

HƯỚNG DẪN SỬ DỤNG JMETER TESTERTOP _13

1	Giới thiệu chung về kiểm thử hiệu năng (performance testing).....	2
1.1	Kiểm thử hiệu năng.....	2
1.1.1	Tại sao cần kiểm thử hiệu năng.....	2
1.1.2	Mục đích	2
1.1.3	Khái niệm.....	2
1.1.4	Các yếu tố được kiểm thử	2
1.1.5	Các thuật ngữ trong kiểm thử hiệu năng	2
1.1.6	Các loại kiểm thử hiệu năng.....	3
1.2	Các yếu tố ảnh hưởng đến kiểm thử tải	4
1.3	Quy trình thực hiện kiểm thử hiệu năng	4
1.3.1	Các hướng kiểm thử	4
1.3.2	Quy trình kiểm thử hiệu năng	4
1.3.3	Quy trình thực hiện load test.....	5
2	Giới thiệu về Jmeter.....	9
2.1	Giới thiệu Jmeter	9
2.1.1	Định nghĩa.....	9
2.1.2	Các đặc trưng	9
2.1.3	Lợi ích và cách hoạt động của Jmeter	9
2.1.4	Các bước tạo script kiểm thử hiệu năng.....	10
2.2	Các thành phần trong Test Plan	14
2.3	Các phần tử của một Test plan.....	15
2.3.1	ThreadGroup	15
2.3.2	Controllers.....	16
2.3.3	Listener.....	24
2.3.4	Timers	25
2.3.5	Configuration Elements	27
2.3.6	Assertion	34
2.3.7	Pre-Processor Elements.....	36
2.3.8	Post-Processor Elements	36
2.3.9	Thứ tự thực hiện các phần tử của 1 testplan.....	37
3	Hướng dẫn sử dụng	39
3.1	Các bước tạo test plan.....	39
3.2	Tham biến từ file excel	51
3.3	Bypass proxy	53
3.4	Remote testing	54
3.5	Tham biến các giá trị thay đổi.....	57
3.6	Winform.....	60
3.7	Mobile app	60
3.8	JDBC request	61
3.9	FTP request.....	66
3.10	TCP request	67
3.11	SOAP/XML-RPC request.....	68
3.12	Rest webservice	70
3.12.1	Lấy thông tin và kiểm tra rest service (sử dụng FireFox hoặc Chrome)	70
3.12.2	Cách chạy REST web service trên Jmeter.....	71
3.13	Running jmeter non-gui mode	72
3.13.1	Non-GUI mode.....	73
3.14	ZK framework	75
3.14.1	Record script	75
3.14.2	Các bước record requests	77
3.14.3	Tham biến các giá trị thay đổi: dtid, uuid, item.....	79
3.15	Cấu hình jmeter plugin để lấy cpu, ram,	80
3.15.1	Cách cấu hình.....	80
3.15.2	Cách xem thông số của server.....	84
3.15.3	GUI mode.....	86
3.16	Request đính kèm thêm file	87

1 Giới thiệu chung về kiểm thử hiệu năng (performance testing)

1.1 Kiểm thử hiệu năng

1.1.1 Tại sao cần kiểm thử hiệu năng

- Liệu ứng dụng có đáp ứng đủ cho người dùng 1 cách nhanh chóng?
- Liệu việc xử lý của ứng dụng có đáp ứng được yêu cầu người dùng, khả năng chịu tải và hơn thế nữa?
- Liệu ứng dụng có xử lý được số lượng giao dịch theo yêu cầu kinh doanh?
- Liệu ứng dụng có ổn định như mong muốn của người dùng về khả năng chịu tải không?
- Chúng ta có chắc rằng người dùng sẽ có kinh nghiệm trong việc khi nào thì đưa vào sử dụng thực tế?
- ⇒ Kiểm thử hiệu năng làm rõ ràng những rủi ro của việc triển khai phần mềm, ngăn ngừa hệ thống downtime và sẵn sàng trước các vấn đề gặp phải.

1.1.2 Mục đích

- Kiểm thử hiệu năng nhằm giảm bớt những rủi ro của việc ứng dụng, nâng cấp và phát triển phần mềm. Nó cũng có thể dùng để xác nhận và xác minh những thuộc tính chất lượng khác của hệ thống như: khả năng mở rộng, độ tin cậy và mức độ sử dụng tài nguyên.
- Xác định công suất vận hành tối đa của một ứng dụng như các điểm "thắt cổ chai" (bottleneck) và xác định phần tử nào là nguyên nhân gây ra điều đó.
- Thực hiện kiểm thử hiệu năng có thể chỉ ra được các vấn đề:
 - Hệ thống hoặc hệ thống con thực hiện một khối lượng công việc cụ thể nhanh thế nào
 - Khả năng đáp ứng của hệ thống với những hạn mức tải cụ thể.
 - Đánh giá thông lượng của hệ thống (Throughput).
 - Thời gian phản hồi (Response time).
 - Ứng xử của hệ thống trong trường hợp trong ngưỡng tải, cao tải, quá tải.
 - Khả năng mở rộng của hệ thống (Thiết lập baseline),.
 - Hỗ trợ tối ưu hệ thống (Chỉ ra điểm nghẽn trong hệ thống).

1.1.3 Khái niệm

Kiểm thử hiệu năng là:

- Một kỹ thuật kiểm thử phi chức năng để xác định và đánh giá các đặc tính của sản phẩm như: Tốc độ (respond Time), khả năng mở rộng (scalability), tính ổn định (stability), độ tin cậy (reliability).
- Cách kiểm thử đặt yêu cầu trên một hệ thống/thiết bị và đo lường sự vận hành/thao tác/trả lời của hệ thống/thiết bị, được thực thi dưới các điều kiện tải cao hoặc bình thường.

1.1.4 Các yếu tố được kiểm thử

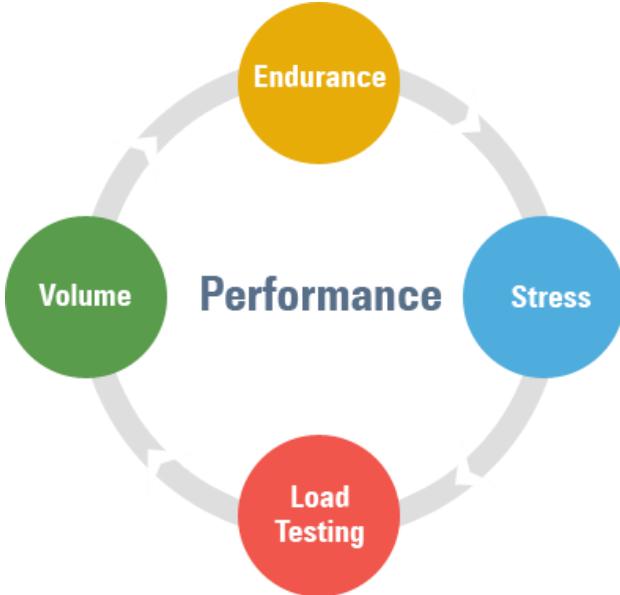
- ✓ Thời gian đáp ứng (response time)
- ✓ Tỷ lệ lỗi (pass/fail)
- ✓ Lưu lượng dữ liệu (throughput)
- ✓ Số yêu cầu trên 1 giây (TPS – transaction per second)
- ✓ Số người dùng đồng thời (concurrent user / virtual user)
- ✓ Tài nguyên máy (RAM, CPU...)

1.1.5 Các thuật ngữ trong kiểm thử hiệu năng

- CCU – Concurrent user: số lượng user đầy tải
- **Resoure Monitering:** (giám sát tài nguyên hệ thống)
- **TPS (Transaction Per Second):** Số Transaction được xử lý thành công trong 1 giây.

- **Throughput:** số Transaction xử lý thành công trong 1 đơn vị thời gian
- **Respond Time:** Thời gian phản hồi của hệ thống, tính từ lúc request được gửi đi tới khi nhận được respond trả về từ hệ thống.
- **Uptime:** Số user tối đa mà chức năng cho phép khi truy cập đồng thời.
- **Breaking point (Điểm chết của hệ thống):** Số user truy cập đồng thời vào hệ thống làm hệ thống bị chết hoặc dán đoạn.

1.1.6 Các loại kiểm thử hiệu năng



- Load testing
 - o Là một quá trình đẩy request và đo lường phản ứng của hệ thống hoặc thiết bị.
 - o Load testing được thực hiện để xác định hành vi của một hệ thống dưới điều kiện tải bình thường và điều kiện tải cao tải.
 - o Load testing giúp xác định khả năng hoạt động tối đa của hệ thống cũng như bất cứ tắc nghẽn (bottlenecks) hay yếu tố gây ra sự suy thoái (degradation).
 - o Các tham số: Respond Time, TPS, throughput,....
- Stress testing:
 - o Là một cách để kiểm tra (độ) tính tin cậy.
 - o Là việc đẩy hệ thống lên trên mức giới hạn của nó (điều kiện tải cực cao) để xác định các điểm yếu (weak point – các lỗi như đồng bộ, thiếu bộ nhớ...), thử làm gián đoạn trang web bằng cách tăng lượng tải cao hơn và kiểm tra xem hệ thống phản ứng như thế nào và phục hồi như thế nào.
 - o Ví dụ: Hệ thống có điểm chết (breaking point) là 1100 user. Spike test là thực hiện đẩy vào hệ thống khoảng 2000 tới 3000 user trong vòng 1-2 s xem hệ thống có bị các lỗi như tràn bộ nhớ hay mất đồng bộ hay không (spike test là một phần của stress test dùng để xác định hiệu năng hệ thống trong điều kiện tải tăng liên tục và vượt ra khỏi dự kiến trong 1 thời gian ngắn).
- Volume testing:
 - o Là kiểm thử phi chức năng.
 - o Kiểm thử với một lượng dữ liệu lớn. Dữ liệu có thể là kích thước cơ sở dữ liệu hoặc kích thước của 1 tập tin giao tiếp (.dat, .xml) là đối tượng của volume testing
 - o VD: Mở rộng kích thước của CSDL → kiểm tra hiệu suất của ứng dụng trên đó.
 - o **Capacity test:** xác định chiến lược để định cỡ (sizing) hệ thống nhằm đáp ứng được

yêu cầu hiệu năng của hệ thống trong tương lai.

- Endurance testing:

- o là một phần của Load test nhằm xác định các thông số hiệu năng của hệ thống trong điều kiện tải dự kiến trong 1 thời gian dài.
- o *Ví dụ: sau khi thực hiện kiểm thử hiệu năng xác định được ngưỡng của hệ thống là 1000 user. Con số khách hàng mong muốn là chức năng thanh toán đáp ứng 500 người dùng đồng thời ở thời điểm bình thường. Bài test độ bền sẽ thực hiện đầy vào hệ thống khoảng 500 user và cho chạy đi chạy lại liên tục trong vòng 2h(hoặc lâu hơn) để xác định xem hệ thống chạy có bền, có đúng và đảm bảo tiêu chí về Respond Time hay % chiếm dụng RAM, CPU như yêu cầu không.*

1.2 Các yếu tố ảnh hưởng đến kiểm thử tải

- Việc lập kế hoạch: Kế hoạch được vạch ra rõ ràng sẽ cho ta một kết quả khả quan, ngược lại nếu phức tạp sẽ cho ta kết quả của nó có xu hướng rối rắm.
- Mục tiêu được đặt ra: ta sẽ có câu trả lời rõ ràng.
- Kỹ năng của nhân viên
- Môi trường thử nghiệm kiểm thử tải
- Cơ sở dữ liệu: Trong môi trường kiểm thử, cơ sở dữ liệu phải được nạp sẵn hoặc là một bản sao của dữ liệu hiện hành hoặc là dữ liệu giả mà nó có kích thước và nội dung như dữ liệu hiện hành
- Công cụ kiểm thử tải: Phải có các tính năng quan trọng như: tham số hóa dữ liệu, nắm bắt các dữ liệu động, theo dõi cơ sở hạ tầng và hỗ trợ nhiều giao thức cho các ứng dụng

1.3 Quy trình thực hiện kiểm thử hiệu năng

1.3.1 Các hướng kiểm thử

- Thực hiện kiểm thử tải cho một hệ thống dựa trên các giới hạn hệ thống đã đưa ra trước
- Thực hiện kiểm thử tải để xác định các giới hạn cho một hệ thống để đưa ra các giới hạn cho một hệ thống, để đưa ra các giới hạn cho việc triển khai duy trì và phát triển hệ thống.

1.3.2 Quy trình kiểm thử hiệu năng

1.3.2.1 Lập kế hoạch test

Kế hoạch kiểm thử cần nêu rõ mục tiêu kiểm thử, yêu cầu kiểm thử, thiết kế kiểm thử và các quản trị dự án. Các bước thực hiện được mô tả một cách rõ ràng, mục đích thu được sau khi test phải được mô tả chi tiết. Xác định yêu cầu về hiệu năng, cấu hình của ranh giới và xác định khi nào bắt đầu kiểm thử

1.3.2.2 Tạo lập Scripts

Scripts là những thao tác thực tế của người dùng được lưu lại nhằm phục vụ cho việc kiểm thử hiệu năng. Với những công cụ hỗ trợ cho việc kiểm thử hiệu năng, chúng ta có thể lưu lại các bước thực hiện kịch bản đó dưới dạng mã lệnh. Mã lệnh này cũng có thể được chỉnh sửa một cách phù hợp nhất để phục vụ nhu cầu của tester trong tình huống đề ra.

1.3.2.3 Thiết lập kịch bản test

Kịch bản là trình tự các thao tác, các script sẽ được thực hiện trong một khoảng thời gian với 1 số người dùng xác định để đạt được các mục đích test khác nhau.

1.3.2.4 Thực thi kịch bản test

Với những kịch bản đã được tạo lập, tester sẽ thực hiện chạy chương trình trên nhiều máy tính khác nhau trong cùng một thời điểm để kiểm thử. Cùng với đó tester sẽ thực hiện việc quản lý và giám sát việc thực thi tình huống trong suốt quá trình chạy.

1.3.2.5 Phân tích kết quả test

Phân tích kết quả sau khi thực thi và đưa ra những kết luận về hiệu năng.

1.3.3 Quy trình thực hiện load test



1.3.3.1 Xác định các tiêu chí về hiệu năng

- Xác định các tiêu chí về hiệu năng có giá trị nhất khi xác định sớm trong vòng đời phát triển của ứng dụng. Các tiêu chí này thường được mô tả dưới dạng các thông số cụ thể và được lưu trữ lại để tất cả các thành viên tham gia kiểm thử có thể xem xét và thảo luận về chúng. Các tiêu chí sẽ được xác định thông qua các tài liệu đặc tả của website.
- Những tiêu chí về hiệu năng thường được đưa ra:
 - o Thời gian đáp ứng (thời gian phản hồi) của web site : là thời gian mà website phản hồi lại những yêu cầu từ người dùng. Ví dụ danh mục sản phẩm phải được hiển thị trong vòng chưa đầy 3 giây khi người dùng thực hiện truy cập vào trang chủ của website thương mại .
 - o Lượng truy cập : Ví dụ : hệ thống phải hỗ trợ 100 giao dịch mỗi giây
 - o Tài nguyên sử dụng : Ví xử lý, bộ nhớ , vào/ra đĩa cứng, vào/ra mạng

- Tải người dùng tối đa: Xác định có bao nhiêu người sử dụng có thể chạy trên một cấu hình phần cứng cụ thể.
- Các số liệu nghiệp vụ liên quan : Ví dụ như số lượng đơn đặt hàng hoặc số lượng các cuộc gọi được xử lý ở một thời điểm xác định

1.3.3.2 Xác định các hành động chính

1. Xác định tất cả các kịch bản cho Website
 - Ví dụ : những website thương mại điện tử thường phải sử dụng những kịch bản dành cho người dùng như : duyệt catalog, tìm kiếm sản phẩm, đặt hàng
2. Xác định các hoạt động liên quan đến kịch bản chính
 - Ví dụ : Một nút đặt hàng kịch bản sẽ bao gồm các hoạt động sau:
 - Đăng nhập vào trang web
 - Duyệt các danh mục sản phẩm
 - Tìm kiếm sản phẩm
 - Thêm các mục vào giỏ mua hàng
 - Xác nhận thông tin về thẻ tín dụng và đặt hàng
 - Xác định kịch bản thường được sử dụng nhất hay thường sử dụng tài nguyên nhiều nhất (đây sẽ là kịch bản chính được sử dụng để thực hiện Load Test)

1.3.3.3 Xác định các mẫu Workload

- Khi xác định mẫu workload, cần xem xét những đặc điểm sau đây để xác định các đặc tính cho kịch bản sử dụng:
 - Một kịch bản người dùng : bao gồm các hoạt động được thực hiện bởi người sử dụng để hoàn thành một nhiệm vụ . Điều này cũng có thể được coi như là một phiên người dùng
 - Một người sử dụng thông thường sẽ thực hiện các hành động ngắn quãng giữa các trang trong một phiên. Điều này được coi như là thời gian nghỉ
 - Một phiên giao dịch sẽ được tính thời gian trung bình khi được đánh giá với nhiều người sử dụng. Điều quan trọng trong thực tế diễn ra khi xác định mức tải là các người dùng sẽ thực hiện đồng thời, chồng chéo nhau

1.3.3.4 Xác định mức tải mục tiêu

- Xác định mức tải mục tiêu được áp dụng cho việc phân phối khối lượng công việc được xác định trong các bước trước. Mục đích của việc xác định các mức tải mục tiêu là để đảm bảo rằng các kiểm thử có thể được sử dụng để dự đoán hoặc so sánh với các điều kiện tải trọng hiệu suất.
- Những yếu tố đầu vào thường được sử dụng để xác định các tải mục tiêu:
 - Khối lượng công việc (hiện tại và dự kiến) Vì nó liên quan đến các mục tiêu hiệu năng
 - Kịch bản chính
 - Phân phối công việc
 - Đặc điểm của phiên giao dịch (danh sách các bước, thời gian, tỷ lệ người dùng mới...)

1.3.3.5 Xác định các thông số

- Trong quá trình kiểm thử hiệu năng, các số liệu thu thập được là không giới hạn. Tuy nhiên, việc thu thập quá nhiều số liệu có thể gây khó khăn khi phân tích cũng như tác động tiêu cực đến thực tế của ứng dụng. Vì vậy, điều quan trọng là xác định các số liệu có liên quan nhất đến mục tiêu hiệu năng, những điều cần thiết sẽ giúp xác định điểm tắc nghẽn.Chỉ những số

liệu được phân tisch một cách chính xác và cung cấp những thông tin có giá trị mới được lựa chọn.Một vài gợi ý giúp xác định các số liệu sẽ cung cấp những thông tin có giá trị nhất :

- Xác định câu hỏi liên quan đến hiệu năng của ứng dụng mà có thể dễ dàng kiểm tra được : ví dụ như thời gian đáp ứng thanh toán khi đặt hàng là gì , làm thế nào để nhiều đơn đặt hàng được đặt trong một phút. Với những câu trả lời cho những câu hỏi ở trên, xác định mục tiêu hiệu năng để so sánh : ví dụ như thời gian check out nên là 30 giây và tối đa là 10 đơn đặt hàng nên được đặt trong một phút. Các câu trả lời được dựa trên nghiên cứu thị trường, dữ liệu lịch sử,..
- Xác định các số liệu: sử dụng danh sách các câu hỏi và câu trả lời liên quan đến hiệu năng, xác định các số liệu cung cấp thông tin liên quan đến những câu hỏi và câu trả lời
- Xác định các số liệu hỗ trợ: Giúp xác định mức tải chấp nhận được cho ứng dụng. Các giá trị cơ bản giúp phân tích hiệu năng của ứng dụng ở các mức tải khác nhau.
- Đánh giá lại các số liệu thu thật được một cách thường xuyên : mục tiêu, ưu tiên , rủi ro và các vấn đề hiện tại bị ràng buộc để thay đổi trong quá trình của dự án. Với mỗi thay đổi, các số liệu khác nhau có thể cung cấp những giá trị nhiều hơn so với những giá trị mà trước đây đã xác định

1.3.3.6 Thiết kế kiểm thử chi tiết

- Việc thiết kế các kịch bản kiểm thử cụ thể cần sử dụng những kịch bản đã tạo ra, những số liệu chính được lựa chọn và các mẫu workload. Với mỗi thử nghiệm khác nhau sẽ có một mục đích khác nhau, thu thập dữ liệu khác nhau, các kịch bản khác nhau và có các mức tải mục tiêu khác nhau.
- Các điểm cần chú ý khi thiết kế kiểm thử bao gồm:
 - Không thay đổi thiết kế kiểm thử vì các thiết kế khó thực hiện trong công cụ
 - Nếu không thể thực hiện việc cài đặt thử nghiệm như thiết kế, hãy đảm bảo rằng những chi tiết liên quan đến việc cài đặt thử nghiệm đã được ghi lại.
 - Đảm bảo rằng mô hình có chứa tất cả các dữ liệu cần thiết để tạo ra những thử nghiệm thực tế.Người dùng lần đầu thường dành nhiều thời gian hoạt động trên từng trang hơn những người dùng có kinh nghiệm (đã từng sử dụng trang ở những lần trước)
 - Các dữ liệu kiểm thử tốt nhất là những dữ liệu thu được từ một cơ sở dữ liệu hoặc log file
 - Không nên quá bị cuốn vào những yếu tố phức tạp và bỏ qua những hành động đơn giản nhất

1.3.3.7 Chạy thử nghiệm

- Đây là bước sử dụng những công cụ có sẵn để thực hiện việc thực thi những gì đã thu được ở 6 bước trên. Ở bước này , tester sẽ thực hiện việc mô phỏng kịch bản đã được xây dựng với những mẫu workload đã được xây dựng tương ứng bằng những công cụ kiểm thử tự động
- Việc mô phỏng tải sai lệch có thể làm cho tất cả những thiết kế ở 6 bước trên trở nên vô ích. Để hiểu được các dữ liệu thu thập từ việc thực thi thử nghiệm, mô phỏng tải phải phản ánh được thiết kế thử nghiệm vì khi mô phỏng không phản ánh được thiết kế thì kết quả sẽ bị hiểu sai. Những bước khi chuẩn bị mô phỏng tải như sau:
 - Bước 1: Cấu hình môi trường thử nghiệm theo một cách mà nó phản ánh môi trường sử dụng thực tế càng nhiều càng tốt, và nếu có sự khác biệt, nên chú ý sự khác biệt đó.
 - Bước 2: Đảm bảo rằng các bộ đo hiệu năng cho các số liệu xác định được và không can thiệp vào tính chính xác của mô phỏng tải.

- Bước 3: Sử dụng các công cụ tạo tải trọng thích hợp để tạo ra tải với các đặc điểm được mô tả trong thiết kế thử nghiệm.
- Bước 4: Một số điểm lưu ý trong quá trình thực hiện thử nghiệm bao gồm:
 - + Bắt đầu với một số lượng nhỏ người dùng phân phối theo hồ sơ người dùng, và sau đó tăng dần tải trọng. Điều này là để có thời gian cho hệ thống ổn định giữa tăng tải trong khi đánh giá tính chính xác của các mô phỏng.
 - + Xem xét việc tiếp tục tăng tải và ghi lại hành vi cho đến khi đạt đến ngưỡng cho các nguồn lực được xác định trong mục tiêu hiệu suất đã được đặt ra, ngay cả khi tải đó vượt quá tải trọng mục tiêu quy định trong thiết kế thử nghiệm. Thông tin về hành vi của hệ thống khi đi qua ngưỡng xác định cũng quan trọng như giá trị của các số liệu tại tải mục tiêu của thử nghiệm.

1.3.3.8 Phân tích kết quả

- Phân tích các kết quả thu được để tìm điểm tắc nghẽn hiệu năng giữa mỗi lần chạy thử nghiệm hoặc sau khi toàn thành tất các thử nghiệm. Dưới đây là các bước để phân tích các dữ liệu:
 - Phân tích các dữ liệu thu thập được và so sánh kết quả đó với mức độ chấp nhận được của số liệu để xác định xem hiệu suất của các Website đang được thử nghiệm chỉ ra những kết quả gì.
 - Phân tích các số liệu đo để chuẩn đoán tắc nghẽn . Dựa trên các phân tích, nắm bắt số liệu bổ sung trong vòng kiểm tra tiếp theo

2 Giới thiệu về Jmeter

2.1 Giới thiệu Jmeter

2.1.1 Định nghĩa

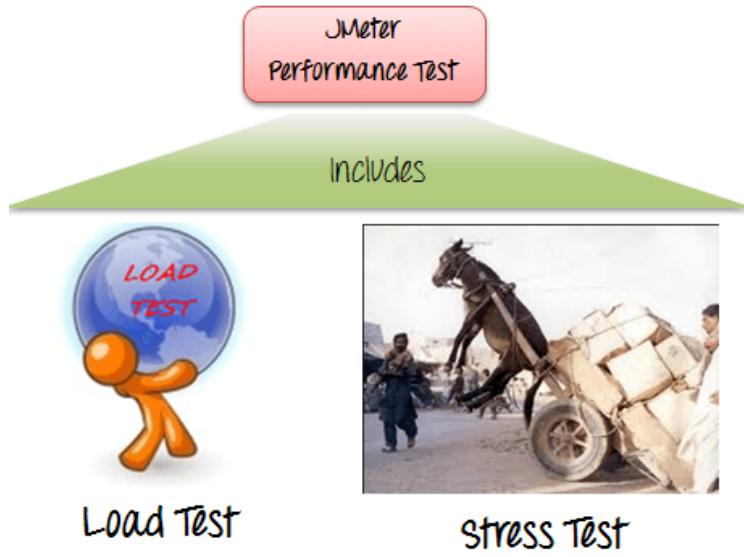
- JMeter là công cụ đo độ tải và hiệu năng của đối tượng, có thể sử dụng để test hiệu năng trên cả nguồn tĩnh và nguồn động, có thể kiểm tra độ tải và hiệu năng trên nhiều loại server khác nhau như Web-HTTP, HTTPS, SOAP, Database via JDBC, LDAP, JMS, Mail – SMTP(S), POP3(S), IMAP(S)…
- JMeter là một công cụ mã nguồn mở được viết bằng Java, cha đẻ của JMeter là Stefano Mazzocchi sau đó Apache đã thiết kế lại để cải tiến hơn giao diện cho người dùng và khả năng kiểm thử chức năng.
- Cách thức hoạt động: JMeter được dùng để giả lập một nhóm người dùng gửi các yêu cầu tới một máy chủ, nhận và xử lý các respond từ máy chủ sau đó trình diễn các kết quả đo cho người dùng dưới dạng các báo cáo tóm tắt, bảng biểu và đồ thị dạng cây, đồ thị đồ họa, …

2.1.2 Các đặc trưng

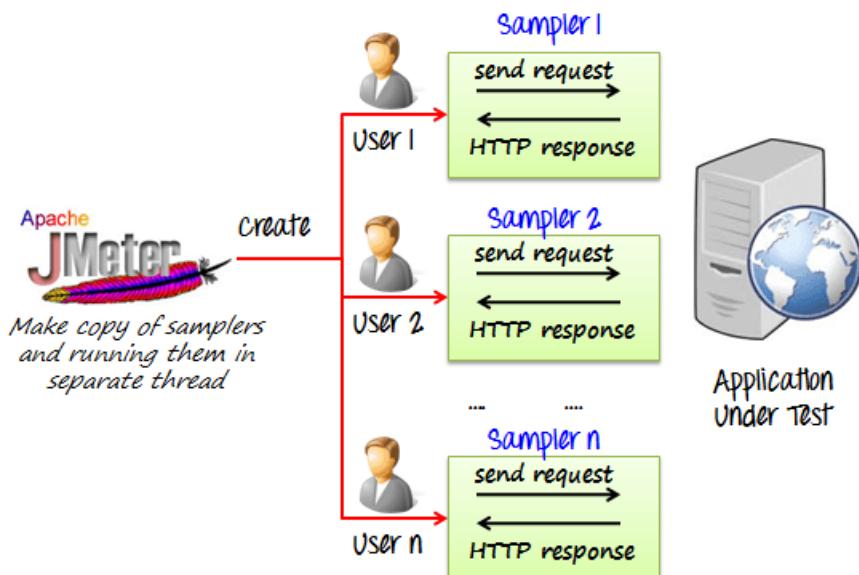
- Mã nguồn mở, giao diện trực quan, dễ sử dụng
- Có thể kiểm thử nhiều kiểu server: Web - HTTP, HTTPS, SOAP, Database - JDBC, LDAP, JMS, Mail - POP3,…
- Có thể chạy trên nhiều nền tảng hệ điều hành khác nhau, trên Linux chỉ cần chạy bằng một shell scrip, trên Windows thì chỉ cần chạy một file .bat
- Đa luồng, giúp xử lý tạo nhiều request cùng một khoảng thời gian, xử lý các dữ liệu thu được một cách hiệu quả.
- Đặc tính mở rộng, có rất nhiều plugin được chia sẻ rộng rãi và miễn phí
- Một công cụ tự động để kiểm thử hiệu năng và tính năng của ứng dụng.
- Jmeter lưu các test plan của nó dưới dạng XML, do đó có thể tạo 1 test plan bằng cách sử dụng trình text editor.

2.1.3 Lợi ích và cách hoạt động của Jmeter

- Jmeter có thể sử dụng để kiểm thử hiệu năng của hai nguồn tài nguyên tĩnh như là Javascript và HTML và tài nguyên động như JSP, Servlets, và AJAX.
- Jmeter có thể phát hiện ra một số lượng lớn các users trong cùng một thời điểm mà website có thể xử lý.
- Jmeter có thể cung cấp phần lớn các phân tích đồ họa của báo cáo performance.
- JMeter Performance Testing gồm:



- Load testing: Mô hình hóa dự kiến sử dụng bởi nhiều người dùng truy cập một dịch vụ website trong cùng thời điểm.
- Stress testing: Tất cả các web server có thể tải một dung lượng lớn, khi mà tải trọng vượt ra ngoài giới hạn thì web server bắt đầu phản hồi chậm và gây ra lỗi. Mục đích của stress testing là có thể tìm ra độ tải lớn mà web server có thể xử lý.
- **Cách thức hoạt động:** Jmeter thực hiện giả lập một nhóm người dùng gửi các yêu cầu tới một máy chủ, nhận và xử lý các phản hồi từ máy chủ.

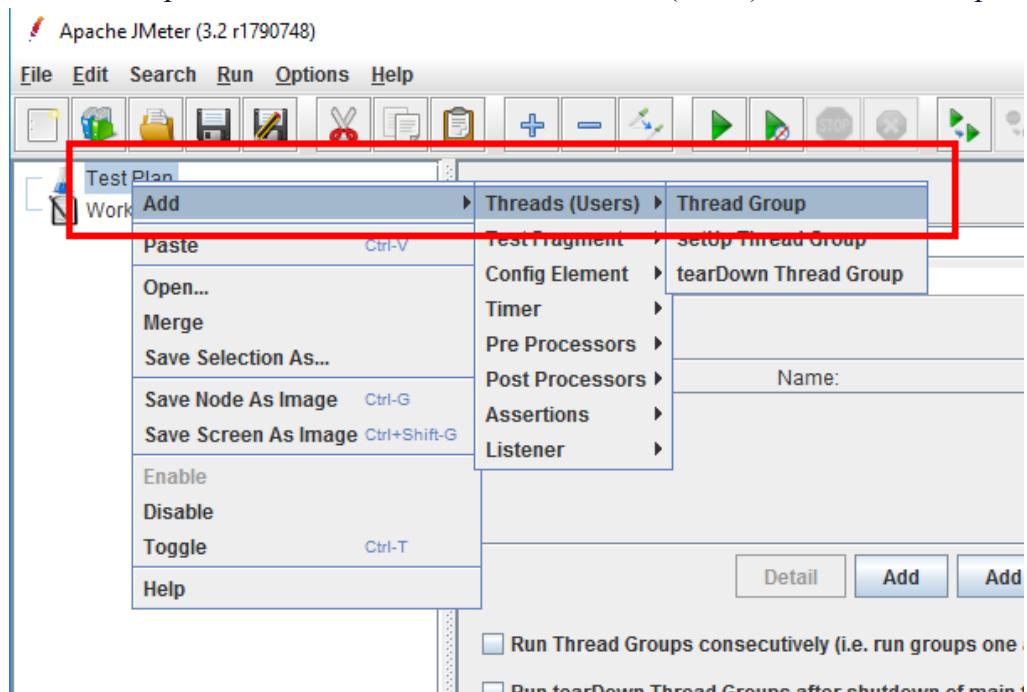


2.1.4 Các bước tạo script kiểm thử hiệu năng

Chi tiết sẽ được hướng dẫn trong phần Hướng dẫn sử dụng.

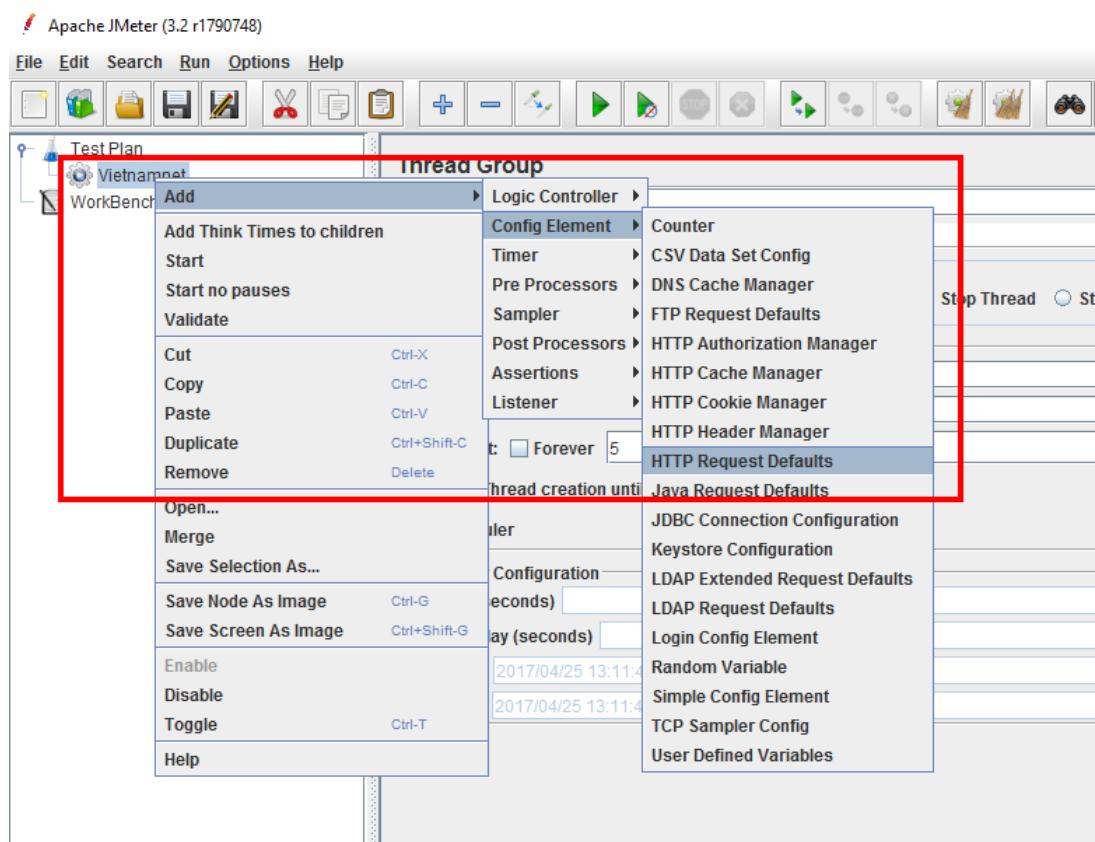
2.1.4.1 Thêm Thread Group

Click chuột phải vào Test Plan >> Add >> Threads (Users) >> Thread Group

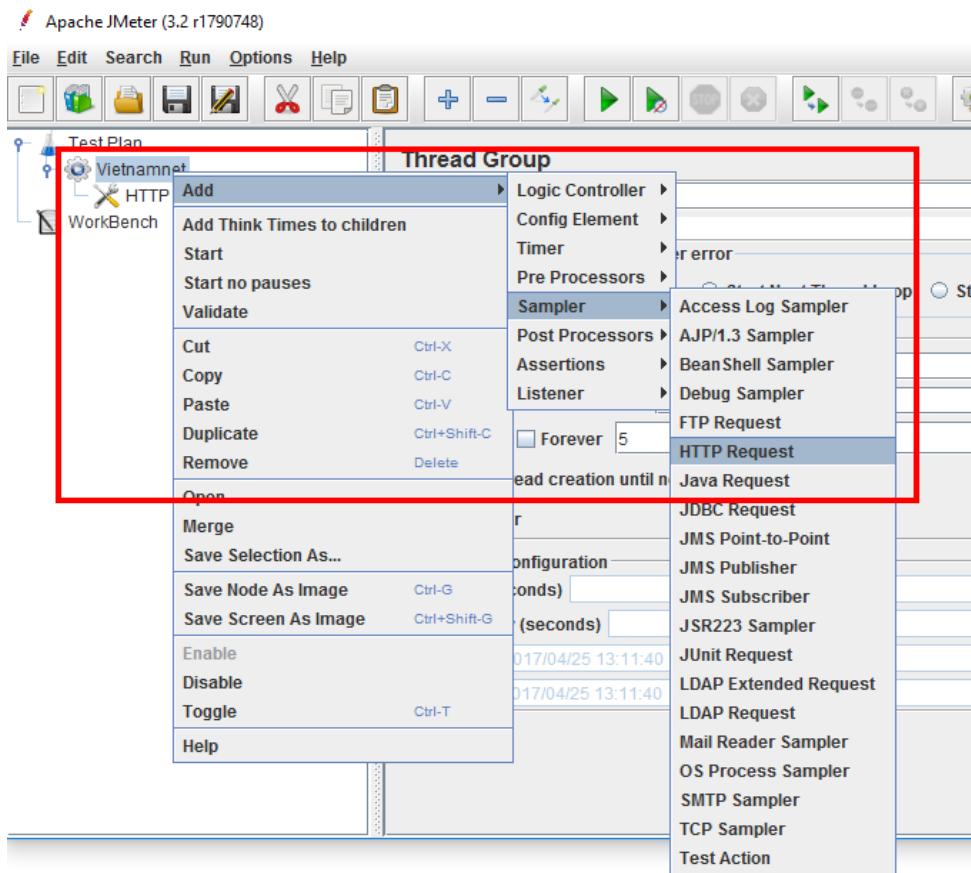


2.1.4.2 Thêm phần tử Jmeter

- **HTTP request default:** Click chuột phải vào Thread Group Vietnamnet >> Add >> Config Element >> HTTP Request Defaults

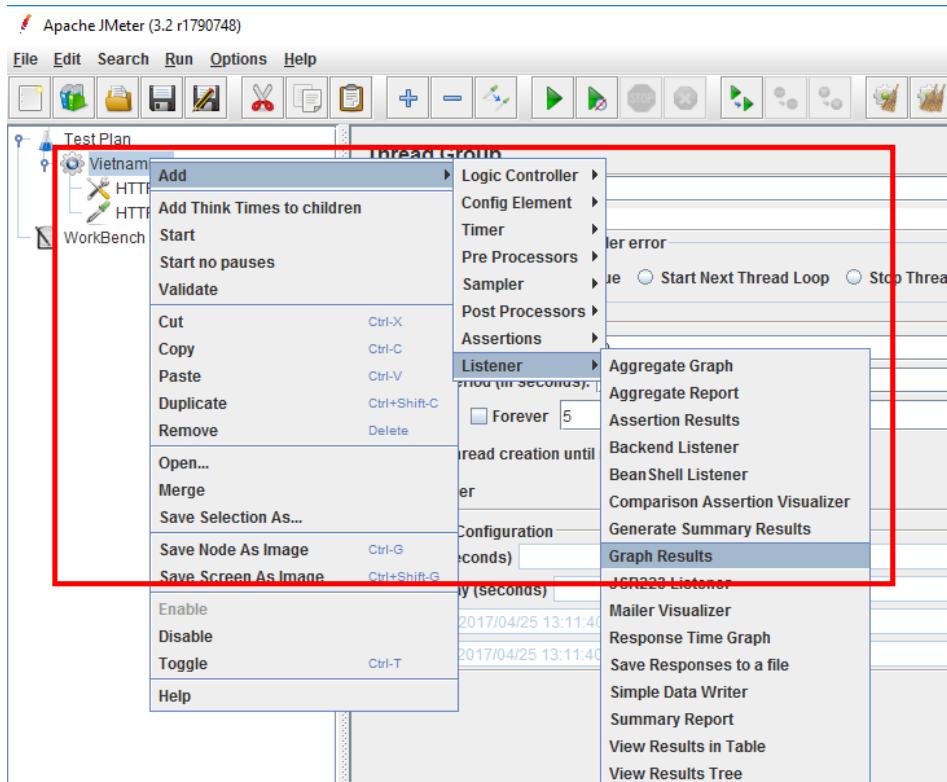


- **HTTP Request:** Click chuột phải vào Thread Group Vietnamnet >> Add >> Sampler >> HTTP Request



