To integrate our SonarQube server with jenkins we need provide the authorization token while configuring the SonarQube server settings in jenkins.

We need to install the SonarQube scanner plugin in the jenkins.

**How to generate authorization token?**

Open SonarQube server and login with your admin credentials.

Graphical user interface, website

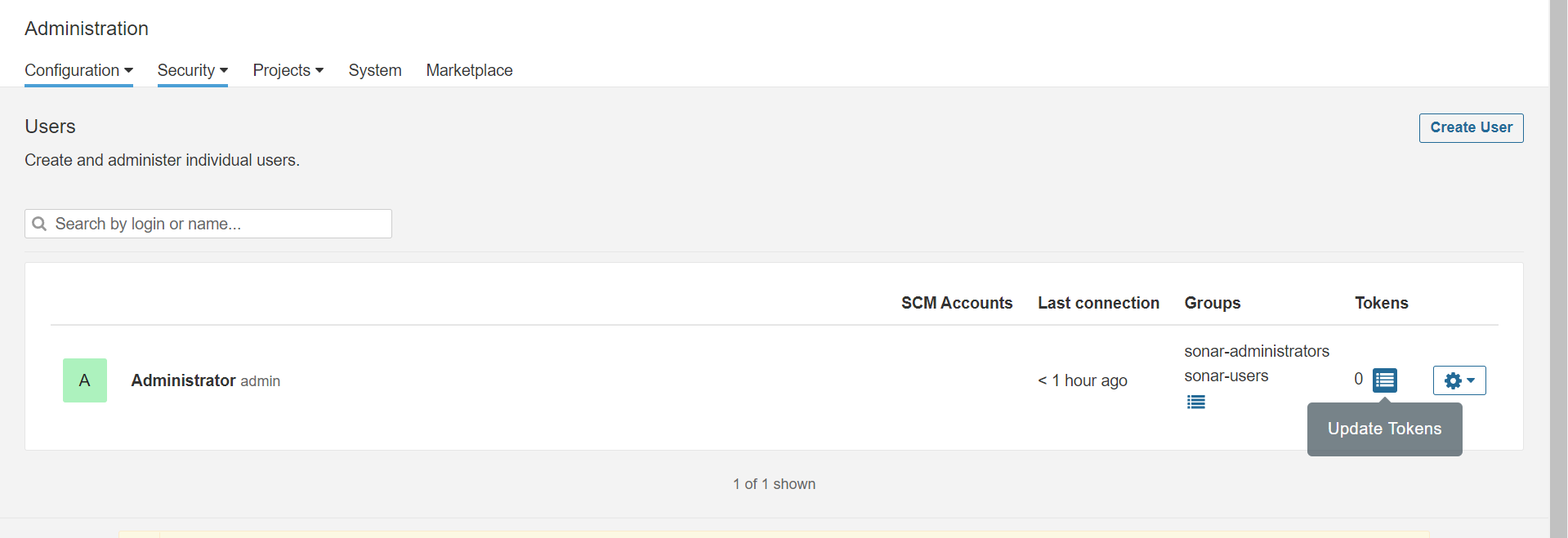
Description automatically generated

Go to “Administration” section

Graphical user interface, application

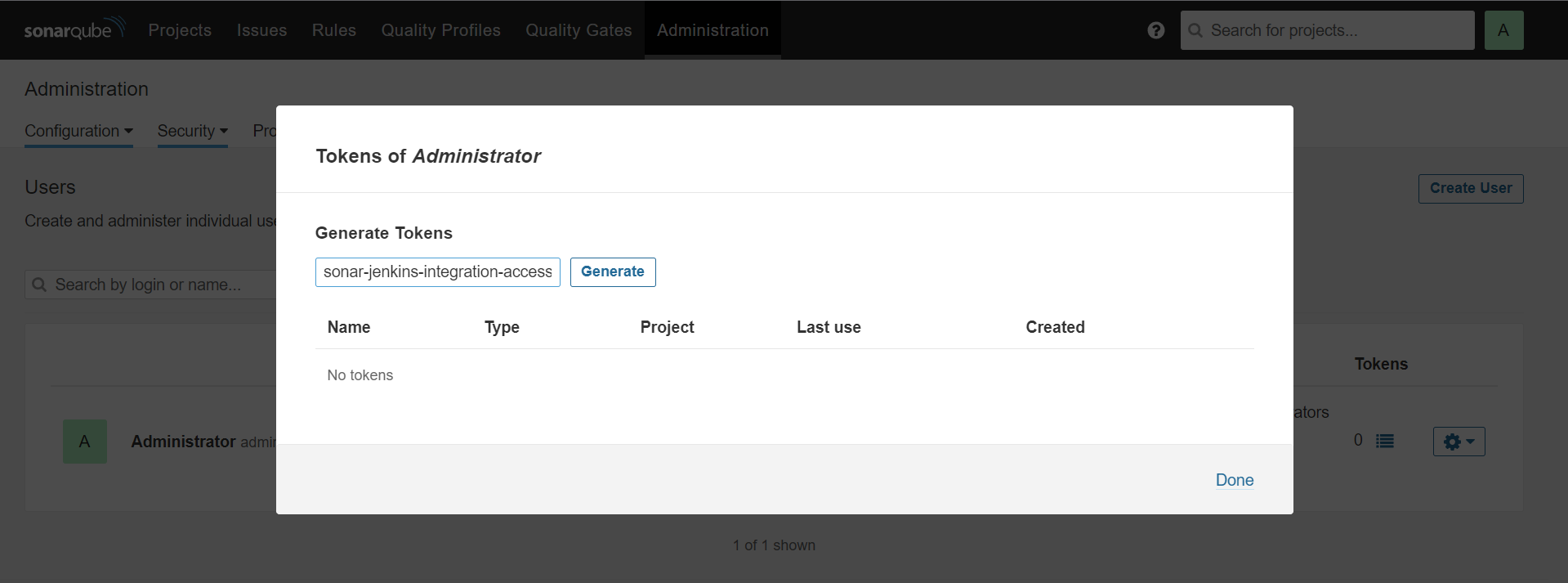
Description automatically generated

Go to “security” and select users.

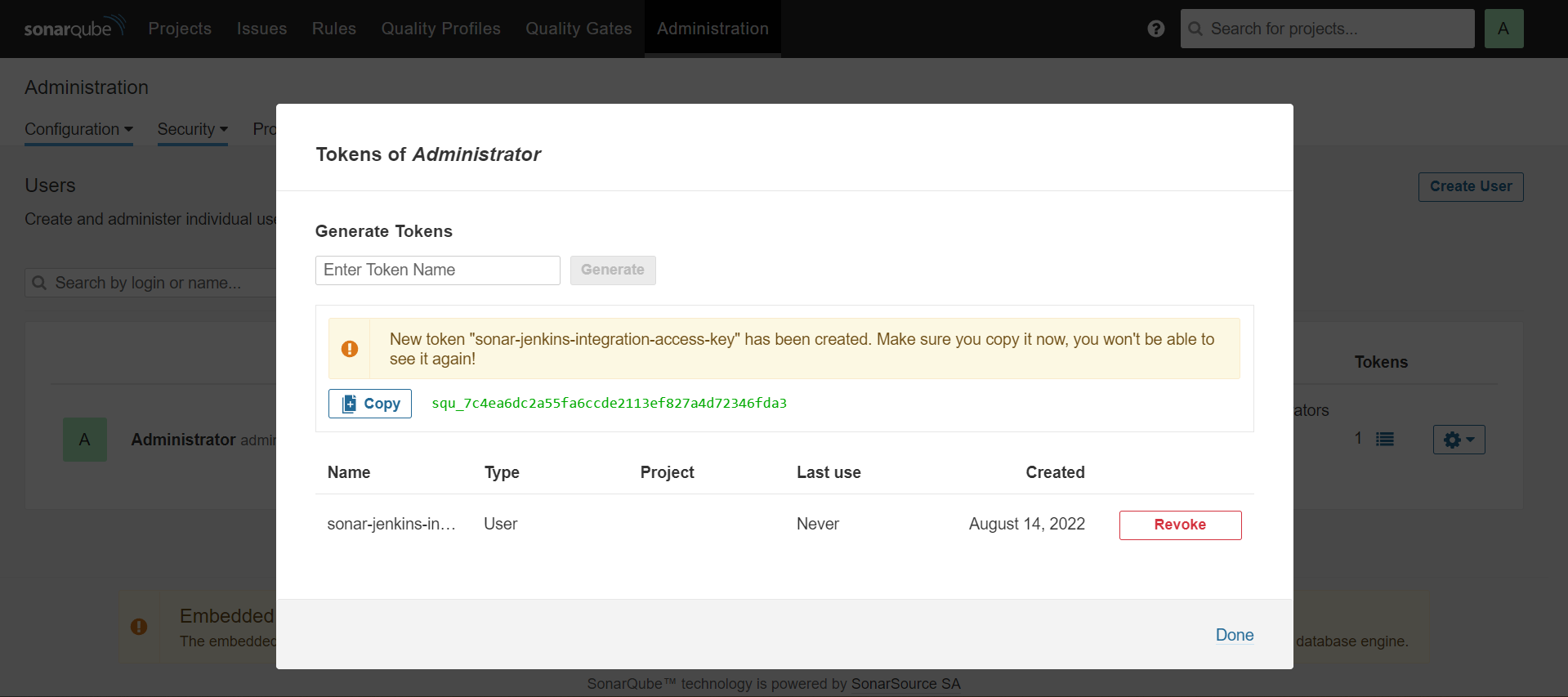


Go to tokens section on select the update tokens box.

