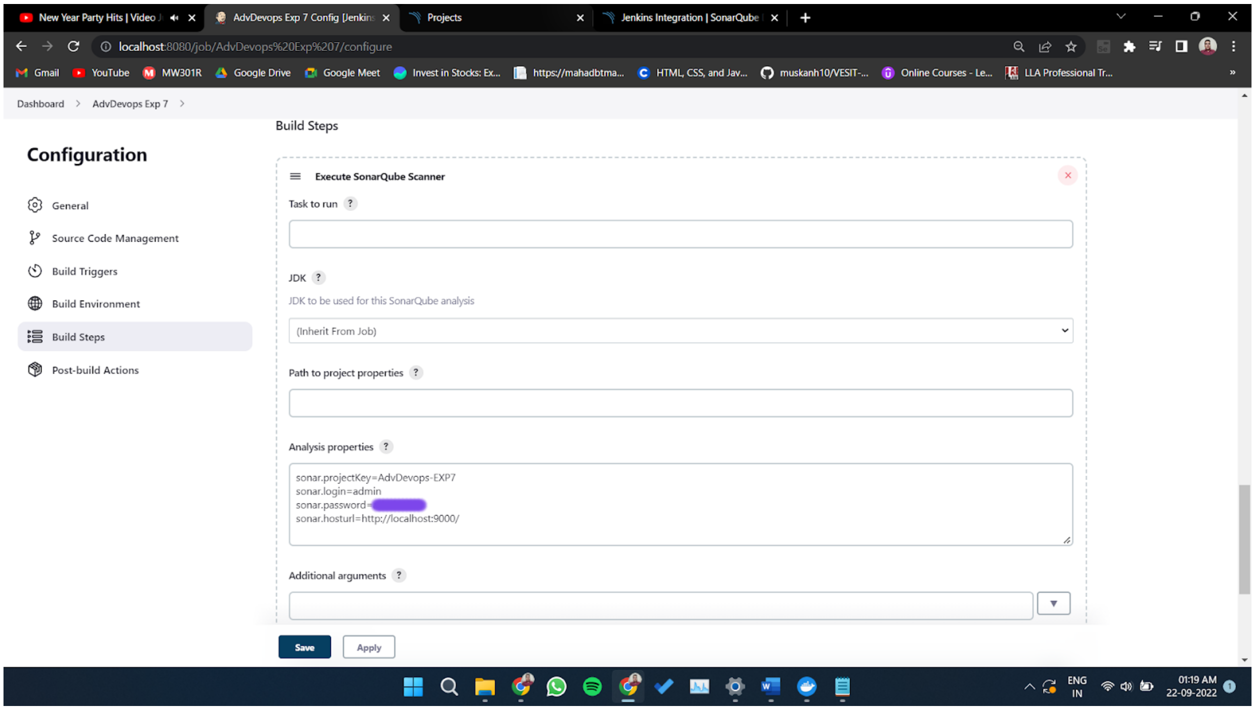
Credentials: Manavjawrani/******

Branches to build

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

Step 10: Under Build > Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, and Host URL.

sonar.projectKey=AdvDevops-EXP7
sonar.login=*your username*
sonar.password=*your password*
sonar.hosturl=http://localhost:9000/



Build Steps

Execute SonarQube Scanner

Task to run

JDK

JDK to be used for this SonarQube analysis

(Inherit From Job)

