**Setting Up SonarQube On Ubuntu Machine Through Linux Commands**

**Server -1:** Install java and Jenkins

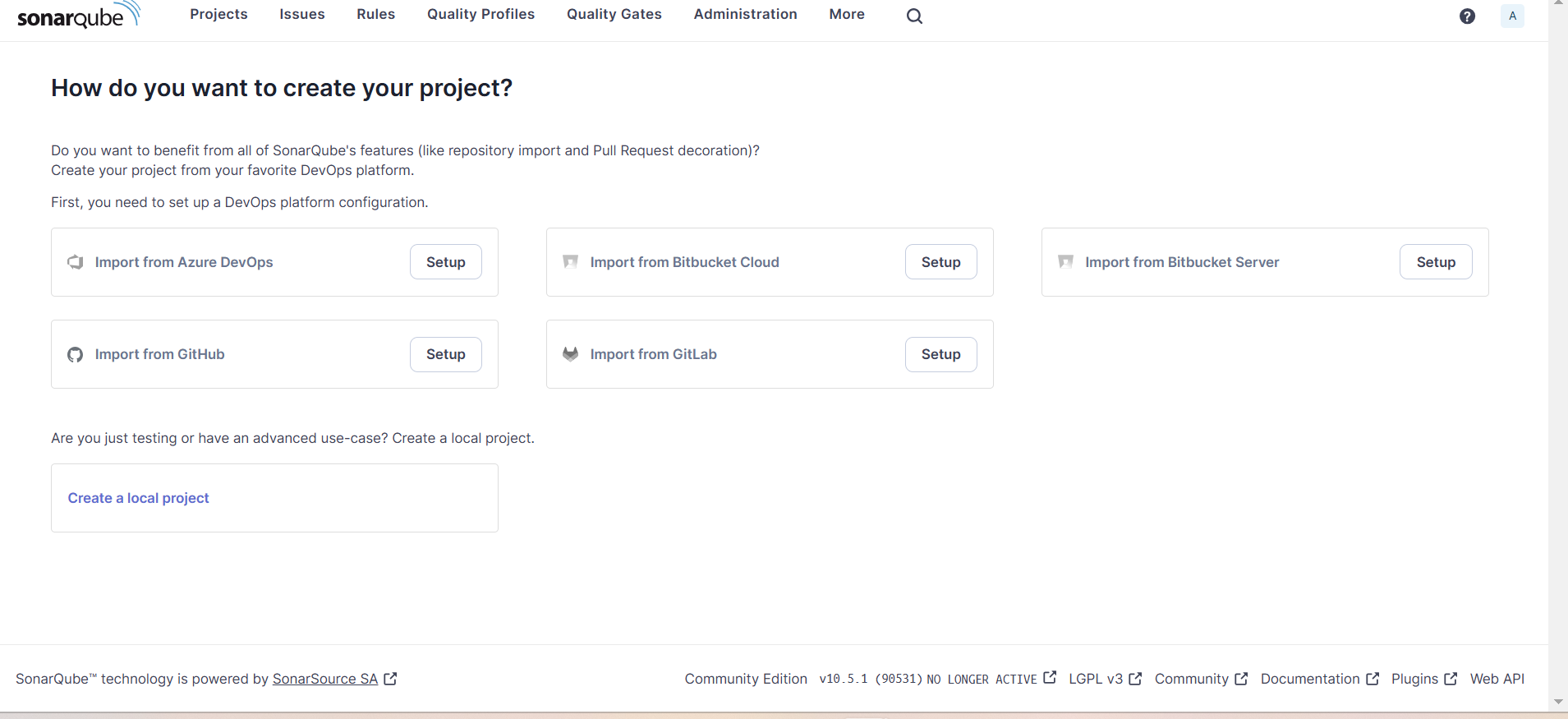
A screenshot of a computer

Description automatically generated

**Server-2:** SonarQube setup with Postgres SQL

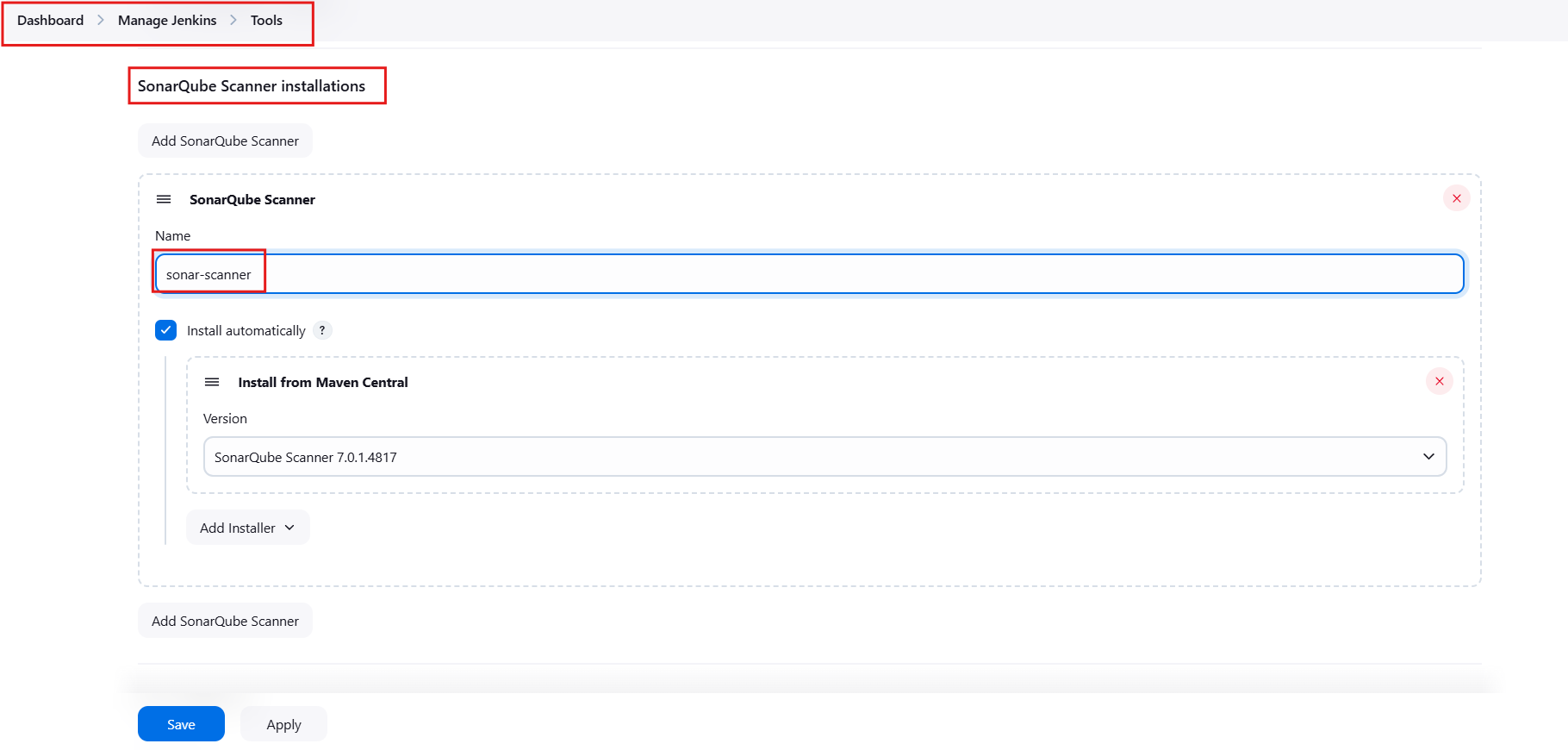
