Jenkins and Gatling

|  |  |
| --- | --- |
| Author: | Mohammed Sultan |
| Document Name: | Jenkins and Gatling |
| Document Location: | \\lukpc210\Projects\Product Projects\Operations\Vitic 2017.1\Project Documents\POC\VXRW-357 Integrate Gatling with Jenkins.docx |
|  |
| Date Created: | 24/03/2017 |
| Date Last Modified: | 24/03/2017 |
| Status: | Draft |
| Confidentiality: | <Select/enter confidentiality status> |
| Circulation: | <Detail circulation list> |
| Document Version: | 0.5 |

Contents

[1. Overview 2](#_Toc478133691)

[2. Jenkins 3](#_Toc478133692)

[2.1 Vitic\_Performance\_Master-0.0.1-SCM 3](#_Toc478133693)

[2.2 Security 4](#_Toc478133694)

[3. Gatling 4](#_Toc478133695)

[4. Amendment Record 8](#_Toc478133696)

# Overview

This document details the working configurations and lessons learned regarding Jenkins and Gatling.

# Jenkins

Although there is already an instance of Jenkins in Lhasa, a separate instance was installed for the Vitic PoC project. This was installed as a vanilla installation with the default set of plugins. Additional plugins that were installed afterwards were:

* NodeJS Plugin: Required to run the Angular 2 unit tests
* SonarQube Plugin: Required for SonarQube integration

Three jobs have been configured in Jenkins which are able to perform a build from a Git repository. The first two are standard jobs and the third is a pipeline job.

Now, we will add new job dedicated for performance measuring. Two plugins are needed to be installed.

1. **Gatling plugin**: integrates [Gatling](http://gatling-tool.org/), an Open Source stress tool, with Jenkins.

This plugin allows you to:

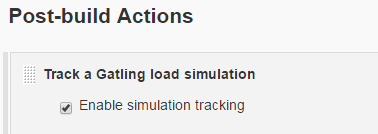
* Keep track of a Gatling simulation, providing performance trends across builds
* Publish detailed reports for each build

1. **HTML Publisher Plugin:**

Needed by Gatling plugin in order to be able to present reports in HTML format.

## 2.1 Vitic\_Performance\_Master-0.0.1-SCM

Used to perform a build of the backend project on the master branch only. This is done via the “Branch Specifier”.

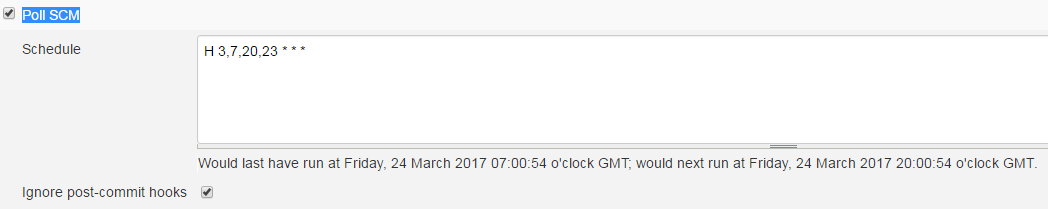
1. After installing Gatling plugin, new option is added in the Post-build Actions called Track a Gatling load simulation. Just enable it.
2. Maven integration

To get the project integrated with Jenkins you need to run the

mvn clean install

then

mvn gatling:execute -Dgatling.simulationClass=org.lhasalimited.vitic.performance.test.SmilesSearchTest -Dfeeder=Exact -DrampUsers=4000

It is recommended to run them as windows batch command to avoid pom location problem and other file reference problems.

1. Build Trigger: in order not to be triggered when merging to master, just tick in the **Ignore post-commit hooks** with updating the reasonable schedule time

## 2.2 Security

Jenkins introduced a security change that breaks Gatling reports display. the graph will not be displayed and you will received error message:

Blocked script execution in 'http://vx.../25/gatling/report/.../' because the document's frame is sandboxed and the 'allow-scripts' permission is not set.

Configuring Content Security Policy:

Two ways to disable it.

1. Temporarily relaxing Content Security Policy To change default Content Security Policy go to Manage Jenkins -> Script Console and type into console the following commands:

System.clearProperty("hudson.model.DirectoryBrowserSupport.CSP");

System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "sandbox allow-scripts; default-src 'self'; script-src \* 'unsafe-eval'; img-src \*; style-src \* 'unsafe-inline'; font-src \*");

1. Permanently relaxing Content Security Policy adding setting of hudson.model.DirectoryBrowserSupport.CSP to it

On Windows there may be a file called jenkins.xml in the Jenkins installation where this can be added to the arguments tag:

**<**arguments**>**

**-**Xrs **-**Xmx256m **-**Dhudson.lifecycle**=**hudson.lifecycle.WindowsServiceLifecycle

"-Dhudson.model.DirectoryBrowserSupport.CSP=sandbox allow-scripts; style-src 'unsafe-inline' \*;script-src 'unsafe-inline' \*;"

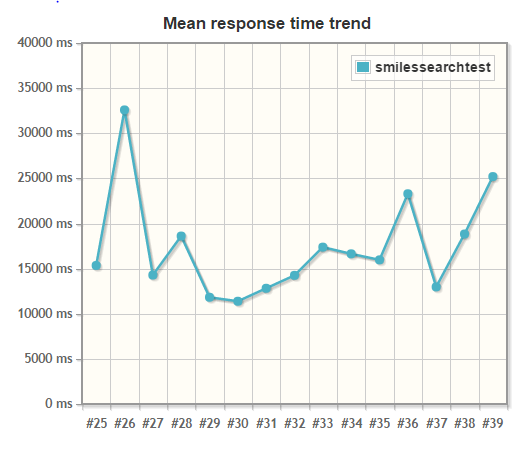
**-**jar "%BASE%\jenkins.war" **--**httpPort**=**8080