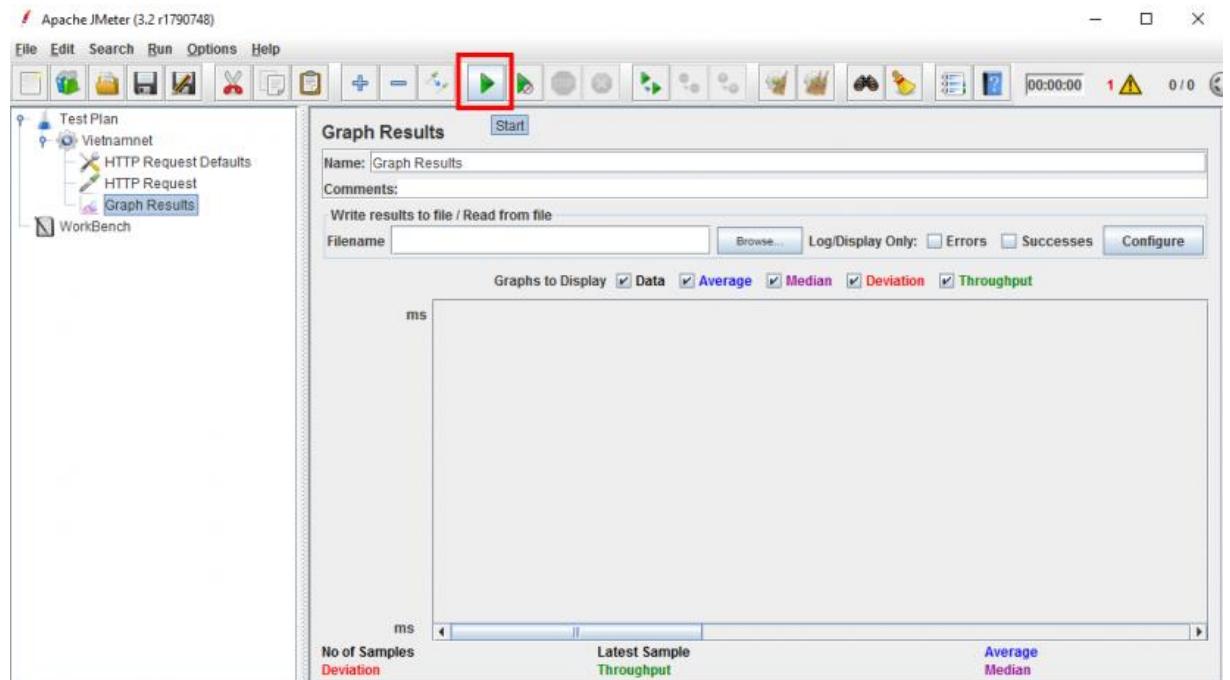
2.1.4.3 Thêm Graph result

Click chuột phải vào Thread Group Vietnamnet >> Add >> Listener >> Graph Results

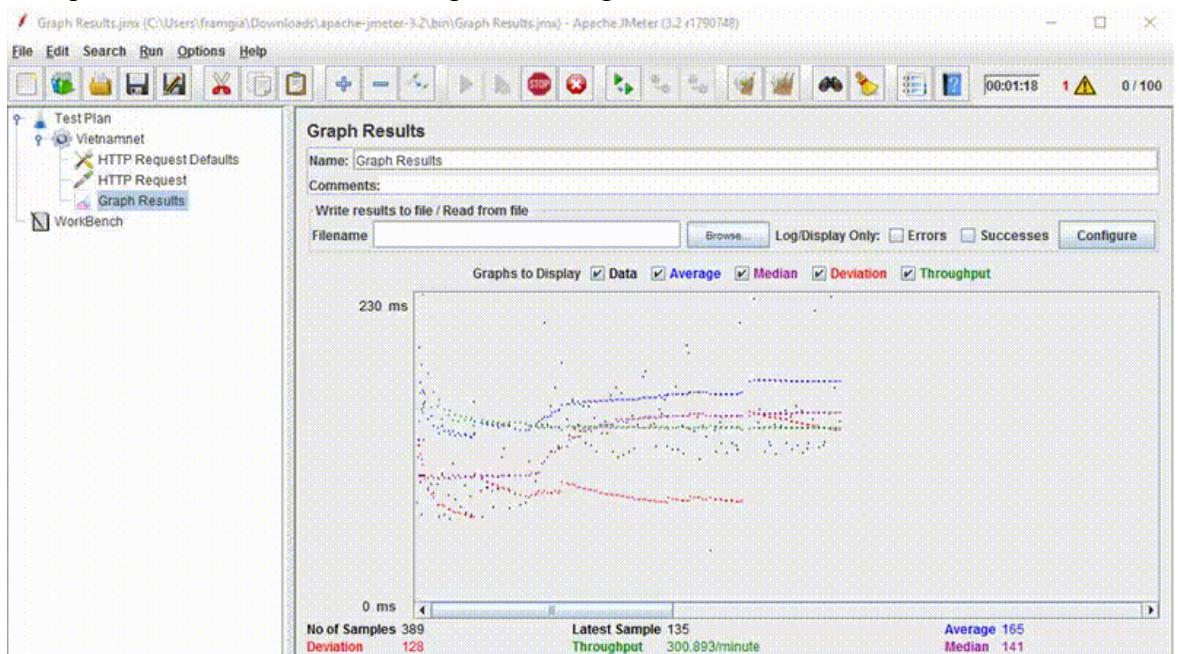


2.1.4.4 Chạy và lấy kết quả

- Click button "Start" hoặc Ctrl + R để chạy:

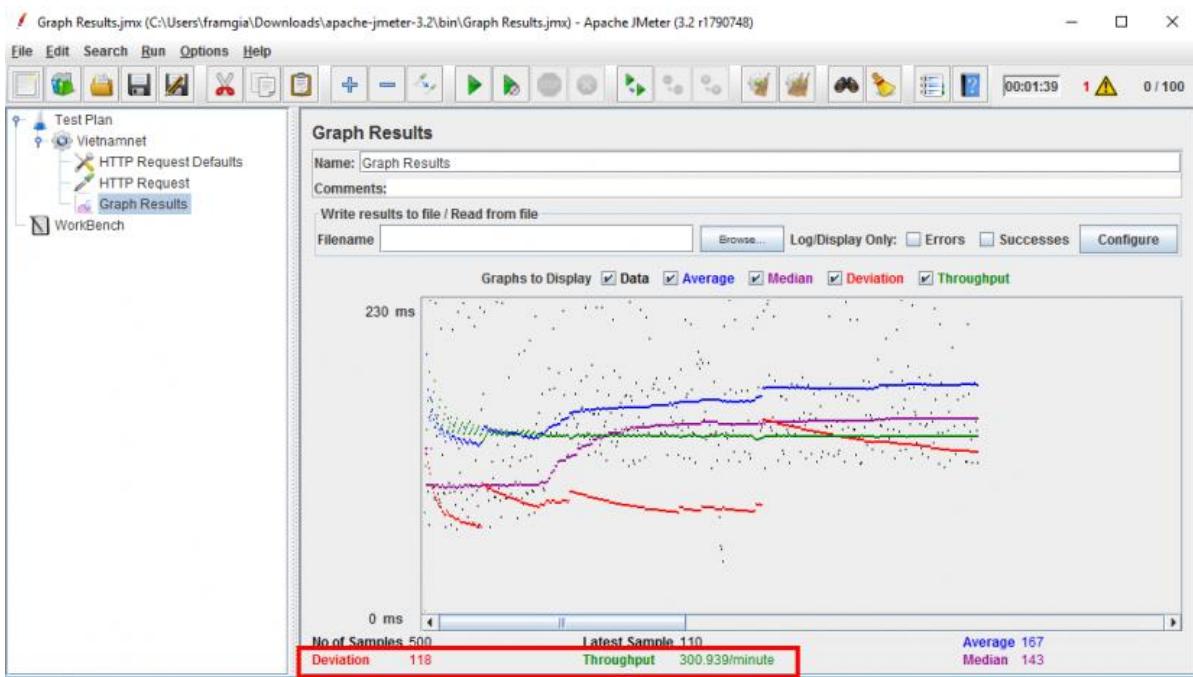


- Kết quả test được hiển thị trên Graph với thời gian thực tế:



2.1.4.5 Phân tích kết quả

Để phân tích Performance của Web server, ta tập trung vào hai thông số: **Throughput** và **Deviation**.



Throughput là thông số quan trọng nhất, nó miêu tả cho khả năng server có thể xử lý được độ tải lớn.

Deviation thể hiện sự sai lệch hiện tại so với mức trung bình, thông số này càng nhỏ thì càng tốt.

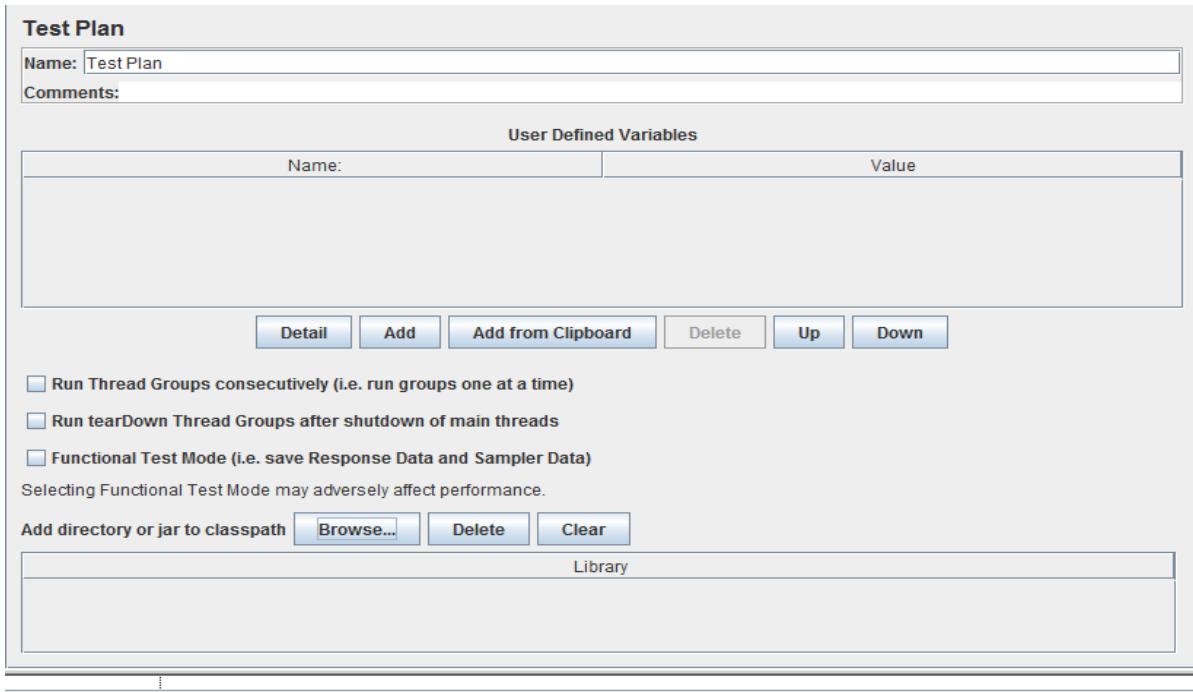
- Trong JMeter, xây dựng Test Plan để mô tả một tập các bước sẽ thực thi trước khi chạy, để kiểm tra hiệu năng, tính năng của ứng dụng. Một testplan hoàn chỉnh sẽ bao gồm 1 hoặc nhiều Thread Group, logic controllers, sample generating controllers, listeners, timers, assertions, and configuration elements.

2.2 Các thành phần trong Test Plan

- Test plan là một chuỗi các bước Jmeter sẽ thực hiện khi chạy.
- Có thể được lưu lại thành các file jmx
- JMeter report các warnings và error trong file jmeter.log, cùng các thông tin test
- Một test plan gồm các thành phần sau:

Thành phần	Mục đích
Thread Groups	Tạo ra được các yêu cầu để gửi tới server
Samples	Những phần tử này gửi các yêu cầu tới server. Có những samplers cho những kiểu request: HTTP/HTTPS, FTP, SOAP, JDBC, "Java", LDAP, MongoDB, TCP,...
Listeners Timers	Tập các kết quả của việc run test, cung cấp cho người dùng các công cụ hiển thị một cách trực quan, dễ hiểu
Logic Controllers	Nếu những request được định nghĩa trong các test plan của bạn được thực thi dựa phụ thuộc vào một vài logic, lúc đó sẽ cần đến Logic Controllers. Chúng thích hợp với cấu trúc if-then-else và loop trong Java hay các ngôn ngữ khác.
Configuration Elements	Chúng làm việc với Samples bằng cách thêm những thông tin chung với những Request

Thành phần	Mục đích
Thread Groups	Tạo ra được các yêu cầu để gửi tới server
Assertions	Cho phép bạn kiểm tra nếu responses bạn lấy chứa dữ liệu mong đợi hay nhận trong phạm vi thời gian đã định sẵn



Một test plan có các đặc tính như sau:

- **User Defined Variables:** cho phép định nghĩa các biến tĩnh, từ đó, cung cấp các giá trị lặp lại trong test của người dùng, như là tên server, cổng, ... Ví dụ, nếu muốn test một ứng dụng trên server www.example-jmeter.net, có thể định nghĩa một biến server với giá trị www.example-jmeter.net. Giá trị này sẽ thay cho biến "\${server}" ở bất kỳ vị trí nào trong test plan.
- Tùy chọn **Run Thread Groups consecutively (i.e. run groups one at a time):** Chạy liên tiếp
- **Run teardown Thread Groups after shutdown of main threads:**
- **Functional test Mode (i.e. save Response Data and Sampler Data):**
- Nếu có 2 hay nhiều Thread Groups trong Test Plan, lựa chọn này sẽ yêu cầu Jmeter chạy serially các ThreadGroup. Nếu không, Jmeter sẽ chạy các Thread Groups simultaneously hoặc parallel.
- Add directory or jar to classpath: Cho phép người dùng thêm vào các file JAR, hoặc các thư mục, trong trường hợp muốn tạo thêm các extension cho Jmeter. Tuy nhiên, cần lưu ý khởi động lại Jmeter khi có thay đổi. Ngoài ra, có thể sử dụng cách khác, đó là copy tất cả các file JAR vào thư mục lib của JMETER. Một cách khác nữa là cấu hình trong Jmeter properties file, Edit "#user.classpath=../classes;../jars/jar1" để thêm vào các thư viện.

2.3 Các phần tử của một Test plan

2.3.1 ThreadGroup

- Thread group là thành phần quan trọng trong Jmeter. Một Thread Group đại diện cho một nhóm người dùng, và nó chứa tất cả những yếu tố khác. Mỗi Thread Group sẽ mô phỏng

những người dùng để thực hiện một trường hợp thử nghiệm cụ thể. Thread Group cho phép tester thực hiện những tùy chỉnh về:

- Number of Threads (users): tổng số thread hoặc user thực hiện tại 1 lần chạy của test plan
- Ram-Up Period (in seconds): Thời gian để khởi tạo (load) tất cả các user (số lượng user được khai báo trong Number of Threads)
- Loop count: số lần test plan thực hiện
- Nếu chọn forever → test plan chạy không dừng lại cho đến khi người dùng stop test plan
- Scheduler: có thể cấu hình start time, end time, duration và start up delay cho load test plan
 - Start time: thời gian bắt đầu chạy test plan
 - End time: thời gian dừng chạy test plan
 - Duration: khoảng thời gian chạy test plan
 - Start delay (seconds): jmeter sẽ không load bắt kè một user nào trong vòng 5s đầu tiên.
- Để thực hiện thêm Thread Group, Click chuột phải vào Test Plan → Add → Thread Group

2.3.2 Controllers

Jmeter có 2 loại controllers: Samplers và Logical Controller, có tác dụng điều khiển thực hiện quá trình test.

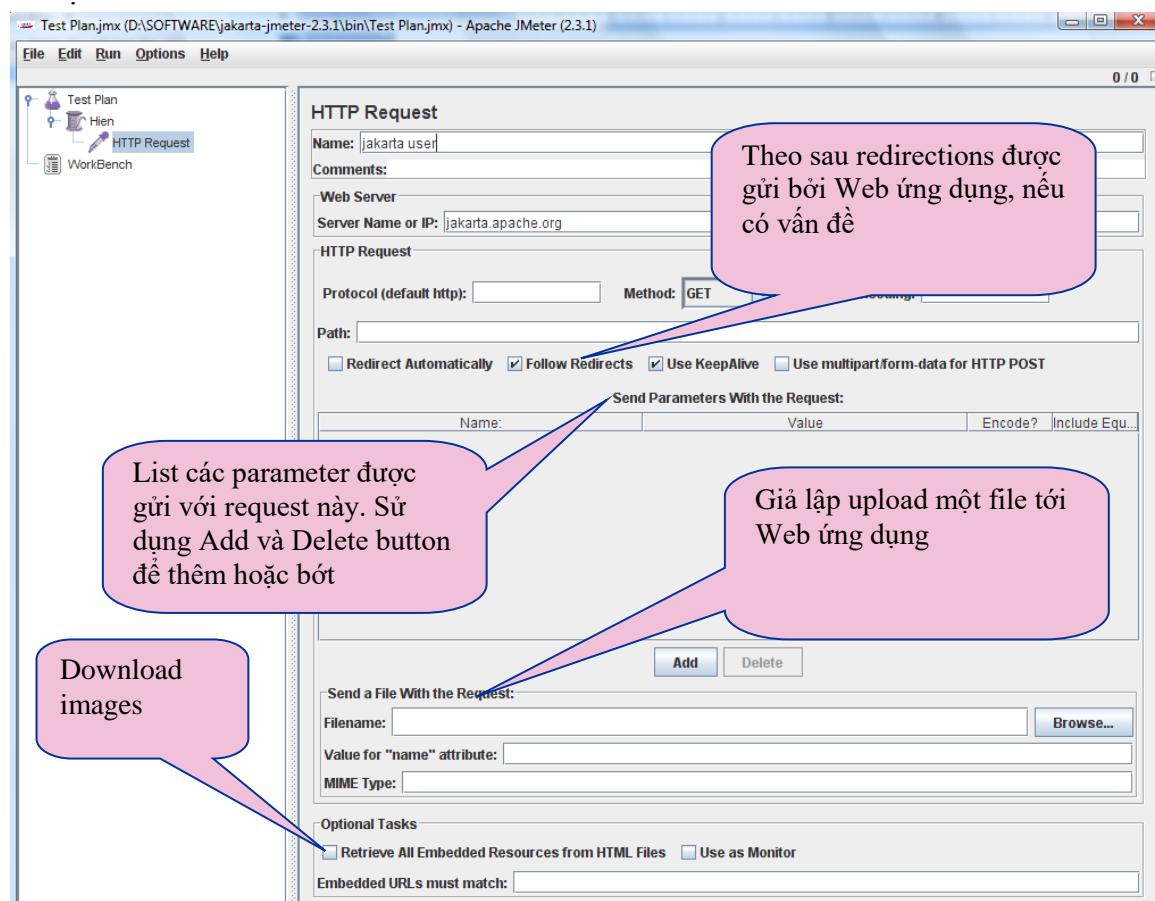
- Vói Samplers, có nhiệm vụ yêu cầu JMeter gửi các requests tới một server. (VD, muốn Jmeter gửi 1 HTTP request, thì tạo một HTTP Request Sampler)
- Vói Logical Controller, cho phép customize trình tự logic Jmeter sử dụng khi gửi các request. Ví dụ, có thể thêm một Interleave Logic Controller giữa 2 HTTP Request Samplers, hoặc có thể sử dụng các Random Controllers để gửi các HTTP requests đến server 1 cách ngẫu nhiên.

2.3.2.1 Samplers

- Jmeter samplers cho phép định nghĩa các request có thể được gửi tới một server. Sampler có thể giả lập các request của người dùng tới target server.
- Mỗi Sampler sinh ra các mẫu kết quả (sample result), với rất nhiều các thuộc tính như hiệu năng, elapsed time, throughput, ...
- Mặc định, Jmeter gửi các request theo thứ tự mà các Samplers xuất hiện trên cây. Tuy nhiên, trật tự xử lý các Sampler có thể được cấu hình mở rộng thêm với các Logic Controller. Các controllers có thể được sử dụng để chỉnh sửa số lần lặp của một sampler
- Các JMeter samplers bao gồm:
 - HTTP Request
 - FTP Request
 - JDBC Request
 - Java Request
 - SOAP/XML-RPC Request
 - WebService (SOAP) Request
 - LDAP Request
 - LDAP Extended Request
 - Access Log Sampler
 - BeanShell Sampler

- BSF Sampler
- TCP Sampler
- JMS Publisher
- JMS Subscriber
- JMS Point-to-Point
- JUnit Request
- Mail Reader Sampler
- Test Action
- Có thể customize mỗi sampler bằng cách thiết lập các thuộc tính của nó, hoặc thêm các Configuration Element.

Ví dụ:



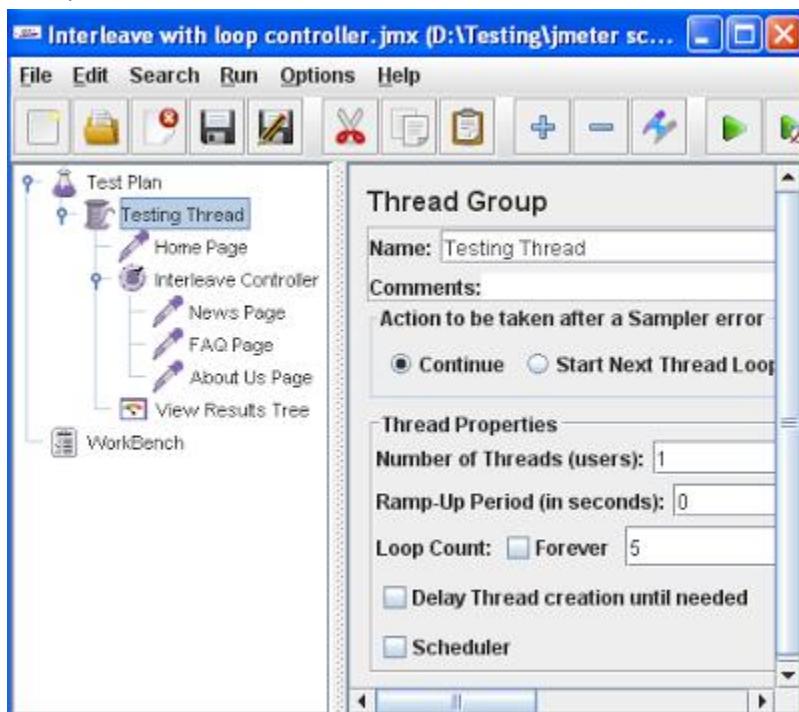
2.3.2.2 Logic Controllers

- Logic Controller cho phép định nghĩa thứ tự xử lý các Samplers trong 1 Thread, ví dụ customize logic mà Jmeter sử dụng để gửi các request (logic: for, if, switch,) Một Logic Controller thay đổi trật tự các request của các phần tử con của nó. Các phần tử con của 1 Logic Controller có thể bao gồm các Samplers, các Configuration Elements, và nhiều loại Logic controllers khác. Với những request này, Jmeter có thể select một cách ngẫu nhiên (Sử dụng Random Controller), repeat (Sử dụng Loop Controller).
- Danh sách các Logic Controller mà Jmeter cung cấp:
 - Simple Controller
 - Loop Controller
 - Once Only Controller
 - Interleave Controller

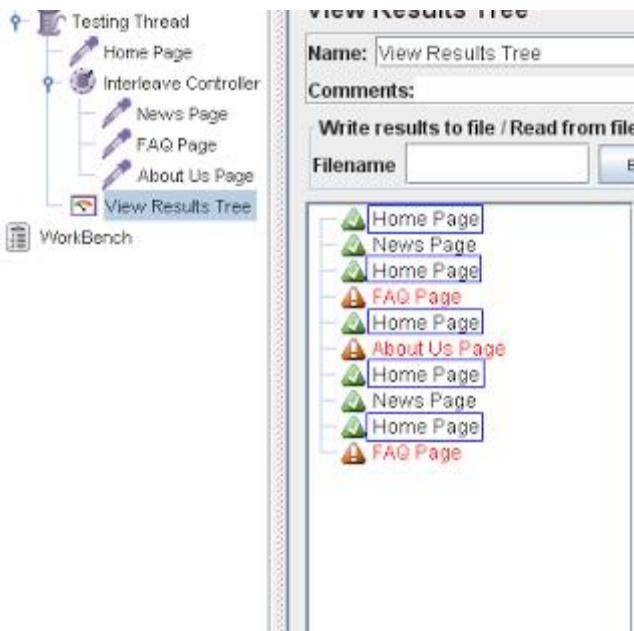
- Random Controller
- Random Order Controller
- Throughput Controller
- Runtime Controller
- If Controller
- While Controller
- Switch Controller
- ForEach Controller
- Module Controller
- Include Controller
- Transaction Controller
- Recording Controller

2.3.2.2.1 Interleave controller

- Add -> Logic Controller -> Select 'Interleave Controller'
- Nếu: 1 ThreadGroup có 5 user, thực hiện 2 lần lặp. Trong Thread Group có interleave controller, với 2 element con của nó là A, hoặc B.
- Thị: Với mỗi user, các phạm vi ngoài interleave controller được thực hiện bình thường, riêng interleave controller, với mỗi user, trong mỗi lần lặp, chỉ thực hiện 1 element con của nó (A hoặc B), lần thực hiện kế tiếp sẽ thực hiện element con tiếp theo của interleave controller, theo thứ tự (VD: Lần 1, thực hiện A, lần 2 thực hiện B, lần 3 lại thực hiện A, lần 4 thực hiện B, ...)
- Ví dụ:

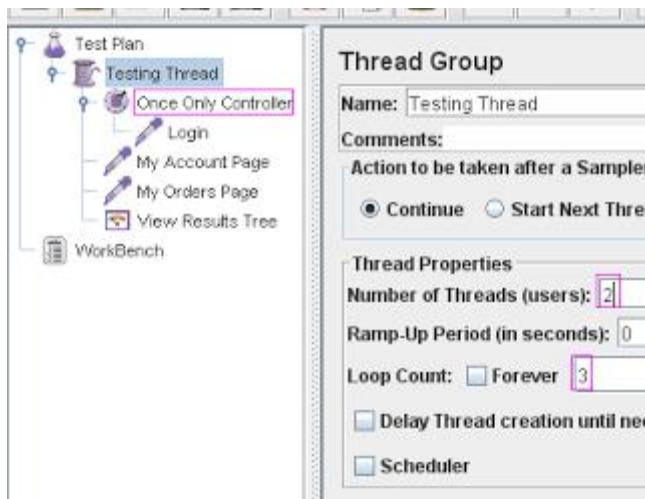


- Kết quả chạy:

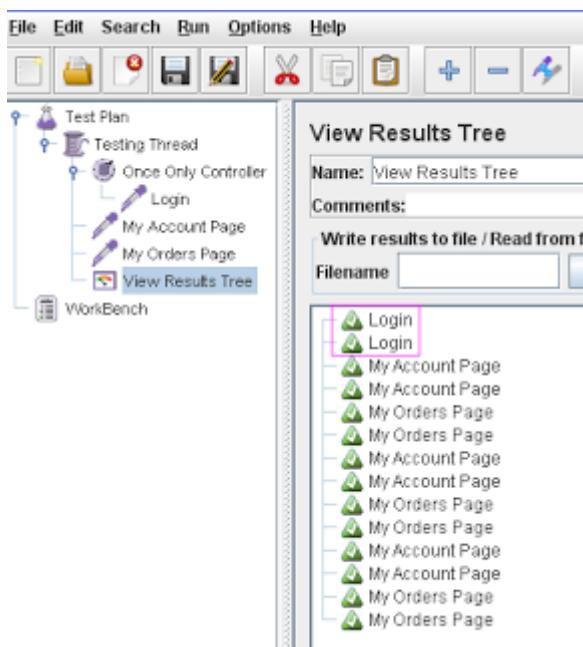


2.3.2.2.2 Once Only Controller

- Lấy một request add vào trong once only controller → request này chỉ được chạy 1 lần duy nhất.
- Ví dụ

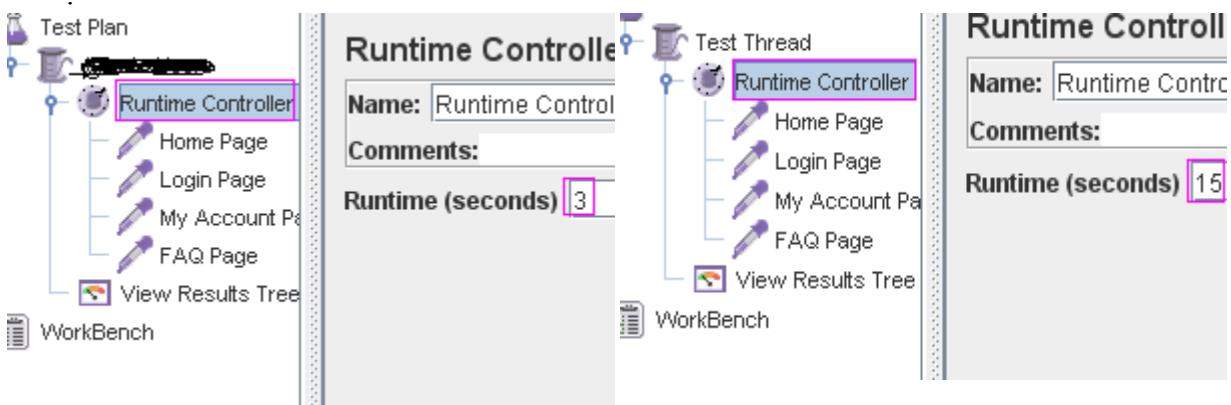


- Kết quả

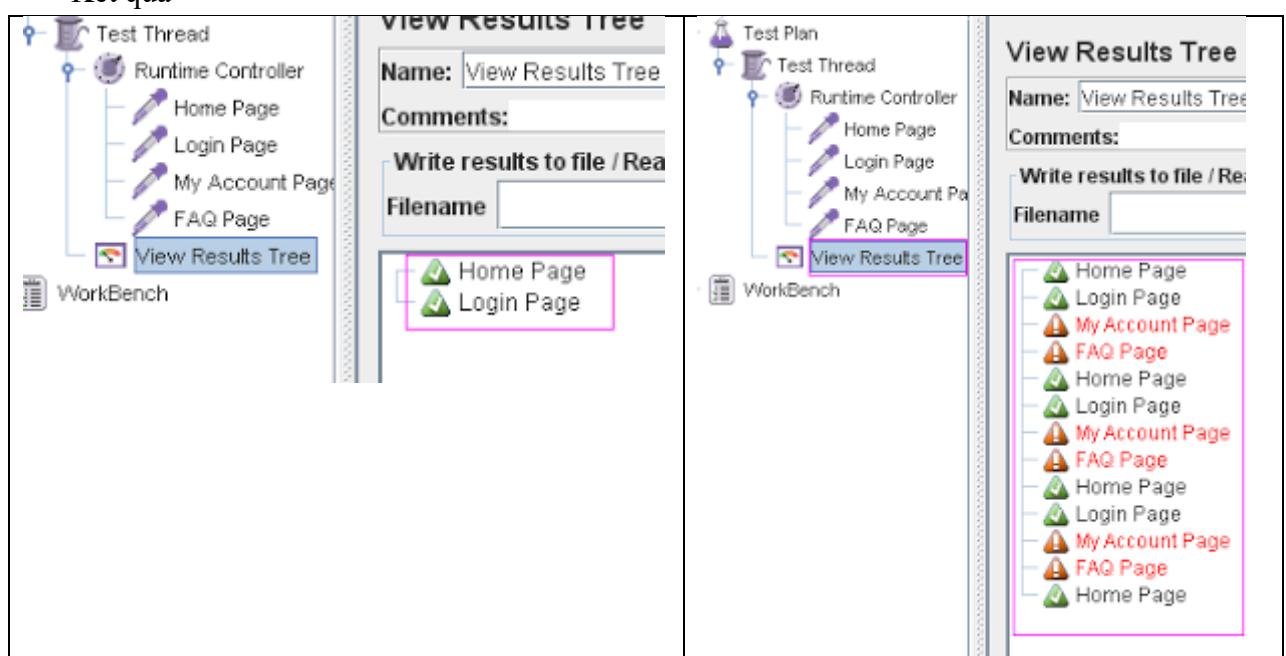


2.3.2.2.3 Runtime Controller

- Thời gian thiết lập để các request chạy, khi hết khoảng thời gian này → test plan dừng lại
- Ví dụ

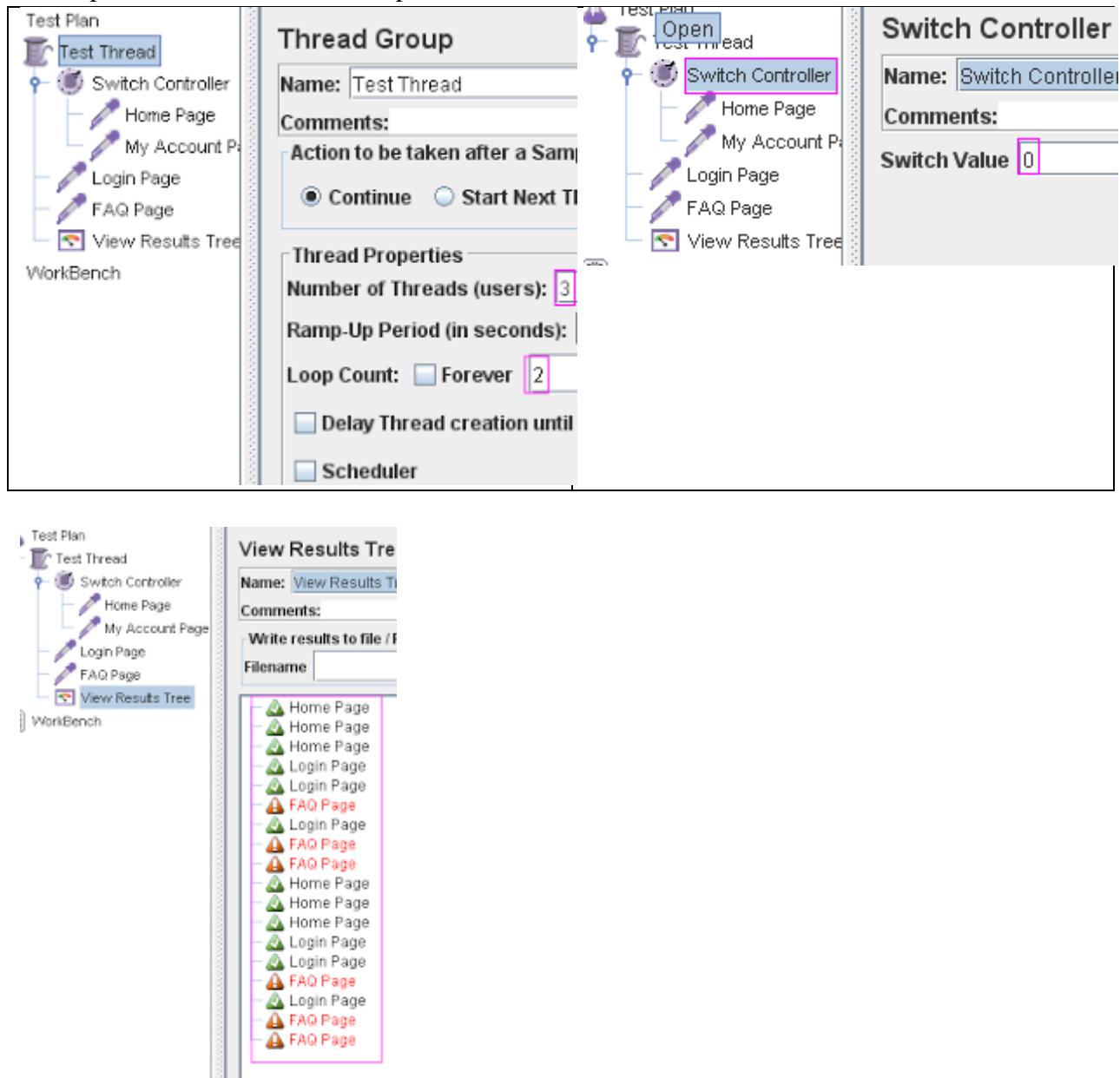


- Kết quả



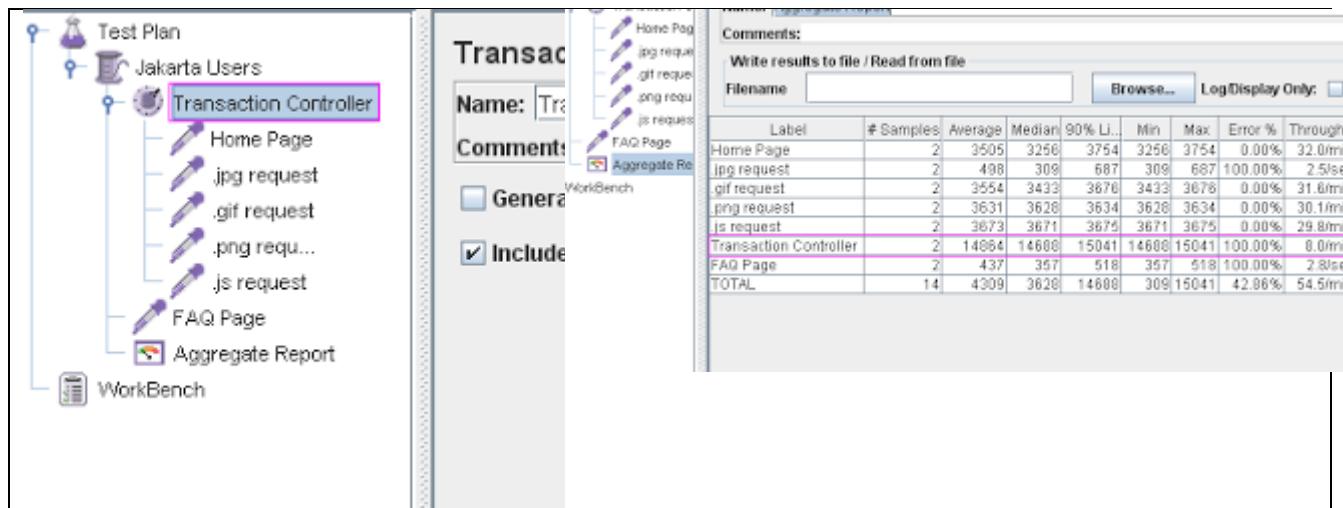
2.3.2.2.4 Switch controller

- Tương tự như interleave Controller, nhưng thay vì lần lượt thực hiện các phần tử con của nó, nó thực hiện 1 phần tử theo giá trị switch
- VD: Trong phần tử switch, có các con là A0, A1, A2, ...An. Giá trị switch sẽ lấy phần tử tương ứng trong mảng con (đánh số từ 0)
- Giá trị switch được chọn có thể là 1 số, cũng có thể là 1 biến, khi đó, giá trị của biến sẽ đưa ra phần tử được chọn thích hợp.



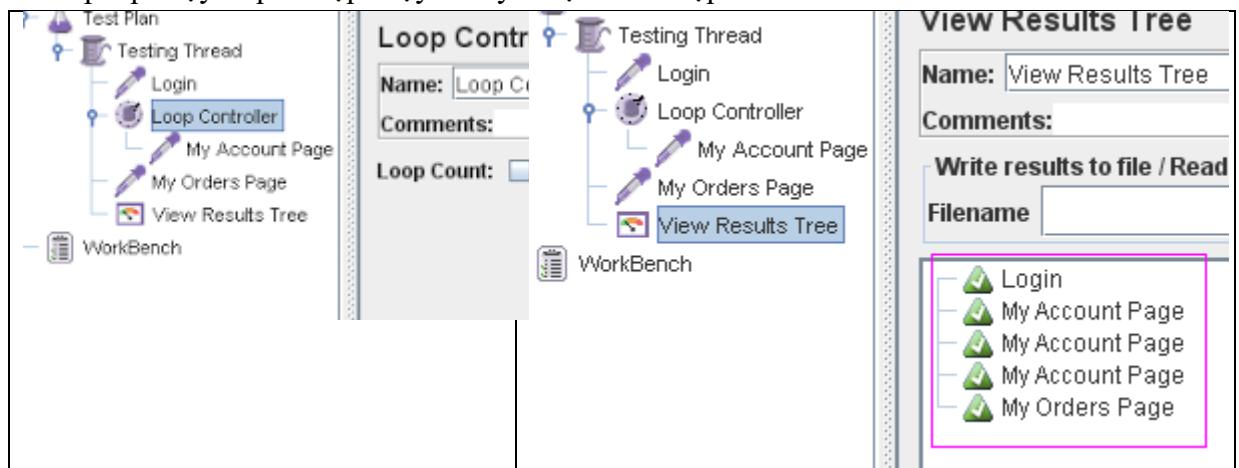
2.3.2.2.5 Transaction Controller

- Transaction Controller cho phép tạo ra các sampler bổ sung, các sampler này sẽ đo thời gian tổng thực hiện để test lồng các yếu tố với nhau.
- Có 2 option:
 - + Additional sample is added after the nested samples: sample bổ sung được thêm vào sau khi các sample lồng nhau.
 - + Additional sample is added as a parent of the nested samples: sample bổ sung được thêm vào là sample cha của các sample lồng nhau.