A computer screen with yellow text

Description automatically generated



A screenshot of a chat

Description automatically generated



A screenshot of a notebook

Description automatically generated

A screenshot of a computer

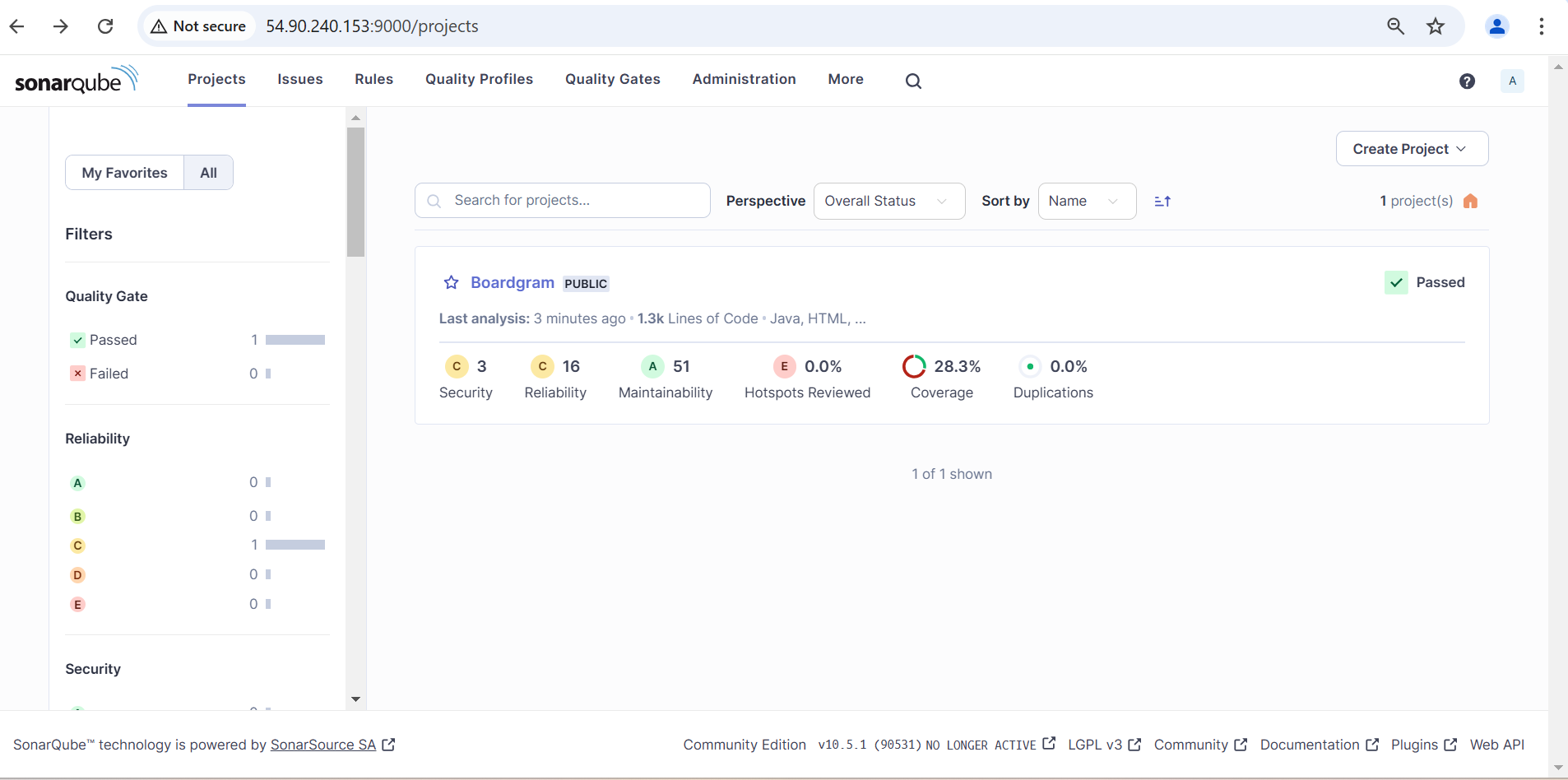
Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

 A screenshot of a computer

Description automatically generated

🡺 Below change for check and setuping the Quality Gate flow.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

