Jenkins Pipeline Documentation: Triggering GitHub Code and SonarQube Analysis

Overview:

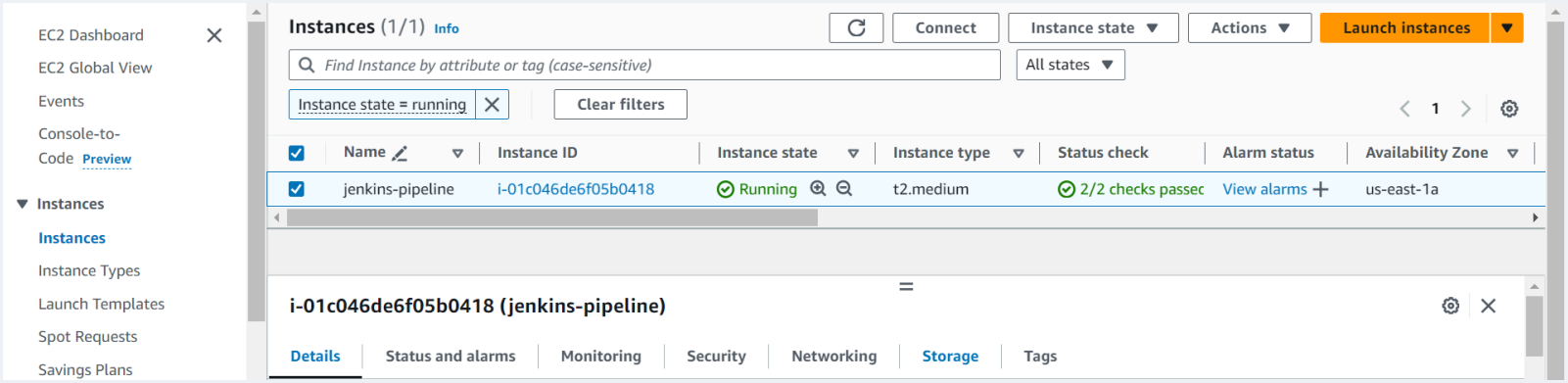
This document provides a step-by-step guide on how to set up a Jenkins pipeline to trigger code hosted on GitHub and also analyse the same code with sonarqube

Prerequisites

1. Jenkins server installed and running.
2. SonarQube server installed and accessible.
3. Jenkins user with appropriate permissions to create and configure pipelines.
4. GitHub repository containing the code you want to trigger.

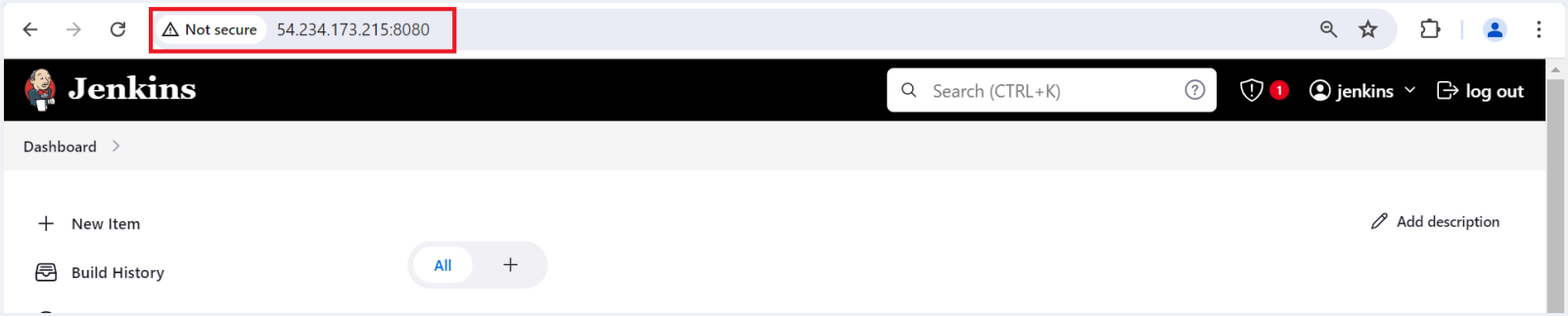
Setting up an instance and jenkins running on that

- Create an ec2 instance from the aws console as shown below



Refer this link https://www.jenkins.io/doc/book/installing/linux/#debianubuntu for installation

Check whether the installation is properly running on the default port by providing public ip address and the port 8080



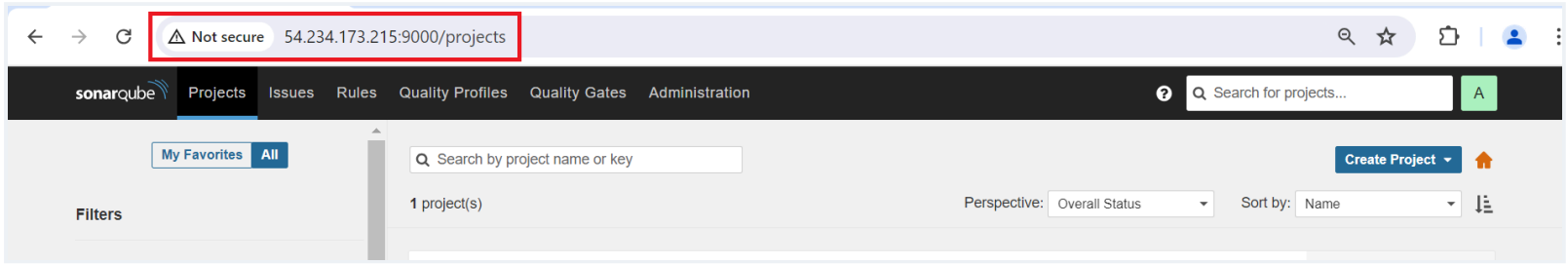
At the same time install sonarqube on the same server and check whether it is running properly on the port 9000

You can install this manually or run the following command to run sonarqube on the container by specifying the port

docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

Note: Make sure docker is installed before running the above command refer the below link for the installation of docker on (ubuntu)

https://docs.docker.com/engine/install/ubuntu/



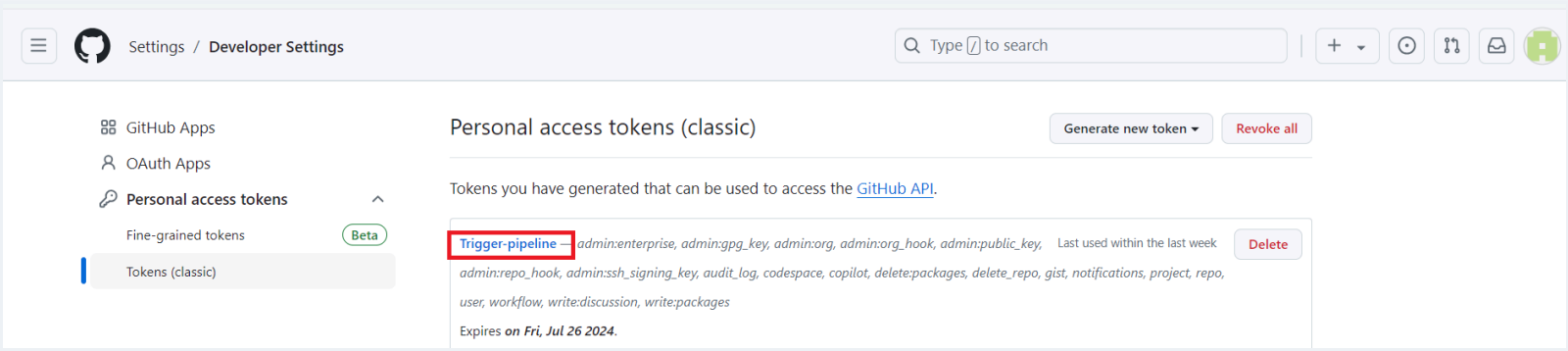
Steps

1. **Setting Up Jenkins Credentials**

Before configuring the pipeline, ensure Jenkins has the necessary credentials to access your GitHub repository

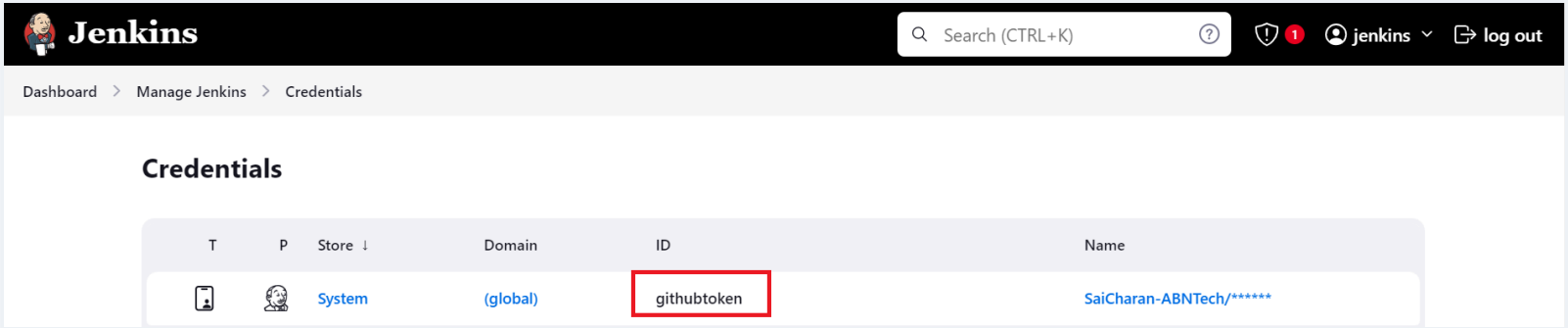
1. Generate GitHub Personal Access Token:

- Go to your GitHub account.
- Navigate to Settings > Developer settings > Personal access tokens.
- Click Generate new token, and give it appropriate permissions (Trigger-pipeline).
- Copy the generated token.



2. Add GitHub Credentials to Jenkins:

- In Jenkins, go to Credentials > System > Global credentials.
- Click Add credentials.
- Choose Secret text for the kind.
- Paste the token in the Secret field and provide an ID (e.g., github-token).
- Save the credentials.



2. **Creating a Jenkins Pipeline**

Now, create a Jenkins pipeline that triggers code from GitHub.

1. Create a New Pipeline Job:

- Go to the Jenkins dashboard.
- Click New Item.
- Enter a name for your job (e.g., Trigger-pipeline) and select Pipeline.

2. Configure Pipeline Settings:

- In the pipeline configuration:

Script ?

```
pipeline {
    agent any

    environment {
        scannerHome = tool 'sonar-scanner'
    }

    stages {
        stage('GitCheckout') {
            steps {
                script {
                    checkout([$class: 'GitSCM', branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[credenti
                }
            }
        }

        stage('SonarScan') {
            steps {
                script {
                    withSonarQubeEnv('sonar-server') {
                        sh "${scannerHome}/bin/sonar-scanner -Dsonar.projectName=HRportal -Dsonar.projectKey=Hrportal"
                    }
                }
            }
        }
    }
}
```

- Modify the git step's url parameter to point to your GitHub repository.
- Customize stages and steps according to your build and deployment requirements.

3. Configuring SonarQube in Jenkins

Ensure Jenkins is configured to communicate with your SonarQube server:

1. Install SonarQube Scanner Plugin:

- In Jenkins, navigate to Manage Jenkins > Manage Plugins.
- Search for and install the "SonarQube Scanner for Jenkins" plugin.

4. Configure SonarQube Server:

- In Jenkins, go to Manage Jenkins > Configure System.
- Scroll down to the SonarQube servers section.
- Click Add SonarQube and provide details (e.g., Name, Server URL, SonarQube token).
- Click on Test Connection to verify Jenkins can connect to the SonarQube server.

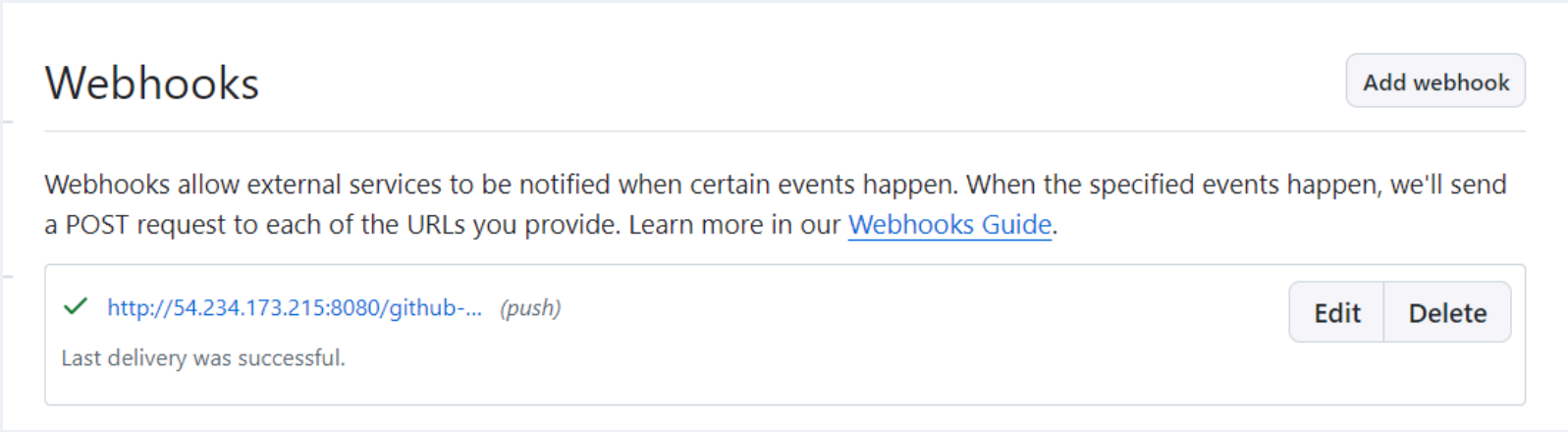
3. **Triggering the Pipeline**

To automatically trigger the pipeline

1. Configure Webhooks in GitHub:

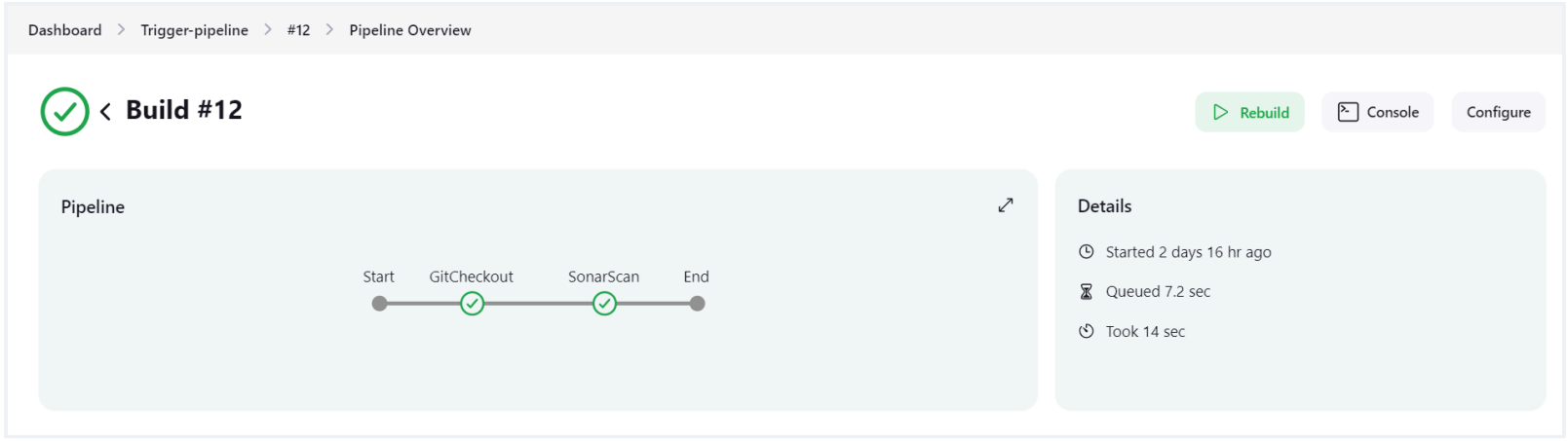
- Go to your GitHub repository.
- Navigate to Settings > Webhooks > Add webhook.
- Set the Payload URL to your Jenkins server webhook URL (http://jenkins-server/github-webhook/).
- Choose application/json as the Content type.

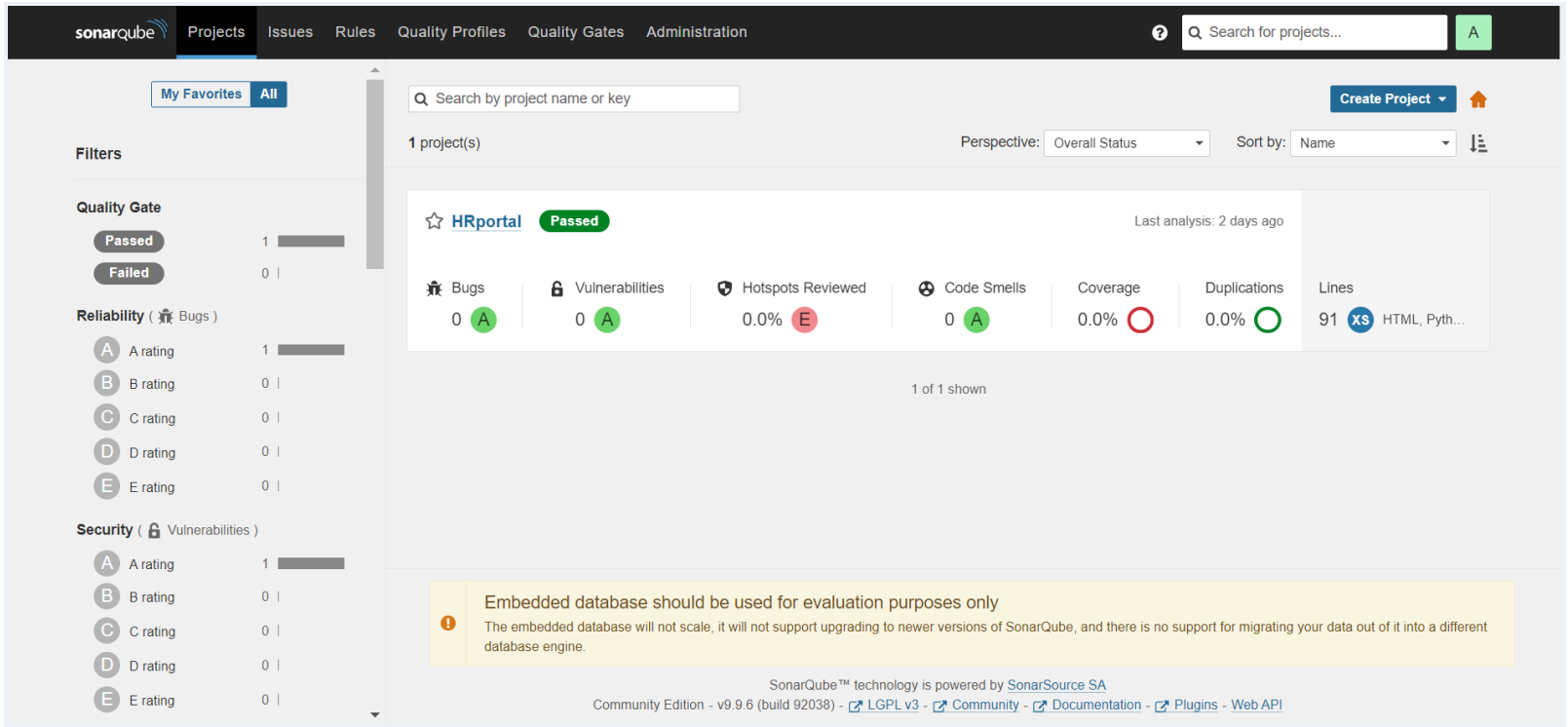
- Select the events you want Jenkins to trigger on (e.g., Just the push event).
- Add the webhook.



2. Testing the Trigger:

- Make a change to your GitHub repository (e.g., push a commit).
  - Check Jenkins console or dashboard for triggered pipeline runs.x


Dashboard
Trigger-pipeline
#12
Pipeline Overview
Build #12
Rebuild
Console
Configure
Pipeline
Start
GitCheckout
SonarScan
End
Details
Started 2 days 16 hr ago
Queued 7.2 sec
Took 14 sec


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
A
My Favorites
All
Filters
Quality Gate
Passed 1
Failed 0
Reliability (Bugs)
A rating 1
B rating 0
C rating 0
D rating 0
E rating 0
Security (Vulnerabilities)
A rating 1
B rating 0
C rating 0
D rating 0
E rating 0
Search by project name or key
Create Project
1 project(s)
Perspective: Overall Status
Sort by: Name
HRportal
Passed
Last analysis: 2 days ago
Bugs 0 A
Vulnerabilities 0 A
Hotspots Reviewed 0.0% E
Code Smells 0 A
Coverage 0.0%
Duplications 0.0%
Lines 91 XS HTML, Pyth...
1 of 1 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.9.6 (build 92038) - LGPL v3 - Community - Documentation - Plugins - Web API