

JMeter – Docker Test Executions

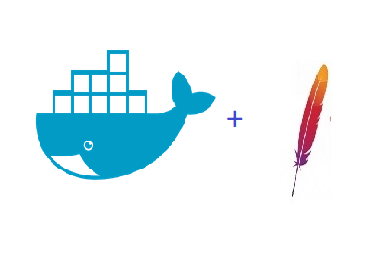
# Introduction

JMeter tests can be done on GUI and non-GUI in a headless front, both of which are easy to set-up for JMeter enthusiasts.

However, there are instances where developers/remote stakeholders/shared testing teams have to execute the Performance tests on different machines.

Even when the testing team does not have any reservations around sharing the testing scripts, the knowledge transfer on how to execute Jmeter scripts is an exhaustive process. There is so much to hand over from test data, folder structures, understanding active listeners, and finally the execution set up.

# Solution



To make the hand-over simple, one can execute JMeter test scripts via Dockers - to reduce the effort of setup significantly for both the parties involved.

Only three things required to get things moving:

* JMeter Docker Image
* Test scripts and data files
* JMeter test run command

# Benefits

* No need to install JMeter software conventionally
  + One Command to download
    - Use a *Pull* command to install the Docker jmeter image with latest version
  + Test Tool Uniformity across all systems
    - Even if there are any additional plugins or jar files and even changes to jmeter properties and user files can be replicated with the single pull command
  + Location
    - The Jmeter image is stored in the **Docker hub** as a public repository for everyone’s access
* Makes it easy to execute the tests with **preferred number of users and ramp up** without having to open the jmeter script via GUI
* No need to move feeder files/PDFs in folder structure(auto managed by GitHub download)
* Provision to Edit scripts for Jmeter enthusiasts
  + Scripts are stored in local drive who want to add/update/delete requests from the script  or change the flow as per the requirements
* Easy set up to handover when time comes

# Step by Step Guide

In a few easy steps, you can execute your first jmeter load test run using Dockers in the local machine.

Please note that this document will not discuss what is Docker and the inner workings of Docker in the interest of keeping the article short and crisp.

|  |  |  |
| --- | --- | --- |
| **ACTIVITY** | **DESCRIPTION** | **REMARKS** |
| **Pre-requisites**  **One time Setup** | **System:**  I will be using a Windows Machine for the set up. Using Mac should also be pretty straightforward | Get Power user or Admin Rights on the PC  \*non-mandatory but good to have |
| **Software:**  1. Docker Desktop  2. Windows PowerShell (pre-installed with OS) | Install Docker desktop on your windows machine  \*\* Latest version  <https://hub.docker.com/editions/community/docker-ce-desktop-windows> |
| **GitHub Repository Set up** | Upload the Project - test scripts to a version repository such as GitHub.  This repository will be used to share test scripts with development or any remote teams  \*\*For this article, I placed some sample test scripts in the below GitHub url. |
| **Folder Structure:**  Create the below folders in the structure mentioned in the screen capture | **Main Root: C folder**  **Root folder**:   * jmeter-base   Under the Root folder>  Create three sub-folders   * html * logs * TestBlazedemo (jmx, csv, pdf etc)   \*\*new for each project |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **ACTIVITY** | **DESCRIPTION** | **REMARKS** |
| **1** | **Pull JMeter Image** | Pull Docker Image of Jmeter(instead of installing it from Apache site)  **Steps**   1. Launch PowerShell CLI   **Command to download JMeter Image**  **docker pull swethapn14/repo\_perf:JmeterLatest**  You should see a message that the Pull is successful. As I already have it, I see that the image is up to date | One-Time Setup |
| **2** | **Download GitHub Scripts** | Place your JMeter/jmx test scripts in the local folder mentioned below  **Local Folder on PC**  **C:/jmeter-base/TestBlazedemo** | Remote or Development teams to download the test scripts from the GitHub repository/SharePoint to the local path  **TestBlazedemo can be downloaded from my public GIT repo**  **https://github.com/swethapn/blazeDemo** |
| **3** | **PowerShell CLI**  **Run the Jmeter Command to execute Load tests against the test script** | **docker run** **-v** C:/[jmeter-base:/workspace swethapn14/repo\_perf:JmeterLatest](http://jmeter-base/workspace%20swethapn14/repo_perf:JmeterLatest) **-Jthreads=10 -Jrampup=20** -n -t /workspace/TestBlazedemo**/Blazedemo.jmx** -l /workspace/logs/**Blazedemo\_10Vu.jtl**-e -o /workspace/html/**10VuReport** | \*\*Run as many times you want to test  Change the attributes in **GREEN** for every test run   * Jthreads – Number of users for the load test * Jrampup – The time for the users to initiate in **seconds** * Change the test script when needed * Log file’s name to change every test run * Html report name to change for every test run   Refer Figure 1 For JMeter execution on Docker PowerShell CLI |
| **4** | **To view the test results after the load test completes** | Go to the html folder given in the command and find the index.html  **Folder**  **C:/jmeter-base/html/10VuReport/index.html** | After each test run, view the results in the html folder name provided In the Command above  Refer Figure 2 for Report Screenshot |

