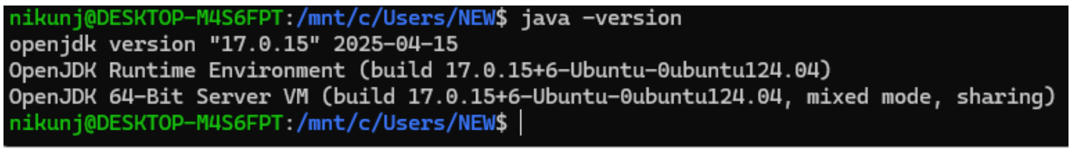
Step:1 Install SonarQube on WSL

Requrements for this is:

Machine must have jdk-17 in wsl

- sudo apt update
- sudo apt install openjdk-17-jdk –y
- java –version



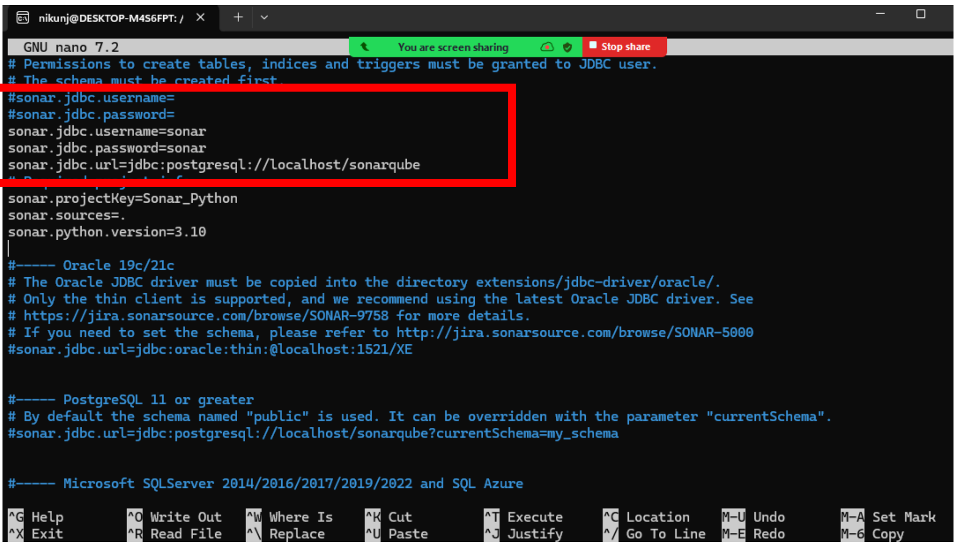
```
nikunj@DESKTOP-M4S6FPT:/mnt/c/Users/NEW$ java -version
openjdk version "17.0.15" 2025-04-15
OpenJDK Runtime Environment (build 17.0.15+6-Ubuntu-0ubuntu124.04)
OpenJDK 64-Bit Server VM (build 17.0.15+6-Ubuntu-0ubuntu124.04, mixed mode, sharing)
nikunj@DESKTOP-M4S6FPT:/mnt/c/Users/NEW$ |
```

Step:2 install PostgreSQl

- sudo apt install postgresql postgresql-contrib –y
  create sonarQube DB and User:
- sudo -u postgres psql
- CREATE USER sonar WITH PASSWORD 'sonar'
- CREATE DATABASE sonarqube OWNER sonar
- \q

Step:3 Install Required dependencies

- sudo apt install unzip wget gnupg2 –y
- cd /opt
- sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.5.1.90531.zip
- sudo unzip sonarqube-10.5.1.90531.zip
- sudo mv sonarqube-10.5.1.90531 sonarqube
- sudo chown –R $USER:$USER sonarqube
- nano /opt/sonarqube/conf/sonar.properties

