Testing in DevOps

DOu – Certified Tester in DevOps (CTD)
Exercise Solutions

HO-2.4.3(HO-0)
Exercise - Demonstrate how to integrate a code performance measurement tool into a DevOps pipeline

- Add code analysis stage in jenkins pipeline and trigger the jenkins build. Discuss code performance reports.

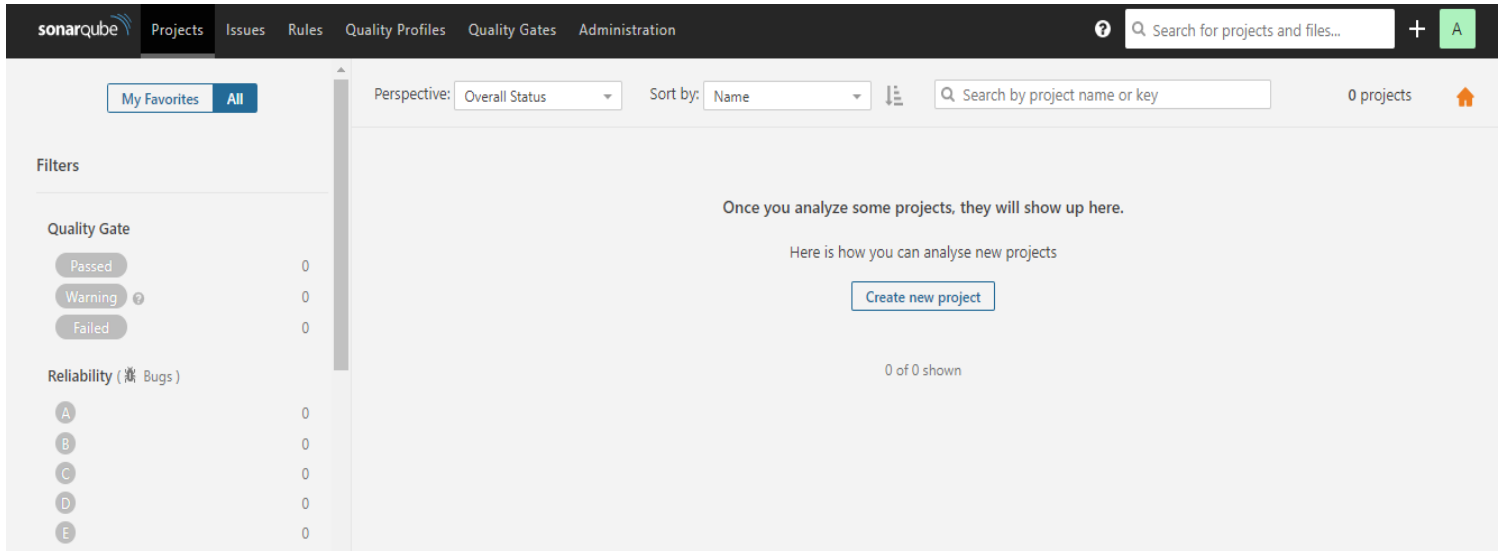
HO-2.4.3(HO-0)
SonarQube Setup

- Create & Launch AWS Ubuntu 18.04 instance(t2.small, 8GB HD)with all traffic

  (Refer to “Testing in DevOps_Exercise-Solutions_Pre-requisite_V0.1” slide-deck)
- Connect with putty
- Run below commands on terminal
  - sudo apt-get update
  - sudo apt-get install docker.io
  - docker –version
  - sudo docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
  - sudo docker images
  - sudo docker container ls -a
- Launch <http://your aws ip address:9000> on browser
- SonarQube UI will be displayed

HO-2.4.3(HO-0)
SonarQube Setup

- Do login(Username → admin, Password → admin)



HO-2.4.3(HO-0)
Generate SonarQube token

- Go to Sonarqube UI dashboard
- Go to Administration tab at top bar
- Go to Security
- Go to Users
- Click Update token icon
- Enter token name
- Click Generate
- Copy the sonarqube token