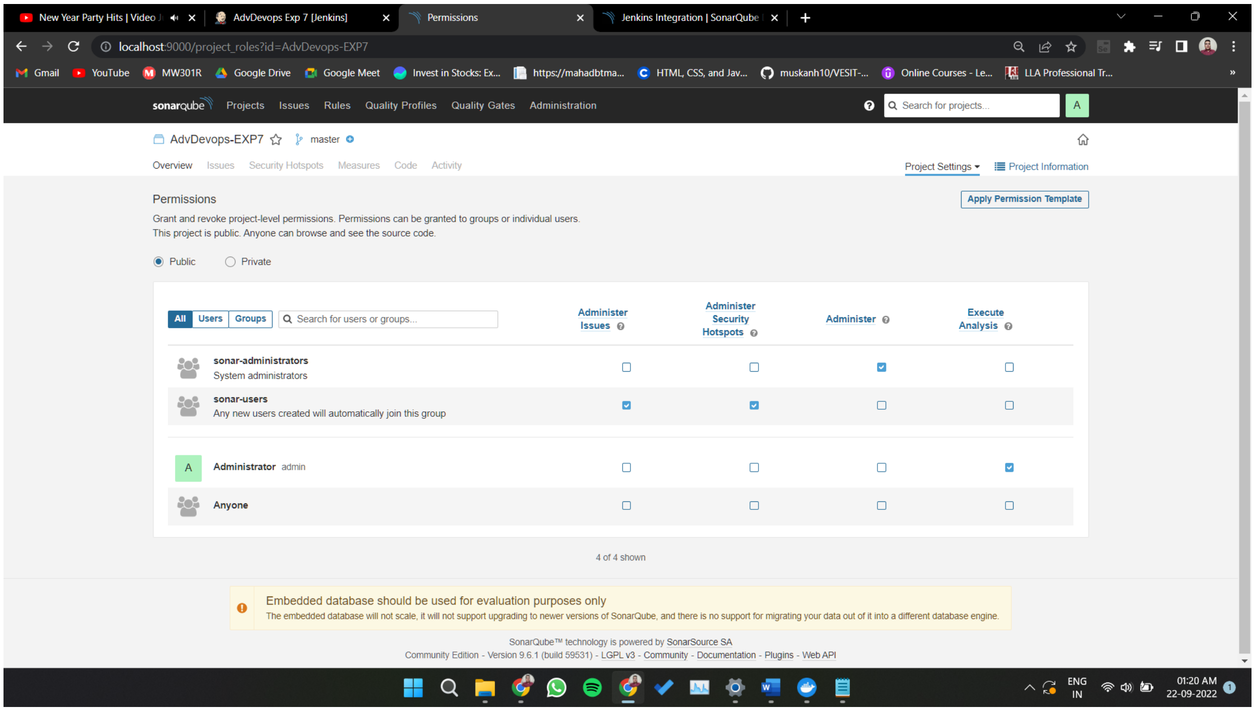
Path to project properties

Analysis properties

```
sonar.projectKey=AdvDevops-EXP7
sonar.login=admin
sonar.password=[redacted]
sonar.hosturl=http://localhost:9000/
```

Additional arguments

Step 11: Go to http://localhost:9000/ and enter your previously created username. Go to Permissions and grant the Admin user Execute Permissions.



Permissions

Grant and revoke project-level permissions. Permissions can be granted to groups or individual users.
This project is public. Anyone can browse and see the source code.

● Public ○ Private

All | Users | Groups — Search for users or groups...

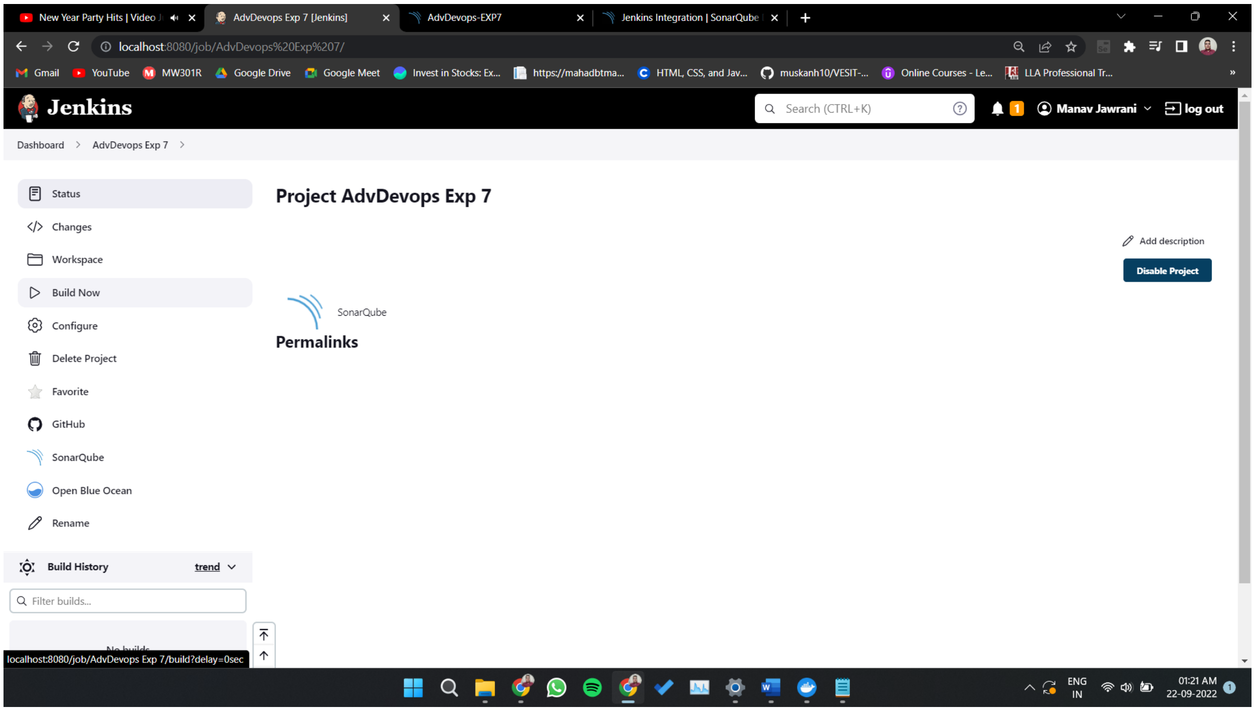
	Administer Issues	Administer Security Hotspots	Administer	Execute Analysis
sonar-administrators System administrators	□	□	☑	□
sonar-users Any new users created will automatically join this group	☑	☑	□	□
A Administrator admin	□	□	□	☑
Anyone	□	□	□	□

