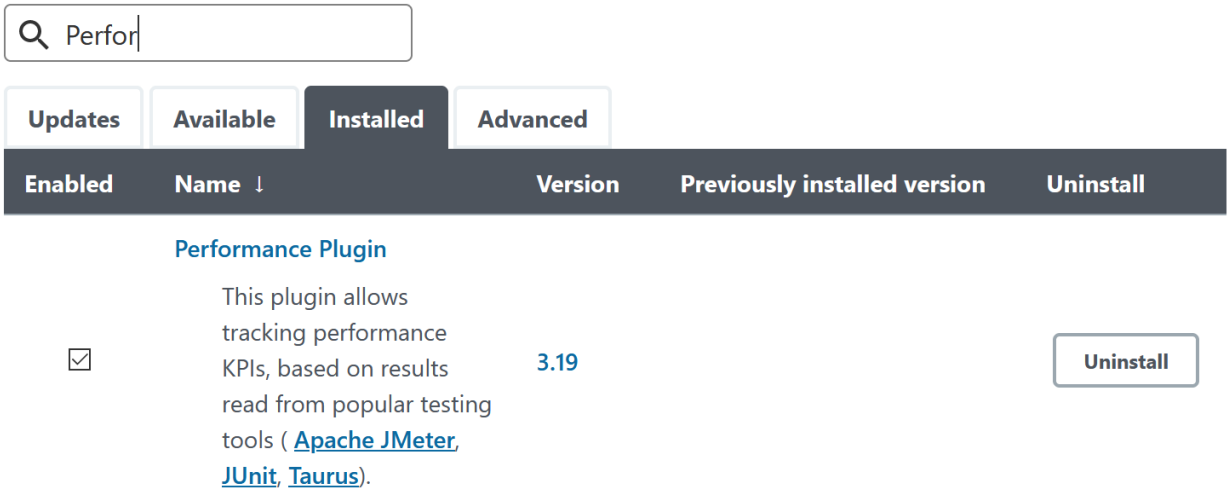
Group B
ENSE 375 - Project Step 4
April 5, 2021

