

Nhóm 08

JMETER



Bộ môn: Kiểm chứng phần mềm

GV: thầy Lâm Quang Vũ

2022 - 2023

Mục lục

CHƯƠNG I JMeter là gì và tại sao chúng ta phải sử dụng JMeter?.....	4
Ưu điểm.....	4
JMeter hoạt động như thế nào?	5
CHƯƠNG II Tải và cài đặt Jmeter	6
Bước 1: Cài đặt Java.....	6
Bước 2: Tải Jmeter	6
Bước 3: Cài đặt	7
Bước 4: Khởi chạy Jmeter	7
CHƯƠNG III Các Elements trong Jmeter.....	11
Thread Group	11
Samplers.....	11
Listeners	12
Yếu tố cấu hình (Configuration Elements).....	14
So sánh Login Config Element vs. CSV Data Config:.....	15
CHƯƠNG IV JMeter GUI: Test plan và Workbench	16
Test Plan	16
WorkBench	17
CHƯƠNG V Cách sử dụng JMeter cho việc kiểm thử hiệu năng và tải trọng	22
JMeter Load Testing.....	22
JMeter Performance Testing.....	22
Các bước tạo một Performance Test Plan trên JMeter	23
CHƯƠNG VI Jmeter Timers: Constant, Gaussian Random, Uniform	31
Bộ hẹn giờ là gì?	31

Cách sử dụng Constant Timer	33
CHƯƠNG VII Cách dùng Assertions trong Jmeter	38
Assertion là gì?.....	38
Các loại Assertion.....	38
CHƯƠNG VIII Controllers in JMeter: Loop, Simple, Transaction, Module, Random	46
Logic Controller là gì ?.....	46
Recording Controller:.....	46
Simple Controller:	47
Loop Controller:.....	47
Random Controller:	48
Module Controller:	49
CHƯƠNG IX Processor in JMeter: PreProcessor & PostProcessor	55
Pre-processor	55
Post-processor	55
CHƯƠNG X Jmeter Distributed (Remote) Testing: Master Slave Configuration .	62
Distributed Testing	62
CHƯƠNG XI HTTP Proxy Server in JMeter: Record Example Script	66
Bước 1: Đặt máy chủ HTTP Proxy.....	67
Bước 2: Ghi lại hoạt động	74
Bước 3: Chạy Test Plan	77
Bước 4: Lưu kết quả test.....	79
CHƯƠNG XII Các phương pháp tốt nhất để luyện tập JMeter Test và Load Testing.....	81
Kiểm tra JMeter là gì?	81
Giới hạn số lượng Threads.	82

Sử dụng Proxy server	82
Sử dụng biến.....	82
Giảm yêu cầu tài nguyên.....	82
Kiểm tra nhật ký JMeter.....	83
Xóa đường dẫn cục bộ khỏi Cấu hình tập dữ liệu CSV.....	83
Thực hiện theo quy ước đặt tên tệp	83

JMeter

CHƯƠNG I

JMeter là gì và tại sao chúng ta phải sử dụng JMeter?

- Apache JMeter là phần mềm mã nguồn mở hoàn toàn bằng Java được thiết kế để kiểm tra tải (load) và hiệu suất (performance) của chức năng.
 - Ví dụ: JMeter cũng có thể mô phỏng tải nặng trên máy chủ bằng cách tạo hàng tấn người dùng ảo đồng thời lên máy chủ web.
- Performance Testing có nghĩa là kiểm thử ứng dụng web với lượng tải nặng, lưu lượng người dùng nhiều và đồng thời.
- JMeter được dùng để test ứng dụng web và FTP application (File Transfer Protocol)
- Giao thức truyền tập tin. -> giờ được sử dụng để kiểm thử chức năng(functional test) (là một trong các quy trình đảm bảo chất lượng của lĩnh vực kiểm thử phần mềm) và kiểm thử cơ sở dữ liệu (database server test).

Ưu điểm

1. Mã nguồn mở.
2. GUI thân thiện: dễ sử dụng.
3. Nền tảng độc lập: 100% Java desktop application, chạy trên đa nền tảng.
4. Đầy đủ framework đa luồng: cho phép mẫu thử diễn ra đồng thời, song song các chức năng khác nhau bởi những luồng riêng biệt.
5. Trực quan hóa kết quả kiểm tra: như thể hiện kết quả kiểm tra ở dạng biểu đồ, bảng, cây và các tệp text, XML, HTML and JSON.

6. Cài đặt dễ dàng: Bạn chỉ cần copy và chạy file *.bat để chạy JMeter. Không cần cài đặt.
7. Khả năng mở rộng cao: Bạn có thể viết các bài kiểm tra của riêng mình. JMeter cũng hỗ trợ các plugin trực quan hóa cho phép bạn mở rộng thử nghiệm của mình
8. Hỗ trợ đa giao thức: JMeter hỗ trợ một vài giao thức như HTTP, FTP, SOAP, JDBC, JMS và LDAP. Nó cũng có thể được sử dụng để kiểm thử hiệu suất của cơ sở dữ liệu của bạn.
9. Nhiều chiến lược thử nghiệm: Load Testing, Distributed Testing (thử nghiệm phân tán), Functional testing.
10. Tính mô phỏng: JMeter có thể mô phỏng nhiều người dùng với các luồng đồng thời, tạo tải nặng cho ứng dụng web đang được kiểm tra.
11. Record & Playback – Ghi lại hoạt động của người dùng trên trình duyệt và mô phỏng chúng trong ứng dụng web bằng JMeter
12. Kiểm tra tập lệnh: Jmeter có thể được tích hợp với Bean Shell & Selenium để kiểm tra tự động.

JMeter hoạt động như thế nào?

1. Jmeter tạo ra các yêu cầu và gửi chúng lên server giống như trình duyệt web yêu cầu một trang
2. Nó nhận được phản hồi từ server, thu thập chúng và hiển thị những chi tiết đó trong biểu đồ hoặc đồ thị.
3. Nó xử lý phản hồi từ server.
4. Nó tạo ra kết quả thử nghiệm theo một số định dạng như text, XML, JSON. Sau đó, tester có thể phân tích dữ liệu

CHƯƠNG II

Tải và cài đặt Jmeter

Bước 1: Cài đặt Java

- [Download Java Platform \(JDK\)](#)

