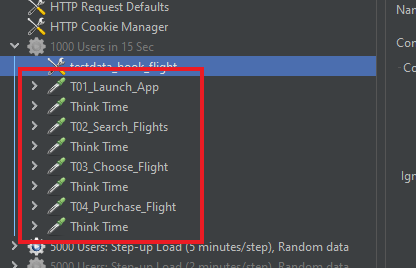
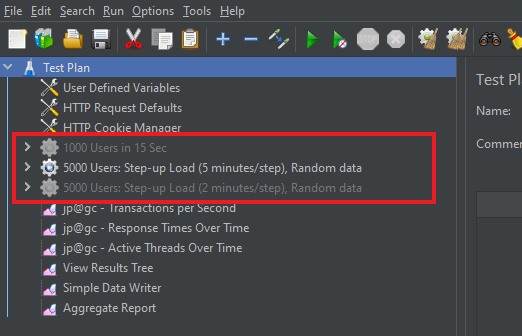
**Solution:**

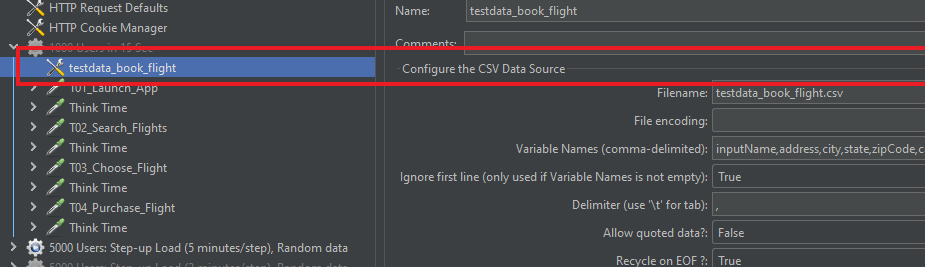
* Apache JMeter was used to create performance testing solution. Refer last section of this document for setting up JMeter in your environment.
* Load was generated against <https://blazedemo.com/> in this load testing solution
* This Test performs four actions on the demo website as shown in screenshot (from JMeter script) below.



* Script contains 3 different test plans (red box in the screenshot above)

