**Introduction:**

Diagram

Description automatically generated

## Overview:

Apache JMeter is an open-source performance testing tool; developed in Java. JMeter does not require any purchasing or licensing cost. It means that you can conduct the performance testing of an application or a software system without spending a single penny on the tool. JMeter becomes a rapidly growing tool due to its unrevealed features which compete for the other licensed tools. To make this tool stronger, Apache JMeter Development Community is continuously working on the requirement raised by JMeter users and provides the useful plug-in. The community also releases the upgraded version of the tool with major updates.

## A brief history of Apache JMeter:

Stefano Mazzocchi developed Apache JMeter and the officially released the first version in 1998. Since then it’s features have been growing day by day. JMeter became a Top-Level Apache project in November 2011, which means it has a Project Management Committee and a dedicated website.

## Features:

Apache JMeter is a Java-based tool which means it is a platform-independent tool. Any popular operating system supports JMeter without any issue i.e. you can install the JMeter easily. The only one thing which an operating system should have is the supportive version of Java. JMeter has nearly all the basic as well as advanced features which a performance testing tool should have.

Following are some important features of Apache JMeter (as on 01/05/2019):

* Apache JMeter is an open-source tool. It means free to use.
* No additional licensing cost or purchasing the pro version.
* Pure Java-based application, so works on any operating system i.e. Platform Independent.
* Lightweight and easy to install
* No maintenance cost
* Easy to upgrade
* Easy to enhance the features by simply adding .jar file i.e. plug-ins
* No explicit hardware is required (depends upon the load requirement)
* Unlimited User load (Thread) generation capability (depending on LG configuration)
* Simple and User-friendly GUI
* Run in both GUI and Non-GUI mode
* Recording option helps to record the user action on an application
* Supports all the basic scripting features like parameterization, correlation, browser simulation delay in request etc.
* Easy to debug the scripts
* Support Android application performance testing
* Test the performance of both static and dynamic resources
* Supports multiple coding languages like BeanShell, Groovy and JavaScript for scripting purpose.
* Supports protocols like Web/HTTP, Web services, Databases, SOAP, etc.
* It can be integrated with the third-party APM tools like DynaTrace, Applicare, Grafana etc.
* Integration with CI tools like Jenkins etc.
* Drag and drop declarative view for scripts.
* Average reporting pattern (Report in HTML format post 3.0 version)
* Proper documents are available to gain knowledge of the functionality
* Free and Commercial Online supports are available

## Limitations:

Along with an ample number of features, Apache JMeter has some limitations too, which are (as on 01/05/2019)

* Limited protocol support
* No dedicated support team
* The script cannot be regenerated
* No Protocol adviser
* Compatibility issue in the existing script while upgrading the version
* No Run time graphs (only available when integrates with some monitoring tool like Grafana)
* Dependency on Bean shell and other coding languages for complex scenario scripting
* GC/Heap memory issue is common while running a test on a limited resource machine.
* Unpredictable issues in distributed testing
* Lack of in-depth analysis features
* Reports having fewer features which make analysis bit difficult

Apache JMeter is a simple and useful performance testing tool. Most of the product based companies use JMeter to test the performance of their products. But, service-based companies have a different thought and shows less interest to accept JMeter. They still prefer and recommend the licensed tool. Although, it is very well accepted by many clients because it saves money and provides an accurate result. To conclude, Apache JMeter is a well-known name in the Performance Testing world which is ease-to-use, free of cost and of-course good for Performance Testing.

**Installation:**

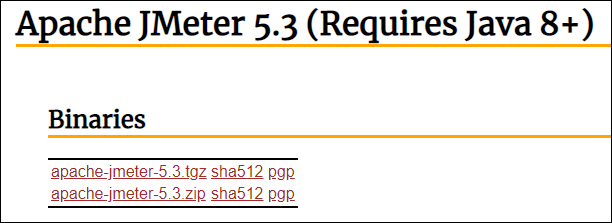
JMeter is a pure Java desktop application, latest JMeter requires a fully compliant JVM 8 or higher. JDK/JRE is pre-requisite for Apache JMeter. If you want to know how to install Apache JMeter then follow these simple steps:

**1. Verify or Install JDK/JRE:** Refer below links to verify JDK/JRE on your system OR install JDK/JRE on various OS:

If you do not have JDK on your system then you can download and install the latest version of Java SE Development Kit by clicking below link:

**2. Download JMeter:** You can download the latest version of Apache JMeter by clicking below link:

Click on the Binaries file (either zip or tgz) to download as shown in the figure below (after clicking the download link, you can see the downloading of the JMeter set-up. JMeter version may vary as this image is just for illustration):



**3. Installation of JMeter:** Installation of JMeter is extremely easy and simple. You simply unzip the zip/tar file into the directory where you want JMeter to be kept (installed). There is no tedious installation screen to deal with! Simply unzip and you are done!

**JMeter Folder Structure**:

After simply unzipping the downloaded file of Apache JMeter you can see some folders in the main directory of Apache JMeter-<version> which are:

* /backups:
* /bin: Contains JMeter script file for starting JMeter
* /docs: JMeter documentation files
* /extras: ant related extra files
* /lib/: Contains the required Java library for JMeter
  + /lib/ext: contains the core jar files for JMeter and the protocols
  + /lib/junit: JUnit library used for JMeter
* /licenses:
* /printable\_docs:
* LICENSE
* NOTICE
* README.md
* While learning the basics of Apache JMeter, it is good to understand the folder structure of JMeter so that you get to know which are the important files or folders and where they are located?

Table

Description automatically generated with medium confidence

Above is the folder structure of JMeter. Each folder contains sub-folders or files. Starting with the root folder.

**Root Folder:**

‘apache-jmeter-<jmeter version number> is the root folder which contains all the JMeter related folder and files. In the above example, apache-jmeter-5.3 is the root folder.

**/backup Folder:**

Under this root folder, the first folder which you will get is ‘/backup’ folder. If you haven’t done scripting yet then you may not get this folder. Because this folder is created by JMeter when you start scripting. At the time of the first script, JMeter automatically creates this folder and saves the script in it after a certain interval. These back-up files are in .jmx format. There is a limit for the number of back-ups taken by JMeter for a particular script. This limit can be changed by modifying the respective value in the jmeter.properties file.

**/bin folder:**

The next folder is the ‘bin’ folder. This is an import folder where all the binary files are located. All the executable files of JMeter are in bin folder just like jmeter.bat, jmeter.sh, heapdump.sh etc. You can use jmeter.bat file to launch the JMeter in GUI mode. Apart from the executable file, JMeter log file, Security Certificate, config file, groovy file and properties files (like jmeter.properties, user.properties etc.) are also available in the bin folder. There are 3 sub-folders i.e. examples, report-templates and templates are also present in the bin folder. The examples sub-folder contain some simple example of CSV file and script. The ‘report templates’ and ‘templates’ subfolder contains the template for JMeter report and test plan respectively.