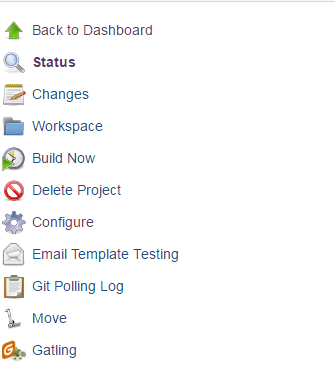
**<**/arguments>

# Gatling

As soon as you've properly configured your job and launched a build, you'll see two changes on your project dashboard:

A new entry will be available in the left summary: Gatling.

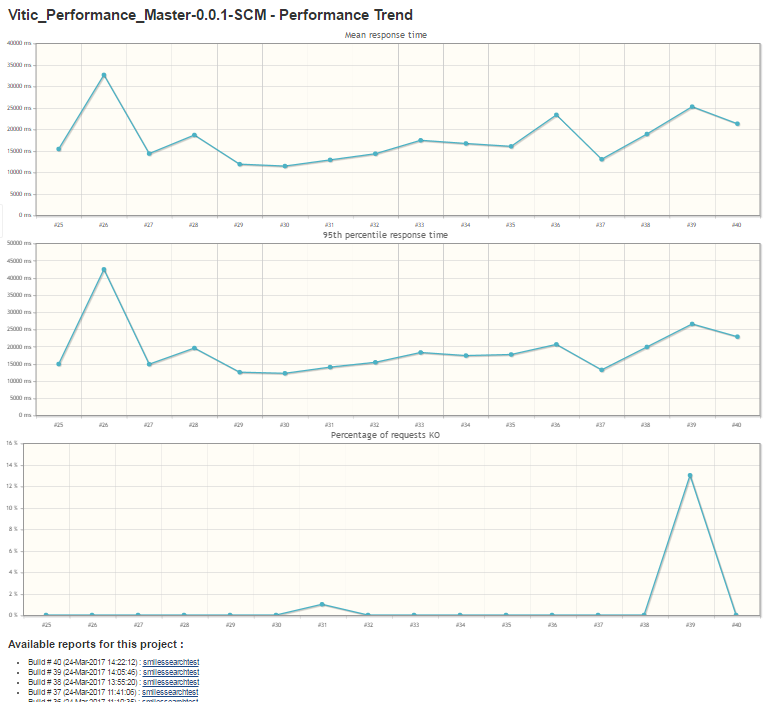
A graph, displaying the mean response time trend of your last 15 builds, will appear.



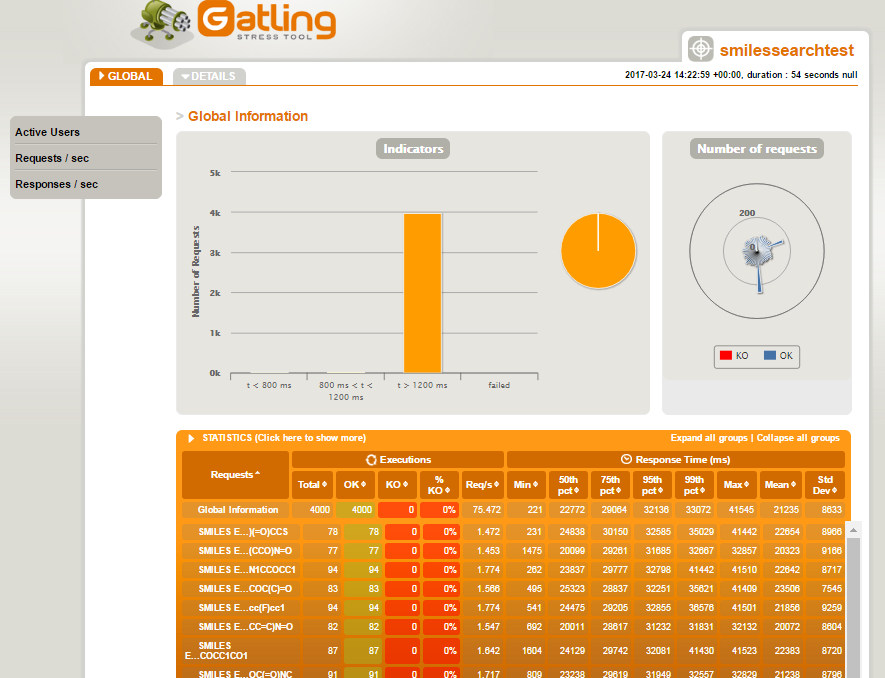
The Gatling entry in the left summary has two purposes, depending on which page you are.

If you are on the project dashboard, clicking on **Gatling** will get you to a more detailed performance trend, displaying for your last 30 builds:

* Mean response time trend
* 95th percentiles response time trend
* Percentage of KO requests



This page will also provide links to detailed reports for all your builds, at the bottom of the page or you can just click on the point in the graph for a specific build



# Amendment Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Amendment Details | Changed/Approved by |
| 24-mar-2017 | 0.1 | Initial version | MS |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |