

Karthik Pradeep Hegadi

2KE20CS032

Assignment 27

Understood. To follow the provided instructions and create the files/directory using the same name and case as provided in the task steps, please provide me with the specific names and case instructions for the files/directory you want to create.

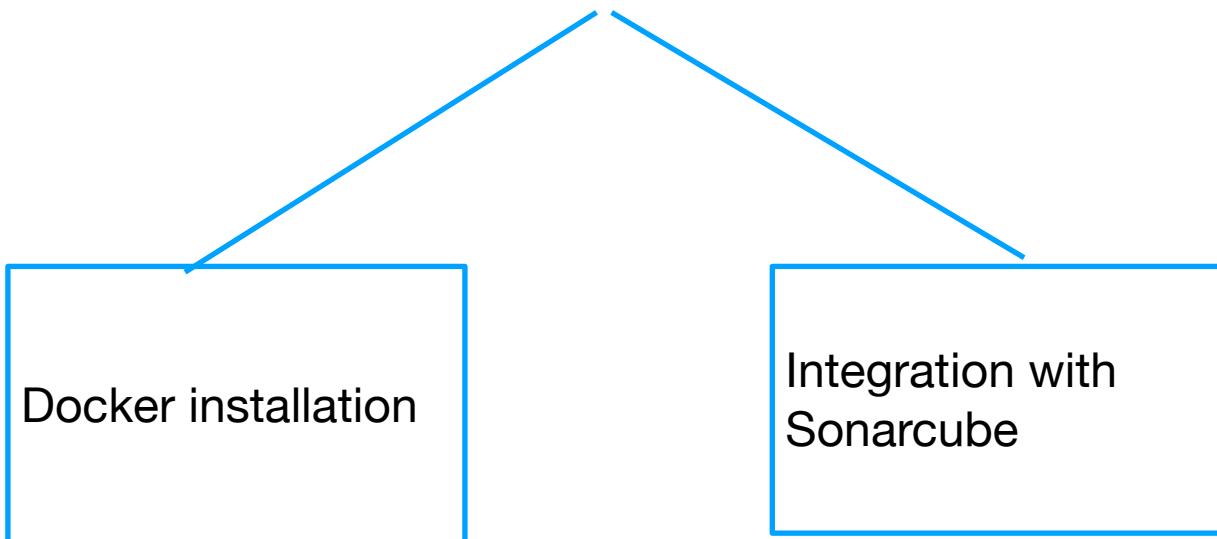
Jenkins -3

Assignment 3 :- Integrate SonarQube scanner into Pipeline

What is sonarqube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells on 29 programming languages

Integrate SonarQube scanner into Pipeline



Docker Installation

Integration with Sonarcube

1. In the jenkins, install the following plugins sonarqube scanner for jenkins and pipeline maven integration

The screenshot shows the Jenkins Plugins page. A search bar at the top contains the text "scanne". Below it, a list of installed and enabled plugins is shown:

- SonarQube Scanner for Jenkins 2.16.1**: Enabled. Description: This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Status: Enabled. Action buttons: a green hexagonal icon with a checkmark and a red hexagonal icon with a minus sign.
- Maven Integration plugin 3.23**: Enabled. Description: This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit. Status: Enabled. Action buttons: a green hexagonal icon with a checkmark and a red hexagonal icon with a minus sign.
- Pipeline Maven Integration Plugin 1345.va_0ef5530a_5ca_**: Enabled. Description: This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path. Status: Enabled. Action buttons: a green hexagonal icon with a checkmark and a red hexagonal icon with a minus sign.

3. Login to your sonarqube server -> Administration -> User -> Security -> tokens
4. Generate a token for jenkins and save it

The screenshot shows the SonarQube Security Tokens page for the "Administrator" user. The title is "Tokens of Administrator".

