Module 9 Project: Integrate SonarQube with GitHub Actions for Code Quality Analysis

# Objective:

Set up SonarQube with a sample project and configure GitHub Actions to automatically analyze code quality on every push. At the end, a SonarQube report should be generated and viewable through the dashboard.

# Project Requirements:

· A sample project in Java, Python, or JavaScript (your choice).  
· GitHub repository for version control and CI setup.

1. Install Docker
2. Install SonarQube

sudo mkdir /opt/sonarqube

cd /opt/sonarqube

sudo nano **docker-compose.yml**

version: "3"

services:

sonarqube:

image: sonarqube:community

container\_name: sonarqube

ports:

- "9000:9000"

environment:

- SONAR\_ES\_BOOTSTRAP\_CHECKS\_DISABLE=true

volumes:

- sonarqube\_data:/opt/sonarqube/data

- sonarqube\_logs:/opt/sonarqube/logs

- sonarqube\_extensions:/opt/sonarqube/extensions

volumes:

sonarqube\_data:

sonarqube\_logs:

sonarqube\_extensions:

# sudo docker-compose up –d

### ****🖥 Access SonarQube Dashboard****

**In your browser:**  
**http://13.61.0.230:9000**

 Username: admin

 Password: admin → Change password after login.

# Create a SonarQube Project & Token

### ****Create a Token****

1. Go to **My Account** → **Security** → Generate a new token (e.g., github-token)
2. Save this token securely.

# Steps to Follow:

## 1. Prepare the Environment

· Install or access a SonarQube server (can be local via Docker or on a shared remote instance).  
· Set up a SonarQube Token for authentication.

## 2. Create a Sample Project

· Initialize a Git repository with a basic Java/Python/JS project.  
· Add at least one class/module with simple logic.  
· Include minimal test cases if possible.

## 3. Configure SonarQube

· Create a new project in the SonarQube dashboard.  
· Configure the project settings to get the projectKey and token.

## 4. Add SonarQube Scanner Configuration

· Add a configuration file (sonar-project.properties) in the root of your project.  
· Fill it with necessary fields like sonar.projectKey, sonar.host.url, and sonar.login.

### Sample sonar-project.properties

sonar.projectKey=js-sonarqube-demo  
sonar.projectName=JS SonarQube Demo  
sonar.sources=.  
sonar.exclusions=\*\*/node\_modules/\*\*  
sonar.tests=.  
sonar.test.inclusions=\*\*/\*.test.js  
sonar.javascript.lcov.reportPaths=coverage/lcov.info  
sonar.host.url=http://16.170.221.76:9000  
sonar.login=${SONAR\_TOKEN}

## 5. Set Up GitHub Actions Workflow

Create a file named `.github/workflows/sonarqube.yml` in your repo with the following content:

### GitHub Actions Workflow File

name: SonarQube Analysis  
  
on:  
 push:  
 branches:  
 - main  
  
jobs:  
 sonarqube:  
 runs-on: ubuntu-latest  
  
 steps:  
 - name: Checkout code  
 uses: actions/checkout@v3  
  
 - name: Set up Node.js  
 uses: actions/setup-node@v3  
 with:  
 node-version: '18'  
  
 - name: Install dependencies  
 run: npm install  
  
 - name: Fix jest executable permission  
 run: chmod +x node\_modules/.bin/jest  
  
 - name: Run tests and generate coverage  
 run: ./node\_modules/.bin/jest --coverage  
  
 - name: Download SonarQube scanner  
 run: |  
 wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-5.0.1.3006-linux.zip  
 unzip sonar-scanner-cli-5.0.1.3006-linux.zip  
  
 - name: Run SonarQube scanner  
 env:  
 SONAR\_TOKEN: ${{ secrets.SONAR\_TOKEN }}  
 run: |  
 ./sonar-scanner-5.0.1.3006-linux/bin/sonar-scanner \  
 -Dsonar.projectKey=js-sonarqube-demo \  
 -Dsonar.sources=. \  
 -Dsonar.tests=. \  
 -Dsonar.test.inclusions=\*\*/\*.test.js \  
 -Dsonar.javascript.lcov.reportPaths=coverage/lcov.info \  
 -Dsonar.host.url=http://16.170.221.76:9000 \  
 -Dsonar.login=$SONAR\_TOKEN

## 6. Run the Workflow

Push your changes to GitHub to trigger the workflow automatically.  
Verify that GitHub Actions runs and connects with SonarQube.

## 7. Verify the Report

After a successful workflow run, open the SonarQube dashboard and check for:  
· Bugs  
· Code Smells  
· Test Coverage  
· Maintainability Score

## 8. Submit

· Share the GitHub repository link.  
· Share a screenshot of the SonarQube report dashboard.

## Additional Notes

If you encounter `Permission denied` errors running Jest in GitHub Actions, add a step to fix permissions as shown above.  
Use `chmod +x node\_modules/.bin/jest` before running Jest.  
Run Jest tests using `./node\_modules/.bin/jest --coverage` to avoid path issues.