**DevOps**

**Day 1**

**Activity 4: Customizing Quality Gates configuration in Jenkins**

Table of Contents

[Table of Contents 2](#_Toc472976068)

[Activity: Customizing Quality Gates configuration in Jenkins 3](#_Toc472976069)

[Access SonarQube and explore 4](#_Toc472976070)

[Configure Quality Gates conditions and assign to project 7](#_Toc472976071)

[Cucumber Test Reports 9](#_Toc472976072)

Activity: Customizing Quality Gates configuration in Jenkins

The list of steps to be performed in this activity is given below:

|  | **Task Description** | **Target Date for Completion** | **Supporting Documents/Personnel** |
| --- | --- | --- | --- |
|  | Access SonarQube and explore | Day1 | Participants |
|  | Configure Quality Gates conditions and assign to project | Day 1 | Participants |
|  | Cucumber Test Reports | Day 1 | Participants |

**Activity Duration:** 45 minutes

**Activity Description:**

Activity Type: Individual

Purpose:

**Technical Overview:**

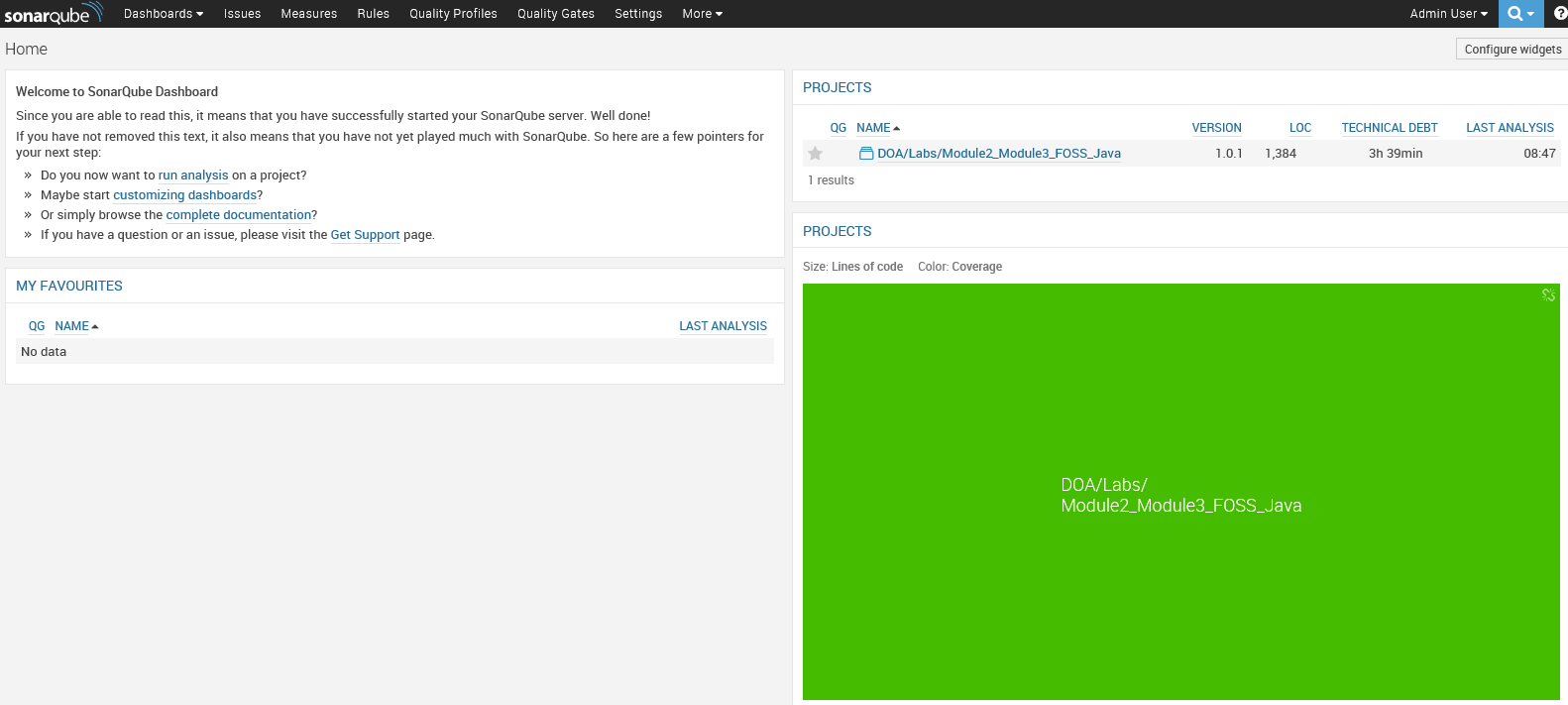
**Activity Inputs/Templates:** NA

**Activity Instructions:**

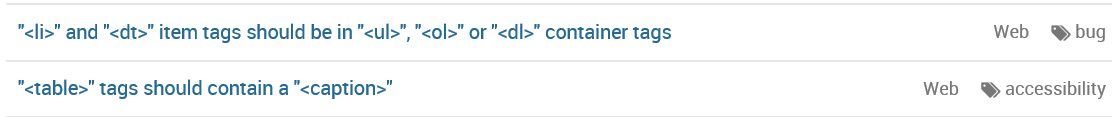
Access SonarQube and explore

Follow the below steps for accessing SonarQube.

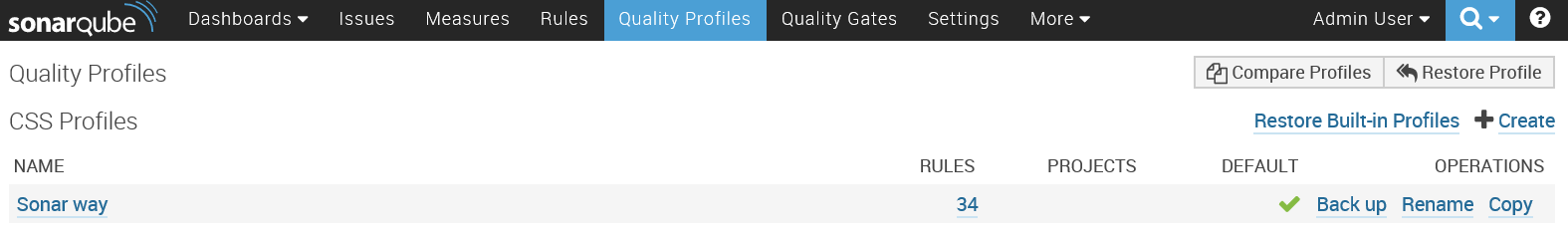
1. Navigate to ADOP home page and click on SonarQube. You should be directed to the SonarQube dashboard which looks like as shown below



1. Click on “Rules” link to view the list of preconfigured rules specific to code analysis. Few rules specific to web page defined are:



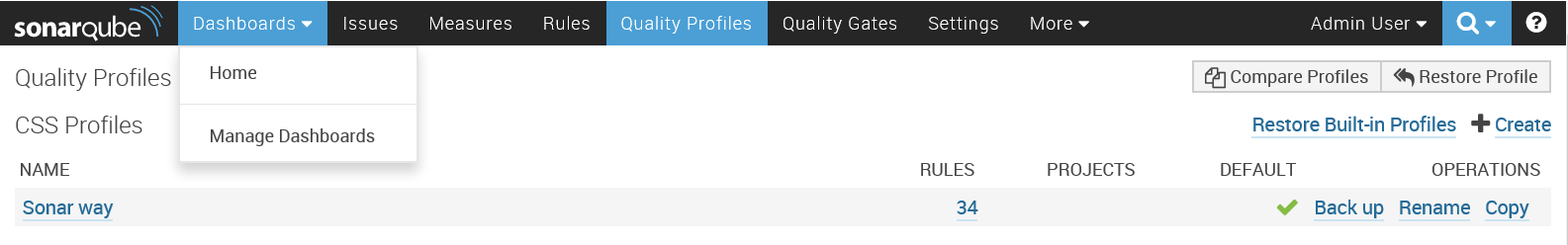
1. To add/modify or view existing rules, navigate to the 'Quality Profiles' page



* + Click on the profile name and rules as shown above to view the breakdown of all the rules set specific to that particular profile

