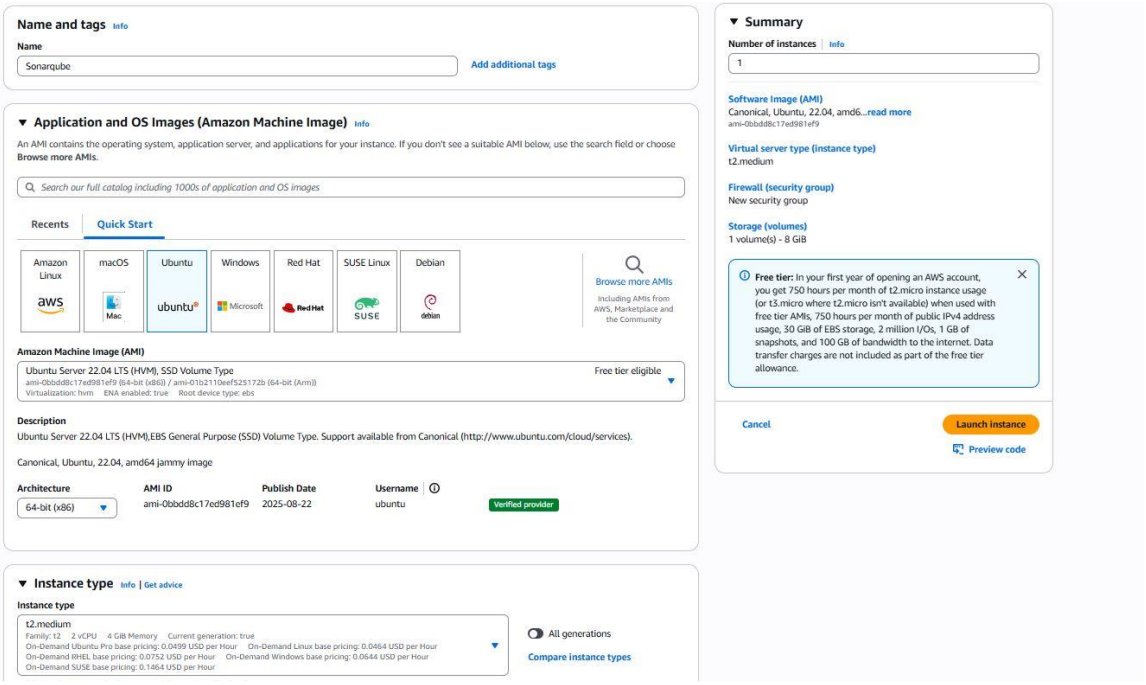
Test Java Code using SonarQube on Linux

This document provides a complete step-by-step guide to test Java code using SonarQube on a Linux server without installing SonarScanner or PostgreSQL. It utilizes Maven's built-in SonarQube plugin and SonarQube's embedded H2 database.

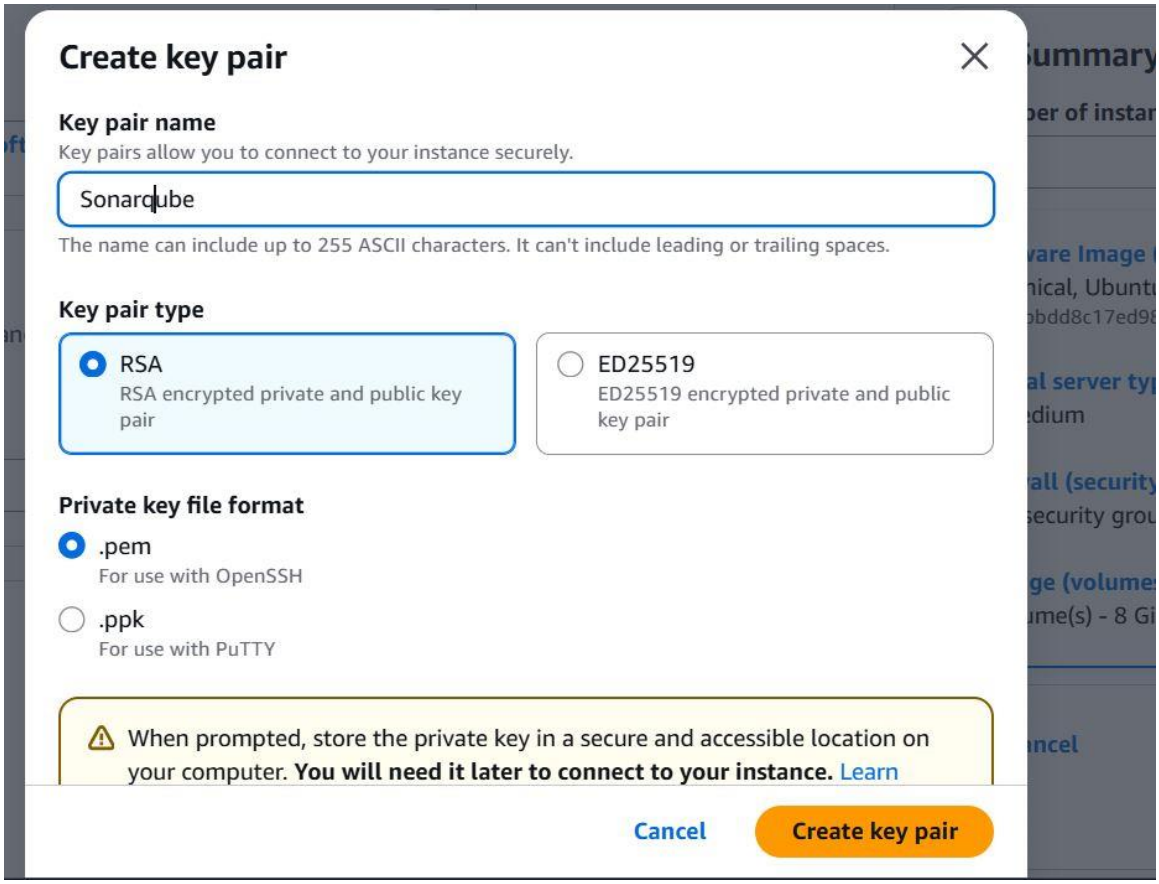
1. Pre-requisites

→ A linux server with 2 CPU's and 4gb RAM
→ Java (JDK 17 or later)
→ Maven (3.6
→ unzip, wget

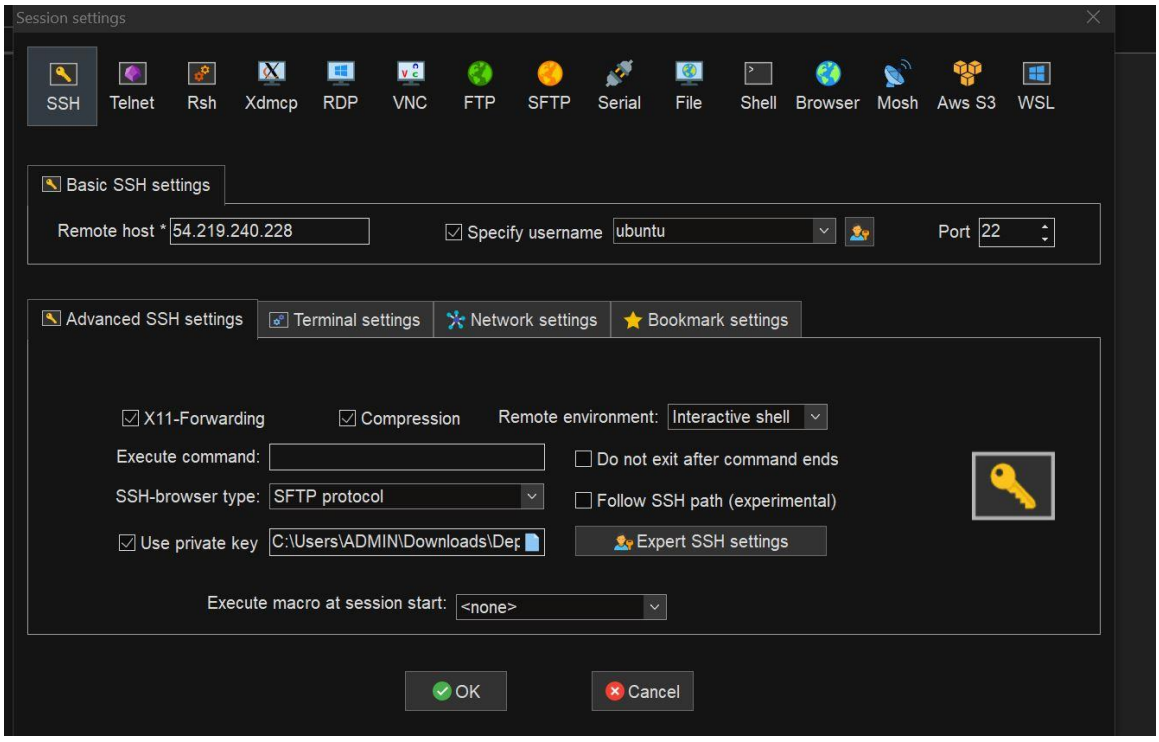
Launch an instance with 2 CPU's and 4gb RAM



Create a key pair with a suitable name



After launching the instance, connect to it with the public IP address of the server



After connecting to the server update the server to the latest package installations

```
ubuntu@ip-172-31-30-110:~$ sudo apt update -y
```

Now, install java version

```
ubuntu@ip-172-31-30-110:~$ java --version
Command 'java' not found, but can be installed with:
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~22.04.1, or
sudo apt install default-jre              # version 2:1.11-72build2
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~22.04.1
sudo apt install openjdk-18-jre-headless  # version 18.0.2+9-2~22.04
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-0ubuntu3~22.04
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~22.04.1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~22.04.2
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~22.04.2
ubuntu@ip-172-31-30-110:~$ sudo apt install openjdk-17-jre-headless
```

After installing java, we need to install maven

```
ubuntu@ip-172-31-30-110:~$ sudo apt install maven -y
```

By this step, we have installed all the necessary pre-requisites on the server.

Now, clone the code from github where your Java code is present

```
ubuntu@ip-172-31-30-110:~$ git clone https://github.com/akracad/JavaWebCal.git
Cloning into 'JavaWebCal'...
remote: Enumerating objects: 29, done.
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 29 (delta 3), reused 29 (delta 3), pack-reused 0 (from 0)
Receiving objects: 100% (29/29), 5.78 KiB | 1.93 MiB/s, done.
Resolving deltas: 100% (3/3), done.
ubuntu@ip-172-31-30-110:~$
```

Now install sonarqube on the server

```
ubuntu@ip-172-31-27-244:~$ wget wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.6.0.92116.zip
--2025-10-21 11:24:07--  http://wget/
Resolving wget (wget)... failed: Temporary failure in name resolution.
wget: unable to resolve host address 'wget'
--2025-10-21 11:24:07--  https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.6.0.92116.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 99.84.188.45, 99.84.188.35, 99.84.188.106, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|99.84.188.45|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 731034761 (697M) [binary/octet-stream]
Saving to: 'sonarqube-10.6.0.92116.zip'

sonarqube-10.6.0.92116.zip      100%[=================================================>] 697.17M   401MB/s    in 1.7s

2025-10-21 11:24:09 (401 MB/s) - 'sonarqube-10.6.0.92116.zip' saved [731034761/731034761]

FINISHED --2025-10-21 11:24:09--
Total wall clock time: 1.8s
Downloaded: 1 files, 697M in 1.7s (401 MB/s)
ubuntu@ip-172-31-27-244:~$ ls
sonarqube-10.6.0.92116.zip
ubuntu@ip-172-31-27-244:~$
```

