Step 1: Install Jenkins

1.1 Update Packages

First, update your system's package list:

```
sudo apt update
sudo apt upgrade -y
```

1.2 Install Java

Jenkins requires Java. If it's not installed, you can install OpenJDK 11 (or any compatible version) using:

```
sudo apt install openjdk-11-jdk -y
```

Verify the Java installation

```
java -version
```

1.3 Add Jenkins Repository

Add the Jenkins repository and its GPG key:

```
wget -q -O - https://pkg.jenkins.io/keys/jenkins.io.key | sudo apt-key
add -
```

Add the Jenkins repository to your sources list:

```
sudo sh -c 'echo deb http://pkg.jenkins.io/debian/ stable main >
/etc/apt/sources.list.d/jenkins.list'
```

1.4 Install Jenkins

Update your package list and install Jenkins:

```
sudo apt update
sudo apt install jenkins -y
```

1.5 Start and Enable Jenkins

Start Jenkins and enable it to start on boot:

```
sudo systemctl start jenkins
sudo systemctl enable jenkins
```

1.6 Access Jenkins

Open Jenkins in a browser:

```
http://<your_vm_ip>:8080
```

To unlock Jenkins, retrieve the initial admin password:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Copy the password, paste it into the browser to unlock Jenkins.

Step 2: Install Tomcat

2.1 Install Dependencies

Install the required packages for Tomcat:

```
sudo apt install wget unzip -y
```

2.2 Download and Install Tomcat

Download the latest Tomcat version from the official website:

```
wget
https://downloads.apache.org/tomcat/tomcat-9/v9.0.69/bin/apache-tomcat
-9.0.69.tar.gz
```

Extract the downloaded archive:

```
tar xvf apache-tomcat-9.0.69.tar.gz
```

Move it to a desired location, for example `/opt`:

```
sudo mv apache-tomcat-9.0.69 /opt/tomcat
```

2.3 Set Permissions

Set the proper permissions for the Tomcat directory:

```
sudo chown -R $USER:$USER /opt/tomcat
```

2.4 Start Tomcat

Navigate to the Tomcat `bin` directory and start Tomcat:

```
cd /opt/tomcat/bin
./startup.sh
```

Tomcat should now be running on port 8080.

2.5 Access Tomcat

Open Tomcat in a browser:

```
http://<your_vm_ip>:8080
```

You should see the Tomcat home page.

Step 3: Install SonarQube

3.1 Install Dependencies

SonarQube requires Java and a database. First, install the necessary packages:

```
sudo apt install openjdk-11-jdk -y
sudo apt install unzip wget -y
```

3.2 Download and Install SonarQube

Download SonarQube from the official website:

```
wget
https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.7.
1.62043.zip
```

Unzip the downloaded archive:

```
unzip sonarqube-9.7.1.62043.zip
```

Move it to the `/opt` directory:

```
sudo mv sonarqube-9.7.1.62043 /opt/sonarqube
```

3.3 Set Permissions

Set the proper permissions:

```
sudo chown -R $USER:$USER /opt/sonarqube
```

3.4 Configure SonarQube

Before starting SonarQube, you might need to configure the database connection in `/opt/sonarqube/conf/sonar.properties`.

Ensure that SonarQube has access to the database server.

3.5 Start SonarQube

Start SonarQube using the `sonar.sh` script:

```
cd /opt/sonarqube/bin/linux-x86-64
./sonar.sh start
```

3.6 Access SonarQube

Open SonarQube in a browser:

```
http://<your_vm_ip>:9000
```

Default login credentials:

- Username: `admin`
- Password: `admin`

Step 4: Integrate Jenkins with SonarQube

4.1 Install SonarQube Scanner Plugin in Jenkins

1. Go to Jenkins dashboard (`http://<your_vm_ip>:8080`).
2. Navigate to **Manage Jenkins** > **Manage Plugins**.
3. Go to the **Available** tab and search for `SonarQube Scanner`.
4. Install the plugin and restart Jenkins if prompted.

4.2 Configure SonarQube in Jenkins

1. Go to **Manage Jenkins** > **Configure System**.
2. Scroll down to the **SonarQube Servers** section.
3. Add a new SonarQube server:
   - Name: `SonarQube`
   - Server URL: `http://<your_vm_ip>:9000`
   - Authentication Token: You can generate a token from SonarQube (under **My Account** > **Security**).
4. Save the configuration.

4.3 Configure SonarQube Scanner

1. Go to **Manage Jenkins** > **Global Tool Configuration**.
2. Scroll down to **SonarQube Scanner** and add a new SonarQube Scanner.
3. Set the name (e.g., `SonarQube Scanner`) and install automatically or provide a custom path.

4.4 Create Jenkins Job to Run SonarQube Analysis

1. Create a new job in Jenkins (e.g., a Freestyle project).
2. In the job configuration, under the **Build** section, add **SonarQube Scanner** and provide the necessary analysis parameters (e.g., `sonar.projectKey`, `sonar.sources`, etc.).

Step 5: Test the Integration

1. **Run the Jenkins job** that has the SonarQube Scanner configured.
2. Verify the analysis report in the SonarQube dashboard.

Step 6: Configure Tomcat as a Deployment Target (Optional)

If you want Jenkins to deploy to Tomcat after a successful build, you can add a post-build action to deploy your application to Tomcat:

1. In Jenkins, install the **Deploy to Container** plugin.
2. In your job configuration, under **Post-build Actions**, add **Deploy war/ear to a container**.
3. Configure the Tomcat server and provide the necessary credentials.