4 of 4 shown

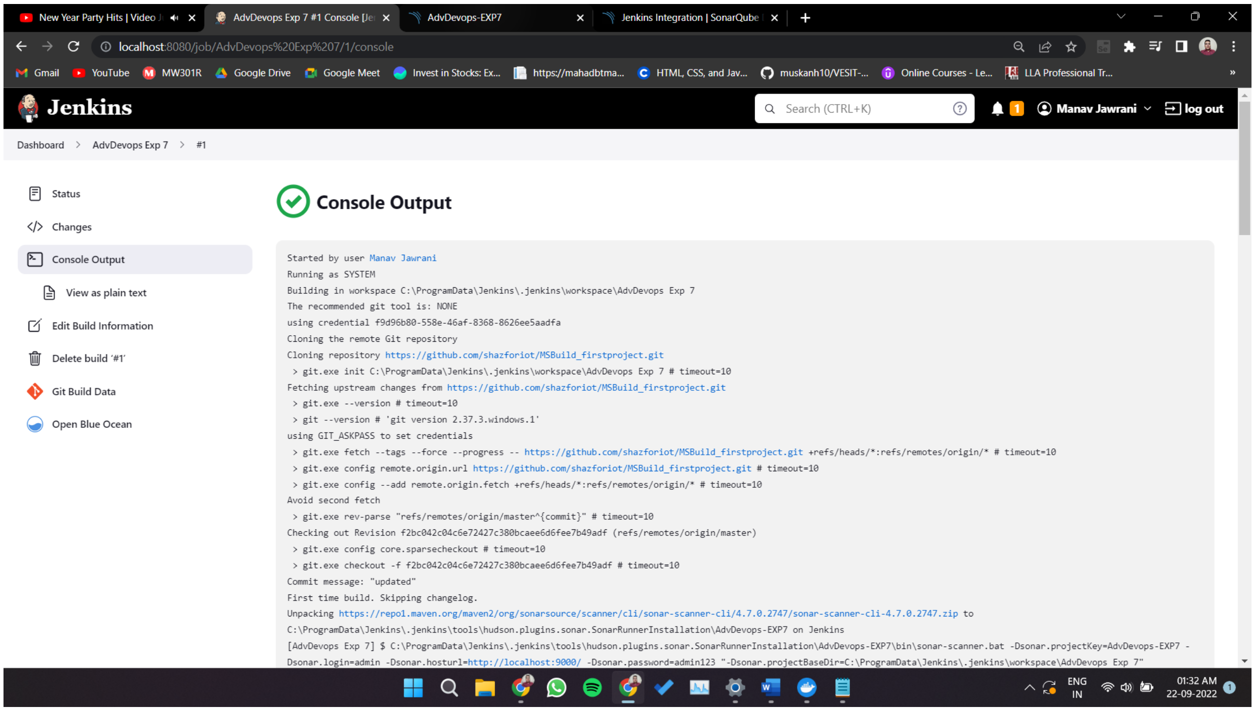
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.6.1 (build 59531) - LGPL v3 - Community - Documentation - Plugins - Web API

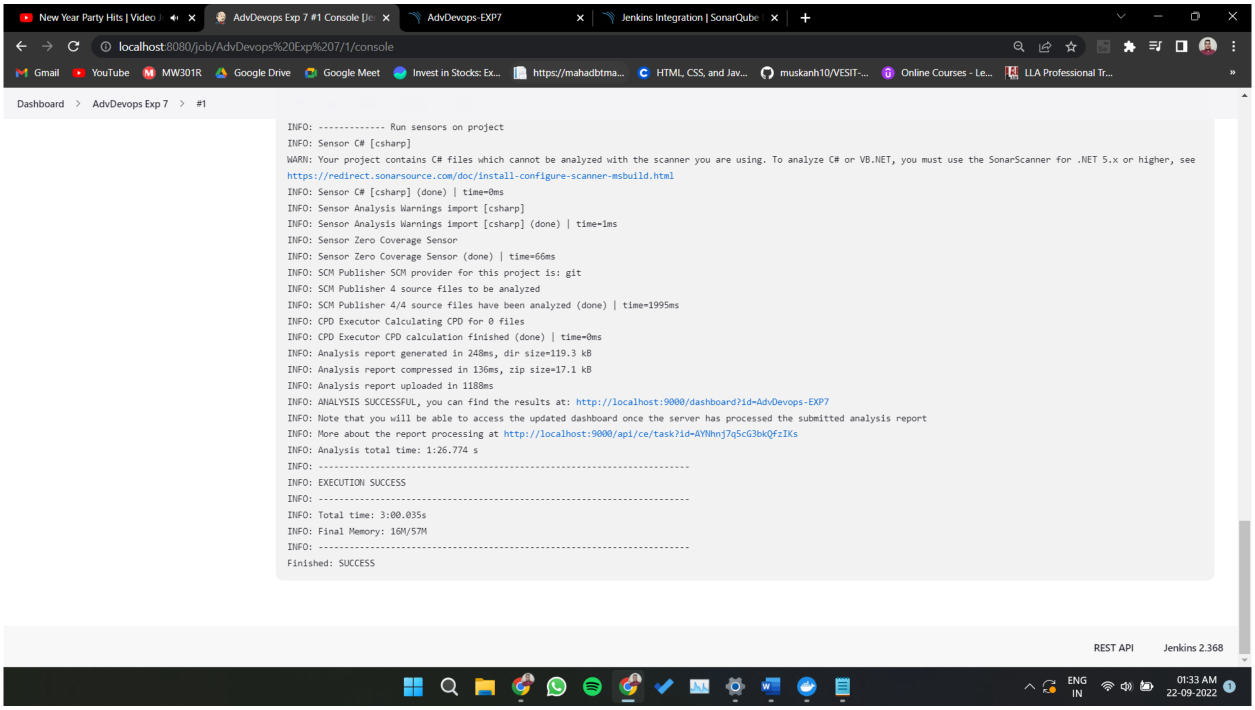
Step 12: Run The Build.



Check the console output.