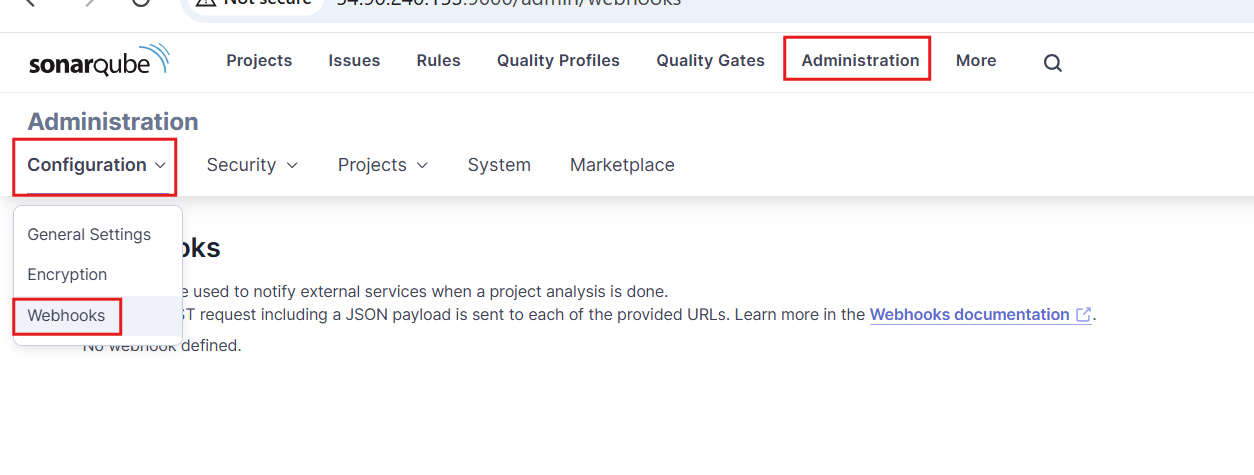
Description automatically generated

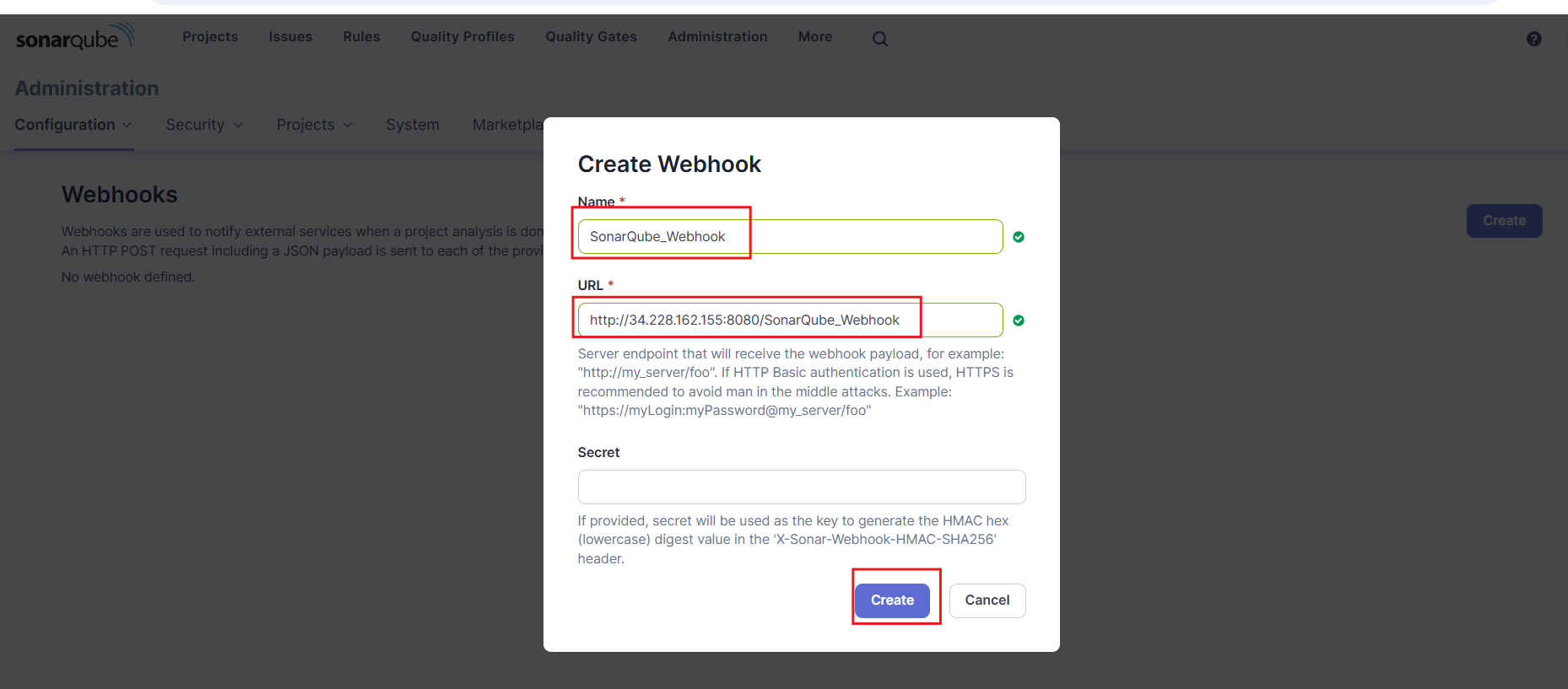
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



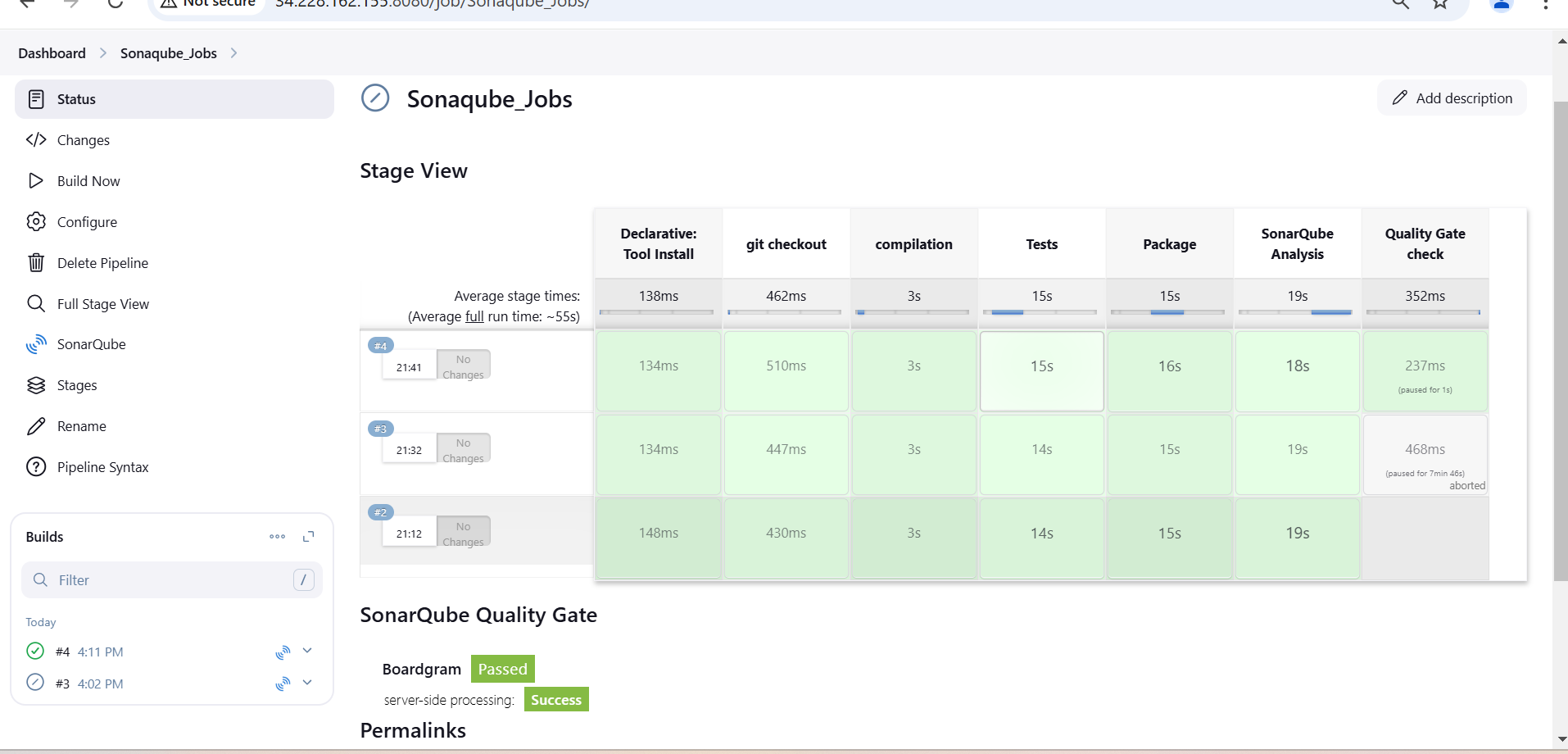


A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

**Refe -Pipeline Code:**

pipeline {

agent any

tools {

maven 'maven3'

}

environment{

SCANNER\_HOME = tool 'sonar-scanner'

}

stages {

stage('git checkout') {

steps {

git branch: 'main', url: 'https://github.com/Jagan-18/Boardgame.git'

}

}

stage('compilation') {

steps {

sh 'mvn compile'

}

}

stage('Tests') {

steps {

sh 'mvn test'

}

}

stage('Package') {

steps {

sh 'mvn package'

}

}

stage('SonarQube Analysis') {

steps {

withSonarQubeEnv('sonar-server') {

sh '''

$SCANNER\_HOME/bin/sonar-scanner -Dsonar.projectkey=Boardgame -Dsonar.projectName=Boardgram \

-Dsonar.java.binaries=target '''

}

}

}

stage('Quality Gate check ') {

steps {

timeout(time: 1, unit: 'HOURS') {

waitForQualityGate abortPipeline: false, credentialsId: 'token\_sonar'

}

}

}

}

}