  E.g. 6263fc7b5f02a54a92b4ab0331ccce312dcd1adf

  Note: you will need to use this token in Jenkins later
- Click Done
- Go to Sonarqube dashboard by clicking on sonarQube logo link at left top section

HO-2.4.3(HO-0)
Setup SonarQube token in Jenkins

- Login to Jenkins UI
- Go to Manage Jenkins
- Go to Manage Credentials
- Click global under "Stores scoped to Jenkins" section
- Click Add Credentials
- Select Kind --> Secret text
- Select Scope --> Global
- Write SonarQube token under Secret section
- Click OK

HO-2.4.3(HO-0)
Setup SonarQube scanner plugin in Jenkins

- Go to Jenkins Dashboard
- Go to Manage Jenkins
- Go to Manage Plugins
- Go to Available Plugins section
- Search “SonarQube plugin”
- Install "SonarQube Scanner for Jenkins“
- Select Restart checkbox at bottom
- After jenkins restarted, login to it

HO-2.4.3(HO-0)
Setup SonarQube Server in Jenkins

- Go to Jenkins Dashboard
- Go to Manage Jenkins
- Go to Configure System
- Go to SonarQube servers section
- Add SonarQube
- Fill following details:
  - Select "Enable injection of SonarQube server configuration as build environment variable“ checkbox
  - Name - My SonarQube Server
  - Server URL - <http://aws ubuntu sonarqube machine ip address:9000>
    e.g. http://192.168.100.47:9000
  - Select “Secret text” from list under Server authentication token section
- Click Apply & Save

HO-2.4.3(HO-0)
Setup SonarQube Scanner in Jenkins

- Go to Jenkins Dashboard
- Go to Manage Jenkins
- Go to Global Tool Configuration
- Go to SonarQube Scanner section
- Add SonarQube Scanner
- Fill following details:
  - Name → sonar-scanner
  - SONAR_RUNNER_HOME → Select “install automatically” checkbox
- Click Apply & Save

HO-2.4.3(HO-0)
Exercise Solution - Demonstrate how to integrate a code performance measurement tool into a DevOps pipeline

- Make sure you are already logged in Jenkins and be on Dashboard UI
- Make sure “sonar-project.properties” file is added to “verity-devops-ex” github repo

  Note: In “sonar-project.properties” file, make sure project name(e.g. verity-devops-ex) should be same as jenkins project name in sonar.sources, sonar.java.tests, sonar.java.binaries path

```
sonar.sources=/var/jenkins_home/workspace/verity-devops-ex/src/main/java/
sonar.java.tests=/var/jenkins_home/workspace/verity-devops-ex/src/test/java
sonar.java.binaries=/var/jenkins_home/workspace/verity-devops-ex/target/classes
```

- Add Code Analysis stage in Jenkinsfile.txt on local machine:
  - Clean(already added)
  - PMD(already added)
  - Compile(already added)
  - **Code Analysis**

HO-2.4.3(HO-0)
Add Stage – Code Analysis

1. **Code Analysis** defines a stage that appears on the Jenkins UI

```
//Code starts for stage Code Analysis
                stage('Code Analysis') {        (1)
            steps {
                script  {
                        scannerHome = tool 'sonar-scanner'
                        }
                        withSonarQubeEnv('My SonarQube Server')
                        {
                        sh "${scannerHome}/bin/sonar-scanner"
                        }
                }
                                                }
//Code ends for stage Code Analysis
```

HO-2.4.3(HO-0)
Exercise Solution - Demonstrate how to integrate a code performance measurement tool into a DevOps pipeline

- Saving Jenkinsfile.txt & Commit it on GitHub
- Running The Pipeline
  - Go to Jenkins Dashboard
  - Build will be triggered automatically because of webhook
  - Click Build no. link
  - Click Console Output link
  - Verify the build result is successful
  - Verify sonarqube reports on sonarqube web ui