

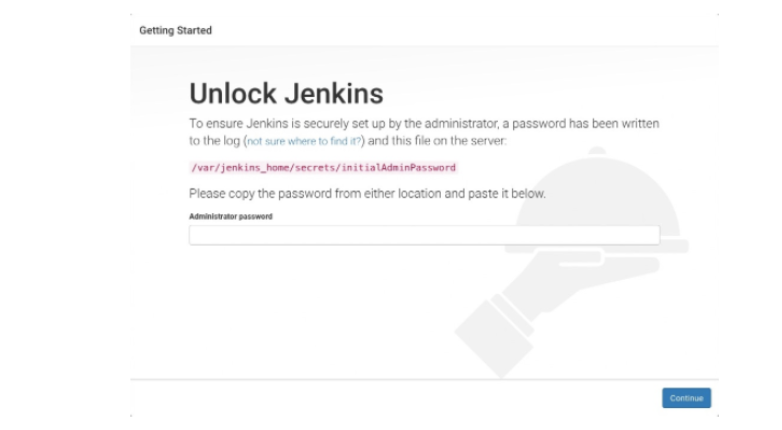
Hướng dẫn cài đặt Jenkin

1. **Cài đặt jdk** Để kiểm tra jdk cài đặt hay chưa bằng cách gõ lệnh: java -version Thực hiện cài đặt: Open Java Development Kit (OpenJDK) như sau: Update the repositories *sudo apt update* search of all available packages: *sudo apt search openjdk* Pick one option and install it: *sudo apt install openjdk-11-jdk* Confirm installation:
2. *sudo apt install openjdk-11-jdk* checking installation: *java -version* Kết quả đúng sẽ ra như sau: *openjdk version "11.0.9.1" 2020-11-04* OpenJDK Runtime Environment (build 11.0.9.1+1-post-Debian-1deb10u2) OpenJDK 64-Bit Server VM (build 11.0.9.1+1-post-Debian-1deb10u2, mixed mode, sharing)
3. **Cài đặt jenkins** Search google: Install jenkins on Ubuntu để tìm trang cài đặt Jenkins trên Ubuntu Gõ lệnh: wget -q -O - <https://pkg.jenkins.io/debian-stable/jenkins.io.key> | sudo apt-key add - Gõ tiếp lệnh: sudo sh -c 'echo deb <https://pkg.jenkins.io/debian-stable> binary/ > /etc/apt/sources.list.d/jenkins.list' Gõ tiếp lệnh: sudo apt-get update Gõ tiếp lệnh:

sudo apt-get install jenkins **3. Start Jenkins** Register the Jenkins service with the command: sudo systemctl daemon-reload You can start the Jenkins service with the command: sudo systemctl start jenkins You can check the status of the Jenkins service using the command: sudo systemctl status jenkins Nếu thực hiện đúng lệnh sẽ hiển thị: ● jenkins.service - LSB: Start Jenkins at boot time Loaded: loaded (/etc/init.d/jenkins; generated) Active: active (exited) since Fri 2021-10-08 08:45:12 +07; 2min 48s ago Docs: man:systemd-sysv-generator(8) Tasks: 0 (limit: 9077) Memory: 0B CGroup: /system.slice/jenkins.service

Thg 10 08 08:45:11 lumi-desktop jenkins[6988]: Correct java version found Thg 10 08 08:45:11 lumi-desktop jenkins[6988]: \* Starting Jenkins Automation Server jenkins Thg 10 08 08:45:11 lumi-desktop su[7045]: (to jenkins) root on none Thg 10 08 08:45:11 lumi-desktop su[7045]: pam\_unix(su-l:session): session opened for user jenkins by (uid=0) Thg 10 08 08:45:11 lumi-desktop su[7045]: pam\_unix(su-l:session): session closed for user jenkins Thg 10 08 08:45:12 lumi-desktop jenkins[6988]: ...done. Thg 10 08 08:45:12 lumi-desktop systemd[1]: Started LSB: Start Jenkins at boot time.

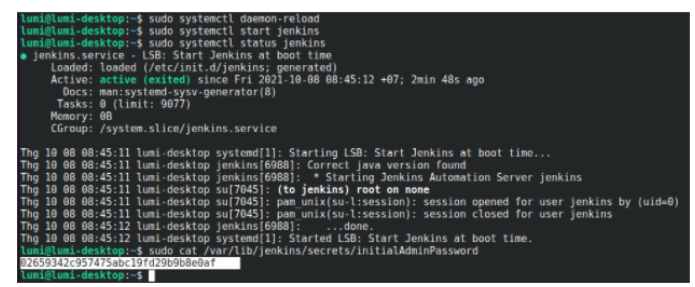
**4.Unlocking Jenkins** Gõ địa chỉ: [http://localhost:8080](http://localhost:8080/), thì sẽ xuất hiện màn hình Unlock Jenkins như sau:



Gõ lệnh:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

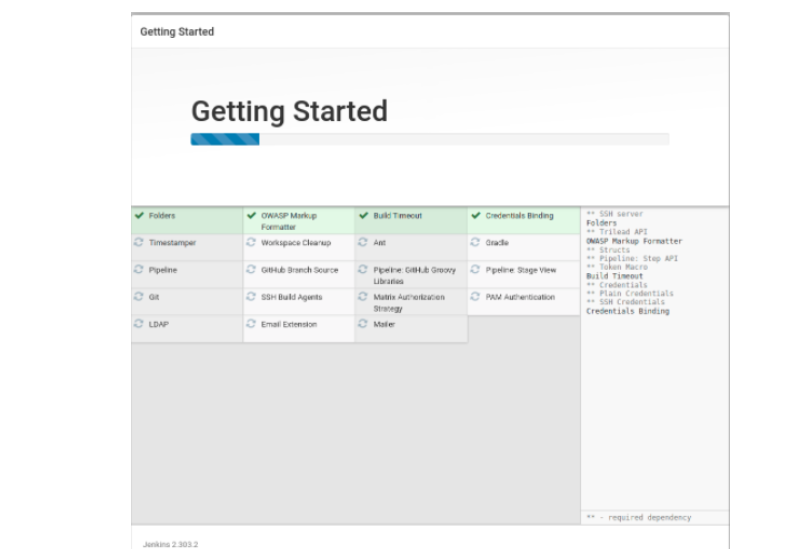
Sau khi gõ lệnh, pass để mở khóa Jenkins hiển thị như sau:



**coppy pass của Jenkins vào trang:** [**http://localhost:8080**](http://localhost:8080/)

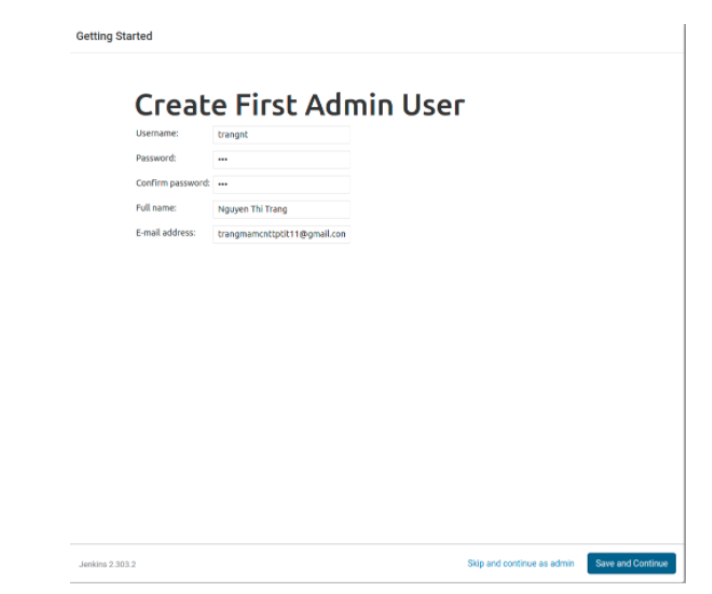
Sau khi dăng nhập với password vừa hiển thị, thì 1 page hiển thị và chọn : Install suggested

plugins



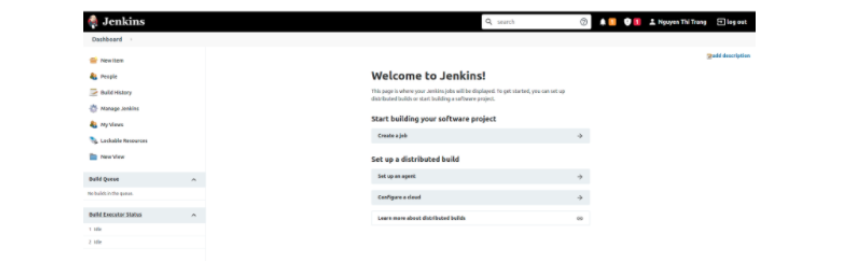
Sau khi build Plugin xong thì màn hình sẽ chuyển qua bước tạo tài khoản Admin User, với các thông tin như sau:

* username
* password
* confirm password
* full name
* Email address



Nhấn vào button: “Save and countinue”, màn hình login xuất hiện, và hoàn thành các bước cài

đặt



**5. Cách xóa Jenkins trên ubuntu**

1. Dừng dịch vụ jenkins đang chạy: sudo service jenkins stop
2. Chạy lệnh xóa: sudo apt-get remove –purge jenkins

Tìm hiểu về tool test Jmeter

[Nguyễn Thị Trang updated over 1 year ago](https://docs.lumi.vn/doc/tim-hieu-ve-tool-test-jmeter-3pwUUVshYy/history)  • [Comment](https://docs.lumi.vn/doc/tim-hieu-ve-tool-test-jmeter-3pwUUVshYy) • [Viewed by 4 people](https://docs.lumi.vn/doc/tim-hieu-ve-tool-test-jmeter-3pwUUVshYy/insights)