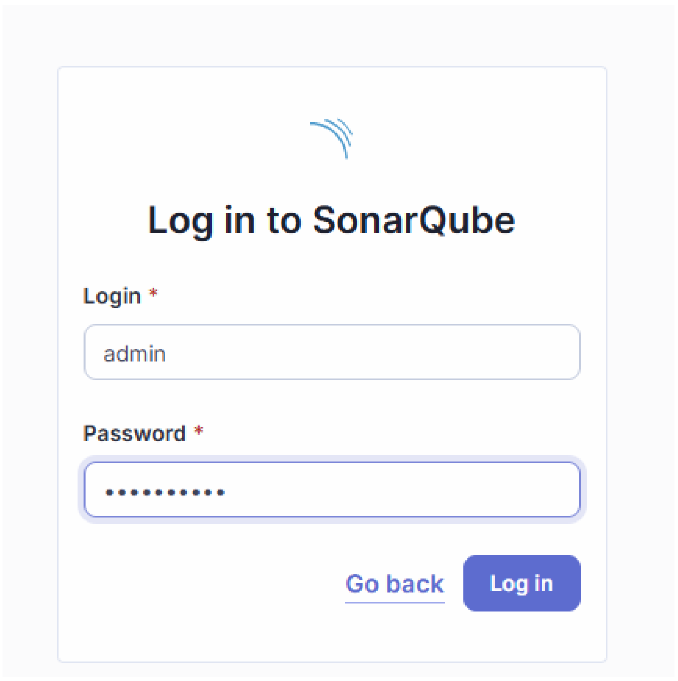


```
  GNU nano 7.2
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
#sonar.jdbc.username=
#sonar.jdbc.password=
sonar.jdbc.username=sonar
sonar.jdbc.password=sonar
sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube
# Required project info
sonar.projectKey=Sonar_Python
sonar.sources=.
sonar.python.version=3.10

#----- Oracle 19c/21c
# The Oracle JDBC driver must be copied into the directory extensions/jdbc-driver/oracle/.
# Only the thin client is supported, and we recommend using the latest Oracle JDBC driver. See
# https://jira.sonarsource.com/browse/SONAR-9758 for more details.
# If you need to set the schema, please refer to http://jira.sonarsource.com/browse/SONAR-5000
#sonar.jdbc.url=jdbc:oracle:thin:@localhost:1521/XE


#----- PostgreSQL 11 or greater