**View Sonar Analysis report of a project:**

1. Click on Dashboards menu and click on Home link to navigate to home page of SonarQube.

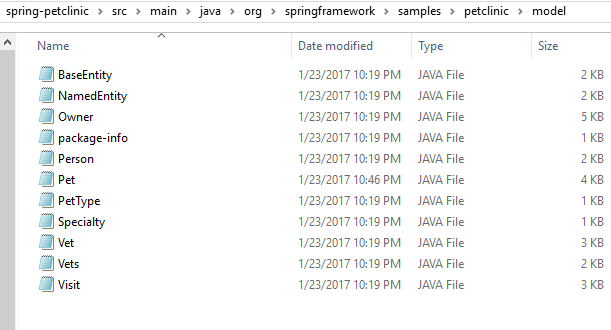


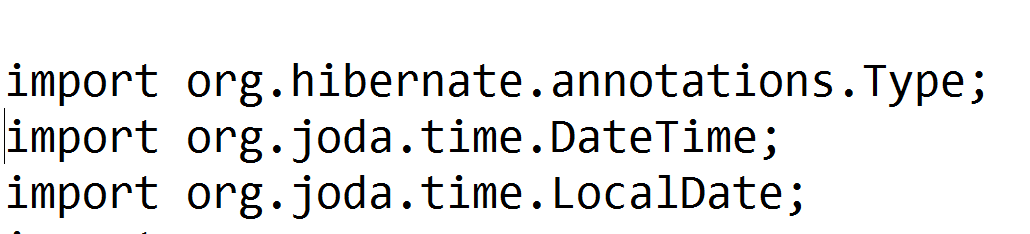
1. On the right pane, clink on the project link,

**Project name:** DOA/Labs/Module2\_Module3\_FOSS\_Java

* 1. Study the report for issues and settings for the project
  2. You will observe that there are
     1. 2 critical
     2. 15 major and
     3. 7 minor issues
  3. Click on number of issues to view the details about the issue such as in which file and line issue exists and recommended solution to resolve this issue.

**Fix a minor issue:**

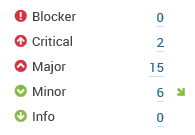
1. Open Pet.java file in notepad which exists in the path Spring-petclinic -> src -> main ->java ->org ->springframework -> samples -> petclinic -> model
2. Remove the below highlighted line from Pet.java and save.



1. Commit the changes and push the change by executing the following commands in git bash.

* git checkout develop
* git add .
* git commit -m “Removed unused import”
* git push –set-upstream origin develop
* git checkout master
* git merge develop
* git push

1. Watch the pipeline execution and after “Reference\_Application\_Code\_Analysis” build job execution completed successfully, navigate to Sonar Qube to find out the number of minor issues as shown below

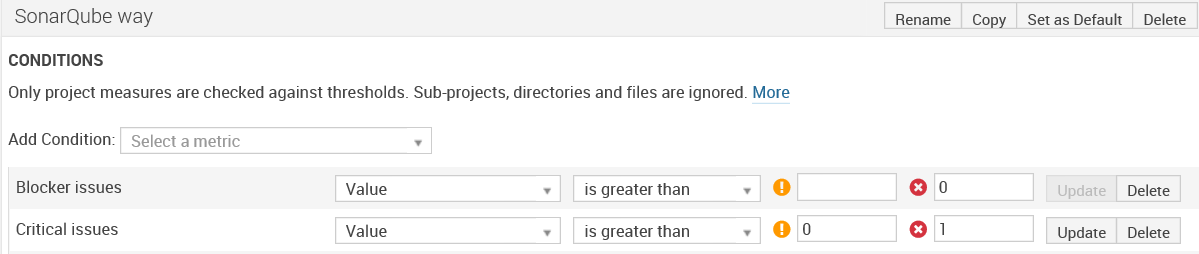


Configure Quality Gates conditions and assign to project

Though there where many issues exist, still “Reference\_Application\_Code\_Analysis” build job has not failed in the pipeline execution. In order to make you to be notified about the critical issues by making build job to be failed, follow the below steps:

**Defining Quality Gate Condition**

1. Click on “Quality Gates” link in the top menu bar.

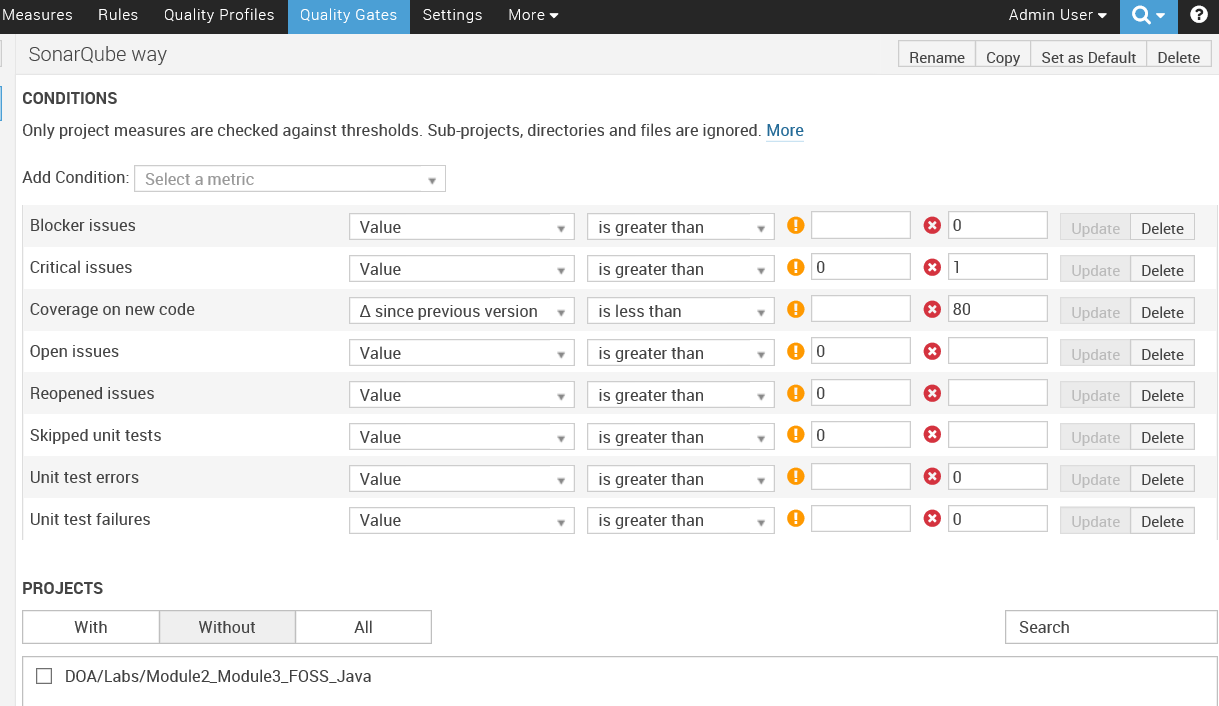


1. Modify the Critical issues condition as highlighted above and click on update button.

Critical Issues 🡪 Value🡪 is greater than 🡪 0 and 1 in error threshold field

**Assigning project to the defined quality gate**

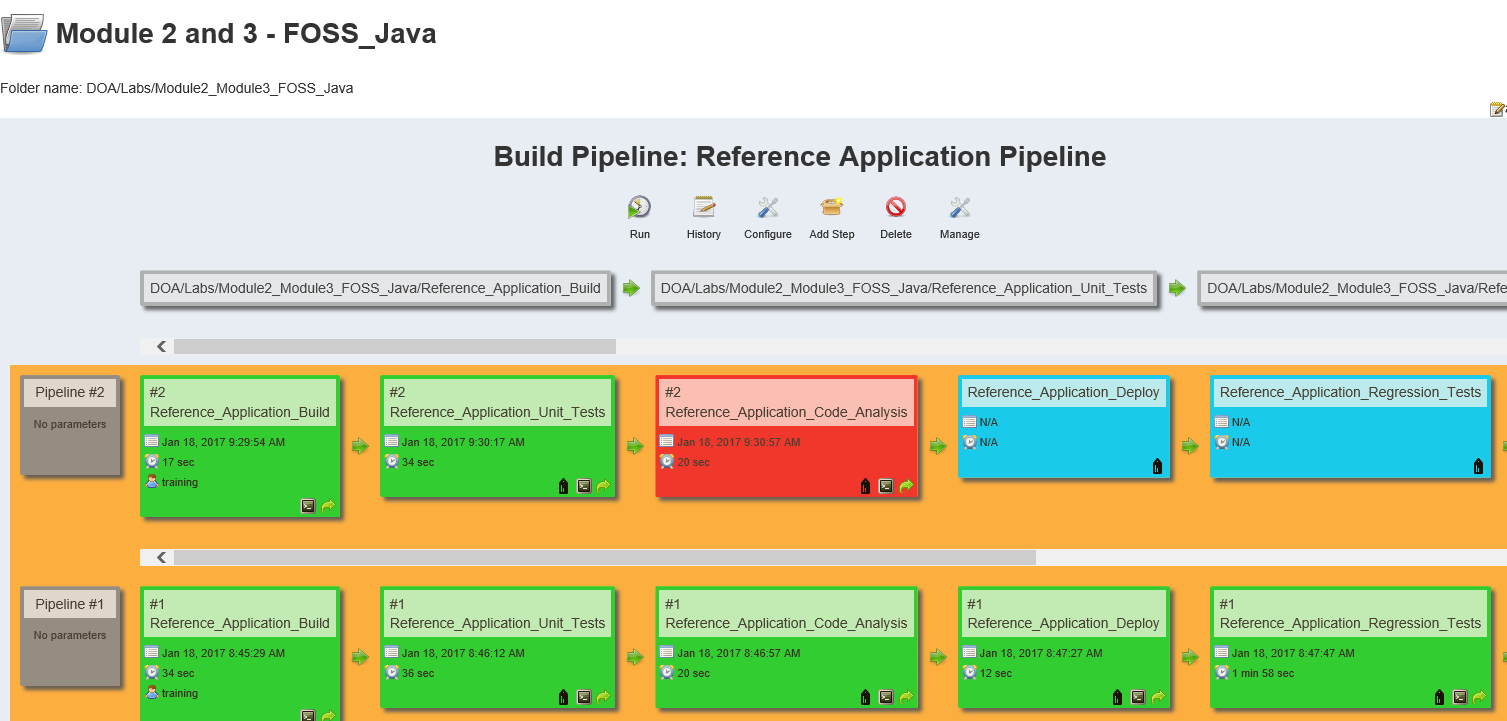