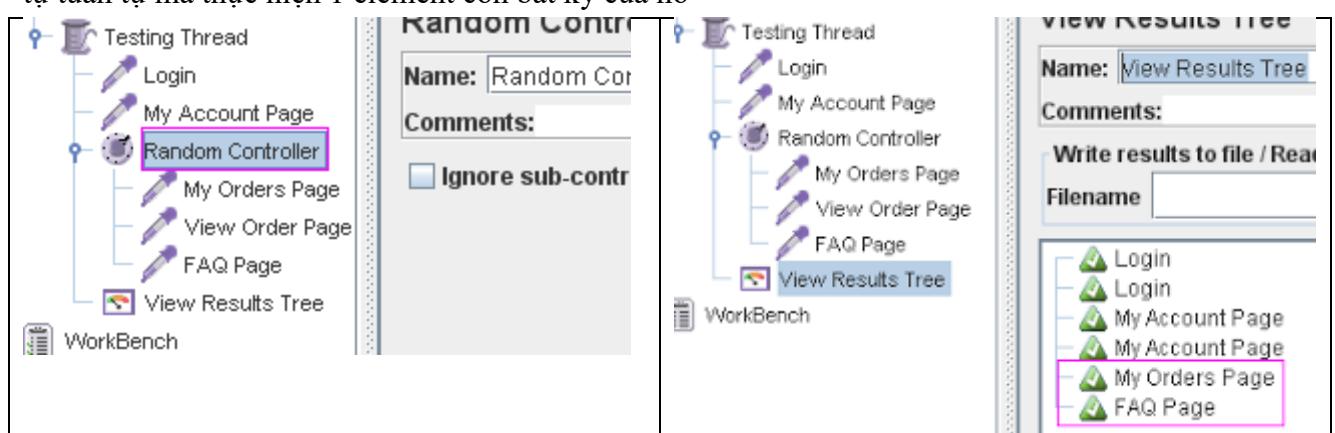
2.3.2.2.6 Loop Controller

- Cho phép chạy request lặp chạy đi tùy thuộc số lần lặp



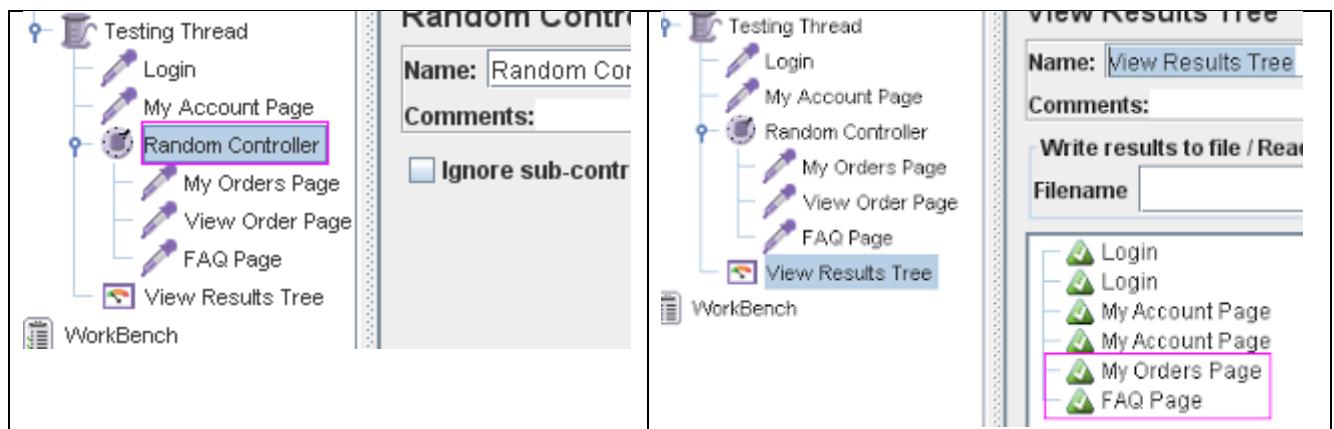
2.3.2.2.7 Random controller

Thực hiện tương tự như Interleave controller, điểm khác biệt nằm ở chỗ nó không thực hiện theo thứ tự tuần tự mà thực hiện 1 element con bất kỳ của nó



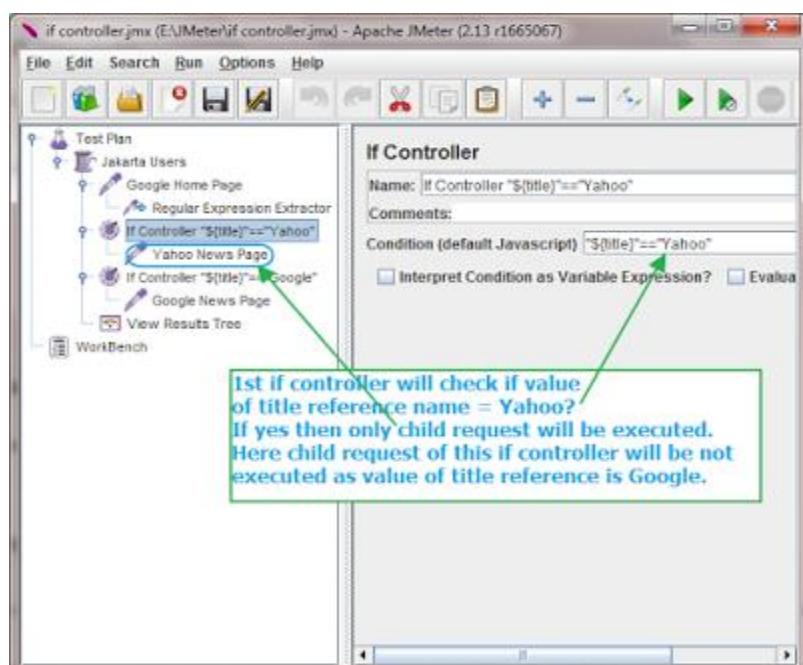
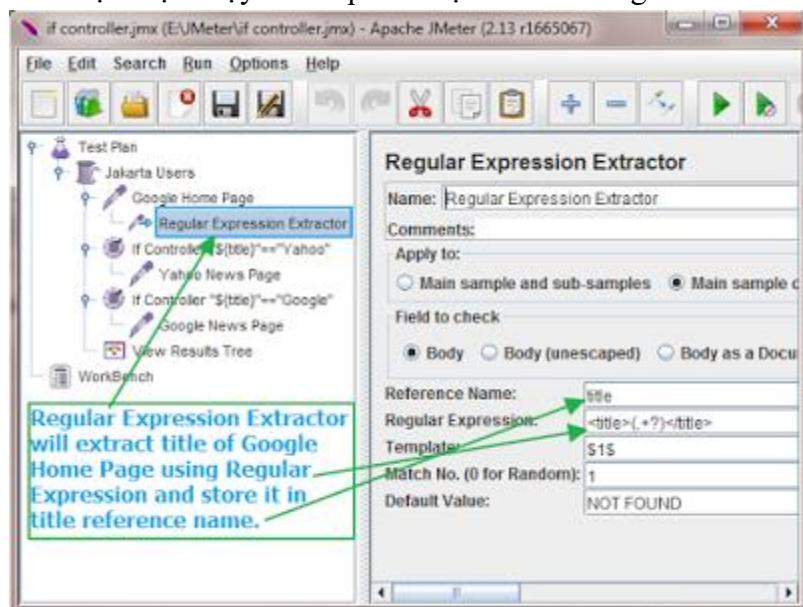
2.3.2.2.8 Random order controller

Thực hiện tương tự như Simpler Controller, điểm khác biệt nằm ở chỗ nó sẽ thực hiện mỗi phần tử con 1 lần, nhưng thứ tự thực hiện các phần tử con là random

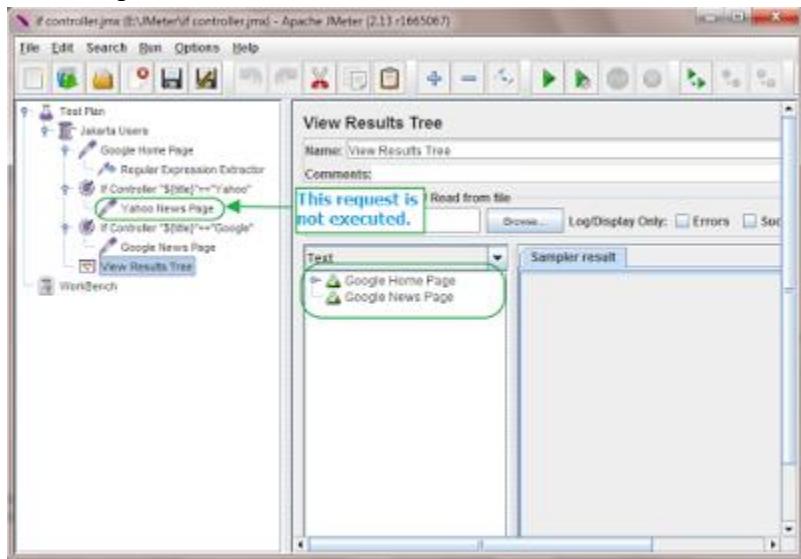


2.3.2.2.9 If Controller

- Thực hiện chạy các request được nhóm chung vào if controller khi điều kiện thỏa mãn



- Kết quả



2.3.3 Listener

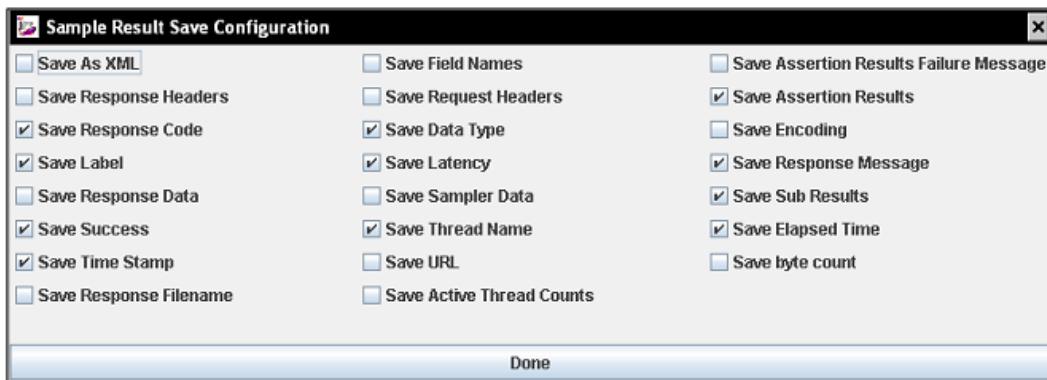
- Được sử dụng để xử lý sau khi request data. Ví dụ, có thể lưu dữ liệu vào 1 file hoặc hiển thị kết quả dạng chart. Jmeter chart không cung cấp các option cấu hình, tuy nhiên nó được thiết kế mở, cho phép thêm một visualization, hoặc thêm các module xử lý dữ liệu
- Listeners cho phép người dùng view kết quả của các Samplers. Kết quả có thể được biểu diễn dưới dạng các tables, các graphs, các trees hay text thông thường trong các log files.
- Mỗi Listener hiển thị các thông tin response theo các cách thức khác nhau.
- Có rất nhiều dạng Listener được JMeter cung cấp, có thể kể đến một số Listener thường được sử dụng để cung cấp như:
 - Graph Full Results: Cung cấp tất cả những kết quả trả về dưới dạng đồ thị : Lỗi, thời gian phản hồi, lưu lượng ...
 - View Results in Table: Hiển thị những thông số về thời gian phản hồi của từng yêu cầu, những yêu cầu thực hiện thành công và thất bại... dưới dạng bảng trong suốt quá trình thực thi thử nghiệm.
 - Summary Report : Cung cấp những thống kê tổng thể.

Note:

- Jmeter Listener cho phép view, save và read các test result. Tất cả các Listener đều lưu dữ liệu vào cùng 1 file (*.jtl), Điểm khác nhau duy nhất nằm ở cách biểu diễn kết quả được ghi lại
- Một Listener có thể sử dụng rất nhiều memory nếu nó đảm nhận ghi dữ liệu cho nhiều Sample.
- Jmeter có thể chạy chậm, nếu như có quá nhiều listener active. Vì vậy, cần chú ý sử dụng một lượng nhỏ hợp lý các listeners.

Một số đặc tính chung

- Configure button: Sử dụng Configure button để chọn các thông tin sẽ được lưu ra file hoặc được sử dụng về sau (các file *.jtl), theo định dạng XML hoặc CSV. Đây chính là cấu hình cho phép customize các kết quả ra.



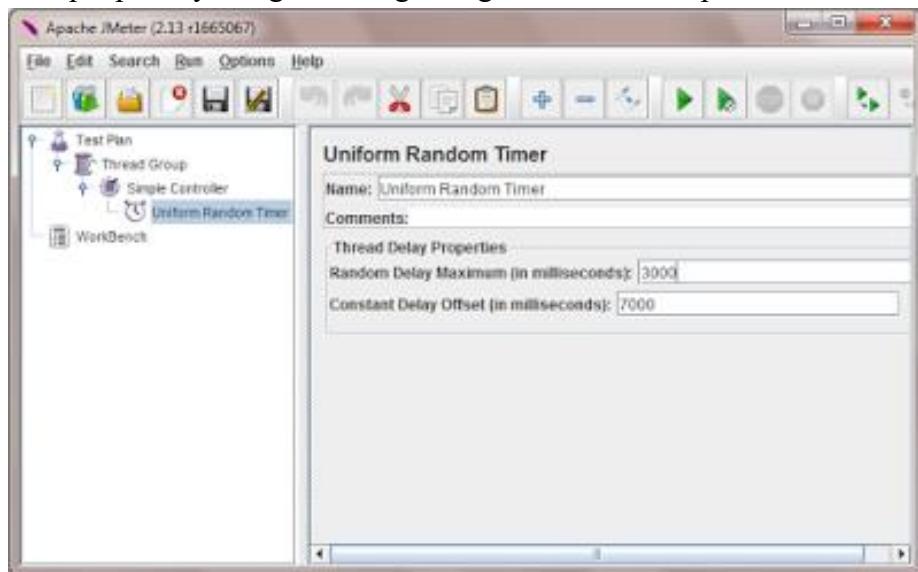
- Browser button: Cho phép đọc, sau đó hiển thị 1 kết quả bất kỳ được lưu từ trước.

2.3.4 Timers

- Timer là một phần rất quan trọng khi xây dựng một Test Plan, nó cho phép cài đặt khoảng thời gian giữa 2 yêu cầu kế tiếp nhau mà người dùng ảo gửi đến máy chủ. Điều này sẽ tạo ra một mô phỏng thực tế nhất so với hoạt động thực tế của người dùng trên website.
- Theo mặc định, Jmeter gửi các request ngay liền nhau, điều này có thể làm server quá tải. Vì vậy, thêm 1 Timer cho phép giảm khả năng break down cho server.

2.3.4.1 Uniform Random Timer

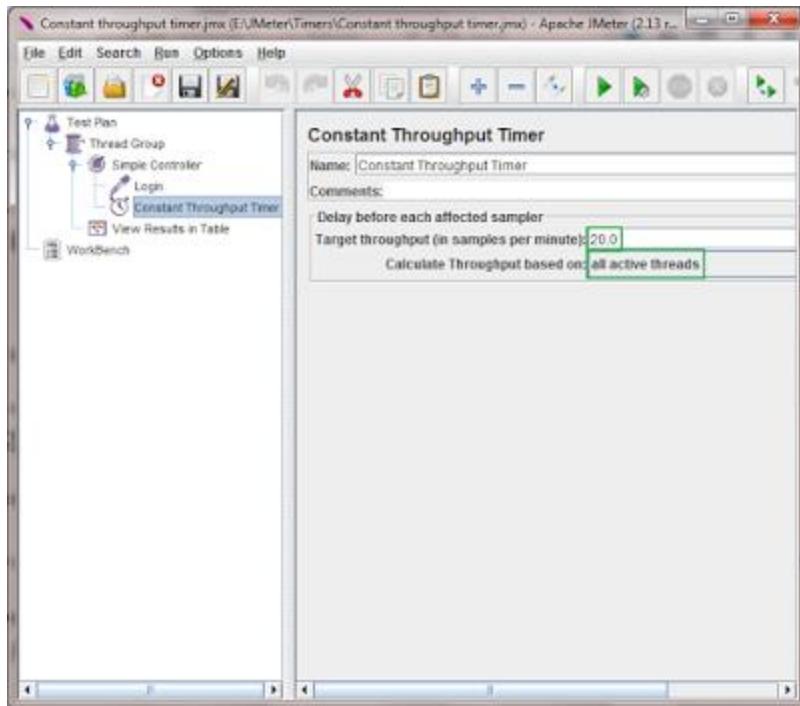
Cho phép delay trong 1 khoảng thời gian được thiết lập



- Các tham số:
 - o **Random Delay Maximum** : Thời gian delay tối đa cho phép
 - o **Constant Delay Offset** : Thời gian delay được đặt cố định.
- Như ví dụ trên, thời gian delay giữa 2 request là từ 7 giây đến 10 giây.

2.3.4.2 Constant Throughput Timer

- Add -> Select Timer -> Constant Throughput Timer
- Cho phép quy định số lượng sample chạy trong 1 phút

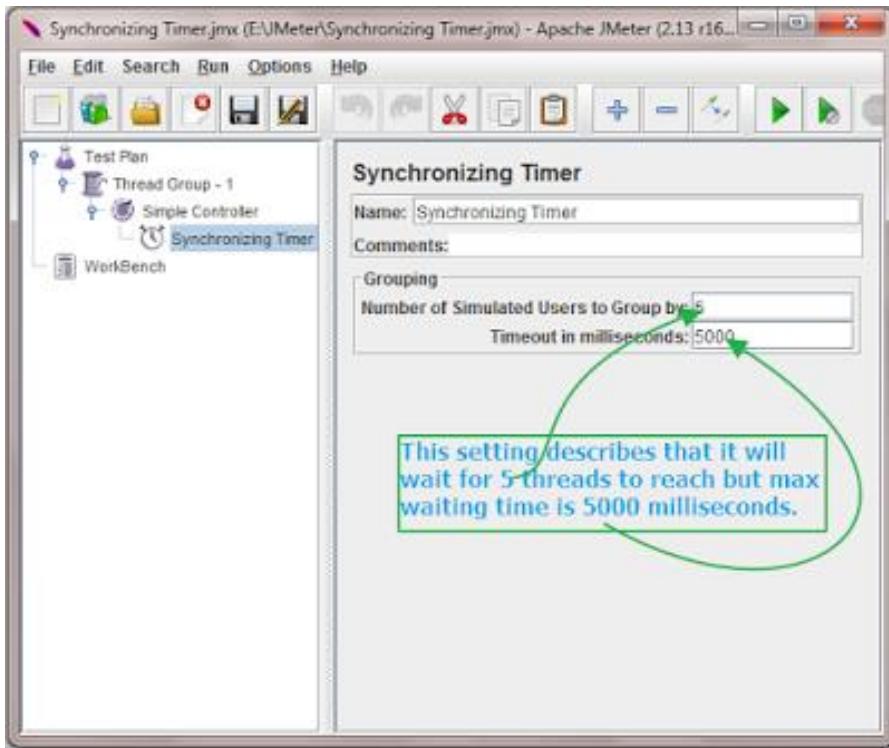


Kết quả:

Sample #	Start Time	Thread Name	Label	Sample Time(ms)
1	12:08:13.708	Thread Group 1-1	Login	300
2	12:08:16.693	Thread Group 1-1	Login	369
3	12:08:19.694	Thread Group 1-1	Login	354
4	12:08:22.693	Thread Group 1-1	Login	354
5	12:08:25.692	Thread Group 1-1	Login	351
6	12:08:28.693	Thread Group 1-1	Login	375
7	12:08:31.693	Thread Group 1-1	Login	356
8	12:08:34.693	Thread Group 1-1	Login	358
9	12:08:37.693	Thread Group 1-1	Login	355
10	12:08:40.693	Thread Group 1-1	Login	342
11	12:08:43.693	Thread Group 1-1	Login	359
12	12:08:46.704	Thread Group 1-1	Login	574
13	12:08:49.693	Thread Group 1-1	Login	359
14	12:08:52.692	Thread Group 1-1	Login	356
15	12:08:55.693	Thread Group 1-1	Login	338
16	12:08:58.694	Thread Group 1-1	Login	2086
17	12:09:01.694	Thread Group 1-1	Login	397
18	12:09:04.692	Thread Group 1-1	Login	369
19	12:09:07.692	Thread Group 1-1	Login	337
20	12:09:10.705	Thread Group 1-1	Login	348
21	12:09:13.694	Thread Group 1-1	Login	383

2.3.4.3 Synchronizing Timer

- Add -> Timer -> Synchronizing Timer
- Cho phép gom nhóm số lượng threads muốn đẩy tải tại 1 thời điểm



- Các tham số
 - o **Number of Simulated Users to Group by:** Số lượng threads muốn đầy tải tại 1 thời điểm trong 1 lần.
 - o **Timeout in milliseconds :** Thời gian timeout tối đa để lấy các thread. Nếu sau thời gian này các thread chưa lấy thành công ➔ sẽ bị dừng tất cả các threads.

2.3.5 Configuration Elements

- Configuration Element cho phép người dùng tạo các giá trị mặc định và các biến được sử dụng trong các Samplers. Chúng được sử dụng để add hoặc modify các request tạo ra từ các Sampler.

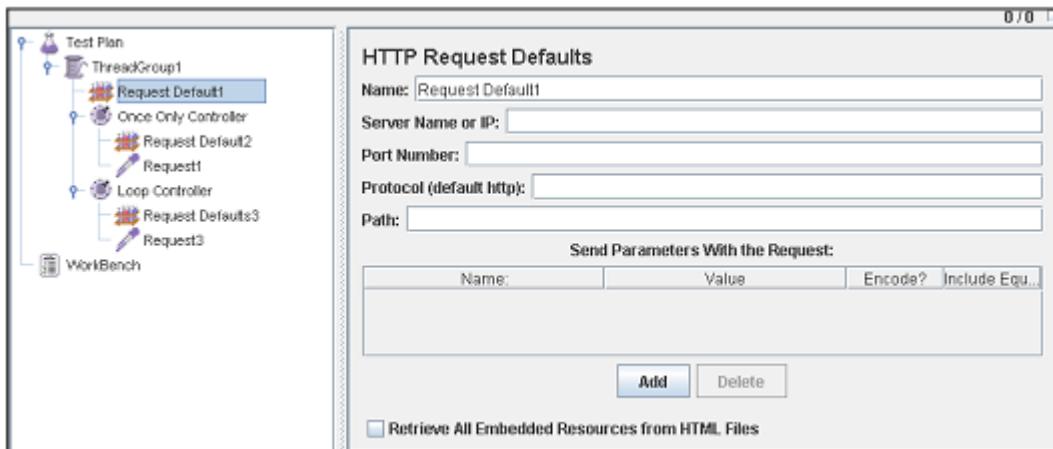
Trong phạm vi của nó, 1 Configuration Element được thực hiện đầu tiên, trước tất cả các Sampler trong cùng phạm vi đó. Vì vậy, 1 Configuration Element chỉ được truy cập trong nội bộ nhánh mà nó được đặt.

Jmeter cung cấp các loại Configuration Elements JMeter như sau:

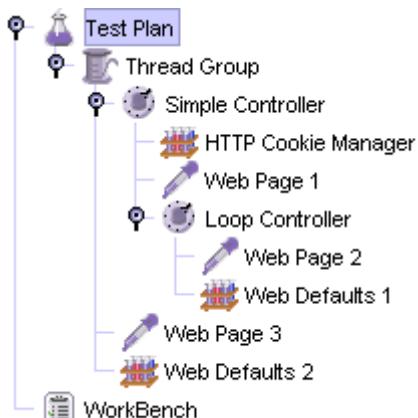
- **CSV Data Set Config**
- **FTP Request Defaults**
- **HTTP Authorization Manager**
- **HTTP Cookie Manager**
- **HTTP Proxy Server**
- **HTTP Request Defaults**
- **HTTP Header Manager**
- **Java Request Defaults**
- **JDBC Connection Configuration**
- **Login Config Element**
- **LDAP Request Defaults**
- **LDAP Extended Request Defaults**
- **TCP Sampler Config**

- User Defined Variables
- Simple Config Element

Ví dụ:



- Trong Testplan trên, Request Default1 có thể được truy cập bởi các Request1 và Request3 Sampler, còn Request Default2 chỉ cho phép truy cập bởi Request1, và Request Default3 chỉ cho phép truy cập bởi Request3.
- Một Configuration Element trong 1 nhánh cây có quyền ưu tiên hơn so với element cùng loại trong nhánh “parent”. Ví dụ, có 2 HTTP Request Defaults elements, là "Web Defaults 1" và "Web Defaults 2". Vì "Web Defaults 1" nằm trong Loop Controller, nên chỉ có "Web Page 2" có thể truy cập tới nó. Các HTTP requests khác sẽ sử dụng "Web Defaults 2", vì nó được đặt trong Thread Group.



Note:

- Với phần tử cấu hình User Defined Variables, được xử lý ở điểm khởi đầu của quá trình test, mà không cần quan tâm nó được đặt ở đâu. Tuy nhiên, để đơn giản và tiện theo dõi, nên đặt phần tử này ở điểm bắt đầu của một Thread Group.
- Trong cùng 1 phạm vi, dù đặt trước hay sau, 1 CSV Data Set Config luôn được ưu tiên hơn so với User Defined Variables (1 biến đọc từ file luôn override giá trị 1 biến đặt từ user)

2.3.5.1 HTTP Cache Manager

- Được sử dụng để thêm chức năng bộ nhớ đệm cho các HTTP Request trong phạm vi của nó nhằm mô phỏng tính năng bộ nhớ đệm của trình duyệt.
- Mỗi một giả lập người dùng ảo có bộ nhớ cache riêng nó. Mặc định quản lý cache sẽ lưu trữ 5000 mục trong bộ nhớ đệm trên mỗi một User ảo, sử dụng thuật toán LRU

HTTP Cache Manager

Name:	HTTP Cache Manager
Comments:	
<input type="checkbox"/> Clear cache each iteration?	
<input type="checkbox"/> Use Cache-Control/Expires header when processing GET requests	
Max Number of elements in cache	5000

- Clear cache each iteration: Tích chọn để xóa cache trước mỗi vòng lặp
- Use Cache Control/Expires header when processing GET requests: Tích chọn để kiểm tra lại giá trị Cache Control/Expires tại thời điểm hiện tại.
- Max Number of elements in cache: Số lượng phần tử tối đa có thể lưu trữ trong bộ nhớ cache

2.3.5.2 HTTP Cookie Manager

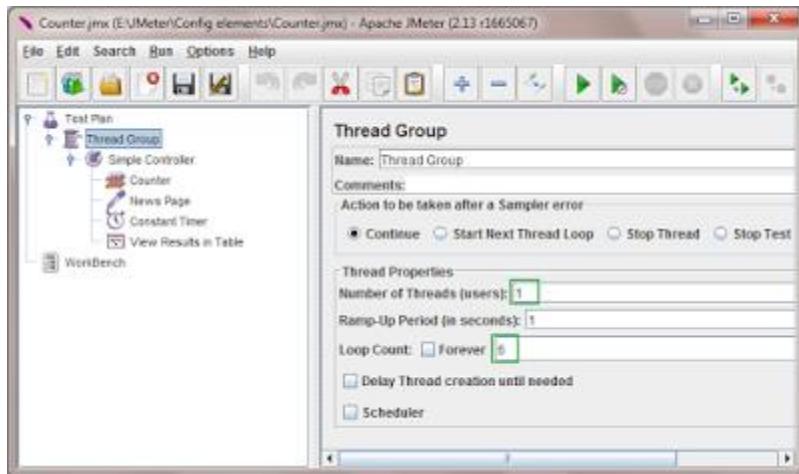
Phần tử Cookie Manager có 2 chức năng:

- Thứ nhất: nó lưu trữ và gửi cookie giống như 1 trình duyệt web. Nếu bạn có một HTTP Request và kết quả trả về có chứa cookie, Cookie Manager sẽ tự động lưu trữ cookie và sẽ sử dụng nó cho tất cả các request tiếp theo trong trang web cụ thể. Mỗi một thread giả lập có khu vực lưu trữ cookie riêng của mình, vì vậy nếu bạn đang kiểm tra 1 trang web có sử dụng cookie để lưu trữ thông tin về phiên, mỗi một thread giải lập sẽ có phiên làm việc riêng của mình.
- Thứ 2: Bạn có thể thêm mới 1 cookie thủ công cho Cookie Manager. Nếu bạn làm điều đó thì cookie sẽ được chia sẻ cho tất cả các thread giả lập

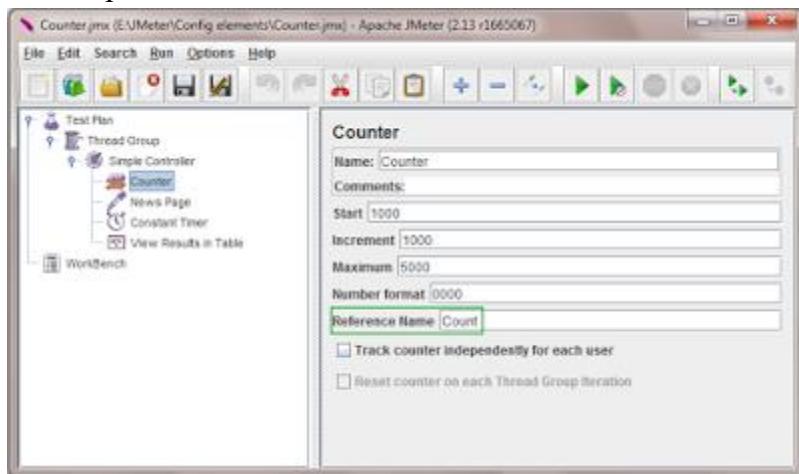
2.3.5.3 Counter config element

- Cho phép thiết lập bộ đếm và có thể gọi nó tại bất kì đâu trong test plan
- Add -> Config Element ->Counter
- Các tham số:
 - Start: là số khởi điểm cho bộ đếm được dùng trong suốt lần lặp đầu tiên.
 - Increment: Lượng thời gian muôn tang cho bộ đếm cho các lần lặp tiếp theo.
 - Maximum: Số lượng tăng lên tối đa cho bộ đếm. Khi đạt tới max nó sẽ đặt lại từ đầu.
 - Format : Định dạng cho các thread chạy, ví dụ: đặt là 0000 thì thread sẽ định dạng là 0001, 0002...
 - Reference Name : Tên dùng để tham chiếu khi các đối tượng khác gọi. Khi elements khác gọi bộ counter sử dụng \$counter_name.
 - Track Counter Independently for each User : Nếu tích vào checkbox này thì mỗi thread sử dụng bộ đếm của riêng nó trong suốt các lần lặp.
 - Reset counter on each Thread Group Iteration : Reset counter sau khi kết thúc việc lặp.
- Ví dụ:

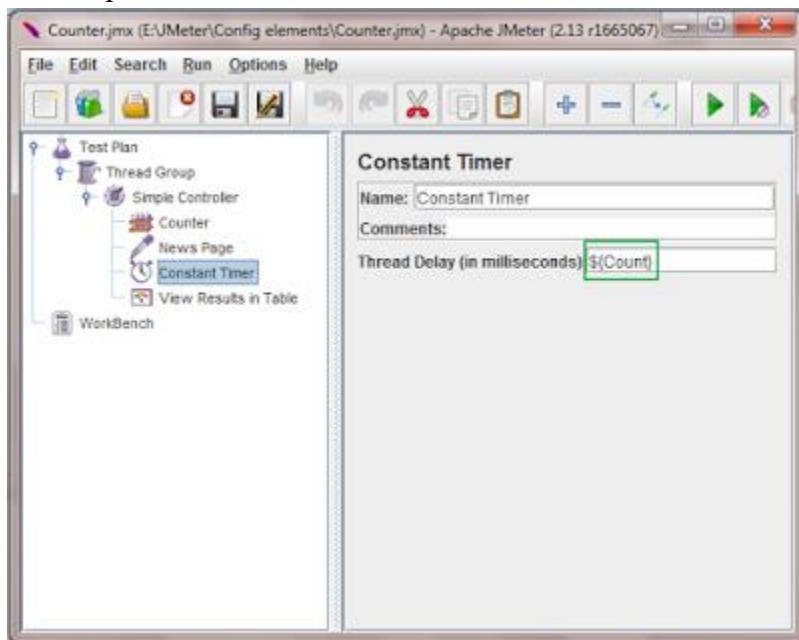
Thiết lập các tham số cho Thread group



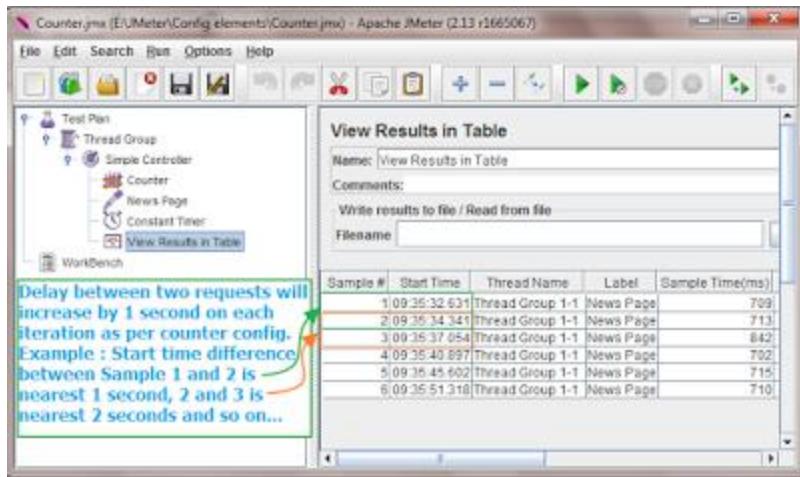
Thiết lập các tham số cho Counter



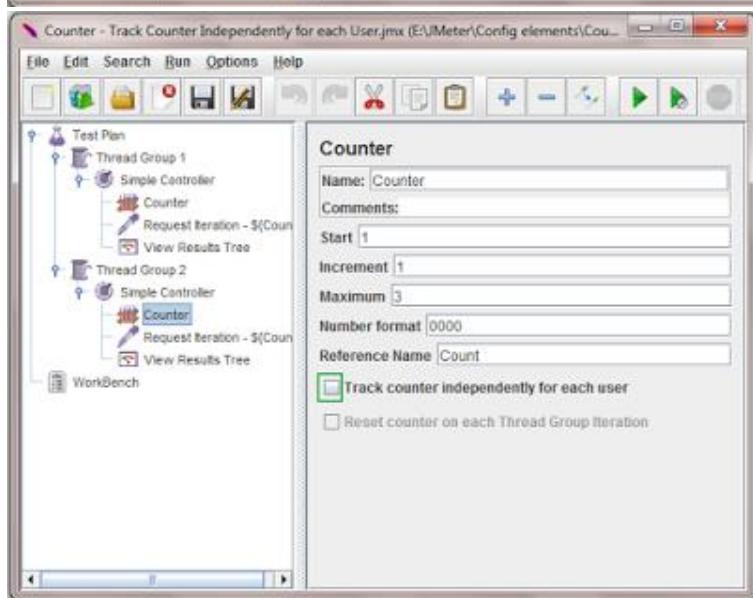
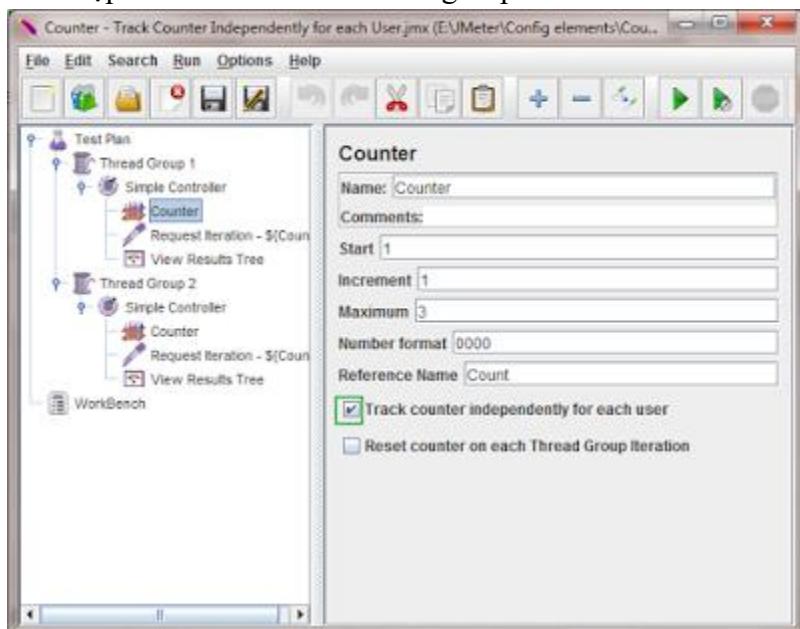
Thiết lập các tham số cho Constant timer



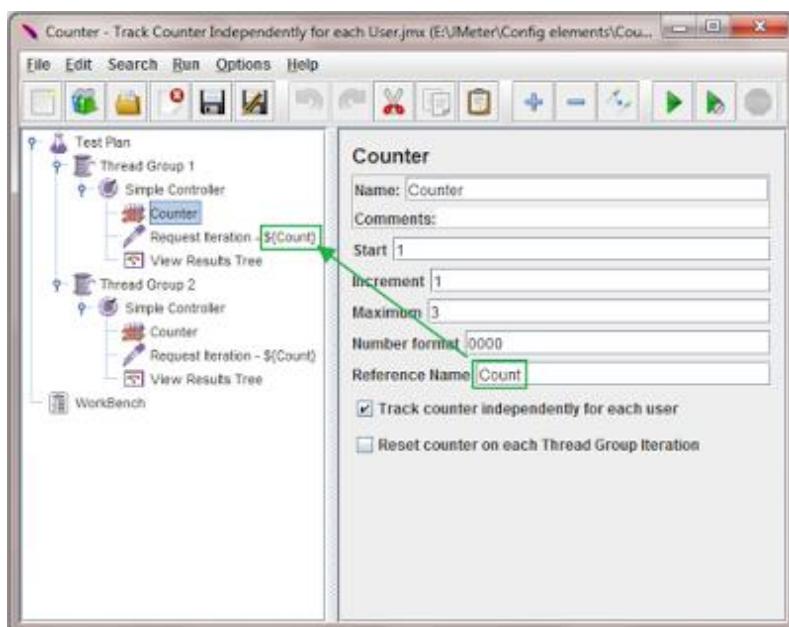
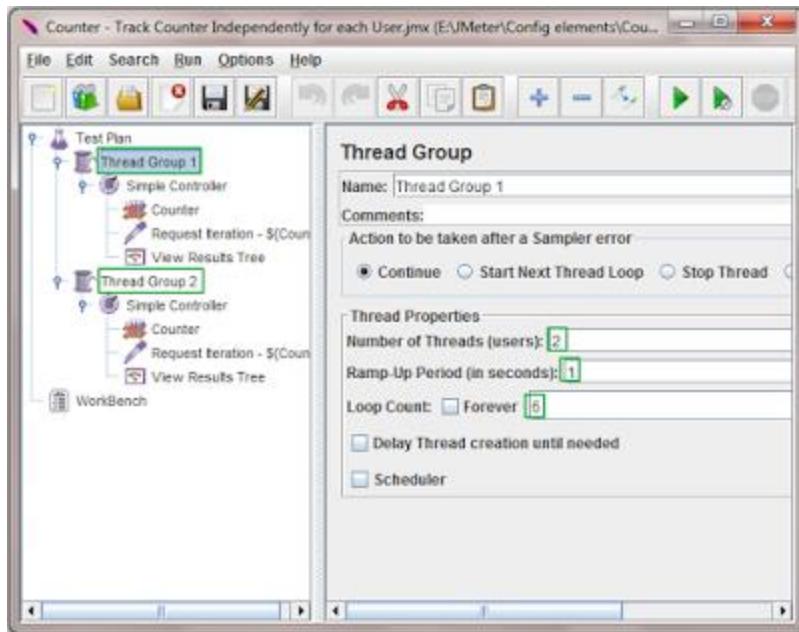
Kết quả:



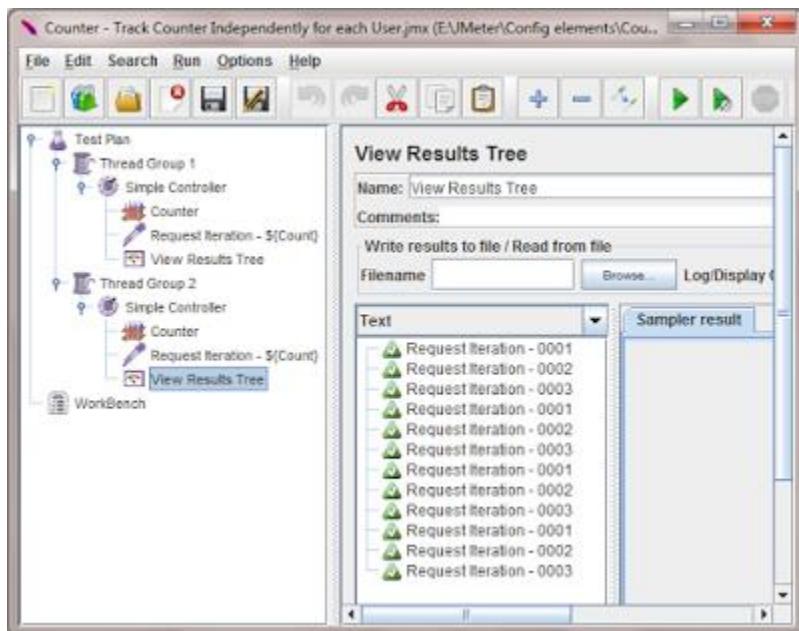
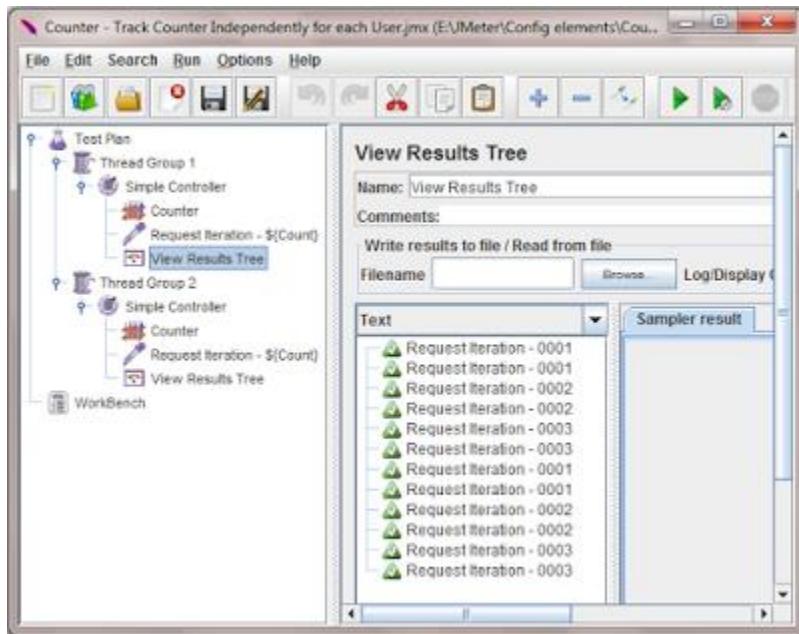
- Ví dụ khi sử dụng track counter
Thiết lập các tham số cho Thread group



Thiết lập các tham số cho counter



Kết quả



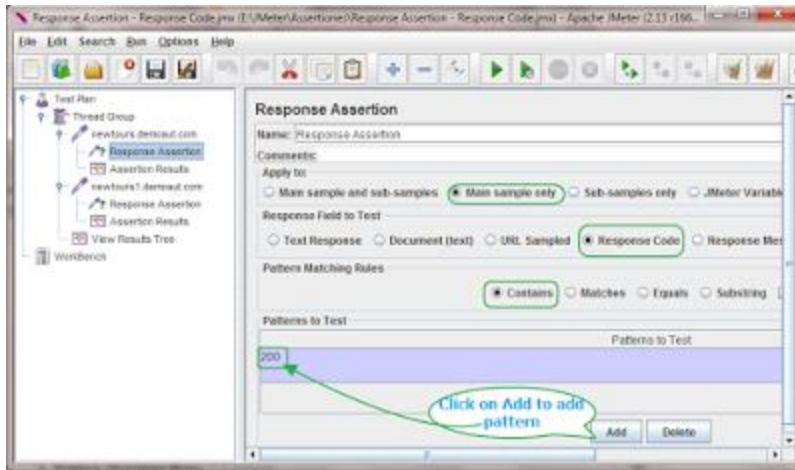
2.3.5.4 Csv data set config

- Cho phép đọc giá trị của biến từ file csv và có thể sử dụng chúng trong samples
- Add -> Config Element -> CSV Data Set Config
- Các tham số:
 - o Filename : Đường dẫn của file data dùng cho test plan. Ví dụ : E:\Test.csv
 - o File Encoding : Khi muốn sử dụng encoding để đọc file.
 - o Variable Names : Tên các biến, phân tách nhau bằng dấu ','. Ví dụ: X,Y,Z
 - o Delimiter : Định nghĩa dấu phân tách sử dụng để đọc file.
 - o Allow quoted data? : Nếu muốn đọc kí tự ("") từ file → tích chọn.
 - o Recycle on EOF? : nếu muốn đọc dữ liệu từ dòng đầu tiên đến dòng cuối cùng. Mặc định là true. Nếu chọn false → khi đọc hết dòng cuối nó sẽ in ra EOF tại vòng lặp tiếp theo. Nếu là true → lần lặp tiếp sẽ quay lại dòng đầu tiên
 - o Stop thread on EOF? : nếu muốn dừng thread khi đặt Recycle on EOF? = false.
 - o Sharing mode : Sử dụng để chia sẻ file giữa tất cả các thread

2.3.6 Assertion

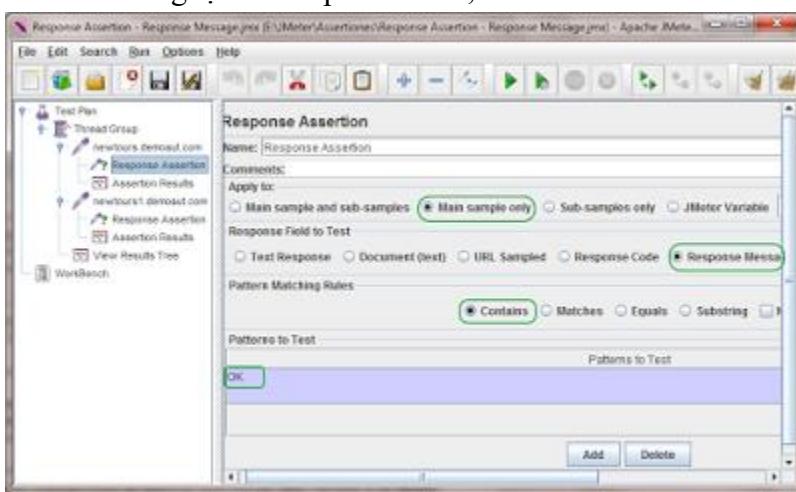
2.3.6.1 Response Assertion To Assert Response Code

- Là element quan trọng trong jmeter, nó giúp kiểm tra mã code trả về trong response của request có giống với mã code mong muốn không.
- Chuột phải vào request -> Add -> Assertions -> Response Assertion



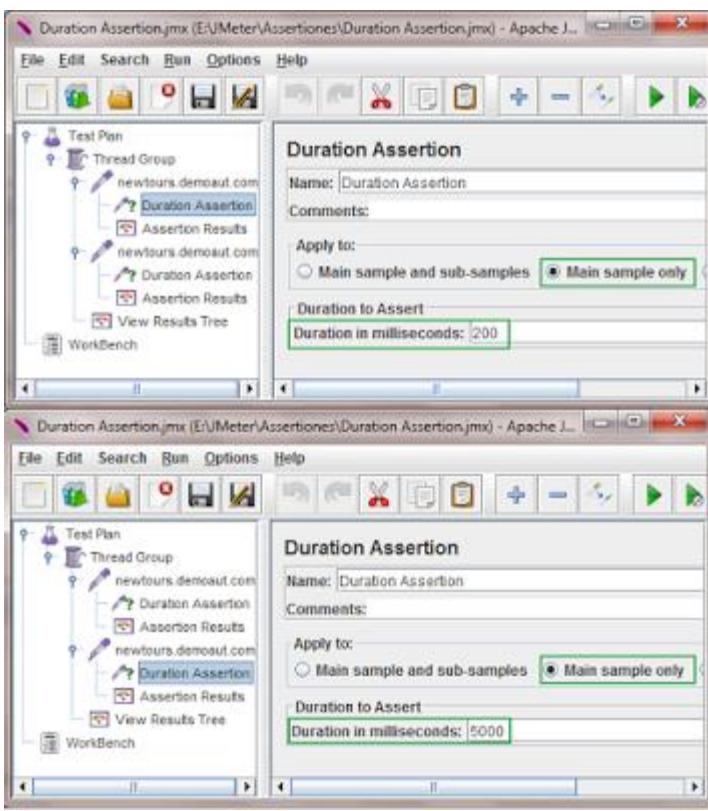
2.3.6.2 Response Assertion To Assert Response Message

- Trong tự như response code, nó kiểm tra text trả về và mong muốn có giống nhau không.



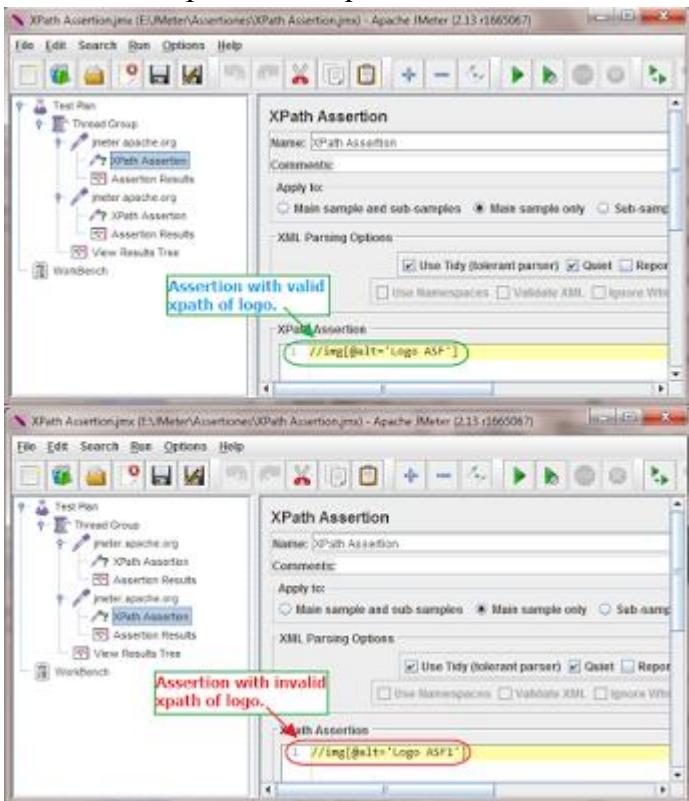
2.3.6.3 Using Duration Assertion

- Chuột phải vào request -> Add -> Assertions -> Duration Assertion



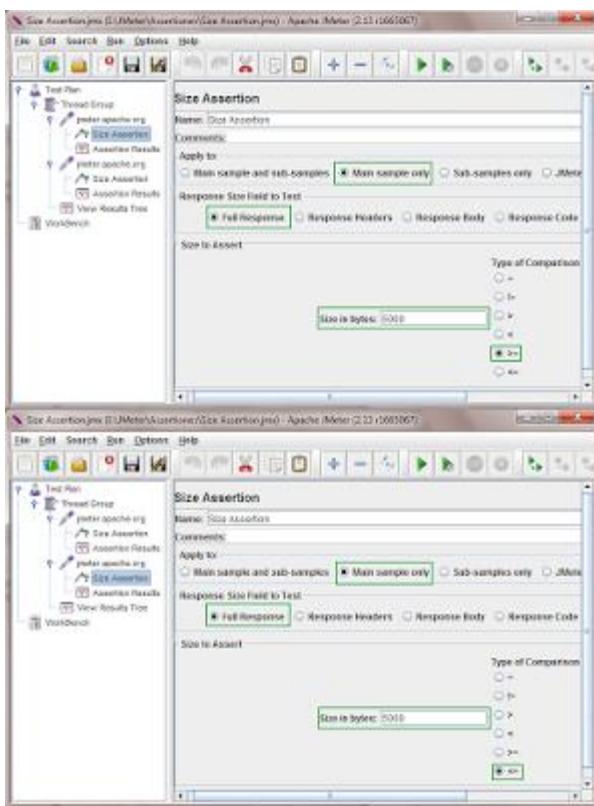
2.3.6.4 XPath Assertion

- Chuột phải vào request -> Add -> Assertions -> Xpath Assertion



2.3.6.5 Size Assertion

- Chuột phải vào request -> Add -> Assertions -> Size Assertion



2.3.7 Pre-Processor Elements

- Pre-processors cho phép chỉnh sửa các Samplers trong phạm vi của nó. Pre-Processor thường được sử dụng để chỉnh sửa thiết lập của một Sample Request trước khi nó được chạy, hoặc update các variables không được extract từ các response text.
- Danh sách các Pre-Processor Elements JMeter cung cấp:
 - o HTML Link Parser
 - o HTTP URL Re-writing Modifer
 - o HTML Parameter Mask
 - o HTTP User Parameter Modifer
 - o User Parameters
 - o Counter
 - o BeanShell PreProcessor

2.3.8 Post-Processor Elements

- Post-processors được thực hiện sau khi một request vừa được tạo ra từ 1 Sampler. Thông thường, Post processor được đặt làm con của một Sampler, để đảm bảo nó được chạy chỉ sau Sampler đó, không liên quan tới các Sampler sau đó. Post Processor Element là đặc biệt hữu dụng để **xử lý các response data**, ví dụ như để thu được các giá trị cụ thể cho các sử dụng về sau
- Danh sách các Post-Processor Elements mà JMeter cung cấp:
 - o Regular Expression Extractor
 - o XPath Extractor
 - o Result Status Action Handler
 - o Save Responses to a file
 - o Generate Summary Results
 - o BeanShell PostProcessor

2.3.9 Thứ tự thực hiện các phần tử của 1 testplan

0. Configuration elements
1. Pre-Processors
2. Timers
3. Sampler
4. Post-Processors (trừ trường hợp SampleResult là null)
5. Assertions(trừ trường hợp SampleResult là null)
6. Listeners(trừ trường hợp SampleResult là null)

Note: Cần chú ý rằng các Timers, Assertions, Pre- và Post-Processors chỉ được xử lý nếu nó được áp dụng cho 1 sampler nào đó. Các Logic Controllers và các Samplers được xử lý theo thứ tự xuất hiện trên cây. Các test elements khác được xử lý theo phạm vi mà chúng được tìm thấy, và theo loại test element. (Trong cùng 1 loại, các elements được xử lý theo thứ tự mà chúng xuất hiện trên cây)

Để rõ hơn, xét ví dụ sau đây:

- Controller
 - Post-Processor 1
 - Sampler 1
 - Sampler 2
 - Timer 1
 - Assertion 1
 - Pre-Processor 1
 - Timer 2
 - Post-Processor 2

Thứ tự thực hiện sẽ là:

Pre-Processor 1

Timer 1

Timer 2

Sampler 1

Post-Processor 1

Post-Processor 2

Assertion 1

Pre-Processor 1

Timer 1

Timer 2

Sampler 2

Post-Processor 1

Post-Processor 2

WorkBench: Được xem như một vùng tạm để làm việc, lưu trữ. Tất cả các thành phần bên trong WorkBench sẽ không được thực thi

Assertion

1

GN_GNDL.jmx (D:\Testplan\testplan day len server\GNGN_GNDL.jmx) - Apache JMeter (2.3.4 r785646)

File Edit Run Options Help 0 / 0

Badboy Test Plan

- Thread Group
 - HTTP Cookie Manager
 - User Parameters
 - HTTP Header Manager
 - TTCP_LOGIN
 - TTCP_LOGINPayment
 - HTTP Header Manager
 - Regular Expression Extractor
 - TTCP_LOGINpassportv3/login?app
 - HTTP Header Manager
 - Regular Expression Extractor
 - Response Assertion
 - TTCP_LOGINPayment
 - HTTP Header Manager
 - TTCP_LOGINPayment/Authenticatio
 - HTTP Header Manager
 - GN_GNDL_1
 - GN_GNDL_1\Payment/share/cssse
 - HTTP Header Manager
 - GN_GNDL_1\Payment/paymentActiv
 - HTTP Header Manager
 - GN_GNDL_1\Payment/searchInvoic
 - HTTP Header Manager
 - GN_GNDL_1\Payment/share/cssse
 - HTTP Header Manager
 - GN_GNDL_1\Payment/share/cssse
 - HTTP Header Manager
 - GN_GNDL_1\Payment/selectInvoice
 - HTTP Header Manager
 - GN_GNDL_1\Payment/paymentInSh
 - HTTP Header Manager
 - GN_GNDL_1\Payment/share/cssse
 - HTTP Header Manager

HTTP Request

Name: KPI_GN_GNDL_3\Payment\paymentForServiceAction\paymentForServicePage.do?struts.enableJSONValidation=true
Comments:

Web Server
Server Name or IP: 10.58.3.4 Port Number: 8210 Connect: Response:

Timeouts (milliseconds)

Protocol (default http): http Method: POST Content encoding: UTF-8
Path: /Payment/paymentForServiceAction/paymentForServicePage.do?struts.enableJSONValidation=true
 Redirect Automatically Follow Redirects Use KeepAlive Use multipart/form-data for HTTP POST

Send Parameters With the Request:

Name:	Value	Encode?	Include Eg...
paymentType	inShop	<input type="checkbox"/>	<input checked="" type="checkbox"/>
invoiceListIdSelected	\$(invoiceListId)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
typeOfOS	Windows	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ISDNNo	\$(isdn)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
contractNo		<input type="checkbox"/>	<input checked="" type="checkbox"/>
IDNo		<input type="checkbox"/>	<input checked="" type="checkbox"/>
tinNo		<input type="checkbox"/>	<input checked="" type="checkbox"/>
subscriberListsNull	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
sysDateForCompare	\$(__time(yyyy), \$(__time(MM), \${__time(dd))}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
isRedAlerted	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
isRedAlertedMSG	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
isPinned	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
contractsGhost	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
serviceIsCancelled	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
remainder	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
isLastInvoice	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>
toInvoice	\$(toInvoice)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
currentInvoice	\$(toInvoice)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
isEditForUpdate		<input type="checkbox"/>	<input checked="" type="checkbox"/>
currInvoiceForUpdate		<input type="checkbox"/>	<input checked="" type="checkbox"/>

Add Delete

Send Files With the Request:

File Path:	Parameter...	MIME Type...
------------	--------------	--------------

Add Browse... Delete

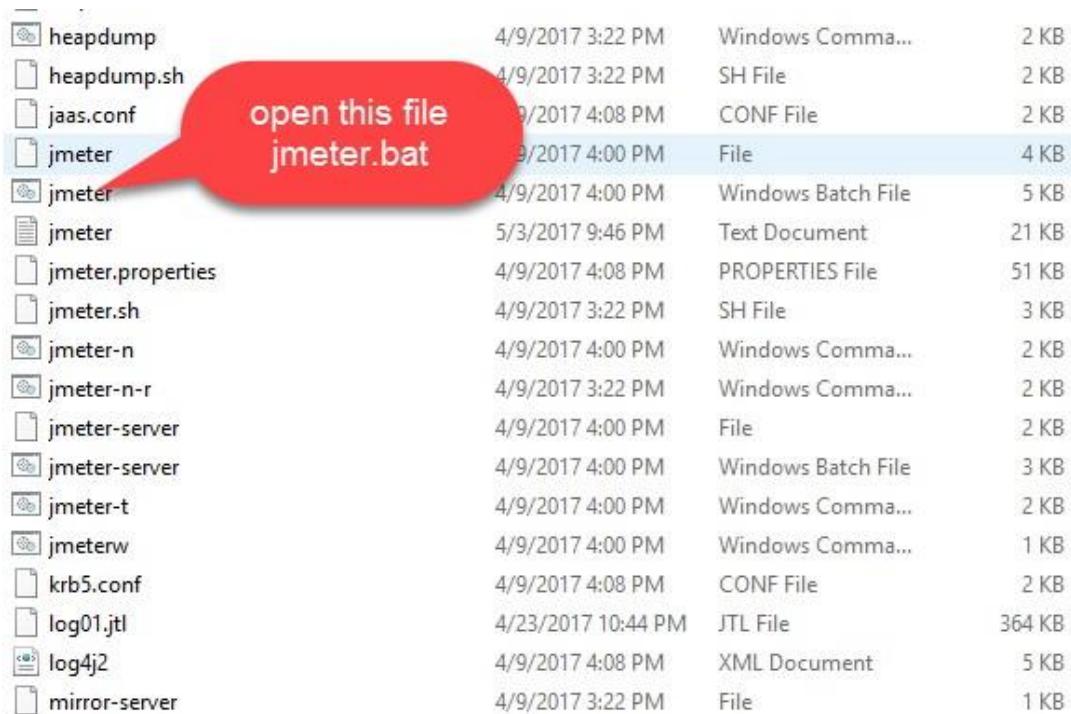
Optional Tasks

Retrieve All Embedded Resources from HTML Files Use as Monitor Save response as MD5 hash?
Embedded URLs must match:

3 Hướng dẫn sử dụng

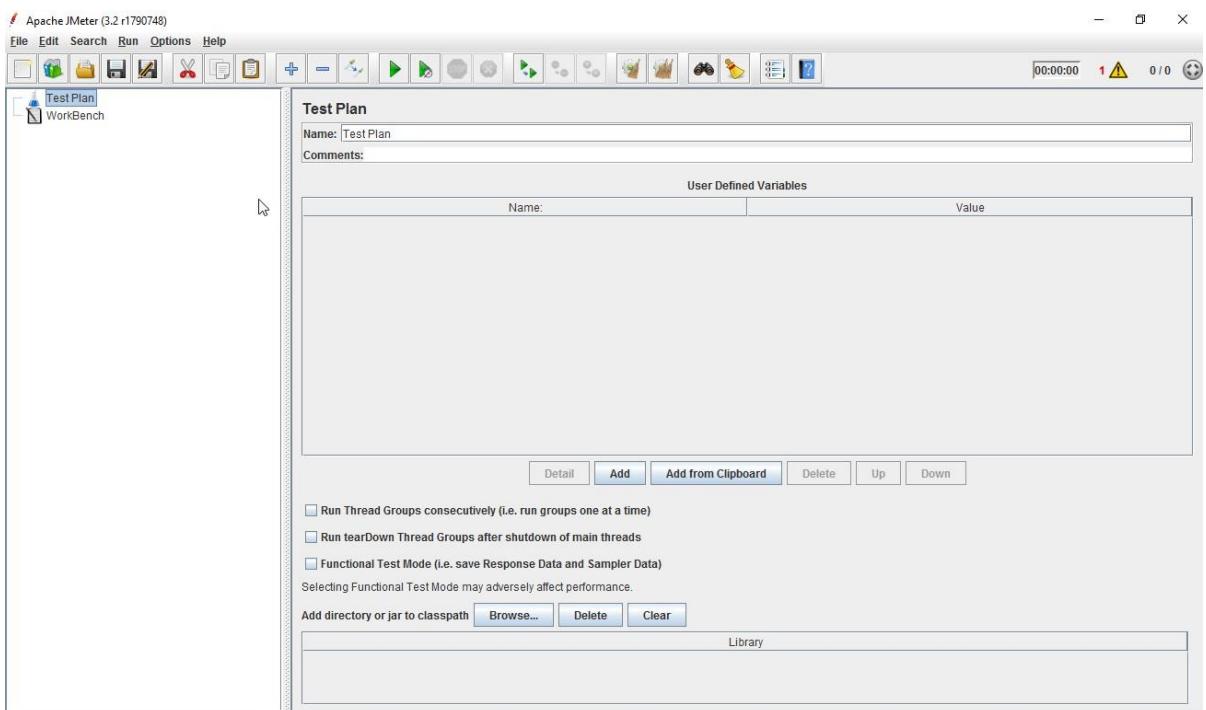
3.1 Các bước tạo test plan

- **Bước 1:** Mở file chạy tool JMETER
- Chạy file jmeter.bat trong thư mục ...\\jmeter_3.2\\bin

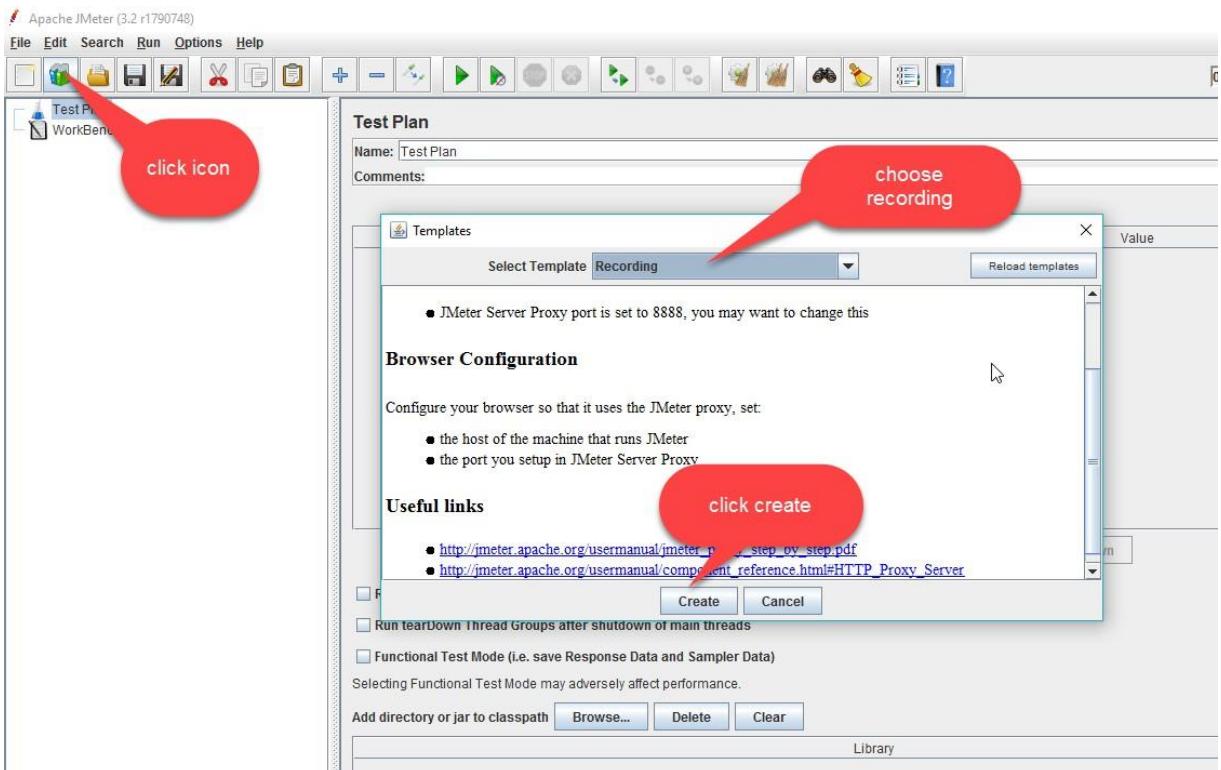


heapdump	4/9/2017 3:22 PM	Windows Comma...	2 KB
heapdump.sh	4/9/2017 3:22 PM	SH File	2 KB
jaas.conf	4/9/2017 4:08 PM	CONF File	2 KB
jmeter	4/9/2017 4:00 PM	File	4 KB
jmeter	4/9/2017 4:00 PM	Windows Batch File	5 KB
jmeter	5/3/2017 9:46 PM	Text Document	21 KB
jmeter.properties	4/9/2017 4:08 PM	PROPERTIES File	51 KB
jmeter.sh	4/9/2017 3:22 PM	SH File	3 KB
jmeter-n	4/9/2017 4:00 PM	Windows Comma...	2 KB
jmeter-n-r	4/9/2017 3:22 PM	Windows Comma...	2 KB
jmeter-server	4/9/2017 4:00 PM	File	2 KB
jmeter-server	4/9/2017 4:00 PM	Windows Batch File	3 KB
jmeter-t	4/9/2017 4:00 PM	Windows Comma...	2 KB
jmeterw	4/9/2017 4:00 PM	Windows Comma...	1 KB
krb5.conf	4/9/2017 4:08 PM	CONF File	2 KB
log01.jtl	4/23/2017 10:44 PM	JTL File	364 KB
log4j2	4/9/2017 4:08 PM	XML Document	5 KB
mirror-server	4/9/2017 3:22 PM	File	1 KB

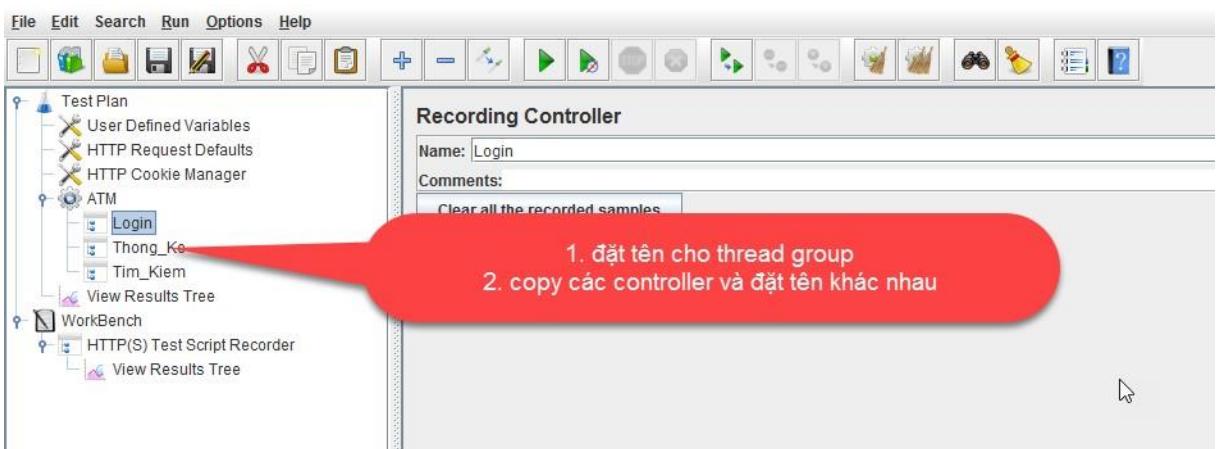
- Khi đó mở ra giao diện tool jmeter



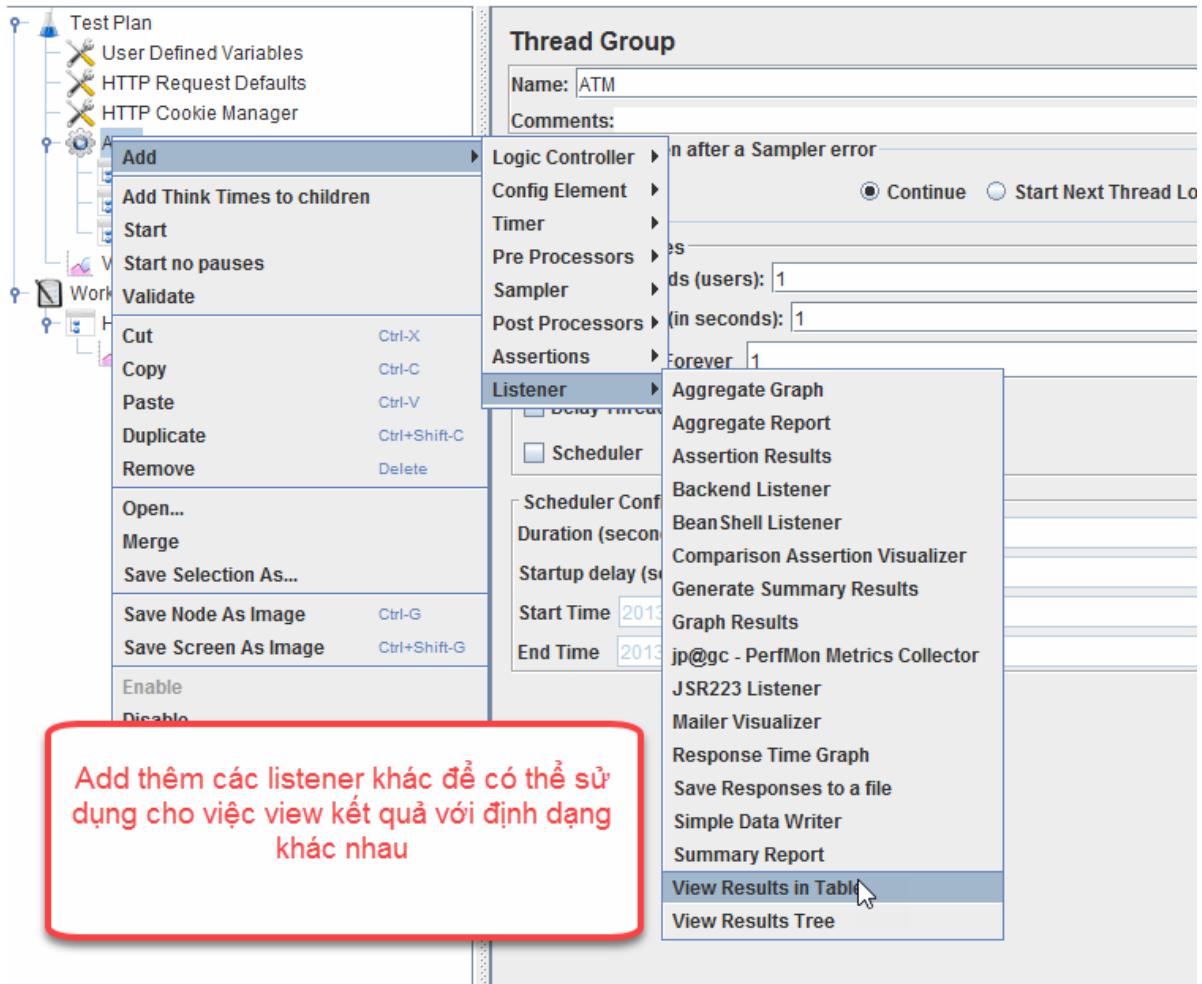
- **Bước 2:** Chọn template có sẵn để có đầy đủ các sampler, controllers, listeners....