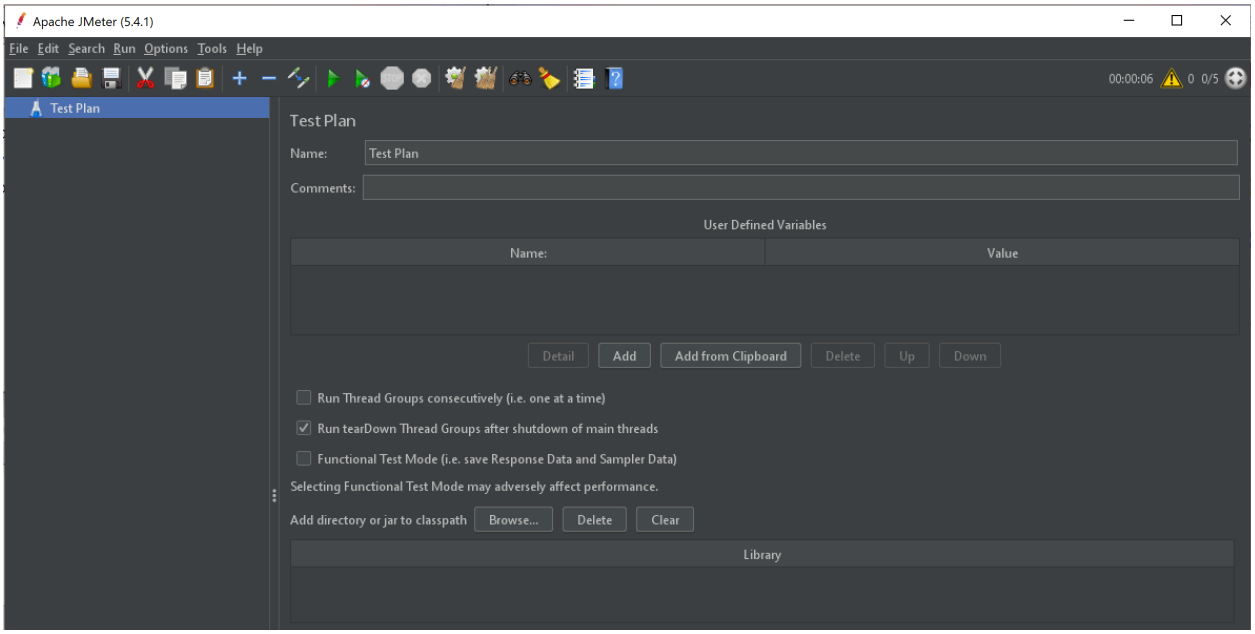
JMeter Process

Below is the step-by-step process we took to install and experiment using JMeter with Jenkins.

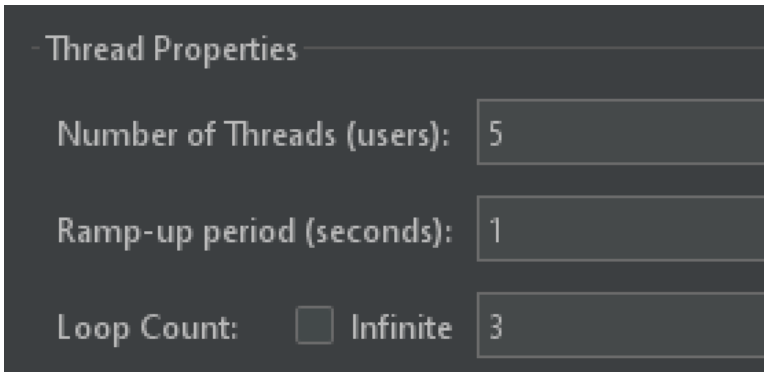
1. Install the Performance Plugin on Jenkins by going into Manage Jenkins -> Manage Plugins and searching Performance under the available plugins tab



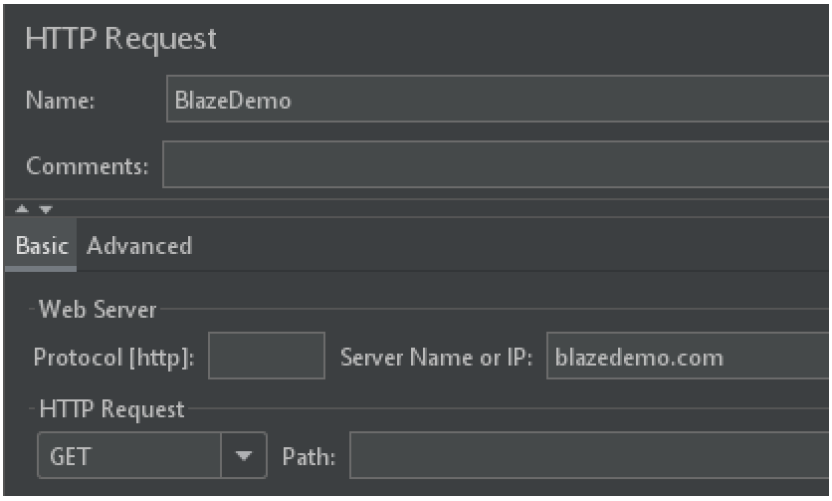
2. Download JMeter from http://jmeter.apache.org/download_jmeter.cgi
3. Unzip the JMeter folder and open the jmeter.bat file. You should see an interface similar to the one below



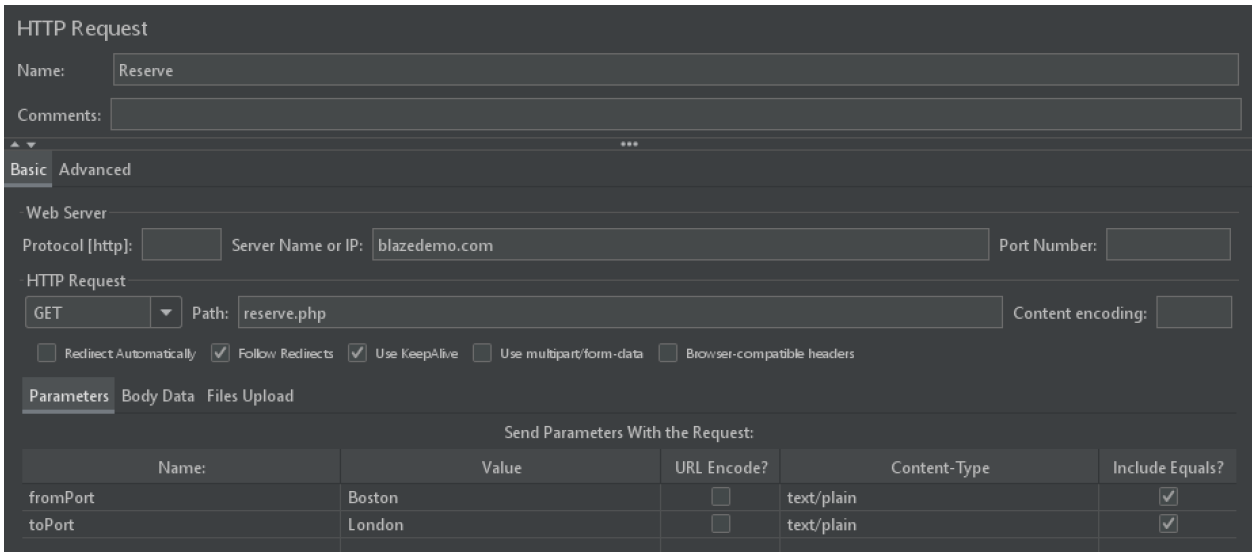
4. Add a thread group to the test plan by right clicking on the test plan and choosing Add -> Threads(Users) -> Thread Group
5. Set the following Thread Group settings
   - Number of Threads: The number of users (threads) you are testing.
   - Ramp-up Period: How quickly (in seconds) you want to add users. For example, if set to zero, all users will begin immediately.
   - Loop Count: How many times the test should repeat.



6. Add a HTTP request to a website (We used the BlazeMeter demo website - blazedemo.com) to the thread group by right clicking the thread group and choosing Add -> Sampler -> HTTP Request
7. Set the following HTTP Request settings
   - Name: Provide a custom name or simply leave it named the default “HTTP Request”.
   - Server Name or IP: The server’s DNS name or IP address. In this case - blazedemo.com.

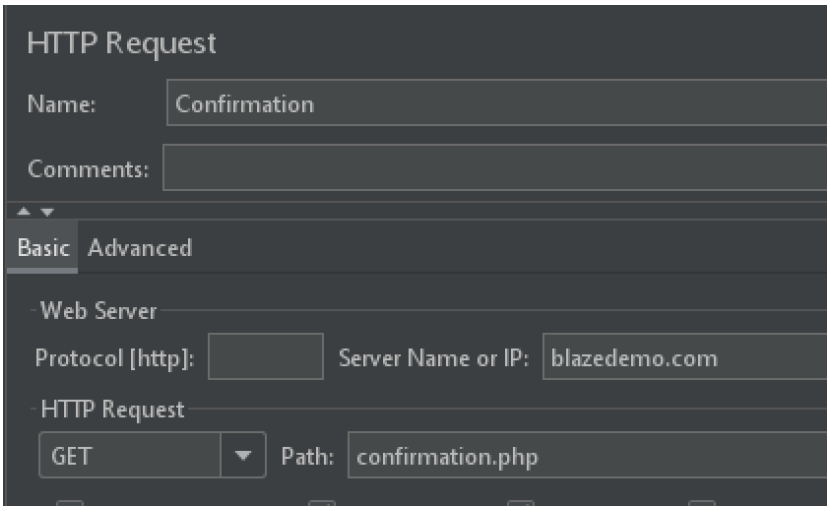


8. Add timer to HTTP Request by right clicking the HTTP Request and choosing Add -> Timer -> Constant Timer. The constant timer will determine how many milliseconds to wait between requests. The default is 300.
9. Add a listener to the thread group by right clicking the thread group and choosing Add -> Listener-> View Results Tree
10. Save the test plan and it will be saved as a .jmx file
11. Run the test plan by clicking the green arrow at the top. The result can be viewed on the listener.
12. Add another HTTP request to the thread group by right clicking the thread group and choosing Add -> Sampler -> HTTP Request. This one will test accessing the blazedemo.com/reserve.php page by entering “Boston” and “London” as parameters into the From and To selection menus.

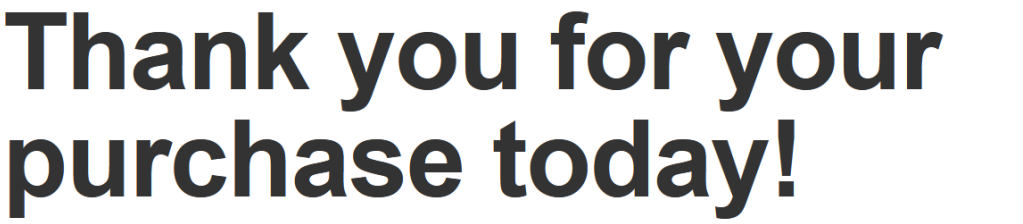


13. Add a listener to the thread group by right clicking the thread group and choosing Add -> Listener-> View Results in Table

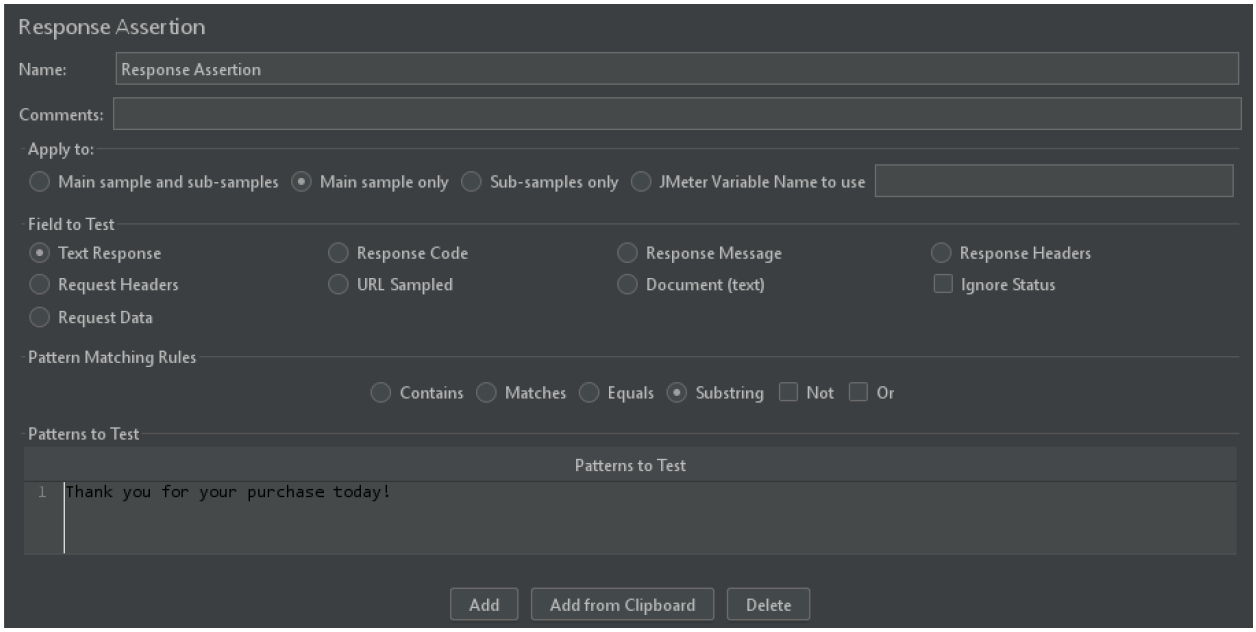
14. Add another HTTP request to the thread group by right clicking the thread group and choosing Add -> Sampler -> HTTP Request. This one will test accessing the blazedemo.com/confirmation.php page as if the user had selected and purchased a flight.



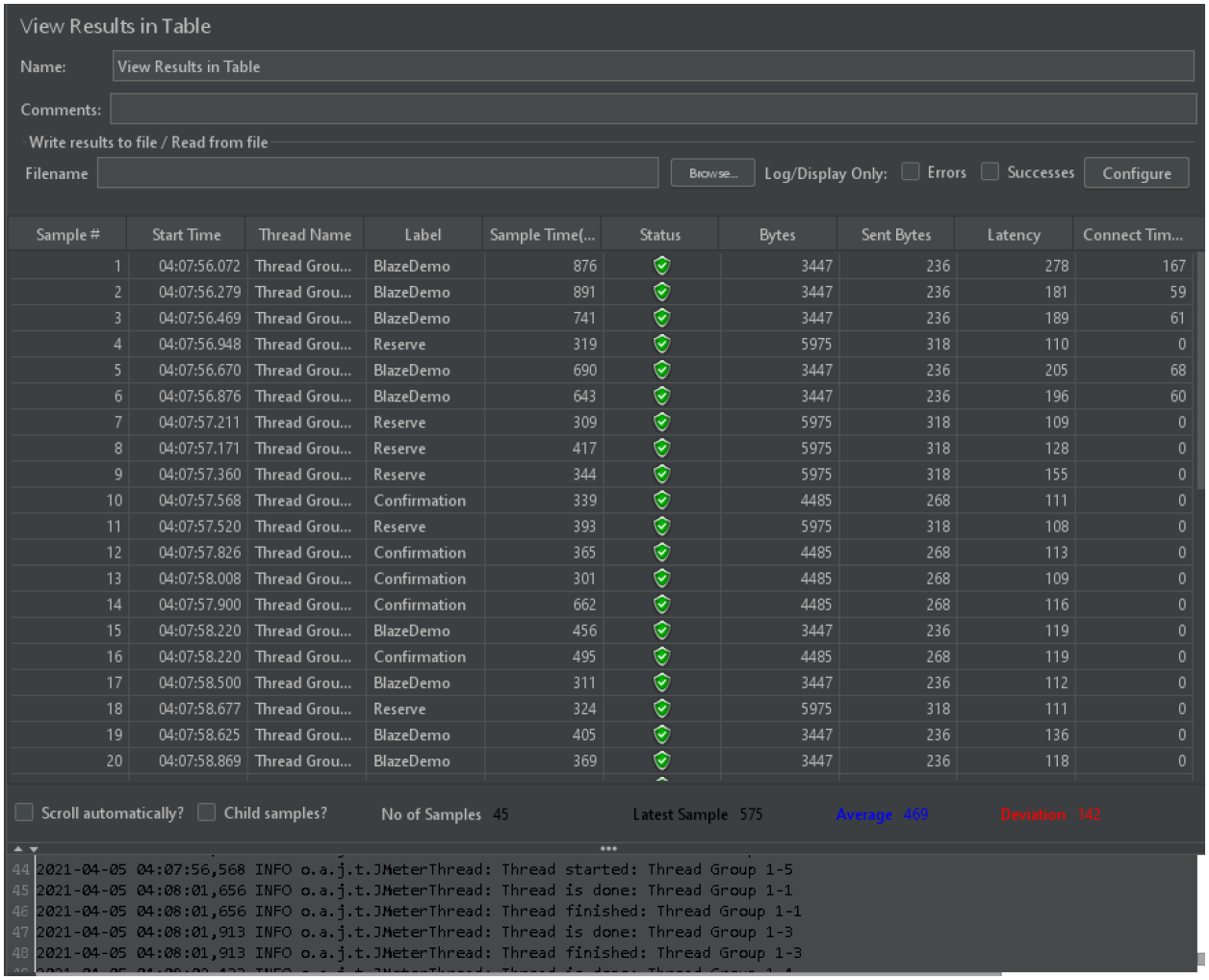
15. Add timer to HTTP Request by right clicking the HTTP Request and choosing Add -> Timer -> Constant Timer.
16. Add an assertion to the HTTP Request by right clicking the HTTP Request and choosing Add -> Assertions -> Response Assertion. An assertion will let you check for errors. In this case, we will test if the webpage displays the expected text. The confirmation page should display the following text.



Setting the following assertion settings will verify that the text is present on the screen.

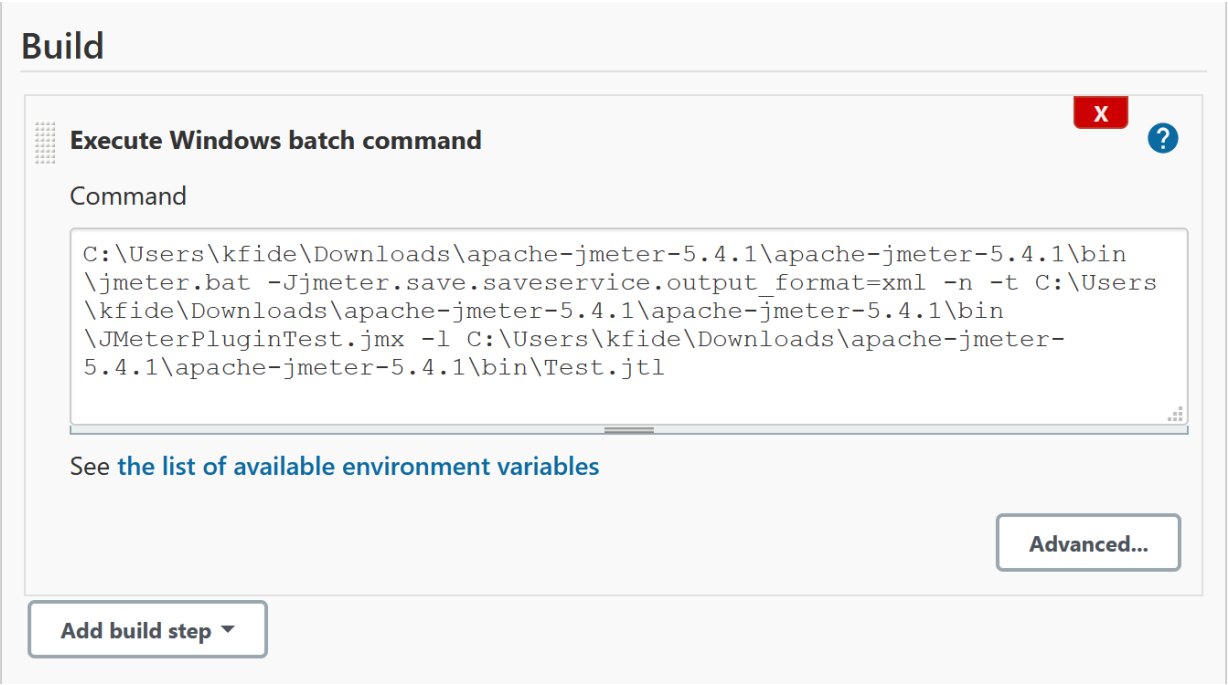


17. Save and run the test plan. The results are as follows:



Using JMeter test plan with Jenkins:

1. Add the “jmeter.save.saveservice.output_format=xml” line to the bottom of the user.properties file (this is located in /bin folder of your JMeter installation)
2. Create a freestyle project with Jenkins and in the Build section, add an “Execute Windows batch command” step. In the command area, enter the following command to pull your saved test plan .jmx file from your JMeter \bin folder as well as to save your Test.jtl file to the same folder.



3. Then add a “Publish Performance test result report” post build action which will use the Performance plugin. Set the source data to the path of your “Test.jtl” file and leave all other settings as default. This can be seen below.

Post-build Actions

Publish Performance test result report

Source data files (autodetects format):

C:\Users\kfide\Downloads\apache-jmeter-5.4.1\apache-jmeter-5.4.1\bin\Test.jtl

Regex for included samplers

Show Trend Graphs

Select evaluation mode

Expert Mode / Standard Mode

Standard Mode

Select mode:

Relative Threshold / Error Threshold

Use Error thresholds on single build:

Unstable	-1
Failed	-1

Advanced...

Use Relative thresholds for build comparison:

	(-)	(+)
Unstable % Range	-1.0	-1.0
Failed % Range	-1.0	-1.0

Compare with previous Build / Compare with Build number 0

Compare based on: Average Response Time

Expert Mode

Constraint settings

Ignore Failed Builds

Ignore Unstable Builds

Save constraint log to workspace

JUnit output file (optional)

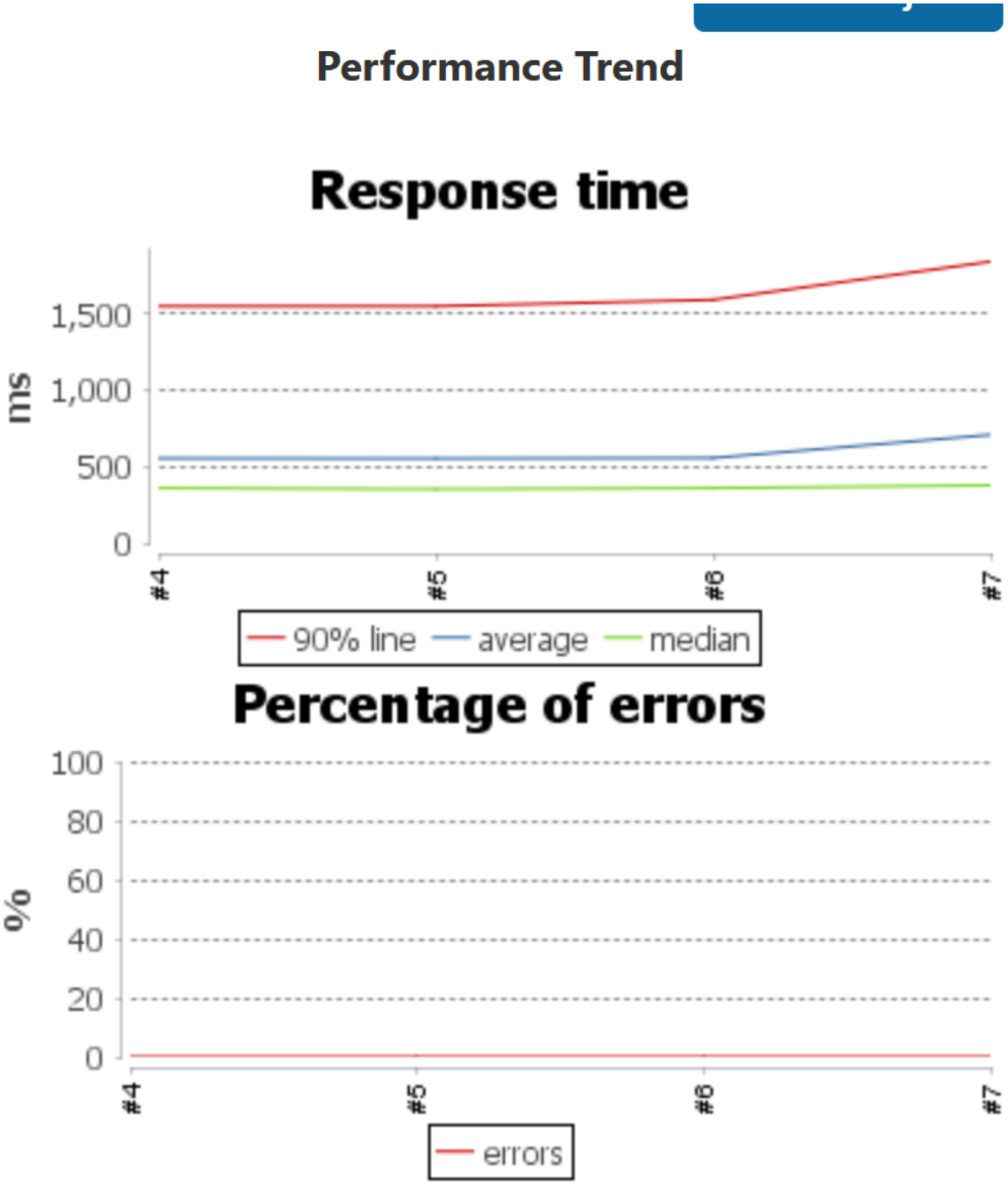
Constraints

Add a new constraint

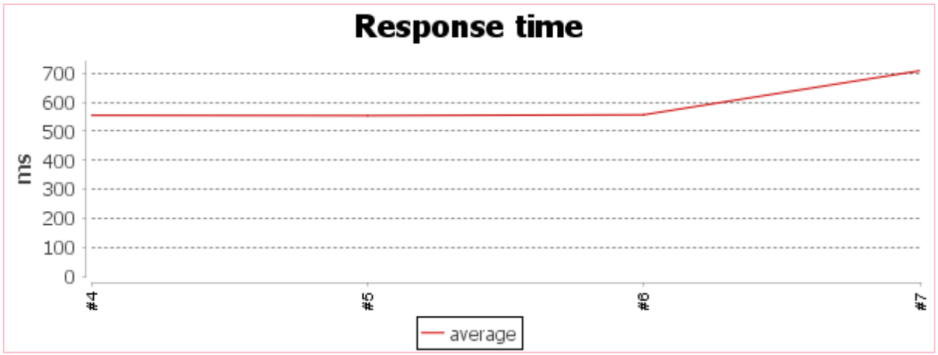
Advanced...

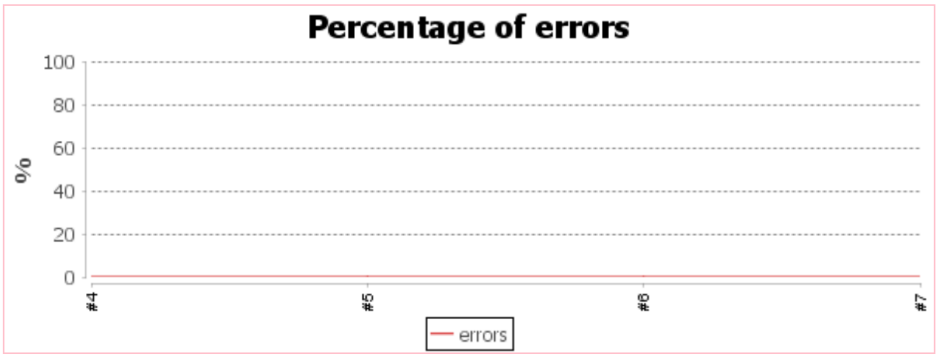
ld action

4. Save your project configuration and “Build Now”. Initially your Performance Trend charts will be empty but after a couple builds you will see results displayed. The charts display average, median and 90 percentile values along with the errors count. You will also receive .xml artifacts. Below is a screenshot of our Performance Trend charts after a few builds.



Performance Breakdown by URI: Test.jtl





[Response time trends for build: "JmeterTestProject #7"]

Comparison with previous build

URI	Samples	Average (ms)	Min(ms)	Median(ms)	Line 90.0(ms)	Max(ms)	Http Code	Errors (%)	Average (KB)	Total (KB)
BlazeDemo	75 +15	1180 +260	303 0	428 +17	2300 +58	5607 +3144	200	0.0 % 0.0 %	3.37 0.0	252.47 +50.49
Confirmation	75 +15	492 +131	297 0	348 +5	650 +232	1936 +1350	200	0.0 % 0.0 %	4.38 0.0	328.49 +65.7
Reserve	75 +15	455 +66	302 0	372 +12	627 +119	1891 +1231	200	0.0 % 0.0 %	5.83 0.0	437.62 +87.52
All URIs	**225 +45**	**709 +152**	**297 0**	**378 +18**	**1843 +250**	**5607 +3144**		**0.0 % 0.0 %**	**4.53**	**1018.58**

Our .jmx, .jtl, and our .xml artifact files have been included in our github directory.