**/doc Folder:**

The docs folder contains image, CSS and API subfolder. The stuff in this folder is helpful for JMeter documentation and presentation. The image sub folder contains JMeter logo, the CSS sub folder contain relateds style.css file etc.

**/extras Folder:**

This folder contains XML, XSL, BSH, sh files which are supporting files for JMeter GUI.

**/lib Folder:**

The lib folder contains all the executable jar files which are required for JMeter functioning. These files are as same as jar file of java. /lib folder contains two subfolders /ext and /JUnit. The /ext or extension sub-folders are useful when you add an external plugin in the JMeter.

**/license Folder:**

The next folder is named as Licenses folder. As a legal process, Apache and other companies produce the licenses, terms and conditions, notice, copyrights etc. to use and distribute the executable jar files which are in the lib folder. All the licenses and notices are located in this folder.

**/printable\_docs Folder:**

The last folder is printable\_docs, in which you can see JMeter related documents, demos, JMeter tutorial in PDF format which are under extending sub-folder and user manuals in HTML format. If you want to open any file just double click on it and the related topic will be opened.

Along with the folders which we have discussed above, you will also get three files which are License, Notice and Readme.md. These files belong to overall Apache JMeter license, terms and conditions and copyrights notice.

**Launching mode:**

After successful installation of JMeter, the next step is to launch the JMeter tool for scripting purpose

There are 3 modes to launch Apache JMeter application on your system. Each mode has its own benefits:

1. GUI Mode
2. Command Line Mode (Non-GUI Mode)
3. Server Mode (Non-GUI Mode)

*Noten: GUI mode is generally used for creating the test script, Non-GUI mode must be used for load testing because Non-GUI mode consumes less memory.*

## 1. JMeter GUI Mode

**Purpose:** The GUI mode is used for preparing the test script, debugging the script and for performing the smoke test. In this mode, JMeter consumes relatively more memory than Non-GUI mode.

For Windows and Mac users, JMeter has the simplest way to launch the GUI. Navigate to the /bin folder of Apache JMeter and double click on **jmeter.bat**. This will first open JMeter console followed by JMeter GUI.

To open JMeter GUI through Windows command line, you need to follow below steps:

1. Go to ‘Run’
2. Type ‘cmd’
3. Navigate to /bin folder of Apache JMeter
4. Type ‘jmeter.bat’
5. Hit ‘Enter’ key

## 2. JMeter Non-GUI Mode

**Purpose:** The Non-GUI mode is used for test execution because JMeter consumes less memory in this mode; helps to provide more space for threads and prevents test failure due to lack of memory.

To launch the JMeter or execute the test in Non-GUI mode, you need to open the console, navigate to the /bin folder of Apache JMeter and run the below command:

**For Windows:** jmeter -n -t <script name with path> -l <log file name with path>

**For Unix:** ./jmeter.sh -n -t <script name with path> -l <log file name with path>

In addition, if you want to know about all the arguments for JMeter command then type ‘jmeter -?’, you will get below output:

--?

print command line options and exit

-h, --help

print usage information and exit

-v, --version

print the version information and exit

-p, --propfile <argument>

the jmeter property file to use

-q, --addprop <argument>

additional JMeter property file(s)

-t, --testfile <argument>

the jmeter test(.jmx) file to run. "-t LAST" will load last

used file

-l, --logfile <argument>

the file to log samples to

-i, --jmeterlogconf <argument>

jmeter logging configuration file (log4j2.xml)

-j, --jmeterlogfile <argument>

jmeter run log file (jmeter.log)

-n, --nongui

run JMeter in nongui mode

-s, --server

run the JMeter server

-E, --proxyScheme <argument>

Set a proxy scheme to use for the proxy server

-H, --proxyHost <argument>

Set a proxy server for JMeter to use

-P, --proxyPort <argument>

Set proxy server port for JMeter to use

-N, --nonProxyHosts <argument>

Set nonproxy host list (e.g. \*.apache.org|localhost)

-u, --username <argument>

Set username for proxy server that JMeter is to use

-a, --password <argument>

Set password for proxy server that JMeter is to use

-J, --jmeterproperty <argument>=<value>

Define additional JMeter properties

-G, --globalproperty <argument>=<value>

Define Global properties (sent to servers)

e.g. -Gport=123

or -Gglobal.properties

-D, --systemproperty <argument>=<value>

Define additional system properties

-S, --systemPropertyFile <argument>

additional system property file(s)

-f, --forceDeleteResultFile

force delete existing results files and web report folder if

present before starting the test

-L, --loglevel <argument>=<value>

[category=]level e.g. jorphan=INFO, jmeter.util=DEBUG or com

.example.foo=WARN

-r, --runremote

Start remote servers (as defined in remote\_hosts)

-R, --remotestart <argument>

Start these remote servers (overrides remote\_hosts)

-d, --homedir <argument>

the jmeter home directory to use

-X, --remoteexit

Exit the remote servers at end of test (non-GUI)

-g, --reportonly <argument>

generate report dashboard only, from a test results file

-e, --reportatendofloadtests

generate report dashboard after load test

-o, --reportoutputfolder <argument>

output folder for report dashboard

To stop the test in the middle, again you need to navigate to the /bin folder of JMeter and type **stoptest**in command prompt for Windows machine and ***./stoptest.sh*** for Unix machine. If more than one JMeter test scripts are executed then you need to stop the individual scrip by appending the process ID into the stop command like **stoptest 4445**.

There are some additional useful commands for Windows OS:

* jmeter.bat: To launch JMeter in GUI mode
* jmeterw: To run JMeter without the windows shell console (in GUI mode by default)
* jmeter -n: To run JMeter test in a non-GUI test
* jmeter -n -r: To run JMeter test in a non-GUI test remotely
* jmeter -t: To drop a JMX file on this to load
* jmeter-server.bat: To start JMeter in server mode
* mirror-server: To run the JMeter Mirror Server in non-GUI mode
* shutdown: To shutdown client to stop a non-GUI instance gracefully
* stoptest: To shutdown client to stop a non-GUI instance abruptly

## 3. JMeter Server Non-GUI Mode

**Purpose:** Server mode is used for distributed testing. This testing works as a client-server model. In this model, start the server(s) by running jmeter-server[.bat] on each server host and control those servers using GUI mode.

The script also lets you specify the optional firewall/proxy server information:  
-H [proxy server hostname or IP address]  
-P [proxy server port]

Example: jmeter-server -H proxy.server -P 8080

If you are testing from behind a firewall/proxy server, you need to provide JMeter with the firewall/proxy server hostname and port number.