- Sau khi cài đặt thành công bạn có thể kiểm tra bằng cách vào **Terminal** và gõ **java -version**.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22621.1702]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL>java -version
java version "11.0.18" 2023-01-17 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.18+9-LTS-195)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.18+9-LTS-195, mixed mode)
```

Bước 2: Tải Jmeter

- [Download JMeter](#)

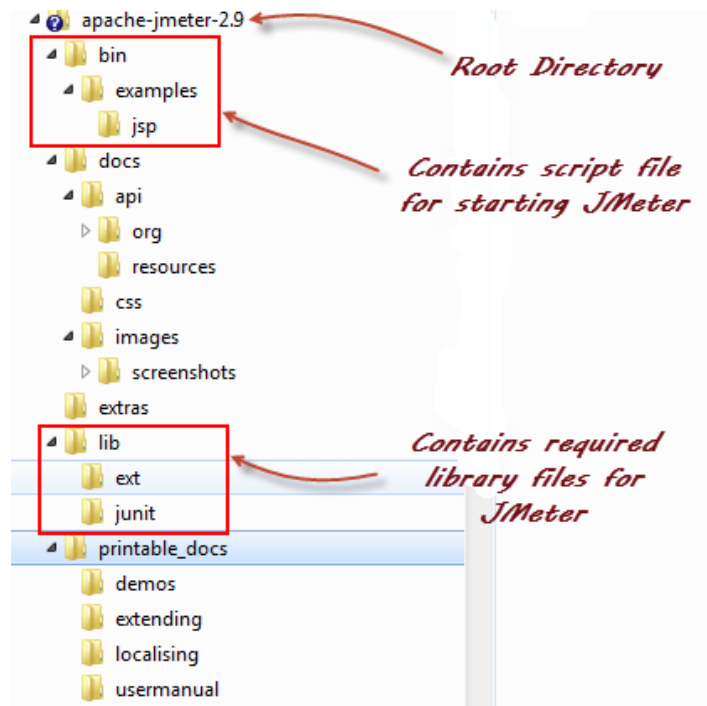
- Chọn một trong hai loại này:

Apache JMeter 5.5 (Requires Java 8+)

Binaries

[apache-jmeter-5.5.tgz sha512 pgp](#)
[apache-jmeter-5.5.zip sha512 pgp](#)

Bước 3: Cài đặt



/bin: Chứa tập lệnh JMeter để khởi động JMeter

/docs: Tập tài liệu JMeter

/extras: tập tin bổ sung liên quan đến kiến

/lib/: Chứa thư viện Java cần thiết cho JMeter

/lib/ext: chứa các tập jar cốt lõi cho JMeter và các giao thức

/lib/junit: Thư viện Junit dùng cho JMeter

/printable_docs

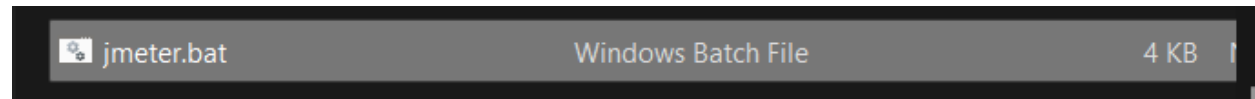
Bước 4: Khởi chạy Jmeter

- Có 3 cách khởi chạy JMeter:

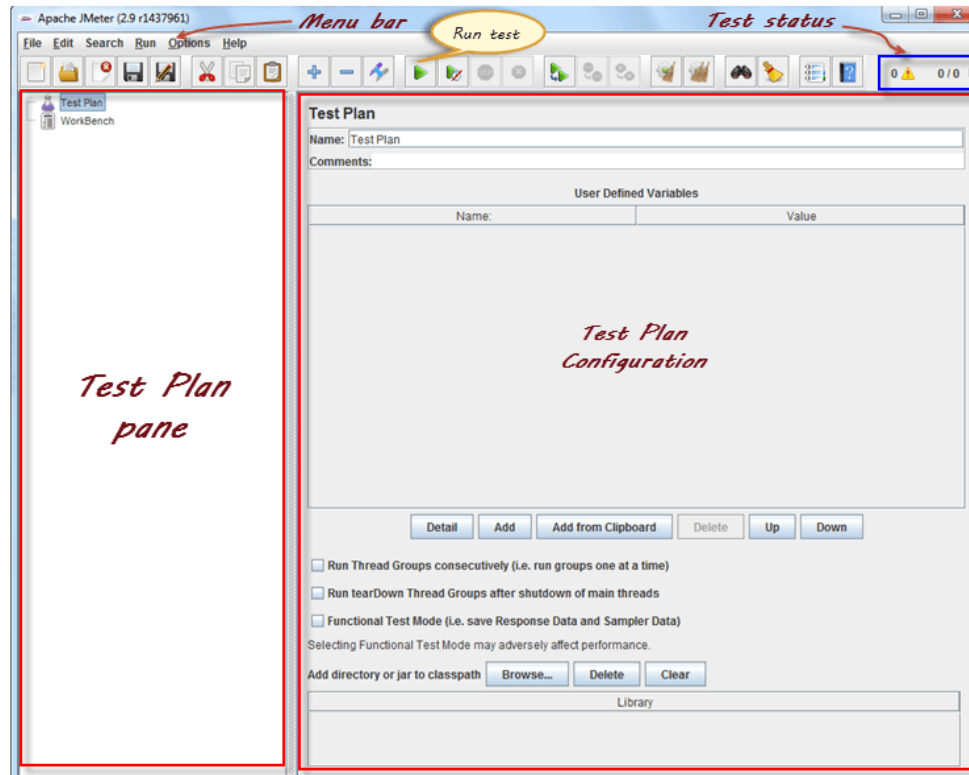
- GUI Mode
- Server Mode
- Command Line Mode

- GUI Mode:

Khởi chạy file **/bin/jmeter.bat**



apache-jmeter-2.9	8/21/2013 10:35 PM	File folder
Download	8/27/2013 10:15 PM	File folder
Entertainment	8/8/2013 10:41 PM	File folder
home	7/21/2013 9:07 PM	File folder
Intel	1/3/2008 4:55 AM	File folder
MSOCCache	1/3/2008 12:33 PM	File folder
Nguyen	8/27/2013 7:21 PM	File folder
PerfLogs	7/14/2009 10:20 AM	File folder
Perl	2/24/2013 11:41 AM	File folder
Program Files	8/17/2013 10:57 AM	File folder
Program Files (x86)	8/21/2013 10:09 PM	File folder
ProgramData	8/7/2013 7:22 PM	File folder
Repositories	8/5/2013 8:41 PM	File folder
Share	8/24/2012 10:33 PM	File folder
Users	1/3/2008 6:01 AM	File folder
Van	8/17/2013 7:08 PM	File folder
Windows	8/11/2013 8:05 AM	File folder
Windows.old	1/3/2008 5:36 AM	File folder
work	6/26/2013 7:14 PM	File folder



- Non GUI Mode:

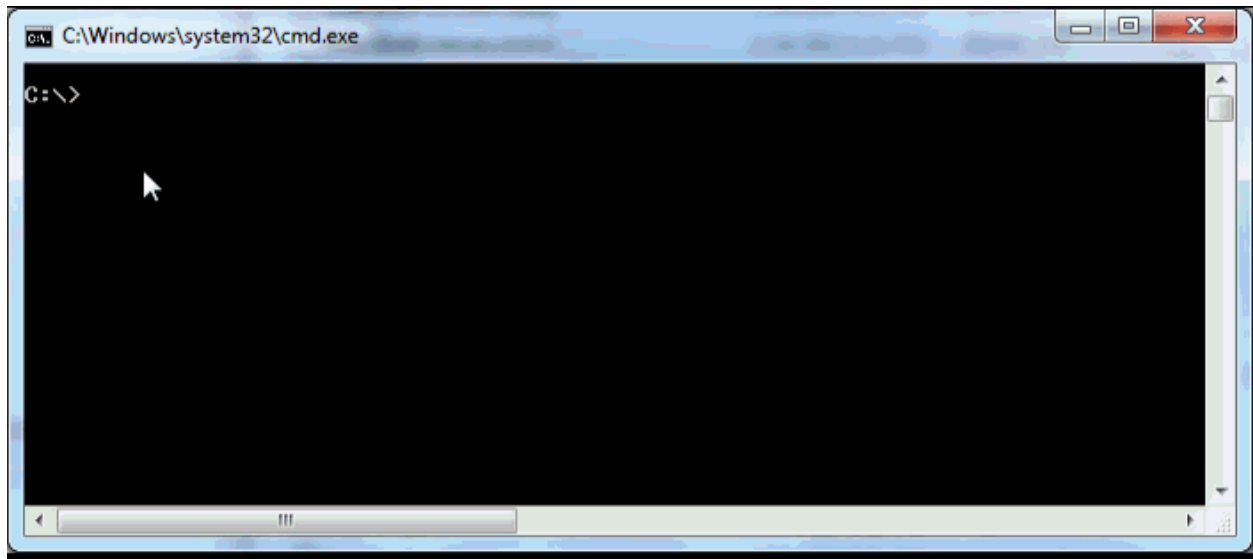
- Server Mode:

Khởi chạy bat file: `bin\jmeter-server.bat`



- Command line mode:

`$jmeter -n -t testPlan.jmx -l log.jtl -H 127.0.0.1 -P 8000`



Additional Packages

- Java Compiler
- SAX XML parser
- Email Support
- JDBC driver

CHƯƠNG III

Các Elements trong Jmeter

Thread Group

- Thread Group là một nhóm các Luồng. Mỗi luồng đại diện cho một người dùng sử dụng ứng dụng đang được kiểm tra. Mỗi luồng mô phỏng một yêu cầu từ người dùng thực tế tới máy chủ.

- Các điều khiển cho một nhóm luồng cho phép bạn đặt số lượng luồng cho mỗi nhóm.

Ví dụ: nếu bạn đặt số luồng là 100, JMeter sẽ tạo và mô phỏng 100 yêu cầu từ 100 người dùng tới máy chủ đang được kiểm tra.

Samplers

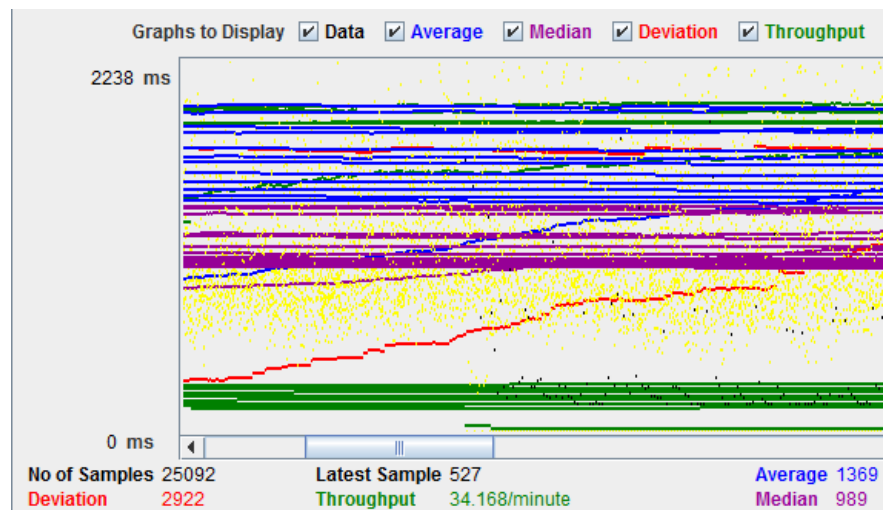
- Sampler là thành phần thực hiện các yêu cầu đến máy chủ hoặc ứng dụng được kiểm tra.

- FTP request: Sử dụng để thực hiện yêu cầu download hoặc upload file từ máy chủ FTP.
- HTTP request: Sử dụng để gửi yêu cầu HTTP/HTTPS tới máy chủ web.
- JDBC request: Sử dụng để thực thi các truy vấn SQL tới cơ sở dữ liệu.
- BSF Sampler: Cho phép viết một sampler bằng ngôn ngữ kịch bản BSF.
- Access Log Sampler: Đọc access log và tạo các yêu cầu HTTP.
- SMTP Sampler: Sử dụng để gửi email sử dụng giao thức SMTP.

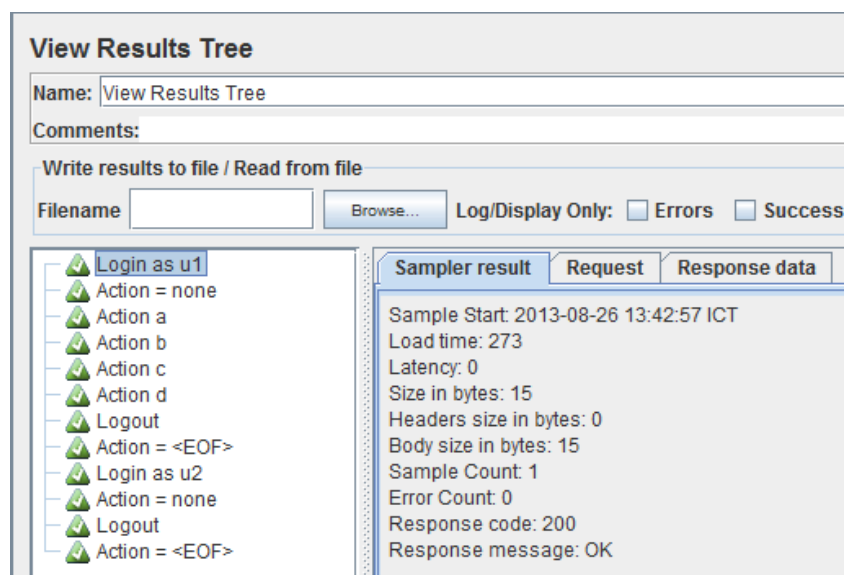
Listeners

Listeners: hiển thị kết quả của việc thực hiện kiểm thử, được hiển thị kết quả ở một định dạng khác, chẳng hạn như cây, bảng, biểu đồ hoặc tệp log.

- Graph (biểu đồ)



- Tree (cây)



- Table (bảng)

View Results in Table

Name:

View Results in Table

Comments:

Write results to file / Read from file

Filename

Browse...

Log/D

Sample #	Start Time	Thread Name	Label	Sample Time(ms)
1	11:20:29.282	Thread Group 1-1	HTTP Request	1430
2	11:20:31.714	Thread Group 1-1	HTTP Request	1490
3	11:20:34.206	Thread Group 1-1	HTTP Request	534
4	11:20:35.743	Thread Group 1-1	HTTP Request	1966
5	11:20:38.714	Thread Group 1-1	HTTP Request	1247
6	11:20:40.964	Thread Group 1-1	HTTP Request	1140
7	11:20:43.107	Thread Group 1-1	HTTP Request	1631
8	11:20:45.740	Thread Group 1-1	HTTP Request	683

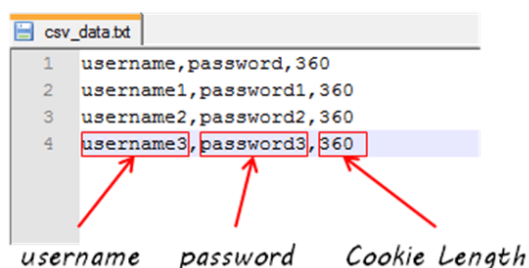
- Log file

sampler_label	aggregate	average	aggregate
/v6exp3/redir.html	187	3517	185
/v6exp3/iframe.htm	168	1595	165
/first-android-testin	94	6009	1581
/search/adi/g.php	101	2999	21
/quality-center-tuto	67	3292	146
/b	41	3220	119
/bn/at_300.html	12	2174	1108
/getting-started-wit	8	2115	872
/sql.html	1	908	908
TOTAL	679	3225	21

Yếu tố cấu hình (Configuration Elements)

- **Cấu hình tập dữ liệu CSV:**

Cho phép đọc các thông tin từ tệp văn bản, chẳng hạn như tên người dùng và mật khẩu. Bằng cách sử dụng nó, bạn có thể tham số hóa kịch bản để thử nghiệm đăng nhập với nhiều thông tin khác nhau mà không cần ghi lại kịch bản nhiều lần.



- **Trình quản lý cookie HTTP:**

Tự động lưu trữ và sử dụng các cookie từ phản hồi HTTP để xác thực trong các yêu cầu sau này. Nó giống như cách trình duyệt web hoạt động với cookies.

- **Yêu cầu HTTP mặc định:**

HTTP Request Defaults trong JMeter cho phép bạn thiết lập giá trị mặc định cho các yêu cầu HTTP. Thay vì phải nhập giá trị cho mỗi yêu cầu riêng biệt, bạn có thể chỉ cần cấu hình một lần duy nhất trong HTTP Request Defaults. Các yêu cầu HTTP tiếp theo sẽ tự động sử dụng giá trị mặc định này, giúp tiết kiệm thời gian và công sức trong việc thiết lập các yêu cầu.

HTTP Request Defaults

Name: HTTP Request Defaults

Comments:

Web Server

Server Name or IP: google.com Port Number: 80

Timeouts (milliseconds)

Connect: 1000 Response: 2000

HTTP Request

Implementation: Protocol [http]: Content encoding:

- **Cấu hình đăng nhập:**

Cho phép bạn thêm hoặc ghi đè cài đặt tên người dùng và mật khẩu trong các yêu cầu người dùng. Bằng cách sử dụng nó, bạn có thể định nghĩa tên người dùng và mật khẩu để mô phỏng đăng nhập vào một trang web.

Login Config Element

Name: Login Config Element

Comments:

Username: guru99

Password:

username and password setting

So sánh Login Config Element vs. CSV Data Config:

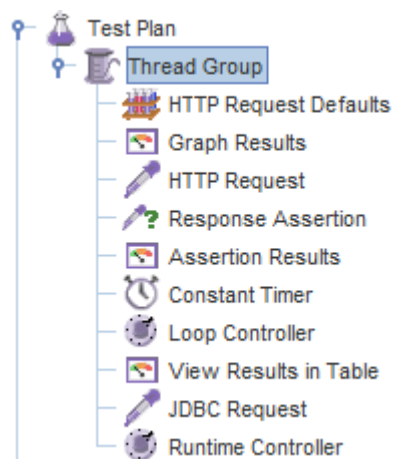
- Login Config Element được sử dụng để mô phỏng đăng nhập của một người dùng duy nhất, thích hợp cho việc đăng nhập với các tham số cố định như tên người dùng và mật khẩu.
- CSV Data Config được sử dụng để mô phỏng đăng nhập của nhiều người dùng, thích hợp cho việc đăng nhập với nhiều tham số và số lượng lớn. Nó cho phép bạn đọc các tham số từ một tệp CSV để sử dụng cho các yêu cầu của người dùng.

CHƯƠNG IV

JMeter GUI: Test plan và Workbench

Test Plan

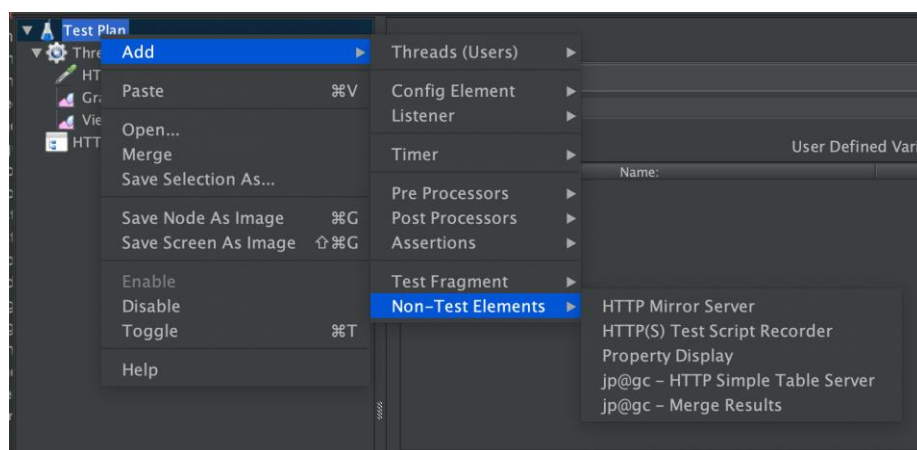
- Là nơi để thêm các yếu tố cần thiết cho JMeter Test.
- Nó lưu trữ tất cả các thành phần (như ThreadGroup, Timers, v.v.) và cài đặt tương ứng của chúng cần thiết để chạy các Thử nghiệm mong muốn của bạn.



WorkBench

Chỉ đơn giản là cung cấp một nơi để tạm thời lưu trữ các phần tử thử nghiệm. WorkBench không liên quan đến Test Plan. JMeter sẽ không lưu nội dung của WorkBench. Nó chỉ lưu nội dung của nhánh Test Plan.

Bạn có thể tìm thấy tất cả các tính năng của Workbench tại đây:



Adding các elements

Giả sử, bạn muốn thêm 2 yếu tố vào Test Plan BeanShell Assertion và Java Request Default.

Nhấp chuột phải vào *Test Plan* -> *Add* -> *Assertion*-> *Bean Shell Assertion*.