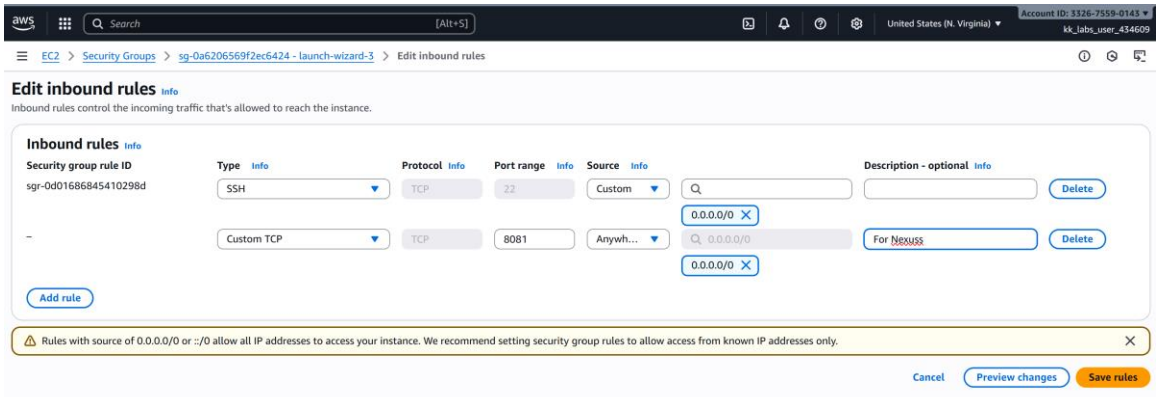
Now untar the sonarqube zip file

```
ubuntu@ip-172-31-27-244:~$ unzip sonarqube-10.6.0.92116.zip
Archive:  sonarqube-10.6.0.92116.zip
   creating: sonarqube-10.6.0.92116/
   creating: sonarqube-10.6.0.92116/jres/
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_x64_linux_hotspot_17.0.11_9.tar.gz
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_aarch64_linux_hotspot_17.0.11_9.tar.gz
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_x64_alpine-linux_hotspot_17.0.11_9.tar.gz
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_x64_windows_hotspot_17.0.11_9.zip
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_x64_mac_hotspot_17.0.11_9.tar.gz
  inflating: sonarqube-10.6.0.92116/jres/OpenJDK17U-jre_aarch64_mac_hotspot_17.0.11_9.tar.gz
  inflating: sonarqube-10.6.0.92116/dependency-license.json
  inflating: sonarqube-10.6.0.92116/COPYING
   creating: sonarqube-10.6.0.92116/bin/
   creating: sonarqube-10.6.0.92116/bin/windows-x86-64/
  inflating: sonarqube-10.6.0.92116/bin/windows-x86-64/SonarService.bat
   creating: sonarqube-10.6.0.92116/bin/windows-x86-64/lib/
  inflating: sonarqube-10.6.0.92116/bin/windows-x86-64/lib/SonarServiceWrapper.exe
  inflating: sonarqube-10.6.0.92116/bin/windows-x86-64/lib/find_java.bat
   creating: sonarqube-10.6.0.92116/bin/winsw-license/
  inflating: sonarqube-10.6.0.92116/bin/winsw-license/LICENSE.txt
   creating: sonarqube-10.6.0.92116/data/
  inflating: sonarqube-10.6.0.92116/data/README.txt
   creating: sonarqube-10.6.0.92116/extensions/
   creating: sonarqube-10.6.0.92116/extensions/jdbc-driver/
```

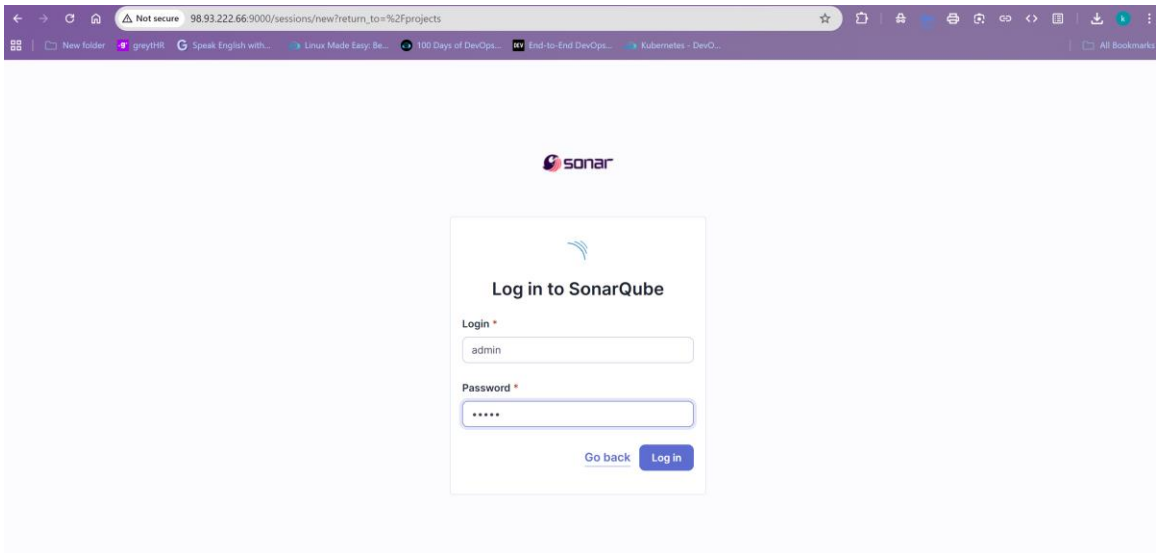
Now start the sonarqube server by navigating into Sonar/bin/linux

```
ubuntu@ip-172-31-27-244:~/sonar$ ls
COPYING  bin  conf  data  dependency-license.json  elasticsearch  extensions  jres  lib  logs  temp  web
ubuntu@ip-172-31-27-244:~/sonar$ cd bin/
ubuntu@ip-172-31-27-244:~/sonar/bin$ ls
elasticsearch  linux-x86-64  macosx-universal-64  windows-x86-64  winsw-license
ubuntu@ip-172-31-27-244:~/sonar/bin$ cd linux-x86-64/
ubuntu@ip-172-31-27-244:~/sonar/bin/linux-x86-64$ ls
sonar.sh
ubuntu@ip-172-31-27-244:~/sonar/bin/linux-x86-64$ ./sonar.sh
/usr/bin/java
Usage: ./sonar.sh { console | start | stop | force-stop | restart | status | dump }
ubuntu@ip-172-31-27-244:~/sonar/bin/linux-x86-64$ ./sonar.sh start
/usr/bin/java
Starting SonarQube...
Started SonarQube.
```

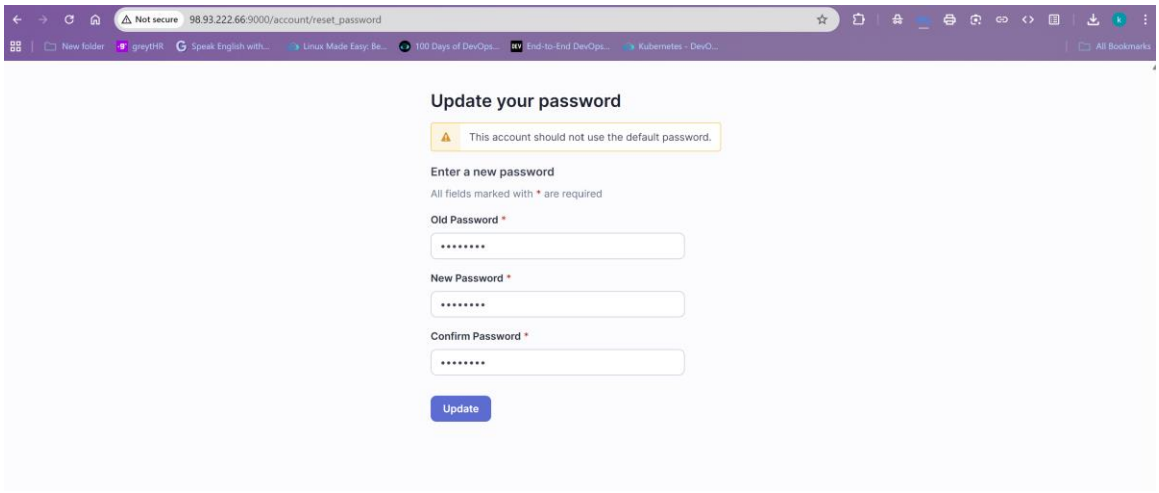
Now allow the port number int the inbound rules of security group of ec2-instance



After starting sonarqube, access it on the browser with the public ip and port number.

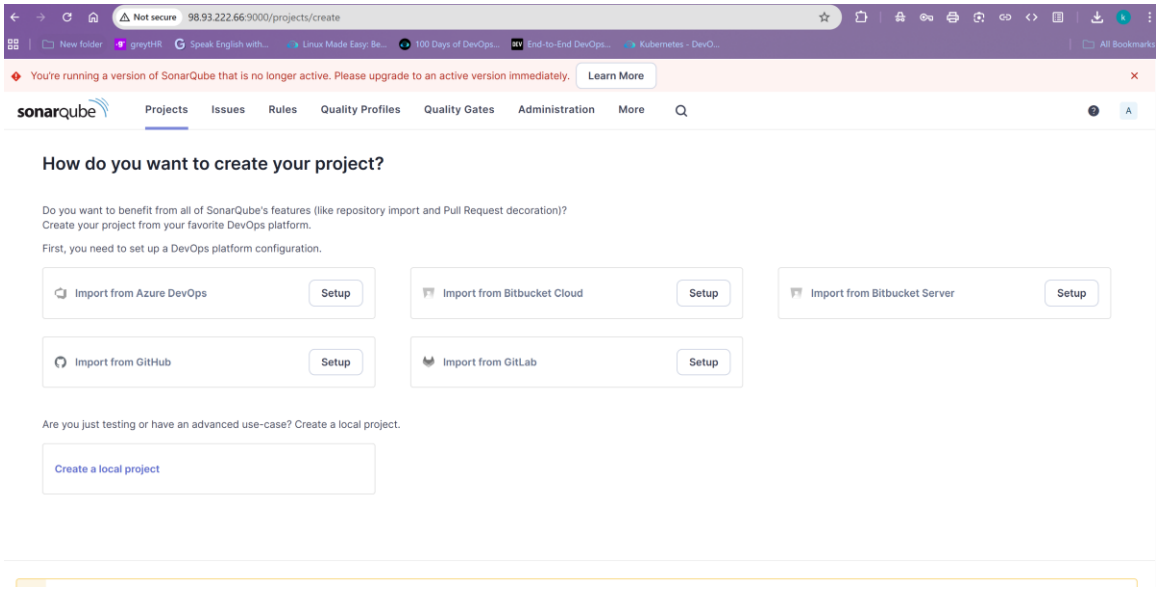


Update your password, once you login into sonarqube

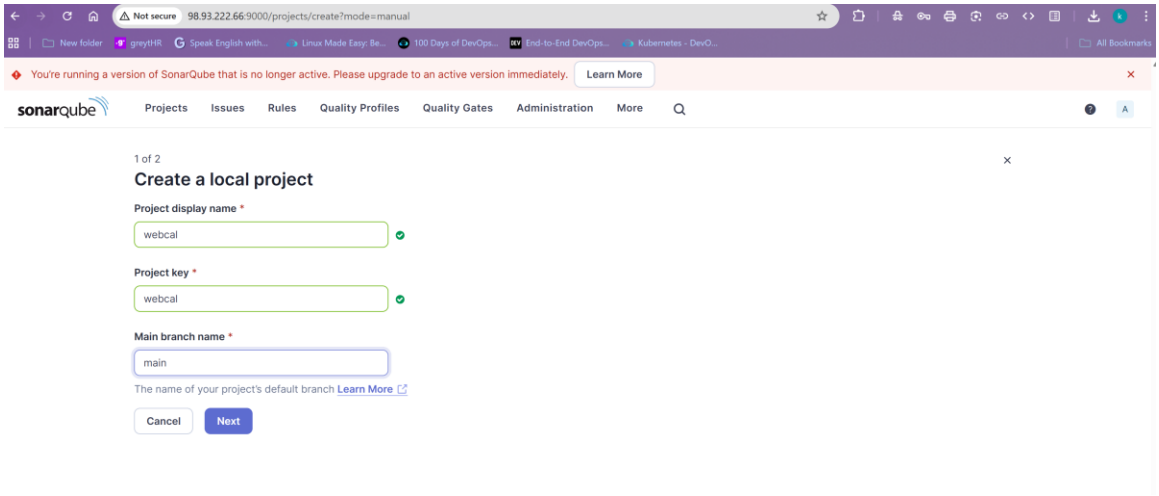


Create a Project in SonarQube UI

1. Log in to SonarQube.
2. Go to Projects → Create Project → Manually.

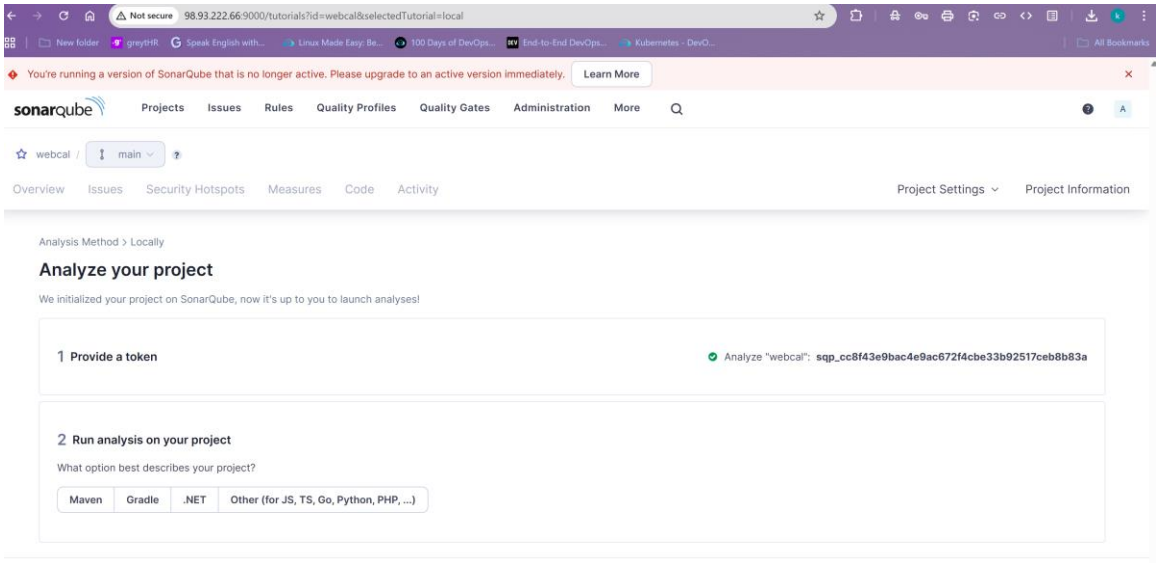


3. Enter Project Key: webcal.
4. Enter Project Name: webcal.



5. Select “Use Maven” method.

6. Generate an authentication token and save it.



Navigate to Java Project directory

Assume your project directory is /home/user/my-java-app/ and contains a pom.xml file.

Run Analysis using Maven (No SonarScanner)

cd /home/user/my-java-app/
mvn clean verify sonar:sonar \
 -Dsonar.projectKey=my-java-app \
 -Dsonar.host.url=http://<server-ip>:9000 \
 -Dsonar.login=<your-generated-token>