-H [proxy server hostname or IP address]  
-P [proxy server port]  
-N [nonproxy hosts]  
-u [username for proxy authentication – if required]  
-a [password for proxy authentication – if required]

Example: jmeter -H proxy.server -P 8080 -u myname -a mypassword -N localhost

To execute the test in server non-GUI mode, first navigate to bin folder of JMeter and run the following command:

**For Windows:** jmeter -n -t <script path> -l <log file path> -r

**For Unix:** ./jmeter.sh -n -t <script path> -l <log file path> -r

The test can also be executed by selecting specific slave machines.

**For Windows:** jmeter -n -t <script path> -l <log file path> -R server1,server2,server3,

**For Unix:**./jmeter.sh -n -t <script path> -l <log file path> -R server1,server2,server3,

**How to record JMeter script for a secured (https) application?**

Before going to record any secured web application through JMeter, there is a requirement to add JMeter secured CA certificate in the browser. The secured web application can be recorded using the HTTP(S) Test Script Recorder in JMeter. The secured (HTTPS) connections use certificates to authenticate the connection between the browser and the webserver. When connecting via HTTPS, the server presents the certificate to the browser. To authenticate the certificate, the browser checks that the server certificate is signed by a Certificate Authority (CA) that is linked to one of its in-built root CAs. JMeter needs to use its own certificate to enable secured connection to intercept the HTTPS connection from the browser. Effectively JMeter has to pretend to be the target server.

### ****How to install JMeter certificate in different browsers?****

You can follow below steps to add the JMeter certificate in the respective browser:

**Mozilla FireFox:**

1. Go to Firefox settings
2. Search for the keyword ‘View Certificates’ **or** Go to ‘Privacy & Security’ tab
3. Go to ‘Certificates’ section
4. Click ‘View Certificates’
5. Click ‘Authorities’
6. Press ‘Import …’ button
7. Browse ‘ApacheJMeterTemporaryRootCA.crt’ (available at apache-jmeter-X.X -> bin)
8. Click Open
9. Click ‘View’ and check that the certificate details agree with the ones displayed by the JMeter Test Script Recorder
10. If OK, select ‘Trust this CA to identify websites’ and press OK
11. Close dialogues by pressing OK as necessary

**Google Chrome:**

1. Go to Chrome ‘Settings’
2. Type ‘Certificate’ in ‘Search Settings’ text box **or** Go to ‘Privacy and security’
3. Click ‘More’
4. Click ‘Manage Certificates’
5. Press ‘Imports…’ button
6. ‘Certificate Import Wizard’ will open.
7. Click ‘Next’
8. Browser ‘ApacheJMeterTemporaryRootCA.crt’ (available at apache-jmeter-X.X -> bin)
9. Click ‘Next’, ‘Next’ and ‘Finish’

**Microsoft Edge**:

1. Go to Edge browser ‘Settings’
2. Type ‘Certificate’ in ‘Search settings’ text box **or** Go to ‘Privacy and services’ and go to ‘Privacy’ section
3. Under the ‘Privacy’ section you will get ‘Manage Certificates’, click on it.
4. Goto
5. Click ‘Manage Certificates’
6. Press ‘Import…’ button
7. ‘Certificate Import Wizard’ will open.
8. Click ‘Next’
9. Browser ‘ApacheJMeterTemporaryRootCA.crt’ (available at apache-jmeter-X.X -> bin)
10. Click ‘Next’, ‘Next’ and ‘Finish’

### ****Some Important Points regarding Apache JMeter Root CA Certificate:****

1. This certificate is created or updated automatically when you [launch the JMeter](https://www.perfmatrix.com/jmeter-launching-modes/) for recording.
2. The root CA certificate is used for recording the secured application or website. A non-secure application does not require this certificate.
3. The validity of the root CA certificate is for 7 days from the date of generation.
4. In case of expiration, you can delete the old certificate from the browser and import the newly generated certificate.
5. JMeter root CA certificate is available in the /bin folder of Apache JMeter.

**How to record JMeter script?**

Apache JMeter is a Java-based tool which records the communication between client i.e. end-user and server and converts this communication into a test script. This test script is used to generate the load on the server. To record the communication through JMeter and convert it into a script, you must have below listed 5 pre-requisites:

1. Application or Website Link
2. Test Data like user credentials
3. Transaction Flow Diagram / Navigation Flow / Use Case
4. Web-browser (by default you have) like Chrome, Firefox etc.
5. Apache JMeter

### ****JMeter Script Recording Steps:****

1. Launch JMeter by navigating to <JMETER\_HOME>/bin (**Example:** C:\apache-jmeter-5.3\bin)

2. Click jmeter.bat (Windows) or  Enter command ‘jmeter’ (Linux/Unix).

Graphical user interface, text, application

Description automatically generated

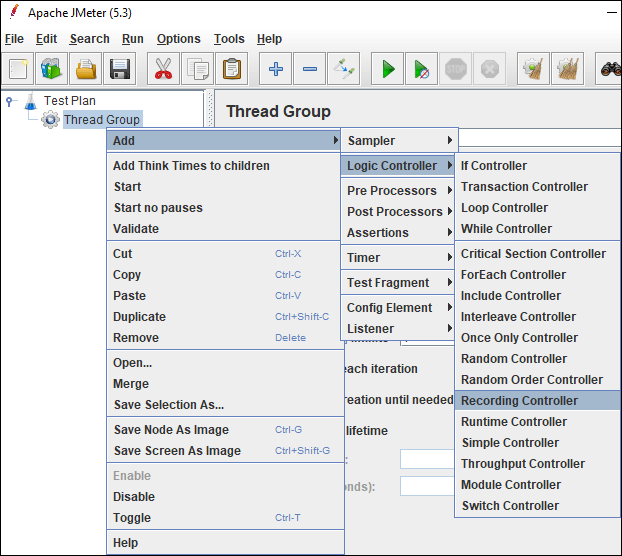
3. Select ‘Test Plan’ on the tree

4. Right-click on the ‘Test Plan’ and add a ‘Thread Group’

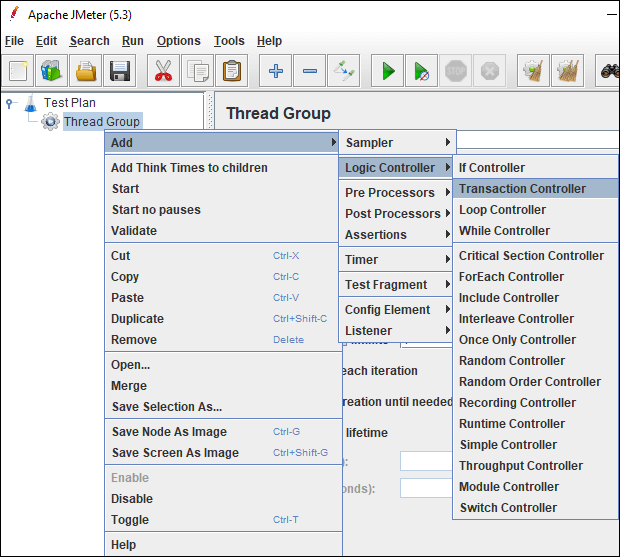
Graphical user interface, text, application, email

Description automatically generated

5. Select ‘Thread Group’ and right-click to add ‘Recording Controller’.



*Note: You can also use ‘Transaction Controller’ instead of ‘Recording Controller’.*



6. Right-click on the ‘Test Plan’ and add ‘HTTP(S) Test Script Recorder’ which is a ‘Non-test Element’.

Graphical user interface, text, application, email

Description automatically generated

7. Under ‘HTTP(S) Test Script Recorder’, add ‘View Results Tree’ listener by right-clicking on ‘HTTP(S) Test Script Recorder’ element and provide a .jtl file path in the Filename. This .jtl file is required to save the recording logs.

Text

Description automatically generated with medium confidence

8. In ‘HTTP(S) Test Script Recorder’, under ‘Global Settings’ section, provide the port number as ‘8080’ and application URL in ‘HTTPS Domain’ text field. While providing application URL do not use http, https,: and /.

Graphical user interface, text, application, email

Description automatically generated

9. In ‘HTTP(S) Test Script Recorder’, under ‘Test Plan Creation’ tab, the default value of the ‘Target Controller’ is ‘Use Recording Controller’ In case you use ‘Transaction Controller’ as shown in step 5 then select the particular transaction controller as target controller.

Graphical user interface, text, application, email

Description automatically generated

10. **Optional Step:**On HTTP(S) Test Script Recorder, under ‘Request Filtering’ tab, click the ‘Add’ button in ‘URL Patterns to Include’. This will create a blank entry; write ‘.\*\.html’.  
Also click the ‘Add suggested Excludes’ button in ‘URL Patterns to Exclude’ section, this will add a new row i.e. (?i).\*.(bmp|css|js|gif|ico|jpe?g|png|swf|woff|woff2). After excluding such request pattern, JMeter does not record those requests which have excluded keywords. You can also add new custom pattern or keyword in this list.

Graphical user interface, text, application, email

Description automatically generated

11. Now, Launch Browser (Chrome/Firefox/Edge) and set Manual Proxy.

**Address:**localhost or IP address of your system  
**Port:** 8080

Refer respective screenshot:

**FireFox:**

Graphical user interface

Description automatically generated with medium confidence

**Chrome:**

Graphical user interface, application

Description automatically generated

12. In case the application is secured (https) then you need to add ‘JMeter Root CA certificate’ to the browser.

13. Go back to JMeter, in ‘HTTP(S) Test Script Recorder’ and click the ‘Start’ button.

Graphical user interface, application

Description automatically generated

14. A ‘Root CA Certificate’ pop-up will open. Click ‘OK’.

Graphical user interface, text

Description automatically generated

15. Go to your browser and type the application URL and follow the transaction flow. You can see a floating ‘Recording Bar’ while recording the flow.

16. Once navigation is completed, click the ‘Stop’ button on Recording Bar or in ‘HTTP(S) Test Script Recorder’.

Graphical user interface, text, application

Description automatically generated

**How to handle dynamic value (correlation) in JMeter?**

‘Correlation’ term refers to the handling of dynamic values coming from the server. These dynamic values are the unique values which are generated by the server for security purpose like the session ID, authorization token etc. In some cases, dynamic values also refer to the web content like values in a drop-down list, calendar date, item ID, product ID, order number etc. Through correlation, you can capture these dynamic values and pass in the subsequent requests. This is the basic concept of ‘Correlation’ in JMeter.

### ****Why do we need to correlate the dynamic values?****

To get the answer to this question, firstly you need to understand what exactly happens at the time of script recording, script replay and after correlating the dynamic values.

Let’s see, how the client and server act when some dynamic values are exchanged between them? This is the scenario while you record a script:

A screenshot of a computer

Description automatically generated with medium confidenceFigure 01: Recording Scenario

When you replay the script without any changes then the script fails because of the dynamic value (sessionid) generated by the server does not match with the value return back by the client. Refer to the given Figure 02, while replaying the script, the server-generated the session id as 222, but JMeter script sent the recorded value i.e. 111 which was captured during recording (Figure 01). Hence the server refused to serve the request and threw an error.

A screenshot of a computer

Description automatically generated with medium confidenceFigure 02: Replay without correlation Scenario

Now, when you correlate the dynamic value (sessionid) and replay the script then the JMeter script captures and saves the latest value generated by the server and sends back to the server in the next request. The server validates the returned value with generated value and gives the proper response. Hence the request is marked as passed at user end.

A screenshot of a computer

Description automatically generated with low confidenceFigure 03: Replay after Correlation

**What are the common values which require correlation?**

1. Session ID
2. Access Token
3. Customer Name / ID
4. Order Number
5. Bill Number
6. Number of records displayed on a page
7. Current Date and Time

There could be more values which depend on the type of the application and the term used to denote them. Keep in mind your ultimate goal must be to find out all the dynamic values which cause the failure of the script and correlate them. Now, the next question arises **‘HOW?’**

### ****How to handle dynamic value (correlation) in JMeter?****

There are 3 major steps in correlation which are:

1. Scripting
2. Identification (of dynamic value)
3. Implementation (of correlation logic)

We have already discussed the first step in the [previous article](https://www.perfmatrix.com/how-to-record-jmeter-script/), so now starting with the second step i.e. identification of dynamic values. In JMeter, two elements are very much required to identify the dynamic values and capture it in a variable. These elements are Postprocessor and Listener. Below is the list of postprocessors. You can use any of the postprocessors as per your convenience and requirement:

* Postprocessor
  + Regular Expression Extractor
  + Boundary Extractor
  + JSON Extractor
  + JSON JMESPath Extractor
  + CSS Selector Extractor
  + XPath Extractor XPath2 Extractor
* Listener
  + View Results Tree (Used during script recording)

**Steps for dynamic value identification:**

1. Add a ‘View Results Tree’ listener under thread group

Text

Description automatically generated with medium confidence

1. In the Thread Group, select the ‘Stop Thread’ option for ‘Action to be taken after a sampler error’.
2. Make sure the input value of Number of Threads (users), Ramp-up period (seconds), Loop Count should be 1.