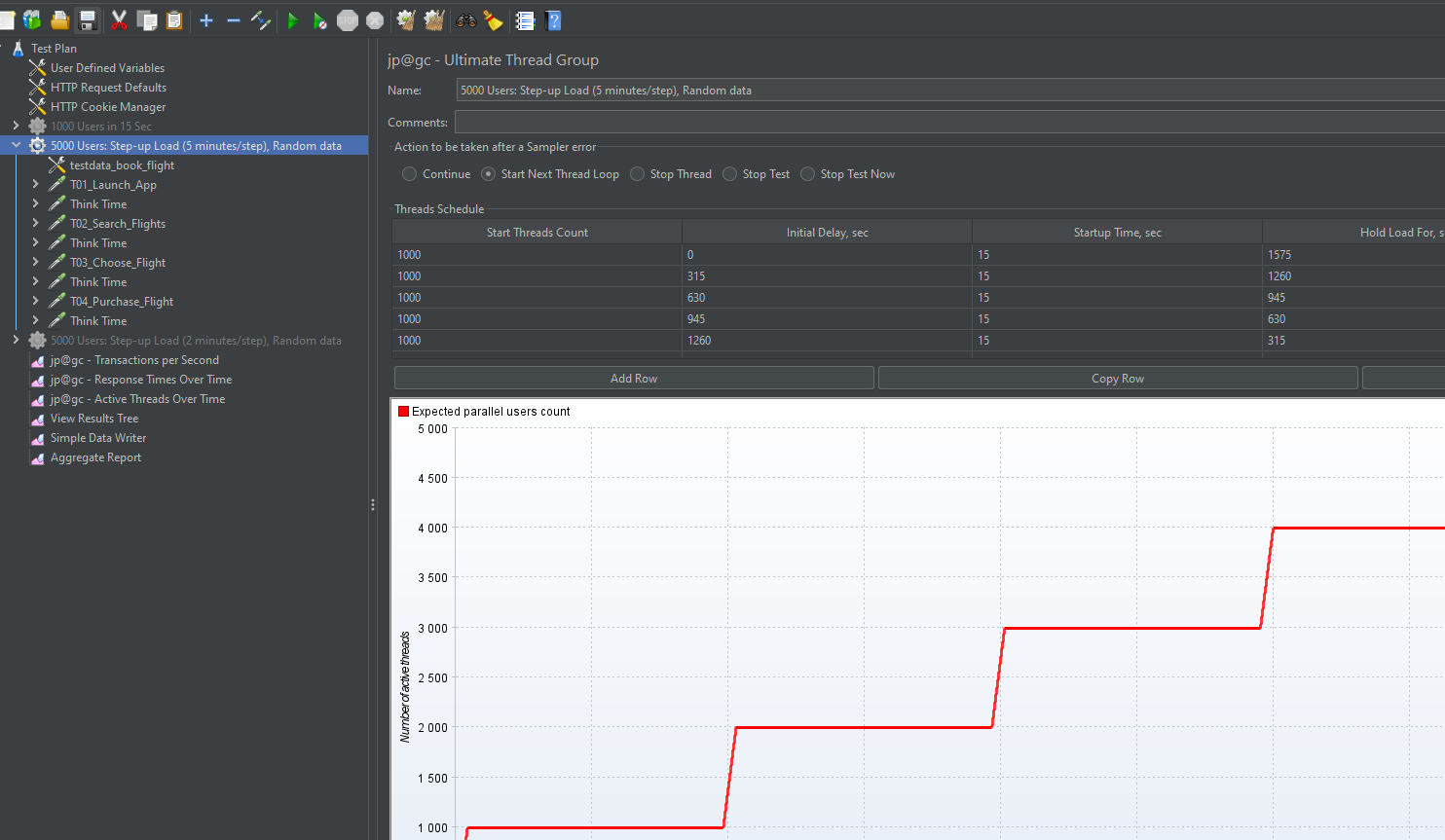


1. **1000 Users in 15 Sec:**
   1. As per the requirement, this test plan will simulate 1000 users logging into the application within 15 sec. and continue executing defined steps indefinitely (until test is stopped from JMeter)
   2. Test was designed in a way that each virtual user will use unique test data and this was achieved by using CSV data Set Config as shown in picture below

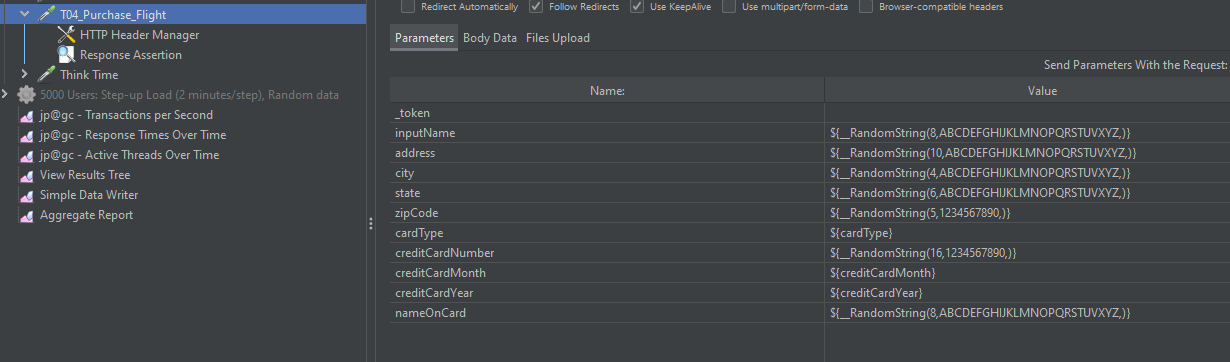


* 1. Random (between 2 to 6 sec.) think time (Sleep time) was used between 2 actions to simulate real user behavior and to avoid uncontrolled load on the application.
  2. Several listeners were used in the script to have real time monitoring of the performance when load test is running and also to write results into a file so that detailed report can be generated later.

1. **5000 Users: Step-up Load (5 minutes/step), Random data:**
   1. This test plan was designed to identify impact of increasing load on the web application performance.
   2. Test plan uses step-up load approach as visible in the screenshot. This approach help to find the limits quickly by running increasing load from the same test, instead of running several tests separately with different load parameters.



* 1. In this test plan, test data was made random so new data will be used by each user during each iteration. This was implemented to avoid any possible caching on the application server or DB. Test results might differ from reality if same data are used and if those are being cached by the AUT.