One pop-up window will be open.



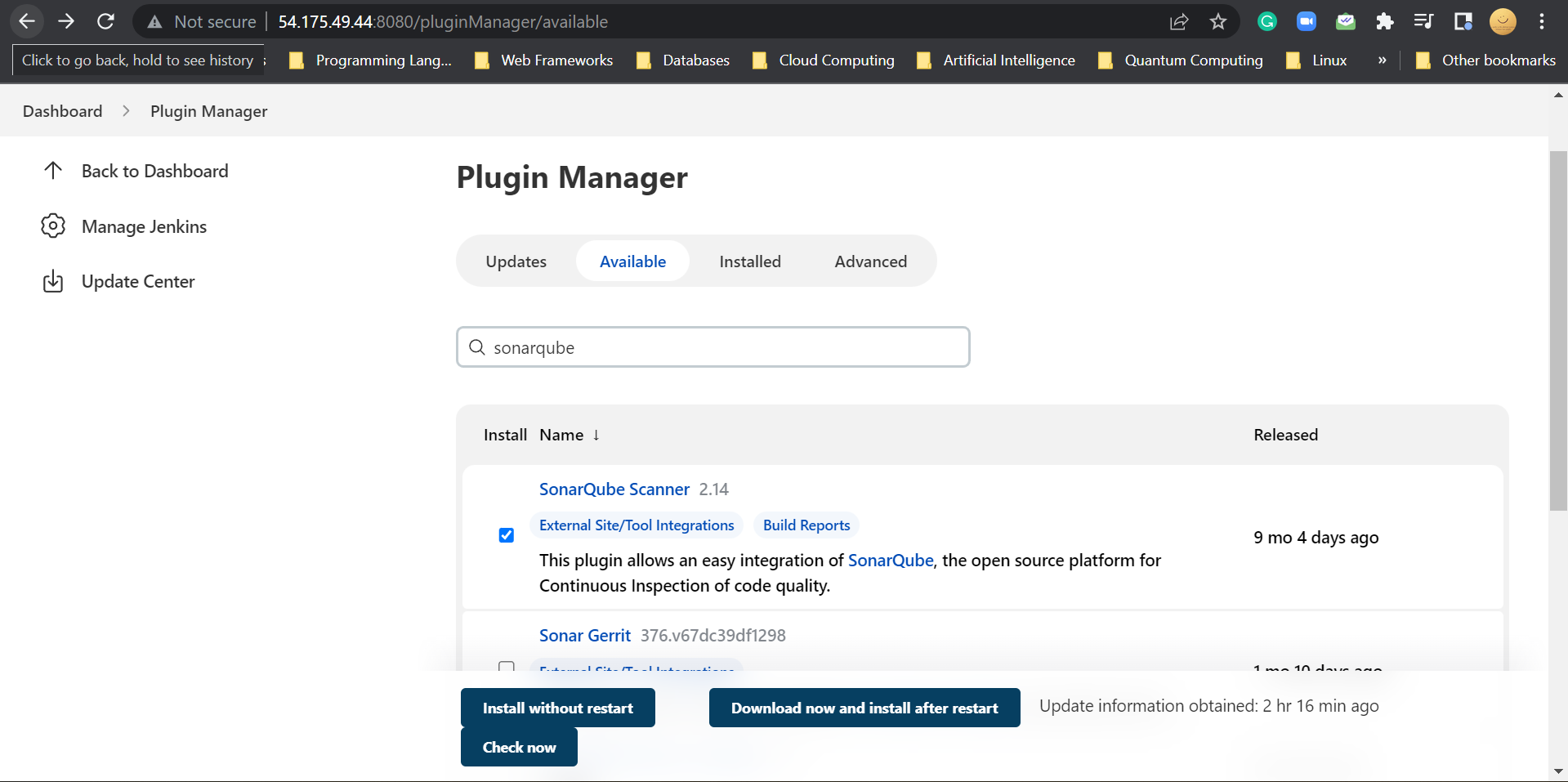
Enter the name for the token and click on “Generate”.



New authorization token generated successfully, let’s copy this token and save somewhere.

**How install SonarQube plugin in jenkins?**

Go to “Manage jenkins” 🡺 Manage plugins 🡺 Available 🡺 search for “SonarQube”



Install the plugin without restart.

Graphical user interface, text, application

Description automatically generated

Once installed successfully, restart the jenkins server once.

Select that check box to restart the server.

Graphical user interface, text, website

Description automatically generated

Jenkins is restarting.

Graphical user interface, application

Description automatically generated

Jenkins started again, login to the jenkins.

Now, add the SonarQube server authorization token in jenkins “Manage credentials” section.

Go to “Manage jenkins” 🡺 “Manage credentials”

Graphical user interface, text, application

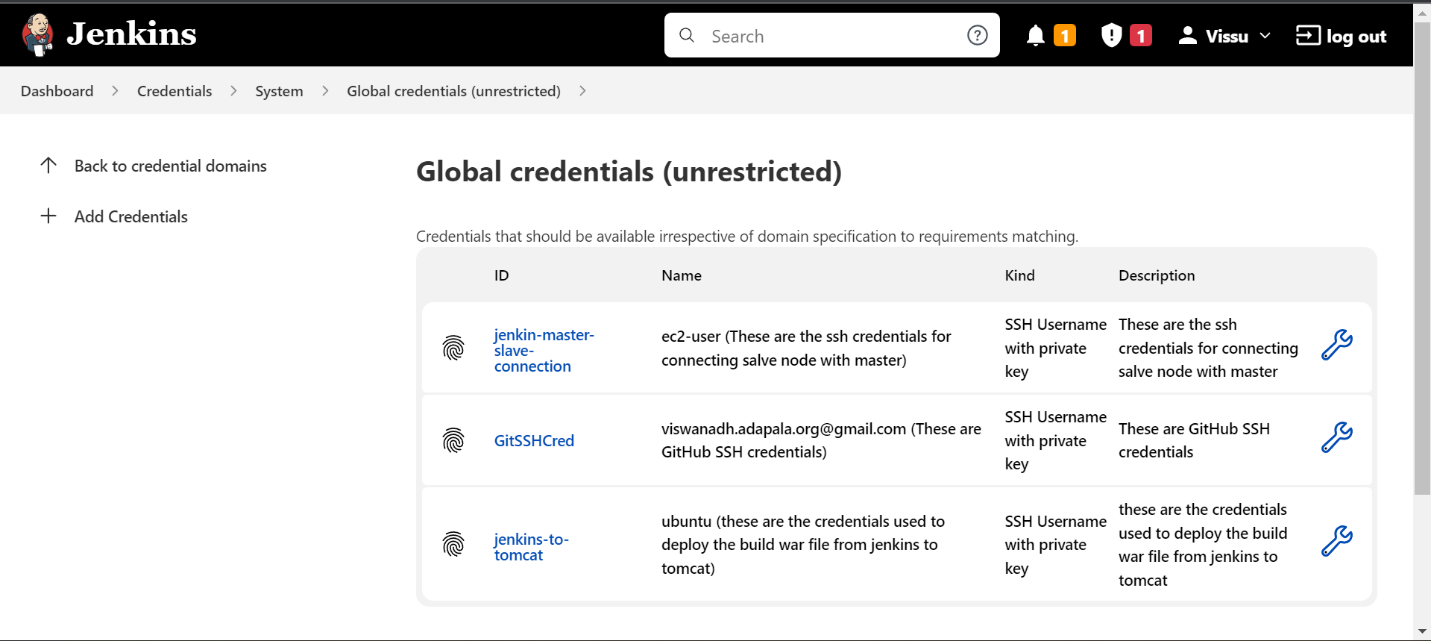
Description automatically generated

Select “Jenkins”.