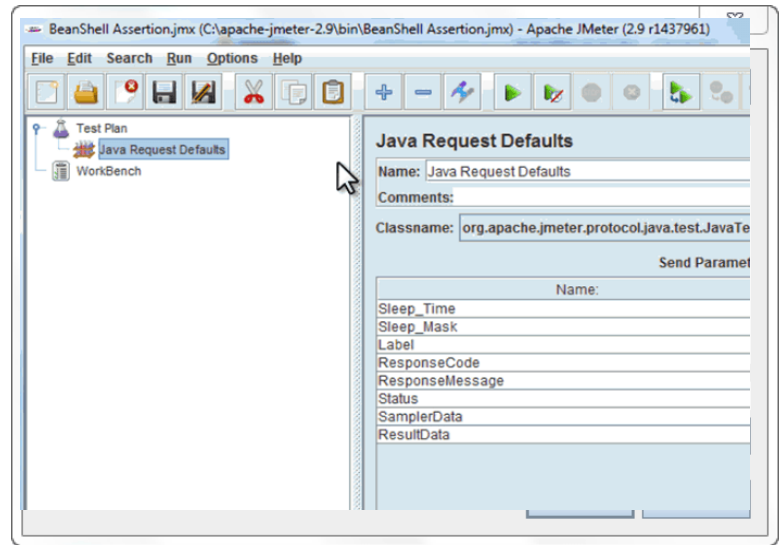
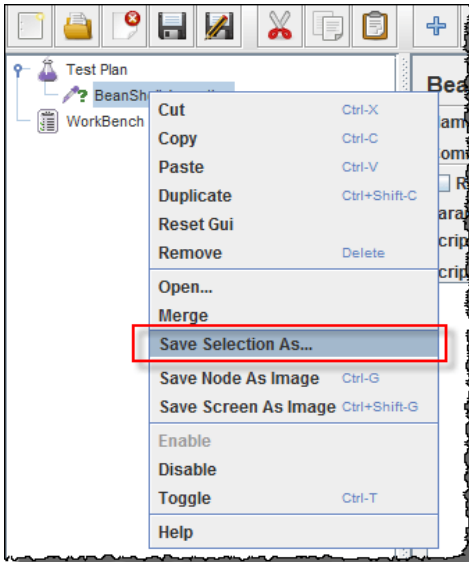
Nhấp chuột phải vào *Test Plan* -> *Add* -> *Config Element* -> *Java Request Default*.

Loading và Saving các elements

1. Tạo JMX file

Click chuột phải vào *BeanShell Assertion* -> chọn *Save Selection As*

(JMX là viết tắt của Java Management Extensions).

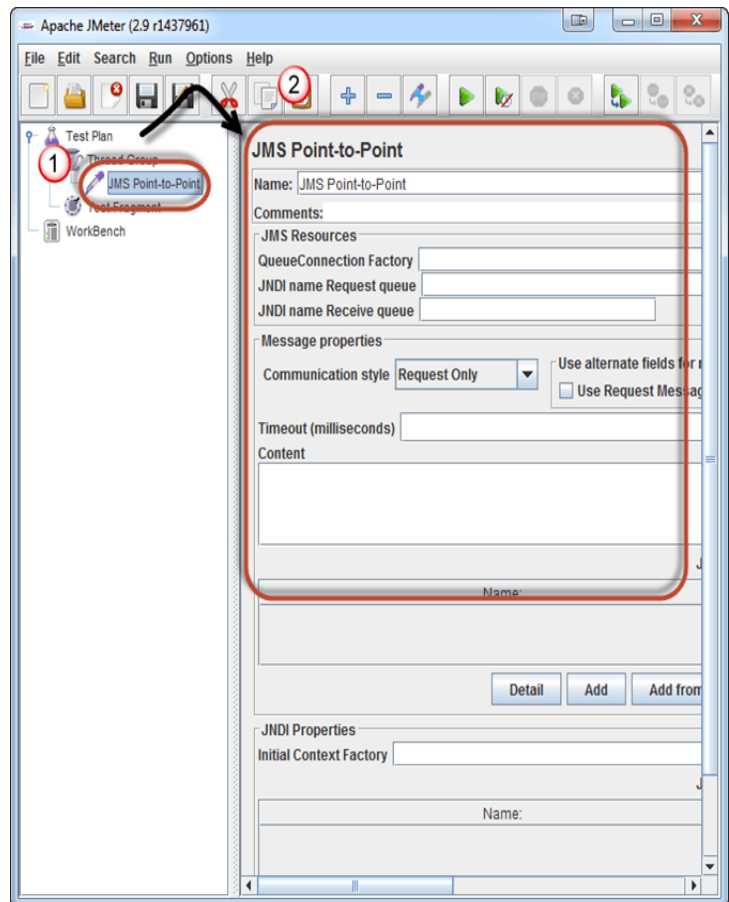


2. Run JMX file

1. Nhấp chuột phải vào *Java Request Defaults* -> chọn *Merge*.
2. Chọn tệp Elements (BeanShell Assertion.jmx.) trong thư mục. Yếu tố này sẽ được thêm vào kế hoạch thử nghiệm hiện tại của bạn.

3. Configure Elements (Cấu hình)

1. Chọn phần tử trong Tree on Left Pane
2. Nhập cài đặt cấu hình trên Khung bên phải



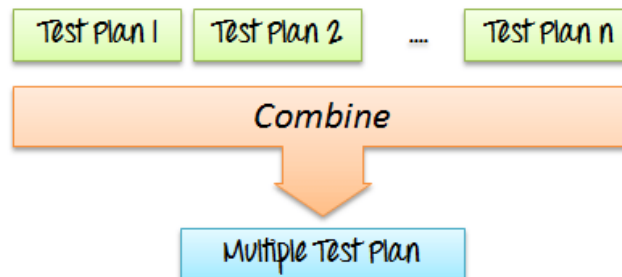
4. Lưu Test Plan

1. File -> Save Test Plan -> hiển thị Dialog box.
2. Nhập tên tệp Test Plan -> click Save.

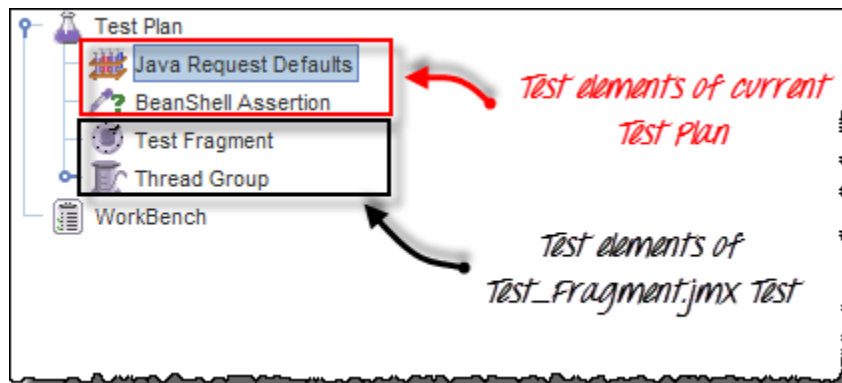
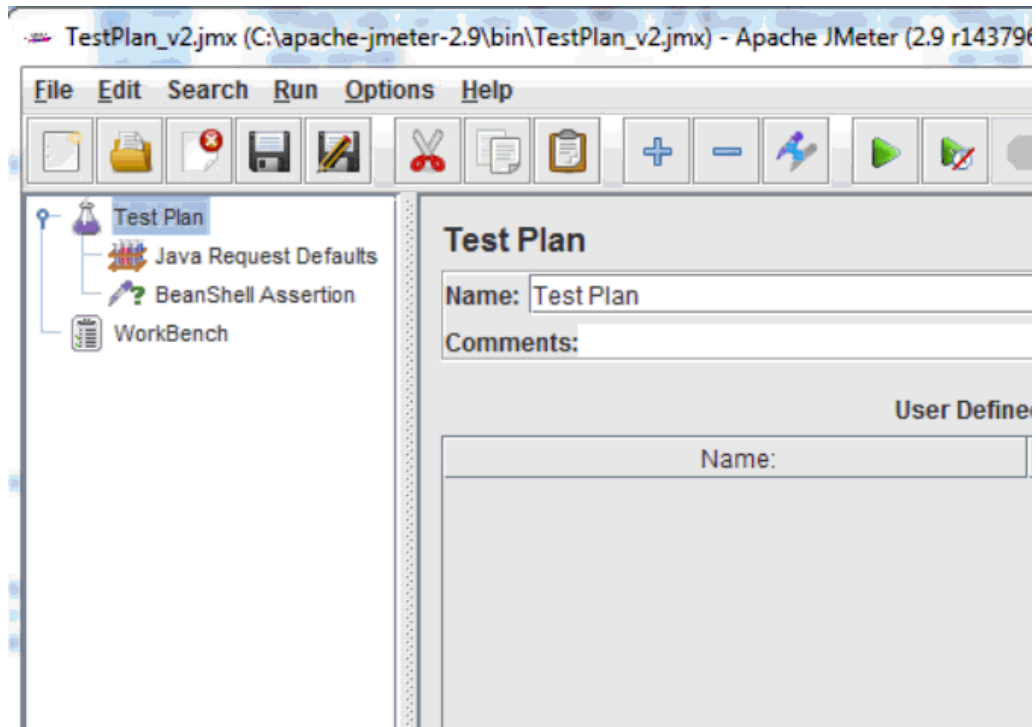
Lưu ý: Lưu Test Plan khác với lưu elements.

Lưu Test Plan	Lưu elements
Test plan bao gồm một hoặc nhiều elements	Element là một thành phần cơ bản của JMeter
Khi bạn lưu Test plan của mình, tất cả các element trong kế hoạch sẽ được lưu.	Khi bạn lưu các element của mình, chỉ có một element được lưu.

5. Tạo Combo Test Plan

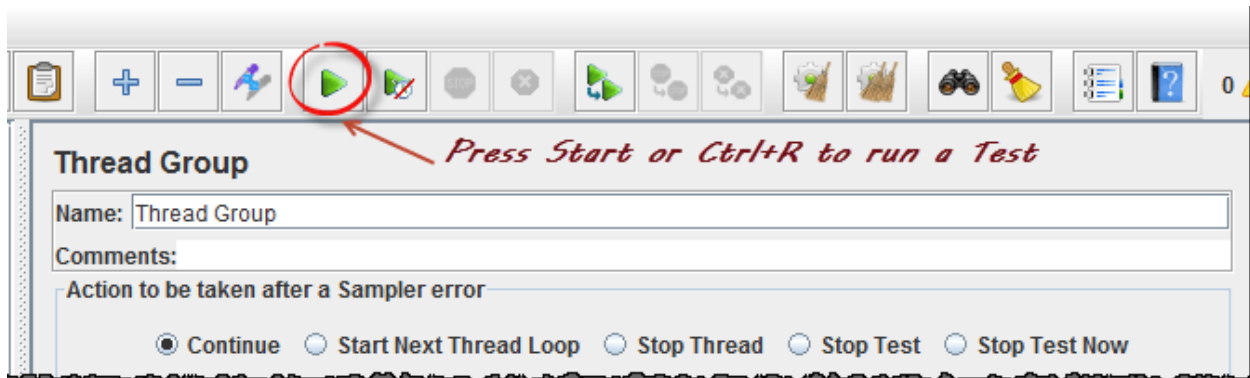


Contains all elements of each test plan



6. Run Test Plan

Chọn *Start (Ctrl + R)* từ mục menu Run



Khi JMeter đang chạy, nó sẽ hiển thị một hộp nhỏ màu xanh lá cây ở cuối bên phải của thanh menu.



Các số ở bên trái hộp màu xanh lá cây là số chuỗi đang hoạt động/tổng số chuỗi. Để Stop test, nhấn nút Dừng hoặc sử dụng phím tắt Ctrl + '.'



CHƯƠNG V

Cách sử dụng JMeter cho việc kiểm thử hiệu năng và tải trọng

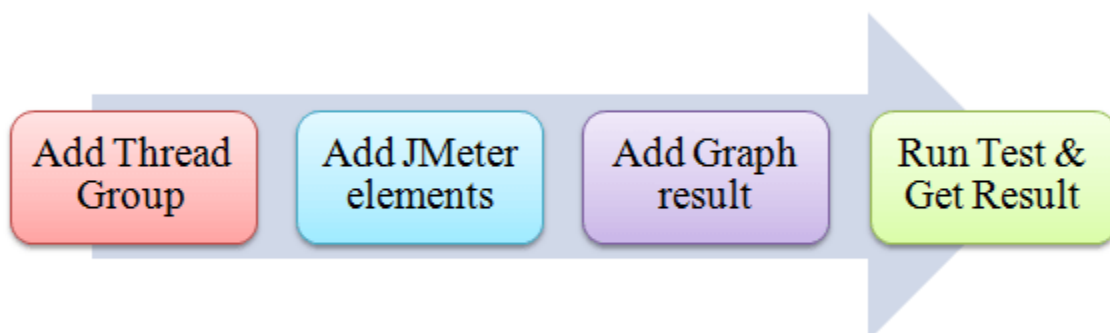
JMeter Load Testing

JMeter Load Testing là một quy trình kiểm thử được thực hiện bằng cách sử dụng công cụ kiểm thử tải trọng được gọi là Apache JMeter, một ứng dụng desktop mã nguồn mở dựa trên Java. JMeter For Load Testing là một công cụ quan trọng để xác định xem ứng dụng web đang được kiểm thử có thể đáp ứng được yêu cầu tải cao hay không. Nó cũng giúp phân tích tổng thể hiệu năng máy chủ khi có tải trọng nặng.

JMeter Performance Testing

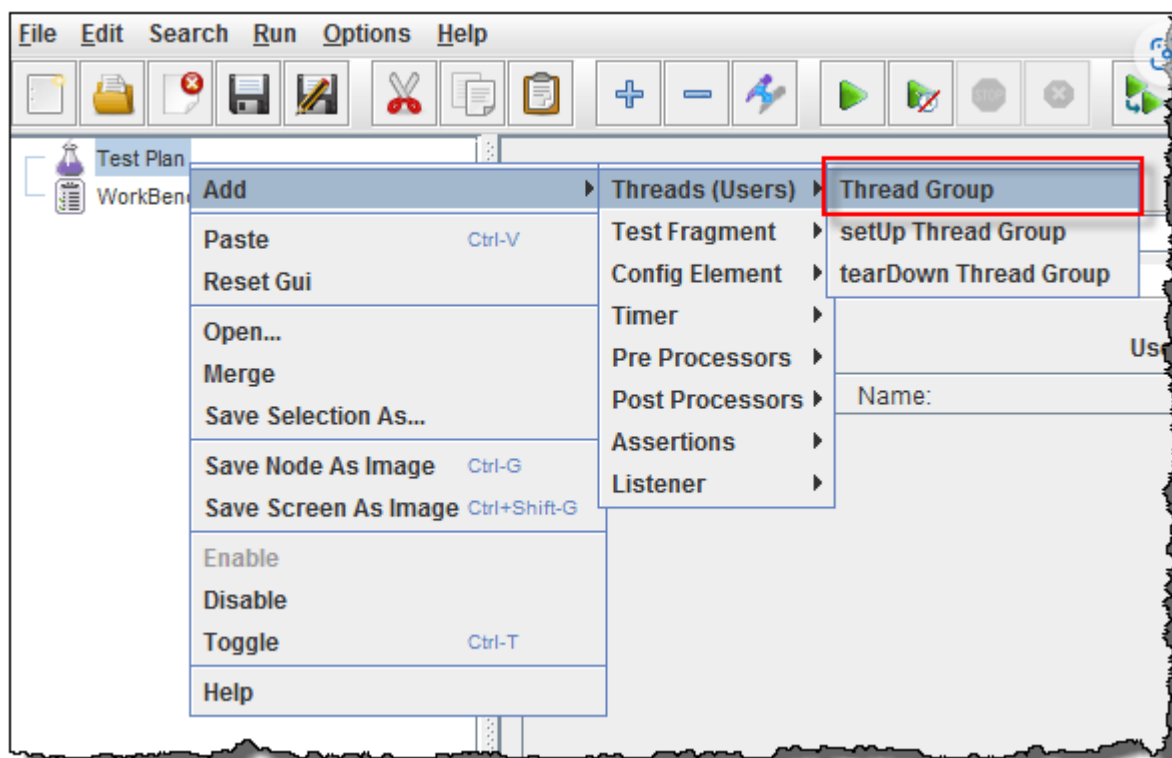
JMeter Performance Testing là phương pháp kiểm thử được thực hiện bằng cách sử dụng Apache JMeter để kiểm thử hiệu năng của một ứng dụng web JMeter For Performance Testing giúp kiểm thử cả tài nguyên tĩnh và động, giúp phát hiện người dùng đồng thời trên trang web và cung cấp nhiều đồ thị phân tích đồ họa cho kiểm thử hiệu năng. Kiểm thử hiệu năng bằng JMeter bao gồm kiểm thử tải và kiểm tra tính ổn định (stress test) của ứng dụng web.

Các bước tạo một Performance Test Plan trên JMeter



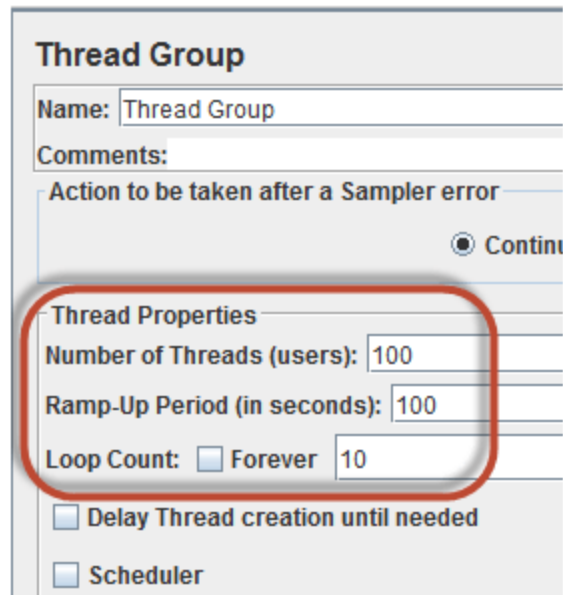
Bước 1: Tạo nhóm các luồng sự kiện

1. Khởi động JMeter
2. Chọn Test Plan trên cây thư mục
3. Tạo Thread Group



Trên bảng điều khiển, nhập Thread Properties như sau:

- Number of Threads: 100 (Số user kết nối đến website: 100)
- Loop Count: 10 (Số lần thực hiện kiểm thử)
- Ramp-Up Period: 100



Thread Group

Name: Thread Group

Comments:

Action to be taken after a Sampler error

☒ Continue

Thread Properties

Number of Threads (users): 100

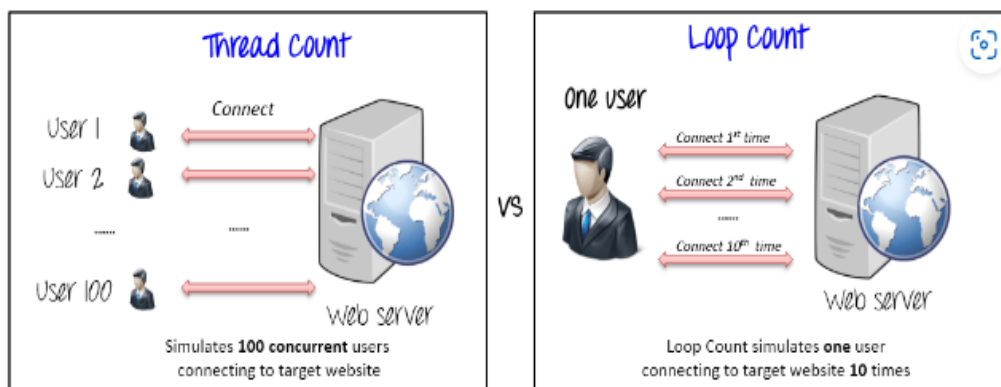
Ramp-Up Period (in seconds): 100

Loop Count: ☒ Forever 10

☐ Delay Thread creation until needed

☐ Scheduler

Sự khác nhau giữa Thread và Loop



Ramp-Up Period cho JMeter biết khoảng thời gian trì hoãn trước khi bắt đầu người dùng tiếp theo. Ví dụ: nếu chúng tôi có 100 người dùng và khoảng thời

gian Tăng tốc là 100 giây, thì độ trễ giữa những người dùng bắt đầu sẽ là 1 giây (100 giây /100 người dùng)



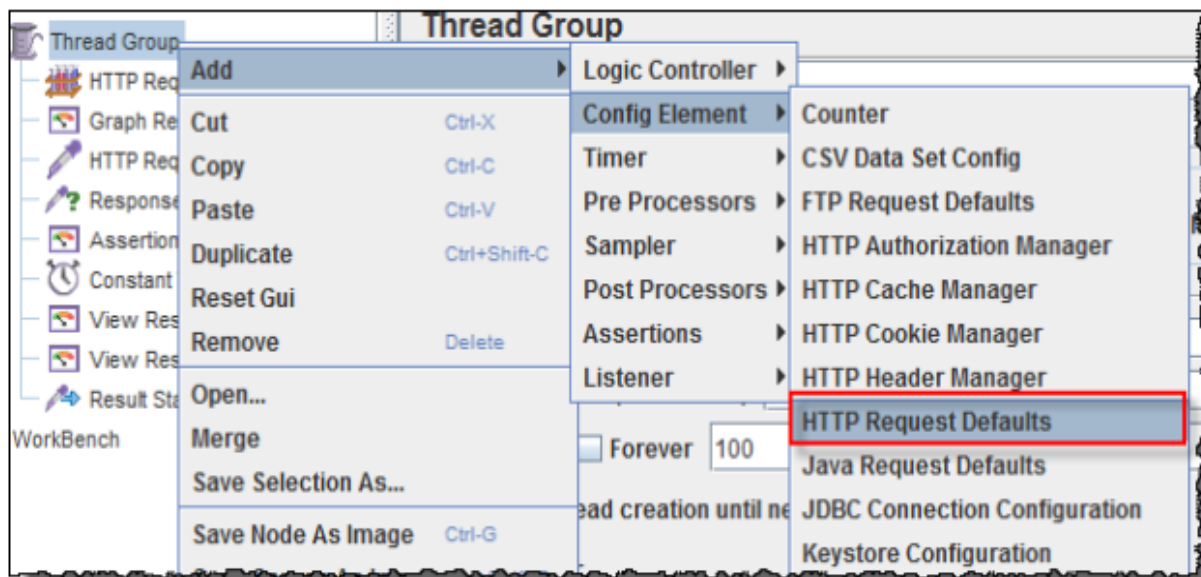
Bước 2: Thêm yếu tố

Bây giờ chúng tôi xác định những yếu tố JMeter trong thử nghiệm này. Các yếu tố là

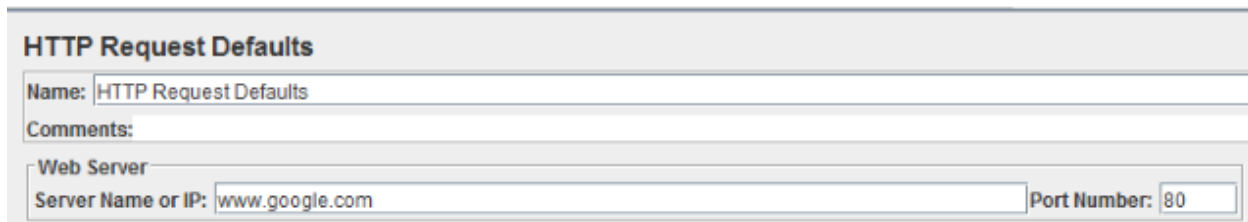
- **HTTP request Default**

Có thể thêm phần tử này bằng cách nhấp chuột phải vào Nhóm chủ đề và chọn:

Add -> Config Element -> HTTP Request Defaults.



Trong bảng điều khiển HTTP Request Defaults, nhập tên Website đang kiểm tra:



HTTP Request Defaults

Name: HTTP Request Defaults

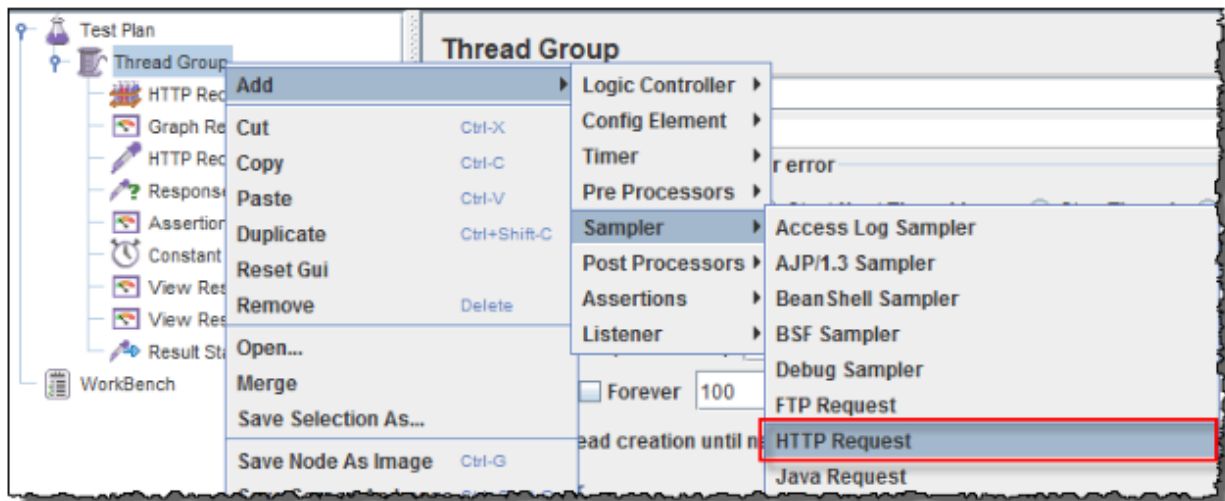
Comments:

Web Server

Server Name or IP: www.google.com Port Number: 80

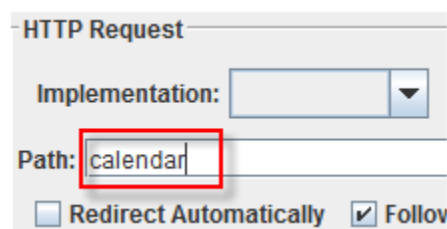
- **HTTP Request**

Nhấp chuột phải vào Nhóm chủ đề và chọn: **Add -> Sampler -> HTTP Request**.



Trong HTTP Request Control Panel, trường Path cho biết bạn muốn URL request nào tới máy chủ Google.

Ví dụ: nếu bạn nhập “[lich](http://www.google.com/calendar)” vào trường Đường dẫn. JMeter sẽ tạo URL request <http://www.google.com/calendar> đến server



HTTP Request

Implementation: [Dropdown]

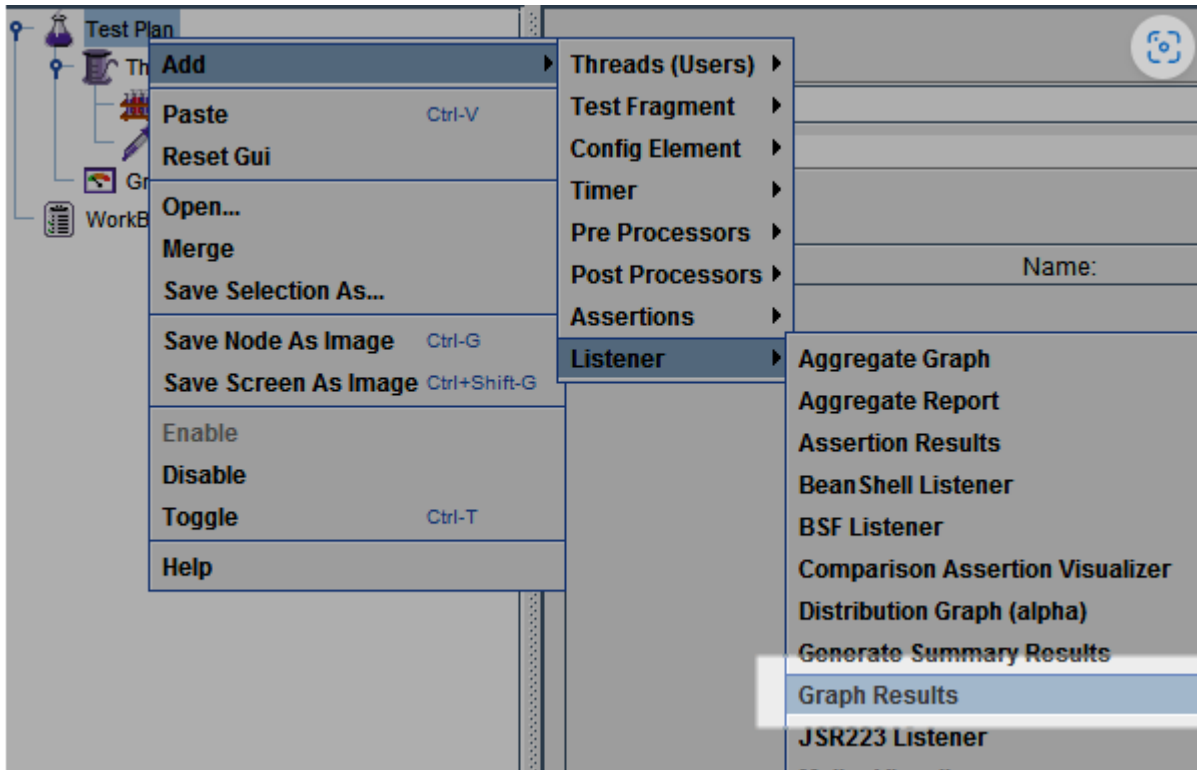
Path: calendar

☐ Redirect Automatically ☒ Follow

Nếu bạn để trống trường Đường dẫn JMeter sẽ tạo URL request <http://www.google.com>

Bước 3: Thêm biểu đồ kết quả

Chọn phải **Test Plan**, **Add -> Listener -> Graph Results**

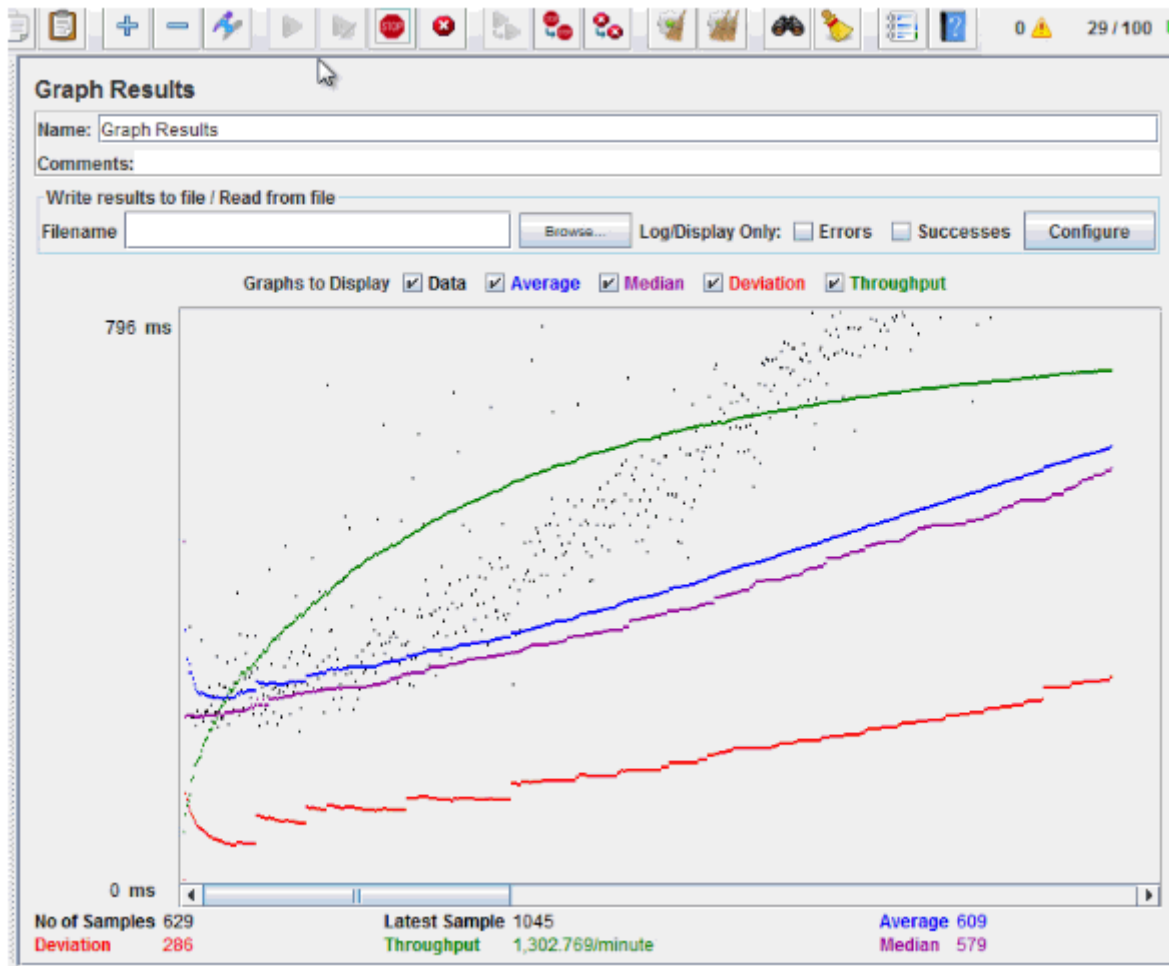


Bước 4: Chạy kiểm thử và nhận kết quả

Nhấn nút Run (Ctrl + R) trên Toolbar để bắt đầu quá trình kiểm thử phần mềm. Bạn sẽ thấy kết quả kiểm tra hiển thị trên Biểu đồ trong thời gian thực.

Hình ảnh dưới đây trình bày một biểu đồ của kế hoạch thử nghiệm, trong đó chúng tôi mô phỏng 100 người dùng đã truy cập vào trang web

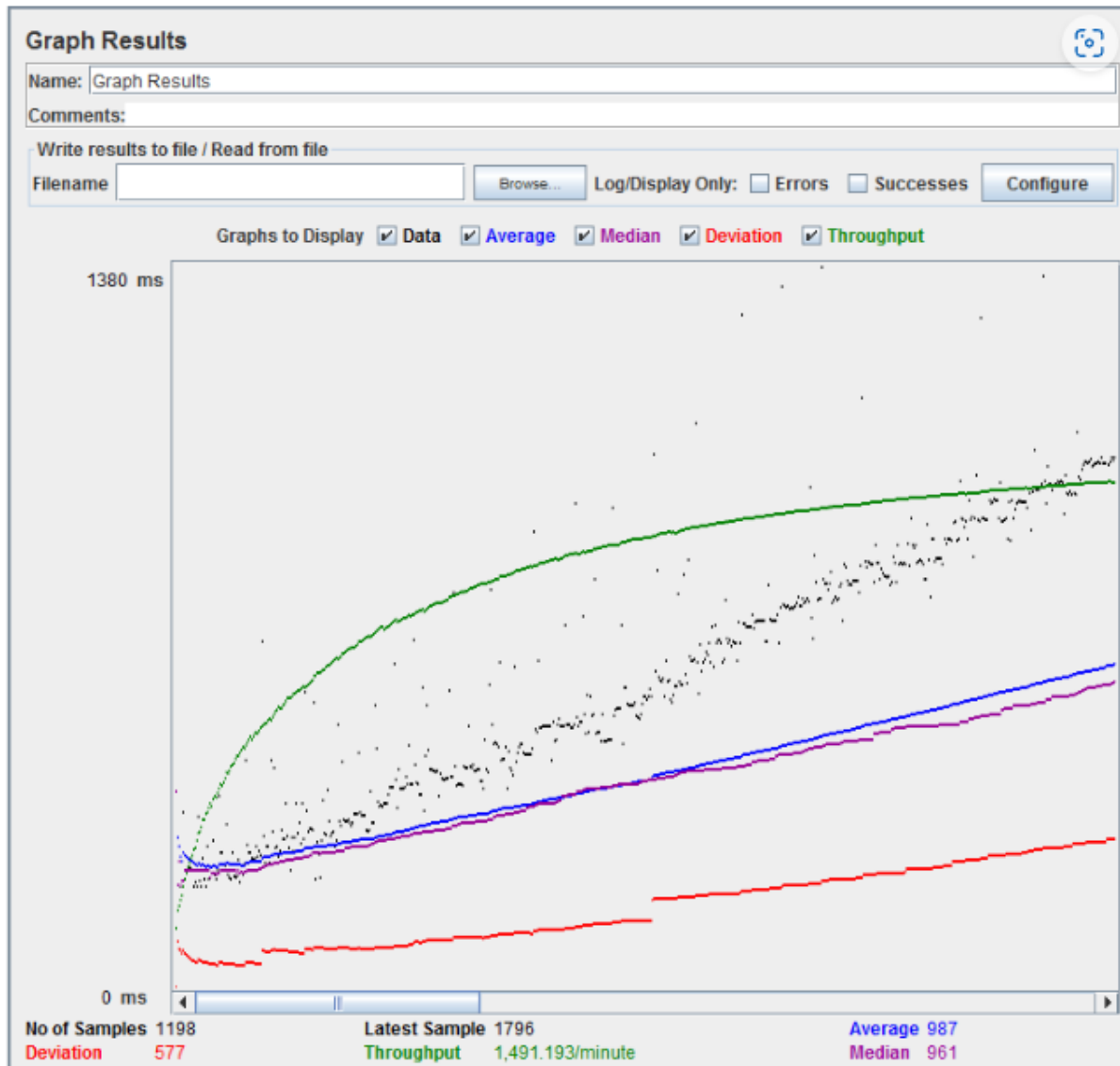
www.google.com.



Ở dưới cùng của hình ảnh, có các số liệu thống kê sau đây, được thể hiện bằng màu sắc:

- **Đen:** Tổng số mẫu hiện tại đã gửi.
- **Màu xanh dương:** Mức trung bình hiện tại của tất cả các mẫu được gửi.
- **Màu đỏ:** Độ lệch chuẩn hiện tại.
- **Màu xanh lục:** Tốc độ thông lượng biểu thị số lượng yêu cầu mỗi phút mà máy chủ xử lý

Hãy phân tích hiệu suất của máy chủ Google trong hình bên dưới.



Để phân tích hiệu suất của web đang kiểm tra, bạn nên tập trung vào 2 thông số

- **Thông lượng(Throughput)**
- **độ lệch (Deviation)**

Thông lượng là tham số quan trọng nhất. Nó thể hiện khả năng xử lý tải nặng của máy chủ. Thông lượng càng cao thì hiệu suất của máy chủ càng tốt.

Trong thử nghiệm này, thông lượng của máy chủ Google là 1.491,193/phút. Điều đó có nghĩa là máy chủ Google có thể xử lý 1.491.193 yêu cầu mỗi phút. Giá trị này khá cao nên có thể kết luận máy chủ Google hoạt động tốt

Độ lệch được hiển thị bằng màu đỏ – nó biểu thị độ lệch so với mức trung bình. Càng nhỏ càng tốt.

LƯU Ý: Các giá trị trên phụ thuộc vào một số yếu tố như tải máy chủ hiện tại tại Google, tốc độ internet, sức mạnh CPU của bạn, v.v. Do đó, rất ít khả năng bạn sẽ nhận được kết quả tương tự như trên. Vì vậy, đừng hoảng sợ!

Xử lý sự cố:

Nếu bạn gặp sự cố trong khi chạy kịch bản trên... hãy làm như sau

1. Kiểm tra xem bạn có đang kết nối với internet qua proxy không.
Nếu có, hãy xóa proxy.
2. Mở một phiên bản mới của Jmeter.
3. Mở PerformanceTestPlan.jmx trong Jmeter.
4. Nhấp đúp chuột vào Nhóm chủ đề → Kết quả đồ thị.
5. Chạy thử nghiệm.

CHƯƠNG VI

Jmeter Timers: Constant, Gaussian Random, Uniform

Bộ hẹn giờ là gì?

Theo mặc định, JMeter gửi yêu cầu mà không tạm dừng giữa mỗi yêu cầu. Trong trường hợp đó, JMeter có thể làm máy chủ thử nghiệm của bạn quá tải bằng cách thực hiện quá nhiều yêu cầu trong một khoảng thời gian ngắn.