1. Navigate to the “Without” tab of the “PROJECTS” section in Quality Gates page and click on the check box next to the project name as shown below



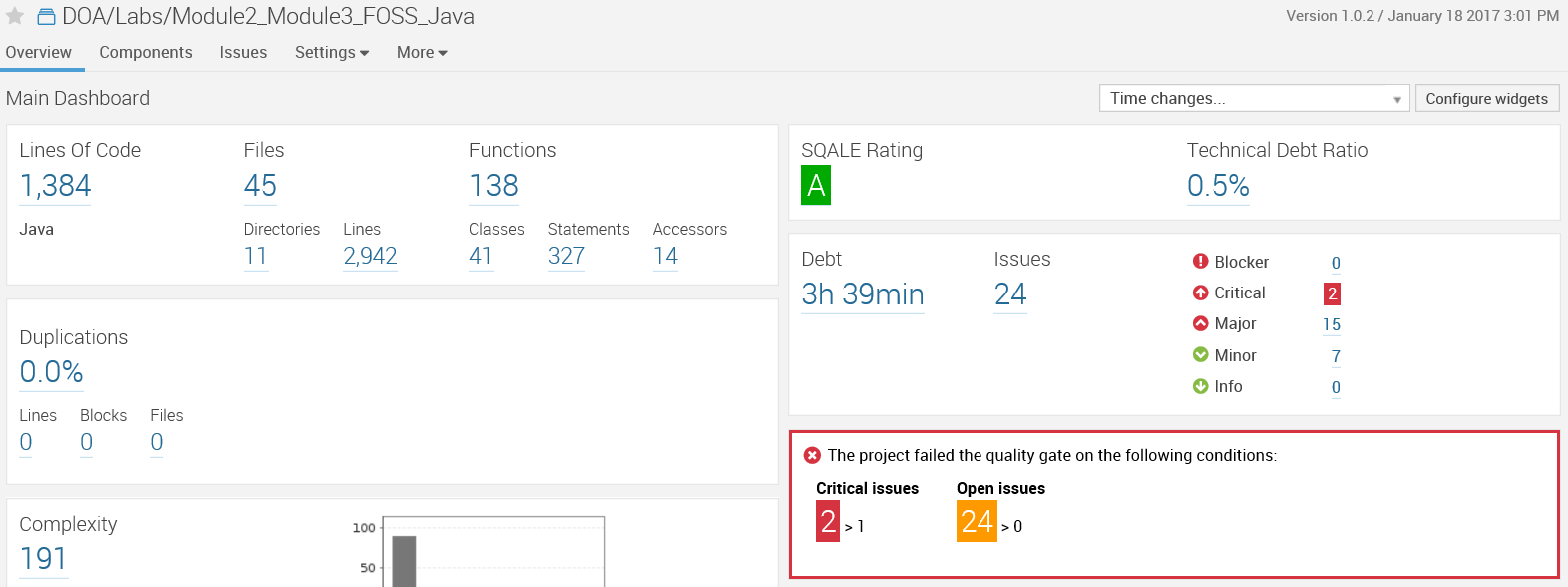
1. The above step helps to assign the quality gate condition to the selected project and now selected project is moved from without category to with category.