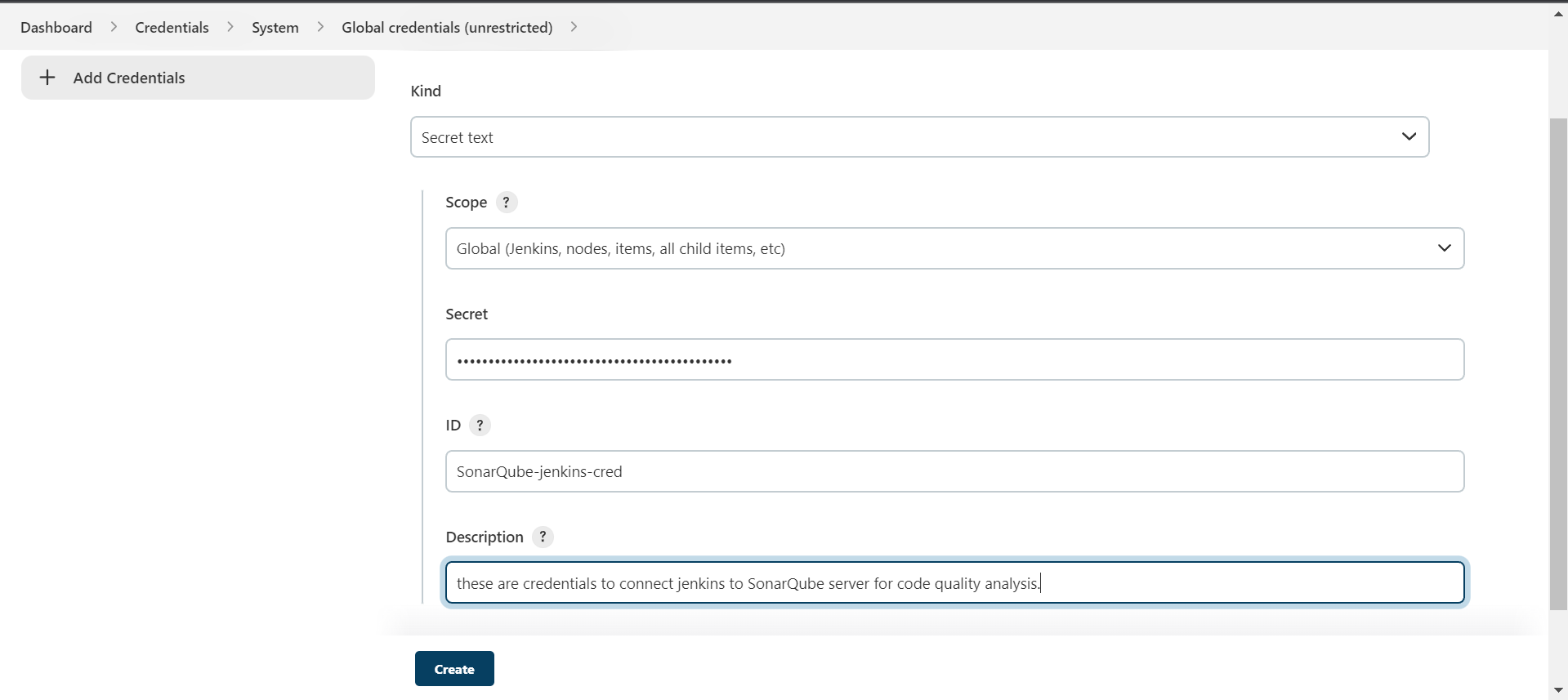
Graphical user interface, application

Description automatically generated

Select “global credentials”



Click on “Add Credentials”



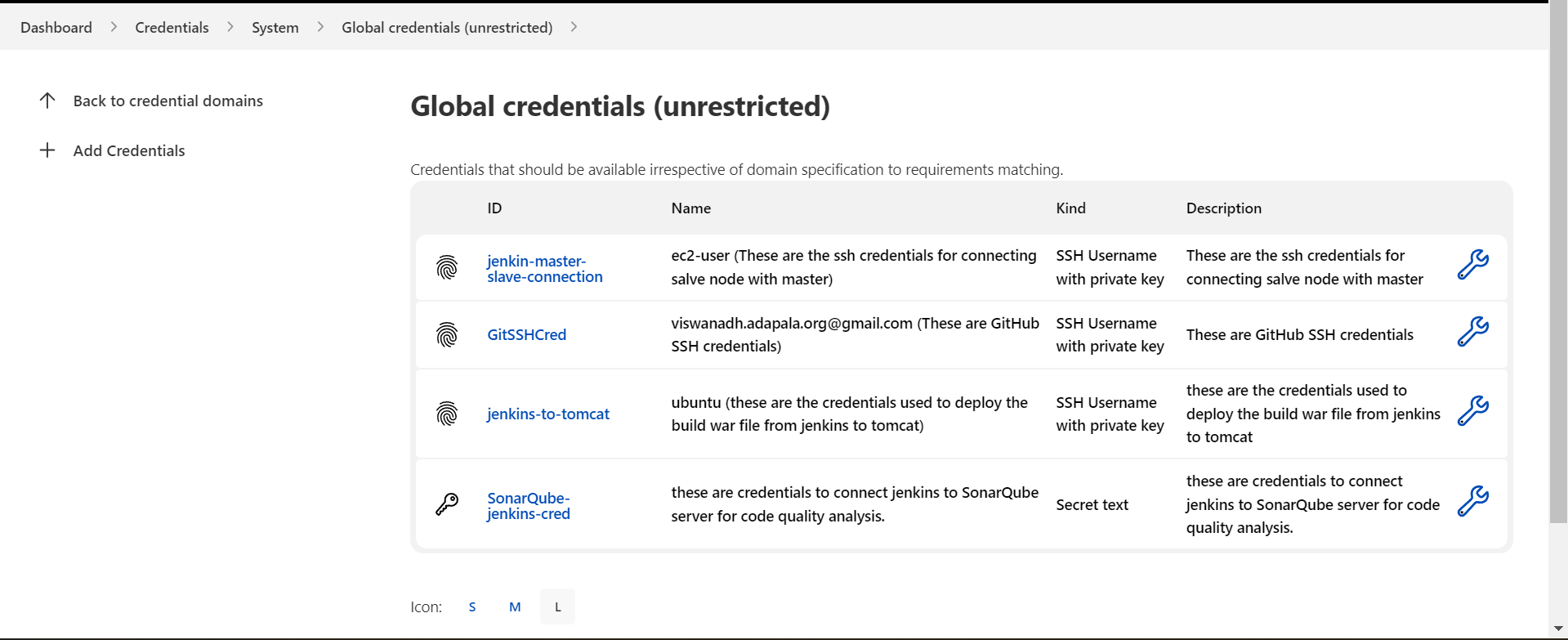
Select “kind = secret text”

paste the saved SonarQube server authorization token in the “secret” filed.

Enter the “ID for the credentials”

Enter the description.

Click on “Create”.