# By default the schema named "public" is used. It can be overridden with the parameter "currentSchema".
#sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube?currentSchema=my_schema


#----- Microsoft SQLServer 2014/2016/2017/2019/2022 and SQL Azure

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

sonar.jdbc.username=sonar
sonar.jdbc.password=sonar
sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube

Step:5 Run the SonarQube

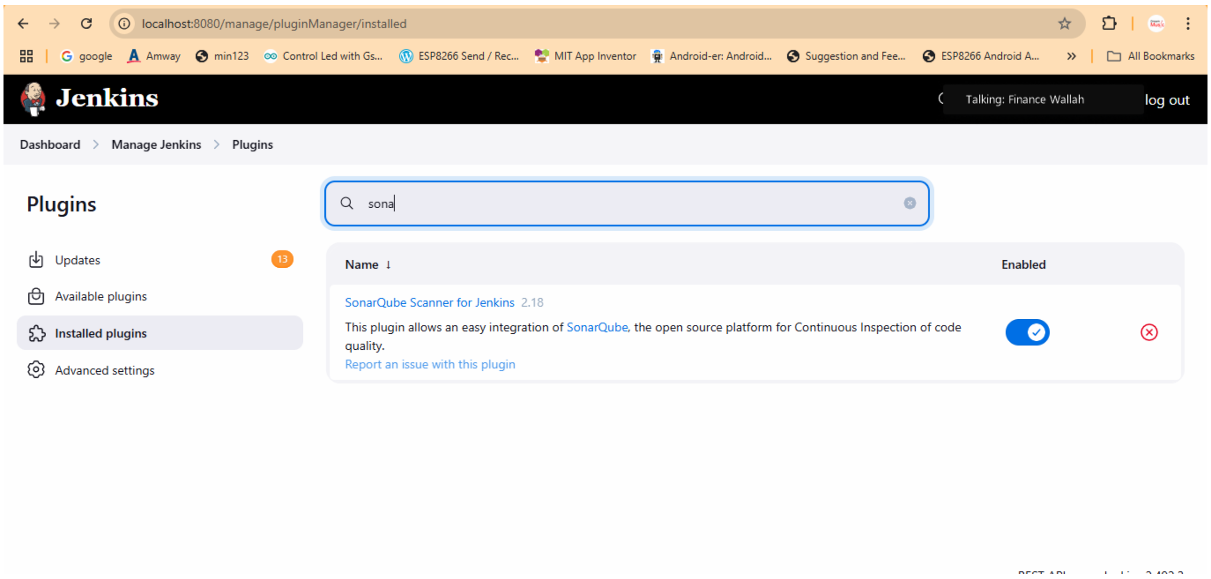
- cd /opt/sonarqube/bin/linux-x86-64
- ./sonar.sh start
  Check the status
- ./sonar.sh status



Login:admin
Password:admin

Starting with Jenkins Part

- Open wsl
- sudo syatemctl start Jenkins
- localhost:8080
- goto>manage Jenkins> plugins
- download SonarQubeScanner


localhost:8080/manage/pluginManager/installed

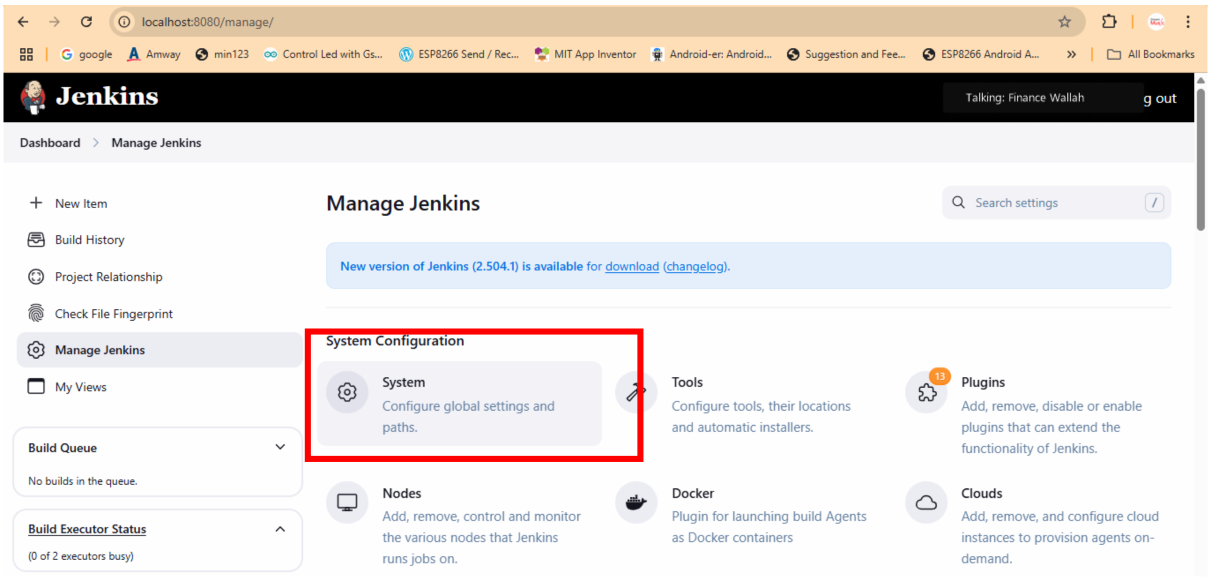
Dashboard > Manage Jenkins > Plugins

Plugins

sona

| Name ↓ | Enabled |
|---|---|
| SonarQube Scanner for Jenkins 2.18 This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin | ✓ |

Goto> manage Jenkins> system configuration


localhost:8080/manage/

Dashboard > Manage Jenkins

Manage Jenkins

New version of Jenkins (2.504.1) is available for download (changelog).

System Configuration

System
Configure global settings and paths.

Tools
Configure tools, their locations and automatic installers.

Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

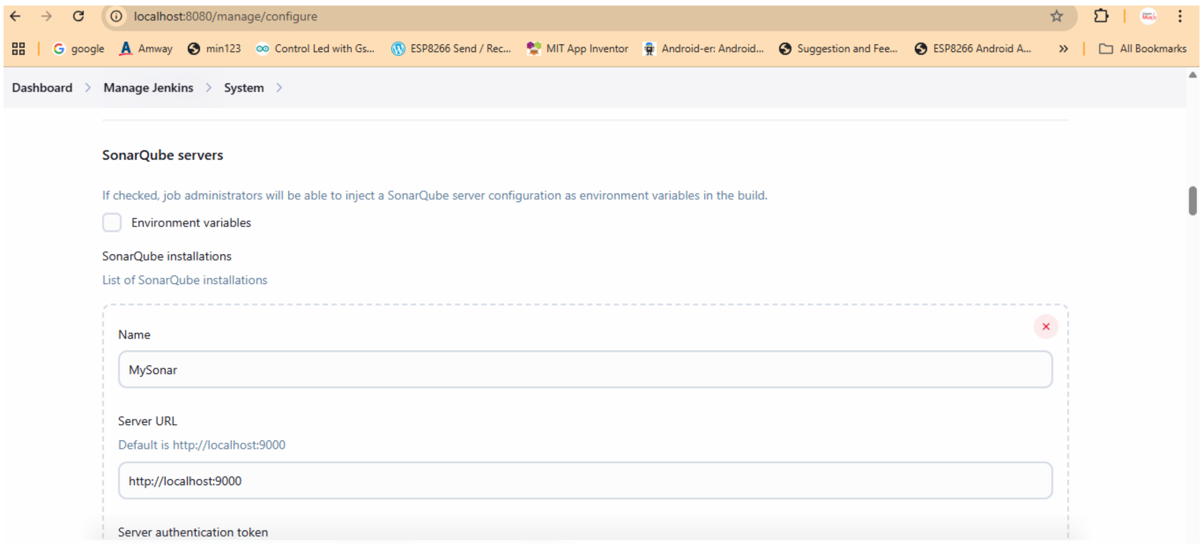
Docker
Plugin for launching build Agents as Docker containers

Clouds
Add, remove, and configure cloud instances to provision agents on-demand.

Build Queue
No builds in the queue.

Build Executor Status
(0 of 2 executors busy)

Goto SonarQube Servers>


localhost:8080/manage/configure

Dashboard > Manage Jenkins > System >

SonarQube servers

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

☐ Environment variables

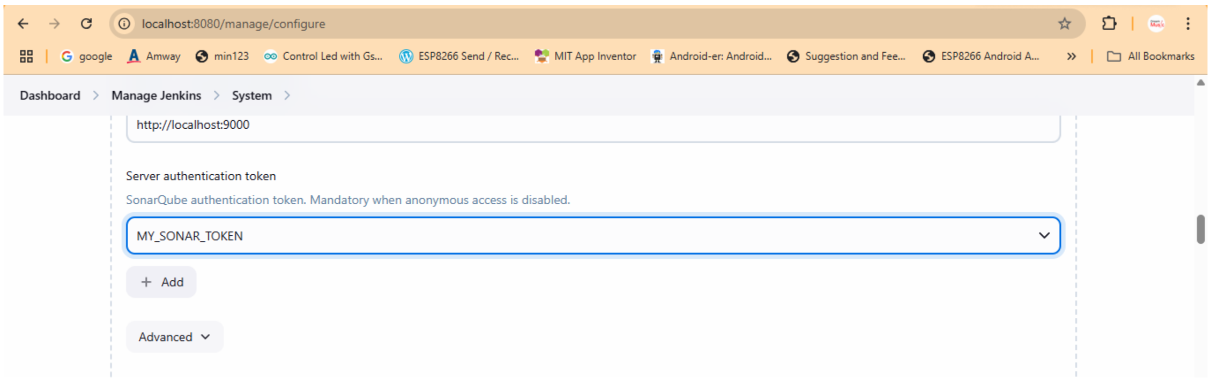
SonarQube installations
List of SonarQube installations

Name
MySonar

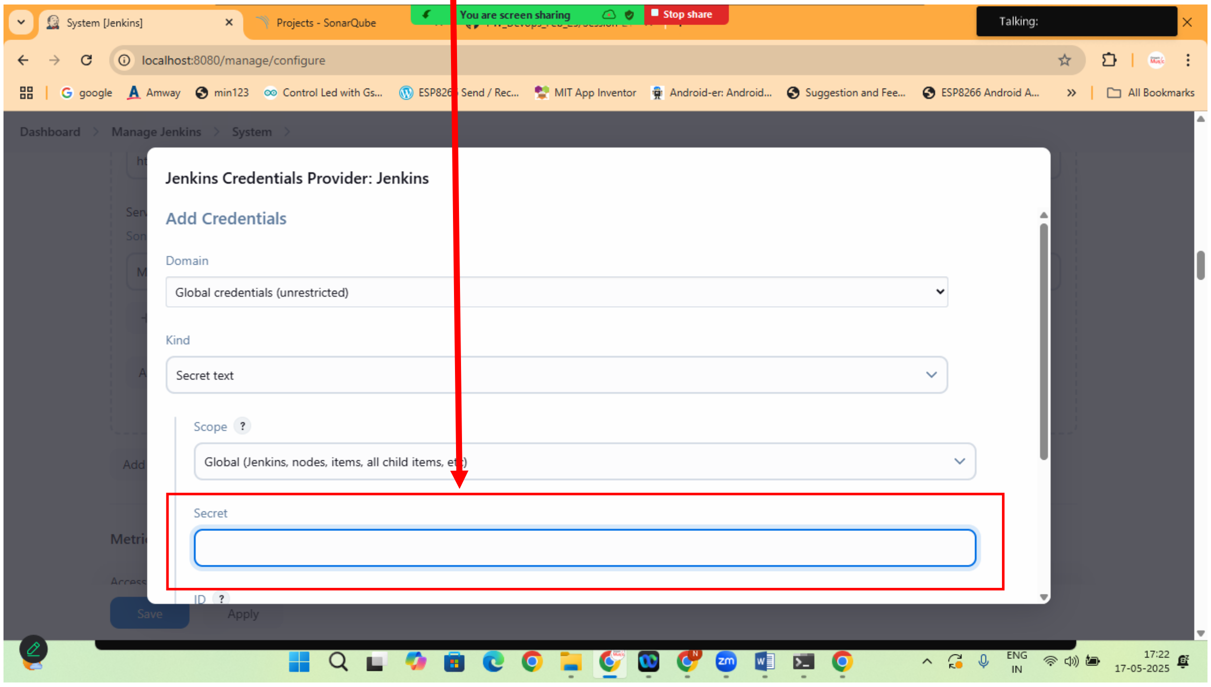
Server URL
Default is http://localhost:9000
http://localhost:9000

Server authentication token

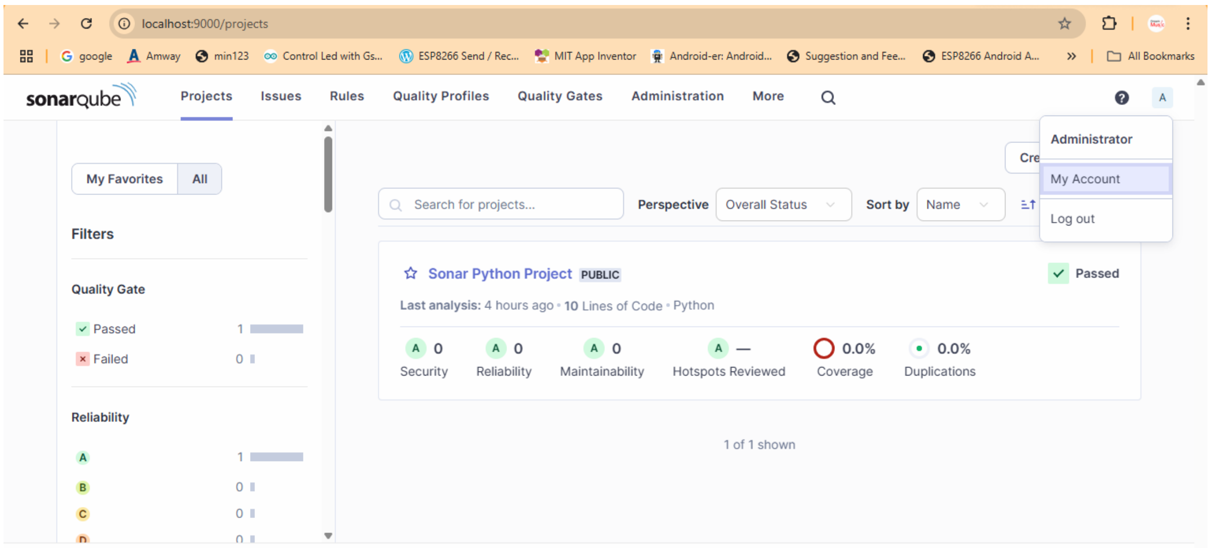
Give the name : MySonar
Server URL:http://localhost:9000
In server Authentication token > click on add


localhost:8080/manage/configure
Dashboard > Manage Jenkins > System
http://localhost:9000
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
MY_SONAR_TOKEN
+ Add
Advanced

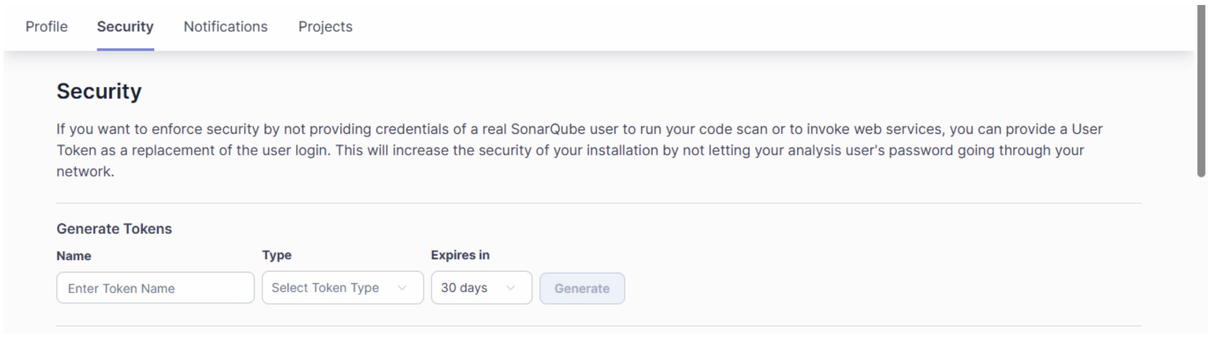
Choose secret text
Add your authentication token in Secret filed:


localhost:8080/manage/configure