Step 7: Set Up Jenkins to Automatically Start on Boot

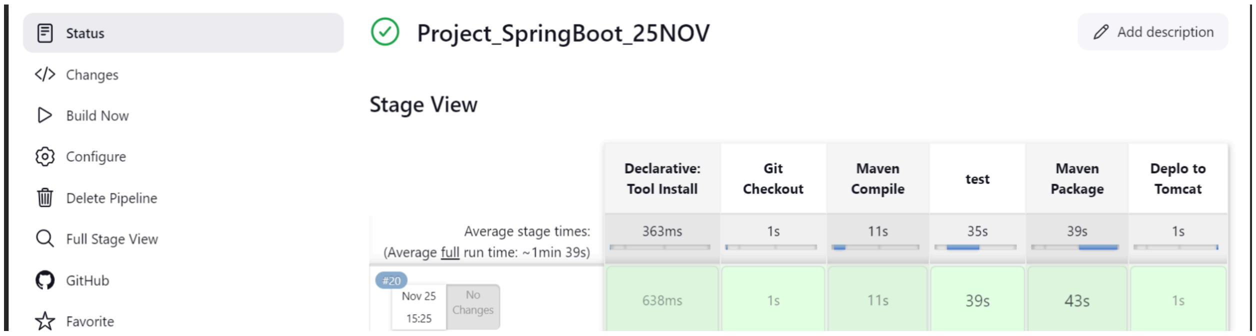
Ensure that Jenkins starts automatically when your VM is restarted:

```
sudo systemctl enable jenkins
```

Conclusion

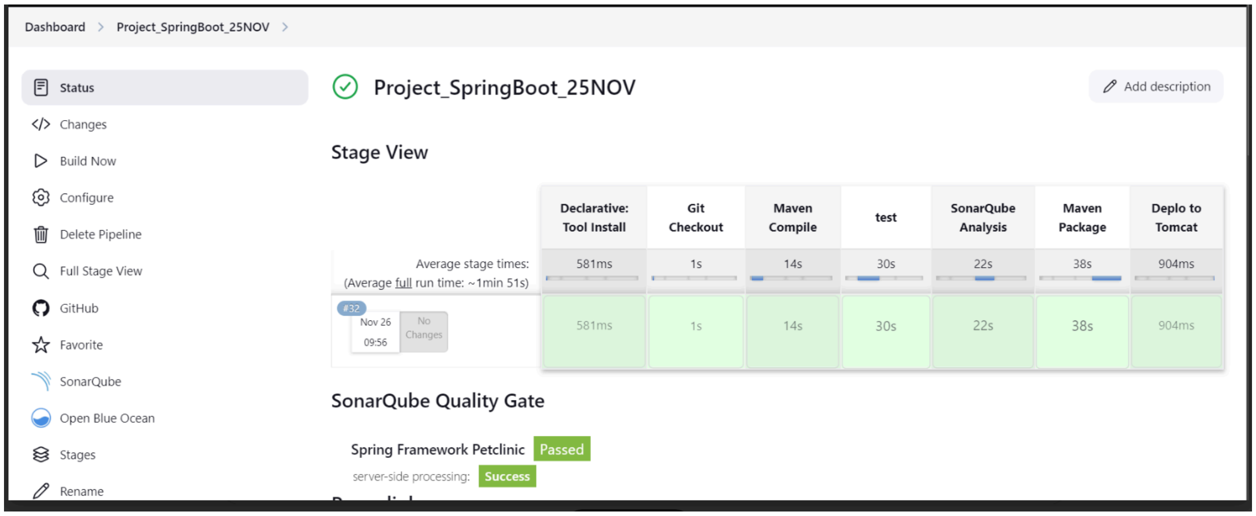
Jenkins Pipeline

```
           pipeline {
    agent any

   tools {
    jdk 'jdk11'   // Ensure JDK 21 is configured in Global Tool Configuration
    maven 'maven3' // Ensure Maven 3 is configured in Global Tool Configuration
  }

   environment {
        SCANNER_HOME = tool 'sonar-scanner'
    }

    stages {
      stage("Git Checkout"){
          steps{
                git branch: 'main', url: 'https://github.com/RahulDubey-Devops/Petclinic-demo.git'
          }
      }
      stage("Maven Compile"){
          steps{
              sh 'mvn clean compile'
          }
      }
      stage("test"){
          steps{
              sh 'mvn test'
          }
      }

stage('SonarQube Analysis') {
    steps {
        // Make sure you use the correct SonarQube server environment as defined in Jenkins
configuration
        withSonarQubeEnv('sonar-server') {
         sh 'mvn sonar:sonar'
        }
    }
}

      stage('Maven Package') {
          steps {
            // Package the project
            sh 'mvn package'
          }
        }


        stage("Deplo to Tomcat"){
            steps{
          // Switch to root user and provide the password for su
            sh "sudo cp target/*.war /opt/tomcat/webapps/"
            }
        }
    }
}
```


Project_SpringBoot_25NOV

Stage View

	Declarative: Tool Install	Git Checkout	Maven Compile	test	Maven Package	Deplo to Tomcat
Average stage times: (Average full run time: ~1min 39s)	363ms	1s	11s	35s	39s	1s
#20 Nov 25 15:25 No Changes	638ms	1s	11s	39s	43s	1s


Dashboard > Project_SpringBoot_25NOV >

Project_SpringBoot_25NOV

Stage View

	Declarative: Tool Install	Git Checkout	Maven Compile	test	SonarQube Analysis	Maven Package	Deplo to Tomcat
Average stage times: (Average full run time: ~1min 51s)	581ms	1s	14s	30s	22s	38s	904ms
#32 Nov 26 09:56 No Changes	581ms	1s	14s	30s	22s	38s	904ms

SonarQube Quality Gate

Spring Framework Petclinic Passed

server-side processing: Success