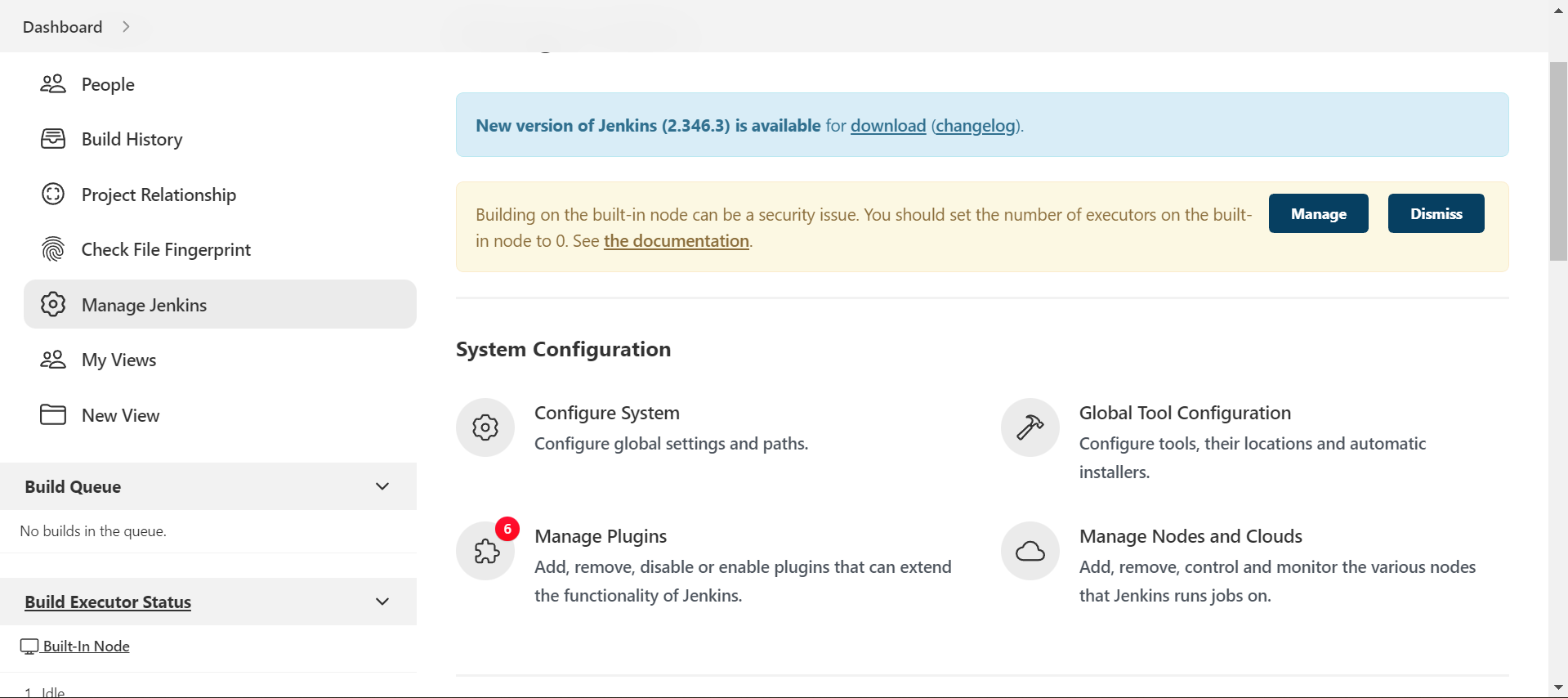


New credentials are added.

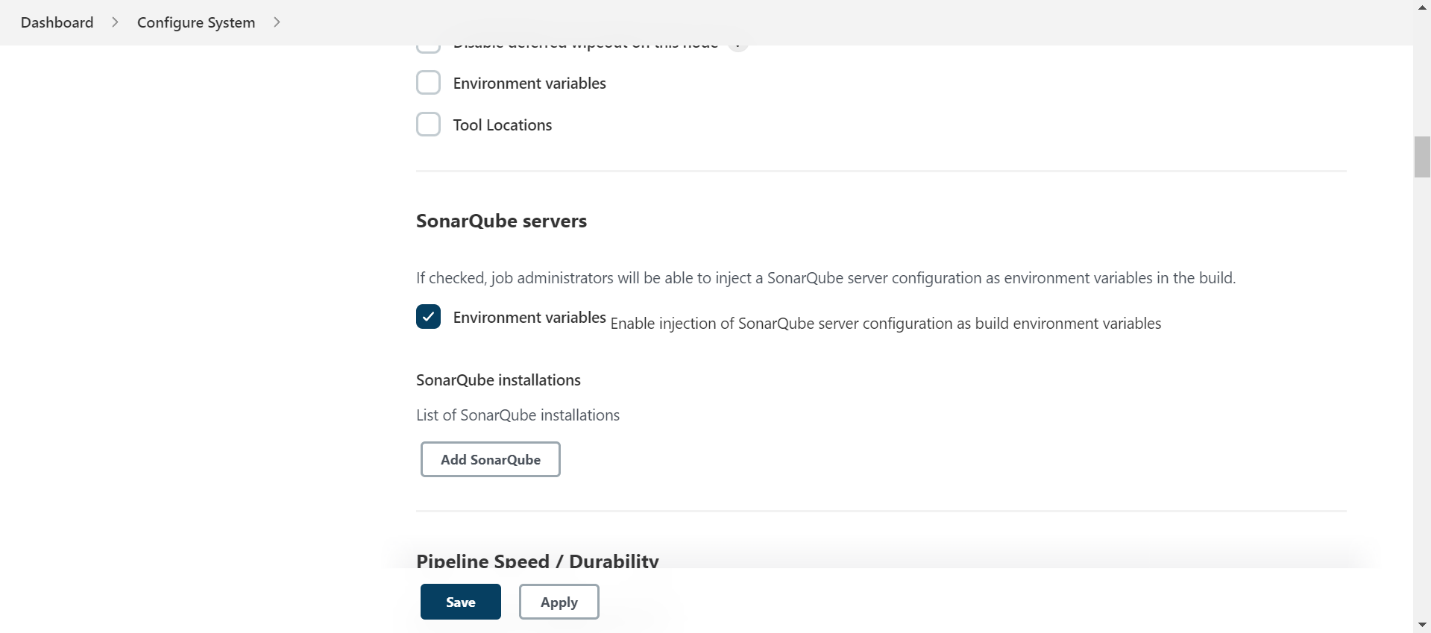
Now, integrate the SonarQube server with jenkins using the authentication token and plugin installed.

**How to configure SonarQube server settings in Jenkins?**

Go to “Manage jenkins”

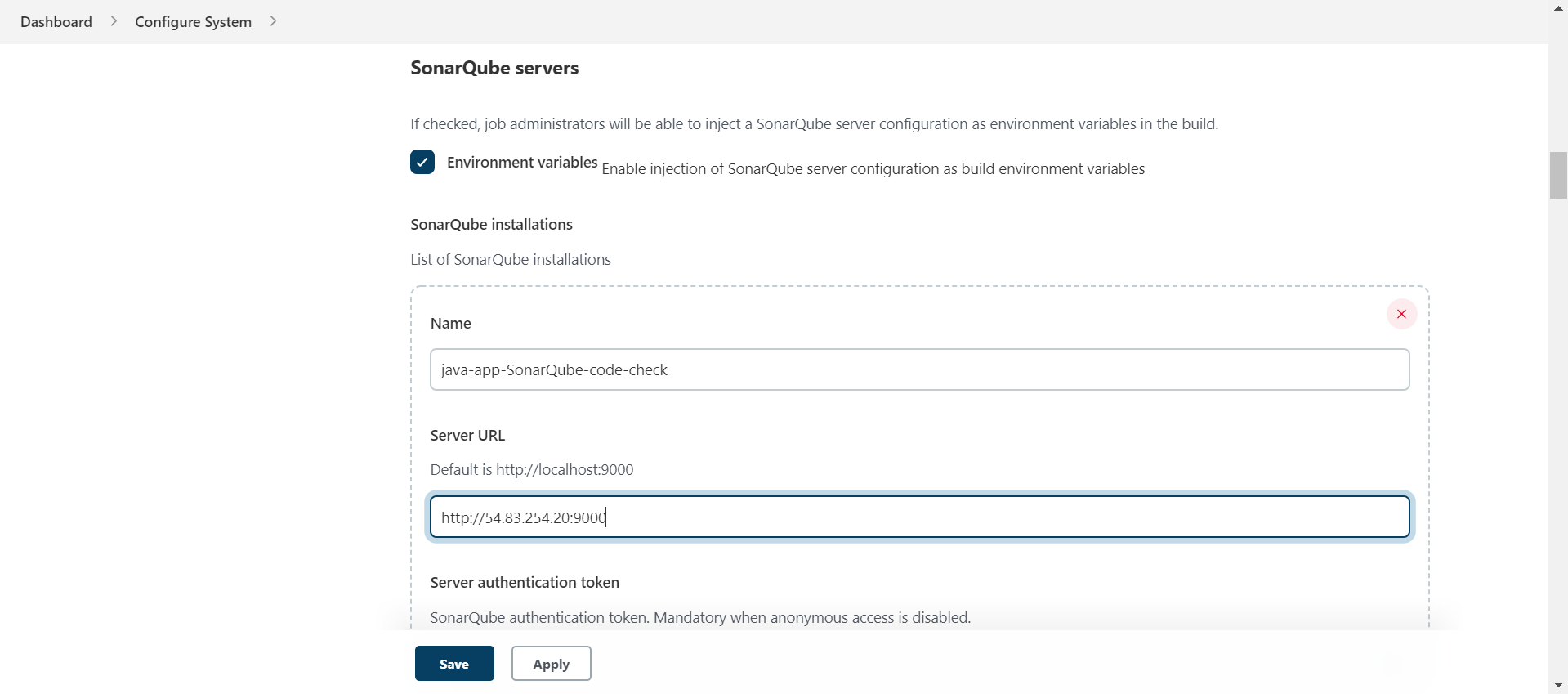


Select “Configure system”



Go to SonarQube servers’ section and select the “Environmental variables” checkbox.

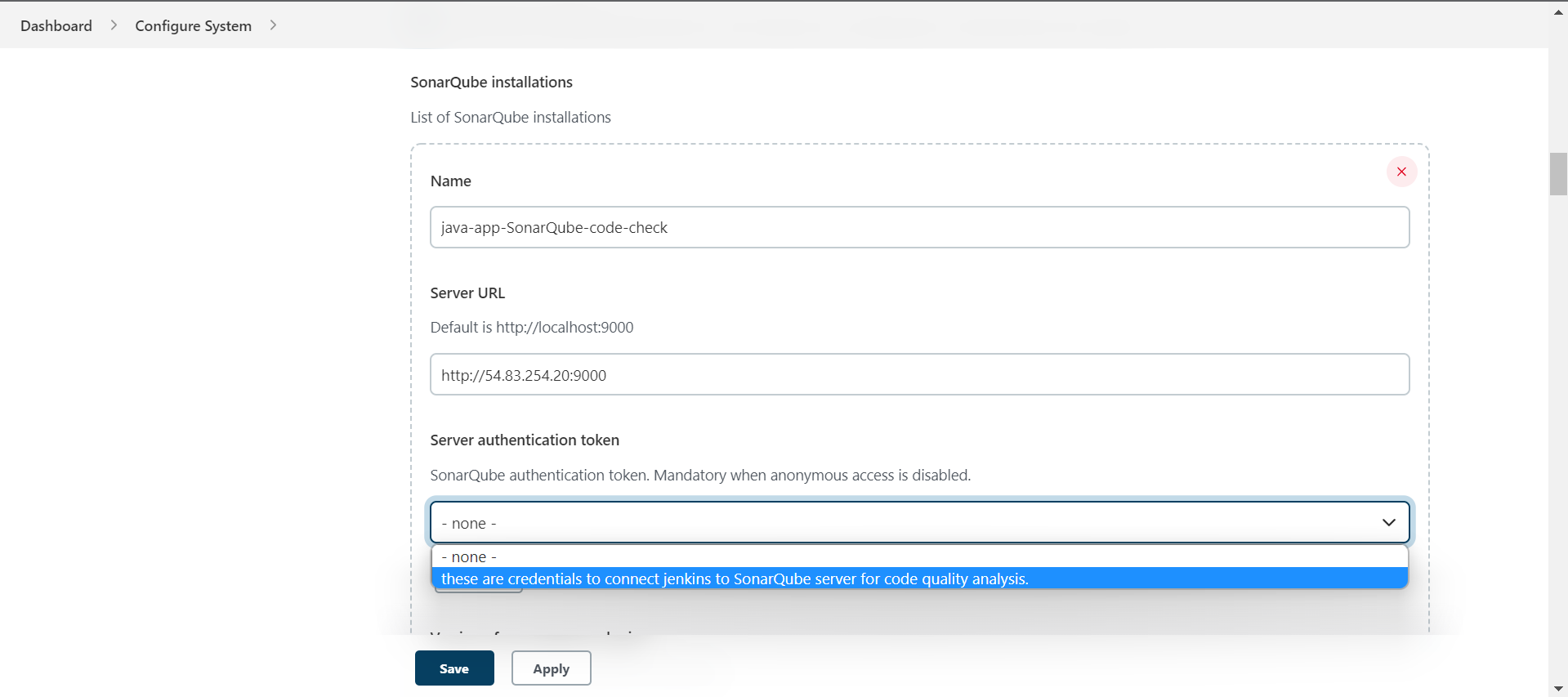
Click on “Add SonarQube”.



Enter the name for the server

Enter the SonarQube server url, this will be like this

http://<public-ip>:port



Select the above added SonarQube server credentials.

Click on apply and save.

Now, lets add the SonarQube code quality check stage to our pipeline.

**How to add code quality check stage to our pipeline?**

Go to jenkins dashboard and open our java pipeline job.

Graphical user interface, application

Description automatically generated

Click on “Configure” and go to the pipeline section of the job.

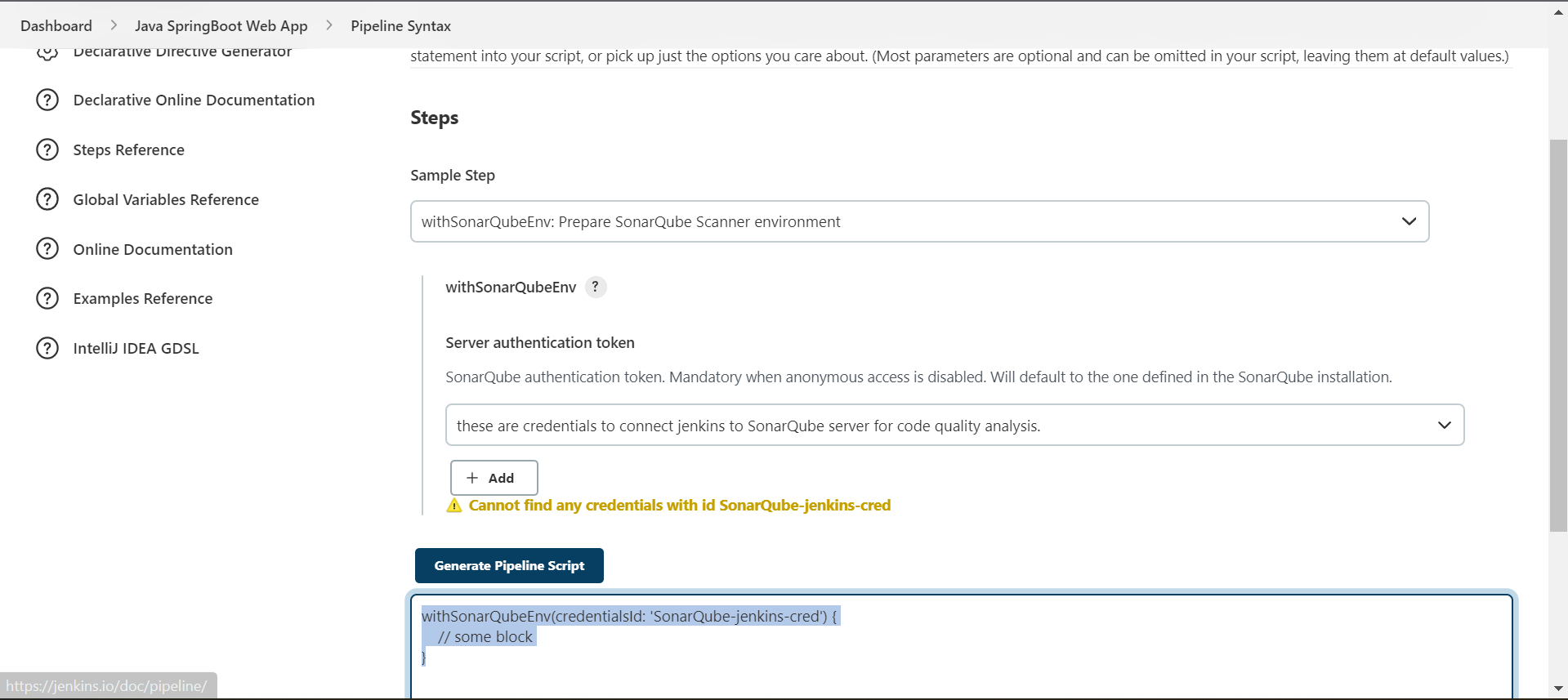
Graphical user interface, text, application

Description automatically generated

Added a new stage called “code quality check”.

Now write the code in this step.

If we don’t know how to write take the help of pipeline syntax.



Copy the generated code and paste it in the pipeline step.

Graphical user interface, text, application

Description automatically generated

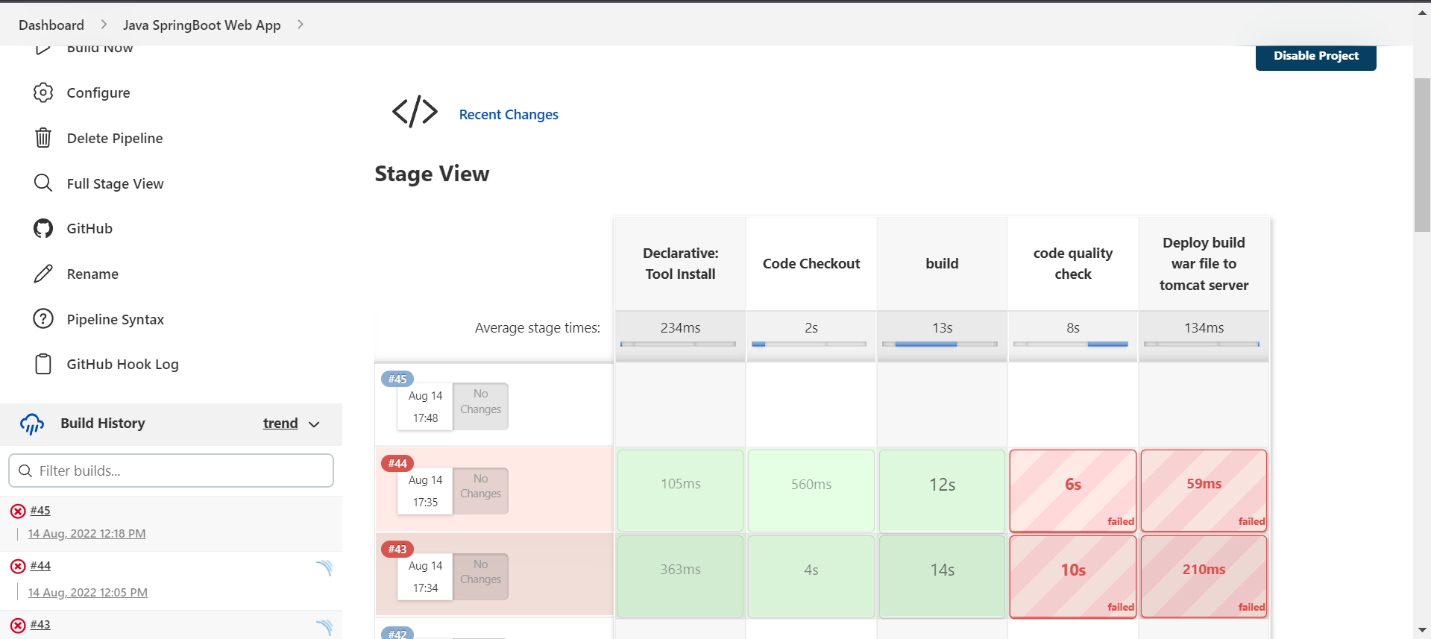
* mvn clean sonar:sonar # command to run the SonarQube scanner.

Click on apply and save.

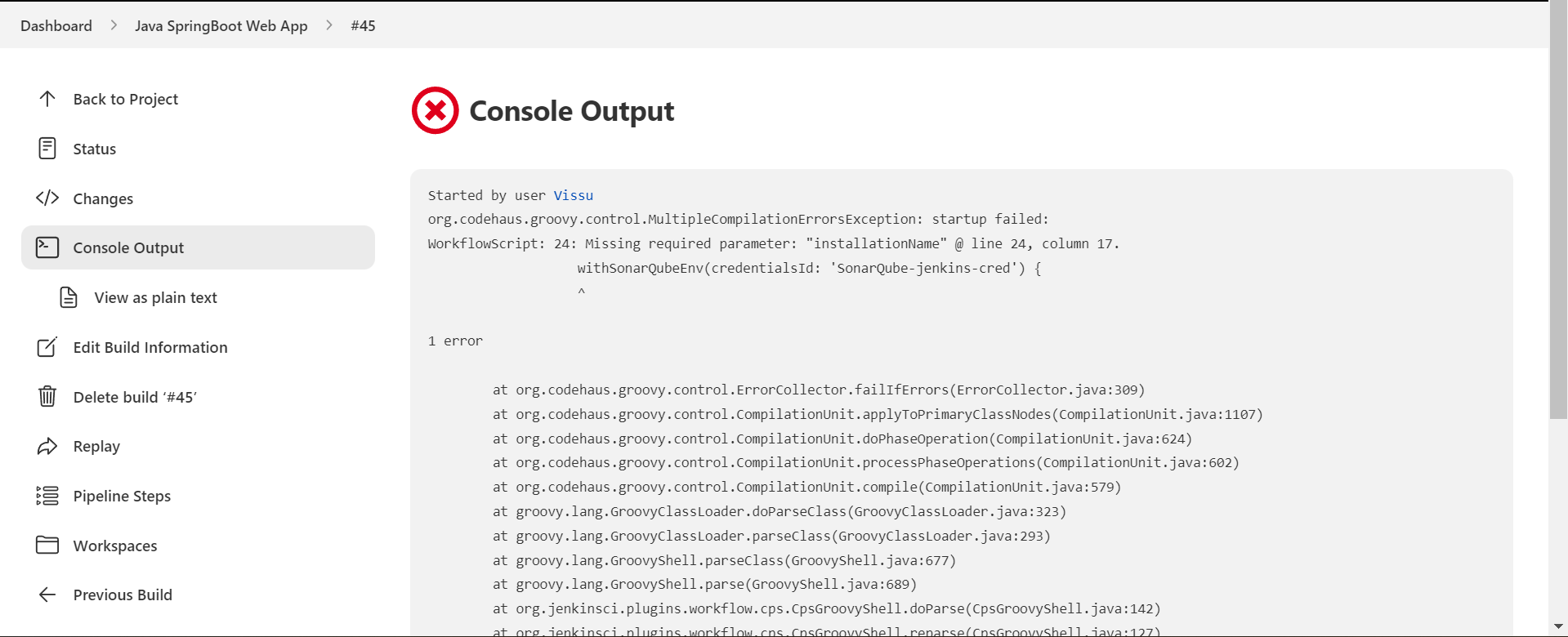
Now, run the build

Graphical user interface, text, application

Description automatically generated

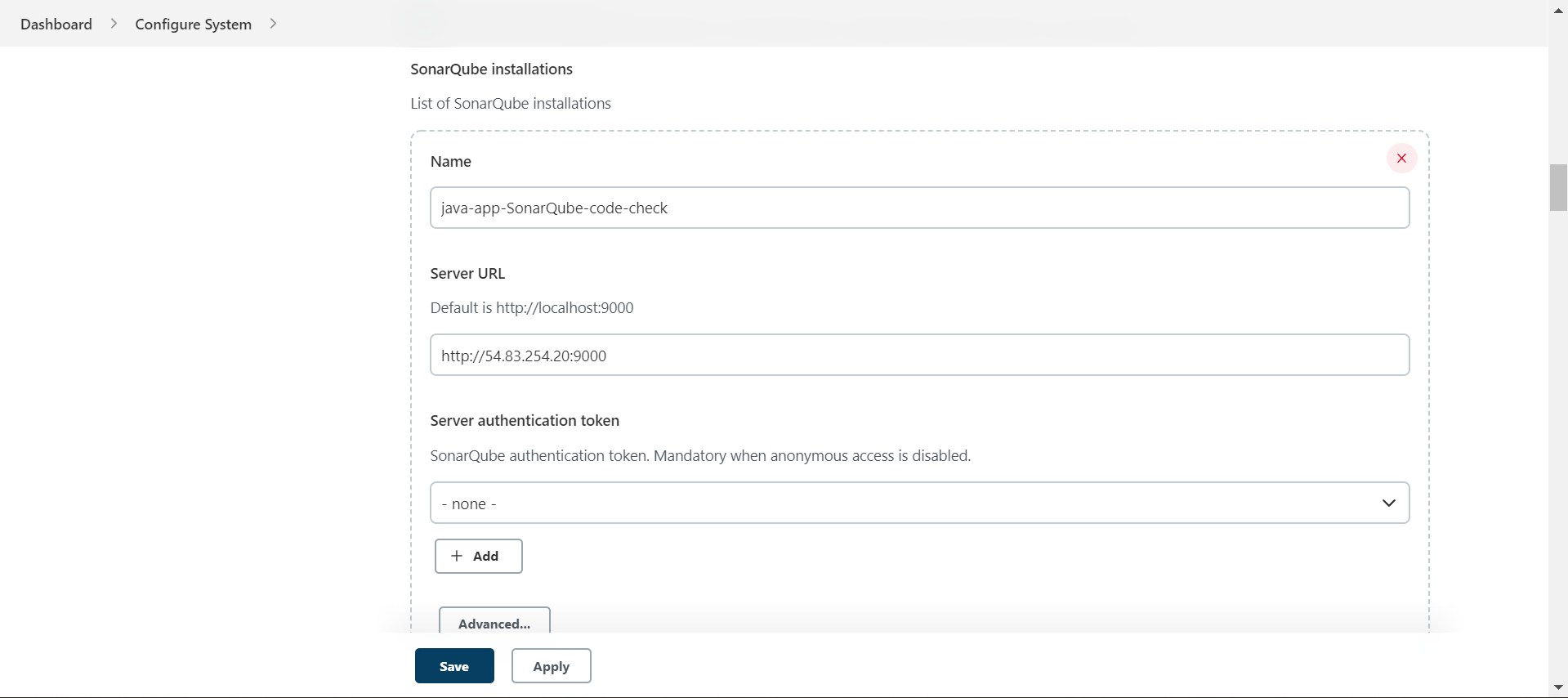


Build 45 is failed to start, let’s check the logs



It saying we are missing one required argument “installationName” in the “withSonarQubeEnv()” method.