Jenkins Credentials Provider: Jenkins
Add Credentials
Domain
Global credentials (unrestricted)
Kind
Secret text
Scope
Global (Jenkins, nodes, items, all child items, etc)
Secret
ID

This token will be generated from sonarqube:> click on MyAccount



Click on security tab and create your token >



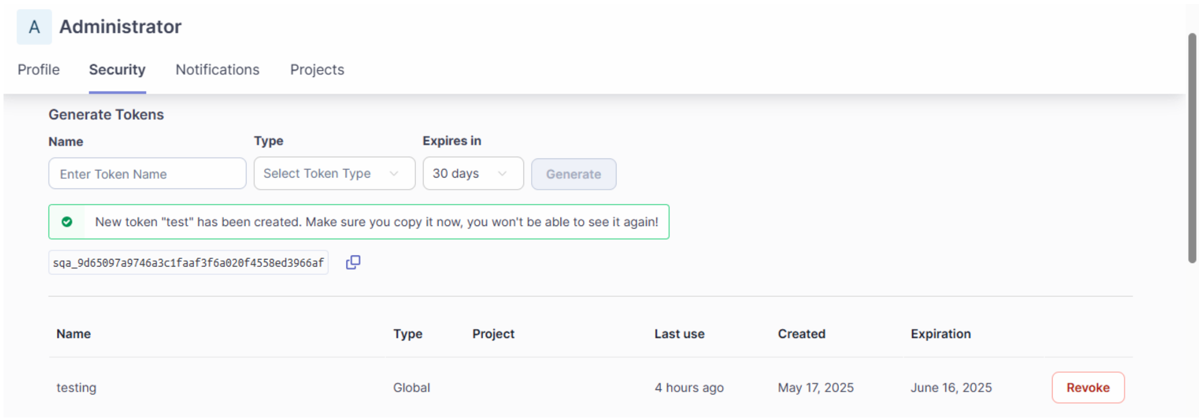
**Security**

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.

**Generate Tokens**

| Name | Type | Expires in | |
|---|---|---|---|
| Enter Token Name | Select Token Type | 30 days | Generate |

Genrate token >



**Generate Tokens**

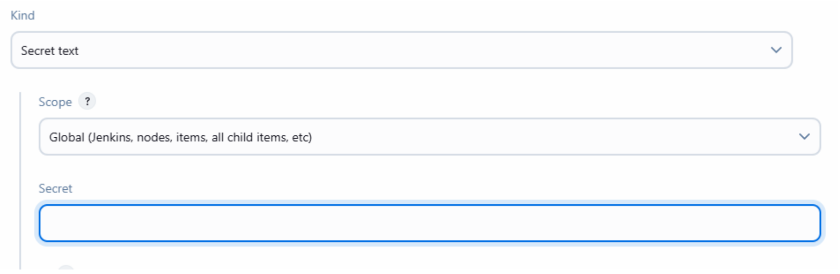
| Name | Type | Expires in | |
|---|---|---|---|
| Enter Token Name | Select Token Type | 30 days | Generate |

New token "test" has been created. Make sure you copy it now, you won't be able to see it again!

```
sqa_9d65097a9746a3c1faaf3f6a020f4558ed3966af
```

| Name | Type | Project | Last use | Created | Expiration | |
|---|---|---|---|---|---|---|
| testing | Global | | 4 hours ago | May 17, 2025 | June 16, 2025 | Revoke |

Copy this token and past it to the Jenkins> secret text

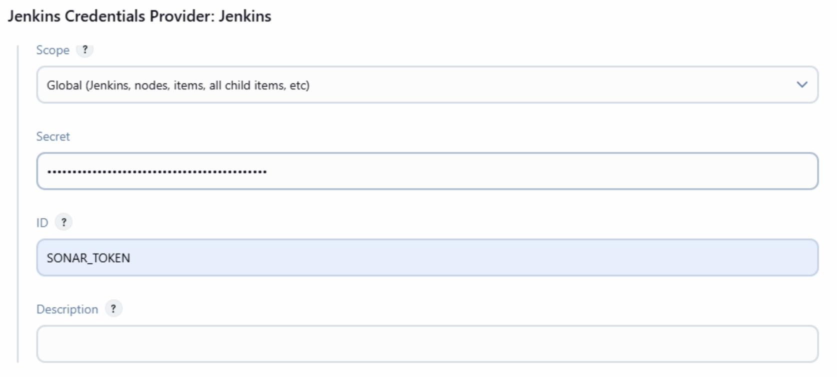


Kind: Secret text

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

Secret:

Give the unique name: MY_SONAR_TOKEN



**Jenkins Credentials Provider: Jenkins**

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

Secret: ••••••••••••••••••••••••••••••••••••••••••••

ID: SONAR_TOKEN

Description:

Click on Save and Apply

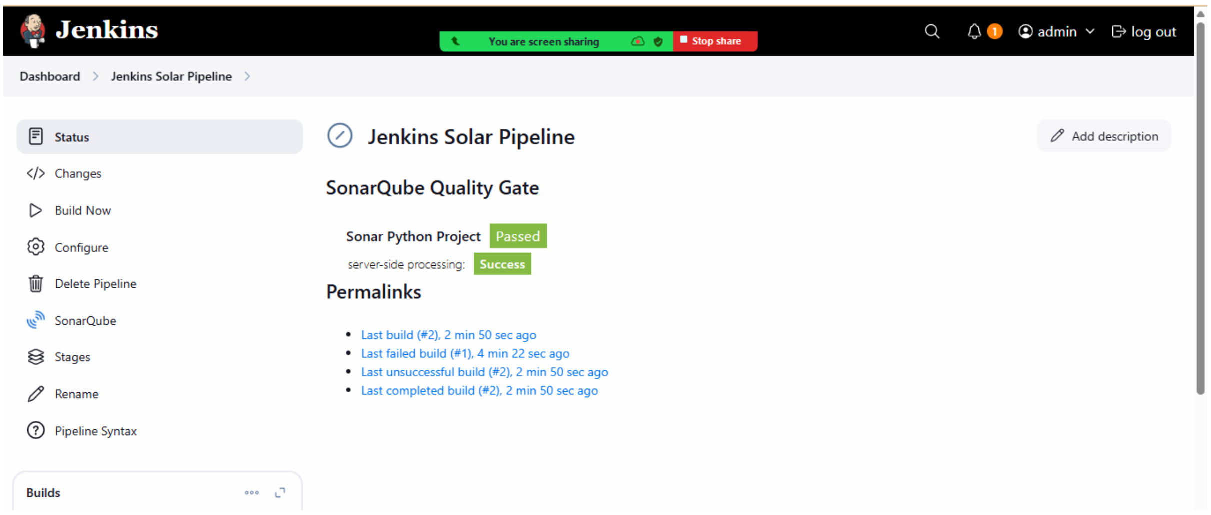
All set with integration

Create new pipeline> add the below pipeline