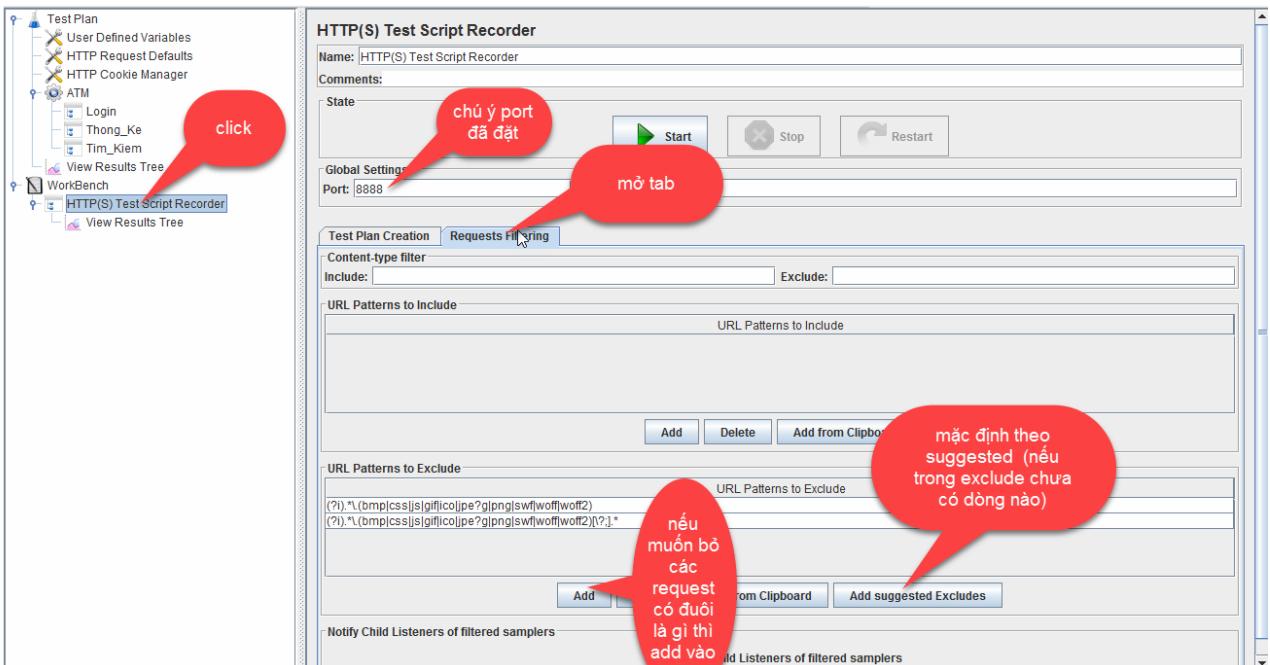
- Đặt tên cho các bước



- **Bước 3:** Add listener để kiểm tra các thông tin các request gửi lên server



- Click chuột phải vào Thread Group → Add → Listener. Bạn có thể chọn các loại report sau:
 - Summary report : báo cáo tổng hợp
 - View Result in Tree: xem kết quả thực hiện của từng giao dịch: thông tin request, thông tin response
 - View result in Table: Xem kết quả thực hiện của từng giao dịch: Pass/Fail, Thời gian xử lý
 - **Bước 4:** Chính sửa HTTP Proxy Server: mục đích là điều khiển quá trình record

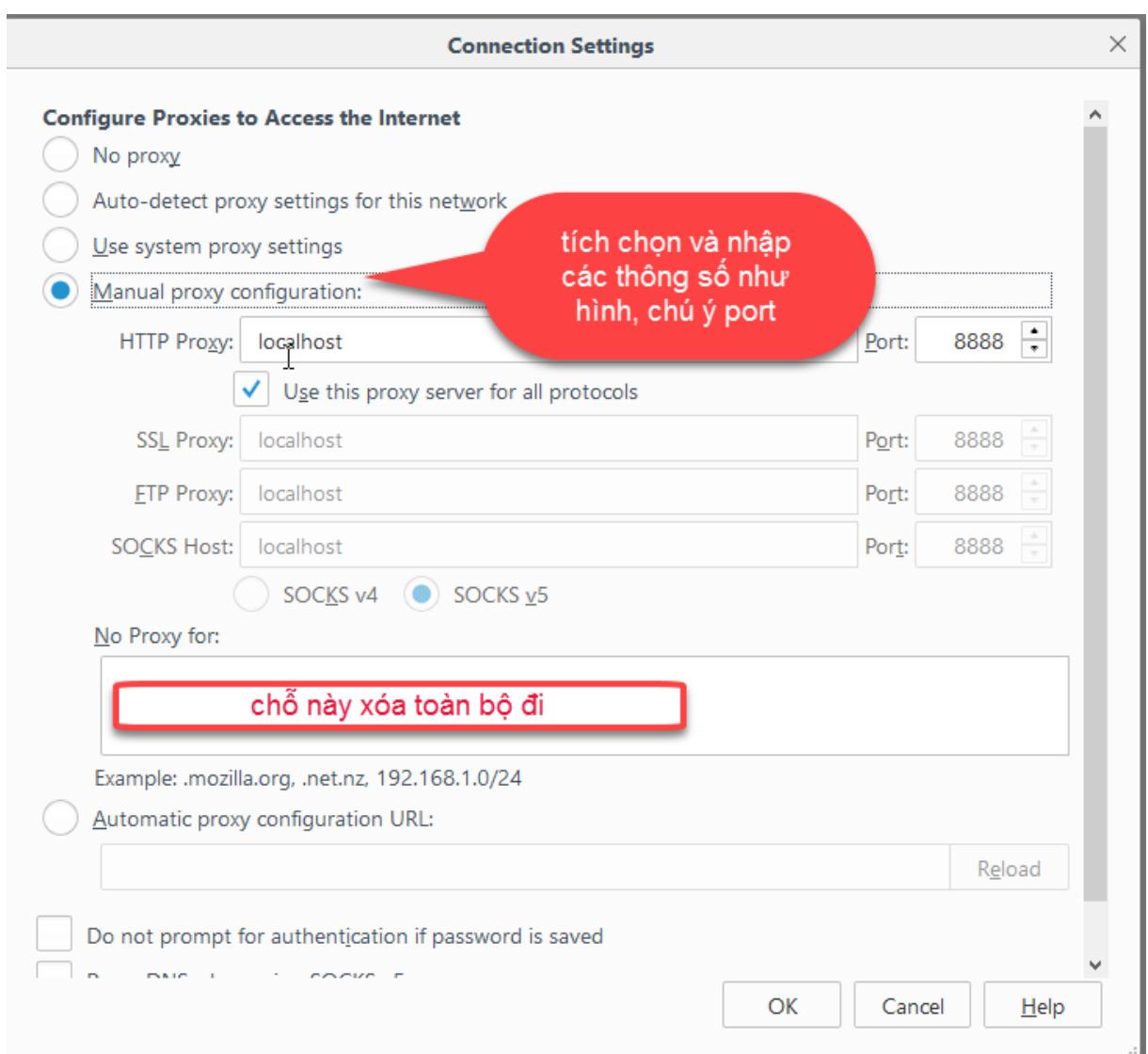
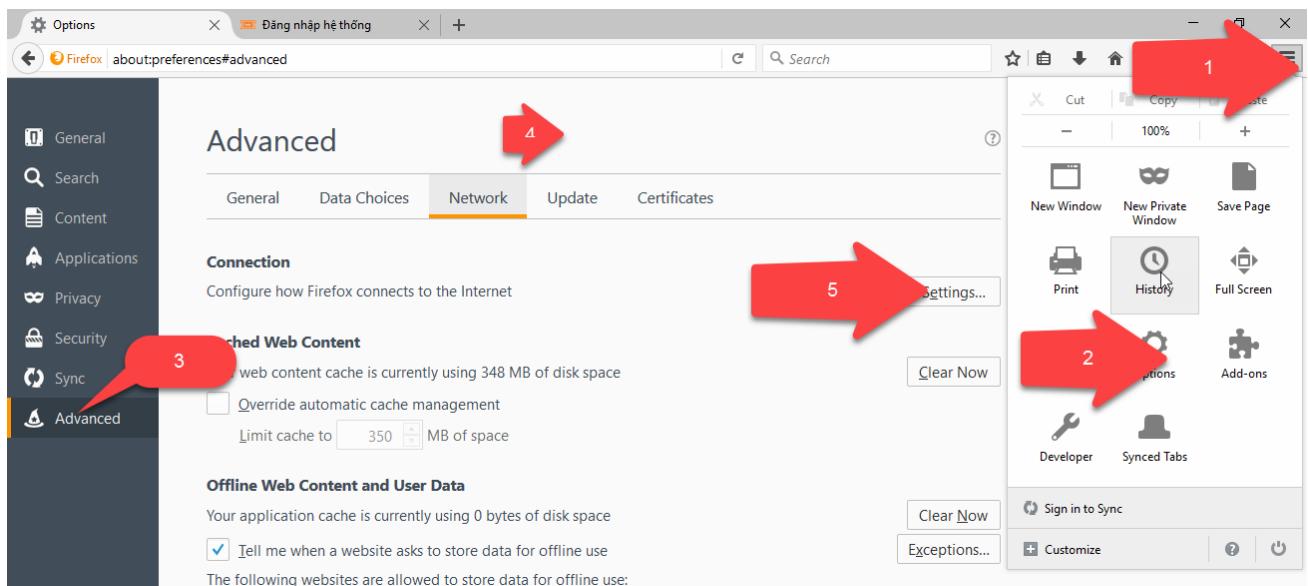


- **Ví dụ :** Các bước thực hiện nghiệp vụ:

1. Đăng nhập vào hệ thống ATM SafeOne theo đường dẫn <http://atm.safeone.vn:8005/login.jsp> theo account: thund1/123456a@
2. Từ menu chương trình chọn thống kê, nhập tên 1 trạm muốn thống kê với các tiêu chí chọn: trạng thái, từ ngày, đến ngày.
3. Nhấn nút tìm kiếm

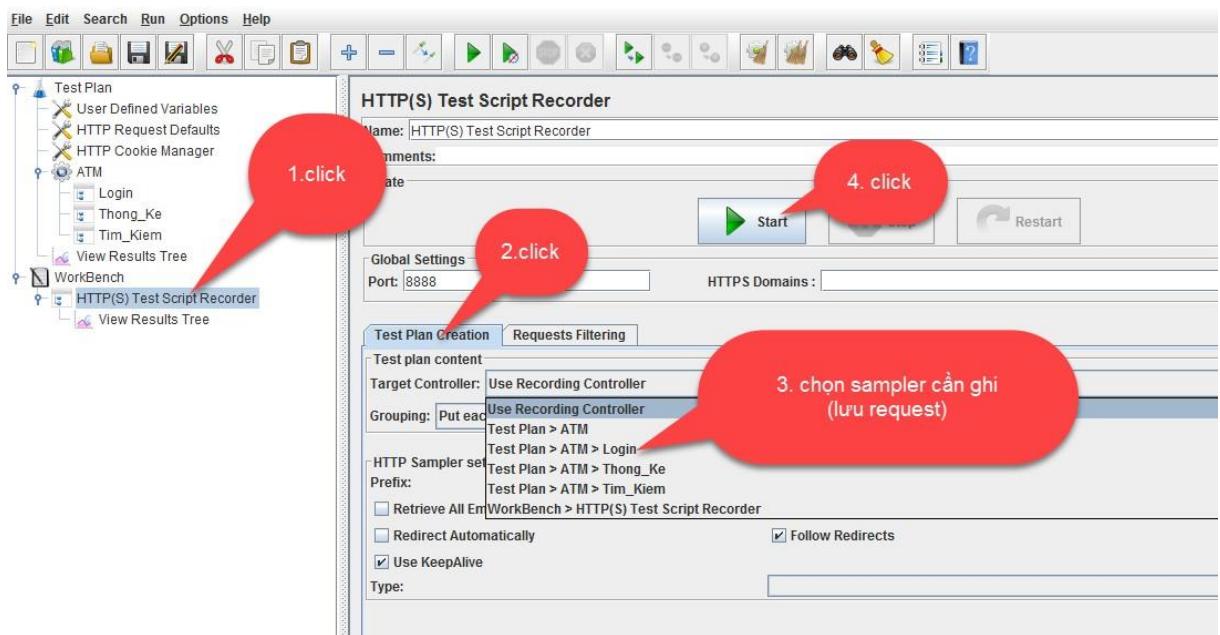
- **Bước 5:**

1. Mở hệ thống bằng trình duyệt firefox <http://atm.safeone.vn:8005/login.jsp>
2. Sửa lại proxy cấu hình mạng là localhost, cổng 8888 (trùng với port cấu hình trong jmeter)



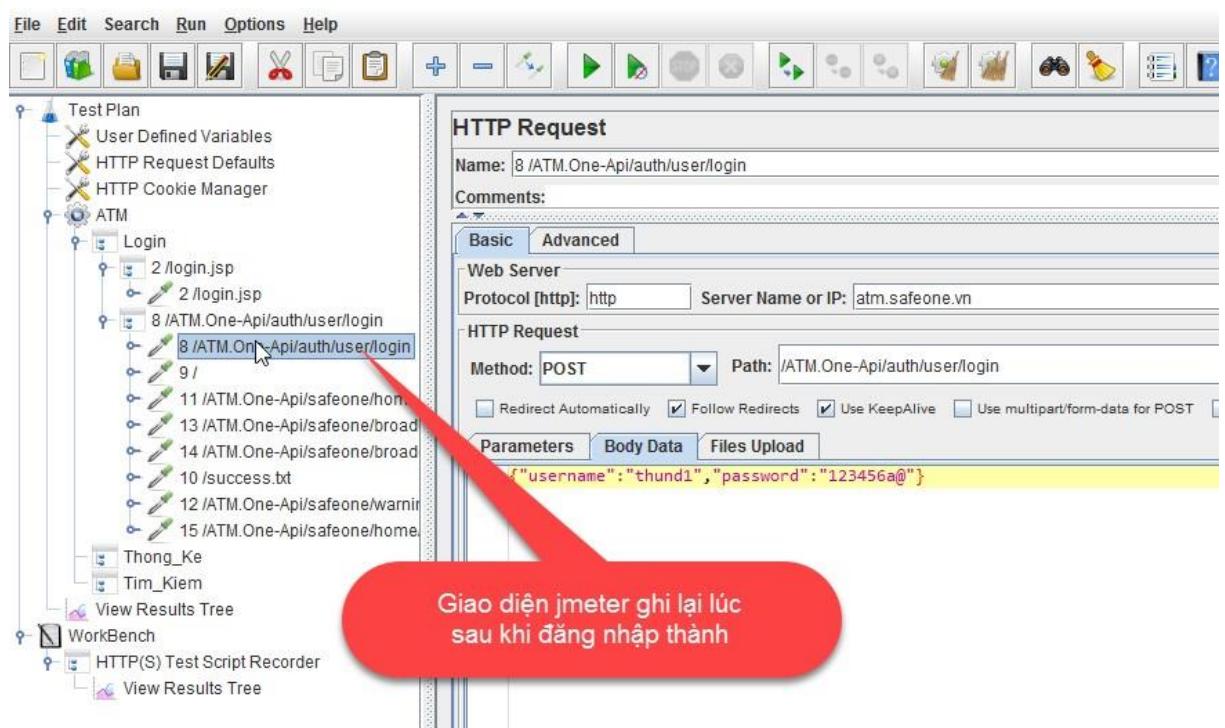
- **Bước 5:** Thực hiện ghi lại từng bước nghiệp vụ theo các bước đã đặt tên ở trên, như sau:

- Trên tool jmeter, tại màn hình của HTTP Proxy Server chọn Target Controller là bước cần record, sau đó nhấn “Start”

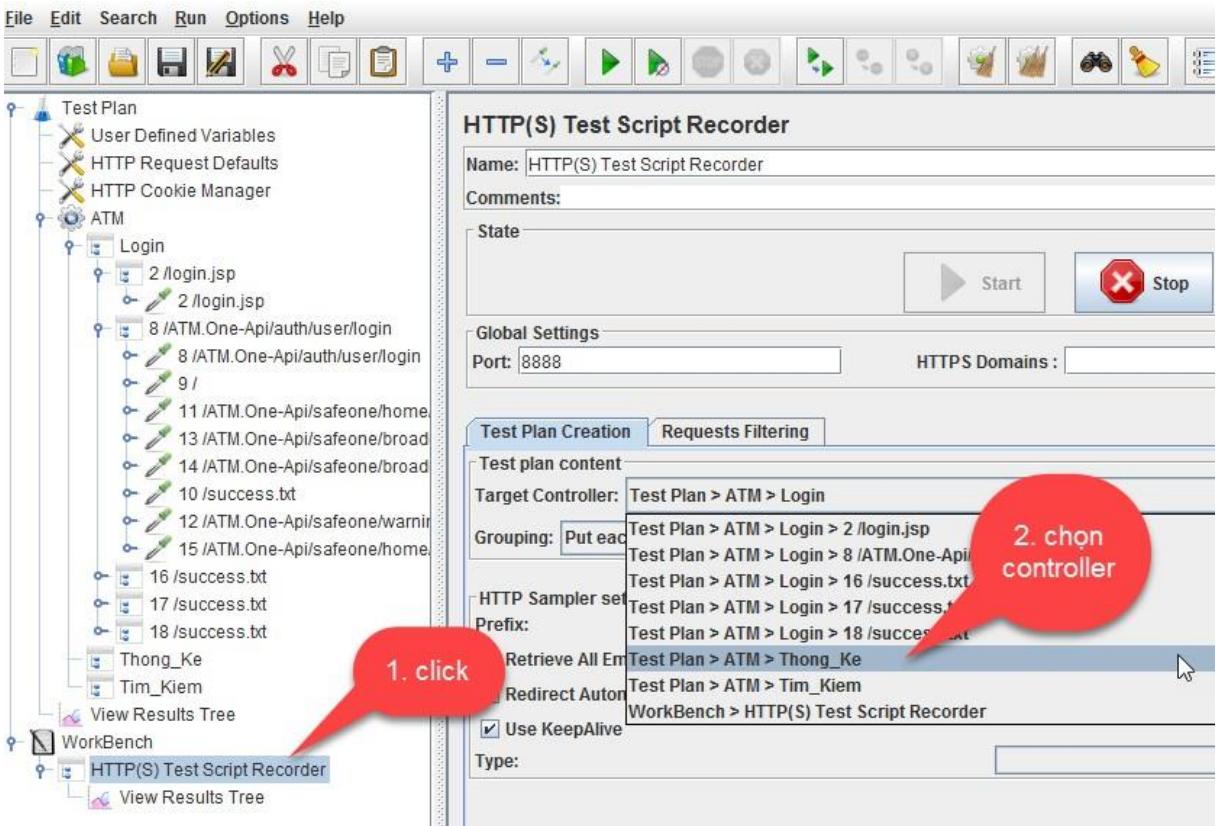


Quay sang trang trình duyệt thực hiện bước tương ứng

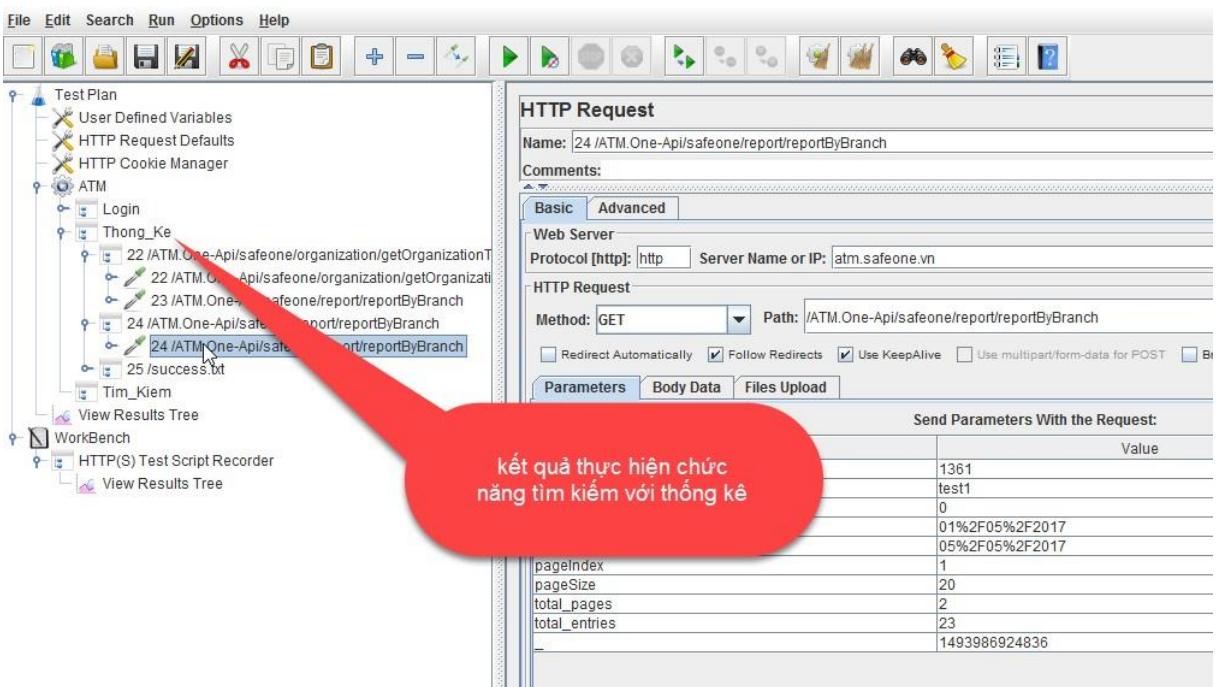
1. Nhấn F5 refresh lại trang <http://atm.safeone.vn:8005/login.jsp>
2. Đăng nhập và ghi lại bình thường các thao tác đăng nhập



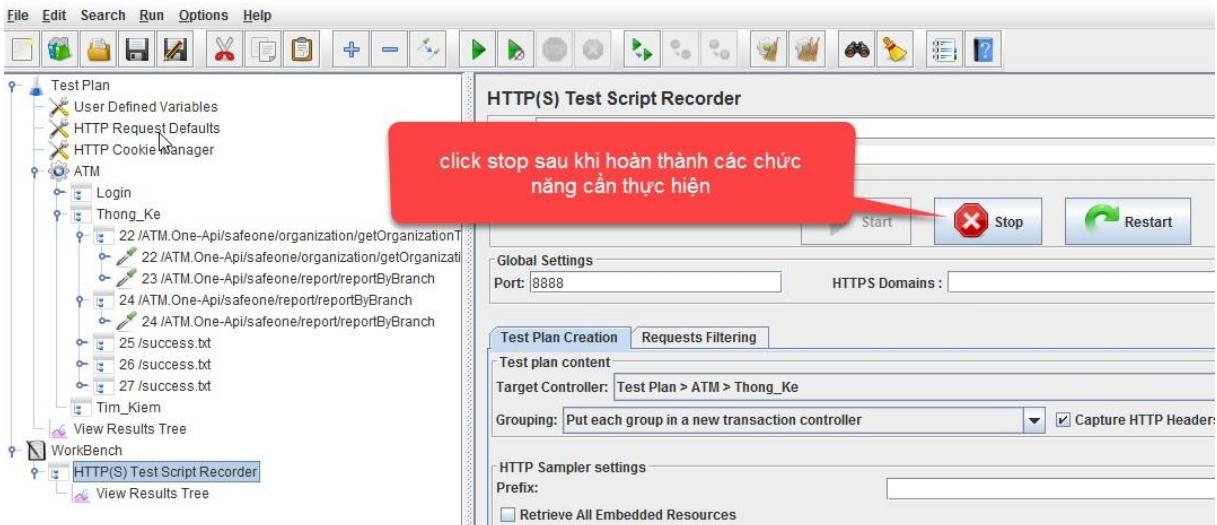
3. Quay lại màn hình jmeter vào lại HTTP (s) Test Script Recorder chọn tiếp sampler cần ghi request vào. Chọn record thông kê:



- Thực hiện ghi tiếp với thao tác trên firefox. Kết quả ghi được các request như sau:



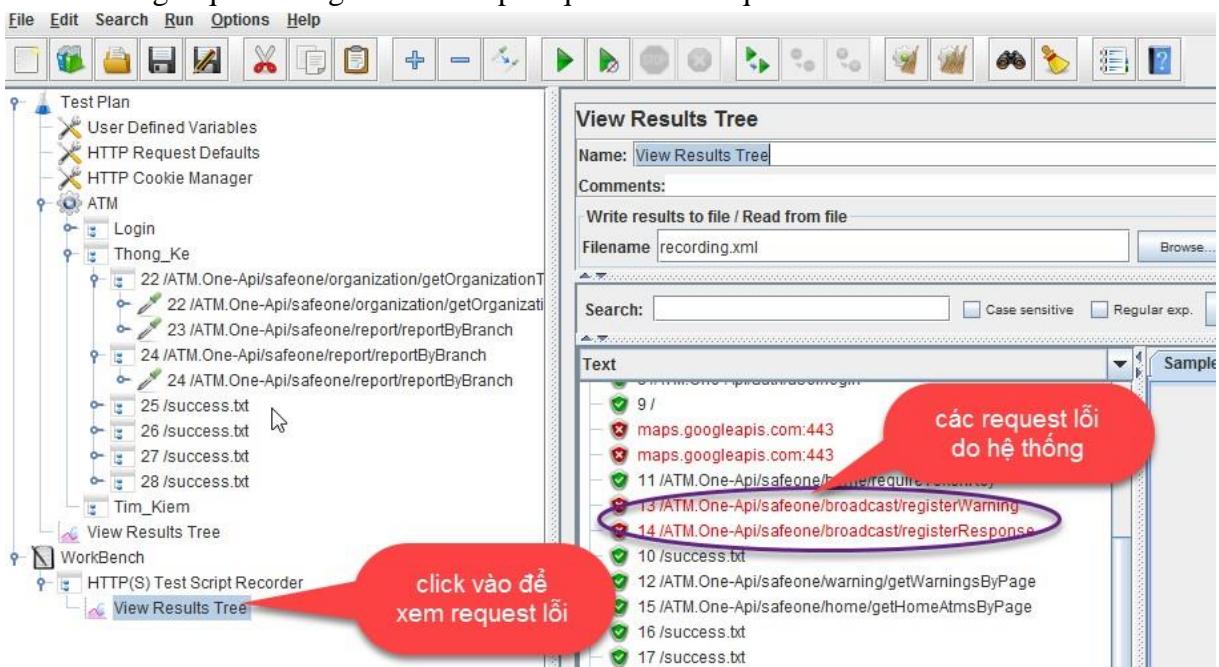
- Xong các bước trên quay lại jmeter, màn hình HTTP Proxy Server, nhấn “Stop”.



Bước 6: Thực hiện tinh chỉnh lại các request cần thiết.

1. Loại bỏ các request không phải chức năng của hệ thống: /success.txt

2. Kiểm tra trong **Workbench\View Results Tree**, những request nào màu đỏ → loại bỏ trong request trong threadGroup: request 13 và request 14.



3. Sau khi loại bỏ các request không cần thiết ta có các request hoàn chỉnh như sau:

The screenshot shows the JMeter Test Plan interface. On the left, there's a tree view of the test plan structure under 'Test Plan'. It includes sections like 'User Defined Variables', 'HTTP Request Defaults', 'HTTP Cookie Manager', 'ATM', 'Login', 'Thong_Ke', 'View Results Tree', 'WorkBench', and 'HTTP(S) Test Script Recorder'. A specific 'HTTP Request' sampler is selected in the tree, which is shown in detail on the right. The 'Basic' tab of the 'HTTP Request' dialog is open, displaying the following configuration:

- Web Server**: Protocol [http], Server Name or IP: atm.safeone.vn
- HTTP Request**: Method: POST, Path: /ATM.One-Api/auth/user/login
- Parameters**: A table with one row containing the parameter: {"username": "thund1", "password": "123456a@0"}

Bước 7: Đặt assertion cho request cần kiểm tra

- Nếu kết quả trả về có dòng chữ này -> request là chạy đúng
- Nếu kết quả trả về không có dòng chữ này -> request bị fail

1. Kiểm tra trên web:

- Login thành công sẽ hiển thị chữ “Thu”
- Thống kê thành công sẽ hiển thị như hình.

» Thống kê

The screenshot shows a web-based reporting application. At the top, there are two tabs: 'Báo cáo theo chi nhánh' and 'Báo cáo theo ATM'. Below the tabs is a search form with fields for 'Đơn vị' (test1), 'Trạng thái' (Chưa xử lý), and 'Từ ngày' (01/05/2017). There are also 'Tim kiem' (Search) and 'Xuất ra Excel' (Export to Excel) buttons. Below the form is a section titled 'Kết quả tìm kiếm' (Search results) containing a table with the following data:

STT	Mã cây ATM	Địa chỉ	Chi nhánh
1	34352443	Hà nội	test1
2	34354363	%\$T%\$%^	test1
3	ADWDJGHF	<p> hà nội</P>	test1
4	ADW214332	Hà nội	test1

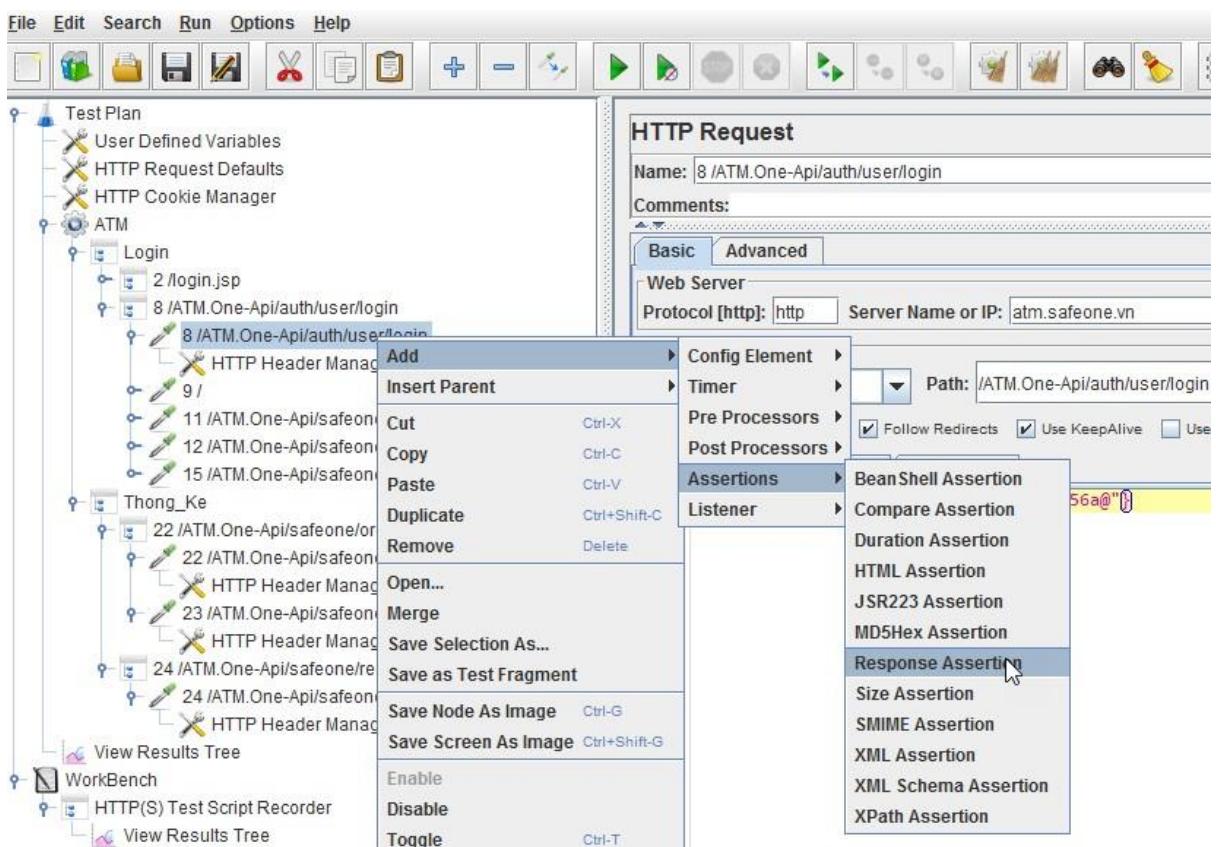
At the bottom of the results table, it says 'Hiển thị trang 1 / 1. Số bản ghi 4.'

2. Mở lại workbench\view result tree

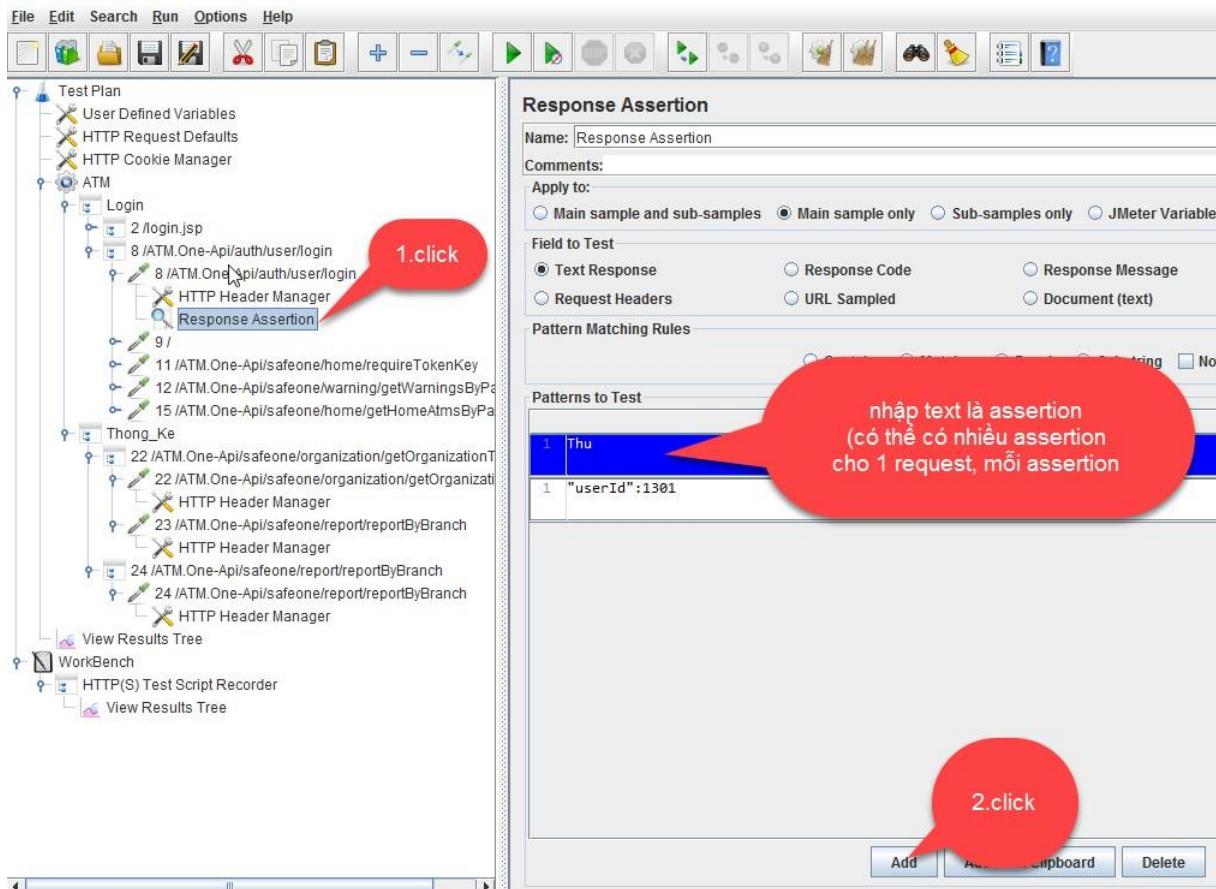
Kiểm tra các request có dữ liệu gửi lên và response trả về tương ứng: request 8 và request 24 (xem tại tab request có chứa các dữ liệu nhập/chọn được gửi đi), sang tab response sẽ có dữ liệu trả về tương ứng.

Kiểm tra trong response để tìm những điểm xác định request thực hiện thành công. Và add assertions tương ứng cho các request trên ThreadGroup.

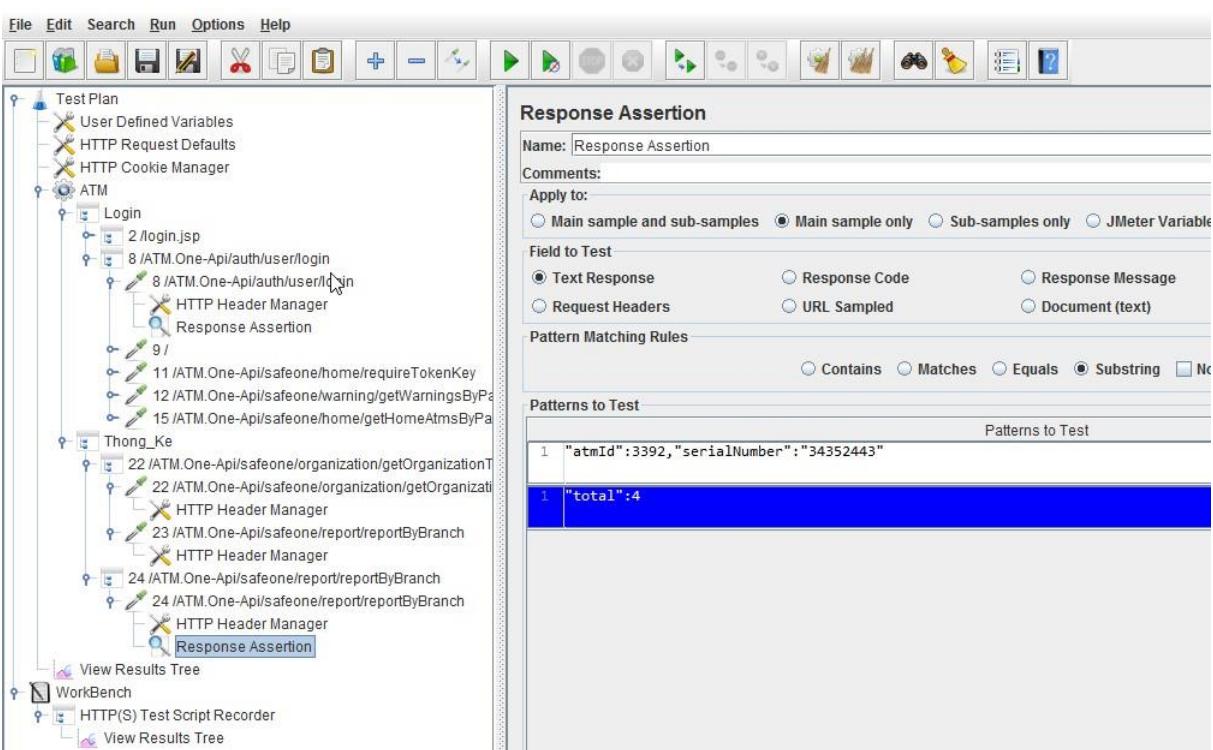
3. Cách add Assertions



Add assertions như sau:

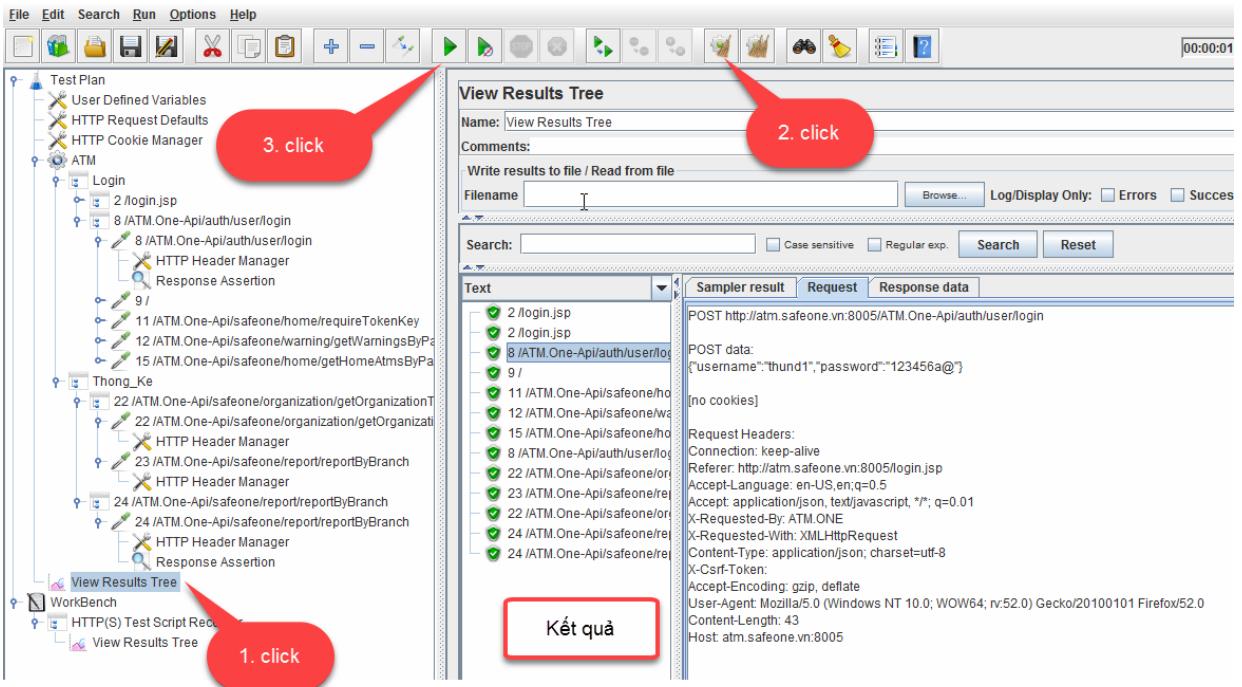


Làm tương tự với thông kê,



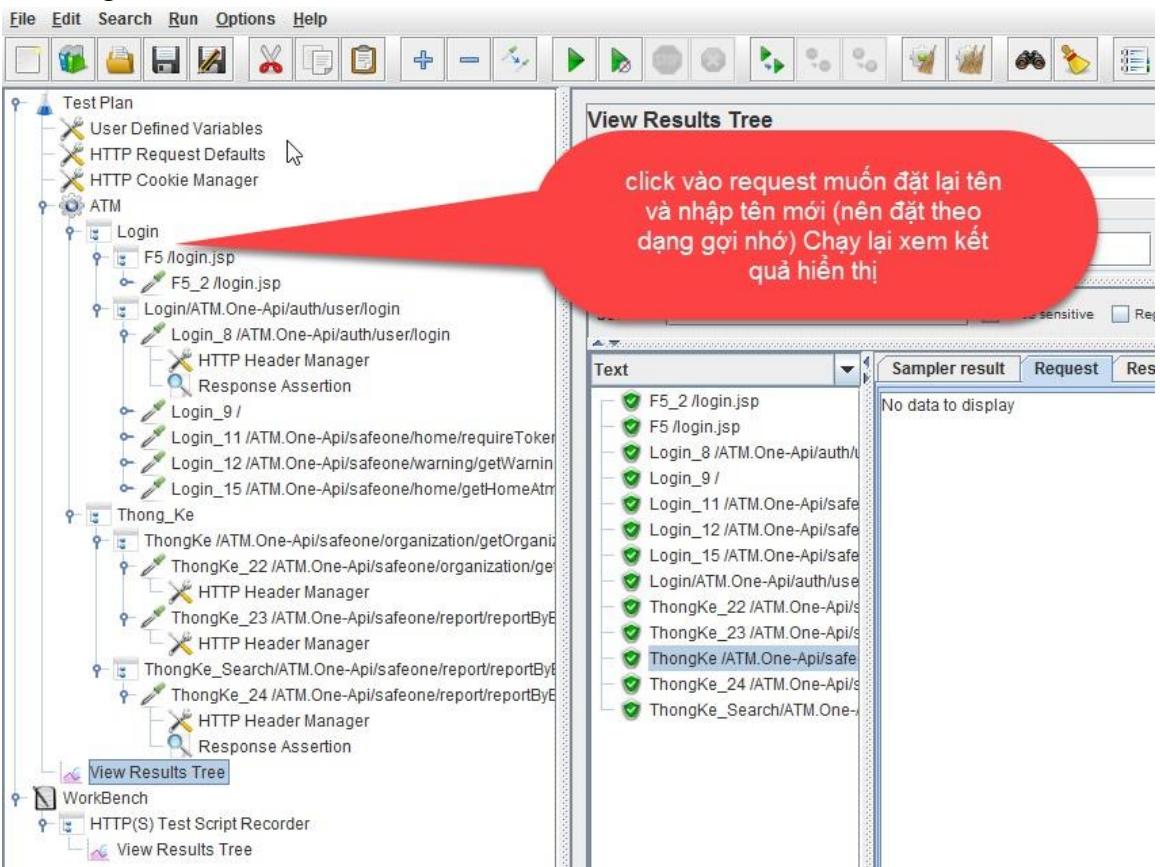
Bước 8: Thực hiện chạy kiểm tra kết quả các script đã được record

- Trên màn hình của View results tree, ấn tổ hợp phím Ctrl + R hoặc nhấn Run trên thanh điều khiển để
- chạy lại testplan. Kết quả được chỉ ra trên cửa sổ:



- Các request màu xanh là không bị lỗi, các request màu đỏ là bị lỗi.
- Nếu request bị lỗi là các hàm truy vấn dữ liệu quan trọng: phối hợp với dev kiểm tra lại xem có phải lỗi để xử lý

Bước 9: Chuẩn hóa lại các bước trong testplan Đặt mã bước vào đầu các request trong bước đó.