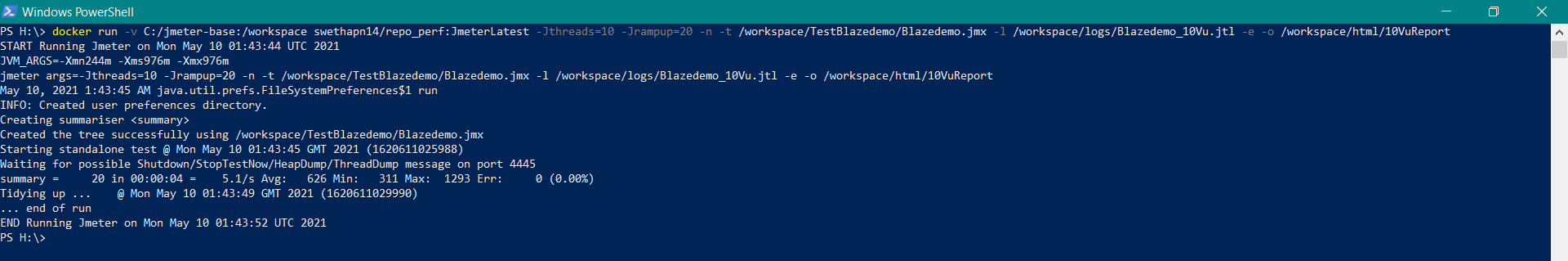


Figure 1: PowerShell CLI Docker Run

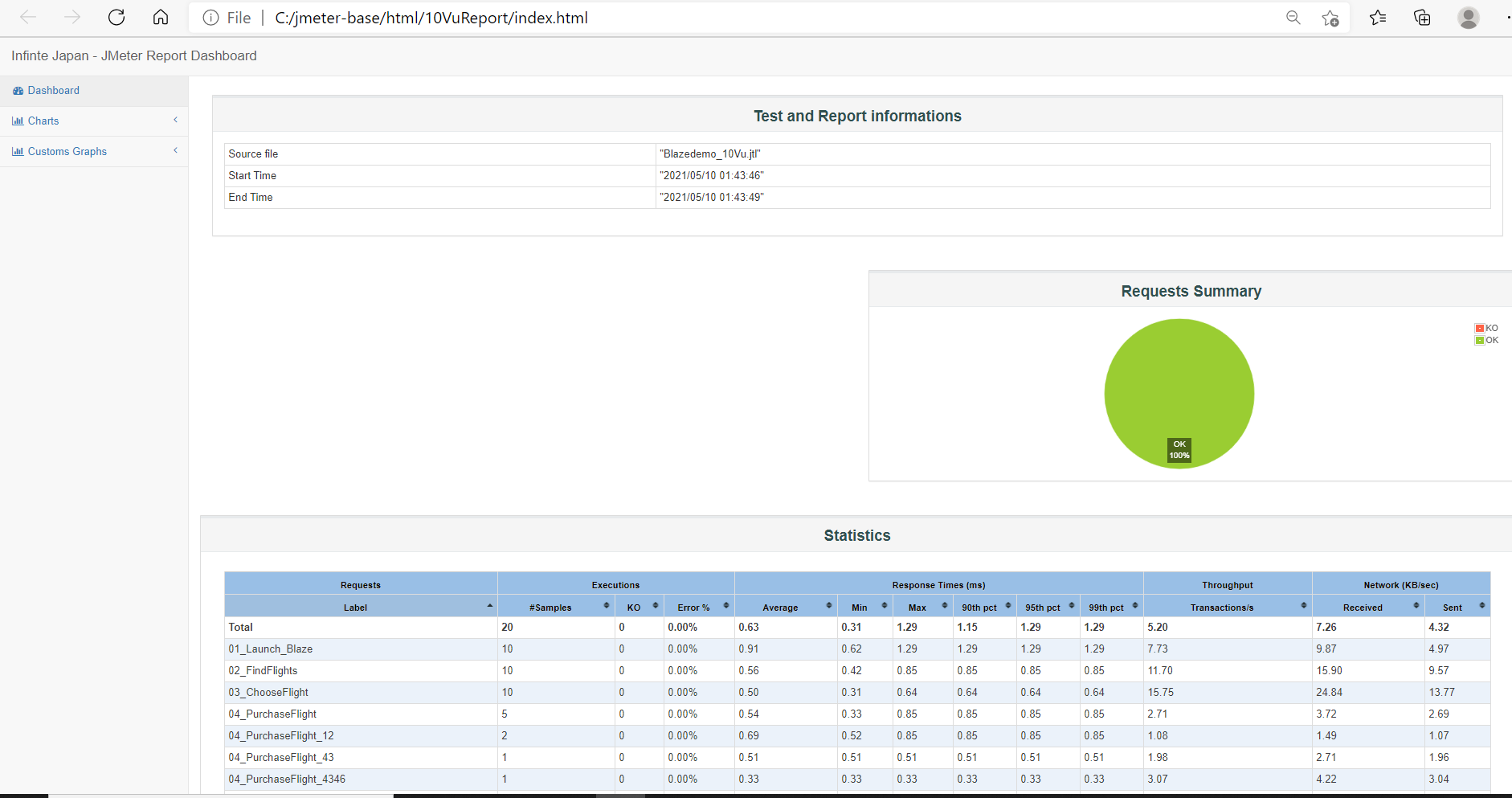


Figure 2: HTML Report

# JMeter Command Explained

JMeter command to execute test runs via Docker -

**docker run** **-v** C:/[jmeter-base:/workspace swethapn14/repo\_perf:JmeterLatest](http://jmeter-base/workspace%20swethapn14/repo_perf:JmeterLatest) **-Jthreads=10 -Jrampup=20** -n -t /workspace/TestBlazedemo**/Blazedemo.jmx** -l /workspace/logs/**Blazedemo\_10Vu.jtl**-e -o /workspace/html/**10VuReport**

The command does the work needed to execute a load test with ‘10’ users that will ramp up in ‘20’ seconds. The users inject traffic on the test script **Blazedemo.jmx** and write the results to a log file named **TestBlazedemo\_10Vu\_5Loop.jtl**. After the test finishes, the JMeter HTML report will be automatically generated & placed in the **html/10VuReport folder** in the workspace(C:/jmeter-base)