Generate Tokens

Name	Expires in
Enter Token Name	30 days

Actions

Name	Type	Project	Last use	Created	Expiration	Actions
jenkins	User		< 1 hour ago	October 14, 2023	January 12, 2024	Revoke

[Done](#)

Administration

Configuration ▾ Security ▾ Projects ▾ System Marketplace

Users Create and administer individual users.

Search by login or name... All users

Name	SCM Accounts	Last connection	Last SonarLint connection	Groups	Tokens	Actions
A Administrator admin		< 1 hour ago	Never	2	1 <input type="button" value="Update tokens"/>	

1 of 1 shown

Jenkins

Dashboard > Manage Jenkins > Credentials

Credentials KEY Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	sonarcube_key	sonarcube_key

Stores scoped to Jenkins

5. Navigate to Manage jenkins -> configure system ->sonarqube servers

Dashboard > Manage Jenkins > System > path

SonarQube servers

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

Environment variables

SonarQube installations

List of SonarQube installations

Name
sonar

Server URL

Default is http://localhost:9000

http://10.211.55.20:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonarcube_key

Add Advanced

The same name which we have mentioned while creating KEY

In above step Provide the name sonar, your sonarqube server url and port, in the dropdown select

the secret text that is generated from your server

(Note : You need to add sonarqube credentials in Manage jenkins -> credentials -> system -> Global credentials. Select kind and copy sonarqube secret key, provide any id and description and add it)

7. Now create a pipeline using the script placed in the link

Enter an item name

SonarQube_pipeline_script|
» A job already exists with the name 'SonarQube_pipeline_script'

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.

Dashboard > SonarQube_pipeline_script > Configuration path

Configure **Pipeline**

General Advanced Project Options Pipeline

Definition Pipeline script

Script ?

```
1 * pipeline {
2   agent any
3   stages{
4     stage('Source') {
5       steps{
6         git branch: 'main', url: 'https://github.com/maraks-gradious/maven-test1'
7       }
8     }
9     stage('Build') {
10    steps {
11      withSonarQubeEnv('sonar'){
12        // Optionally use a Maven environment you've configured already
13        withMaven{
14          sh "mvn clean verify sonar:sonar -X"
15        }
16      }
17    }
18  }
19}
```