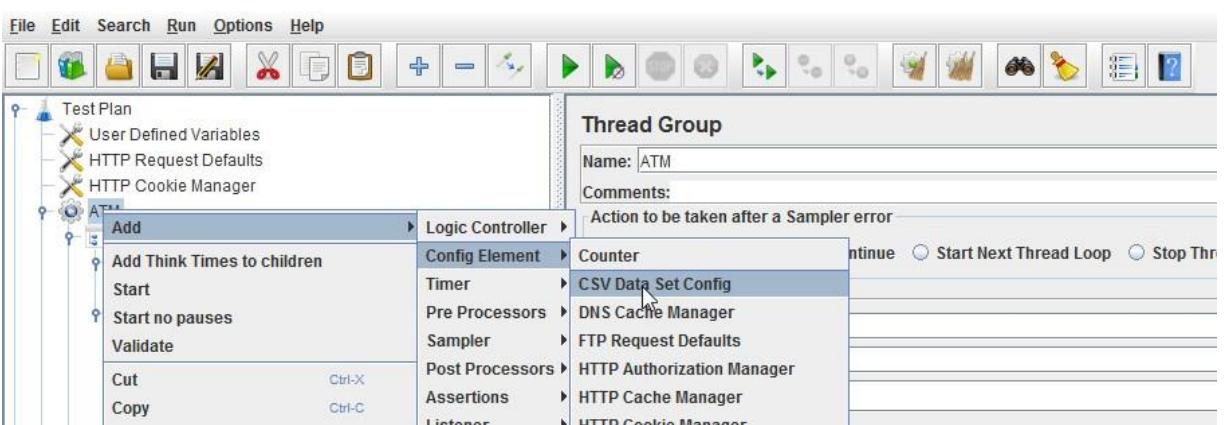


3.2 Tham biến từ file excel

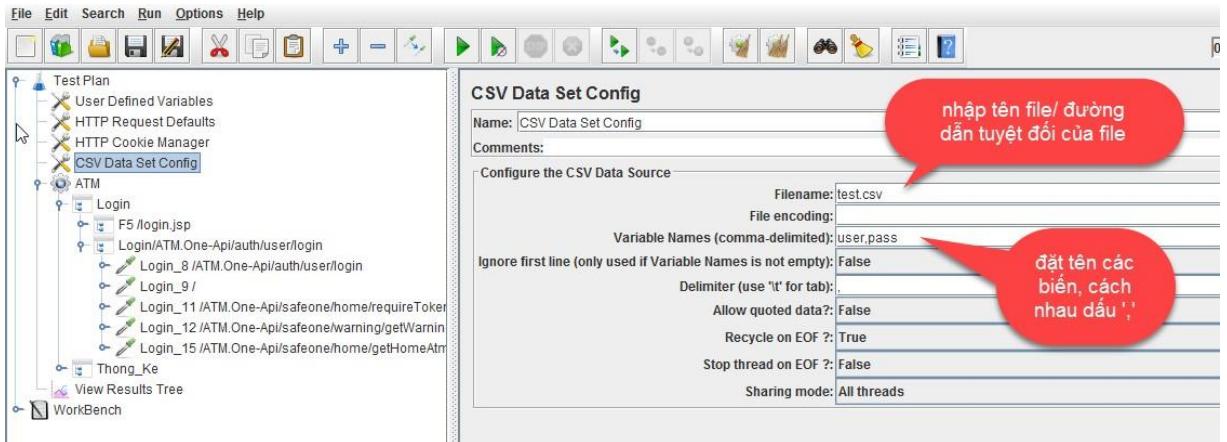
- Để thuận tiện cho việc thao tác dữ liệu Jmeter hỗ trợ việc truyền tham số của biến. Giả sử bạn có biến là a, để lấy ra giá trị của biến, bạn cần gọi theo cấu trúc \${a} **CSV Data Set Config**.
- Khi muốn chạy nhiều CCU, cần sử dụng đến danh sách các account. Jmeter hỗ trợ việc lấy dữ liệu từ file .csv và .txt (nên sử dụng file .csv)

Note: file dữ liệu nên để trong thư mục **bin**. VD: Testplan đăng nhập thì cần đầu vào là các bộ account đăng nhập

- Phải chuột vào Thread Group -> Add -> Config Element -> CSV Data Set Config

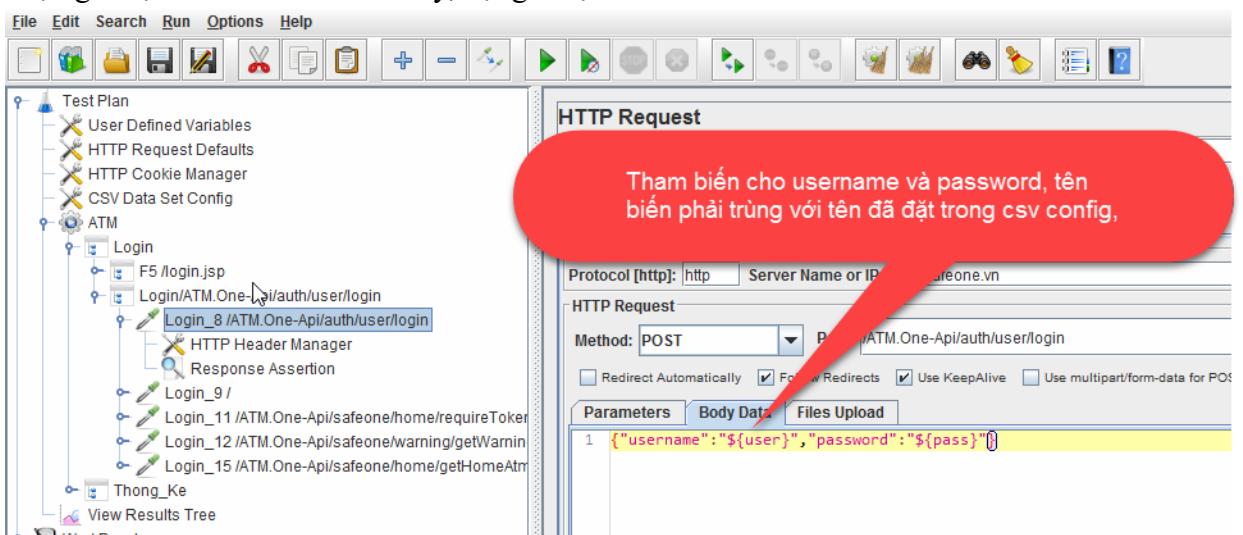


- Nhập các thông tin trên màn hình giao diện của CSV



Bao gồm:

- Filename: đường dẫn\tên file csv chứa dữ liệu đầu vào
- Variable Names: khai báo thứ tự các biến trong file csv
- Delimiter: định nghĩa dấu phân cách các biến trong file csv Sau đó tại các request nhận giá trị đầu vào từ file csv này, đặt giá trị biến



3.3 Bypass proxy

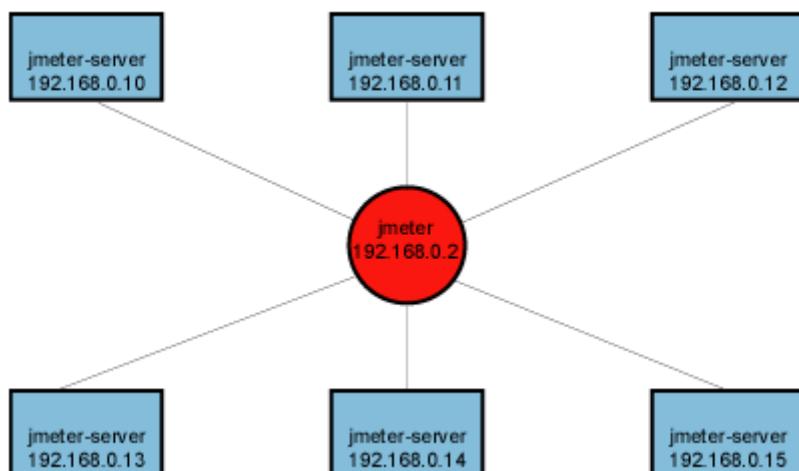
- Để test hiệu năng các hệ thống cần thông qua proxy, cần thực hiện các bước sau:
 1. Tạo 1 file .bat có cấu trúc như sau:
 - jmeter -H my.proxy.server -P 8000 -u username -a password -N localhostcác tham số:
 - H [proxy server hostname or ip address]
 - P [proxy server port]
 - N [nonproxy hosts] (e.g. *.apache.org|localhost)
 - u [username for proxy authentication - if required]
 - a [password for proxy authentication - if required]
- Ví dụ :
- Proxy: 10.61.11.38
Port: 3128
- ➔ Jmeter -H 10.61.11.38 -P 3128 -N **localhost**

2. Lưu file dưới đuôi .bat (vd: jmeterproxy.bat)
3. Chạy lại jmeter với file jmeterproxy.bat

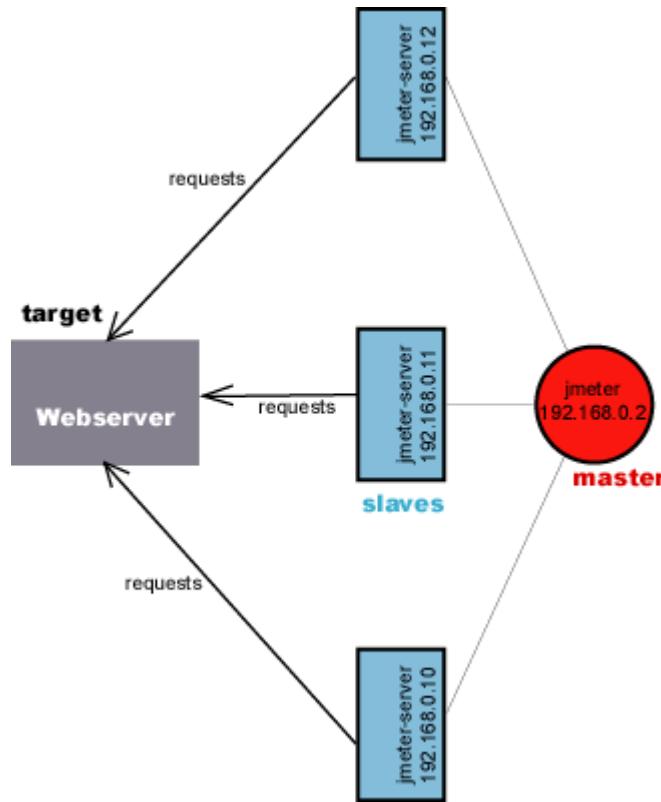
 jmeter.sh	5/14/2016 11:47 AM	SH File	3 KB
 jmeter-n	5/14/2016 11:51 AM	Windows Comma...	2 KB
 jmeter-n-r	5/14/2016 11:47 AM	Windows Comma...	2 KB
 jmeterproxy	5/5/2017 8:52 PM	Windows Batch File	1 KB
 jmeter-server	5/14/2016 11:51 AM	File	2 KB
 jmeter-server	5/14/2016 11:51 AM	Windows Batch File	3 KB
 jmeter-t	5/14/2016 11:51 AM	Windows Comma...	2 KB

3.4 Remote testing

- Trong quá trình test hiệu năng, để có thể sử dụng cùng lúc nhiều máy client đầy tải mà không cần phải vào từng máy client thực hiện đầy, ta có thể dùng chức năng remote testing của Jmeter.
- Để thực hiện được remote testing, cần những điều kiện sau:
 - ✓ Đảm bảo các máy client đã tắt hết tường lửa
 - ✓ Các máy client thuộc cùng subnet
 - ✓ Máy chủ chịu tải có thể cùng subnet hoặc không.
 - ✓ Đảm bảo Jmeter có thể truy cập vào máy chủ
 - ✓ Sử dụng cùng 1 phiên bản Jmeter trên các máy client đầy tải.
- Sau khi thiết lập xong, remote testing sẽ hoạt động theo mô hình master – slave. Một máy client sẽ được chọn làm master để điều khiển từ xa các máy client đóng vai trò slave.



- Mô hình kiểm thử Master – Slave trên Jmeter như sau:



- Master: Máy client đóng vai trò điều khiển test từ xa, chạy Jmeter GUI.
- Slave: Máy client chạy file Jmeter-server để nhận lệnh từ máy client Master – gửi request lên hệ thống máy chủ chịu tải.
- Target: hệ thống máy chủ chịu tải.

o Các bước cơ bản tạo Remote testing

Bước 1: Trên các máy slave, chạy file Jmeter-server.bat.

Bước 2: Trên máy master, mở file Jmeter.properties, thêm các địa chỉ của máy slave vào dòng “Remote_host = 127.0.0.1,”

```

222 #
223 #-----#
224 # Remote hosts and RMI configuration
225 #-----#
226
227 # Remote Hosts - comma delimited
228 remote_hosts=127.0.0.1, // diều các host slave tại đây
229 #remote_hosts=localhost:1099,localhost:2010
230
231 # RMI port to be used by the server (must start rmiregistry with same port)
232 #server_port=1099
233
234 # To change the port to (say) 1234:
235 # On the server(s)
#-----#

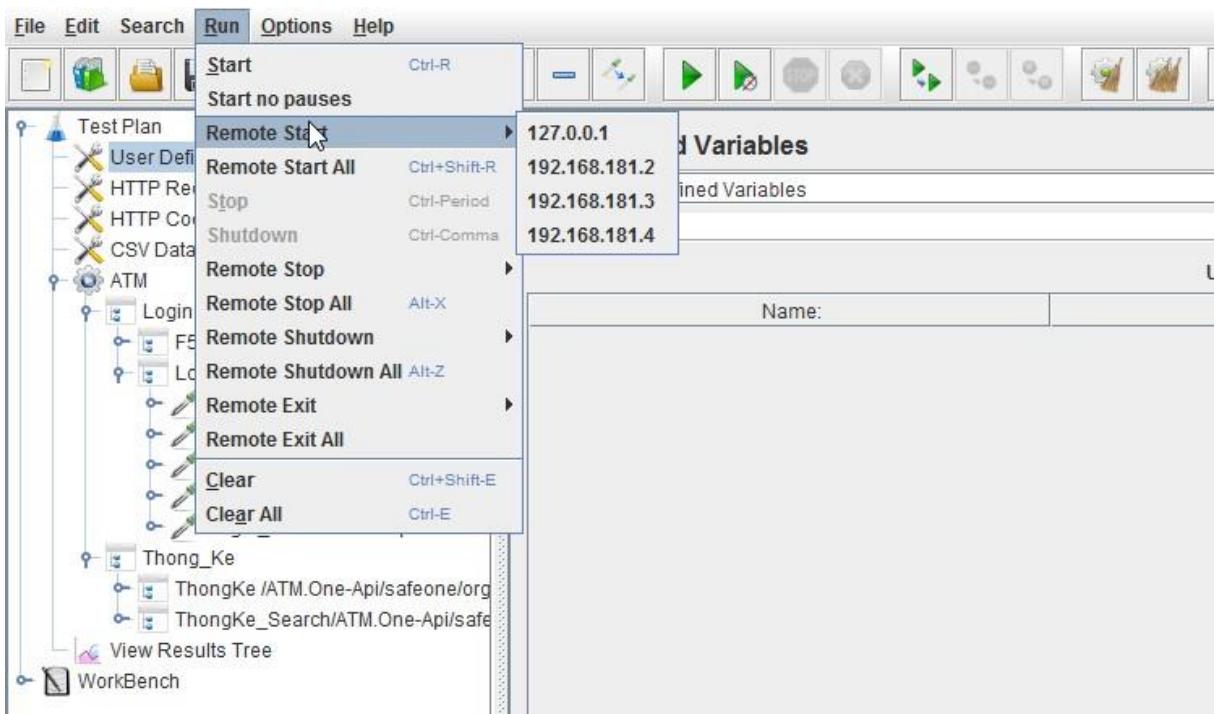
```

Bước 3: Mở giao diện Jmeter GUI trên máy Master, mở test plan cần chạy.

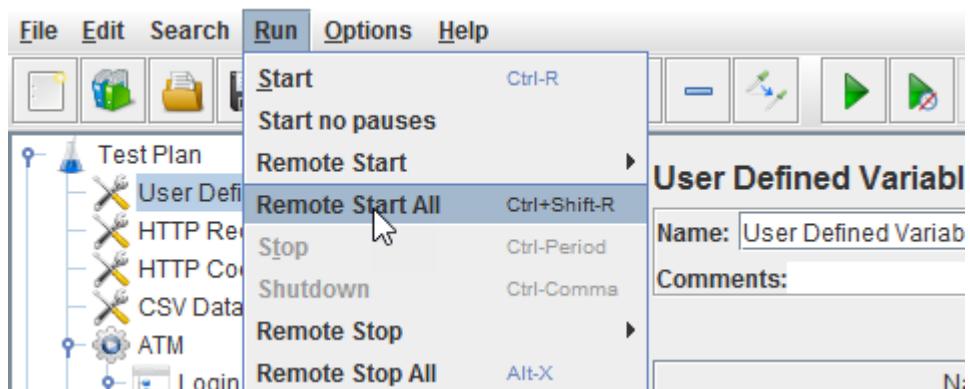
Bước 4: Thực hiện test.

⊕ Nếu đẩy tải từ 1 máy slave:

Run/ Remote start ➔ chọn ip máy client muốn đẩy tải.



 **Đẩy tải từ tất cả các máy slave**

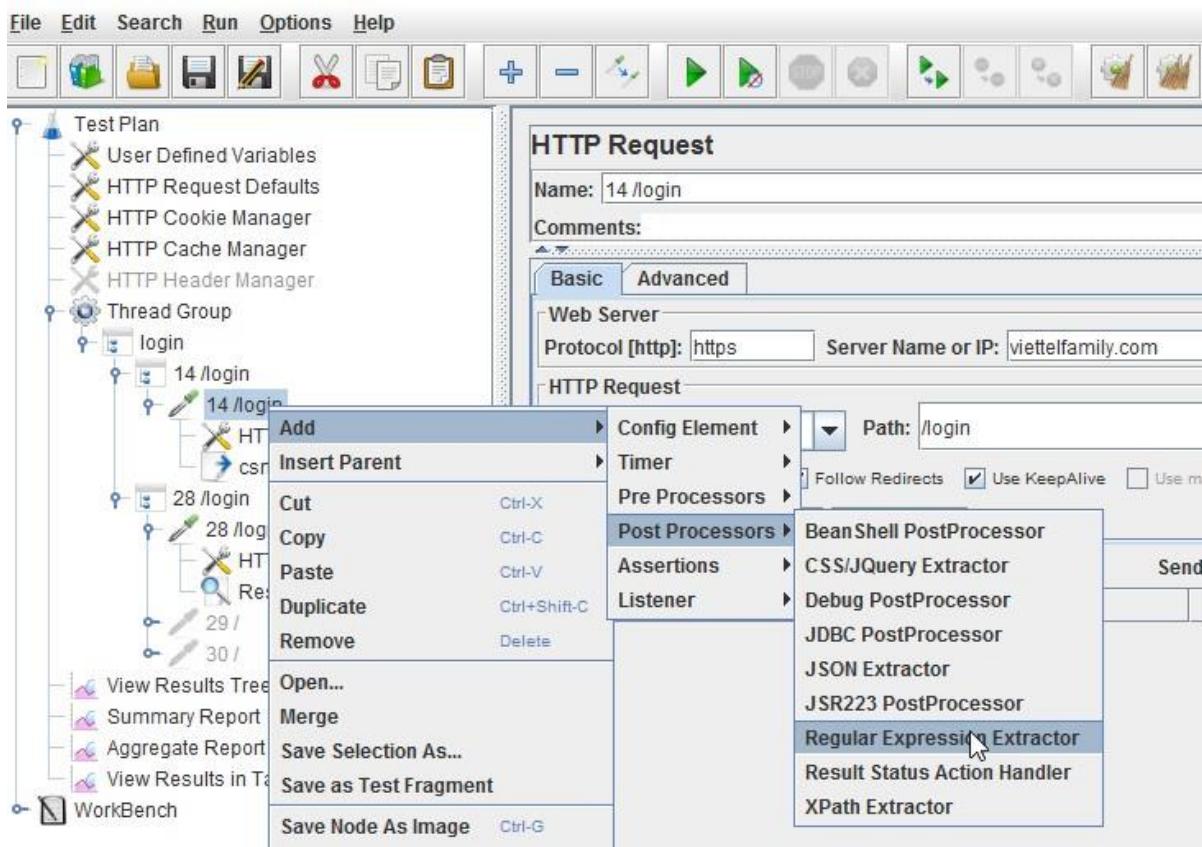


3.5 Tham biến các giá trị thay đổi

- Giá trị thay đổi là một biến được gán các giá trị khác nhau khi thực hiện 1 thao tác hay gửi 1 request. Ví dụ: token (mỗi lần thực hiện gửi 1 request giá trị của token sẽ khác nhau)



- Regular Expression Extractor(REE) cho phép bạn lấy ra được giá trị của 1 biến nào đó từ server trả về trong response message, sử dụng cấu trúc của biểu thức chính quy. REE được xử lý sau request mà nó muốn lấy kết quả trả về, nó được add bên trong request mà bạn cần lấy thông tin trong bản tin response tương ứng.
- Ví dụ: Bạn cần lấy thông tin về giá của sản phẩm abc trong bản tin response của bản tin Request 14/login
- Trong Testplan, click chuột phải vào Request 14/login → Add → Post-Processor → Regular Expression Extractor.



Regular Expression Extractor

Name:	Regular Expression Extractor
Comments:	
Apply to:	<input checked="" type="radio"/> Main sample only <input type="radio"/> Sub-samples only <input type="radio"/> Main sample and sub-samples <input type="radio"/> JMeter Variable <input type="checkbox"/>
Response Field to check	<input checked="" type="radio"/> Body <input type="radio"/> Body (unesaped) <input type="radio"/> Headers <input type="radio"/> URL <input type="radio"/> Response Code <input type="radio"/> Response Message
Reference Name:	<input type="text"/>
Regular Expression:	<input type="text"/>
Template:	<input type="text"/>
Match No. (0 for Random):	<input type="text"/>
Default Value:	<input type="text"/>

Các thông tin tại màn hình Regular Expression Extractor bao gồm:

- Reference name: Tên của biến nhận thông tin
- Regular Expression: Cấu trúc của dòng chứa thông tin cần lấy, vị trí chưa thông tin cần lấy. Thông tin filter cần phải đặt trong 2 dấu () sẽ được viết dưới dạng ngữ pháp của regular expression. Cấu trúc của dòng này sẽ được lấy từ View Results tree -> Show Text (tại bước request tương ứng).
- Ví dụ dòng thông tin chứa giá của sản phẩm abc có dạng html như sau: Name='abc', price=10000, quantity=50
Khi đó Regular Expression: name='abc', price=(/d+), quantity
- Template: \$1\$
- Match No.: 0
- Default Value: Giá trị hiển thị trong trường hợp không lấy được thông tin cần thiết.

Một số cú pháp thông dụng:

- .+: bất kể kí tự nào
- \d+: tất cả các kí tự số
- [^"] : tất cả các kí tự trừ kí tự “ ”

Regular Expression Extractor

Name: csrf_token

Comments:

Apply to:

Main sample and sub-samples Main sample only Sub-samples only JMeter Variable

Field to check

Body Body (unescape) Body as a Document Response Headers Request Headers U

Reference Name:	csrf_token
Regular Expression:	<input type="hidden" name="user[_csrf_token]" value="(.*?)"
Template:	\$1\$
Match No. (0 for Random):	1
Default Value:	not found
	<input type="checkbox"/> Use empty default value

- Để regex được giá trị respond trả về ta cũng có thể tìm kiếm ngay trên màn hình respond trả về, khi tìm kiếm cần check chọn Regular exp.

View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: Errors Successes Configure

Search: Case sensitive Regular exp.

Text

```

<section id="Login">
    <div class="loginContent vertical-middle">
        <center>
            <a href="/">
        </center>
    <div class="formLogin" style="font-size: 14px;">
        <div class="titleLogin colorVTF">Đăng nhập</div>
        <form id="contact-form" method="post" action="" class="standard-form idea-form" enctype="multipart/form-data">
            <input type="hidden" name="user[_csrf_token]" value="5a9b94faeb23f3e85e5de84e1a0aea" id="user__csrf_token" />
            <div class="error-messages-content">
                <input type="hidden" name="_csrf_token" value="--><!-->-->
                <div class="form-group inputHasIcon">
                    <input class="inputLogin hasIcon" placeholder="Tài khoản SSO" type="text" name="user[email]" value="" id="user_email" />
                    <span class="username2 formIcon colorVTF"></span>
                </div>
                <div class="form-group inputHasIcon">
                    <input class="inputLogin hasIcon" placeholder="Mật khẩu SSO" type="password" name="user[password]" id="user_password" />
                </div>
                <div class="command-button form-group" style="float: right; font-weight: 500; font-style: normal;" class="colorBlue" href="/changePassword">Đổi mật khẩu</a>
                <div style="clear: both;"></div>
            </div>
        </form>
    </div>
</div>

```

Scroll automatically?

Search: <input type="text" value="<input type="hidden" name="user[_csrf_token]" value="(.*?)" />"/> Case sensitive Regular exp.

3.6 Winform

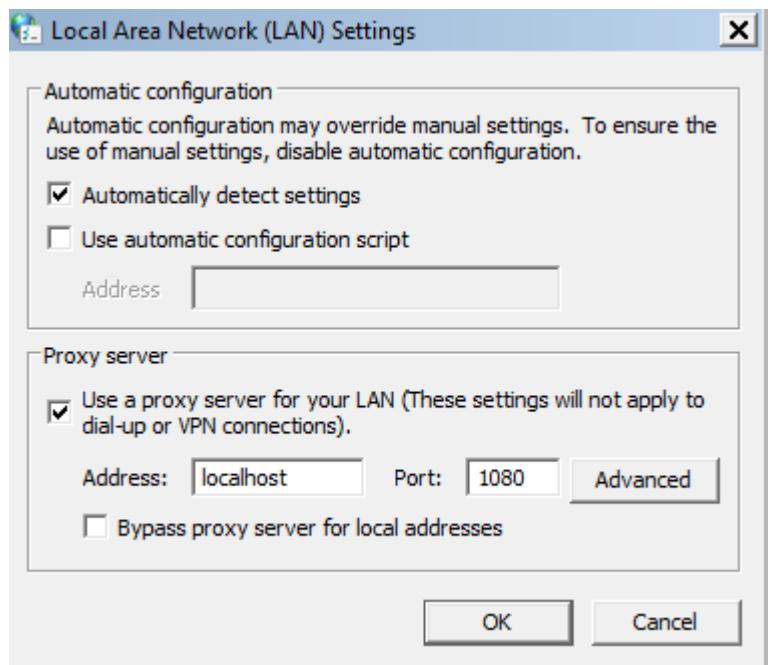
Bước 1: Tạo thread group, add Recording controller nhưng bình thường

Bước 2: Trong phần workbench vẫn Add=> non test element=> https recode test script

- config như sau: Chọn cổng là 1080, các thông số để như record trên web

Bước 3: Trên máy client chạy winfrom vào phần cấu hình mạng

Open network and sharing => internet option => tab connectioin=> Lan setting chỉnh lại thành localhost, port phải là 1080



Tương tự như khi record ứng dụng web phải cài đặt là localhost 8080 hoặc 8888
thì với winform là **1080**.

Lưu ý: chỉ cấu hình như trên khi bắt đầu record qua Jmeter. Nếu chưa record muốn mở winform lên phải cài đặt lại như bình thường không được để localhost

3.7 Mobile app

Bước 1: Tạo thread group, add Recording controller nhưng bình thường

Bước 2: Trong phần workbench vẫn Add=> non test element=> https recode test script

- config như sau: Chọn cổng là 80 hoặc 2020, các thông số để như record trên web

Bước 3: Trên máy mobile chạy ứng dụng

Mở phần setting=> wifi=> nhấn lâu vào wifi đang kết nối=> chọn cài đặt cấu hình nâng cao:

- proxy: manual
- port: 80 hoặc 2020 (tương ứng trong jmeter)
- proxy host: IP của máy tính cài Jmeter
- Ghi script như bình thường

Note: wifi để kết nối từ máy cài Jmeter với mobile có app phải được đặt trong cùng 1 dải mạng

3.8 JDBC request

Để giả lập request truy cập DB, bạn cần phải config các thông tin kết nối đến DB như sau

Bước 1: Click chuột phải vào Thread Group → Add → Config Element → JDBC Connection Configuration.

JDBC Connection Configuration

Name: JDBC Connection Configuration

Comments:

Variable Name Bound to Pool

Variable Name:

Connection Pool Configuration

Max Number of Connections:

Pool Timeout:

Idle Cleanup Interval (ms):

Auto Commit: True

Connection Validation by Pool

Keep-Alive: True

Max Connection age (ms):

Validation Query:

Database Connection Configuration

Database URL:

JDBC Driver class:

Username:

Password:

Bước 2: Nhập các thông tin kết nối đến DB

- Database URL: chuỗi thông tin địa chỉ của DB, tùy thuộc vào loại DB mà cấu trúc của URL khác nhau
- Username/password: Thông tin user kết nối đến DB
- JDBC Driver class: là thư viện dung để kết nối đến DB. Class này cần phải có trong thư mục lib của Jmeter. Nếu trên máy bạn chưa có thì có thể download trên mạng về mà đẩy vào trong thư mục Jmeter/Lib. Việc chọn drivertype được mô tả trong bảng:

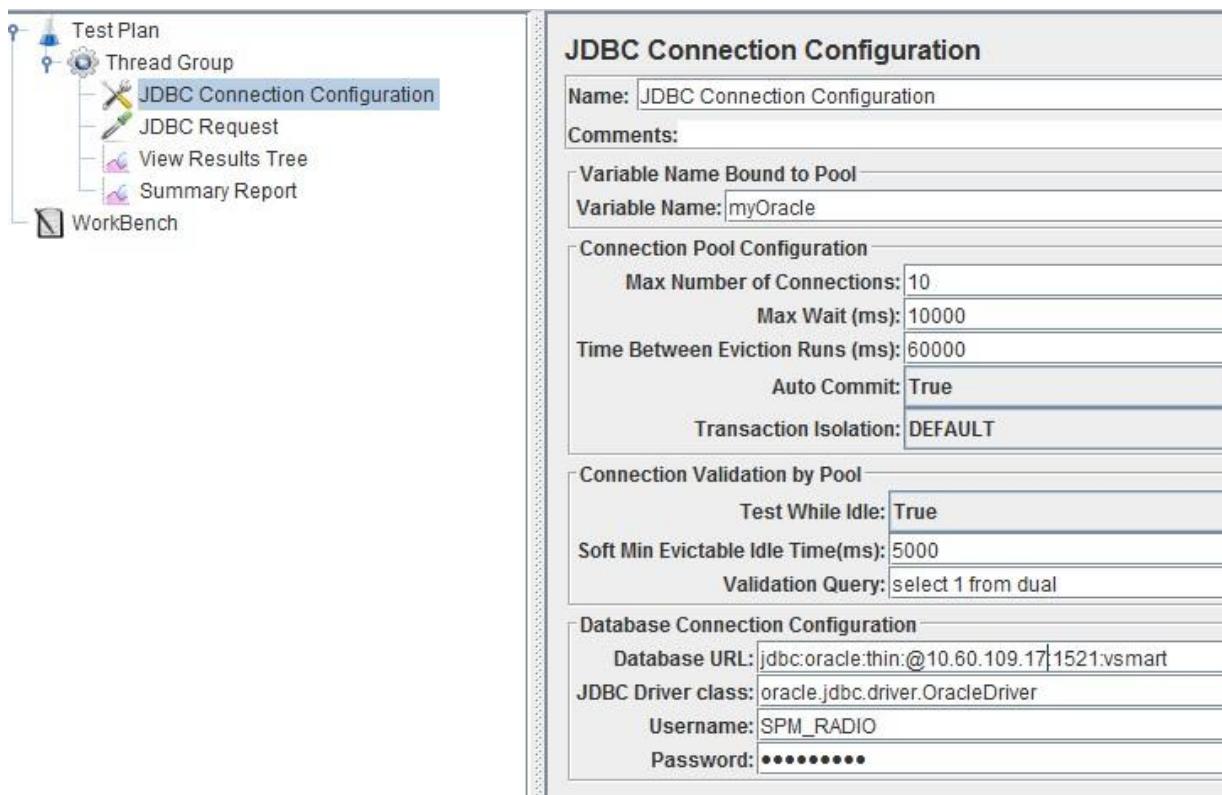
Driver type	Usage	drivertype
Thin Driver	For client-side use without an Oracle	thin
OCI Driver	For client-side use with an Oracle	oci
Server-Side Thin Driver	Same as Thin Driver, but runs inside an Oracle server to access a remote server	thin
Server-Side Internal	Runs inside the target server	kprb

- Ví dụ:
- String url = “`jdbc:oracle:thin:@dbHost:1521:productDB`”
- String url = “`jdbc:oracle:oci:tiger/scott@dbHost:1521:productDB`”
- String url = “`jdbc:oracle:oci:tiger/scott@localhost:1521:productDB`”
- Đăng ký Oracle JDBC driver:

The Oracle JDBC driver class name is **oracle.jdbc.OracleDriver**.

- Thiết lập user/pass cho DB

Ví dụ sau khi tạo kết nối cho Oracle xong:



Bước 3: Add Jdbc Request để thực hiện giả lập các thao tác đến

DB Click chuột phải vào Thread

Group → Add → Sampler → JDBC Request

JDBC Request

Name:	JDBC Request
Comments:	
Variable Name Bound to Pool	
Variable Name:	
SQL Query	
Query Type:	Select Statement
Query:	
Parameter values:	
Parameter types:	
Variable names:	

- ✓ Query Type sau:
 - Update statement: Sử dụng cho các câu update, insert
 - Select statement: Sử dụng cho câu select
 - Callable statement: Sử dụng cho việc thực hiện Store procedure Cách gọi đối với store procedure như sau:
Giả sử bạn có thủ tục tên request_enqueue, khi đó bạn có thể gọi:

```
call request_enqueue('anhdttn','1','1','84972869301','5055','huy tt hanoi',1,'xs ')
```

- Add từ khóa call vào trước tên thủ tục
- Prepare select statement: Sử dụng cho câu select
- Prepare update statement: Sử dụng cho các câu insert, update
- ✓ SQL query: Chứa câu truy vấn thực hiện giả lập giao dịch mà bạn mong muốn Ví dụ: select * from mo_his where id=15s

- ✓ Parameter values: Giá trị các tham số truyền vào trong câu sql(sử dụng cho TH Prepare update statement, Prepare select statement). Các tham số được cách nhau bởi dấu ,

Câu Prepare update/select statement có dạng

sau: Select * from mo_his where id=?

Update mo_his

Set status=?,

msisdn=? Where

id=?

Khi đó giá trị của các trường được truyền vào thông qua Parameter Value, theo thứ tự của các biến. Ví dụ Parameter values=1,84983213076,15 thì giá trị truyền vào trong câu update trên sẽ là : status=1,msisdn=849832130176,id=15

- ✓ Parameter Type: Kiểu dữ liệu của các giá trị parameter values ở trên

- ✓ Variable names: Danh sách tên biến chứa kết quả trả về từ câu select Ví dụ: select msisdn, status from mo_his a where id=15

Variable names=sdt,status

Khi đó từ kết quả trả về từ câu truy vấn msisdn=84983213076, status=1 thì giá trị của biến sdt=84983213076, status=1

JDBC Request

Name:	Update Prices
Comments:	
Variable Name Bound to Pool	
Variable Name:	Pool1
SQL Query	
Query Type:	Prepared Update Statement
Query:	
update Prices set price = ? where item = ?	
Parameter values:	9.99,kettle
Parameter types:	DOUBLE,VARCHAR
Variable names:	
Result variable name:	

JDBC Request

Name: JDBC Request

Comments:

Variable Name Bound to Pool

Variable Name:

SQL Query

Query Type: Update Statement

Query:

```
INSERT INTO SMS_MO
(REQ_ID,MO_TIME,SENDER,RECEIVER,CMD_CODE,MESSAGE,RECV_TIME,CP_CODE)
VALUES (1,now(),841297031852,8162,'TIIN','TIIN bi 100 gui binh luan lan thu nhat
',now(),192)
```

Parameter values:

Parameter types:

Variable names:

Bước 4: Sau khi đã tạo ra các JDBC request, bạn add Listener để check kết quả trả về

3.9 FTP request

a. Chuẩn bị môi trường

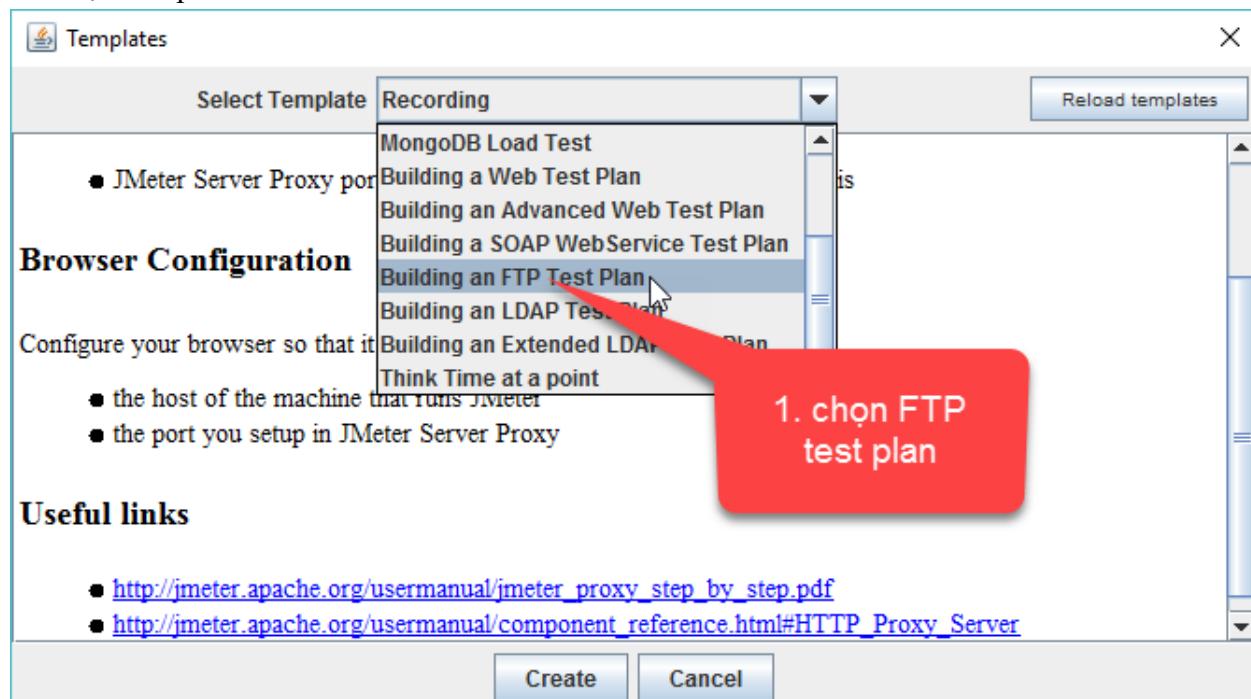
- Cài đặt addon trong Firefox “FireFTP” hoặc FileZilla(<https://filezilla-project.org/>) ==> để lấy FTP request