|  |  |  |
| --- | --- | --- |
| **Command segment** | **Meaning** | **To be Changed** |
| **docker run** | mandatory | Don’t Change |
| **-v** C:/[jmeter-base:/workspace](http://jmeter-base/workspace%20swethapn14/repo_perf:JmeterLatest) | setting up a docker **volume** to connect your local C drive to docker workspace  --Local drive--  **C:/**[**jmeter-base**:/workspace](http://jmeter-base/workspace)  Do not change unless you have a different folder structure  --Note--  Your local C:/**jmeterbase** folder is called as **workspace** going forward.  This folder contains the scripts, test data files and pdfs for any upload functionality | **Don’t change unless the folder path in your system is different** |
| swethapn14/repo\_perf:JmeterLatest | refers to the jmeter image you pulled in  Activity 1 | Change if pulling a different jmeter image |
|  |  |  |
| **o Arguments** |  |  |
| **-H**  **-P** | Host and port to access internet based app | Update accordingly |
| **-Dhttps.protocols=TLSv1,TLSv1.1,TLSv1.2** | To provide the SSL layer for JMeter |
|  |  |  |
| -**Jthreads=1 -Jrampup=4** | jmeter arguments to input number of users and ramp up, if not provided, default will be 10 users in 20 seconds as recorded in the test script | To Change |
| Edit the numbers reasonably - for every 1 user, ramp up 2 seconds  eg: 50 users, 100 seconds ramp up etc |
| **-n** | To make the jmeter test run headless – Non GUI | Don’t change |
| **-t** /workspace/TestBlazedemo**/Blazedemo.jmx** | test script with jmx extension, point to the Script to execute in workspace | Text in black: Don’t change  Text in **Green**: To Change depending on the script/file names |
| **-l** /workspace/logs/**Blazedemo\_10Vu.jtl** | Log file name |
| **-e** | Command to generate the report automatically after the test run | **Don’t change** |
| **-o** /workspace/html/**10VuReport** | The generated report is placed in the **O**utput path.  After the test run completes, please visit the html folder and open the index.html for the test results | Text in black: Don’t change  Text in **Green**: To Change for every test run |

# Command to execute JMeter tests on Proxy with Security Layer

**docker run** **-v** C:/[jmeter-base:/workspace swethapn14/repo\_perf:JmeterLatest](http://jmeter-base/workspace%20swethapn14/repo_perf:JmeterLatest) **–H proxy.xxx.com –P 3120 -Dhttps.protocols=TLSv1,TLSv1.1,TLSv1.2 -Jthreads=10 -Jrampup=20** -n -t /workspace/TestBlazedemo**/TestScriptName.jmx** -l /workspace/logs/**ProjectName\_testEnv\_Datetime\_NoOfVu\_Vu.jtl**-e -o /workspace/html/**TestReportName\_NoOfVu\_Project\_Report**

**–H proxy.xxx.com –P 3120 : The Host and Port on which the Proxy of your Organization is set up**

**- Dhttps.protocols=TLSv1,TLSv1.1,TLSv1.2 : Transport Layer security to access the Application under test**

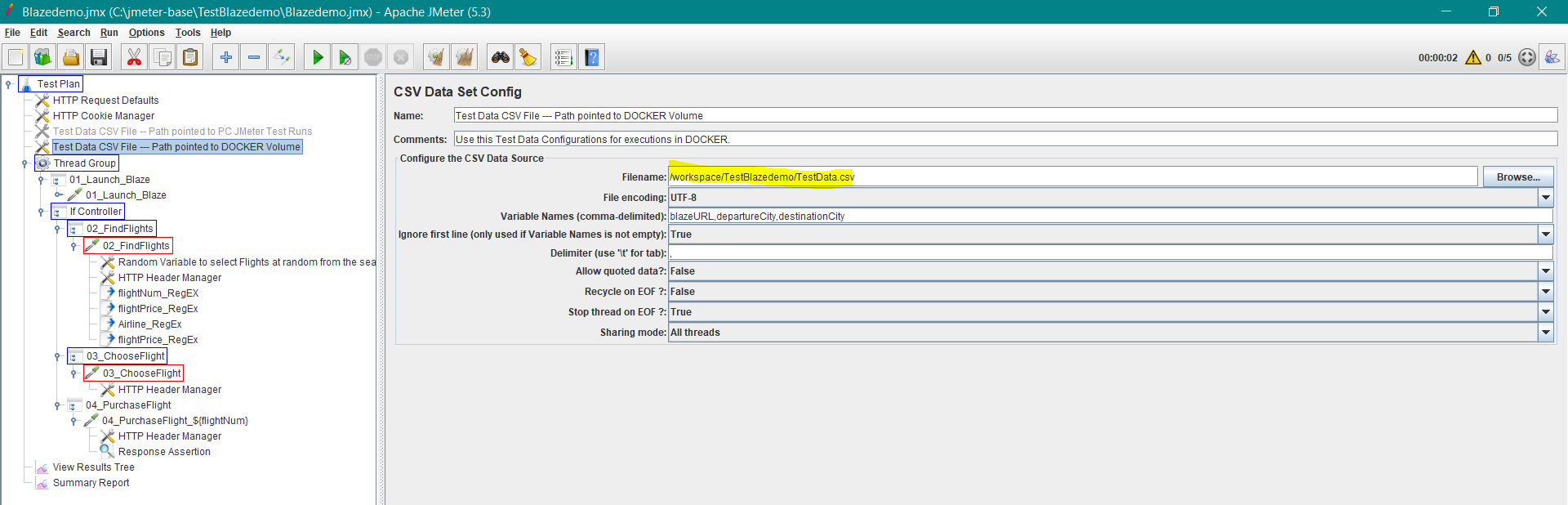
# Pointers on CSV Data Set Config – TestData

CSV Data Set configs are popularly used to feed test data to jmeter scripts. It is included in the Test Plan and contains mapping to the CSV file location.

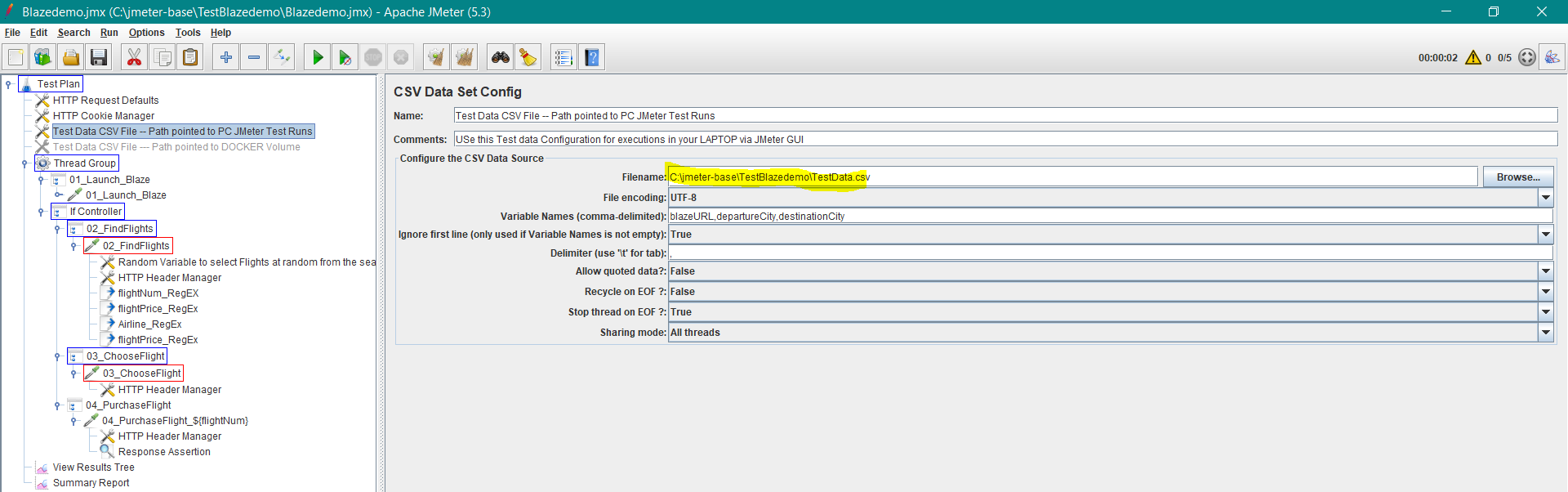
Please note that there are a couple of pointers to note for executions on Docker vs executions in JMeter GUI.

Keep a look out on the forward and backward slashes (‘/’ ‘\’) in the file paths specified

* Docker Executions
  + CSV file Path
    - The path should be mapped to Docker volume
    - /workspace/ProjectName/Filename.csv



* For JMeter GUI Test Runs
  + CSV File Path should point to the Windows C:/ folder path
  + C:\jmeter-base\ProjectName\Filename.csv



# Recap

The above steps allows developers/team members to do one of the following:-

|  |
| --- |
| I. Novice   1. Pull JMeter image from Docker hub 2. Download JMeter scripts(latest from GitHub project repository) 3. Run the test by changing the # of users and # ramp up time |
| II. Expert  Edit scripts after downloading from GitHub as the scripts will be in the local workspace |
| III. Novice  View reports in html folder after the test completes  Analyze and log bugs |