```
pipeline {
  agent any


  environment {
    SONARQUBE_ENV = 'MySonar'              // Jenkins → Manage Jenkins → Configure System
    SONAR_TOKEN = credentials('MY_SONAR_TOKEN')   // From Jenkins Credentials store


  }


  stages {
    stage('Checkout') {
      steps {
        git url: 'https://github.com/Nikunj-Java/Sonar_Python.git', branch: 'main'
      }
    }


    stage('SonarQube Analysis') {
      steps {
        withSonarQubeEnv("${SONARQUBE_ENV}") {
          sh '''
            /opt/sonar-scanner/bin/sonar-scanner \
            -Dsonar.projectKey=Sonar_Python \
            -Dsonar.sources=app \
            -Dsonat.tests=tests \
            -Dsonar.python.version=3.10 \
            -Dsonar.login=${MY_SONAR_TOKEN}
          '''
        }
      }
    }
```

```
        stage('Quality Gate') {
            steps {
                timeout(time: 2, unit: 'MINUTES') {
                    waitForQualityGate abortPipeline: true
                }
            }
        }
    }
}
```

Build and configure



Jenkins Solar Pipeline

SonarQube Quality Gate

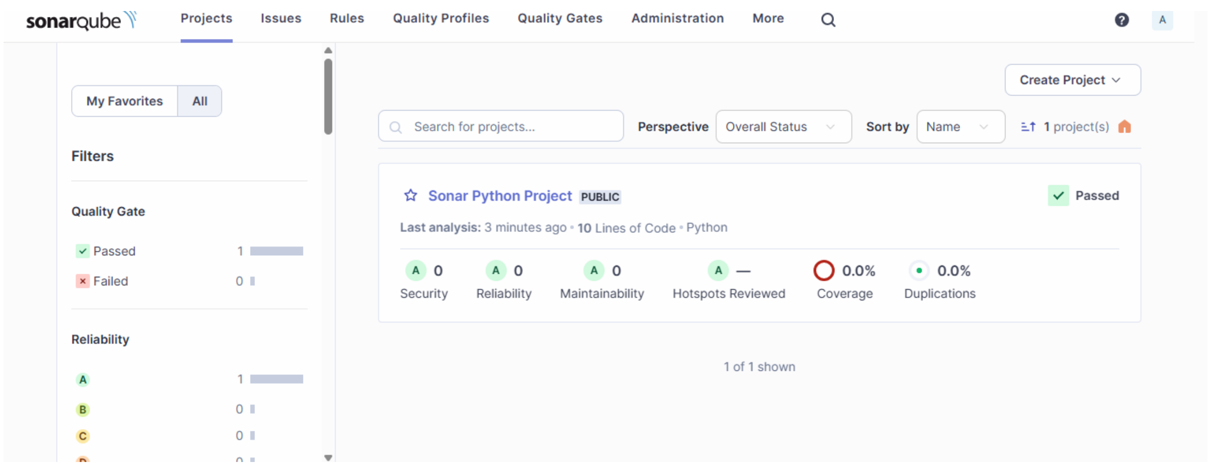
Sonar Python Project Passed

server-side processing: Success

Permalinks

- Last build (#2), 2 min 50 sec ago
- Last failed build (#1), 4 min 22 sec ago
- Last unsuccessful build (#2), 2 min 50 sec ago
- Last completed build (#2), 2 min 50 sec ago

Check the output in sonar cube as well



1 project(s)

Quality Gate

| | |
|---|---|
| ✓ Passed | 1 |
| ✗ Failed | 0 |

Reliability

| | |
|---|---|
| A | 1 |
| B | 0 |
| C | 0 |

Sonar Python Project PUBLIC — ✓ Passed

Last analysis: 3 minutes ago • 10 Lines of Code • Python

| Security | Reliability | Maintainability | Hotspots Reviewed | Coverage | Duplications |
|---|---|---|---|---|---|
| A 0 | A 0 | A 0 | A — | 0.0% | 0.0% |

1 of 1 shown