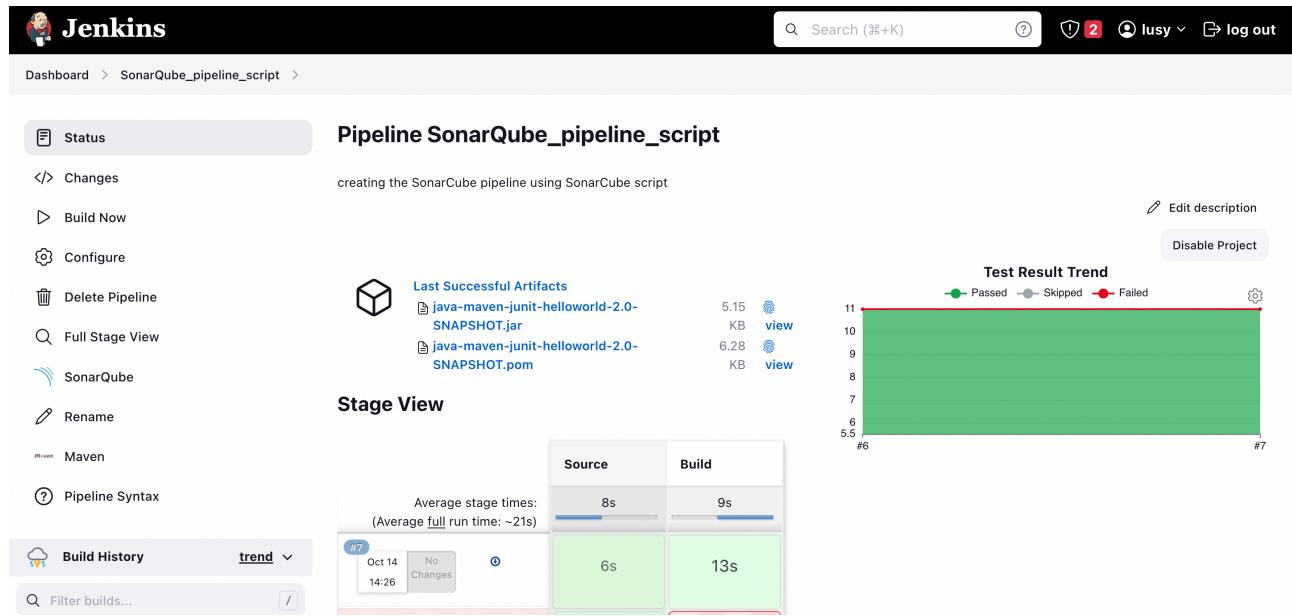
note: I have -X at last line

8. Build the pipeline and verify the build status

The screenshot shows the SonarQube Pipeline build status for build #7, which was successful on October 14, 2023, at 2:26:58 PM. A note in red text says "After all few build failure.. Finally i got it ..😊😊". The build artifacts section shows two files: "java-maven-junit-helloworld-2.0-SNAPSHOT.jar" (5.15 KB) and "java-maven-junit-helloworld-2.0-SNAPSHOT.pom" (6.28 KB). The repository information indicates it was started by user "lusy" from the "https://github.com/marak-s-gradious/maven-test1" repository, specifically at revision 70707b80cef3f91a57347a7ed0f7482b4e3bf126, pointing to the "refs/remotes/origin/main" branch. There were no failures in the test results.

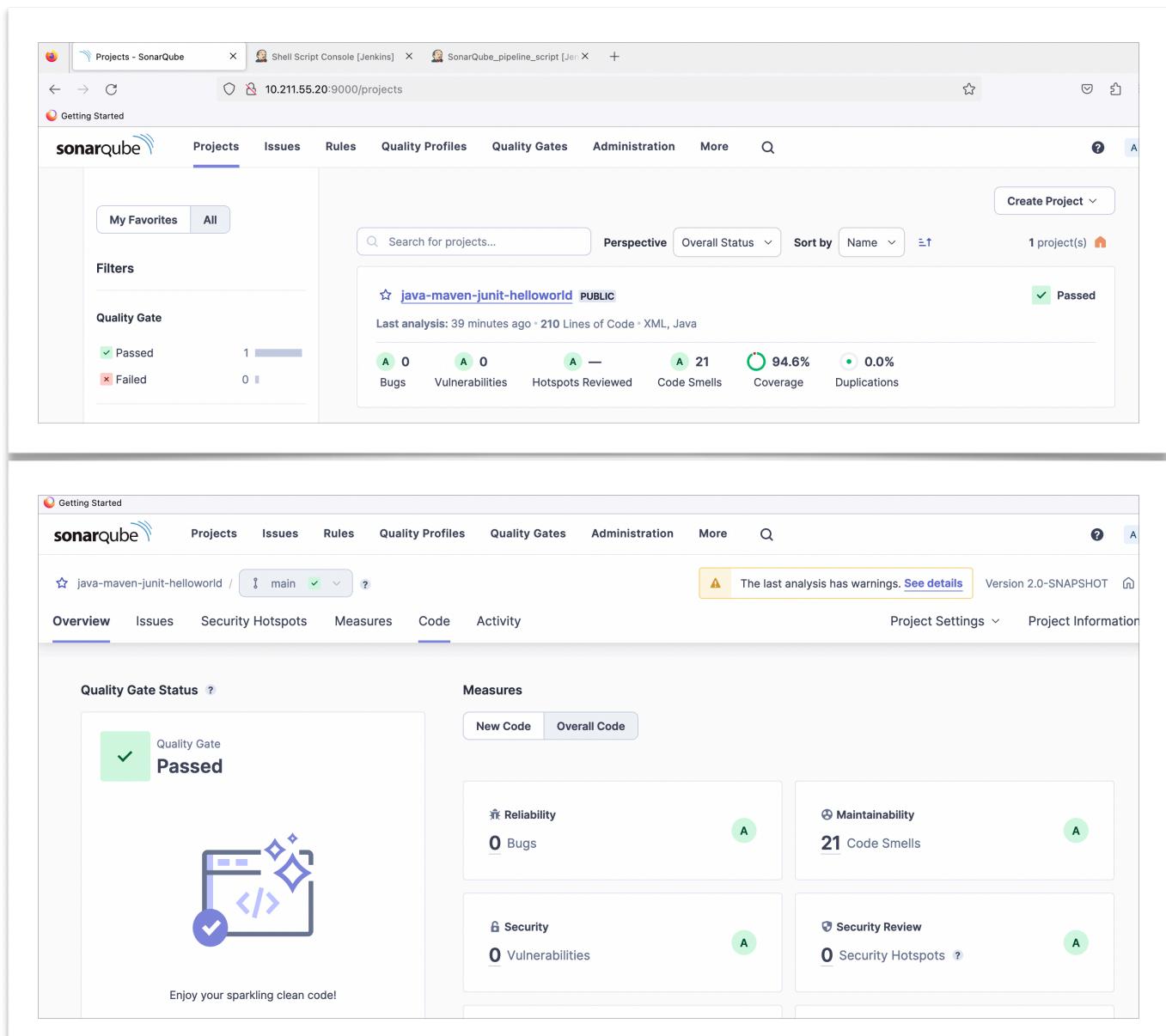
The screenshot shows the Jenkins build status for build #7, which was successful. The "Console Output" tab is selected, showing the log output. The log starts with "[DEBUG]" entries for various Maven dependencies being compiled, such as "org.apache.maven:maven-model-builder:jar:3.0:compile" and "org.apache.maven:maven-aether-provider:jar:3.0:runtime". The log continues with "[DEBUG]" entries for Aether components like "aether-impl", "aether-spi", "aether-api", and "aether-util". It also includes entries for "plexus-interpolation" and "plexus-classworlds" dependencies. The log ends with "[DEBUG]" entries for "plexus-component-annotations". A note above the log says "Skipping 97 KB.. Full Log".