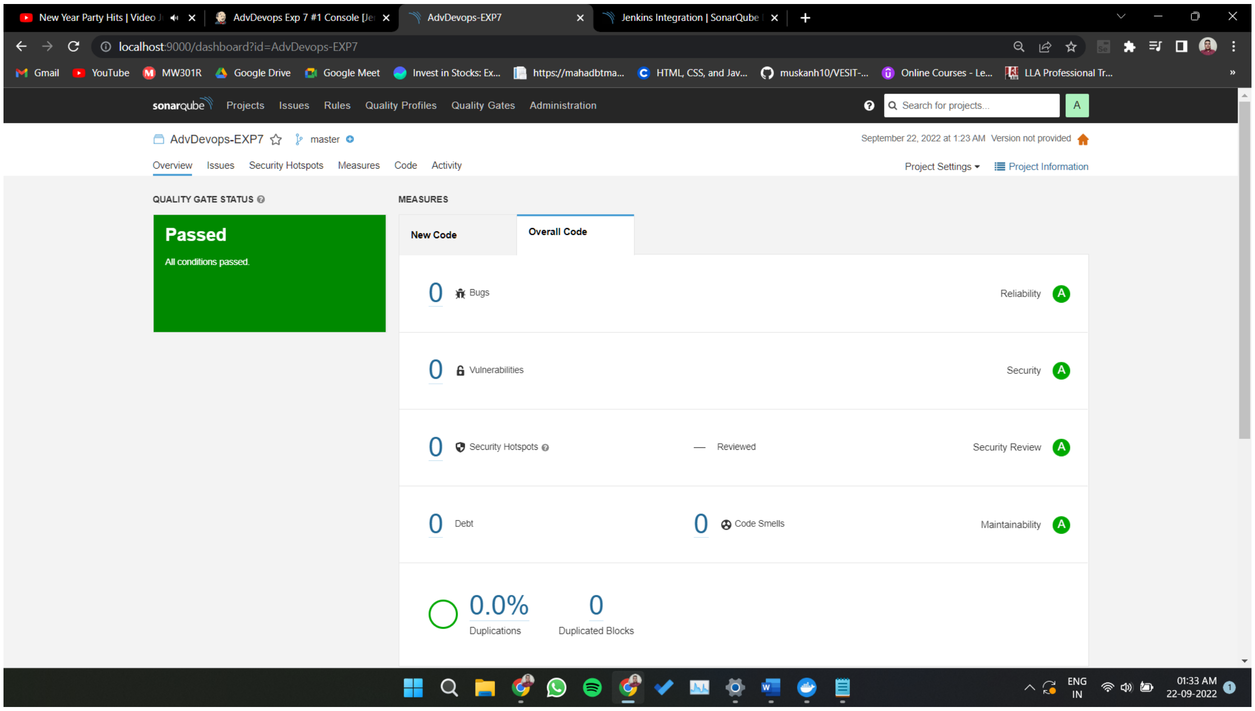


```
Started by user Manav Jawrani
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\AdvDevops Exp 7
The recommended git tool is: NONE
using credential f9d96b80-558e-46af-8368-8626ee5aadfa
Cloning the remote Git repository
Cloning repository https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe init C:\ProgramData\Jenkins\.jenkins\workspace\AdvDevops Exp 7 # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.37.3.windows.1'
using GIT_ASKPASS to set credentials
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
 > git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
First time build. Skipping changelog.
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/4.7.0.2747/sonar-scanner-cli-4.7.0.2747.zip to
C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\AdvDevops-EXP7 on Jenkins
[AdvDevops Exp 7] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\AdvDevops-EXP7\bin\sonar-scanner.bat -Dsonar.projectKey=AdvDevops-EXP7 -
Dsonar.login=admin -Dsonar.hosturl=http://localhost:9000/ -Dsonar.password=admin123 "-Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\AdvDevops Exp 7"
```


localhost:8080/job/AdvDevops%20Exp%207/1/console
Dashboard > AdvDevops Exp 7 > #1

```
INFO: ------------- Run sensors on project
INFO: Sensor C# [csharp]
WARN: Your project contains C# files which cannot be analyzed with the scanner you are using. To analyze C# or VB.NET, you must use the SonarScanner for .NET 5.x or higher, see
https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
INFO: Sensor C# [csharp] (done) | time=0ms
INFO: Sensor Analysis Warnings import [csharp]
INFO: Sensor Analysis Warnings import [csharp] (done) | time=1ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=66ms
INFO: SCM Publisher SCM provider for this project is: git
INFO: SCM Publisher 4 source files to be analyzed
INFO: SCM Publisher 4/4 source files have been analyzed (done) | time=1995ms
INFO: CPD Executor Calculating CPD for 0 files
INFO: CPD Executor CPD calculation finished (done) | time=0ms
INFO: Analysis report generated in 248ms, dir size=119.3 kB
INFO: Analysis report compressed in 136ms, zip size=17.1 kB
INFO: Analysis report uploaded in 1188ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=AdvDevops-EXP7
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=AYNhnj7q5cG3bkQfzIKs
INFO: Analysis total time: 1:26.774 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 3:00.035s
INFO: Final Memory: 16M/57M
INFO: ------------------------------------------------------------------------
Finished: SUCCESS
```

Step 13: Once the build is complete, check the project in SonarQube.



September 22, 2022 at 1:23 AM Version not provided

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code | Overall Code

0 Bugs — Reliability A

0 Vulnerabilities — Security A

0 Security Hotspots — Reviewed — Security Review A

0 Debt — 0 Code Smells — Maintainability A

0.0% Duplications — 0 Duplicated Blocks

In this way, we have integrated Jenkins with SonarQube for SAST.

Conclusion :

Thus with the implementation of this Experiment, we first learned about SAST, which may be very beneficial to the correct examine of Code. There might be certain problems that we could overlook , but SAST also identifies those. We have integrated Sonar Qube with Jenkins so that we can SAST our projects that will be running on Jenkins. SonarQube is a platform that aids in SAST and provides the output in concise manner that allows user to understand errors and many other things.