installationName is nothing but the server name we have given while configuring in the “configure system” section



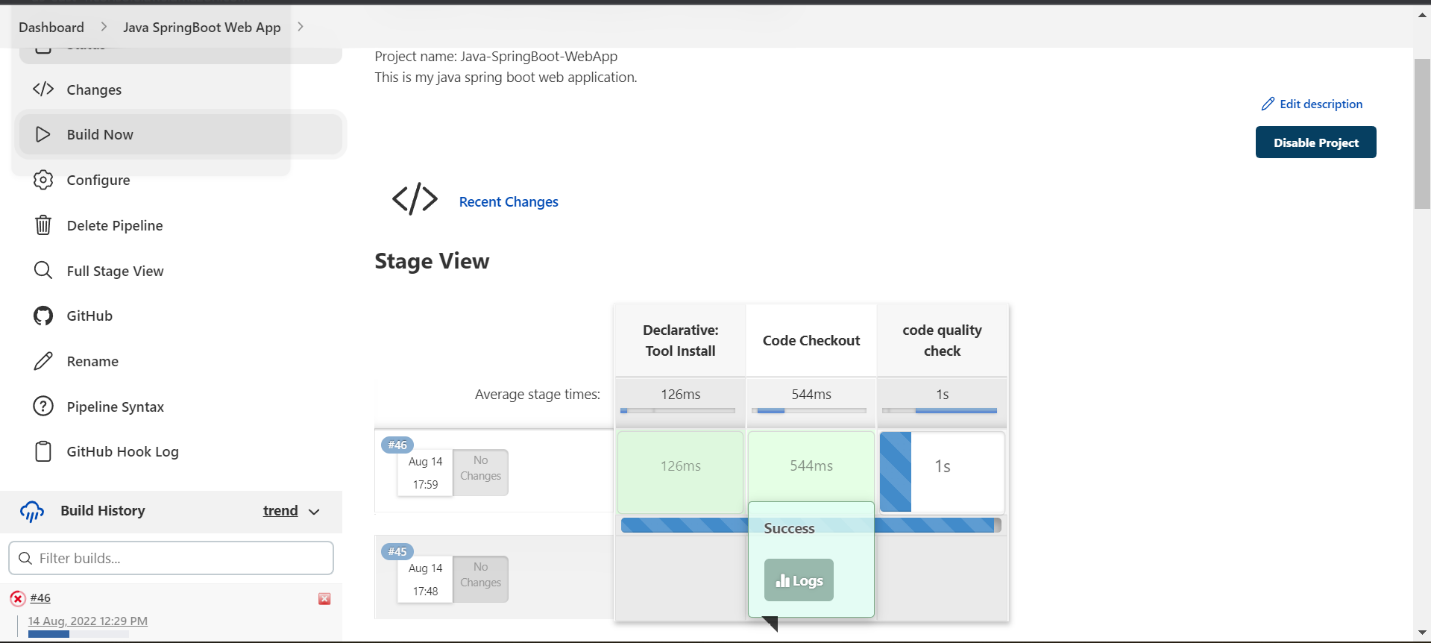
Specify this name in “withSonarQubeEnv()” method.

Graphical user interface, text, application

Description automatically generated

Click on apply and save.

Now, start the build



Build number 46 is started.

Graphical user interface, application, table

Description automatically generated

Build 46 got failed at the stage “code quality check”.

Let’s check the build logs.



Your error tells you, that you are using JDK 8 (52.0) but the plugin code was compiled with JDK 11 (55.0). So your JDK is unable to understand the plugin classes.

What to do? Change the JDK with which you execute the maven build to JDK 11. Your source code can still be java 8 and you can also compile for jdk8 - but the excuting JDK must be 11+.

Why I am getting this error, because I am using java 8 on my slave node machine where my maven will use jdk-8 to compile the code but when send the code to SonarQube server for analysis I am using java-11. So, the analysis will be performed using jdk-11. So this is the problem.

To overcome this, I created a new slave node with java-11 installed and executed this pipeline in that new salve node.

Graphical user interface, table

Description automatically generated

Now, run the build.



Now, the new build 47 got executed in the new slave node.

This build got failed. Lets check the error in the logs.

Text

Description automatically generated

This means there is now build files, that means no java compiled files.

Why I am getting this error means.

After executing the “mvn clean install” command in build stage which will produce a build file.

After this stage in “code quality check” stage I am again using command

“mvn -clean sonar:sonar”

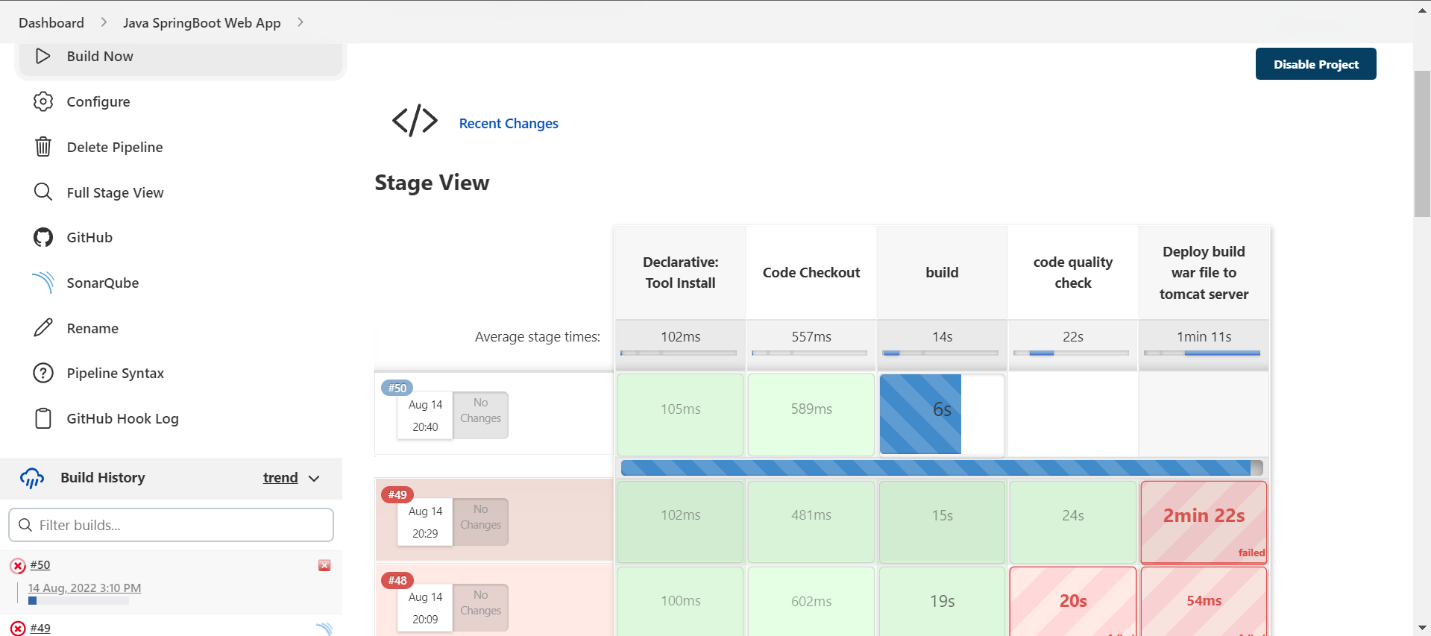
clean will delete the build file produced by build stage, so as there is no build to do the analysis this code quality check is failing.

How to solve this,

Simply remove “clean” word in maven code check command.

“mvn sonar:sonar”

Now, run the build



New build 50 got started,

Table

Description automatically generated

Build got succeeded.

Now, go to our SonarQube server and refresh it once.