Graphical user interface, text, application, email

Description automatically generated

1. Click the ‘Run’ button
2. The script may fail at a particular sampler where correlation is required.
3. Refer to the script and click on the same sampler which is failed during replay.
4. You will get some parameters and their values which are passing in the request
5. Copy the parameter value present under the ‘Value’ column.

Graphical user interface, text, application, email

Description automatically generated

Step 1: Refer to the recorded ‘View Results Tree’ listener

Step 2: Paste the copied value in the search field of recorded View Results Tree listener and press ‘Search’ button.

Step 3: JMeter will highlight the request where this value is available. Now select the very first highlighted request.

A picture containing diagram

Description automatically generated

Step 4: Click ‘Response Data’ tab.

Step 5: Under ‘Response Data’ tab, click ‘Response Body’ tab.

Step 6: Again paste the same value in the search field of Response Body.

Step 7: Click the’ Find’ button. Check the place where this dynamic value appeared in the response body. The searched value will be highlighted.

Graphical user interface, text, application, email

Description automatically generated

Step 8: Once JMeter finds out the value in the response body, then copy the string in such a way so that you can write a unique regular expression.

Learn regular expression from below:

<https://www.perfmatrix.com/regular-expression-regex-easy-to-learn/>

Step 9: Prepare a regular expression statement which can capture the dynamic value. In this example, the regular expression statement will be:

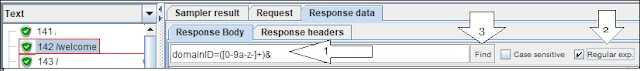
domainID=([0-9a-z-]+)&

Graphical user interface, text, application, email

Description automatically generated

Step 10: Verify the correctness of regular expression.

1. To verify the correctness of regular expression statement, write the same regular expression in the search box of the response of the sampler (Recorded View Results Tree Listener).
2. Checkmark the ‘Regular exp.’ option.
3. Press ‘Find’ button



**OR**

1. You can also select the ‘RegExp Tester’ from the drop-down of view results tree listener
2. Paste the regular expression statement
3. Click ‘Test’.

Text

Description automatically generatedFigure 11

The result in both the cases should highlight the correct dynamic value.

1. Repeat these steps until you find all the dynamic values.

Now, the third step is implementation.

**Steps to implement the correlation:**

1. For this purpose, add a regular expression extractor under the same sampler whose response contains the dynamic value and paste the regular expression statement (from step 9) in the ‘Regular Expression’ field. Also, provide the Name of created variable along with the Template, Match No. and Default Value.

[Read about ‘Regular Expression Extractor’ postprocessor](https://perfmatrix.blogspot.com/2017/01/apache-jmeter-regular-expression-extractor-postprocessor.html)

Graphical user interface, text, application

Description automatically generatedFigure 12: Regular Expression Extractor

1. The last step is to replace all the occurrence of dynamic values in the subsequent requests by ${<RegExVaribaleName>} syntax. Refer to the below screenshot:

[Graphical user interface, text, application, email

Description automatically generated](https://3.bp.blogspot.com/-6ESEHL-si9s/XNEzhqUi-5I/AAAAAAAAFJw/8uKvJwVbz546QCAq6yLP8YQmRPVpocFnwCEwYBhgL/s1600/JMeter%2BTest%2BExecution%2B11.JPG)Figure 13

1. Re-run the script and verify the correctness of correlation by referring the view results tree listener or by adding a debug sampler.

There could be 3 possible cases of the availability of dynamic values. Each case has a slightly different approach than others. Click the link to get full detail on each approach:

1. **Dynamic value in the request body**

In the correlation article of JMeter, you got the detailed knowledge on:

* What is Correlation?
* Why correlation is required?
* How to capture dynamic values?

The purpose of this article is to elaborate on the process of finding out the dynamic value when it is present in the request body. This is a general scenario because most of the time dynamic value is available in the request body and you can easily identify it.

### Steps to identify the dynamic value present in the request body:

1. Add a ‘View Results Tree’ listener under thread group

Text

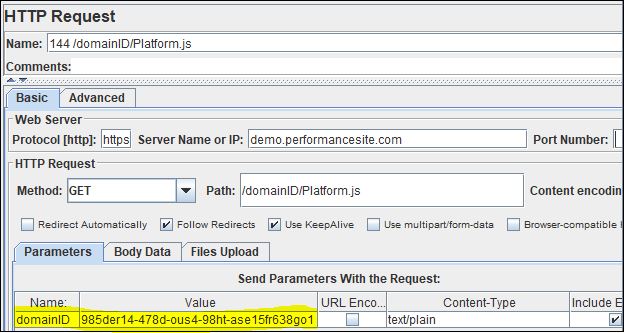
Description automatically generated with medium confidenceFigure 01

1. In the Thread Group, select the ‘Stop Thread’ option for ‘Action to be taken after a sampler error’.
2. Make sure the input value of Number of Threads (users), Ramp-up period (seconds), Loop Count should be 1.

Graphical user interface, text, application, email

Description automatically generatedFigure 02: Thread Group

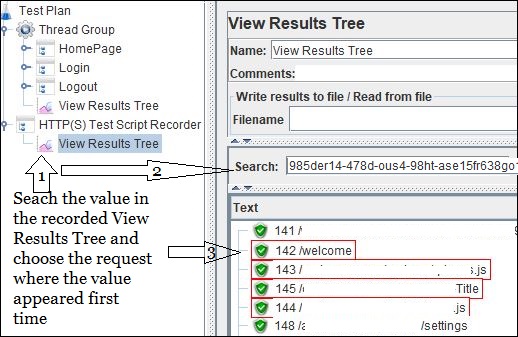
1. Click the ‘Run’ button
2. The script may fail at a particular sampler where correlation is required.
3. Refer to the script and click on the same sampler which is failed during replay.
4. You will get some parameters and their values which are passing in the request
5. Copy the parameter value present under the ‘Value’ column.

Figure 03: Sampler with Dynamic Value

Step 1: Refer to the recorded ‘View Results Tree’ listener

Step 2: Paste the copied value in the search field of recorded View Results Tree listener and press ‘Search’ button.

Step 3: JMeter will highlight the request where this value is available. Now select the very first highlighted request.

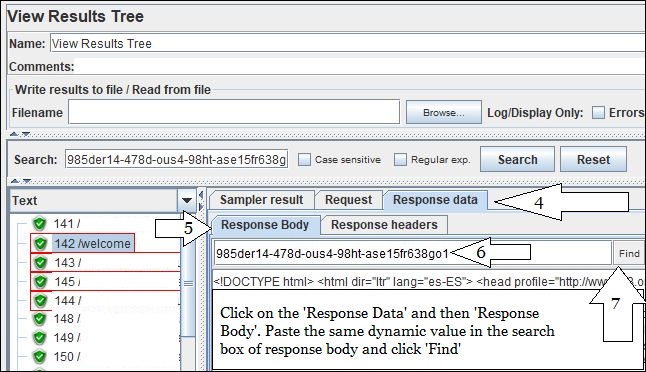
Figure 04: View Results Tree

Step 4: Click ‘Response Data’ tab.

Step 5: Under ‘Response Data’ tab, click ‘Response Body’ tab.

Step 6: Again paste the same value in the search field of Response Body.

Step 7: Click the ‘Find’ button. Check the place where this dynamic value appeared in the response body. The searched value will be highlighted.

Figure 05

Step 8: Once JMeter finds out the value in the response body, then copy the string in such a way so that you can write a unique regular expression.

[Learn – How to write the regular expression?](https://perfmatrix.blogspot.com/2016/12/regular-expression.html)

Step 9: Prepare a regular expression statement which can capture the dynamic value. In this example, the regular expression statement will be:

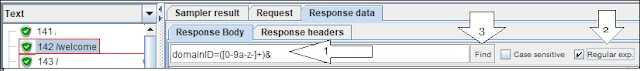
domainID=([0-9a-z-]+)&

Graphical user interface, text, application, email

Description automatically generatedFigure 06

Step 10: Verify the correctness of regular expression.

1. To verify the correctness of regular expression statement, write the same regular expression in the search box of the response of the sampler (Recorded View Results Tree Listener).
2. Checkmark the ‘Regular exp.’ option.
3. Press ‘Find’ button

Figure 07

**OR**

1. You can also select the ‘RegExp Tester’ from the drop-down of view results tree listener
2. Paste the regular expression statement
3. Click ‘Test’.

Text

Description automatically generatedFigure 08

The result in both the cases should highlight the correct dynamic value.

1. Repeat these steps until you find all the dynamic values.

Now, the next step is implementation.

**Steps to implement the correlation:**

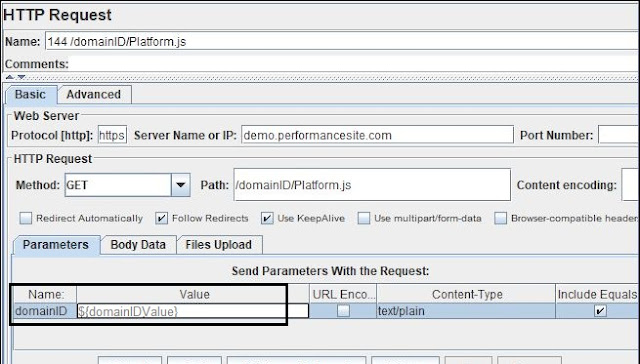
1. For this purpose, add a regular expression extractor under the same sampler whose response contains the dynamic value and paste the regular expression statement (from step 9) in the ‘Regular Expression’ field. Also, provide the Name of created variable along with the Template, Match No. and Default Value.

[Read about ‘Regular Expression Extractor’ postprocessor](https://perfmatrix.blogspot.com/2017/01/apache-jmeter-regular-expression-extractor-postprocessor.html)

Graphical user interface, text, application

Description automatically generatedFigure 09: Regular Expression Extractor

1. The last step is to replace all the occurrence of dynamic values in the subsequent requests by ${<RegExVaribaleName>} syntax. Refer to the below screenshot:

[](https://3.bp.blogspot.com/-6ESEHL-si9s/XNEzhqUi-5I/AAAAAAAAFJw/8uKvJwVbz546QCAq6yLP8YQmRPVpocFnwCEwYBhgL/s1600/JMeter%2BTest%2BExecution%2B11.JPG)Figure 10

1. Re-run the script and verify the correctness of correlation by referring the view results tree listener or by adding a debug sampler.
2. **Dynamic value in the request URL**

In the [correlation article](https://www.perfmatrix.com/how-to-handle-dynamic-value-correlation-in-jmeter/) of JMeter, you got the detailed knowledge on:

* What is Correlation?
* Why correlation is required?
* How to capture dynamic values?

The purpose of this article is to elaborate on the process of finding out the dynamic value when it is present in the request URL. This is a general scenario because most of the time dynamic value is available in the response body and you can easily identify it.

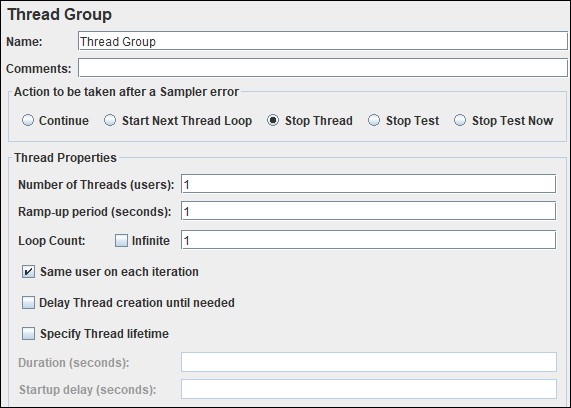
### Steps to identify the dynamic value present in the request URL:

1. Add a ‘View Results Tree’ listener under thread group

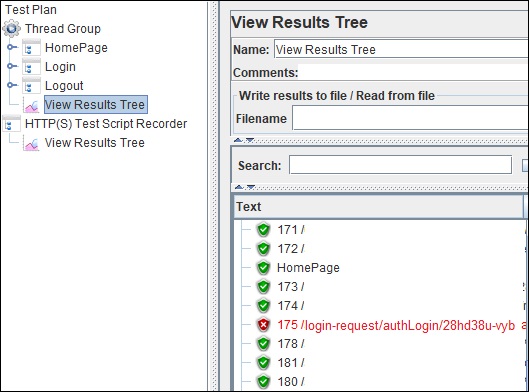
Text

Description automatically generated with medium confidenceFigure 01

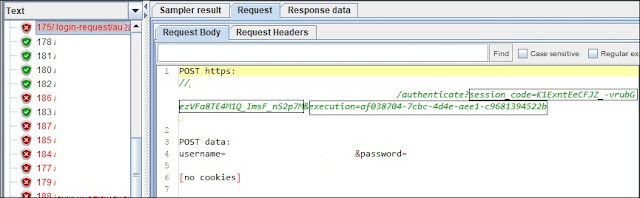
1. In the Thread Group, select the ‘Stop Thread’ option for ‘Action to be taken after a sampler error’.
2. Make sure the input value of Number of Threads (users), Ramp-up period (seconds), Loop Count should be 1.

Figure 02: Thread Group

1. Click the ‘Run’ button
2. The script may fail at a particular sampler where correlation is required.

Figure 03

1. Refer to the script and click on the same sampler which is failed during replay.
2. In case there is no value in the Parameters and Body Data tab of the sampler then refer to the request URL in ‘Path’ field.
3. If you identify the value in the URL then copy it.