9. Now navigate to sonarqube server to check the code quality results



The screenshot shows the Jenkins Pipeline SonarQube pipeline script dashboard. On the left, there's a sidebar with various project management options like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, SonarQube, Rename, Maven, Pipeline Syntax, and Build History. The main area is titled "Pipeline SonarQube_pipeline_script". It displays "Last Successful Artifacts" (java-maven-junit-helloworld-2.0-SNAPSHOT.jar, java-maven-junit-helloworld-2.0-SNAPSHOT.pom) and a "Test Result Trend" chart showing a green bar from 5.5 to 11. Below that is a "Stage View" table with two rows: Source (8s) and Build (9s). A timeline shows a build step from Oct 14 14:26 with "No Changes", taking 6s. To the right is a "Build History" section for build #7, which took 13s. A "Filter builds..." search bar is at the bottom.

10. You can click your project and see the code test results,



The screenshot shows the SonarQube interface for the "java-maven-junit-helloworld" project. At the top, there are tabs for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. The "Quality Gates" tab is selected, showing a "Passed" status with 1 issue. The main content area displays the project's last analysis (39 minutes ago) with 210 Lines of Code, XML, Java. It shows metrics: 0 Bugs, 0 Vulnerabilities, 1 Hotspots Reviewed, 21 Code Smells, 94.6% Coverage, and 0.0% Duplications. Below this, the "Overview" tab is selected, showing a "Quality Gate Status" of "Passed" with a green checkmark icon. The "Measures" section includes Reliability (0 Bugs), Maintainability (21 Code Smells), Security (0 Vulnerabilities), and Security Review (0 Security Hotspots). A message at the bottom says "Enjoy your sparkling clean code!"