Graphical user interface, text, application, email

Description automatically generated

A new project is created with the name that is given in our project pom.xml file

Graphical user interface, text, application, email

Description automatically generated

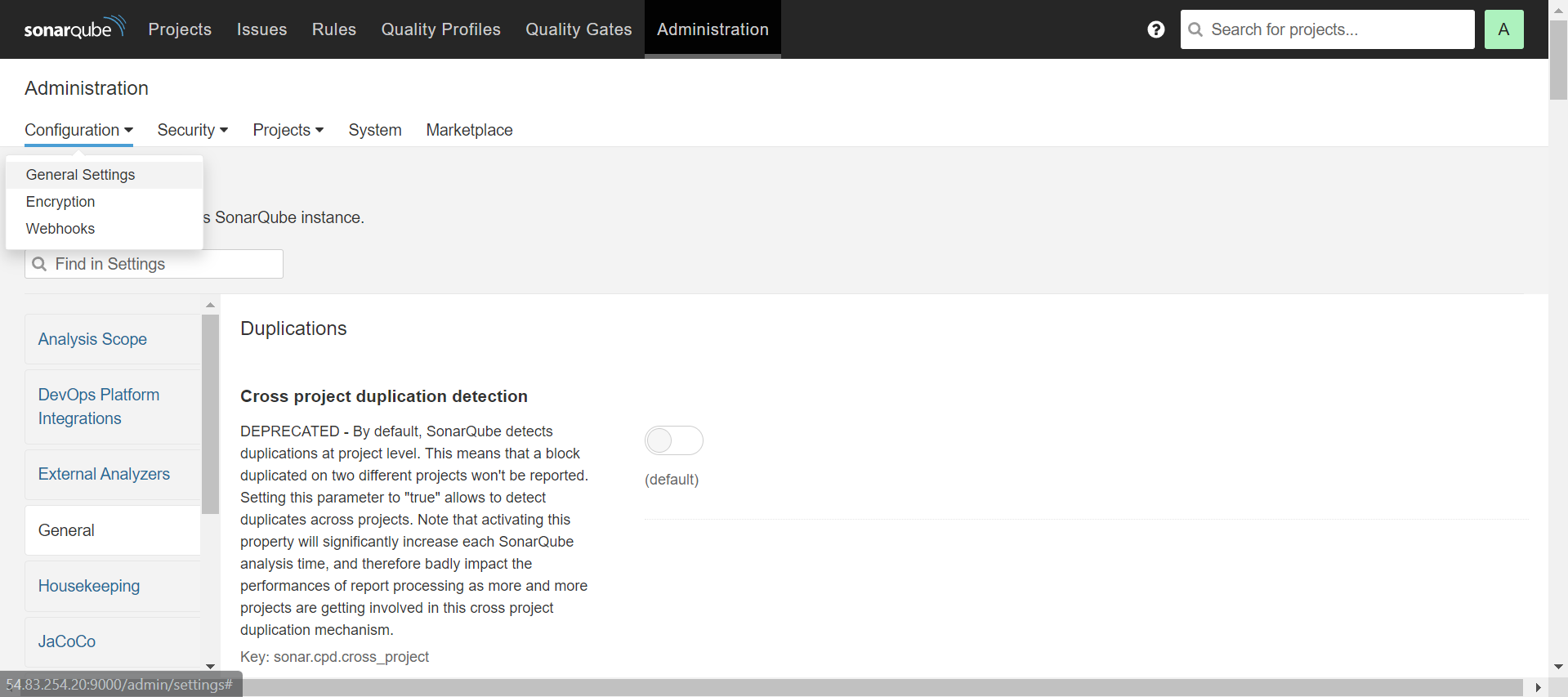
Click on the project name to see the detailed report.

A screenshot of a computer

Description automatically generated

Now, let’s say we want to continue further stages in our pipeline only if the “Quality Gate Status” is passed.

To do this, we need to add a webhook in our SonarQube server.



Go to “Administration” 🡺 Configuration 🡺 webhooks

Graphical user interface, text, application, website

Description automatically generated

Click on “Create”

This will prompt a window.

Graphical user interface, application

Description automatically generated

Enter the name for webhook

Enter the webhook url

This will be our Jenkins url + sonarqube-webhook

<http://jekins-server-ip:8080/sonarqube-webhook/>

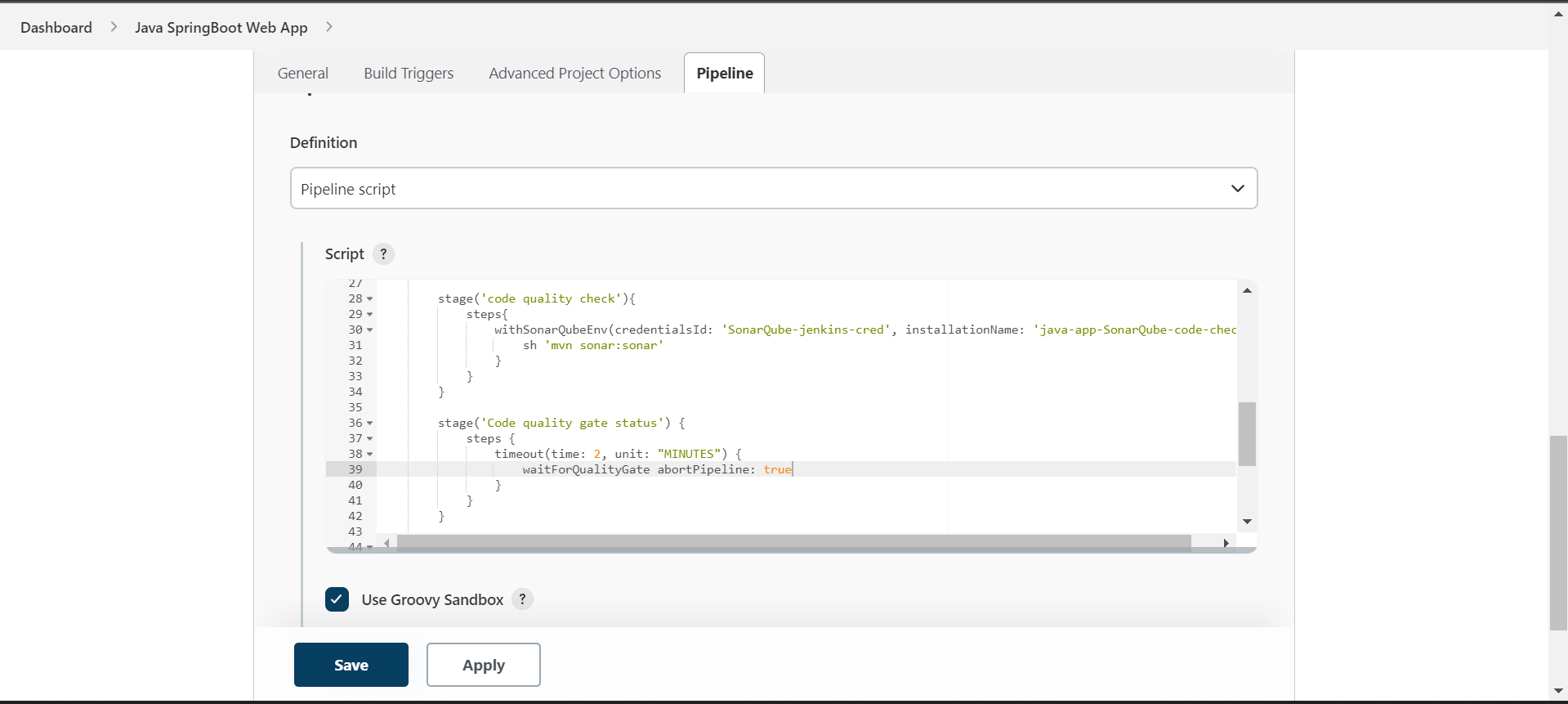
click on “Create”

Graphical user interface, text, application

Description automatically generated

Webhook is added successfully.

Now, add the new stage for “Quality Gate State” after code quality check stage in the pipeline.



stage('Code quality gate status') {

steps {

timeout(time: 2, unit: "MINUTES") {

waitForQualityGate abortPipeline: true

}

}

}

What will happen here,

once code analysis stage started it will start the analysis of code in the SonarQube server and the “'Code quality gate status” will wait for 2 minutes for the response from SonarQube server, this is where our webhook will be used to send the code analysis gate status to jenkins from SonarQube server.

waitForQualityGate abortPipeline: true

this line says, wait for quality gate status, if it’s failed then stop the pipeline.