Figure 04

Step 1: Refer to the recorded ‘View Results Tree’ listener. Paste the copied value in the search field of recorded View Results Tree listener and press ‘Search’ button.

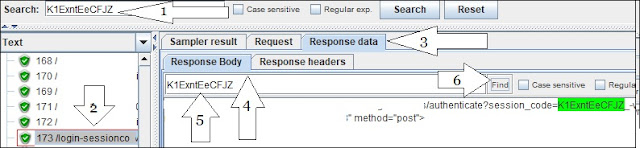
Step 2: JMeter will highlight the request where this value is available. Now select the very first highlighted request.

Step 3: Click ‘Response Data’ tab.

Step 4: Under ‘Response Data’ tab, click ‘Response Body’ tab.

Step 5: Again paste the same value in the search field of Response Body.

Step 6: Click the ‘Find’ button. Check the place where this dynamic value appeared in the response body. The searched value will be highlighted.

Figure 05

Step 7: Once JMeter finds out the value in the response body, then copy the string in such a way so that you can write a unique regular expression.

Step 8: Prepare a regular expression statement which can capture the dynamic value. In this example, the regular expression statement will be:

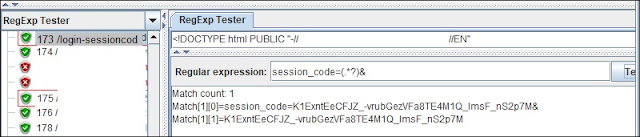
session\_code=([0-9a-z-]+)&

Step 9: Verify the correctness of regular expression.

1. To verify the correctness of regular expression statement, write the same regular expression in the search box of the response of the sampler (Recorded View Results Tree Listener).
2. Checkmark the ‘Regular exp.’ option.
3. Press ‘Find’ button

**OR**

1. You can also select the ‘RegExp Tester’ from the drop-down of view results tree listener
2. Paste the regular expression statement
3. Click ‘Test’.

Figure 06

The result in both the cases should highlight the correct dynamic value.

1. Repeat these steps until you find all the dynamic values.

Now, the next step is implementation.

**Steps to implement the correlation:**

1. For this purpose, add a regular expression extractor under the same sampler whose response contains the dynamic value and paste the regular expression statement (from step 8) in the ‘Regular Expression’ field. Also, provide the Name of created variable along with the Template, Match No. and Default Value.

[Read about ‘Regular Expression Extractor’ postprocessor](https://perfmatrix.blogspot.com/2017/01/apache-jmeter-regular-expression-extractor-postprocessor.html)

Graphical user interface, text

Description automatically generated with medium confidenceFigure 07

1. The last step is to replace all the occurrence of dynamic values in the subsequent requests by ${<RegExVaribaleName>} syntax. Refer to the below screenshot:

Graphical user interface, text, application, email

Description automatically generatedFigure 08

Re-run the script and verify the correctness of correlation by referring the view results tree listener or by adding a debug sampler.

1. **Dynamic value in the redirected request**

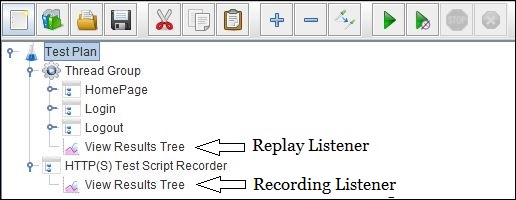
In the correlation article of JMeter, you got the detailed knowledge on:

* What is Correlation?
* Why correlation is required?
* How to capture dynamic values?

The purpose of this article is to elaborate on the process of finding out the dynamic value when it is present in the redirected request.

### Steps to identify the dynamic value present in the redirected request:

1. Add a ‘View Results Tree’ listener under thread group

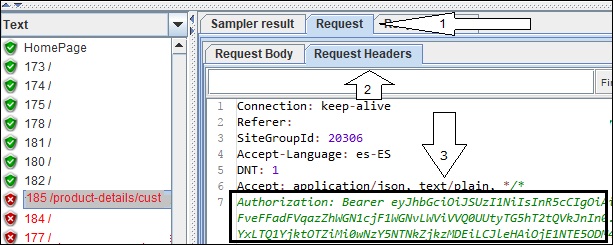
Figure 01

1. In the Thread Group, select the ‘Stop Thread’ option for ‘Action to be taken after a sampler error’.
2. Make sure the input value of Number of Threads (users), Ramp-up period (seconds), Loop Count should be 1.

Graphical user interface, text, application, email

Description automatically generatedFigure 02: Thread Group

1. Click the ‘Run’ button
2. The script may fail at a particular sampler where correlation is required.

Figure 03

1. Check the response code of the main request, if it is 302/Redirect then you need to refer to the ‘View Results Tree’ of recording log
2. In this case, the dynamic value appears either in the body or header of the redirected request.
3. Generally, these dynamic values fall in the below categories:
   1. Authorization Token
   2. Access Token
   3. CSRF token
   4. Security code
4. Copy the dynamic value.

Step 1: Refer to the recorded ‘View Results Tree’ listener

Step 2: Paste the copied value in the search field of recorded View Results Tree listener and press ‘Search’ button.

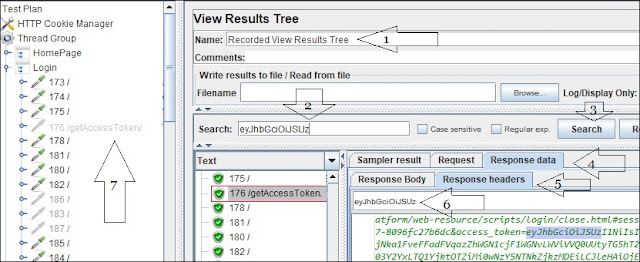
Step 3: JMeter will highlight the request where this value is available. Now select the very first highlighted request.

Step 4: Click ‘Response Data’ tab.

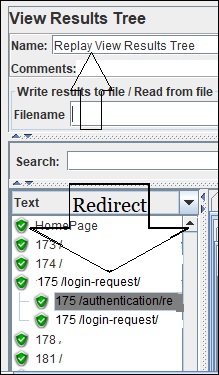
Step 5: Under ‘Response Data’ tab, click ‘Response Header’ or ‘Response Body’ tab (wherever dynamic value is present).

Step 6: Again, paste the same value in the search field of Response Header/Body and click the ‘Find’ button. Check the place where this dynamic value appeared in the response header/body. The searched value will be highlighted.

Step 7: You can see the redirected sampler will be disabled in the test plan tree.

Figure 04

1. In this special case, the request sampler under the thread group is greyed out. This is because the request is re-directed by the previous sampler. JMeter disabled this sampler so that direct request cannot be sent while replaying and the previous request should be redirected automatically to this request which is appeared as a child request in the replay ‘View Results Tree’ listener.

Figure 05

1. Once JMeter finds out the value then copy the string in such a way so that you can write a unique regular expression.
2. Prepare a regular expression statement which can capture the dynamic value. In this example, the regular expression statement will be:  
   access\_token=(.\*?)&

Now, the next step is implementation.

**Steps to implement the correlation:**

1. For this purpose, add a regular expression extractor under the same sampler whose response contains the dynamic value and paste the regular expression statement (from step 12) in the ‘Regular Expression’ field. Also, provide the Name of created variable along with the Template, Match No. and Default Value. One point to be noted here is the scope of dynamic value search would be ‘Main sample and sub-samples’ also in some specific cases, verify ‘Field to check’ scope through recorded View Results Tree listener and select the correct option.

A picture containing text

Description automatically generatedFigure 06

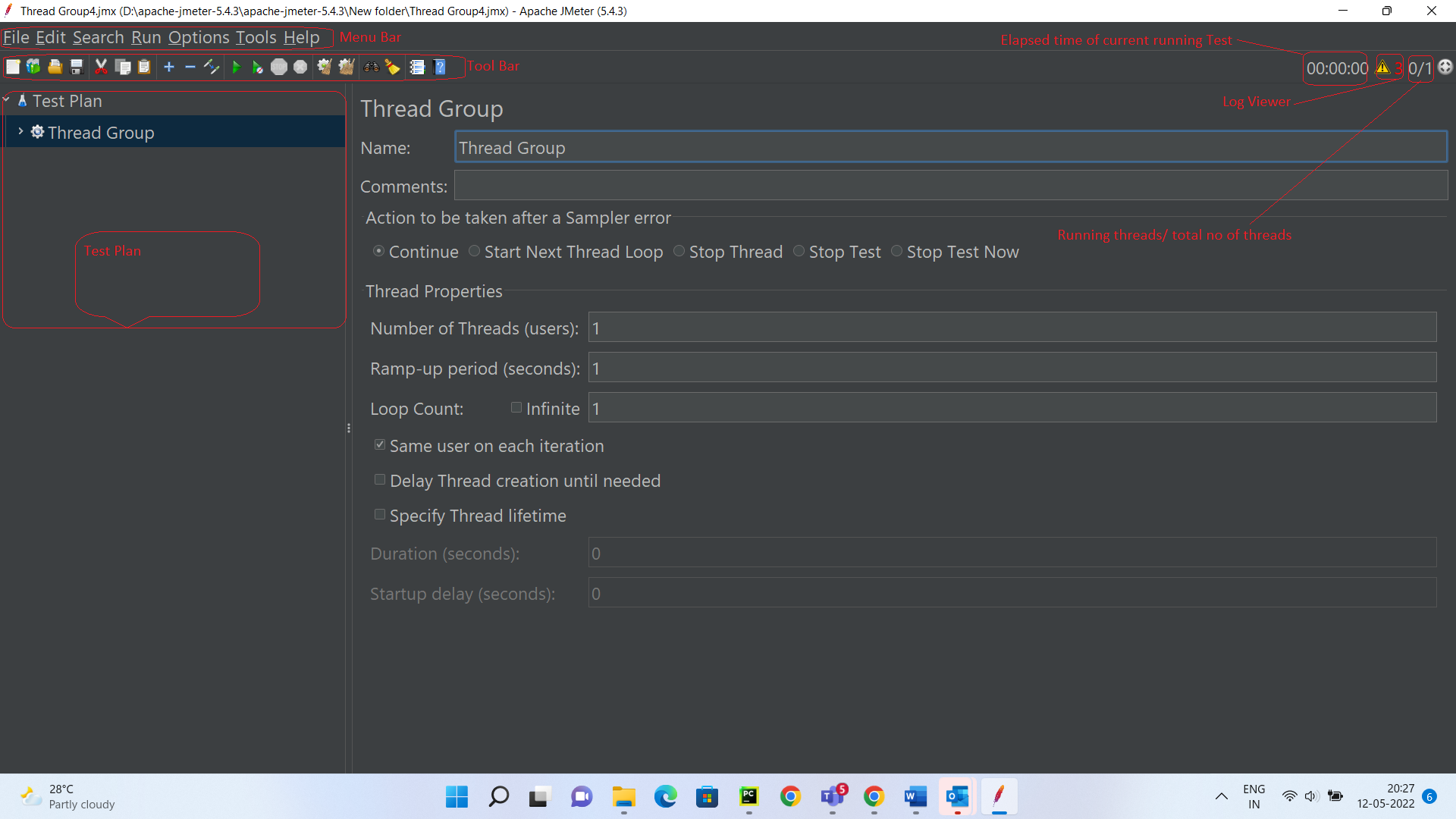
1. The last step is to replace all the occurrence of dynamic values in the subsequent requests by ${<RegExVaribaleName>} syntax. Refer to the below screenshot:

Graphical user interface, text, application

Description automatically generatedFigure 07

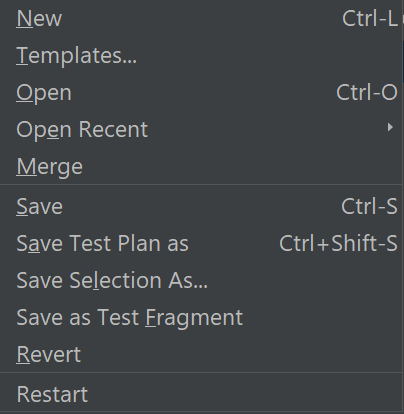
1. Re-run the script and verify the correctness of correlation by referring the view results tree listener or by adding a debug sampler.

**JMeter GUI Overview?**



Menu Bar:

File Menu: It consists following options



* **New**: This option is used to create new test plan
* **Templates**: This option is used to add a template for test plan
* **Open**: This option is used to open .jmx files
* **Open** Recent: This option is used to open recent worked files
* **Merge**: This option is used to merge the test plan with another .jmx file
* **Save**: This option is used to save the test plan
* **Save Test Plan as**: This option is used to save the test plan with specified name
* **Save Selection As**: This option is used to save the selected Test Plan/Thread Group/Transaction in the test plan
* **Save As Test Fragment**: Test Fragment element is a special controller which can be added directly under JMeter test plan like Thread Group. But it does nothing except holding other elements inside!! It gets executed only when it is referenced by a Module/Include controller from other Thread Groups
* **Revert**: This option is used to revert test plan elements to last saved.
* **Restart**: This option is used to restart JMeter.