**Watch pipeline execution:**

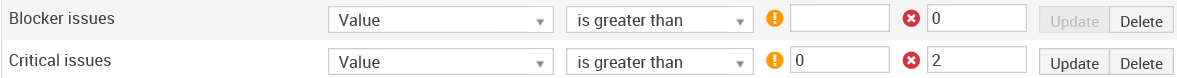
1. Open the Jenkins Dashboard page to watch the pipeline execution.
2. Navigate to DOA/Labs/Module 2 and 3 – FOSS\_JAVA
3. Select Java Reference Application tab (Note: In the last pipeline execution instance, all the build job execution was success though there were issues).



1. Click on “Run” and notice the failure of “Reference\_Application\_Code\_Analysis” build job.
2. Navigate to SonarQube project page and observe the project failure because of the quality gate condition as highlighted below:

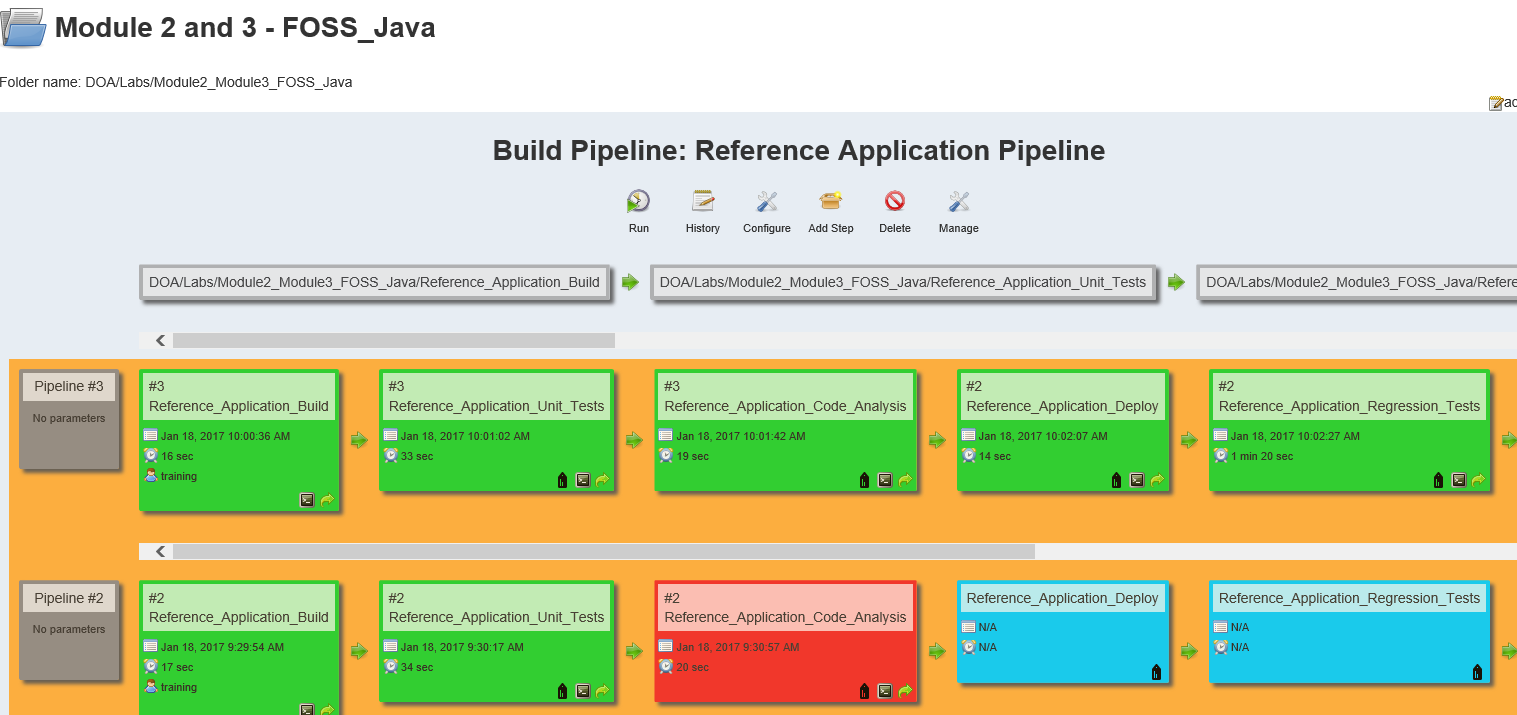


1. You can update the quality gate condition again as shown below and trigger & watch the pipeline execution. Now all the build job execution will be success.



Cucumber Test Reports

1. From the Jenkins job pipeline view, click on the "Reference\_Application\_Regression\_Tests“ for a successful step (green colored)

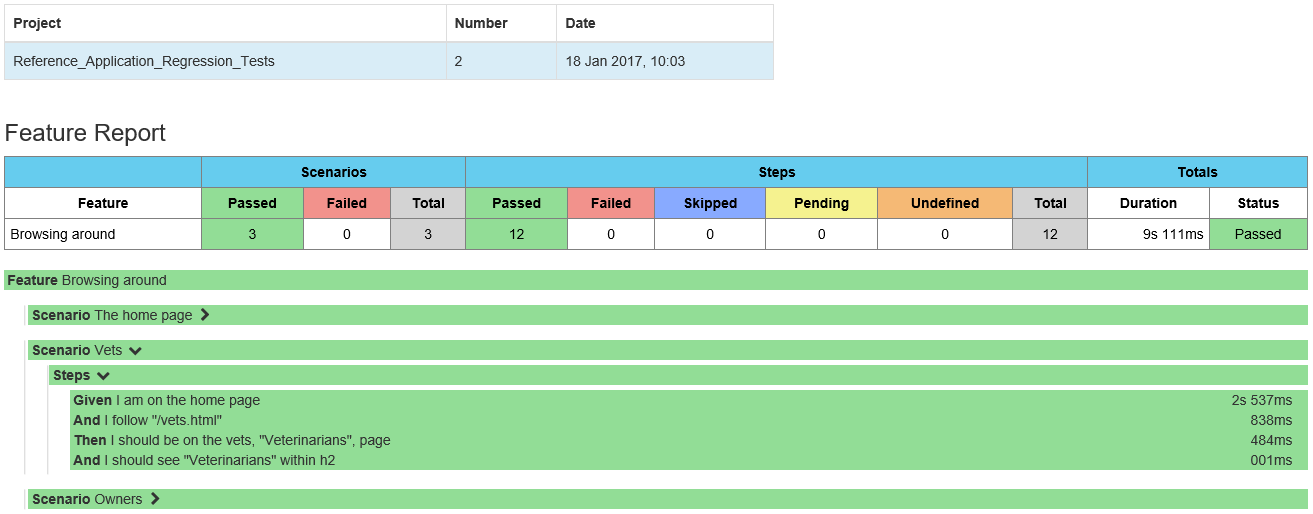


1. Click on the "Cucumber-JVM Reports" link
2. Click on features and observe it

For an example:

* 1. Click on “Browsing Around” feature, then select “vets” scenario, and follow the steps in the deployed application as shown below:
  2. Since these steps are result in right action the cucumber test case is pass.

You can try to do the steps manually to verify the same.



**Make Cucumber Test Fail:**

1. Open the file spring-petclinic\src \main\webapp\WEB-INF\jsp\vets\vetList.jsp file in spring-petclinic project which was cloned during Day 1- Activity 3.
2. Change line number 18 from: (This change will make cucumber feature to fail).

<h2>**Veterinarians**</h2>

to:

<h2>**Vets**</h2>

1. Push the change which cause the pipeline execution to be triggered:
   1. Switch to develop branch if you are in master branch using “git checkout develop” command
      1. git add .
      2. git commit -m "Checking if unit tests are intact"
      3. git push
   2. Switch to master branch
      1. git checkout master
      2. git merge develop
      3. git push
2. Watch out the result in cucumber test report.