Hãy tưởng tượng rằng bạn gửi hàng nghìn yêu cầu đến một máy chủ web đang được kiểm tra trong vòng vài giây. Đây là những gì sẽ xảy ra!

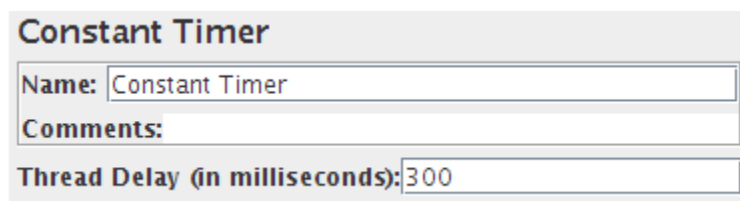
Bộ hẹn giờ cho phép JMeter trì hoãn giữa mỗi yêu cầu mà một luồng thực hiện.

Bộ đếm thời gian có thể giải quyết vấn đề quá tải của máy chủ.

Ngoài ra, trong thực tế, khách truy cập không đến một trang web cùng một lúc mà vào các khoảng thời gian khác nhau. Vì vậy, Timer sẽ giúp bắt chước hành vi thời gian thực.

Sau đây là một số loại timer phổ biến trong JMeter

Constant Timer: trì hoãn mỗi yêu cầu của người dùng trong cùng một khoảng thời gian.



The image shows a screenshot of the 'Constant Timer' configuration window in JMeter. It has a title bar 'Constant Timer'. Below the title bar, there are three input fields: 'Name:' with the value 'Constant Timer', 'Comments:', and 'Thread Delay (in milliseconds):' with the value '300'.

Gaussian Random Timer: trì hoãn mỗi yêu cầu của người dùng trong một khoảng thời gian ngẫu nhiên.

Gaussian Random Timer

Name: Gaussian Random Timer

Comments:

Thread Delay Properties

Deviation (in milliseconds): 100.0

Constant Delay Offset (in milliseconds): 300

Specified particular value for Timer

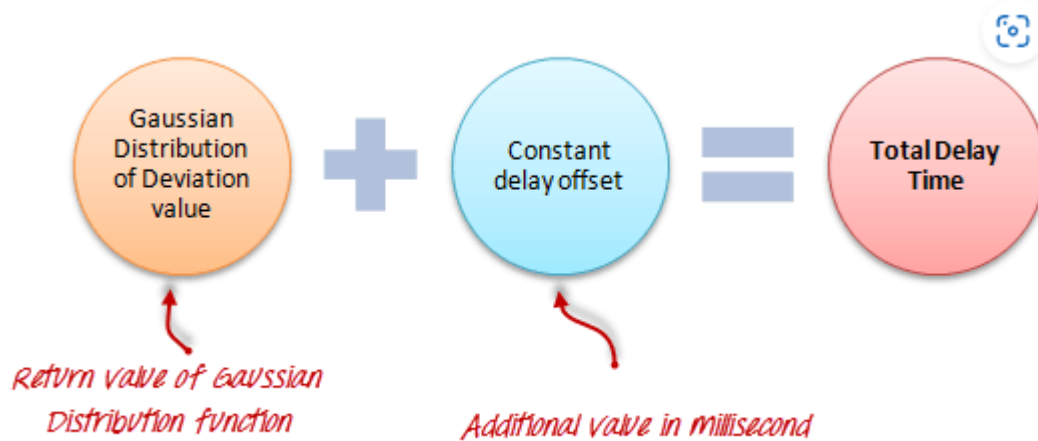
Các tham số:

Name: Tên mô tả cho bộ đếm thời gian này được hiển thị trong cây

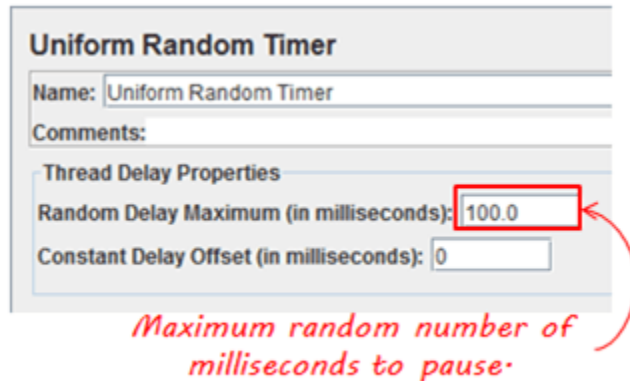
Deviations (mili giây): Một tham số của hàm phân phối Gaussian

Constant Delay Offset (mili giây): Giá trị bổ sung tính bằng mili giây

Vì vậy, tổng độ trễ được mô tả như hình dưới đây:



Uniform Random Timer: trì hoãn mỗi yêu cầu của người dùng trong một khoảng thời gian ngẫu nhiên.



Name: Tên mô tả cho bộ đếm thời gian này được hiển thị trong cây

Random Delay Maximum: Số mili giây ngẫu nhiên tối đa để trì hoãn.

Constant Delay Offset (milliseconds): Giá trị bổ sung tính bằng mili giây
Tổng độ trễ là tổng của giá trị ngẫu nhiên và giá trị bù.

BeanShell Timer: có thể được sử dụng để tạo thời gian trễ giữa mỗi yêu cầu của người dùng.

BSF Timer: có thể được sử dụng để tạo độ trễ giữa mỗi yêu cầu của người dùng bằng ngôn ngữ BSF.

JSR223 Timer: có thể được sử dụng để tạo độ trễ giữa mỗi yêu cầu của người dùng bằng ngôn ngữ JSR223

Cách sử dụng Constant Timer

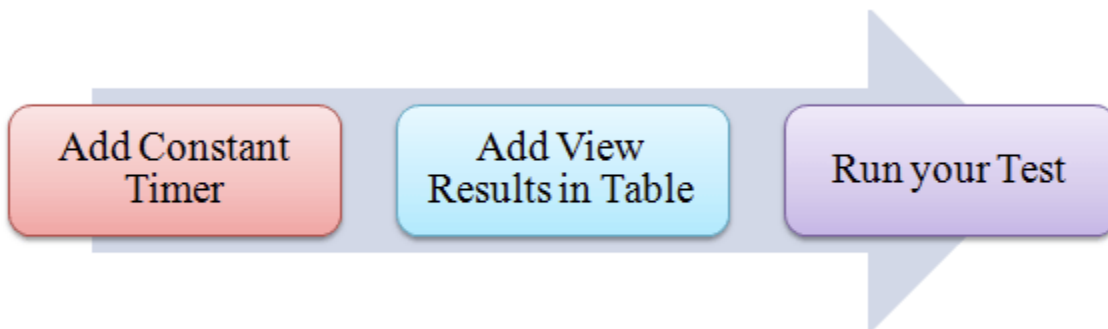
Trong ví dụ này, bạn sẽ sử dụng Constant Timer để đặt độ trễ cố định giữa các yêu cầu của người dùng tới google.com

Hãy bắt đầu với một kịch bản thử nghiệm đơn giản

JMeter tạo một yêu cầu người dùng tới <http://www.google.com> 100 lần

Độ trễ giữa mỗi yêu cầu của người dùng là 5000 ms

Đây là roadmap cho ví dụ thực tế này:



Bước 1: Tạo nhóm các luồng sự kiện

Nhấp chuột phải vào Test Plan và thêm một nhóm luồng mới: Add-> Threads (Users) -> Thread Group

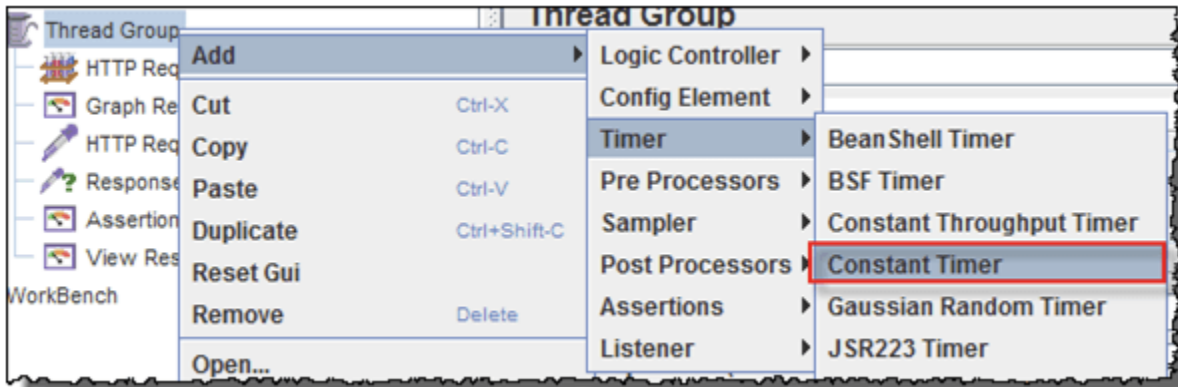
Trong bảng điều khiển Nhóm Chủ đề, nhập Thread Properties như sau

Bước 2: Thêm yếu tố

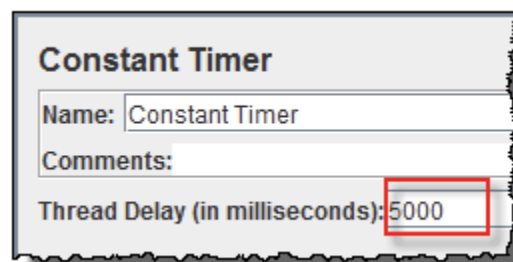
- Thêm HTTP request default
- Thêm HTTP request

Bước 3: Thêm Constant Timer

Chọn phải **Thread Group -> Timer -> Constant Timer**

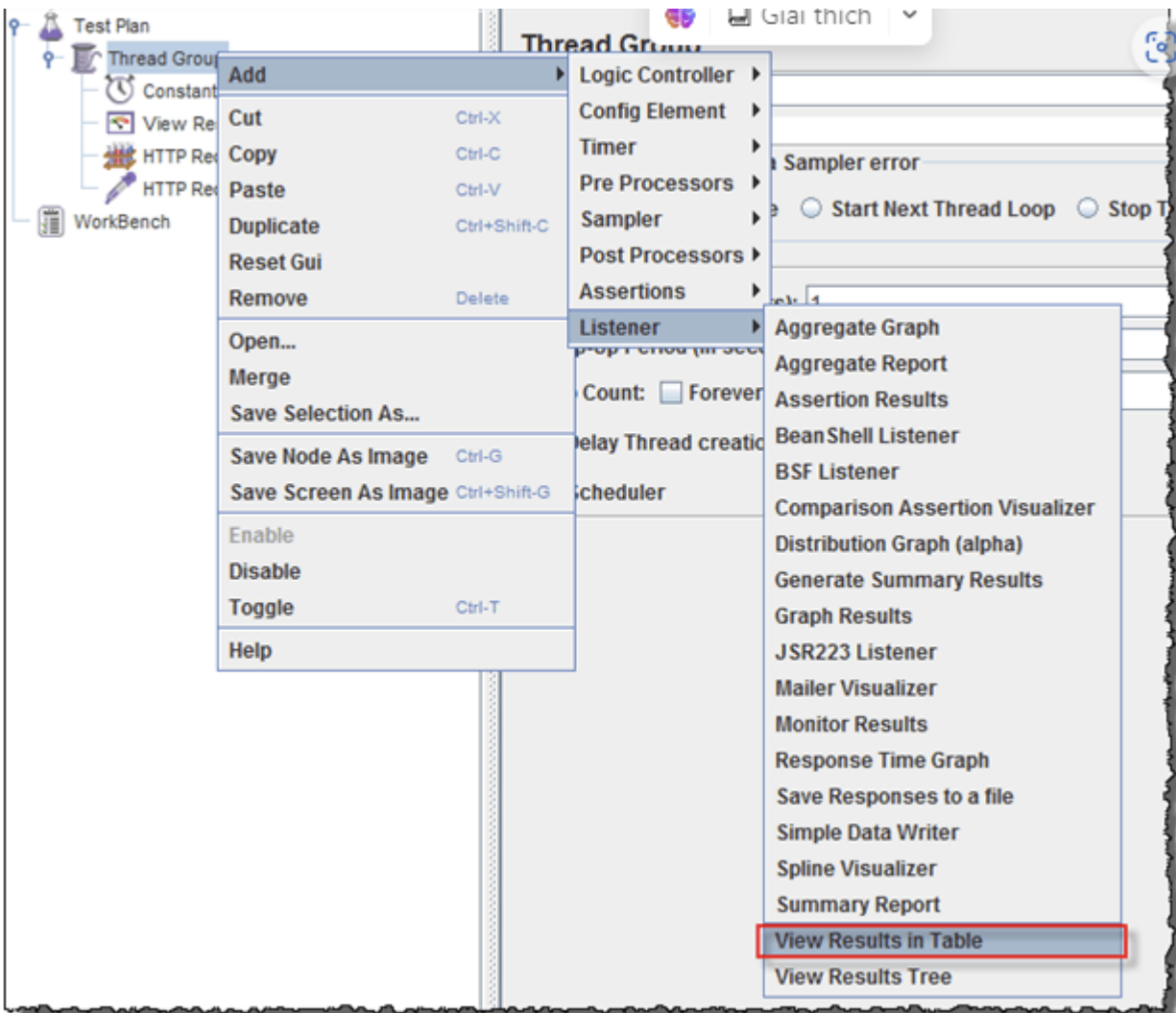


Cấu hình Thread Delay 5000 mili giây



Bước 4: Thêm bảng xem kết quả

Chọn phải Add -> Listener -> View Result in Table



View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Latency
Sample No.	Start time	Name of Thread	type of user request	Time to finish a request	status of request (success, fail)	Size of request	Measure of time delay

Bước 5: Chạy kiểm thử

Khi bạn đã sẵn sàng để chạy thử nghiệm, hãy nhấp vào nút Run trên thanh menu hoặc phím tắt Ctrl + R

Đây là kết quả của bài kiểm tra này

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: *time delay between request is around 5000ms* Browse... Log/Display Only: ☐ Errors ☐ Success

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes
1	22:04:56.509	Thread Group 1-1	HTTP Request	356	✓	1
2	22:05:01.866	Thread Group 1-1	HTTP Request	172	✓	1
3	22:05:07.039	Thread Group 1-1	HTTP Request	173	✓	1
4	22:05:12.213	Thread Group 1-1	HTTP Request	172	✓	1
5	22:05:17.386	Thread Group 1-1	HTTP Request	170	✓	1
6	22:05:22.558	Thread Group 1-1	HTTP Request	167	✓	1
7	22:05:27.727	Thread Group 1-1	HTTP Request	168	✓	1
8	22:05:32.896	Thread Group 1-1	HTTP Request	167	✓	1
9	22:05:38.064	Thread Group 1-1	HTTP Request	172	✓	1
10	22:05:43.237	Thread Group 1-1	HTTP Request	170	✓	1
11	22:05:48.408	Thread Group 1-1	HTTP Request	184	✓	1
12	22:05:53.593	Thread Group 1-1	HTTP Request	287	✓	1
13	22:05:58.880	Thread Group 1-1	HTTP Request	171	✓	1
14	22:06:04.053	Thread Group 1-1	HTTP Request	168	✓	1
15	22:06:09.222	Thread Group 1-1	HTTP Request	170	✓	1
16	22:06:14.393	Thread Group 1-1	HTTP Request	766	✓	1
17	22:06:20.161	Thread Group 1-1	HTTP Request	176	✓	1
18	22:06:25.338	Thread Group 1-1	HTTP Request	168	✓	1
19	22:06:30.507	Thread Group 1-1	HTTP Request	171	✓	1
20	22:06:35.680	Thread Group 1-1	HTTP Request	169	✓	1
21	22:06:40.851	Thread Group 1-1	HTTP Request	173	✓	1
22	22:06:46.025	Thread Group 1-1	HTTP Request	183	✓	1
23	22:06:51.209	Thread Group 1-1	HTTP Request	171	✓	1
24	22:06:56.382	Thread Group 1-1	HTTP Request	169	✓	1
25	22:07:01.552	Thread Group 1-1	HTTP Request	178	✓	1
26	22:07:06.730	Thread Group 1-1	HTTP Request	169	✓	1
27	22:07:11.901	Thread Group 1-1	HTTP Request	199	✓	1

- Start time: 22:05:01.866
- Sample Time của Sample 2: 172 ms
- Constant Timer: 5000 ms
- End Time of this sample: $22:05:01.866 + 172 + 5000 = 22:05:07.038$

CHƯƠNG VII

Cách dùng Assertions trong Jmeter

Assertion là gì?

Assertion giúp xác minh rằng máy chủ của bạn đang được thử nghiệm trả về kết quả như mong đợi.

Các loại Assertion

- **Response Assertion**

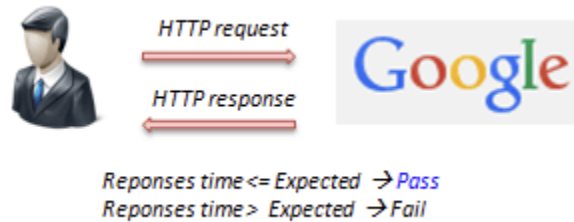
Response Assertion cho phép bạn thêm các chuỗi mẫu để so sánh với các trường khác nhau của phản hồi máy chủ.

Ví dụ: bạn gửi yêu cầu của người dùng đến trang web <http://www.google.com> và nhận được phản hồi của máy chủ. Bạn có thể sử dụng Xác nhận phản hồi để xác minh xem phản hồi của máy chủ có chứa chuỗi mẫu dự kiến hay không (ví dụ: "OK").

- **Duration Assertion**

Duration Assertion kiểm tra xem từng phản hồi của máy chủ đã được nhận trong một khoảng thời gian nhất định hay chưa. Bất kỳ phản hồi nào mất nhiều thời gian hơn số mili giây đã cho (do người dùng chỉ định) được đánh dấu là phản hồi không thành công.

Ví dụ: một yêu cầu của người dùng được gửi tới www.google.com bởi JMeter và nhận được phản hồi trong thời gian dự kiến 5 ms sau đó Trường hợp kiểm tra vượt qua, nếu không, trường hợp kiểm tra không thành công.



- **Size Assertion**

Size Assertion kiểm tra xem mỗi phản hồi của máy chủ có chứa số byte dự kiến trong đó hay không. Bạn có thể chỉ định rằng kích thước bằng, lớn hơn, nhỏ hơn hoặc không bằng một số byte nhất định.

JMeter gửi yêu cầu của người dùng tới www.google.com và nhận được gói phản hồi có kích thước nhỏ hơn byte dự kiến 5000 byte cho một trường hợp thử nghiệm vượt qua. Nếu khác, trường hợp thử nghiệm thất bại.

- **XML Assertion**

XML Assertion kiểm tra xem dữ liệu phản hồi có chứa một tài liệu XML chính xác hay không.

XML Assertion
Name: XML Assertion
Comments:

- **HTML Assertion**

HTML Assertion cho phép người dùng kiểm tra cú pháp HTML của dữ liệu phản hồi. Nó có nghĩa là dữ liệu phản hồi phải đáp ứng cú pháp HTML.

HTML Assertion

Name: HTML Assertion

Comments:

Tidy Settings

Doctype: omit

Format

☒ HTML

☐ XHTML

☐ XML

☐ Errors only

Error threshold: 0 Warning threshold: 0

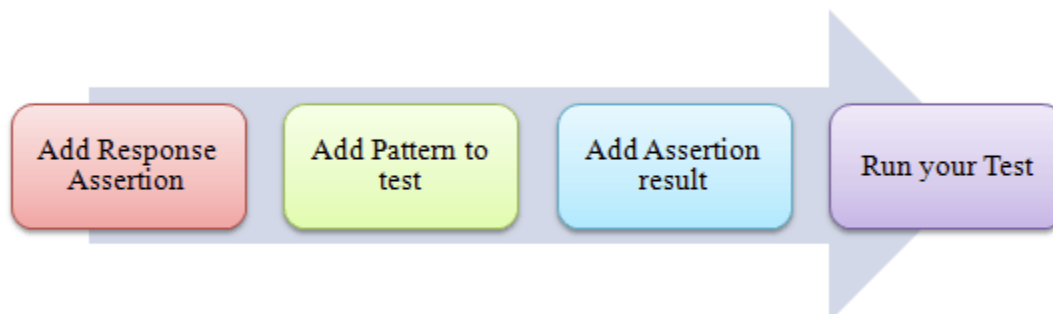
Write Jtidy report to file

Filename

Các bước sử dụng Response Assertion

Chúng ta sẽ tiếp tục tập lệnh mà chúng tôi đã phát triển trong hướng dẫn trước đó. Trong thử nghiệm này, chúng ta đang sử dụng Response Assertion để so sánh gói phản hồi từ www.google.com khớp với chuỗi dự kiến của bạn.

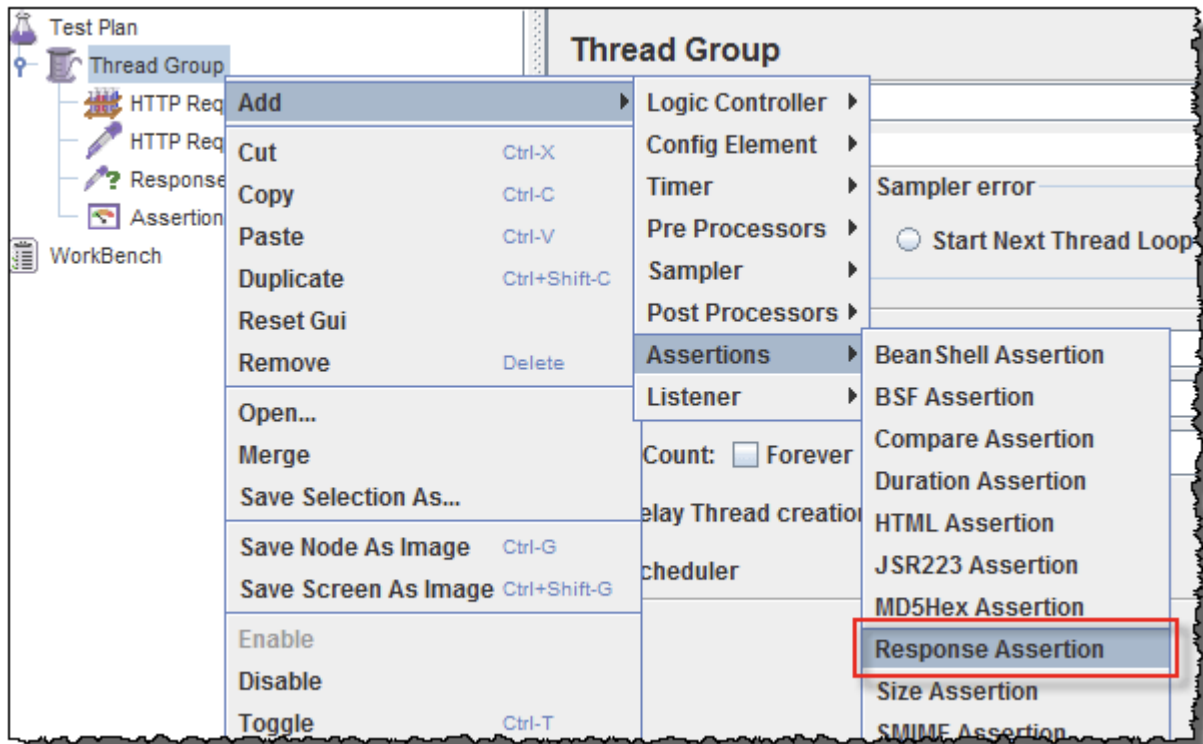
Đây là roadmap cho bài kiểm tra này:



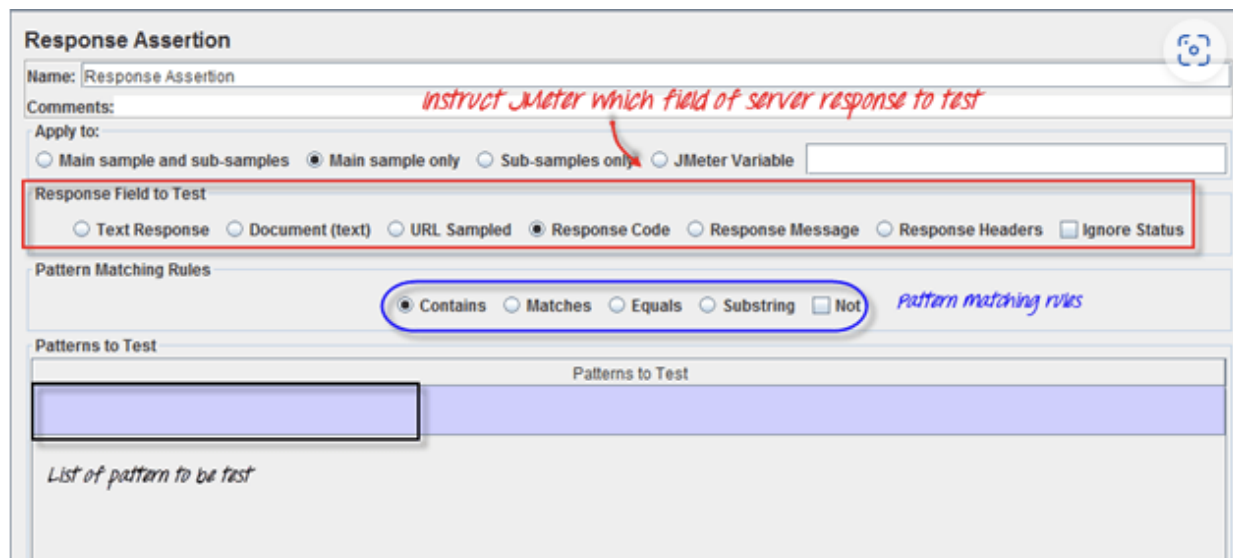
Bảng điều khiển response assertion cho phép bạn thêm các chuỗi mẫu để so sánh với các trường khác nhau của phản hồi.

Bước 1: Thêm Response Assertion

Click phải Thread Group -> Add -> Assertions -> Response Assertion



Màn hình hiển thị như bên dưới



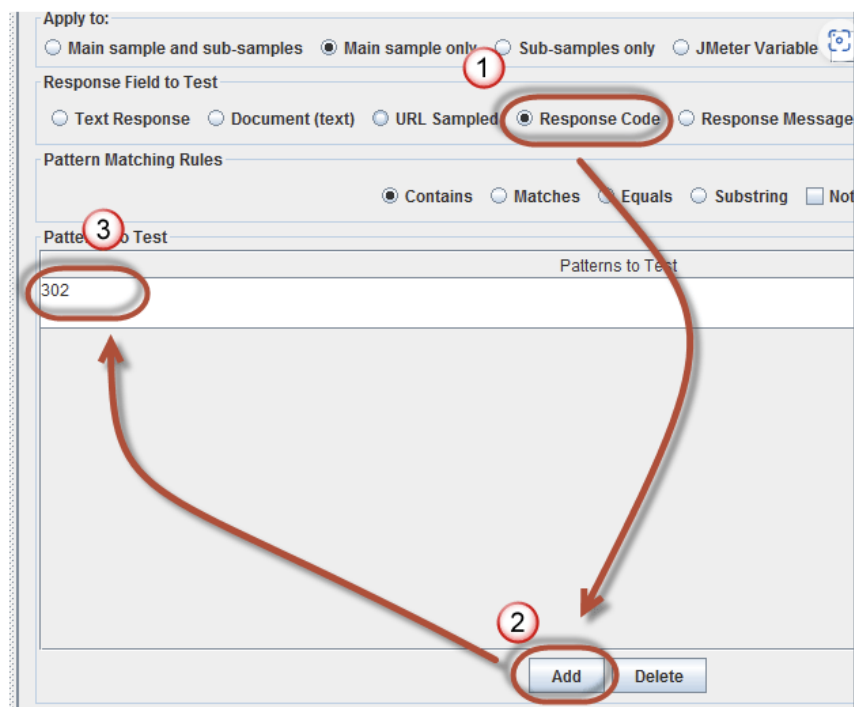
Bước 2: Thêm Pattern

Khi bạn gửi yêu cầu tới máy chủ Google, nó có thể trả về một số mã phản hồi như sau:

- 404: Lỗi máy chủ
- 200: Máy chủ Ok
- 302: Máy chủ web chuyển hướng đến các trang khác. Điều này thường xảy ra khi bạn truy cập google.com từ bên ngoài Hoa Kỳ. Google chuyển hướng đến trang web cụ thể theo quốc gia. Như được hiển thị bên dưới, google.com chuyển hướng đến google.co.in cho Người dùng Ấn Độ.

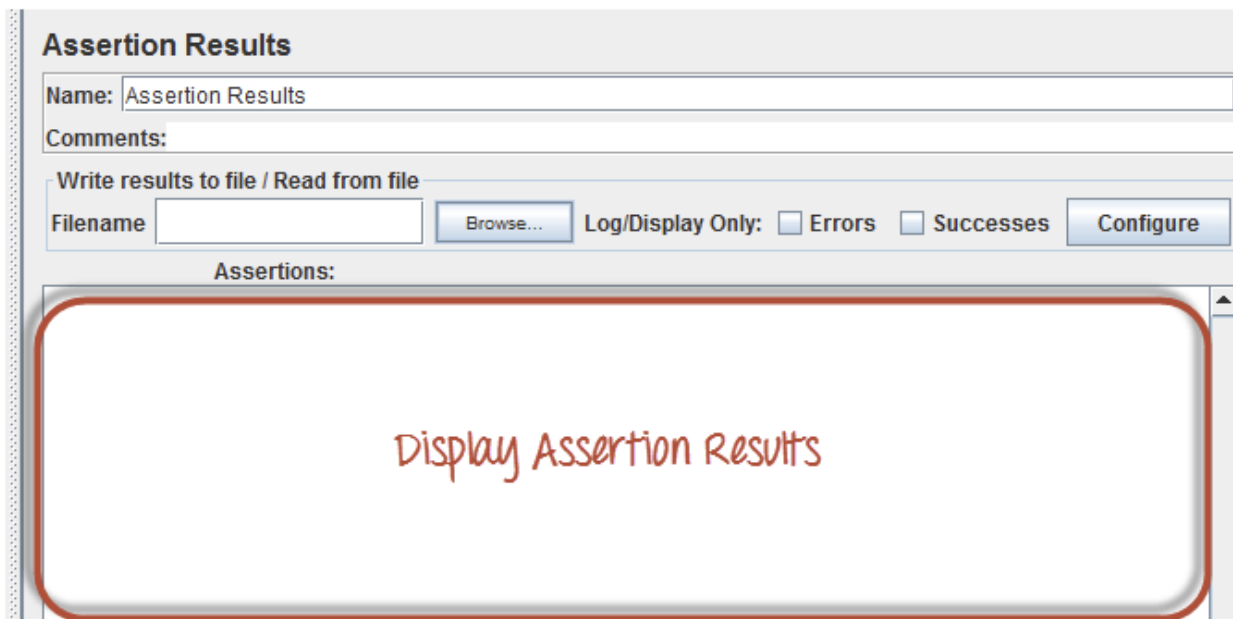
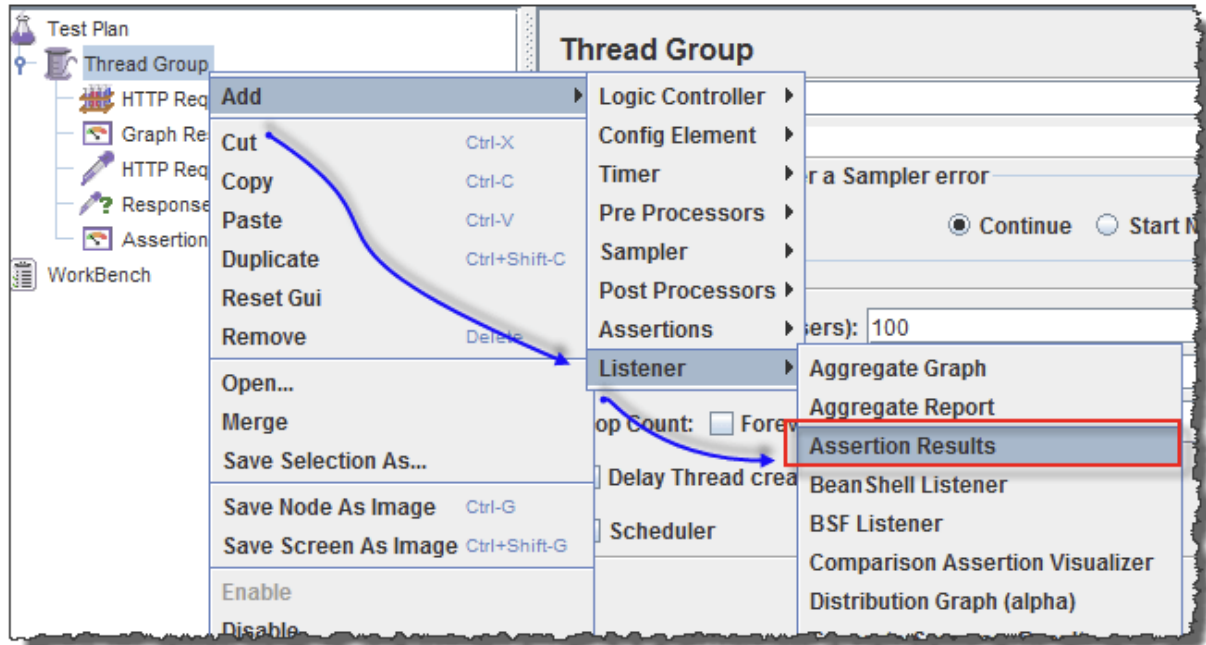
Giả sử rằng bạn muốn xác minh rằng mã phản hồi của máy chủ web google.com có chứa mẫu 302:

1. Trên Trường phản hồi để kiểm tra, chọn Mã phản hồi,
2. Trên Bảng xác nhận phản hồi, hãy nhấp vào Thêm -> một mục nhập trống mới hiển thị -> nhập 302 trong Mẫu để kiểm tra.



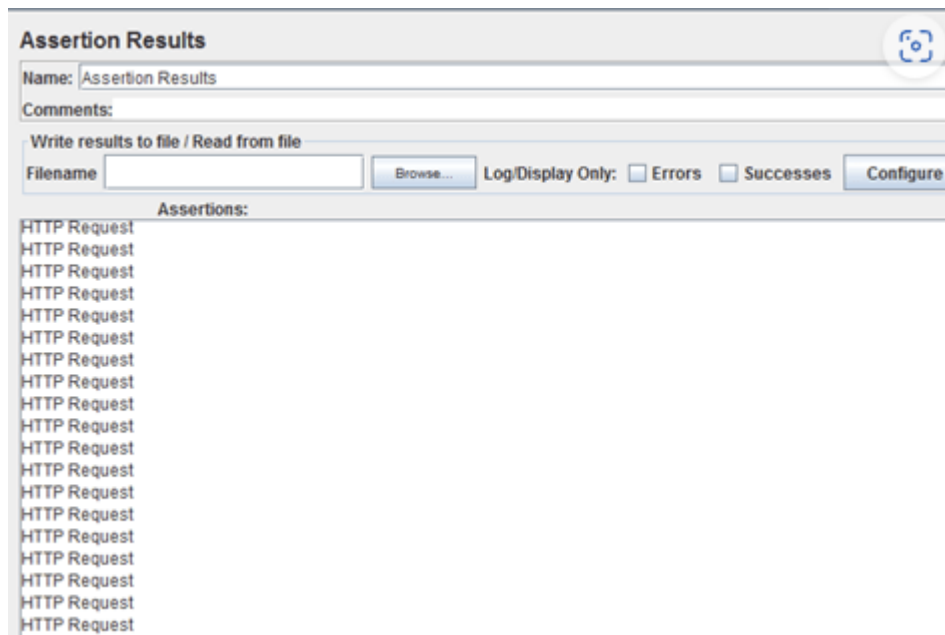
Bước 3: Thêm kết quả

Click phải Thread Group, Add -> Listener -> Assertion Results

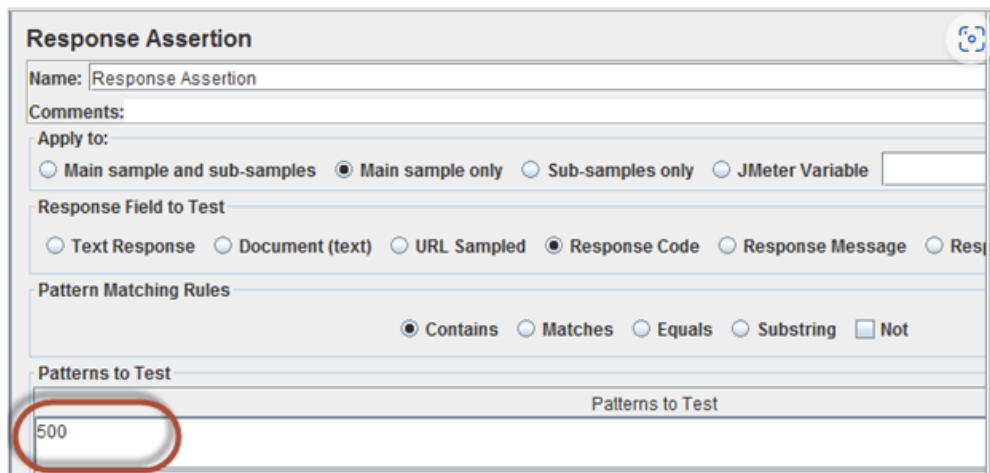


Bước 4: Chạy kiểm thử

1. Click vào Thread Group -> Assertion Result
2. Khi bạn đã sẵn sàng chạy thử nghiệm, hãy nhấp vào nút Run trên thanh menu hoặc phím tắt Ctrl+R.
3. Kết quả kiểm tra sẽ hiển thị trên khung Kết quả khẳng định. Nếu mã phản hồi của máy chủ Google chứa mẫu 302, trường hợp thử nghiệm sẽ được thông qua. Bạn sẽ thấy thông báo hiển thị như sau:



Bây giờ quay lại Response Assertion Panel, bạn thay đổi Pattern to test từ 302 thành 500.



Vì mã phản hồi của máy chủ Google không chứa mẫu này nên bạn sẽ thấy trường hợp kiểm tra Không thành công như sau:



HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/
HTTP Request	Response Assertion : Test failed: code expected to contain /500/

CHƯƠNG VIII

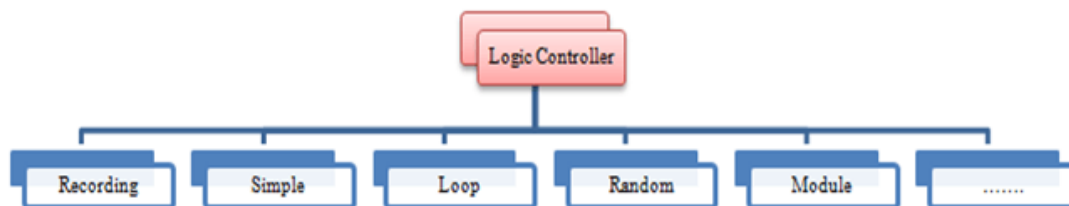
Controllers in JMeter:

Loop, Simple, Transaction, Module, Random

Logic Controller là gì ?

Bộ điều khiển logic cho phép bạn xác định thứ tự xử lý yêu cầu trong một Chủ đề. Nó cho phép bạn kiểm soát “thời điểm” gửi yêu cầu của người dùng đến máy chủ web. Ví dụ: bạn có thể sử dụng Bộ điều khiển ngẫu nhiên để gửi yêu cầu HTTP đến máy chủ một cách ngẫu nhiên

Bộ điều khiển logic xác định thứ tự thực hiện yêu cầu của người dùng. Dưới đây là một số bộ điều khiển Logic thường được sử dụng:



Recording Controller:

JMeter có thể ghi lại các bước Kiểm tra của bạn; bộ điều khiển ghi là trình giữ chỗ để lưu các bước ghi này.



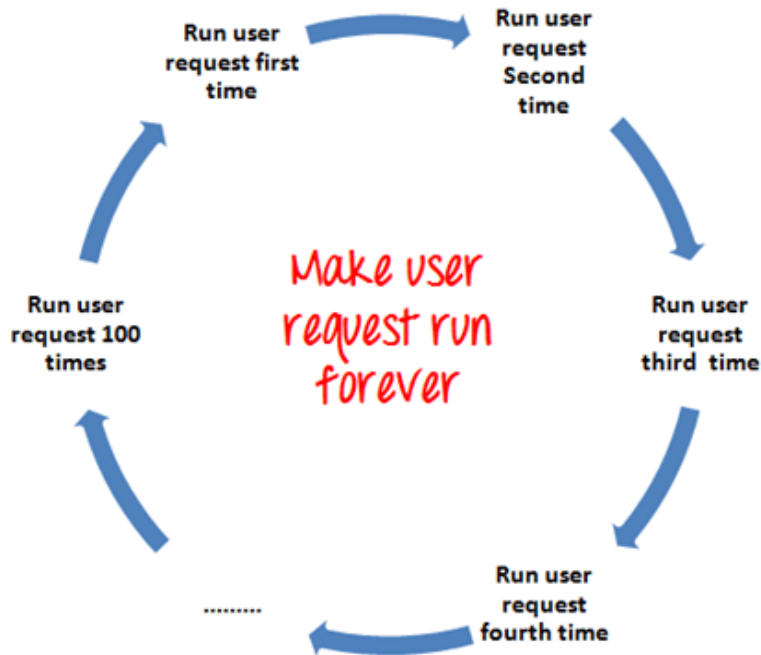
Simple Controller:

Bộ điều khiển đơn giản chỉ là nơi chứa yêu cầu của người dùng.



Loop Controller:

Bộ điều khiển vòng lặp làm cho yêu cầu của người dùng chạy một số lần được chỉ định hoặc chạy mãi mãi như trong hình:



Random Controller:

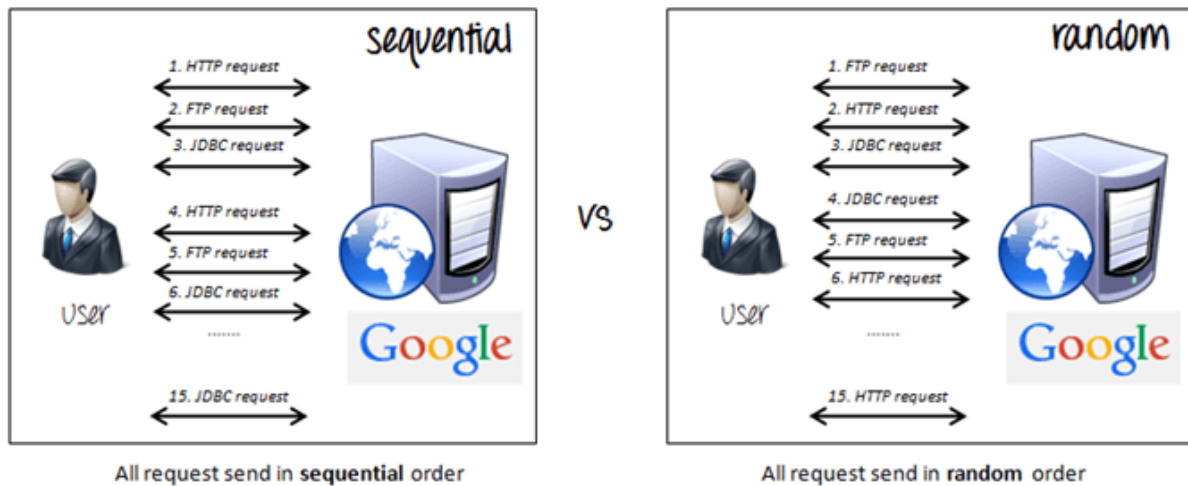
Bộ điều khiển ngẫu nhiên làm cho tất cả các yêu cầu của người dùng chạy theo thứ tự ngẫu nhiên trong mỗi chu kỳ vòng lặp.

Ví dụ: bạn có 3 yêu cầu người dùng tới trang web <http://www.google.com> theo thứ tự sau:

- Yêu cầu HTTP.
- Yêu cầu FTP.
- Yêu cầu JDBC.

3 yêu cầu này sẽ chạy 5 lần. Tổng số 15 yêu cầu của người dùng sẽ được JMeter gửi đến máy chủ Google.

Theo thứ tự tuần tự, các yêu cầu được gửi tuần tự theo thứ tự sau cho mỗi vòng lặp: Yêu cầu HTTP -> Yêu cầu FTP-> Yêu cầu JDBC



Theo thứ tự ngẫu nhiên, các yêu cầu được gửi ngẫu nhiên đối với mỗi vòng lặp.

Yêu cầu FTP -> Yêu cầu HTTP-> Yêu cầu JDBC

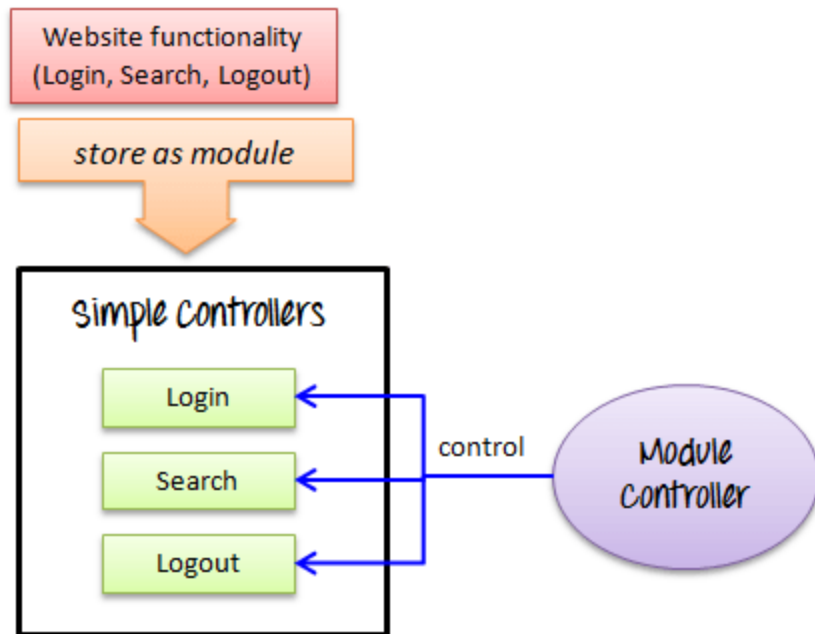
Hoặc

Yêu cầu JDBC -> Yêu cầu FTP-> Yêu cầu HTTP

Module Controller:

Mục tiêu của Bộ điều khiển mô-đun là thêm tính mô-đun vào JMeter.

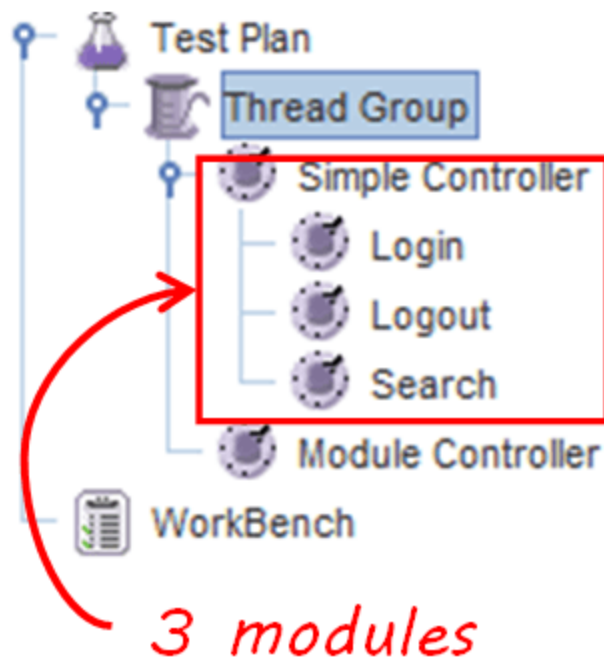
Ý tưởng chung là các ứng dụng web bao gồm các đơn vị chức năng nhỏ (tức là Đăng nhập, Tạo tài khoản, Đăng xuất...). Chức năng này có thể được lưu trữ trong Bộ điều khiển đơn giản dưới dạng “mô-đun”. Bộ điều khiển mô-đun sẽ chọn mô-đun nào cần chạy.



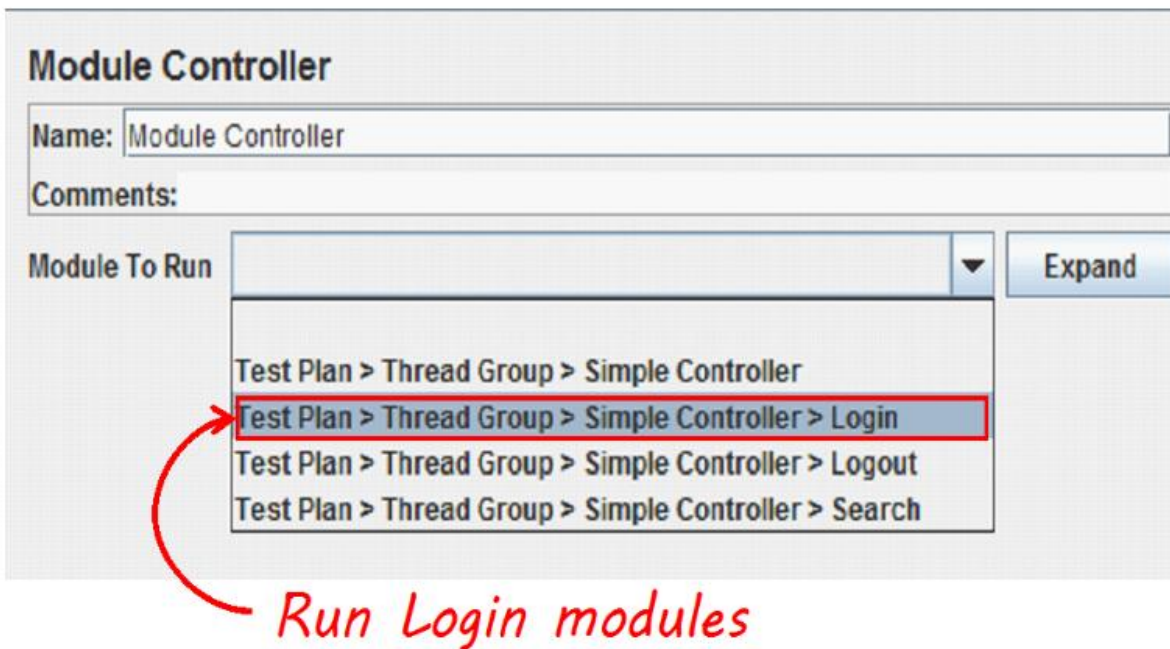
Hãy xem xét kịch bản sau đây - Bạn muốn mô phỏng:

- 50 người dùng đăng xuất,
- 100 người dùng đăng nhập
- 30 người dùng tìm kiếm www.google.com

Bạn có thể sử dụng JMeter để tạo 3 mô-đun. Mỗi mô-đun mô phỏng từng hoạt động của người dùng: Đăng nhập, Đăng xuất và Tìm kiếm.

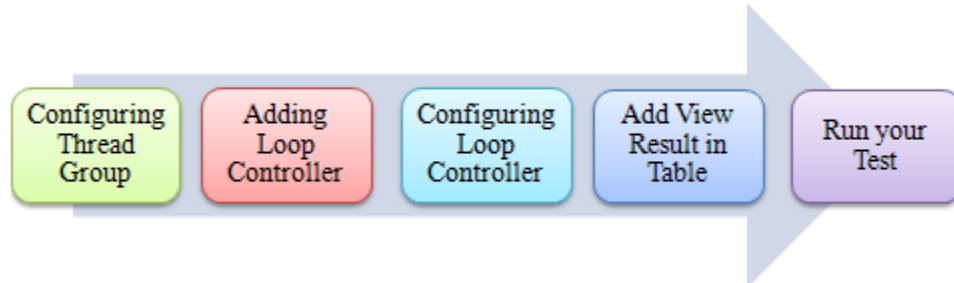


Bộ điều khiển module chọn module nào cần chạy.



Ví dụ về bộ điều khiển vòng lặp

Phần này hiển thị cho bạn hướng dẫn từng bước để thêm Bộ điều khiển vòng lặp vào kế hoạch kiểm tra hiệu suất hiện tại của bạn.



Bước 1: Cấu hình nhóm luồng sự kiện

- **Tạo nhóm các luồng sự kiện**

Nhấp chuột phải vào Test Plan và thêm một nhóm luồng mới: Add-> Threads (Users) -> Thread Group

Trong bảng điều khiển Nhóm Chủ đề, nhập Thread Properties như sau

Thread Properties

Number of Threads (users): 1

Ramp-Up Period (in seconds): 1

Loop Count: ☐ Forever 2

☐ Delay Thread creation until needed

☐ Scheduler

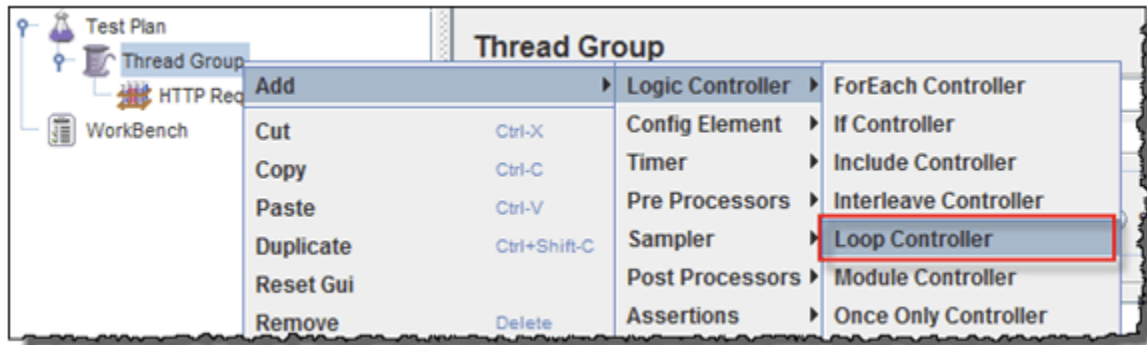
Nó sẽ yêu cầu một người dùng đến máy chủ web google.com và chạy nó 2 lần.

- **Thêm phần tử JMeter**

Thêm yêu cầu HTTP mặc định vào www.google.com.

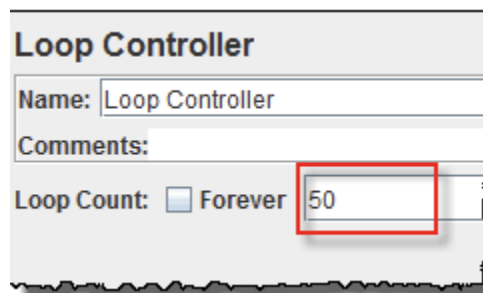
- **Thêm bộ điều khiển vòng lặp**

Nhấp chuột phải vào Nhóm chủ đề -> Bộ điều khiển logic -> Bộ điều khiển vòng lặp

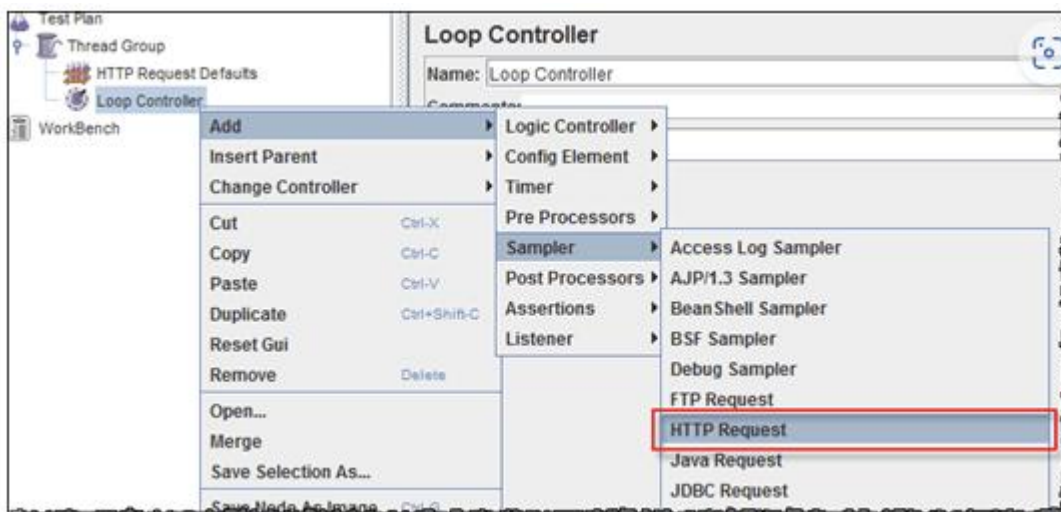


Bước 2: Cấu hình Loop Controller

Thêm giá trị 50 vào trường Loop Count như hình bên dưới. Nó sẽ khiến một yêu cầu của người dùng tới máy chủ web google.com chạy nó 50 lần, ngoài giá trị vòng lặp =2, bạn đã chỉ định cho Nhóm chủ đề ở trên. Vì vậy, JMeter sẽ gửi tổng cộng $2 * 50 = 100$ Yêu cầu HTTP.

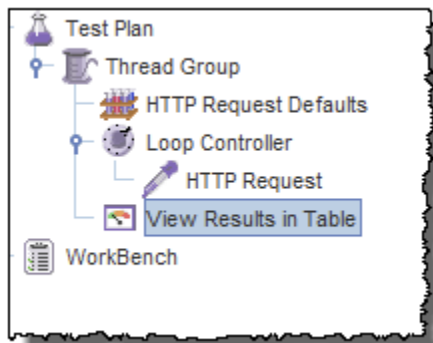


Nhấp chuột phải vào Loop Controller, Add -> Sampler -> HTTP request



Bước 3: Thêm bảng kết quả

Chúng ta sử dụng lại Bước 2 trong Bộ đếm thời gian để thêm Xem kết quả trong Bảng. Vì vậy, kế hoạch thử nghiệm được hiển thị trong hình bên dưới



Bước 4: chạy kiểm thử

Bây giờ trở lại Xem kết quả trong Bảng, nhấp vào nút Bắt đầu trên thanh Menu (Ctrl + R) để chạy thử nghiệm.

Như thể hiện trong hình bên dưới, JMeter mô phỏng một yêu cầu của người dùng, yêu cầu này được gửi 100 lần tới máy chủ web <http://www.google.com/>. Thử nghiệm bị dừng sau khi yêu cầu của người dùng được gửi 100 lần.

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse...