- Link demo : <https://www.swfwmd.state.fl.us/data/ftp/>

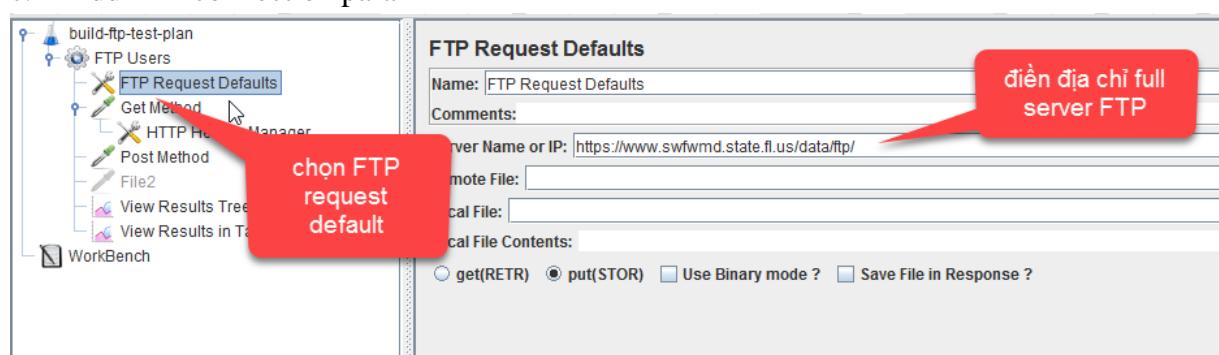
- Server: ftp.swfwmd.state.fl.us

- User Name: Anonymous/email

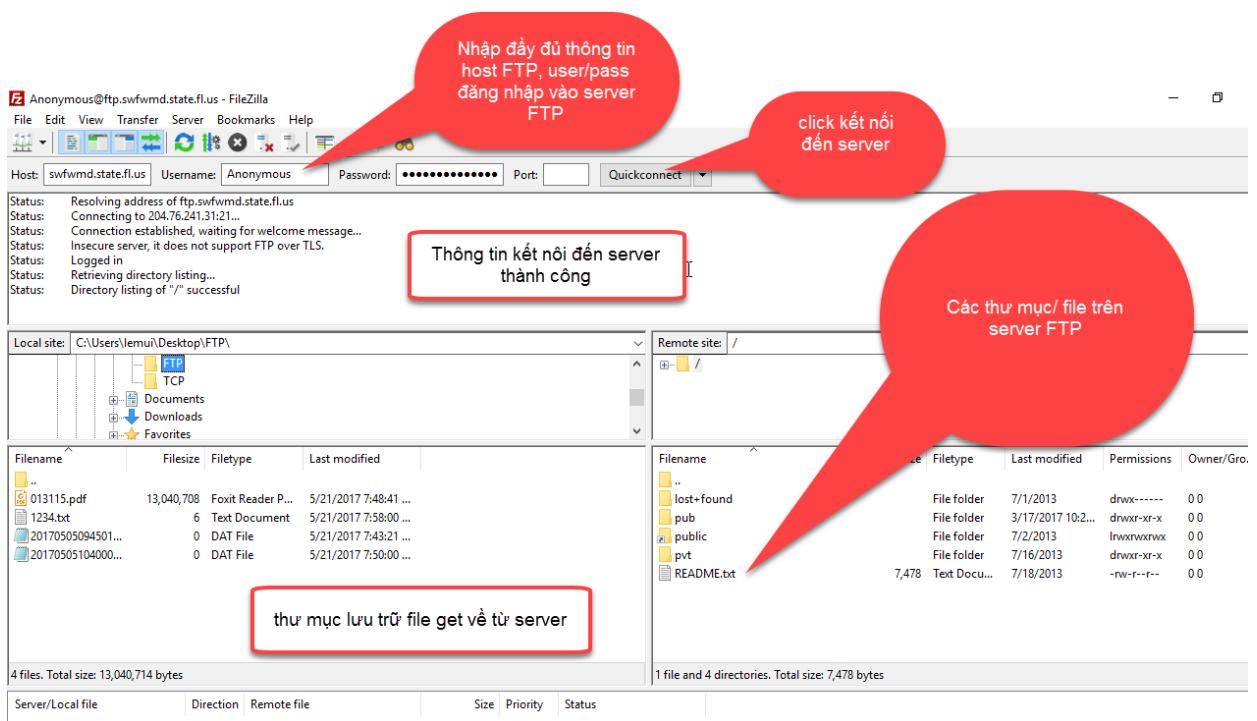
b. Tạo test plan FTP



c. Add FTP connection param

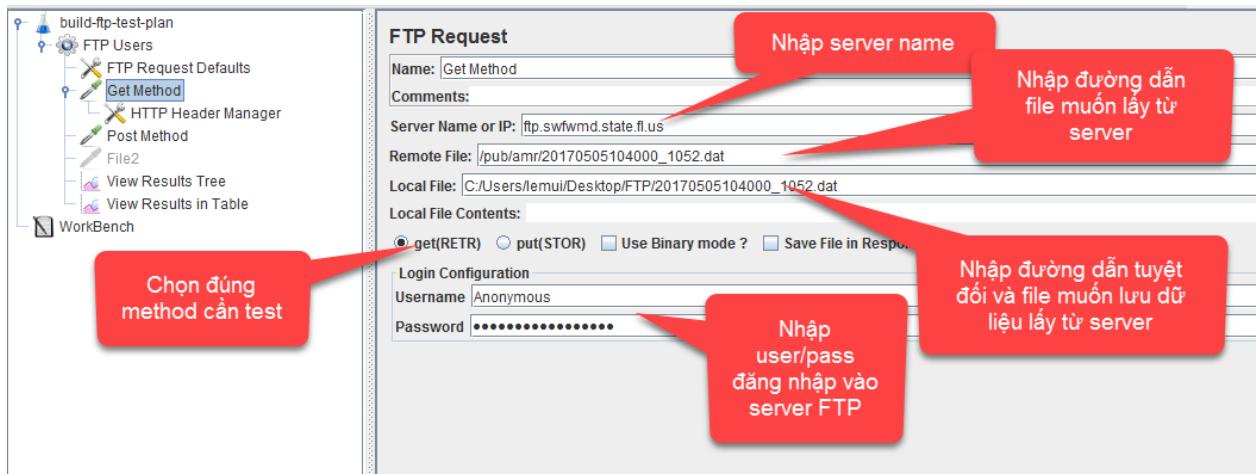


d. Tìm file dữ liệu cần lấy bằng FileZilla



e. Test with GET protocol

Với giao thức POST các thông tin làm tương tự, chú ý: chọn method POST



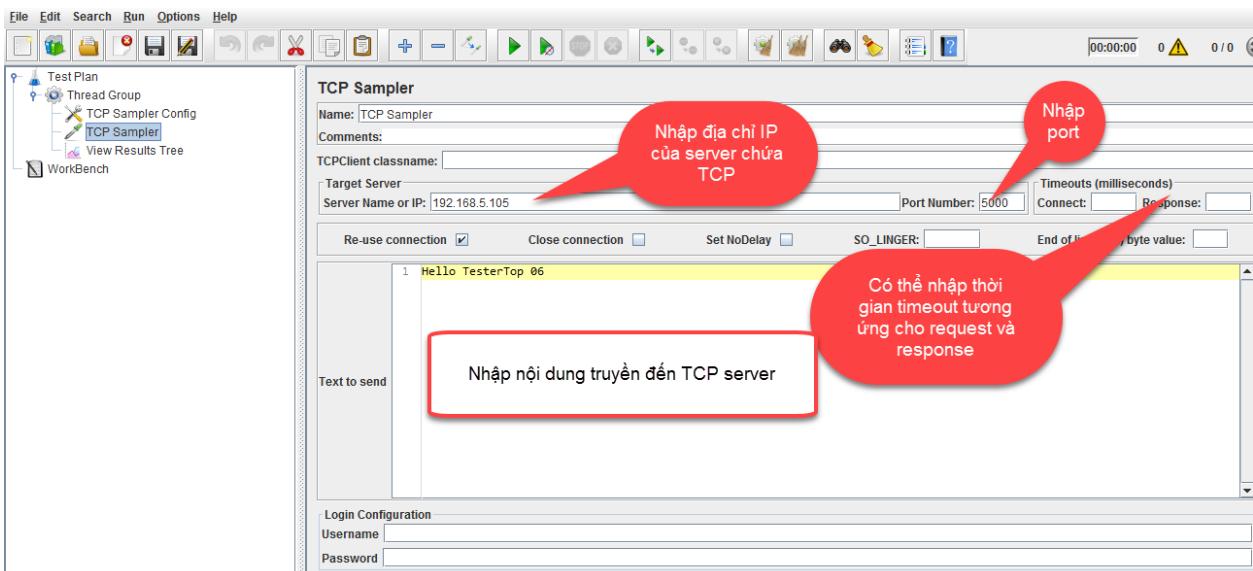
Chú ý:

- Remote File: luôn luôn điền đường dẫn đến file được lấy từ server
- Local file: đường dẫn tuyệt đối đến file muốn lưu nội dung được lấy từ file trên server ftp
- Tên file và thư mục lưu trữ file cần chú ý cách đặt tên (không sử dụng kí tự đặc biệt)

3.10 TCP request

Mục đích: Dùng để giả lập các request TCP

- Bước 1: Click Test Plan --> Add --> Thread Group
- Bước 2: Click chuột phải vào Thread Group → Add → Config Element --> TCP Sampler config
- Bước 3: Click chuột phải vào Thread Group → Add → Sampler--> TCP Sampler
- Bước 4: Nhập đầy đủ thông tin server TCP

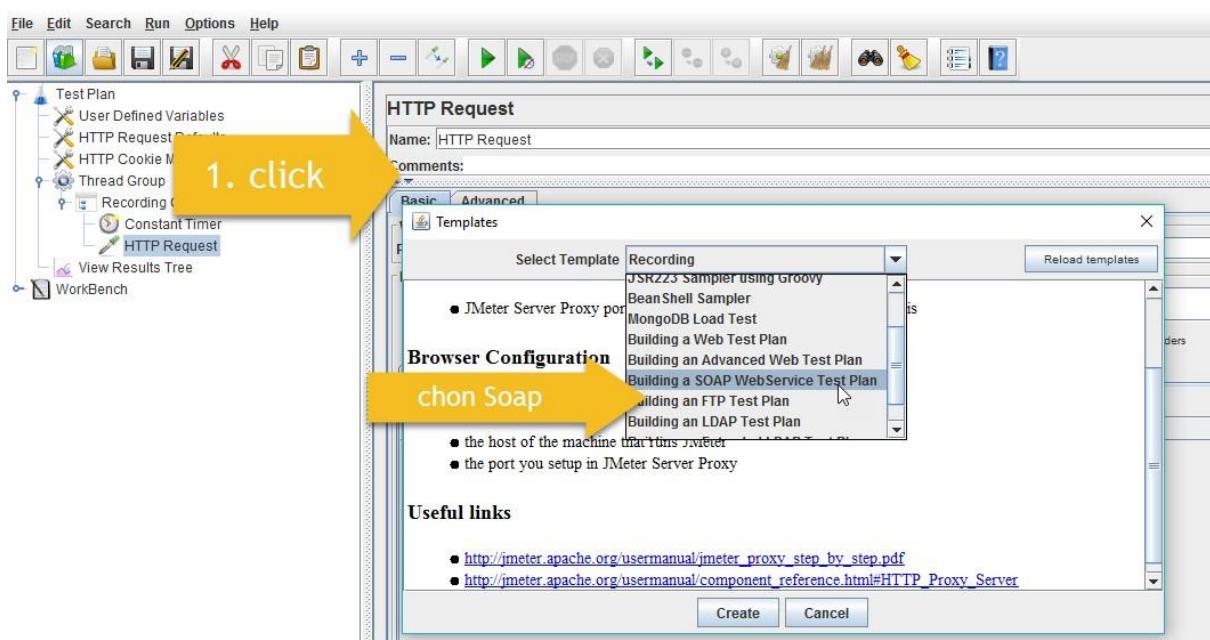


3.11 SOAP/XML-RPC request

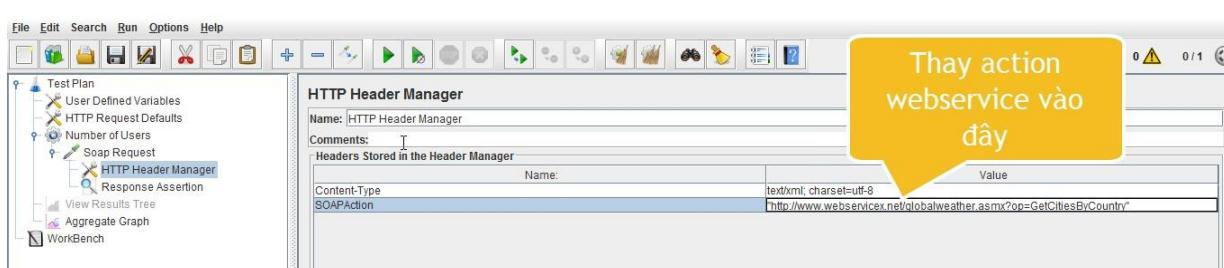
Mục đích: Dùng để giả lập các request đến các Webservice, SOAP/XML-RPC

Bước 1: Click chuột phải vào Thread Group → Add → Sampler → SOAP/XML-RPC Request (với các version 3.0 trở về trước).

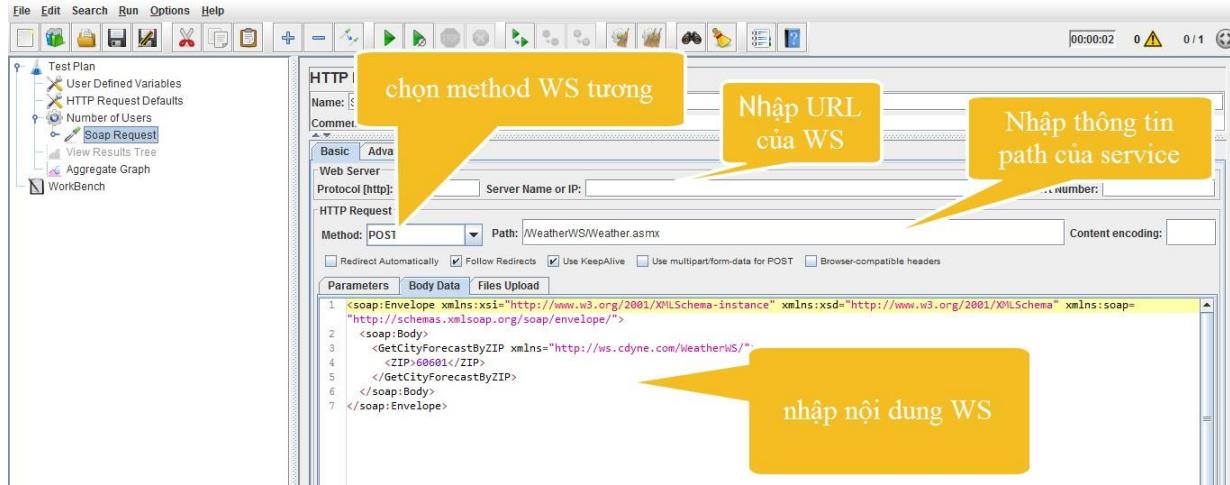
Với các version sau 3.0, thực hiện như hình



Bước 2: Nhập thông tin cho header



Bước 3: Nhập thông tin URL và thông tin webservice (WS) vào ô textbox tương ứng.



Ví dụ có webservice nhu sau:

Link: <http://www.webservicex.net/globalweather.asmx?op=GetCitiesByCountry>

Thông tin webservice

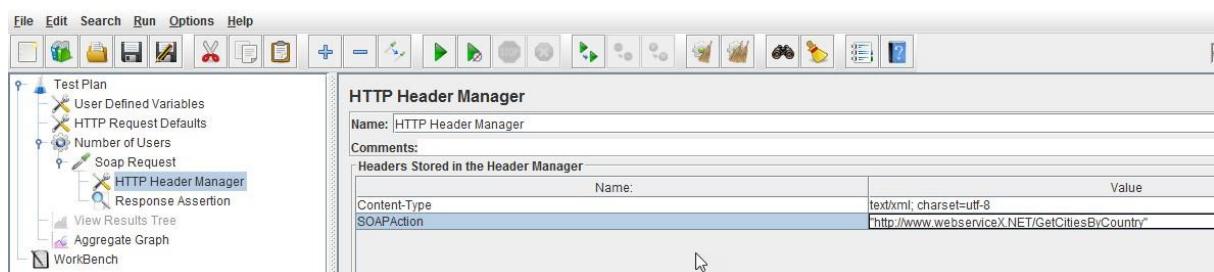
```
The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

POST /globalweather.asmx HTTP/1.1
Host: www.webservicex.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.webserviceX.NET/GetCitiesByCountry"

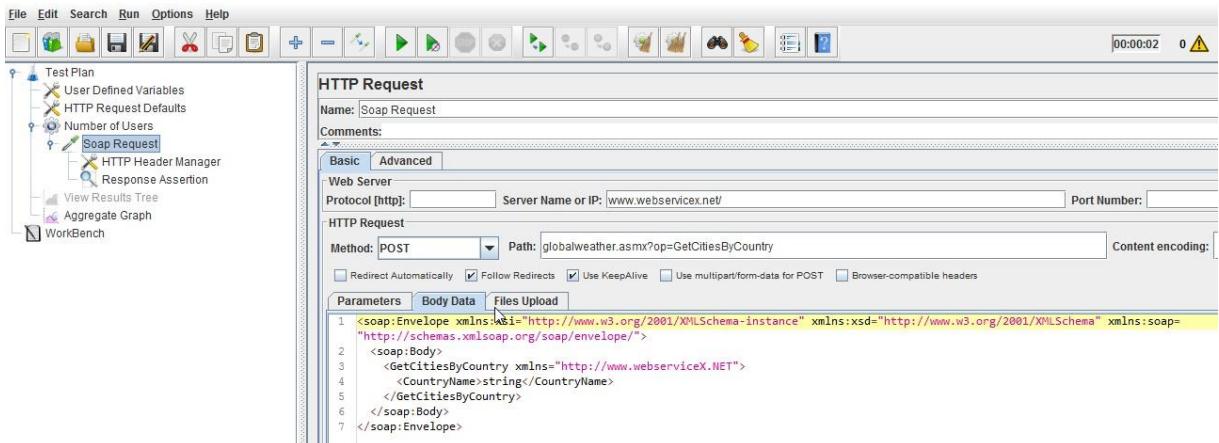
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope">
  <soap:Body>
    <GetCitiesByCountry xmlns="http://www.webserviceX.NET">
      <CountryName>string</CountryName>
    </GetCitiesByCountry>
  </soap:Body>
</soap:Envelope>
```

Sau khi thực hiện các bước trên được hình nhu dưới

- Header



- Request

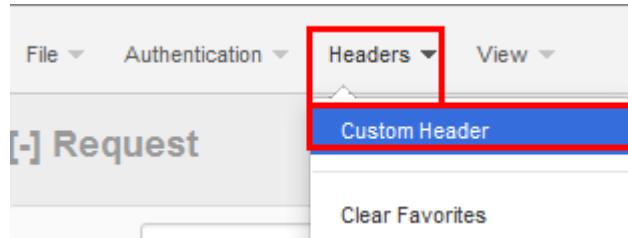


Bước 5. Thêm Listener và run ➔ view kết quả trả về (giống như làm trên web)

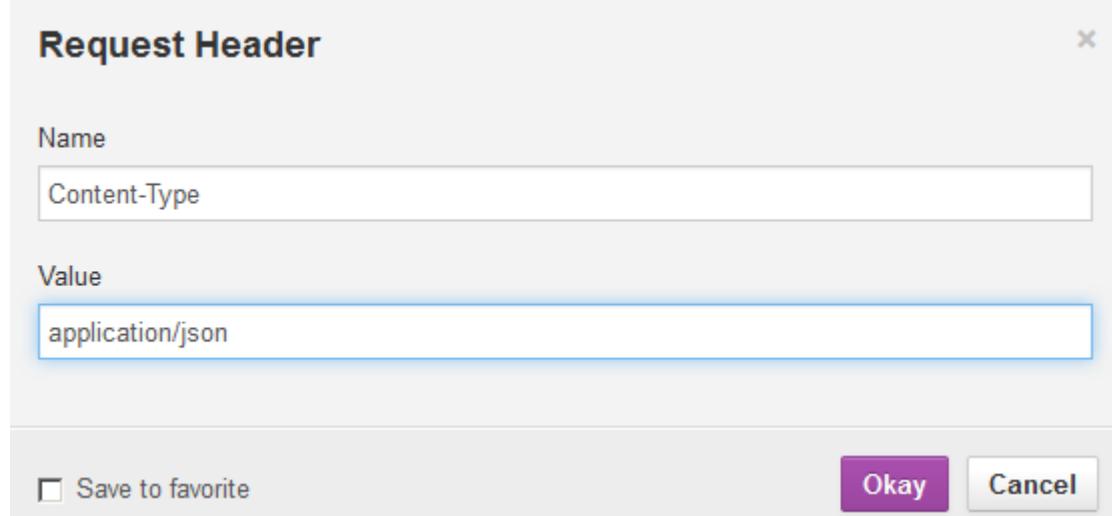
3.12 Rest webservice

3.12.1 Lấy thông tin và kiểm tra rest service (sử dụng FireFox hoặc Chrome)

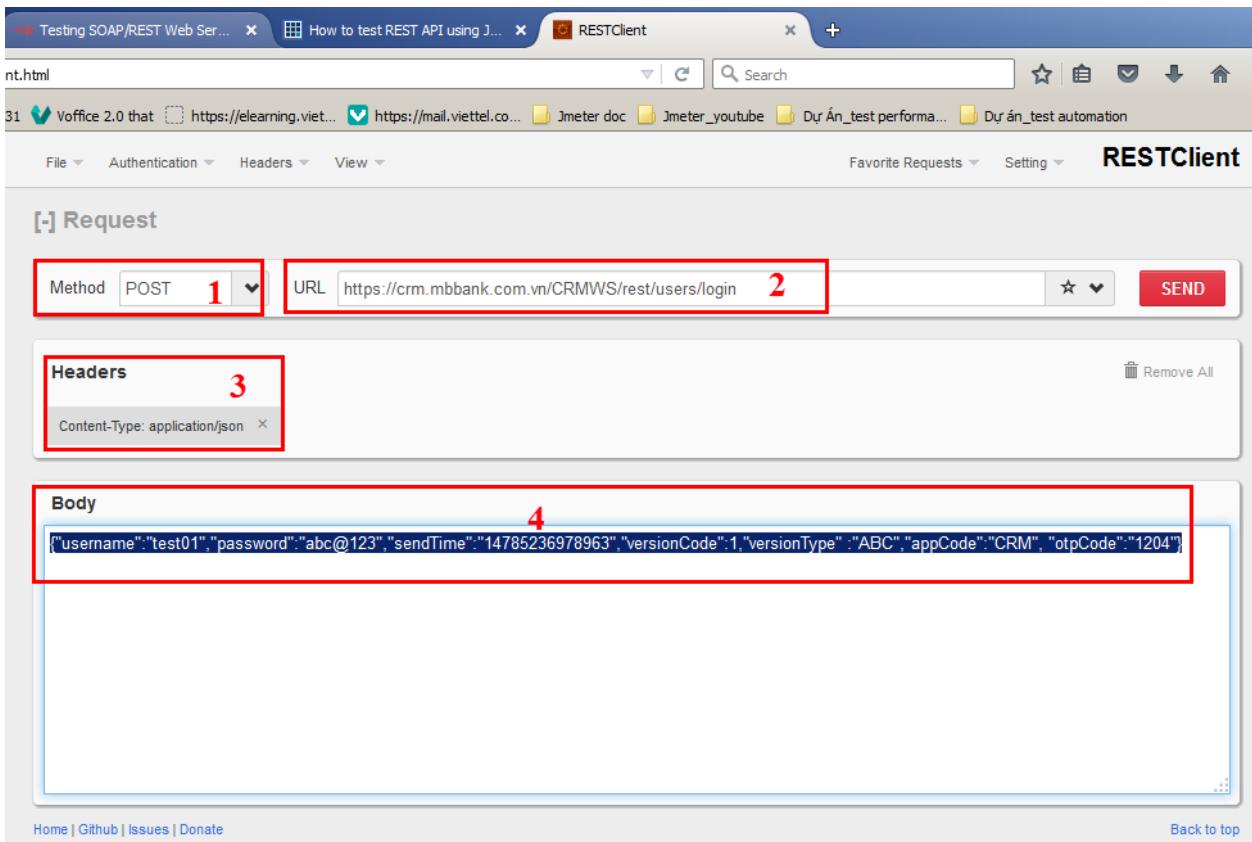
- B1: cài Add on RESTClient trên trình duyệt FireFox hoặc Advanced REST client trên Chrome
- B2: Ví dụ chạy REST web service bằng RESTClient trên FireFox: Áp dụng ví dụ chạy REST web service của hệ thống CRM_MB.
- Cấu hình Request Header:
- Vào Menu Headers -> chọn Custom Header (như ảnh dưới)



- Ứng dụng sẽ mở ra màn hình cho phép Custom Header => yêu cầu nhập Name và Value của Header cần cấu hình. Mỗi hệ thống sử dụng Name và Value khác nhau, hỏi đội dự án để biết Name và Value của Request Header.



- Nhập dữ liệu vào form rồi nhấn nút **SEND** để thực hiện REST web service.



1. Nhập phương thức cho Web Service, ví dụ: method=POST
2. Nhập URL: ví dụ: URL = <https://crm.mbbank.com.vn/CRMWS/rest/users/login>
3. Phần Header: sẽ hiển thị Name và Value mà đã nhập lúc trước
4. Phần Body: nhập dữ liệu đầu vào, ví dụ: đầu vào để login

3.12.2 Cách chạy REST web service trên Jmeter

- Các thành phần cần add để tạo 1 test plan Jmeter cho REST:
 - Add Test Plan, Thread Group, HTTP Header Manager, HTTP Request Defaults, Sampler HTTP Request for REST
 - Add CVS Data Set Config
 - Add Response Assertion
 - Add Listener (View Result Tree, Aggregated Report)
 - jp@gc-Response Times over Time
 - Run the Test Plan
 - View Output

Dưới đây là các phần bắt buộc phải cấu hình để có thể chạy được REST:

1. Add “HTTP Header Manager” Config element

HTTP Header Manager quyết định nội dung HTTP header trong các requests và thường dùng để thêm hoặc ghi đè lên HTTP request headers. Dưới đây là một số cách cấu hình tham số phổ biến cho REST

Accept: */*

Content-Type: application/json; charset=UTF-8; text/xml

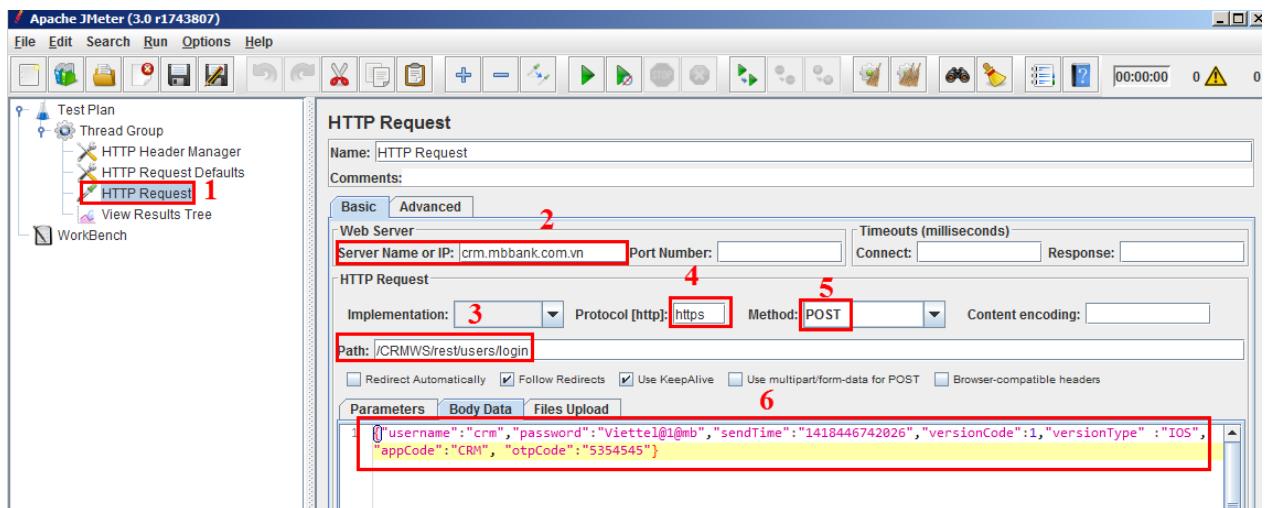
2. Add “HTTP Request Defaults” Config element.

Nếu kịch bản kiểm thử chứa nhiều requests, các requests có nhiều tham số giống nhau thì ta tận dụng cấu hình những tham số chung ấy trong HTTP Request Defaults luôn để không phải viết lại nhiều lần trên các HTTP Request Sampler.

Ví dụ: Server Name or IP, Protocol, Method, ...

3. Add “HTTP Request” Sampler element.

Cấu hình các tham số trong HTTP Request như hình dưới:



(2): Server Name or IP: nhập thông tin server
(3): Path: nhập path

(4): Protocol: thường sẽ nhập HTTP hoặc HTTPS

(5): Method: REST thường sử dụng các phương thức GET, PUT, POST, DELETE, ...

(6): Nhập dữ liệu đầu vào, tùy thuộc vào định dạng dữ liệu đầu vào mà ta sẽ nhập dữ liệu vào tab Body Data hay Parameters

Ví dụ: nếu đầu vào của REST có format application/json -> dữ liệu đầu vào nhập trong tab Body Data, nếu đầu vào ở dạng text/xml -> dữ liệu đầu vào nhập trong tab Parameters.

(những thành phần khác thực hiện thêm vào TestPlan như các dự án bình thường)

3.13 Running jmeter non-gui mode

- Vì sao nên đầy tải ở chế độ Non-GUI mode, mà không nên đầy tải ở chế độ GUI mode:
 1. GUI mode (chạy chế độ khi mở jmeter lên) --> tốn tài nguyên (vì phải gen giao diện)-> nên request được gửi đi "lát" --> gửi chậm (gửi được ít request) --> server nó xử lý nhanh --> Response Time thấp
 2. non-GUI mode --> ít tốn tài nguyên (vì không phải gen giao diện) --> request được gửi đi đúng với tốc độ của nó --> gửi nhanh gửi liên tục ko bị delay gì hết --> server nó bị "tấn công" ồ ạt như vậy --> xử lý ko kịp --> lỗi --> Response Time cao
- Thực tế khi đầy tải trên tool, thường đầy trên 1 máy hoặc share tải trên 1 vài máy, rồi giả lập số user lên tới vài trăm hoặc vài nghìn users.
 - GUI mode => máy đầy tải phải sinh và xử lý request, generate giao diện, show kết quả test cho mỗi user => tốn tài nguyên của client => khả năng đầy request lên server chậm, số lượng thấp => server xử lý từ từ (không đúng với tốc độ thực tế của nó).
 - Non-GUI mode => máy đầy tải chỉ phải sinh và xử lý request => không tốn tài nguyên máy

client => khả năng đẩy request lên server nhanh, số lượng nhiều => server sẽ được hoạt động đúng với tốc độ của nó.

- Ví dụ:

- GUI nó ngôn tài nguyên của máy => RAM có 4GB => nó ngôn hết 3GB (để generate giao diện và show các sampler listenner) => chỉ còn 1GB cho việc sinh và xử lý các requests đó => nên nó chậm
- Non-GUI, nó ko tồn tài nguyên => máy 4GB thì nó trống ~3GB => 3GB đó dùng cho việc generate requests, xử lý requests => thì các request sẽ được sinh ra nhanh hơn, xử lý nhanh hơn.

3.13.1 Non-GUI mode

a. Các lệnh tùy chọn

- Sau đây là các lệnh tùy chọn.
- -n: lệnh này thể hiện Jmeter chạy ở chế độ non-gui mode
- -t: tên JMX file chứa test plan
- -l: tên JTL file để ghi log kết quả test
- -j [name of JMeter run log file].
- -r: chạy bài test trên server bằng cách remote đến “remote_host”
- -R: danh sách server sẽ remote
- -g: đường dẫn để lưu file
- -e: generate báo cáo sau khi đẩy tải
- -o: tên folder sẽ lưu báo cáo sau khi đẩy tải, folder này phải không tồn tại hoặc trống
- Nếu scripts cần chạy qua proxy, cần phải cấu hình thêm 2 lệnh dưới:
- -H: proxy server hostname or ip address
- -P: proxy server port

b. Ví dụ

- Mở màn hình cmd -> trỏ đến thư mục %JMETER_HOME%/bin -> chạy dòng lệnh dưới

```
jmeter -n -t my_test.jmx -l log.jtl -H proxyvitm02 -P  
3128
```

Giải thích dòng lệnh trên:

- -n: chạy ở chế độ non-GUI mode
- -t my_test.jmx: chạy bài test my_test, file **my_test.jmx** được đặt trong thư mục %JMETER_HOME%/bin
- -l log.jtl: kết quả test hiệu năng của các sampler trong Listeners sẽ được lưu ở file log.jtl, file này được lưu trong thư mục %JMETER_HOME%/bin
- -H proxyvitm02: proxyvitm02 là tên proxy mà ứng dụng cần chạy qua (nếu app không cần by pass proxy thì không cần lệnh này).
- -P 3128: 3128 là port của proxyvitm02 (nếu app không cần by pass proxy thì không cần lệnh này).

c. Cách xem kết quả sau khi đẩy tải

- Xem Sampler Listener (Summary Report, View Results in Table,): mở file %JMETER_HOME%/bin /jmeter.bat -> add các Sampler Listener muốn xem -> browser file log.jtl vừa ghi ở lệnh trên (xem ảnh phía dưới).

Mở file log.jtl để xem kết quả test sau khi test ở chế độ non-GUI mode

kết quả test

Label	# Samples	Average	Min	Max	Std Dev	Error %	Throughput	KB/sec	Avg. Bytes
Login	5000	86	65	2228	77.31	0.00%	5.0/sec	1.12	229.0
Lấy thông tin khách hàng	5000	11	7	690	10.42	0.00%	5.0/sec	31.38	6421.0
Lấy thông tin điểm đến	5000	136082	4	170846	41713.81	24.95%	29.0/sec	17.43	616.4
TOTAL	15000	45393	4	170846	68500.27	8.32%	12.8/sec	30.26	2422.1

Mở file log.jtl để xem kết quả sau khi test ở chế độ non-GUI mode

kết quả test

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Latency	Connect Time(ms)
1	08:53:08.000	Login-1-790	Login	2306	✓	229	2305	0
2	08:53:07.000	Login-1-383	Login	2843	✓	229	2843	0
3	08:53:07.000	Login-1-281	Login	2926	✓	229	2924	0
4	08:53:07.000	Login-1-283	Login	3018	✓	229	3012	0
5	08:53:07.000	Login-1-243	Login	3107	✓	229	3107	0
6	08:53:07.000	Login-1-361	Login	2991	✓	229	2991	0
7	08:53:07.000	Login-1-279	Login	3049	✓	229	3048	0
8	08:53:07.000	Login-1-300	Login	3079	✓	229	3079	0
9	08:53:07.000	Login-1-124	Login	3177	✓	229	3177	0
10	08:53:08.000	Login-1-777	Login	2923	✓	229	2923	0
11	08:53:07.000	Login-1-168	Login	3575	✓	229	3574	0
12	08:53:07.000	Login-1-95	Login	3391	✓	229	3389	0
13	08:53:07.000	Login-1-296	Login	3460	✓	229	3460	0
14	08:53:07.000	Login-1-360	Login	3438	✓	229	3438	0
15	08:53:07.000	Login-1-360	Login	3428	✓	229	3428	0
16	08:53:07.000	Login-1-288	Login	3463	✓	229	3463	0
17	08:53:07.000	Login-1-220	Login	3605	✓	229	3604	0
18	08:53:07.000	Login-1-115	Login	4691	✓	229	4691	0

- Muốn lưu kết quả của “jp@gc - PerfMon Metrics Collector”, trong lúc chạy tải phải mở Sampler Listener “jp@gc - PerfMon Metrics Collector” và đặt tên file .csv (xem ảnh dưới):

Mở file log.jtl để xem kết quả sau khi test ở chế độ non-GUI mode

trong lúc chạy tải ở chế độ non-GUI mode => test plan nhỏ enable Sampler Listener (1) và đặt tên file .csv (2). Disable tất cả các sampler khác.

Host/IP	Port	Metric to collect	Metric parameter (see help)
10.60.110.20	8899	CPU	
10.60.110.20	8899	Memory	
10.60.110.131	8566	CPU	
10.60.110.131	8566	Memory	

- Sau khi đầy tải xong, kết quả Resource Monitor sẽ được lưu trong file .CSV và lưu trong %JMETER_HOME%/bin/test.csv.

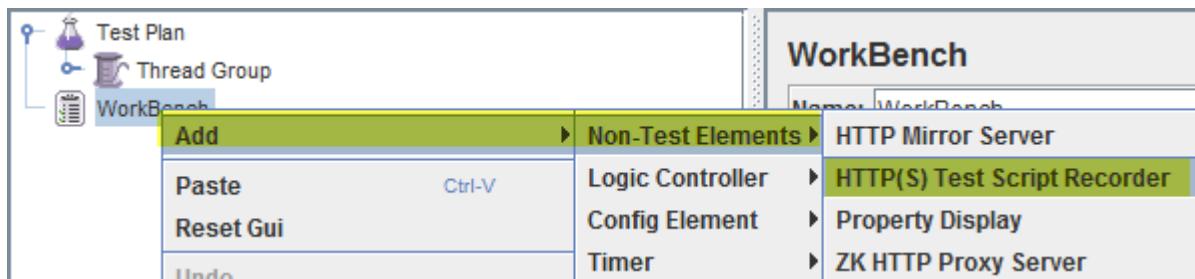
3.14 ZK framework

3.14.1 Record script

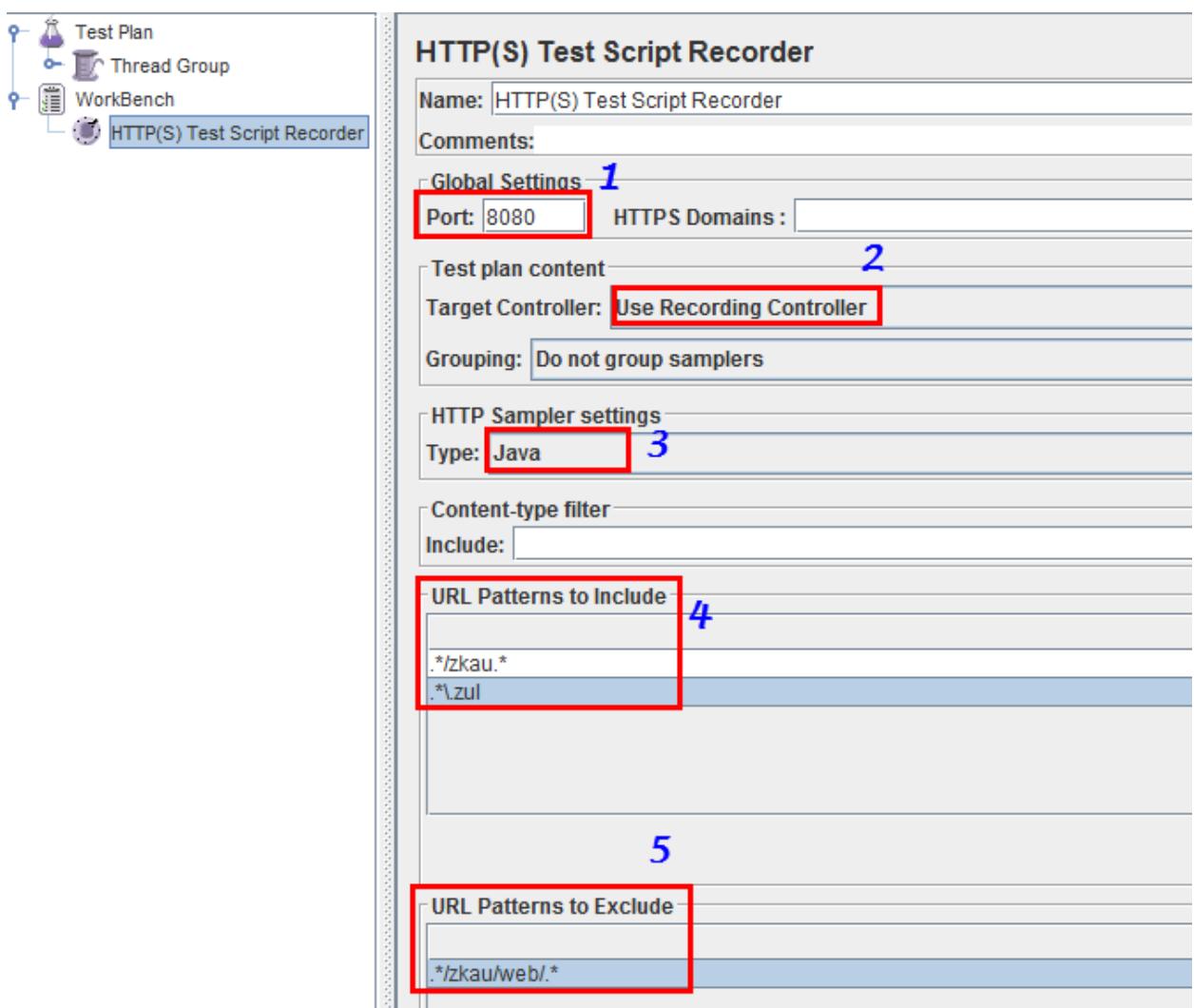
Có 2 cách record request từ ZK fw (ZK framework):

3.14.1.1 Record thông thường:

- Bước 1: chọn cách thức record nhu hình dưới:



- Bước 2: màn hình HTTP(S) Test Script Recorder



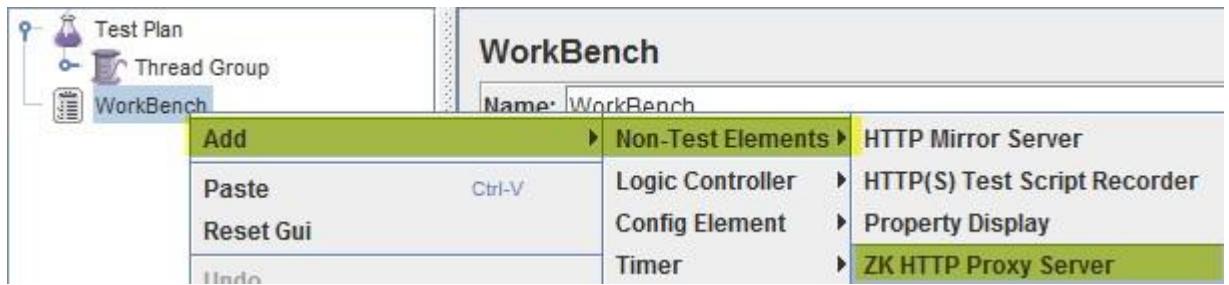
- (1): Đặt cổng = 8080 hoặc 9999
- (2): Chọn đúng Recording controller, chính là recording mà đã tạo trên Thread group của Test plan.
- (3): Chọn type=Java
- (4): Có thể thêm một số đuôi path để giới hạn việc record request: nếu thêm các đuôi mở rộng thì Jmeter sẽ chỉ record những request có đuôi này mà không record các đuôi mở rộng khác.
- (5): tương tự như ý (4).