* 1. All other configurations are same as previous test plan

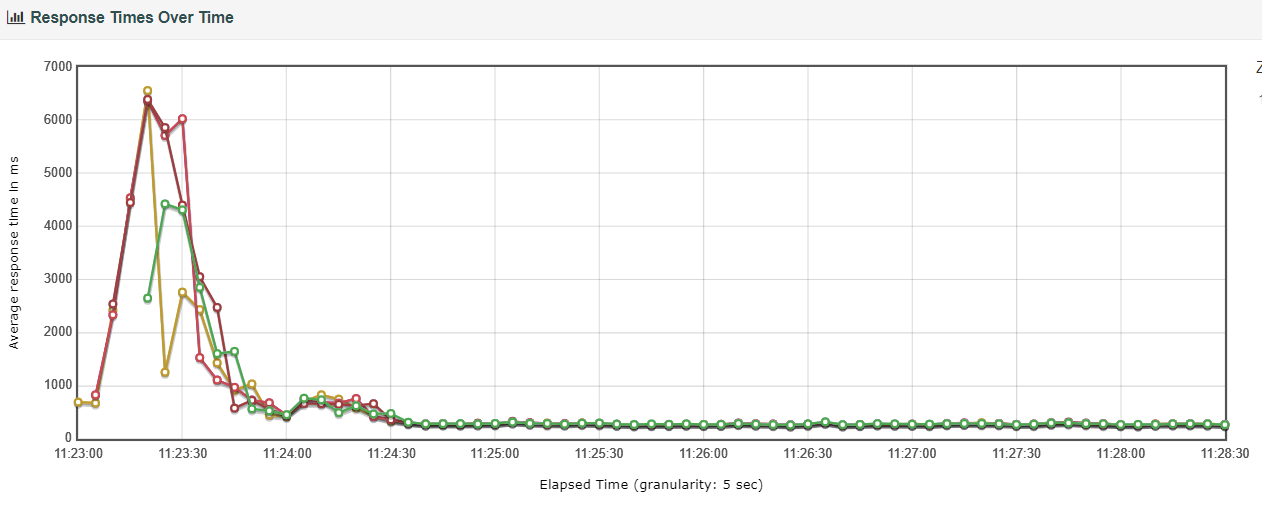
1. **5000 Users: Step-up Load (2 minutes/step), Random data:**
   1. Same as previous test plan, just each step runs for 2 minutes, to get quick idea about impact of the increasing load on the system. Previous test plan on the other hand runs for a longer duration to get the understanding of application stability when under load for longer duration.

**Test Result Analysis:**

Results from two tests are considered for performance analysis.

**Test 1: 1000 Users logged in 15 sec, test duration 5 minutes**

1. Overall 90th percentile response times below 500 msec.
2. No transaction failures
3. As shown in the graph below, application was rendering high response times (up to 7 sec) when users were logging into the application but after a minute or two, response times stabilizes below 500 msec and this clearly indicates that application is returning the response from the cache.



Detailed JMeter Test report:

***Note: To open JMeter report, save the zip folder locally, extract it and open index.html file in the browser.***

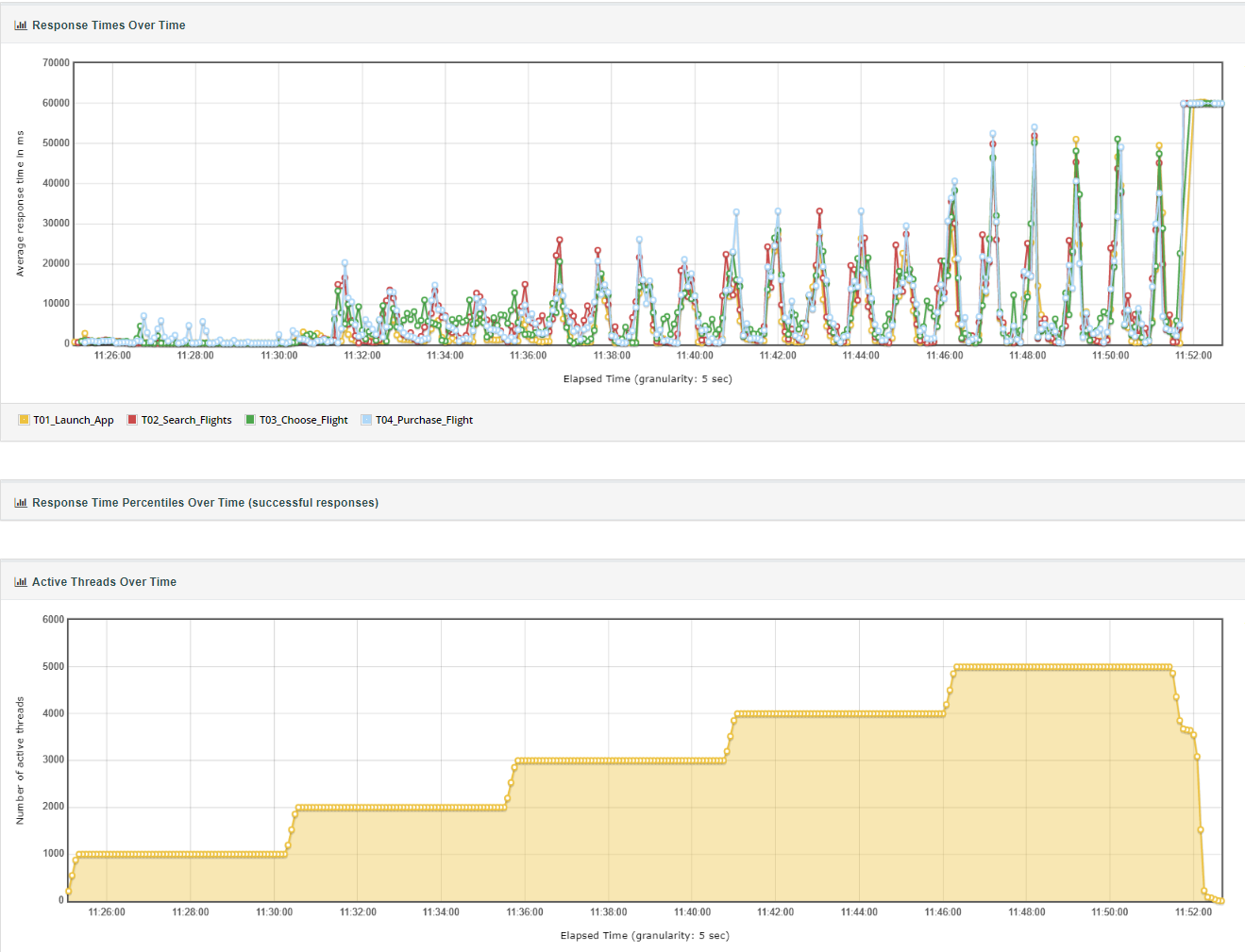


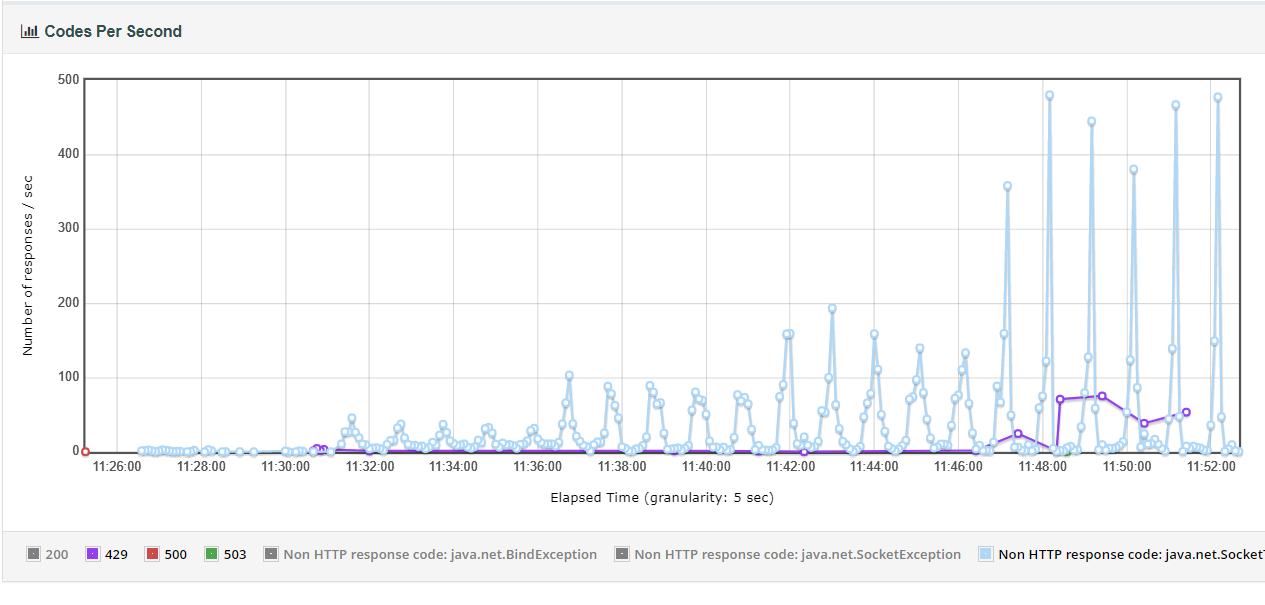
To avoid caching, test implementation was changed to use unique data every time in the next test.

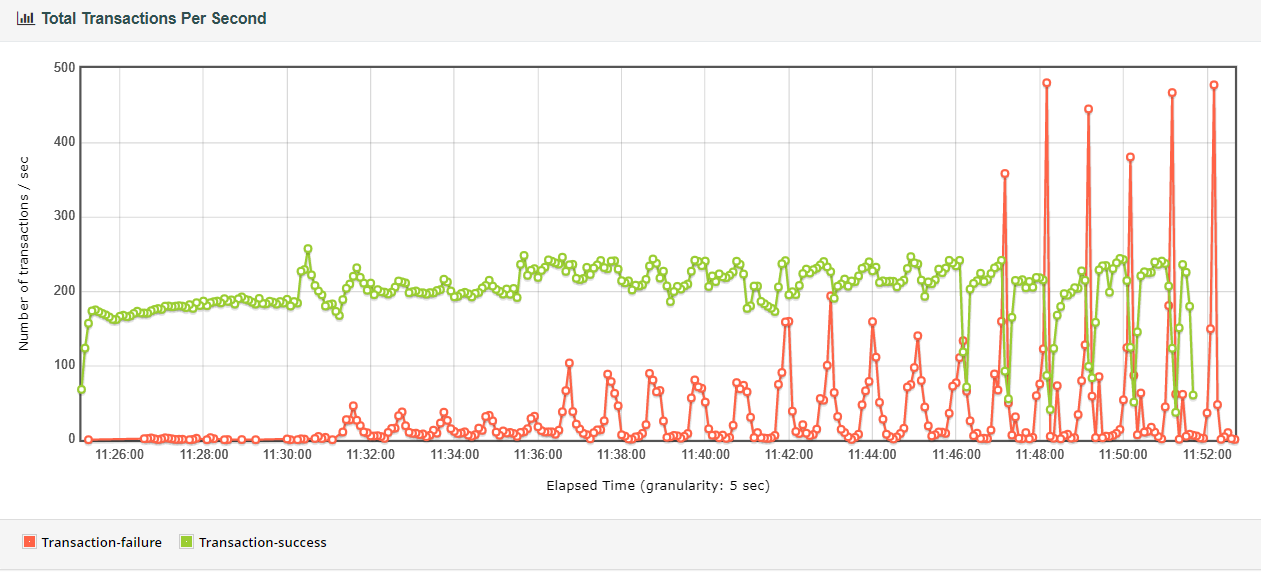
**Test 2: Load up to 5000 Users with each step of 1000 users and 5 minutes duration**

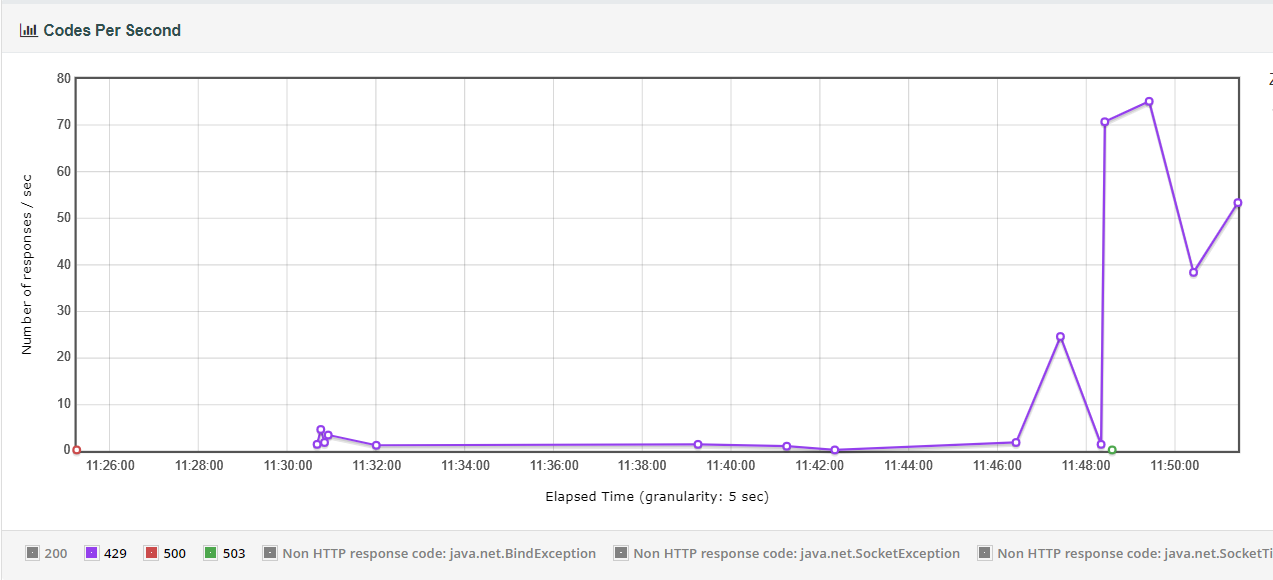
1. Overall 90th percentile response times 60 sec which is timeouts set in the tests.
2. 15% transactions failed mainly due to timeouts and “429/Too Many Requests”
3. **Impact of Load on response times:**

As shown below, response times are increasing in proportion to the amount of load



1. Blue line showing timeout (60 sec) errors. Timeouts increases rapidly as number of users in the system increases.
2. Also, as the number of users increases, failures increase. Throughput for successful transactions remains the same even with increasing load.



1. On increasing load beyond 1000 users, application starts complaining about “429/Too Many Request”

Detailed JMeter Test report:



**Conclusion:**

At max, application can support 1000 users (with avg. think time 4 sec) or in other terms max throughput of 300 requests/sec. After that app starts throwing too many requests errors, timeouts and high response times

**What is the optimal application response time for modern web applications?**

Response times & page load times are key metrics to be considered while determining a web application responsiveness.

**Response times** typically refers to API response times, meaning time it takes to get the response from the server. **One second or anything below 1 sec is considered acceptable response times for an API ideally.**

**Page load time** is the time it takes to download and display whole webpage which includes loading all elements, such as java scripts, CSS, images etc. **3 seconds or below is optimal for web pages in my opinion.**

**How would you define acceptable load for web applications?**

I would use the step-up load approach (like used in this case) to find out the breaking point, beyond which application performance will start degrading.

So Acceptable load for an application would be the point where application response times (90th percentile at least) are within acceptable limits with minimal (< 0.1%) failure rate, where application is able to perform consistently over a long period of time and CPU, Memory and network utilization is withing acceptable limits.

**JMeter Setup:**

Steps to open/run script in JMeter.

1. Download latest JMeter from <https://jmeter.apache.org/download_jmeter.cgi>.
2. Use <https://dlcdn.apache.org//jmeter/binaries/apache-jmeter-5.4.1.tgz> If <https://dlcdn.apache.org//jmeter/binaries/apache-jmeter-5.4.1.zip> doesn’t work. At least for me website was complaining about “Error 503 Backend unavailable”.
3. Extracting .tgz using 7-Zip will result in apache-jmeter-5.4.1.tar. Extract again .tar with 7-Zip and finally you would have apache-jmeter-5.4.1 folder containing JMeter software
4. Open "apache-jmeter-5.4.1\bin\jmeter.bat" file to get JMeter up and running. Note: JDK 8 needs to be installed before opening JMeter
5. Open script file “testcase\_book\_flight.jmx” from JMeter console
6. [Optional] Install necessary plugins.
   1. Follow step to install plugin manager <https://jmeter-plugins.org/install/Install/>
   2. Once plugin manager is installed, Search and Install following plugins directly from plugin manager
      * jpgc-graphs-basic
      * jpgc-casutg
7. [Optional] Change properties in "apache-jmeter-5.4.1\apache-jmeter-5.4.1\bin\user.properties" file for generating report with custom granularity in the graphs