Log/Display Only: ☐ Errors ☐ Successes

*one user request
Thread Group 1-1 run
in 100 times*

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Latency
1	07.24.20.369	Thread Group 1-1	HTTP Request	354		13541	138
2	07.24.20.724	Thread Group 1-1	HTTP Request	182		13573	62
3	07.24.20.907	Thread Group 1-1	HTTP Request	186		13589	65
4	07.24.21.094	Thread Group 1-1	HTTP Request	188		13553	68
5	07.24.21.283	Thread Group 1-1	HTTP Request	185		13585	64
6	07.24.21.469	Thread Group 1-1	HTTP Request	196		13509	65
7	07.24.21.656	Thread Group 1-1	HTTP Request	187		13537	66
8	07.24.21.844	Thread Group 1-1	HTTP Request	187		13525	65
9	07.24.22.032	Thread Group 1-1	HTTP Request	183		13605	64
10	07.24.22.216	Thread Group 1-1	HTTP Request	198		13541	69
11	07.24.22.405	Thread Group 1-1	HTTP Request	187		13541	65
12	07.24.22.593	Thread Group 1-1	HTTP Request	184		13525	66
13	07.24.22.778	Thread Group 1-1	HTTP Request	188		13557	66
14	07.24.22.967	Thread Group 1-1	HTTP Request	194		13517	66
15	07.24.23.152	Thread Group 1-1	HTTP Request	188		13557	67
16	07.24.23.341	Thread Group 1-1	HTTP Request	188		13585	67
17	07.24.23.530	Thread Group 1-1	HTTP Request	187		13589	65
18	07.24.23.718	Thread Group 1-1	HTTP Request	182		13509	63
19	07.24.23.901	Thread Group 1-1	HTTP Request	187		13557	68
20	07.24.24.089	Thread Group 1-1	HTTP Request	188		13565	67
21	07.24.24.277	Thread Group 1-1	HTTP Request	186		13557	66
22	07.24.24.464	Thread Group 1-1	HTTP Request	185		13589	65

CHƯƠNG IX

Processor in JMeter: PreProcessor & PostProcessor

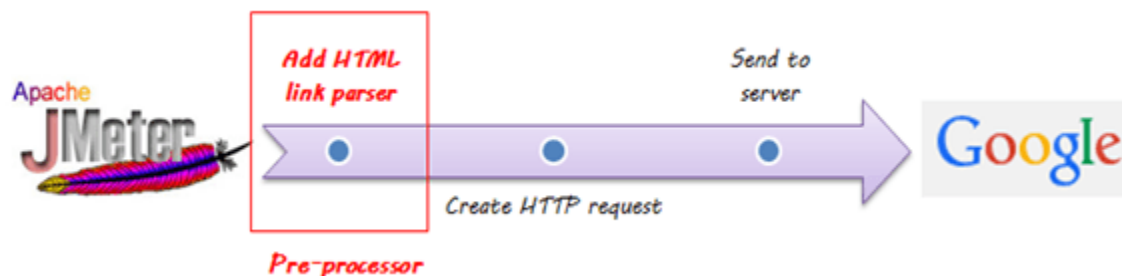
Bộ xử lý được sử dụng để sửa đổi Bộ lấy mẫu trong phạm vi của chúng.

Có 2 loại bộ xử lý:

- Pre-processor
- Post-processor

Pre-processor

Hãy xem xét một ví dụ đơn giản: giả sử bạn muốn JMeter “nhận” qua trang web đang thử nghiệm, phân tích cú pháp liên kết (kiểm tra tất cả các liên kết trên trang) và trả về HTML. Bạn sẽ thêm một số hành động, chẳng hạn như “trình phân tích cú pháp liên kết HTML” vào bộ điều khiển của mình trước khi tạo yêu cầu HTTP.



Post-processor

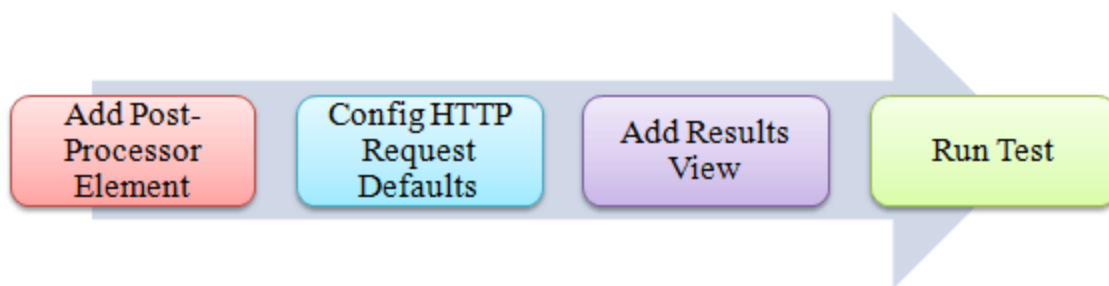
Post-processor thực hiện một số hành động sau khi thực hiện Yêu cầu Trình lấy mẫu. Hãy xem xét một ví dụ đơn giản: JMeter gửi một yêu cầu HTTP đến máy chủ web đang được kiểm tra (v.v. www.google.com) và nhận được phản hồi. Bạn muốn JMeter dừng kiểm tra nếu phản hồi của máy chủ là lỗi. Bạn có thể sử dụng bộ xử lý hậu kỳ để thực hiện tác vụ trên như sau:

Ví dụ về Post Processor:

Hướng dẫn này sẽ chỉ cho bạn các hướng dẫn từng bước về cách sử dụng Post-processor trong JMeter. Hãy bắt đầu với kịch bản thử nghiệm đơn giản.

1. JMeter gửi một yêu cầu HTTP đến máy chủ web được thử nghiệm www.google.com.
2. JMeter nhận phản hồi từ máy chủ Google.
3. Nếu phản hồi của máy chủ là lỗi, JMeter sẽ dừng kiểm tra.
4. Nếu máy chủ phản hồi OK (không có lỗi), JMeter sẽ tiếp tục kiểm tra.

Đây là lộ trình của ví dụ này:



Pre-condition:

Chúng ta sử dụng lại bước 1 và bước 2 trong [JMeter Performance Testing](#).

Bước 1: Thêm Thread Group

Nhấp chuột phải vào [Test Plan](#) và thêm thread group mới:

Add -> Threads (Users) -> Thread Group

Trong Thread Group control panel, nhập thông tin như sau:

Thread Properties

Number of Threads (users): 10

Ramp-Up Period (in seconds): 1

Loop Count: ☐ Forever 10

☐ Delay Thread creation until needed

☐ Scheduler

Cài đặt này cho phép JMeter tạo ra **10** user request đến <http://www.google.com> **10** lần.

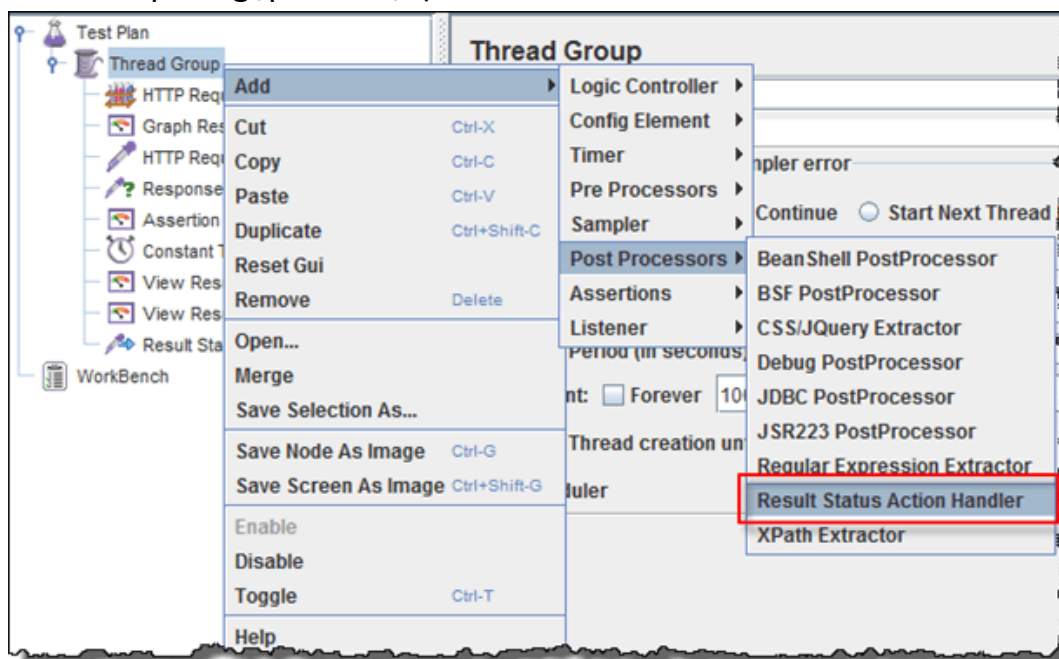
Bước 2: Thêm JMeter elements

- Thêm HTTP request default
- Thêm HTTP request

JMeter vẫn gửi request <http://www.google.com> đến Google server.

Bước 3: Thêm Post-Processor Element

Nhấp chuột phải **Thread Group** -> **Add** -> **Post Processor** -> **Result Status Action Handler** (**Result Status Action Handler** cho phép người dùng dừng thread hoặc cả bộ test nếu request gặp thất bại.)



Trong Result Status Action Handle Pane, chọn **Stop Test Now**. Lựa chọn này sẽ dừng bộ test nếu JMeter gặp error từ phản hồi của server.

Result Status Action Handler

Name:

Comments:

Action to be taken after a Sampler error

☐ Continue
 ☐ Start Next Thread Loop
 ☐ Stop Thread
 ☐ Stop Test
 ☒ Stop Test Now

Stop the test if JMeter get error from server response

Bước 4: Cấu hình HTTP Request

Mở HTTP Request Panel. Nhập “abc” vào the Path field.

HTTP Request

Name:

Comments:

Web Server

Server Name or IP: Port Number:

HTTP Request

Implementation: Protocol [http]: Method: Content encoding:

Path: *Make JMeter create Wrong URL request*
http://www.google.com/abc

☐ Redirect Automatically
 ☒ Follow Redirects
 ☒ Use KeepAlive
 ☐ Use multipart/form-data for POST

Parameters Post Body

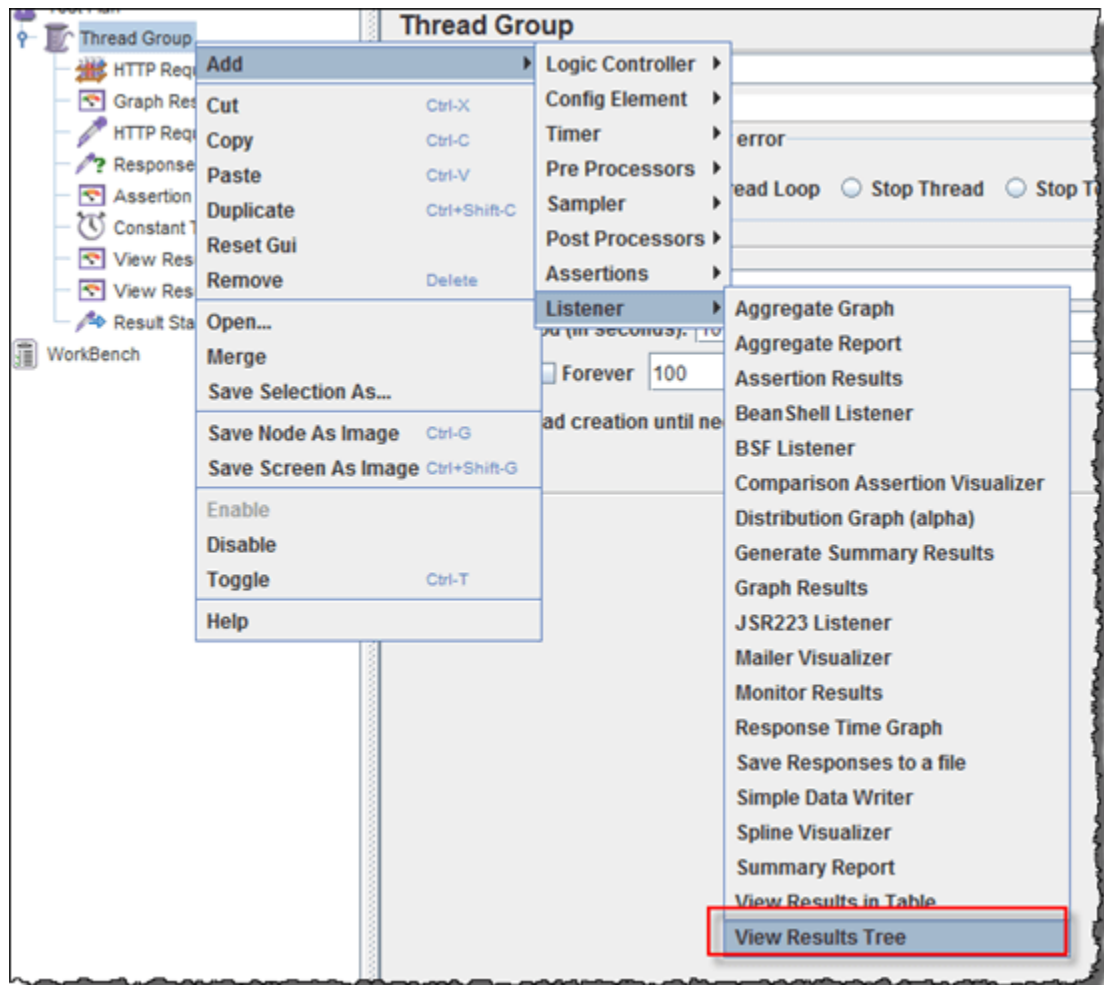
Send Parameters With the Request:

Name:	Value

Khi bạn nhập “abc” vào đường dẫn, JMeter sẽ tạo một yêu cầu URL tới máy chủ Google: <http://www.google.com/abc>. URL này không tồn tại trên máy chủ Google. Yêu cầu URL sai nên máy chủ Google sẽ trả về lỗi.

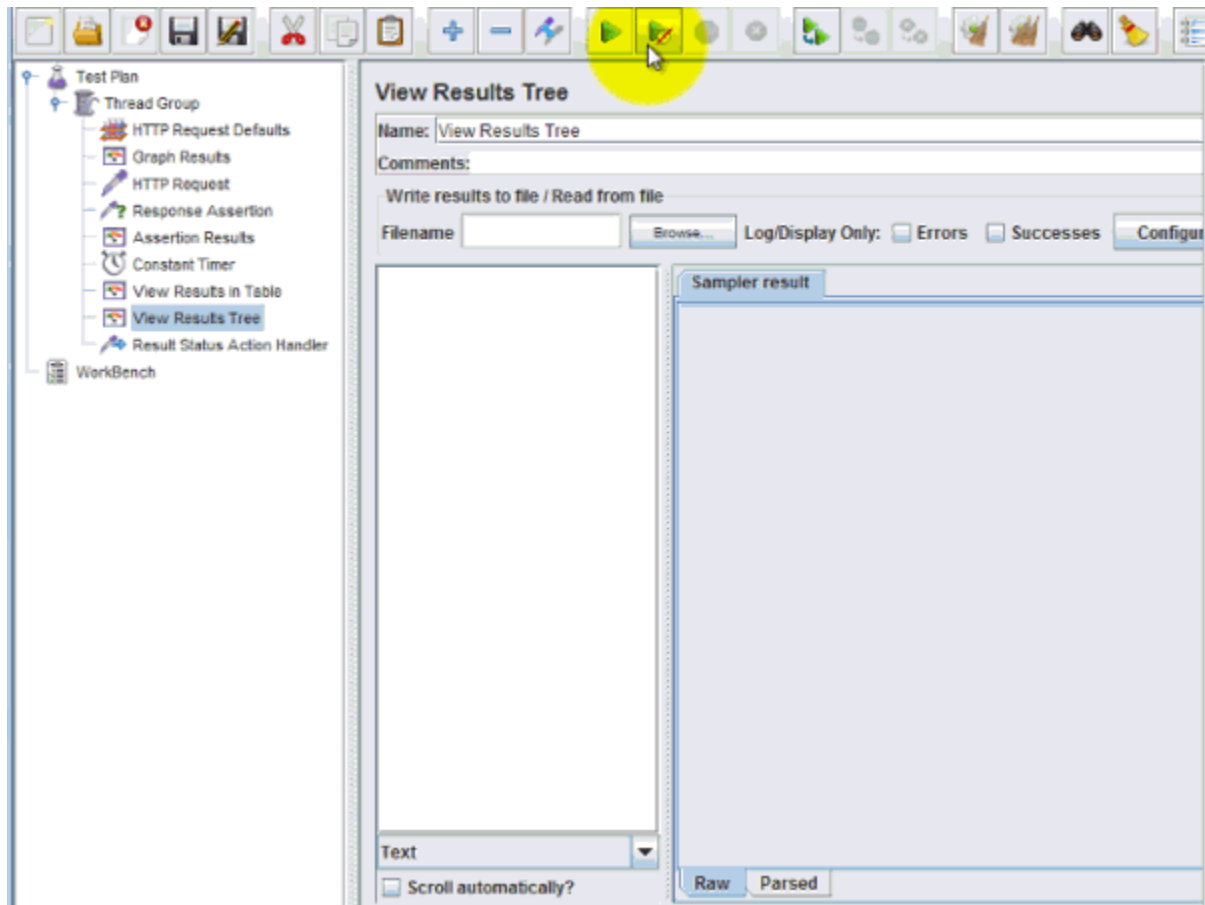
Bước 5: Thêm View Result Tree

Nhấp chuột phải vào **Thread Group** -> **Add** -> **Listener** -> **View Result Tree**

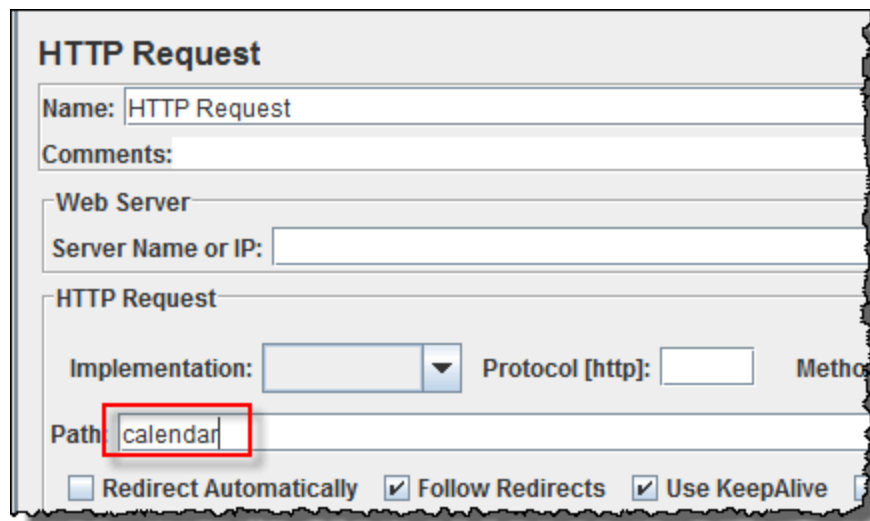


Bước 6: Chạy thử nghiệm

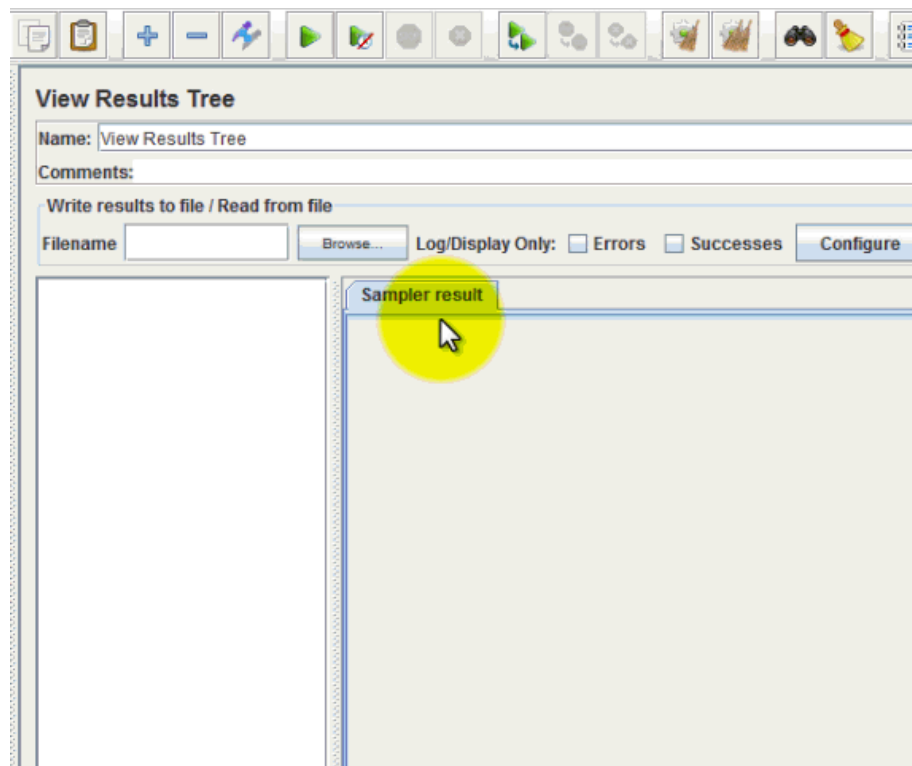
Chọn View Result Tree, nhấn nút Run trên thanh Menu. Bạn sẽ thấy phản hồi lỗi từ máy chủ Google và quá trình kiểm tra sẽ dừng lại nếu không hoàn thành 100 threads.



Bây giờ quay lại bước 4, mở khung Yêu cầu HTTP, nhập “calendar” vào khung. Nó làm cho JMeter tạo yêu cầu URL <https://calendar.google.com/calendar/u/0/r> tới máy chủ Google. Đây là yêu cầu URL chính xác nên máy chủ Google sẽ trả về OK (không có lỗi).



Chọn View Result Tree, nhấn nút Run trên thanh Menu. Bạn sẽ thấy phản hồi OK từ máy chủ Google và quá trình kiểm tra sẽ tiếp tục cho đến khi tất cả 100 Threads hoàn tất.



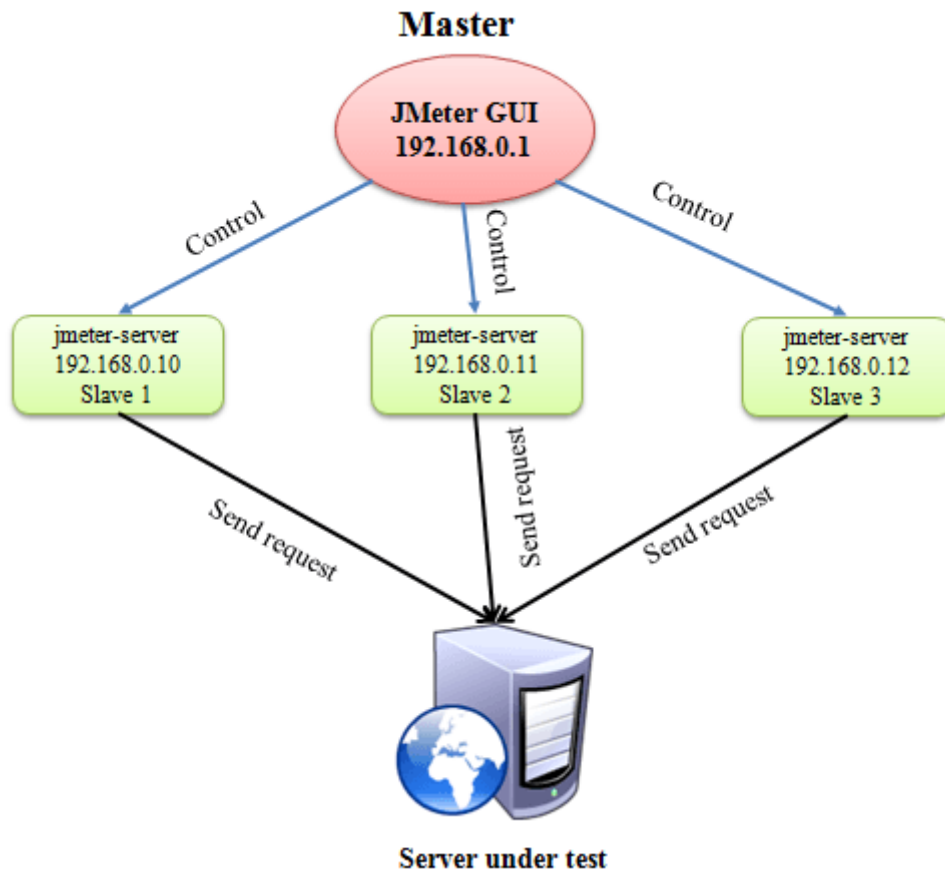
CHƯƠNG X

Jmeter Distributed (Remote) Testing: Master Slave Configuration

Distributed Testing

Kiểm thử phân tán (Distributed Testing) là một loại kiểm thử sử dụng nhiều hệ thống để thực hiện Stress Test. Thử nghiệm phân tán được áp dụng để thử nghiệm các trang web và ứng dụng máy chủ khi chúng đang làm việc với nhiều máy khách cùng một lúc.

Kiểm thử phân tán sử dụng mô hình client-server như hình bên dưới:



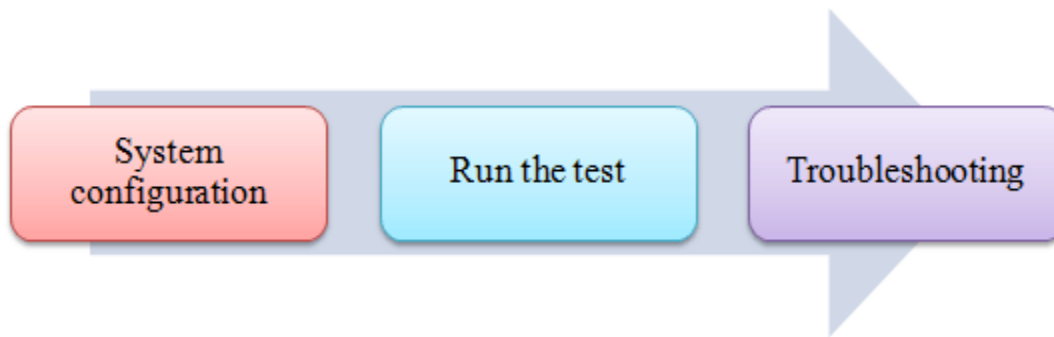
- **Master:** hệ thống chạy JMeter GUI, điều khiển từng Slave.
- **Slave:** hệ thống chạy JMeter-server, nhận lệnh từ master và gửi yêu cầu đến server đang test.
- **Target:** web server đang test, nhận request từ Slave.

Ví dụ Remote Test

Precondition:

- Tường lửa trên hệ thống đã bị tắt. Trong một số trường hợp, tường lửa vẫn có thể chặn lưu lượng. Bạn nên tắt tường lửa Windows hoặc tường lửa Linux.
- Tất cả các máy phải nằm trên cùng một mạng con. Nếu các máy không cùng mạng con, có thể chúng sẽ không nhận ra nhau trong mạng.
- Sử dụng cùng một phiên bản JMeter để tránh các lỗi/sự cố không lường trước được.

Đây là lộ trình cho thử nghiệm này:



Bước 1: Cấu hình hệ thống

Thiết lập **slave** systems, vào thư mục jmeter/bin và chạy tệp “jmeter-server.bat”.

Giả sử rằng một máy phụ có địa chỉ IP: 192.168.0.10. Trên cửa sổ, bạn sẽ thấy một cửa sổ xuất hiện như hình sau:

```
C:\Windows\system32\cmd.exe
Could not find ApacheJmeter_core.jar ...
... Trying JMeter_HOME=..
Found ApacheJMeter_core.jar
Created remote object: UnicastServerRef [liveRef: [endpoint:192.168.0.10:55053]
<local>,objID:[27ee6477:140cd425fd8:-7fff, 5855865920952165896111]
```

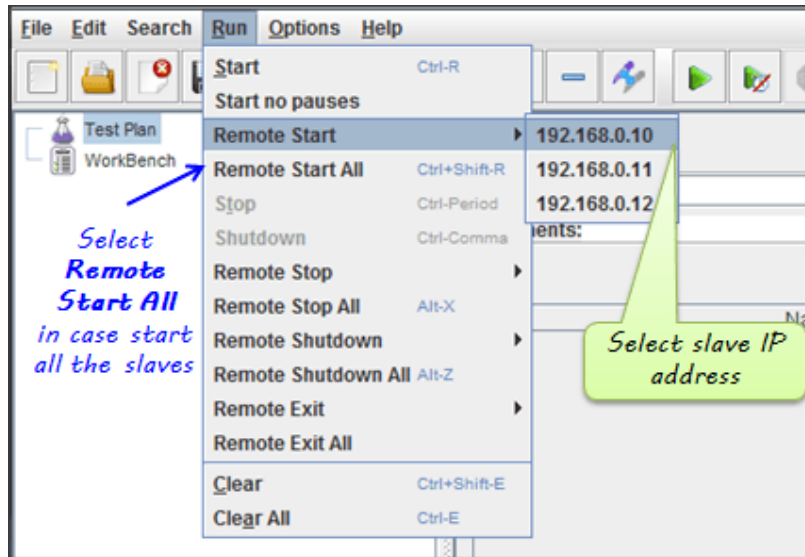
Trên hệ thống master, vào thư mục /bin và sửa file jmeter.properties, thêm IP như bên dưới