Note: Nếu record ZK fw theo cách thông thường người dùng phải chú ý:

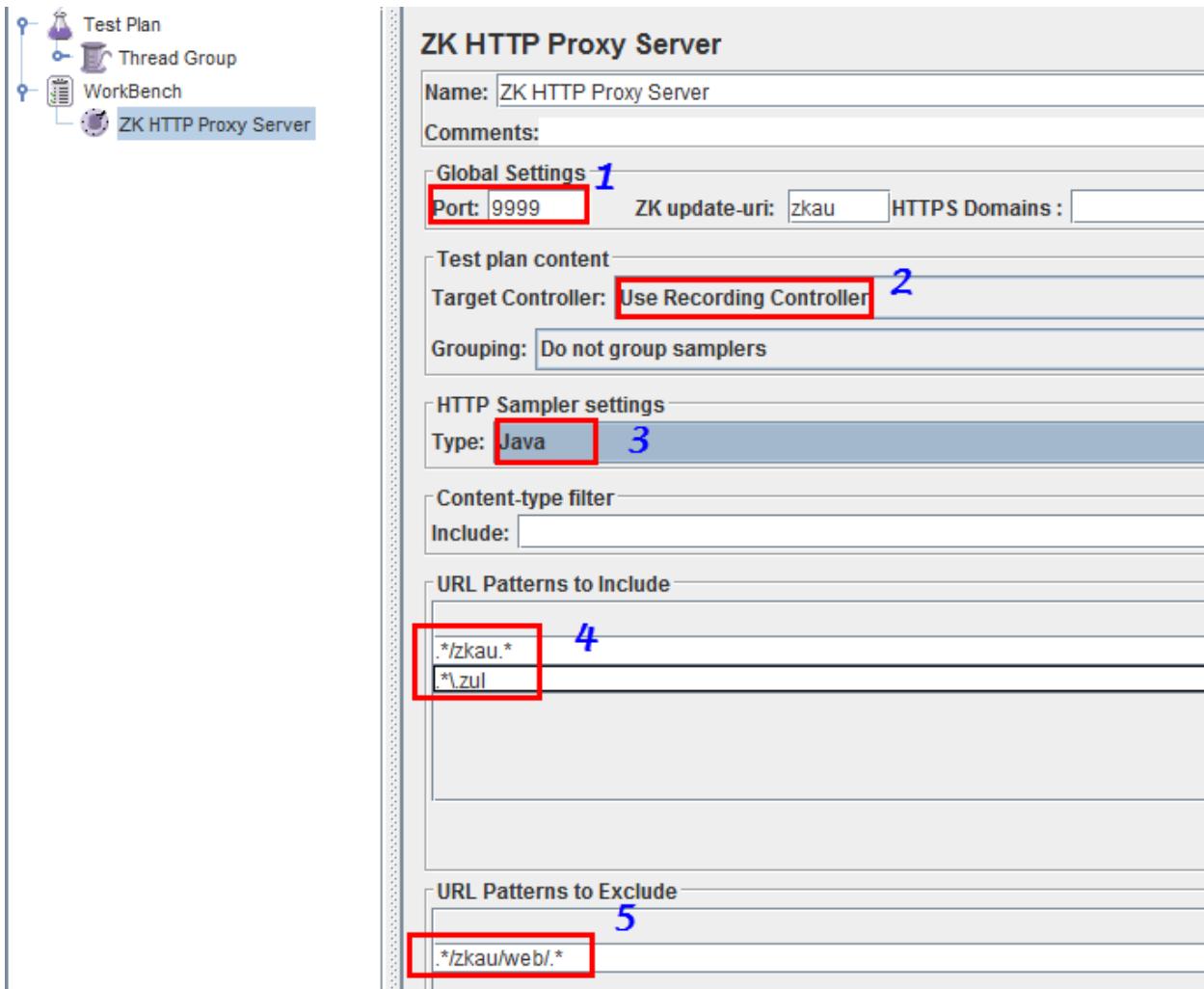
- Không được thêm biến vào User Defined Variables trước khi record, mà phải thêm sau khi record.
- Phải start HTTP(S) Test Script Recorder trước khi thao tác request trên web
- Phải cấu hình proxy trên trình duyệt: giống với cổng đã cấu hình trong HTTP(S) Test Script Recorder chính là ý (1).

3.14.1.2 Record bằng thư viện hỗ trợ bởi ZK

- Bước 1: Tải về zk-jmeter-plugin-0.8.0.jar
- Bước 2: Đặt file .jar vừa tải vào ..\lib\ext của Jmeter
- Bước 3: Chọn cách thức record như hình dưới:



- Bước 4: Màn hình ZK HTTP Proxy Server



(1): Mặc định cổng = 9999

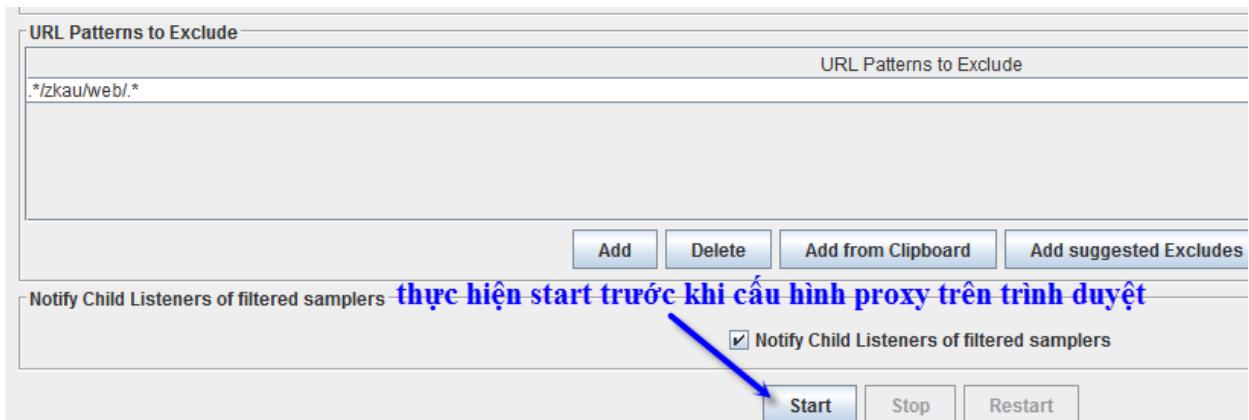
(2): Chọn đúng Recording controller, chính là recording mà đã tạo trên Thread group của Test plan. (3): Chọn type=Java

(4): Có thể thêm một số đuôi path để giới hạn việc record request: nếu thêm các đuôi mở rộng thì Jmeter sẽ chỉ record những request có đuôi này mà không record các đuôi mở rộng khác.

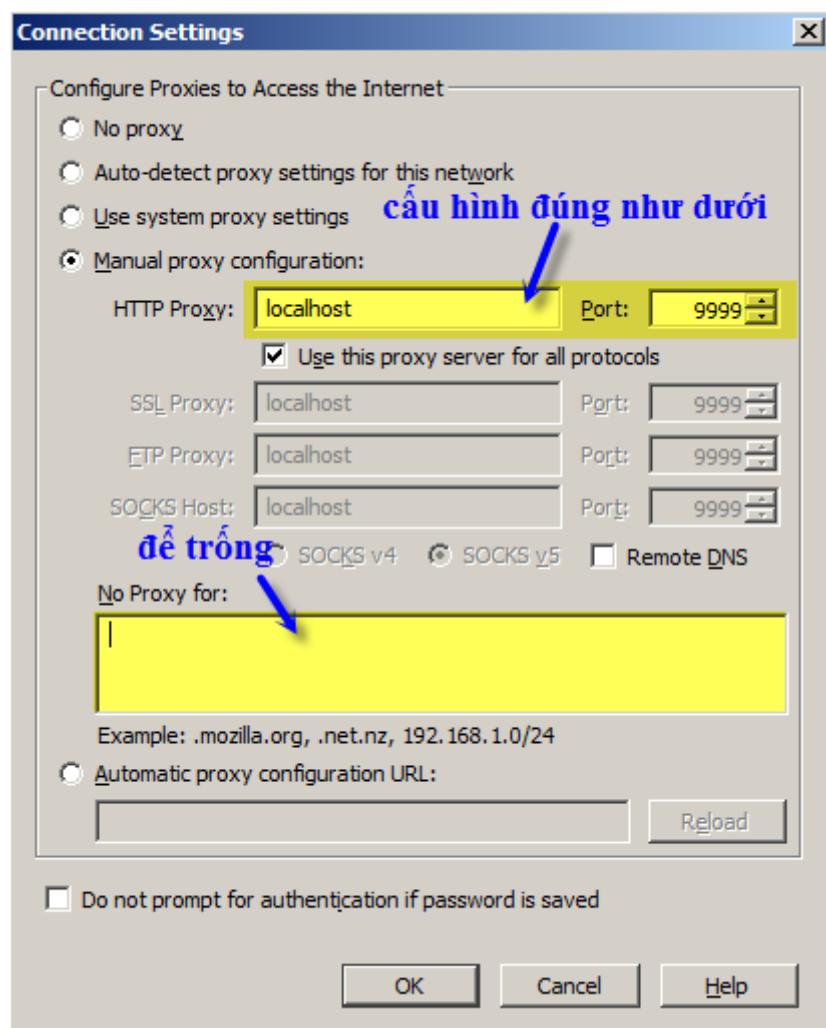
(5): tương tự như ý (4).

3.14.2 Các bước record requests

- Bước 1: Chọn cách thức record request
- Cấu hình các thông số theo từng cách thức record
- Bước 3: Load trang đăng nhập của ứng dụng cần record
- Bước 4: Start HTTP(S) Test Script Recorder (ZK HTTP Proxy Server)



- Bước 5: Cấu hình proxy trên trình duyệt chạy ứng dụng, ví dụ trên trình duyệt Firefox:
- Sửa proxy lại = localhost:9999



➔ Load lại trang đăng nhập của ứng dụng cần record

- Bước 6: Thao tác trên ứng dụng và kiểm tra các request đã được record trên Jmeter trong Recording Controller đã thêm trong Test Plan.

3.14.3 Tham biến các giá trị thay đổi: dtid, uuid, item...

Bước 1: Định nghĩa biến cho những biến nào biến động và giá trị trên User Defined Variables

- Thực tế trong các request có một số biến thay đổi về giá trị sau mỗi lần record, nên phải định nghĩa giá trị cho biến này và thiết lập biến trong các request: trong app này có biến dtid => định nghĩa biến này dtid=0
- Không được đặt biến dtid=0 trước khi record (phải đặt sau khi record request)
- Với những request có đuôi .zul => thêm biến dtid = \${ intSum(\${dtid},1,dtid) }
- Với những request có đuôi .zkau => thiết lập giá trị cho tham số dtid=\${dtid}
- Không để lệnh rmDesktop trong danh sách các tham số của request, nếu có tham số dạng: cmd_...=rmDesktop => thay bằng cmd_...=onSelect

Định nghĩa biến dtid:

Name:	Value:
dtid	0

Thiết lập tham số cho request zul:

Name:	Value:
dtid	\${ intSum(\${dtid},1,dtid) }

Thiết lập tham số cho request zkau:

Name:	Value:
dtid	\${dtid}
cmd_0	onChange
uuid_0	t_494
data_0	{"value":"123-thuytest","start":12}

Sửa lệnh rmDesktop (nếu cần):

Name:	Value	Encode?	In
dtid	<code>\$(dtid)</code>		
cmd_0	<code>\$(dtid)rmDesktop</code>		
opt_0	i		

Bước 2: Thêm assertion và chạy script Bước 3:Xem kết quả test hiệu năng

Chú ý:

Có thể làm theo hướng dẫn trên link:

[## 3.15 Cấu hình jmeter plugin để lấy cpu, ram, ...](https://www.zkoss.org/wiki/Small_Talks/2012/January/Execute_a>Loading_or_Performance_Test_on_ZK_using_JMeter</p>
</div>
<div data-bbox=)

Cơ chế của việc lấy thông số RAM, CPU:

- Jmeter có hỗ trợ plugin AgentServer, AgentServer có nhiệm vụ thu thập thông tin của server
=> vậy nên phải đặt AgentServer trên server chịu tải.
- Jmeter có hỗ trợ plugin jp@gc - PerfMon Metrics Collector trong Listenrs, jp@gc - PerfMon Metrics Collector kết hợp với AgentServer sẽ hiện thị đồ thị và lưu trữ các thông số của server trên Jmeter khi mà Jmeter chạy.

3.15.1 Cách cấu hình

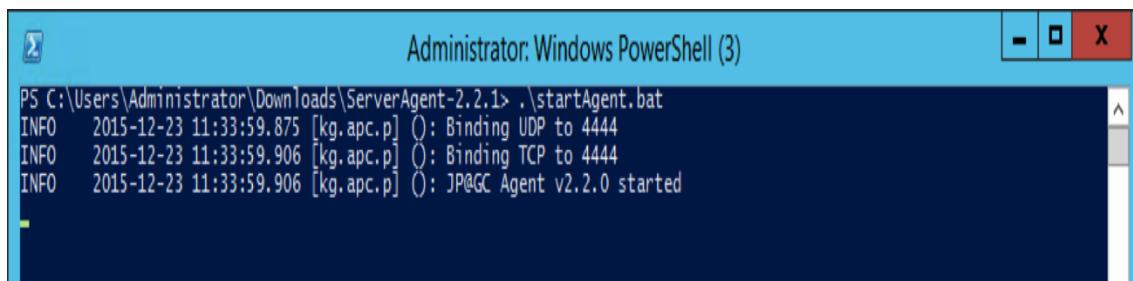
3.15.1.1 Download plugin

- Cấu hình [Metrics Collector Listener](#), tải JMeterPlugins-Standard.jar từ <http://jmeterplugins.com/downloads/index.html>, giải nén -> copy tất cả các file trong ...lib\ext rồi đặt vào thư mục ...lib\ext của Jmeter, mục đích của việc này để có jp@gc - PerfMon Metrics Collector trong Listeners.
- Tải [ServerAgent-xx.xx.xx.zip](#) từ <http://jmeterplugins.com/downloads/index.html>, tải nén rồi copy thư mục ServerAgent-xx.xx.xx lên server chịu tải.
- Server Agent là một ứng dụng dòng lệnh của Java, vì vậy bạn sẽ cần Java Runtime Environment (JRE). Để xuất, nên sử dụng 64-bit Server JRE or JDK (Java SE Development Kit) của phiên bản Java mới nhất cho Jmeter. Có thể download JRE / JDK from Oracle's [Java download page](#). (nếu Jmeter đã cài và khi chạy Server Agent không có vấn đề gì thì không cần quan tâm tới chú ý này).
- Sau khi giải nén ServerAgent-xx.xx.xx.zip ta sẽ được cấu trúc như sau:
 - **lib:** A folder containing dependency libraries (mostly [SIGAR](#))
 - **CMDRunner.jar:** The same .jar file which comes with JMeter Plugins (this lives under /lib/ext folder of JMeter installation). It's used to launch the Server Agent
 - **LICENSE:** The Apache License file

- **ServerAgent.jar:** The Main Server Agent library
- **startAgent.bat:** The Server Agent Launch script for Windows
- **startAgent.sh:** The Server Agent Launch script for Linux/MacOSX/Unix

3.15.1.2 Cấu hình với port mặc định 4444

- Cấu hình Server Agent: bạn có thể unzip [ServerAgent-xx.xx.xx.zip](#) ở bất kì chỗ nào trên server sau đó chạy startAgent.sh đối với Unix, hoặc chạy startAgent.bat đối với Windows. Server Agent thực hiện thu thập các số liệu của server sau đó gửi chúng qua giao thức TCP or UDP trả lại cho Jmeter, có thể xem các số liệu đó thông qua Metrics Collector Listener.
- Khi start Server Agent, bạn sẽ thấy 3 dòng sau:



```

Administrator: Windows PowerShell (3)

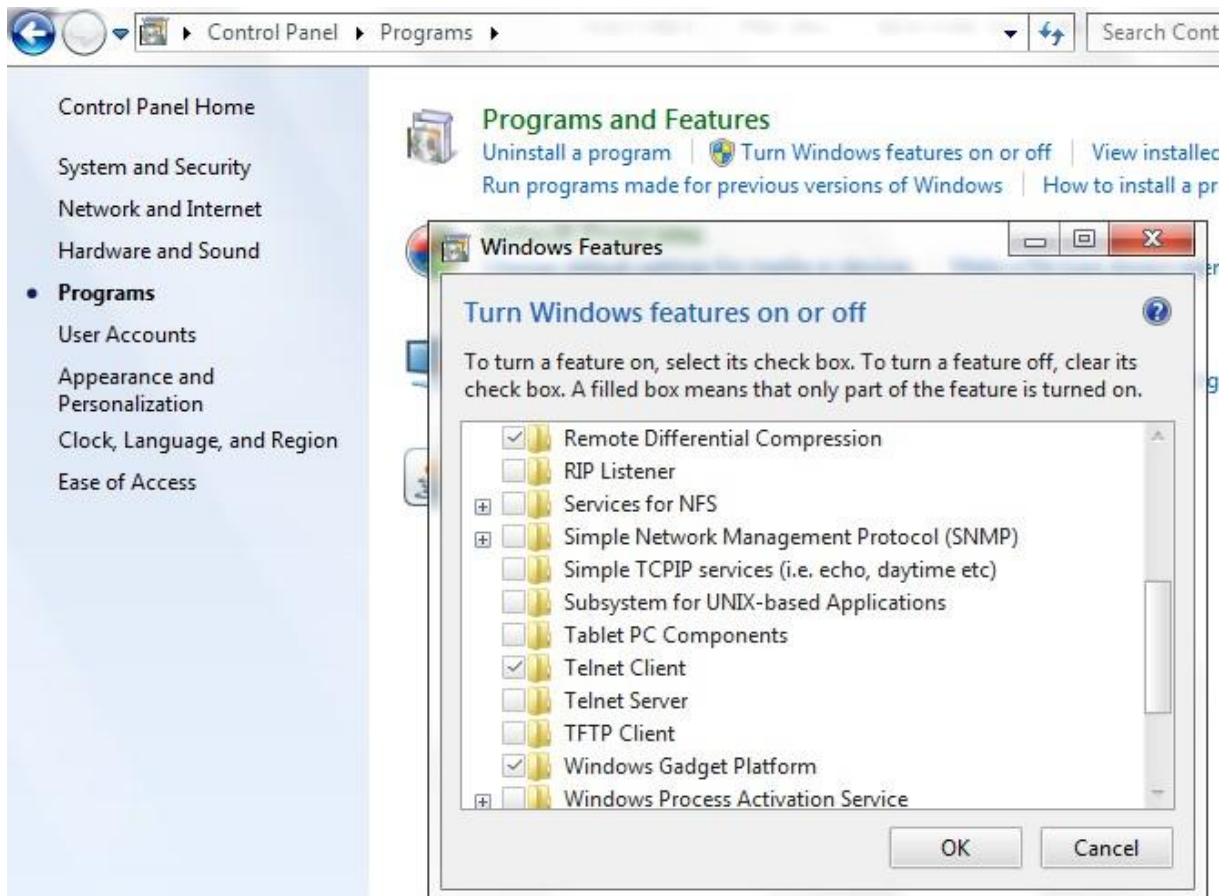
PS C:\Users\Administrator\Downloads\ServerAgent-2.2.1> .\startAgent.bat
INFO 2015-12-23 11:33:59.875 [kg.apc.p] (): Binding UDP to 4444
INFO 2015-12-23 11:33:59.906 [kg.apc.p] (): Binding TCP to 4444
INFO 2015-12-23 11:33:59.906 [kg.apc.p] (): JP@GC Agent v2.0 started

```

Server Agent sẽ chạy mặc định với port 4444.

Cách chạy file startAgent.sh như sau:

- ✚ Bước 1: SSH vào server bằng tài khoản: username\password
- ✚ Bước 2: cd đến thư mục chứa file startAgent.sh
- ✚ Bước 3: Thêm quyền để chạy được file startAgent.sh bằng lệnh: chmod 777 startAgent.sh (nếu account không phải là root thì phải chạy lệnh này trước thì mới có quyền chạy ./startAgent.sh).
- ✚ Bước 4: Chạy lệnh: ./startAgent.sh
 - Bây giờ kiểm tra xem máy cài Jmeter có kết nối được tới server chịu tải qua port 4444 không như sau:
 - + Trên máy cài Jmeter: chạy lệnh: telnet <ip server chịu tải> 4444, nếu có kết nối thì kiểm tra lệnh bắn ra trên server chịu tải “Accepting new TCP connection” có nghĩa là Jmeter và server đã thông kết nối qua port 4444. Nếu không có kết nối thì phải mở kết nối.
 - + Nếu telnet, windows báo không nhận ra lệnh telnet thì vào tính năng “Turn windows features on or off” để enable “Telnet Client” như hình dưới:



a. **Chú ý:**

- Nhiều khi Jmeter không thể connected với server vì firewall trên server chặn, vậy nên cần phải tắt firewall trên server chịu tải (mình thường tắt firewall cả trên máy client và server).
- Cần đảm bảo một vài đặc tính:
 - + Server Agent phải đang chạy trên server chịu tải (luôn luôn chạy).
 - + Máy cài Jmeter và server chịu tải phải thông kết nối với nhau qua port 4444 qua giao thức TCP or UDP (có thể là 1 port free khác)
 - + Metrics Collector Listener enabled and added to the JMeter Test Plan

3.15.1.3 Cấu hình với port free khác

Để start Server Agent, cách đơn giản nhất là chạy file startAgent.sh/ startAgent.bat. Nó sẽ mở port mặc định 4444 qua giao thức UDP/TCP, khi đó Jmeter sẽ kết nối và truy vấn lấy thông tin. Tuy nhiên chúng ta có thể start Server Agent bằng bất kỳ port nào và tất nhiên port này sẽ phải thông kết nối giữa Jmeter và Server Agent.

- Mở Server Agent bằng port free khác:

```
[app@tempcentos ServerAgent-2.2.1]$ ./startAgent.sh --udp-port 0 --tcp-port 9000 INFO  
2016-12-13 15:57:01.741 [kg.apc.p] (): Binding TCP to 9000
```

```
INFO 2016-12-13 15:57:01.773 [kg.apc.p] (): JP@GC Agent v2.2.0 started
```

- Sau khi Server Agent mở port xxxx bất kỳ nào đó trên server thành công như trên, thì việc tiếp theo ta phải kiểm tra kết nối từ máy cài Jmeter tới server chịu tải qua port xxxx? Nếu không có kết nối thì phải mở kết nối. Sau đó thực hiện các bước còn lại như phần b để thu thập thông tin server.
- Có thể sử dụng lệnh tùy chọn `--auto-shutdown` khi mở agent để tự động đóng session agent sau khi test plan hoàn thành. Tính năng này luôn được khuyên dùng với những kết nối TCP.

```
[app@tempcentos ServerAgent-2.2.1]$ ./startAgent.sh --udp-port 0 --auto-shutdown
```

```
INFO 2016-12-13 17:33:41.030 [kg.apc.p] (): Agent will shutdown when all clients disconnected
```

```
INFO 2016-12-13 17:33:41.054 [kg.apc.p] (): Binding TCP to 4444
```

```
INFO 2016-12-13 17:33:41.061 [kg.apc.p] (): JP@GC Agent v2.2.0 started
```

- Bạn có thể sử dụng tùy chọn lệnh `--sysinfo` để xem các đối tượng của hệ thống đang chạy:

```
[app@tempcentos ServerAgent-2.2.1]$ ./startAgent.sh --sysinfo
```

```
INFO 2016-12-13 17:36:39.800 [kg.apc.p] (): *** Logging available processes ***
```

```
INFO 2016-12-13 17:36:39.854 [kg.apc.p] (): Process: pid=4336 name=bash args=/bin/sh ./startAgent.sh --sysinfo
```

```
INFO 2016-12-13 17:36:39.854 [kg.apc.p] (): Process: pid=4338 name=java args=java -jar ./CMDRunner.jar --tool PerfMonAgent --sysinfo
```

```
INFO 2016-12-13 17:36:39.855 [kg.apc.p] (): Process: pid=5850 name=bash#1 args=-bash
```

```
INFO 2016-12-13 17:36:39.855 [kg.apc.p] (): Process: pid=25699 name=java#1 args=/home/app/hanhls1/jdk1.8.0_77/bin/java
```

```
Djava.util.logging.config.file=/home/app/hanhls1/tomcat-for-use-service/conf/logging.properties
```

```
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager - Dfile.encoding=UTF-8 -Xms128m -Xmx1024m -XX:PermSize=64m - XX:MaxPermSize=256m
```

```
Dorg.apache.cxf.stax.allowInsecureParser=1
```

```
Djava.endorsed.dirs=/home/app/hanhls1/tomcat-for-use-service/endorsed -classpath /home/app/hanhls1/tomcat-for-use-service/bin/bootstrap.jar:/home/app/hanhls1/tomcat-for-use-service/bin/tomcat-juli.jar -Dcatalina.base=/home/app/hanhls1/tomcat-for-use-service - Dcatalina.home=/home/app/hanhls1/tomcat-for-use-service
```

```
Djava.io.tmpdir=/home/app/hanhls1/tomcat-for-use-service/temp
```

```
org.apache.catalina.startup.Bootstrap start
```

```

INFO 2016-12-13 17:36:39.862 [kg.apc.p] (): *** Logging available filesystems
***  

INFO 2016-12-13 17:36:39.863 [kg.apc.p] (): Filesystem: fs=/sys type=sysfs  

INFO 2016-12-13 17:36:39.863 [kg.apc.p] (): Filesystem: fs=/proc/sys/fs/binfmt_misc type=binfmt_misc  

INFO 2016-12-13 17:36:39.863 [kg.apc.p] (): Filesystem: fs=/boot type=ext4 INFO 2016-12-13 17:36:39.863 [kg.apc.p] (): Filesystem: fs=/dev/shm type=tmpfs INFO 2016-12-13 17:36:39.863 [kg.apc.p] (): Filesystem: fs=/dev/pts type=devpts INFO 2016-12-13 17:36:39.864 [kg.apc.p] (): Filesystem: fs=/u01 type=ext4 INFO 2016-12-13 17:36:39.864 [kg.apc.p] (): Filesystem: fs=/ type=ext4  

INFO 2016-12-13 17:36:39.864 [kg.apc.p] (): Filesystem: fs=/proc type=proc  

INFO 2016-12-13 17:36:39.864 [kg.apc.p] (): *** Logging available network interfaces ***  

INFO 2016-12-13 17:36:39.865 [kg.apc.p] (): Network interface: iface=lo addr=127.0.0.1 type=Local Loopback  

INFO 2016-12-13 17:36:39.865 [kg.apc.p] (): Network interface: iface=eth1 addr=10.58.71.184 type=Ethernet  

INFO 2016-12-13 17:36:39.865 [kg.apc.p] (): *** Done logging sysinfo  

*** INFO 2016-12-13 17:36:39.865 [kg.apc.p] (): Binding UDP to 4444  

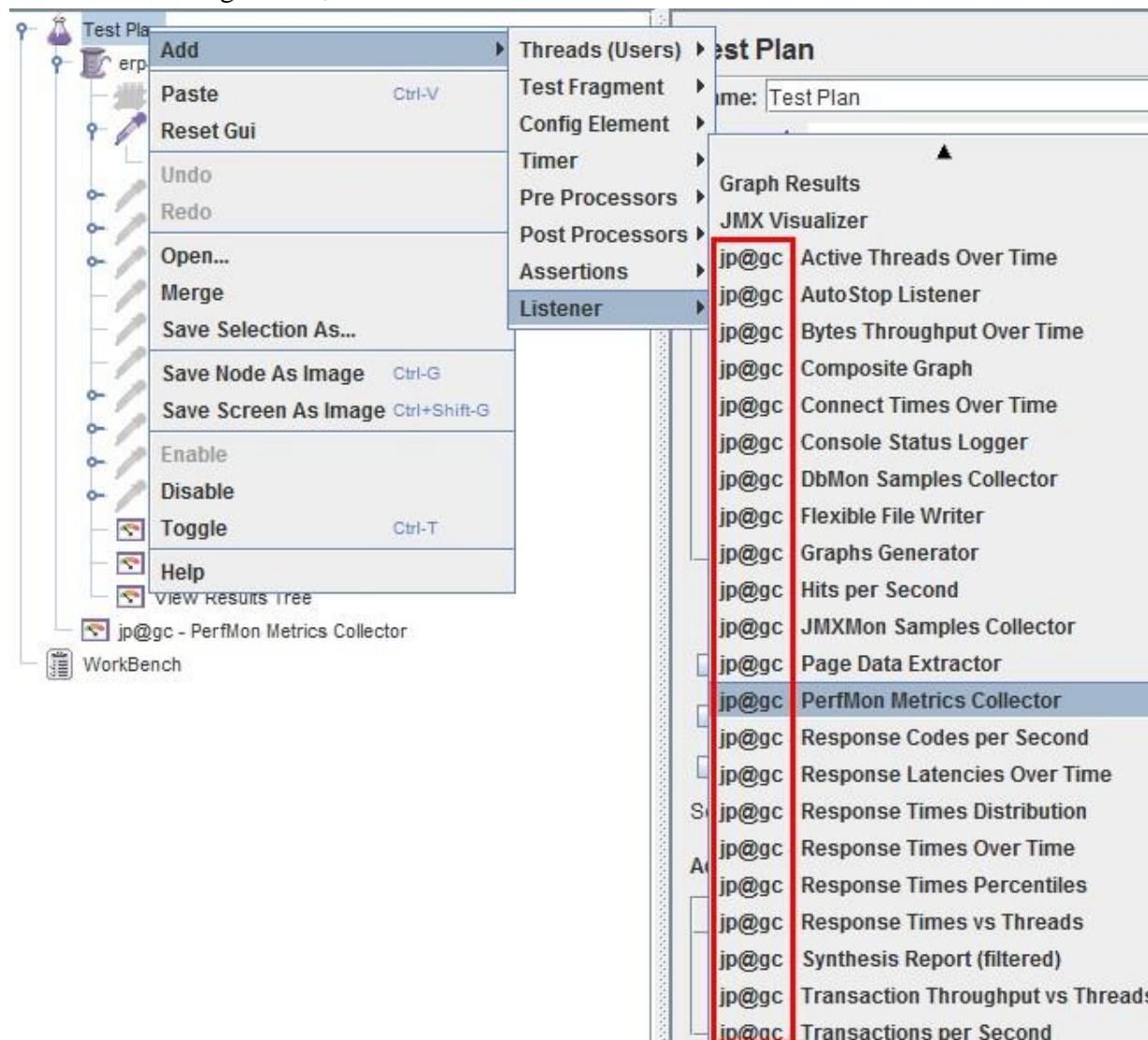
INFO 2016-12-13 17:36:40.868 [kg.apc.p] (): Binding TCP to 4444  

INFO 2016-12-13 17:36:40.873 [kg.apc.p] (): JP@GC Agent v2.2.0 started

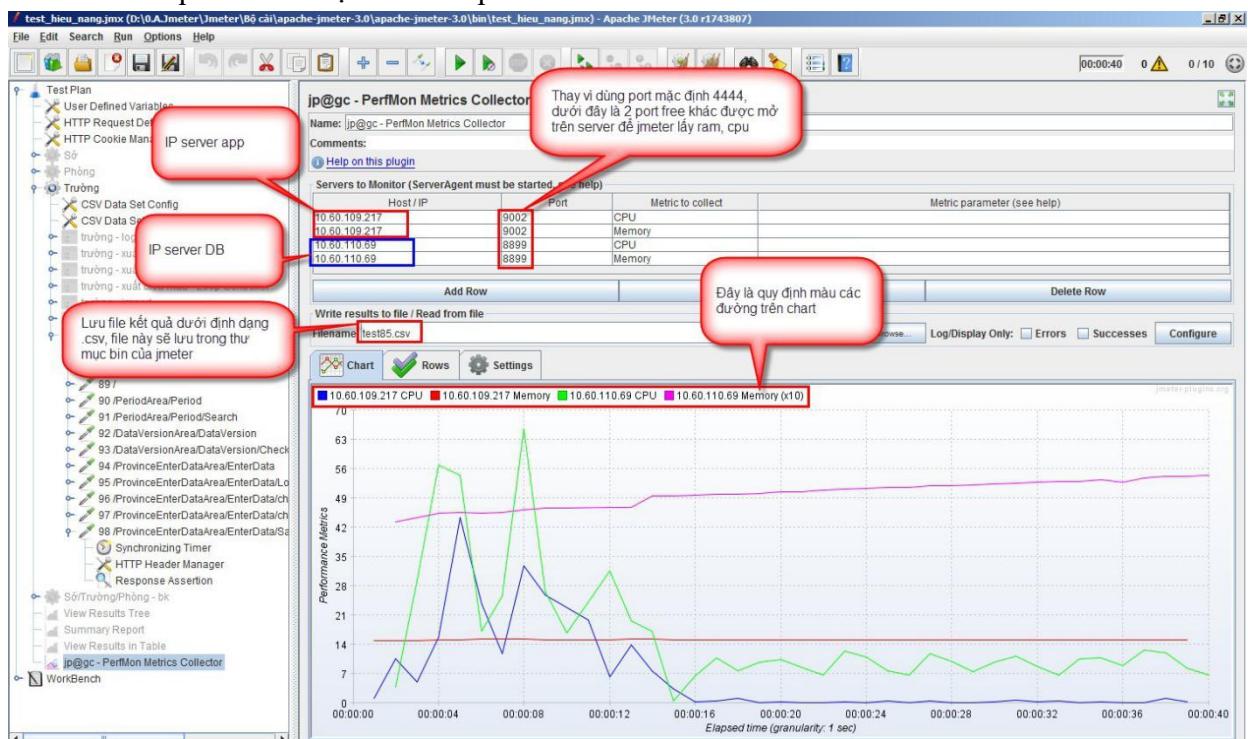
```

3.15.2 Cách xem thông số của server

- Trong Jmeter, add PerfMon Metrics Collector Listener



2. Thêm IP của server chịu tải + port mà PerfMon Server Agen đang lắng nghe (port mặc định là 4444) và chọn Metric to collect. Chạy test plan và xem thông số server qua Chart hoặc lưu kết quả vào file .csv:



3. Việc xem số liệu thông số trên Jmeter như thế này để phục vụ cho việc phát triển và debug kịch bản test. Còn trong trường hợp test tải, thì để xuất test ở chế độ non-GUI mode và đồng thời đẩy dữ liệu các thông số server ra file.
- b. Cách đọc kết quả trong file .csv

- Kết quả lưu trong file .csv như ảnh dưới sẽ lưu vào thư mục/apache-jmeter-3.0\bin của Jmeter:

Write results to file / Read from file
Filename **test85.csv**

- Nội dung file .csv như sau:

	timeStamp	elapsed	label	response	response	threadName	dataType	success	failureMe	bytes	grpThreads	allThreads	Latency	IdleTime
2	12/29/2016 11:22	13275	10.60.109.217 Memory					TRUE		0	0	0	0	0
3	12/29/2016 11:22	11247	10.60.109.217 CPU					TRUE		0	0	0	0	0
4	12/29/2016 11:22	4144	10.60.110.69 Memory					TRUE		0	0	0	0	0
5	12/29/2016 11:22	3922	10.60.110.69 CPU					TRUE		0	0	0	0	0
6	12/29/2016 11:22	13848	10.60.109.217 Memory					TRUE		0	0	0	0	0
7	12/29/2016 11:22	5458	10.60.109.217 CPU					TRUE		0	0	0	0	0
8	12/29/2016 11:22	4217	10.60.110.69 Memory					TRUE		0	0	0	0	0
9	12/29/2016 11:22	8411	10.60.110.69 CPU					TRUE		0	0	0	0	0
10	12/29/2016 11:22	13942	10.60.109.217 Memory					TRUE		0	0	0	0	0
11	12/29/2016 11:22	29690	10.60.109.217 CPU					TRUE		0	0	0	0	0
12	12/29/2016 11:22	4317	10.60.110.69 Memory					TRUE		0	0	0	0	0
13	12/29/2016 11:22	10137	10.60.110.69 CPU					TRUE		0	0	0	0	0
14	12/29/2016 11:22	14221	10.60.109.217 Memory					TRUE		0	0	0	0	0
15	12/29/2016 11:22	53550	10.60.109.217 CPU					TRUE		0	0	0	0	0
16	12/29/2016 11:23	15022	10.60.109.217 Memory					TRUE		0	0	0	0	0
17	12/29/2016 11:23	1212	10.60.109.217 CPU					TRUE		0	0	0	0	0
18	12/29/2016 11:23	4339	10.60.110.69 Memory					TRUE		0	0	0	0	0
19	12/29/2016 11:23	3796	10.60.110.69 CPU					TRUE		0	0	0	0	0
20	12/29/2016 11:23	15022	10.60.109.217 Memory					TRUE		0	0	0	0	0
21	12/29/2016 11:23	10567	10.60.109.217 CPU					TRUE		0	0	0	0	0
22	12/29/2016 11:23	4438	10.60.110.69 Memory					TRUE		0	0	0	0	0
23	12/29/2016 11:23	29367	10.60.110.69 CPU					TRUE		0	0	0	0	0

Giải thích các thông số:

- Timestamp: là thời gian Jmeter lấy được thông tin của server, thời gian này không phụ thuộc vào thời gian trên các server, nó lấy theo ngày/giờ/phút/giây của máy client cài Jmeter.
- Elapsed: là giá trị của các thông số của server
Ví dụ: RAM = 5431 => % sử dụng RAM của server = $5431/1000 = 5.431\%$
- Label: là danh sách các server tải tương ứng với các thông số như RAM, CPU, DISK, ...

Chú ý:

- Để lấy được thời gian (timestamp) ở định dạng ngày/tháng/năm giờ:phút:giây như nội dung file trên, ta phải cấu hình thêm lệnh "`jmeter.save.saveservice.timestamp_format=yyyy/MM/dd HH:mm:ss`" trong file user.properties.
- Để lấy đúng RAM, CPU của request cần Synchronizing Timer => thời gian trả về thông số RAM/CPU = thời gian đồng bộ + thời gian phản hồi (min) -> thời gian hoàn thành test plan.

3.15.3 GUI mode

- Nếu bạn chạy Jmeter ở chế độ GUI, chỉ cần add PerfMon Metrics Collector Listener, định nghĩa servers và các thông số cần monitor, đảm bảo rằng agent đang được chạy trên

server chịu tải và nó không bị khóa bởi firewall, sau đó chạy test plan. Các giá trị sẽ được hiển thị ở biểu đồ thời gian thực.

c. Non GUI Mode

- Nếu bạn chạy Jmeter ở chế độ non-GUI và muốn lưu dữ liệu sau khi monitor ra file, chỉ cần cấu hình file kết quả trong test plan ở các listeners trên GUI, hệ thống sẽ tự lưu lại. Sau khi chạy test plan có thể tải file đã lưu vào trong GUI và xem các giá trị.

3.16 Request đính kèm thêm file

Khi record các request mà có đính kèm thêm file thì giao dịch thường bị lỗi, bạn kiểm tra lại giao dịch đó và sẽ thấy ở mục Send File with the Request chỉ hiển thị tên file, không hiển thị lên đường dẫn đầy đủ đến file. Do đó để chạy được thì bạn cần điền đầy đủ đường dẫn đến file đó hoặc copy file đó vào thư mục bin của Jmeter

Send Files With the Request:		
File Path:	Parameter Name:	MIME Type:
abc.xls		

Add **Browse...** **Delete**

Cần cập nhật lại

Send Files With the Request:		
File Path:	Parameter Name:	MIME Type:
abc.xls		

Add **Browse...** **Delete**