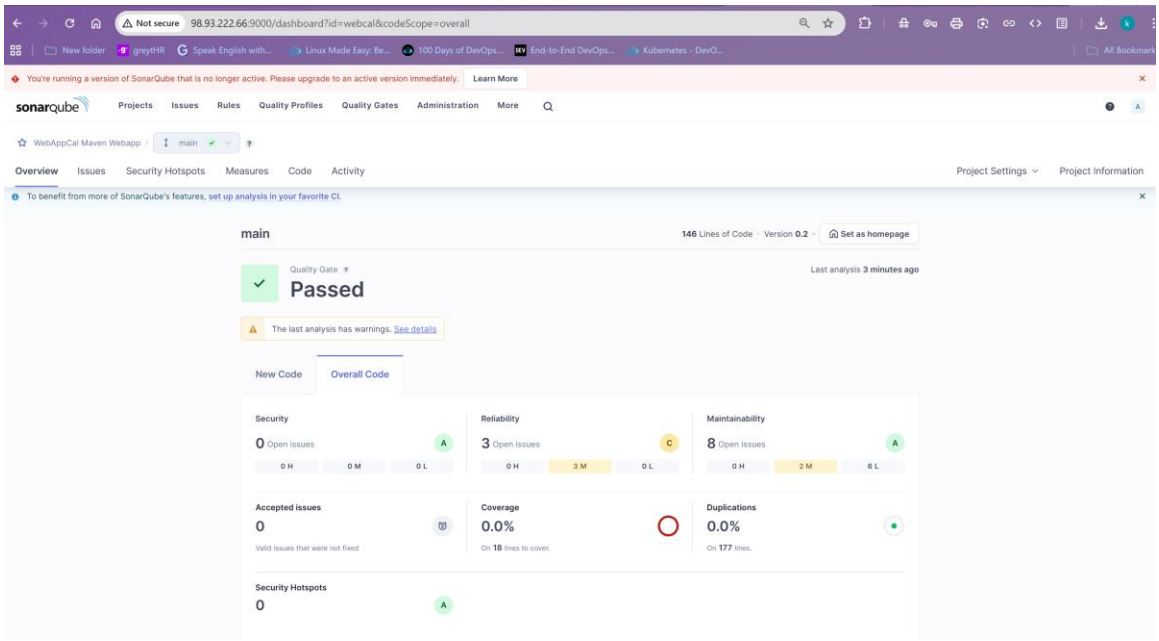
```
ubuntu@ip-172-31-26-216:~/JavaWebCal$ mvn clean verify sonar:sonar \
  -Dsonar.projectKey=webcal \
  -Dsonar.host.url=http://http://98.93.222.66:9000 \
  -Dsonar.login=sqp_cc8f43e9bac4e9ac672f4cbe33b92517ceb8b83a
[INFO] Scanning for projects...
[WARNING] The artifact org.codehaus.mojo:sonar-maven-plugin:jar:4.0.0.4121 has been relocated to org.sonarsource.scanner.mave
n:sonar-maven-plugin:jar:4.0.0.4121: SonarQube plugin was moved to SonarSource organisation
[INFO]
[INFO] ------------------------< com.web.cal:webapp >------------------------
[INFO] Building WebAppCal Maven Webapp 0.2
[INFO] --------------------------------[ war ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ webapp ---
[INFO] Deleting /home/ubuntu/JavaWebCal/target
```

This command builds, tests, and analyzes your Java code automatically.

```
[INFO] 11:52:56.223 SCM Publisher 5 source files to be analyzed
[INFO] 11:52:56.424 SCM Publisher 4/5 source files have been analyzed (done) | time=200ms
[WARNING] 11:52:56.424 Missing blame information for the following files:
[WARNING] 11:52:56.425   * pom.xml
[WARNING] 11:52:56.425 This may lead to missing/broken features in SonarQube
[INFO] 11:52:56.429 CPD Executor Calculating CPD for 2 files
[INFO] 11:52:56.437 CPD Executor CPD calculation finished (done) | time=7ms
[INFO] 11:52:56.445 SCM revision ID '8d9c87629e894e160ee506685ad896e084661847'
[INFO] 11:52:56.583 Analysis report generated in 128ms, dir size=212.9 kB
[INFO] 11:52:56.629 Analysis report compressed in 44ms, zip size=29.9 kB
[INFO] 11:52:56.660 Analysis report uploaded in 29ms
[INFO] 11:52:56.662 ANALYSIS SUCCESSFUL, you can find the results at: http://98.93.222.66:9000/dashboard?id=webcal
[INFO] 11:52:56.663 Note that you will be able to access the updated dashboard once the server has processed the submitted an
alysis report
[INFO] 11:52:56.664 More about the report processing at http://98.93.222.66:9000/api/ce/task?id=25db8891-014a-4e29-a837-ce04a
94c67a8
[INFO] 11:52:56.689 Analysis total time: 14.118 s
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  19.716 s
[INFO] Finished at: 2025-10-21T11:52:56Z
```

8. View Analysis Results

After the analysis completes, open http://<server-ip>:9000/projects to view:



- Bugs
- Code Smells
- Vulnerabilities
- Coverage (if tests exist)

✅You have successfully tested Java code using SonarQube without PostgreSQL or SonarScanner.