```
# Remote Hosts - comma delimited
remote_hosts=192.168.0.10,192.168.0.11,192.168.0.12
#remote_hosts=localhost:1099,localhost:2010
```

IP addresses of each slave

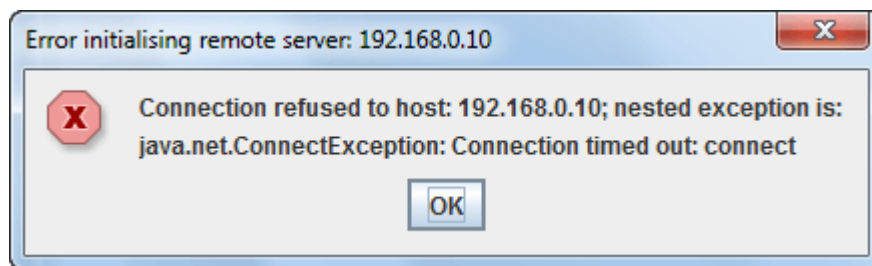
Bước 2: Chạy thử nghiệm

Tại thời điểm này, bạn đã sẵn sàng để bắt đầu load testing. Trên máy chủ, chạy JMeter GUI và mở kế hoạch kiểm tra. Nhấp vào Run trên thanh menu; chọn **Remote start** -> chọn địa chỉ IP của máy.



Step 3) Troubleshooting

Nếu bạn không thể chạy biểu mẫu kiểm tra máy ở trên và thấy lỗi bên dưới, chỉ cần yêu cầu chủ sở hữu của máy phụ chạy Tập JMeter-server.bat.



Vô hiệu hóa Tường lửa trên cả máy chủ và máy phụ để khắc phục lỗi này.

Hạn chế:

Có một số hạn chế cơ bản đối với thử nghiệm phân tán. Dưới đây là danh sách các mục đã biết:

- Máy chủ và tất cả máy khách phải nằm trên cùng một mạng con.
- Thử nghiệm phân tán yêu cầu máy chủ đích phải có sức mạnh xử lý lớn. Máy chủ mục tiêu có thể dễ dàng bị quá tải trong trường hợp nó nhận được quá nhiều yêu cầu bởi các bài kiểm tra JMeter phân tán.
- Một JMeter đơn lẻ chỉ có thể xử lý một số threads giới hạn (100-300 threads).
- Các bài kiểm tra JMeter phân tán rất phức tạp, khó cho người mới bắt đầu xây dựng.

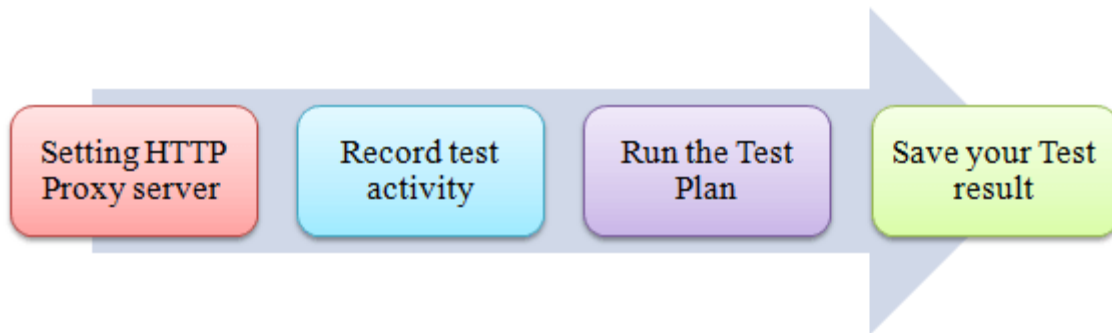
CHƯƠNG XI

HTTP Proxy Server in JMeter: Record Example Script

Record Testing giúp người kiểm tra ghi lại và chạy hoạt động của họ đối với mục tiêu kiểm tra. Nó là một loại thử nghiệm tự động nhưng dành cho nhiều người dùng. Hướng dẫn này hướng dẫn bạn cách sử dụng Proxy Server để ghi lại bài kiểm tra của bạn.

Proxy Server cho phép JMeter xem và ghi lại hoạt động của người dùng khi họ đang duyệt ứng dụng web bằng trình duyệt thông thường.

Đây là lộ trình của ví dụ thực tế này:

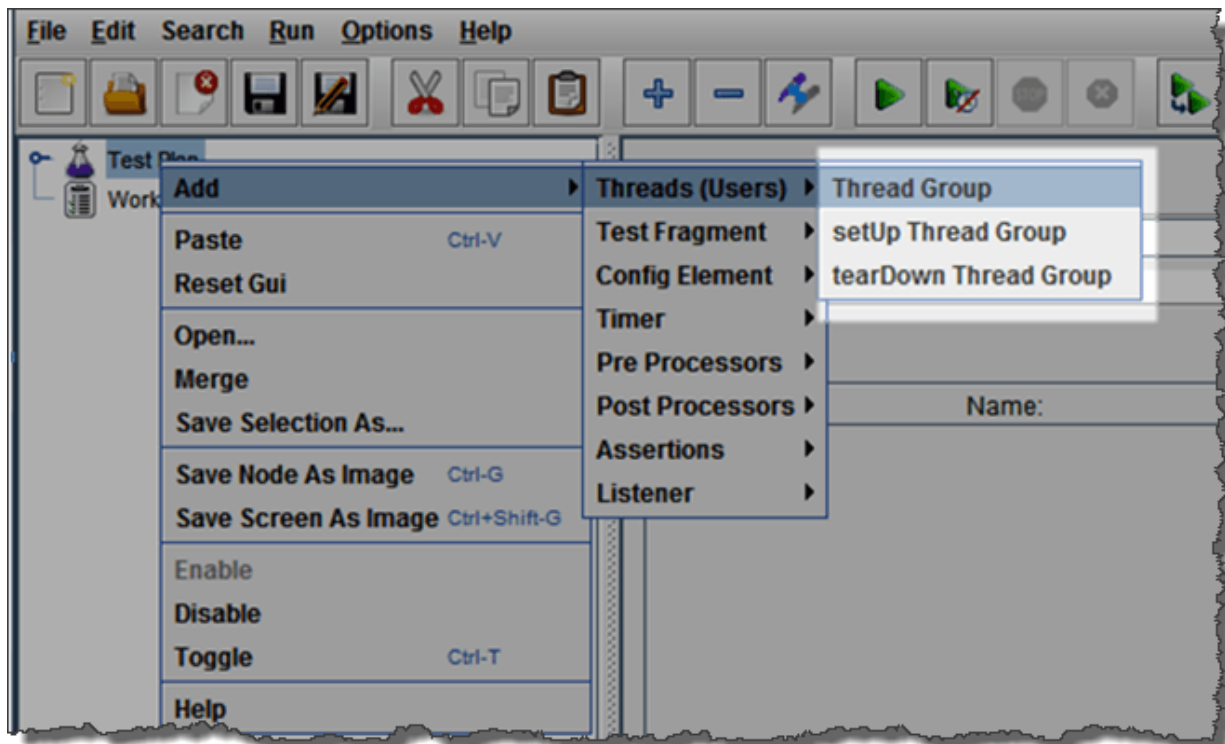


Bước 1: Đặt máy chủ HTTP Proxy

Đây là hướng dẫn từng bước để thiết lập proxy:

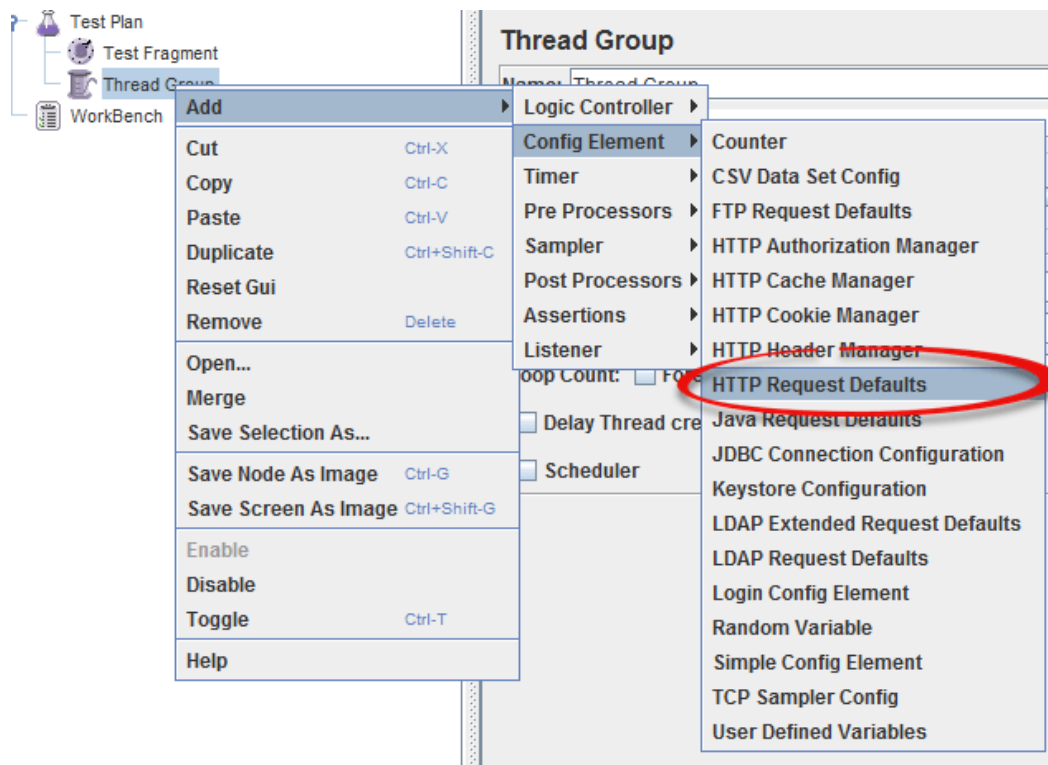
1. Khởi động JMeter
2. Chọn Test Plan trên cây
3. Thêm **Thread Group**

Nhấp chuột phải vào Test Plan và thêm một **Thread Group** mới: Add => Threads (Users) => Thread Group

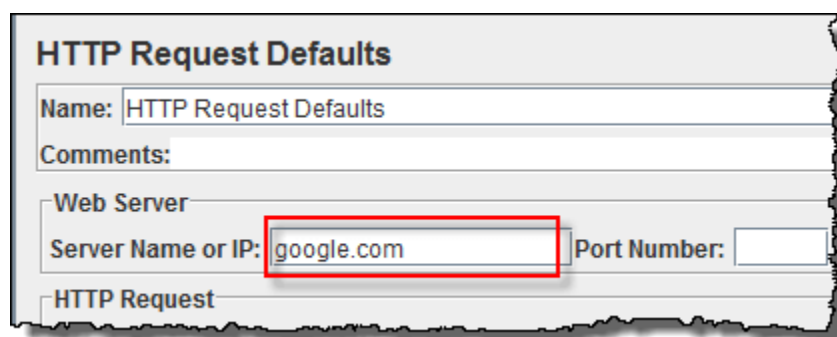


4. Thêm HTTP Request

Chọn Thread Group; nhấp chuột phải **Add** => **Config Element** => **HTTP Request Defaults**

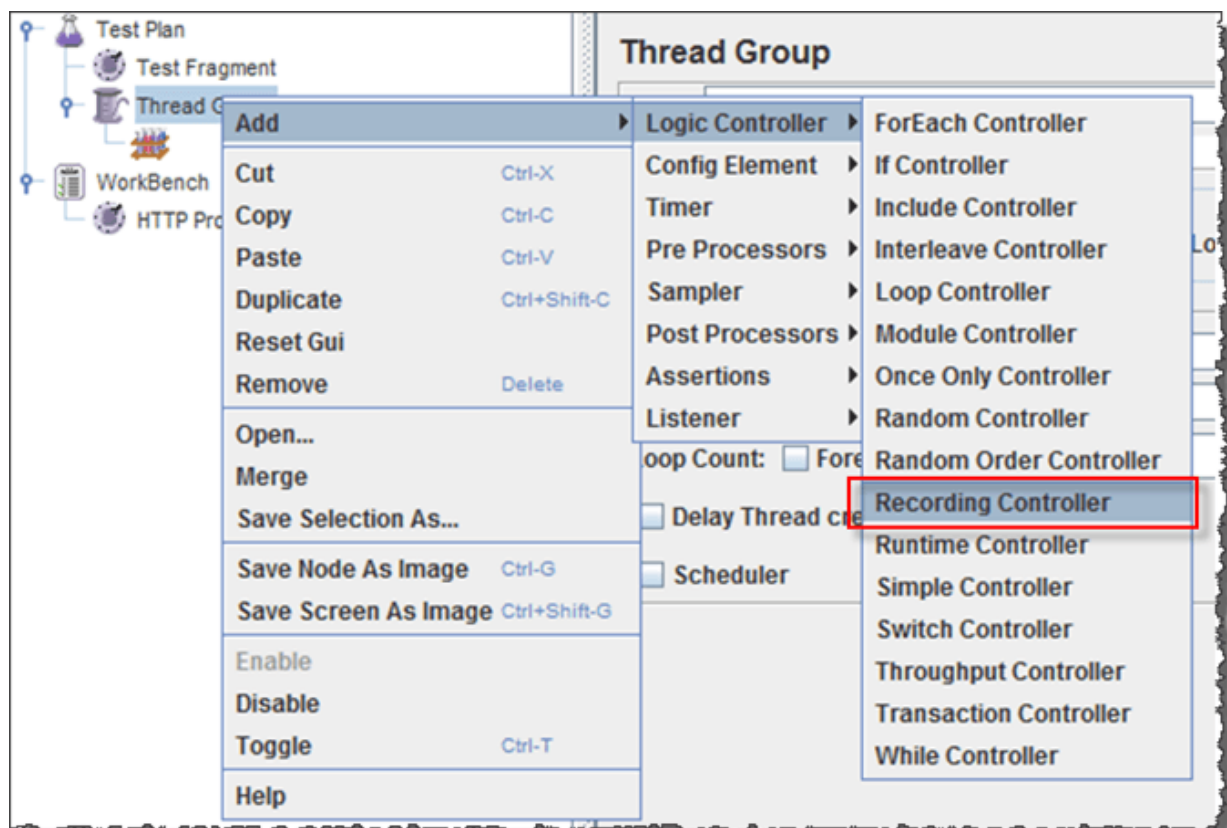


Trong phần tử HTTP Request Defaults mới: Trong tên Server hoặc IP, hãy nhập “google.com”. Bạn nên để trống các trường khác.



5. Add Recording Controller

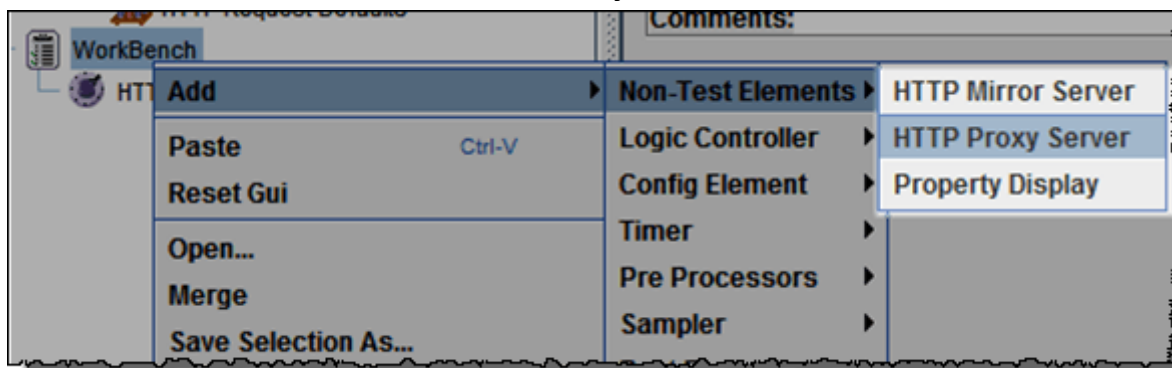
Nhấp chuột phải “Thread Group” và thêm một Recording Controller: **Add => Logic Controller => Recording Controller**



6. Thêm **Proxy Server** vào WorkBench

Nhấp chuột phải vào Workbench và thêm http proxy:

Add => Non-Test Elements => HTTP Proxy Server



7. Cài đặt **Target Controller** nơi các tập lệnh đã ghi của bạn sẽ được thêm vào.

HTTP Proxy Server

Name: HTTP Proxy Server

Comments:

Port: 8080 ☐ Attempt HTTPS Spoofing Optional URL match string:

Test plan content

Target Controller: Test Plan > Thread Group Grouping: Do not group samplers

☒ Capture HTTP Headers ☐ Add Assertions ☐ Regex matching

HTTP Sampler settings

Type: HTTP Request ☐ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☐ Retrieve All Embedded Res

Content-type filter

Include: Exclude:

8. Chạy Proxy Server

Return to HTTP Proxy Server, and click the **Start** button at the bottom. Now your JMeter proxy server start.

Quay trở lại HTTP Proxy Server và nhấp vào nút **Start** ở dưới cùng. Bây giờ JMeter proxy server bắt đầu.

HTTP Proxy Server

Name: HTTP Proxy Server

Comments:

Global Settings

Port: 8080

Test plan content

Target Controller: Test Plan > Thread Group Grouping: Do not g

☒ Capture HTTP Headers ☐ Add Assertions ☐ Regex matching

HTTP Sampler settings

Type: Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive

Content-type filter

Include: Exclude:

URL Patterns to Include

URL Patterns to Include

Add Delete Add from Clipboard

URL Patterns to Exclude

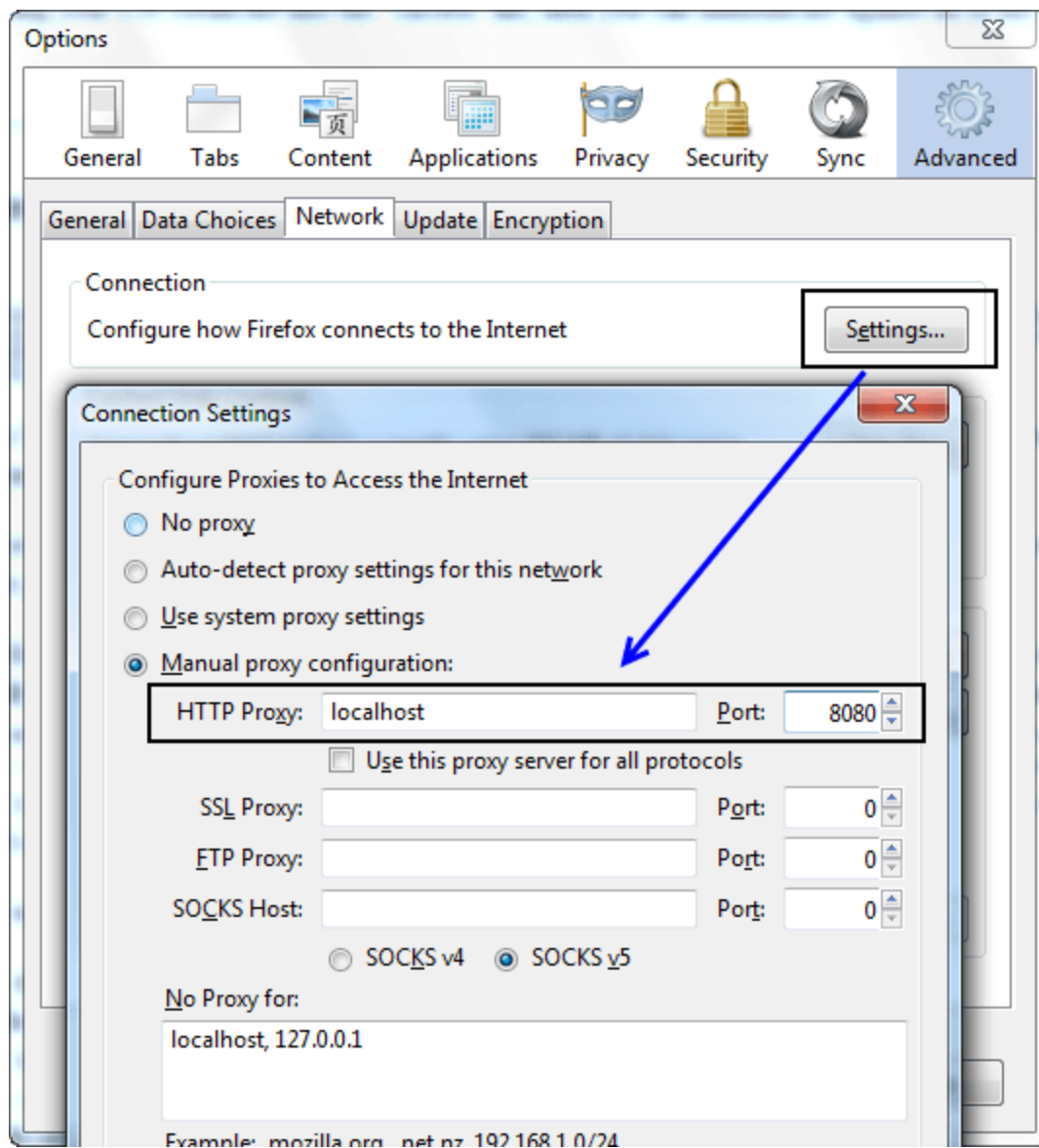
URL Patterns to Exclude

Add Delete Add from Clipboard Add sugges

Start Stop Restart

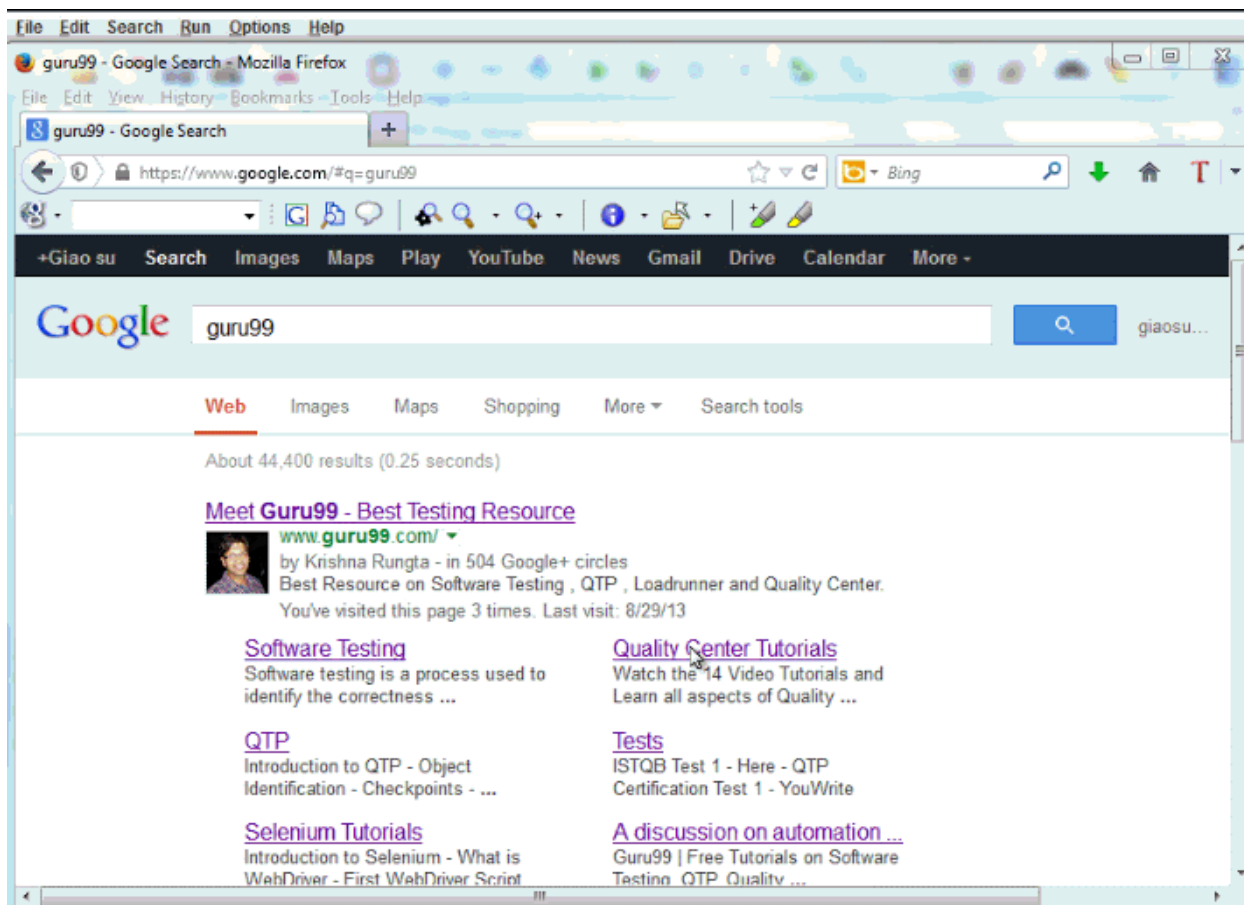
9. Khởi động Trình duyệt, chọn

Tool => Option => Advanced => Network => Setting => Nhập HTTP proxy
như hình bên dưới

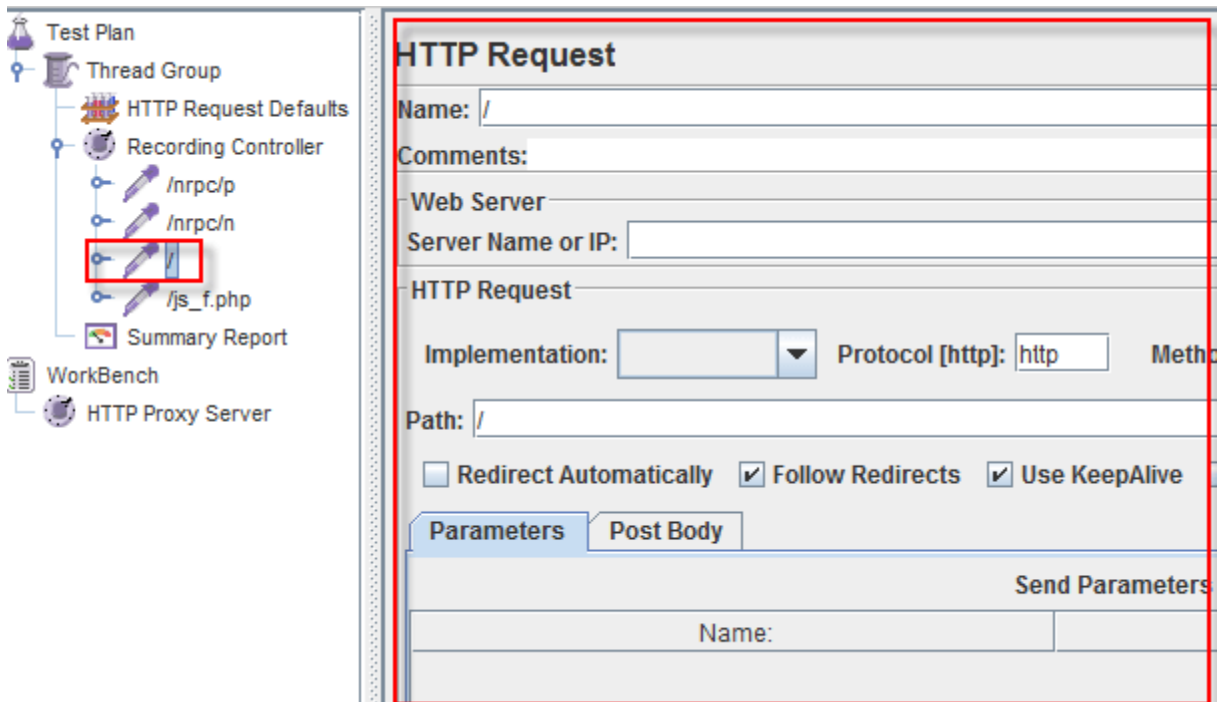


Bước 2: Ghi lại hoạt động

1. Bây giờ Khởi chạy <http://www.google.com> trong trình duyệt web của bạn (JMeter vẫn đang mở)
2. Thực hiện các hoạt động tìm kiếm từ khóa “guru99”.
3. Quay lại JMeter, trong HTTP Proxy Server, click Stop khi hoàn tất.



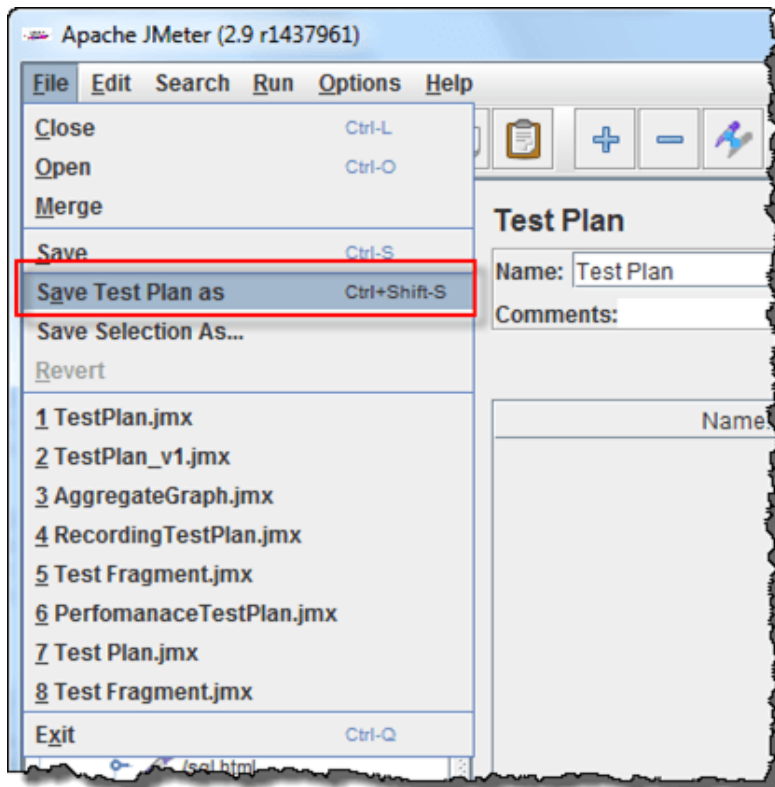
4. Sau khi ghi xong, bạn sẽ thấy JMeter tự động tạo một HTTP request mới như hình bên dưới.



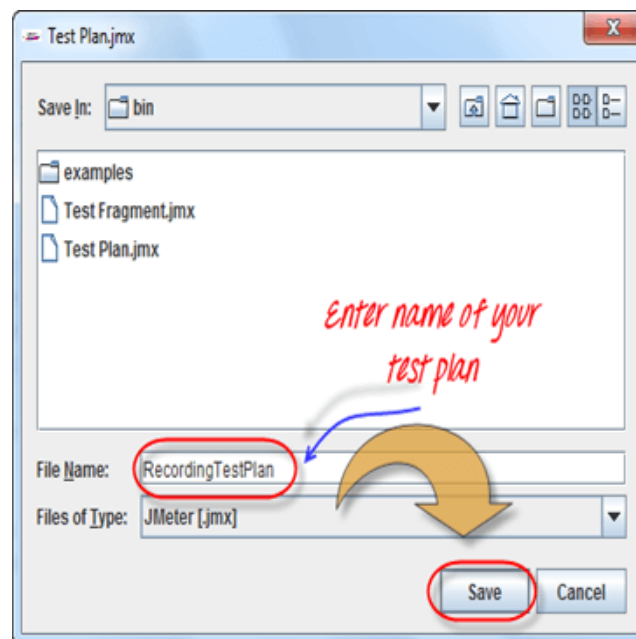
JMeter đã ghi lại yêu cầu của người dùng tới Trang chủ của trang web Google.
<http://www.google.com/>

Các yêu cầu HTTP khác hiển thị trong hình trên, bạn nên xóa chúng. Bởi vì đôi khi JMeter cũng ghi lại một số liên kết quảng cáo trong khi bạn đang tìm kiếm từ khóa trên Google. Chúng ta nên bỏ qua chúng trong Test Plan của mình.

5. Click File => Lưu Test Plan.

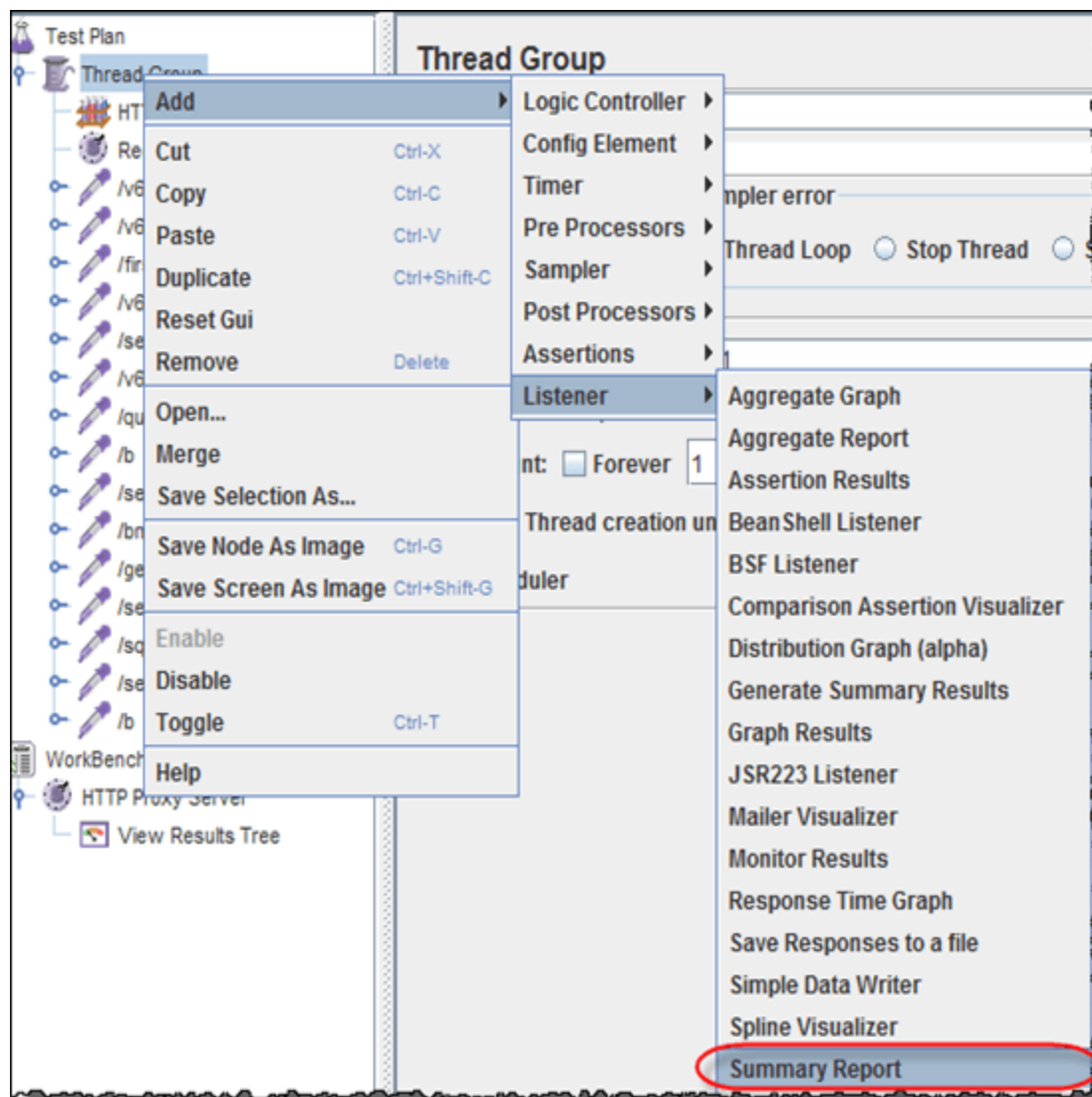


6. Hộp thoại hiển thị => nhập tên test plan tại trường File Name => Click Save.
Bây giờ Test Plan của bạn được lưu dưới tên RecordingTestPlan.jmx



Bước 3: Chạy Test Plan

1. Chọn **Thread Group** => **Add** => **Listener** => **Summary Report**



2. The Summary Report sẽ hiện một số thống kê cơ bản.

Summary Report

Name: Summary Report

Comments:

Write results to file / Read from file

Filename:

Log display Only: ☐ Errors ☐ Successes

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
TOTAL	0	0	9223372...	9223372...	0.00	0.00%	.0/hour	0.00	.0

Callouts:

- Average response time for particular HTTP request (points to Average column)
- How many exceptional cases were found (points to Error % column)
- number of requests per unit of time (points to Throughput column)
- See all the recorded HTTP request (points to Label column)
- Number of HTTP request (points to # Samples column)
- minimum, maximum response time taken by the HTTP request (points to Min and Max columns)
- denotes the error percentage in samples request (points to Error % column)

3. Chọn Thread Group, nhập thông tin như hình dưới

Thread Group

Name: Thread Group

Comments:

Action to be taken after a Sampler error

☒ Continue ☐ Start Next Thread Locally

Thread Properties

Number of Threads (users): 1

Ramp-Up Period (in seconds): 100

Loop Count: ☐ Forever 100

☐ Delay Thread creation until needed

☐ Scheduler

You can refer the article 5 [JMeter Performance Testing.doc](#) to know the detail about Thread Group configuring

4. Trước khi bắt đầu test, hãy chọn “Summary Report”. Khi bạn đã sẵn sàng chạy thử nghiệm, hãy chọn Run => Start (Ctrl+R). JMeter sẽ phát lại hoạt động của bạn sau 100 lần

Khi the test chạy, số liệu thống kê sẽ thay đổi cho đến khi thử nghiệm hoàn tất.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
TOTAL	0	0	92233720...	-9223372...	0.00	0.00%	.0/hour	0.00	.0

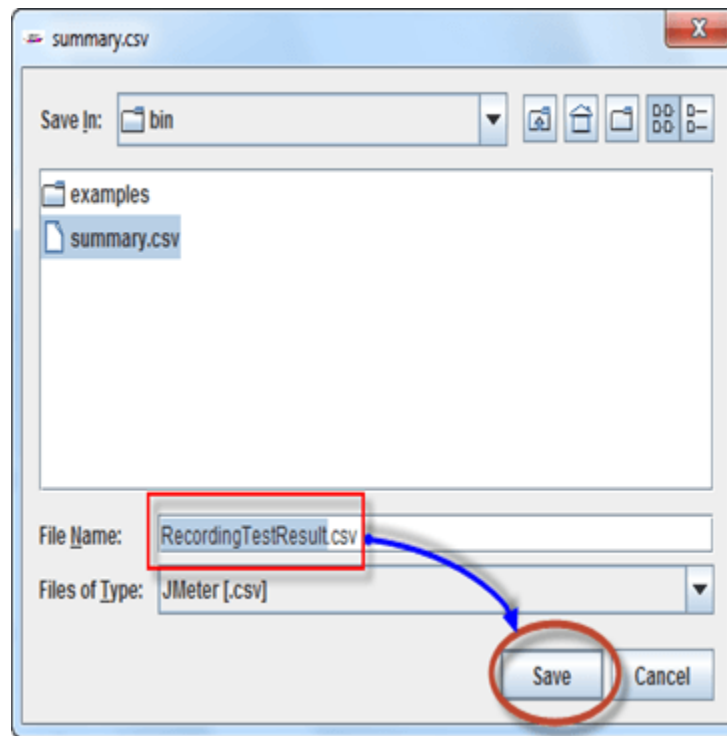
Bước 4: Lưu kết quả test

1. Chọn **Save Table Data** để lưu kết quả test vào file

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
/	100	535	356	782	97.34	100.00%	1.9/sec	6.37	3506.9
TOTAL	100	535	356	782	97.34	100.00%	1.9/sec	6.37	3506.9

☐ Include group name in label? **Save Table Data** ☒ Save Table Header

2. Nhập tên của kết quả test và nhấn vào Save. Test Result trong JMeter được lưu ở định dạng *.csv như mặc định.



CHƯƠNG XII

Các phương pháp tốt nhất để luyện tập JMeter Test và Load Testing

Kiểm tra JMeter là gì?

JMeter Test là quá trình kiểm tra được thực hiện bằng công cụ kiểm tra hiệu năng Jmeter của Apache. Nó giúp kiểm tra các ứng dụng web để kiểm tra hiệu suất, kiểm tra căng thẳng cũng như kiểm tra tải. Nó cũng hỗ trợ các tài nguyên tĩnh và động, đồng thời cung cấp các phân tích đồ họa khác nhau để kiểm tra hiệu suất của ứng dụng web.

Hướng dẫn khắc phục các hạn chế của JMeter trong môi trường phân tán:

1. Giới hạn số lượng Threads.
2. Sử dụng proxy server.
3. Sử dụng biến.
4. Giảm yêu cầu tài nguyên.
5. Kiểm tra nhật ký JMeter.
6. Xóa đường dẫn cục bộ khỏi Cấu hình tập dữ liệu CSV.
7. Tuân theo quy ước đặt tên tệp.

JMeter có một số hạn chế nhất là khi nó được chạy trong môi trường phân tán. Để sử dụng JMeter hiệu quả cho việc kiểm thử, bạn nên sử dụng các nguyên tắc sau:

Giới hạn số lượng Threads.

Số luồng tối đa bạn có thể chạy hiệu quả với JMeter là 300. Giới hạn này là do khả năng của phần cứng. Nếu JMeter được tạo để chạy với số lượng luồng nhiều hơn, độ chính xác của thông tin thời gian sẽ giảm.

Sử dụng Proxy server

Proxy server là một trong những phương pháp hay nhất của JMeter giúp bạn trừu tượng hóa một số phần tử phổ biến nhất định từ các mẫu được ghi lại. Hơn nữa, đó là các tính năng hữu ích để ghi lại thử nghiệm của bạn.

Sử dụng biến

Một số kế hoạch thử nghiệm cần sử dụng các giá trị khác nhau cho những người dùng/luồng khác nhau. Ví dụ: bạn có thể muốn kiểm tra trình tự yêu cầu thông tin đăng nhập duy nhất cho mỗi người dùng. Điều này rất dễ đạt được bằng cách sử dụng các biến JMeter.

Giảm yêu cầu tài nguyên

Chế độ GUI tiêu tốn rất nhiều bộ nhớ máy tính khi tải nặng. Nó gây ra các vấn đề về hiệu suất.

Có một số thực hành tốt nhất về load testing JMeter để giảm yêu cầu tài nguyên:

- Sử dụng chế độ không có GUI
- Vô hiệu hóa trình nghe “View Result Tree” trong quá trình load testing. Vì nó tốn nhiều bộ nhớ hơn và khiến JMeter đang chạy sẽ hết bộ nhớ.
- Vô hiệu hóa tất cả kết quả đồ thị JMeter
- Sử dụng định dạng kết quả kiểm tra CSV.
- Chỉ lưu kết quả xét nghiệm cần thiết. JMeter có thể mất nhiều thời gian để lưu kết quả kiểm tra rất chi tiết.

Kiểm tra nhật ký JMeter

Bất kỳ lỗi nào trong Test Plan hoặc thực hiện kiểm tra sẽ được ghi lại trong tệp nhật ký. Giám sát tệp nhật ký giúp bạn tìm ra lỗi sớm.

Xóa đường dẫn cục bộ khỏi Cấu hình tập dữ liệu CSV

Nếu bạn đang sử dụng tệp dữ liệu CSV hiện có mà bạn đã tạo trên máy tính cục bộ của mình, bạn nên xóa đường dẫn cục bộ hiện có (Đường dẫn hiện tại của tệp CSV). Nếu bạn không xóa đường dẫn cục bộ, JMeter không thể tìm thấy tệp dữ liệu CSV trên PC cục bộ của bạn.

Thực hiện theo quy ước đặt tên tệp

Không lưu kế hoạch kiểm tra dưới tên tệp phức tạp, chỉ sử dụng các ký tự chữ và số.