#**Phần I. Vài nét sơ lược về tool Jmeter**

JMeter là công cụ có thể test tải, test chức năng, test hồi quy... trên nhiều giao thức và các công nghệ khác nhau:

* Web: HTTP, HTTPS
* Web service: SOAP
* Database với JDBC drivers
* FTP service

JMeter là một ứng dụng Java desktop với giao diện đồ hoạ sử dụng Swing graphical API. Do đó nó có thể chạy trên bất kỳ môi trường / máy trạm nào chấp nhận một máy ảo Java, ví dụ như Windows, Linux, Mac, v.v .

JMeter là một phần mềm có thể được sử dụng để thực hiện kiểm tra hiệu năng (**performance testing**), kiểm tra tải (**load testing**) và kiểm tra chức năng (**functional testing**) của các ứng dụng web. JMeter cũng có thể mô phỏng một tải nặng trên một máy chủ bằng cách tạo ra hàng ngàn người dùng ảo đồng thời truy cập máy chủ web.

Jmeter là một mã nguồn mở, hiện nay Jmeter đã trở thành một trong những công cụ kiểm tra phổ biến nhất trên thế giới bên cạnh Selenium, Loadrunner

**1.1. Tính năng của Jmeter** Các tính năng chính của JMeter bao gồm:

* Giấy phép (License): JMeter là một mã nguồn mở, vì vậy nó miễn phí và người dùng có thể dễ dàng tải về và sử dụng.
* Giao diện đồ hoạ người dùng (Graphical User Interface – GUI): Đơn giản, thân thiện với người dùng và dễ học so với các công cụ kiểm thử hiệu năng khác.
* Hỗ trợ Giao thức/ Máy chủ (Server/Protocol Support): JMeter có khả năng thực hiện kiểm thử hiệu năng nhiều ứng dụng, máy chủ, giao thức khác nhau. Một vài giao thức bao gồm HTTP, HTTPS, FTP, SOAP/REST, Database thông qua JDBC, LDAP, JMS, SMTP(S), POP(3) và IMAP(S), Native Commands/Shell Scripts và TCP.
* Nền tảng (Platform): JMeter được phát triển dựa vào Java thuần (pure Java). Do đó, nó là nền tảng độc lập và hỗ trợ tất cả môi trường.
* Mô phỏng (Simulation): Mô phỏng nhiều người dùng bằng cách sử dụng người dùng ảo hoặc người dùng duy nhất để tạo lưu lượng truy cập lớn trên server hoặc web services.
* Hỗ trợ Phân phối tải (Supports Distributed Testing): JMeter tạo ra các Master và Salves để thực hiện phân phối tải, trong đó Master sẽ thực hiện làm nhiệm vụ phân phối việc đẩy request cho các Slaves, các Salves sẽ thực hiện đẩy các request từ script lên server cần kiểm thử hiệu năng.
* Hiển thị kết quả (Test Result Visualization): Kết quả kiểm thử hiệu năng có thể xem với nhiều định dạng khác nhau như Bảng, Biểu đồ, Cây, Báo cáo…
* Báo cáo (Reporting): Theo mặc định, JMeter chỉ cung cấp định dạng báo cáo là XML và CSV. Chúng ta có thể sử dụng JMeter và ANT cùng với nhau để có được báo cáo HTML theo yêu cầu. Từ bản JMeter 3.0 có hỗ trợ HTML Report (Dashboard Report) hiển thị đầy đủ thông tin và chuyên nghiệp.
* Các loại kiểm thử (Testing Types): Ngoài việc kiểm thử hiệu năng, Load, Stress Testing, JMeter vẫn có thể được sử dụng để kiểm thử tốt về Chức năng, Regression Testing hay Soak/Endurance Testing cũng đều có thể sử dụng tốt.
* Record and Playback: JMeter hỗ trợ việc ghi lại (record) kịch bản/hành động của người dùng qua các trình duyệt và playback lại script sau khi record xong.
* Framework: Framework đa luồng (multi-threading) cho phép có thể mô phỏng các requests đồng thời thực hiện các chức năng khác nhau bởi nhiều Thread Groups riêng biệt.
* Cài đặt (Installation): Cài đặt dễ dàng và đơn giản – Chỉ cần download file, giải nén và chạy file JMeter.bat trên Windows hoặc chạy file [JMeter.sh](http://jmeter.sh/) trên Linux.
* Kiến thức (Knowledge): JMeter không yêu cầu phải có kiến thức sâu rộng về lập trình do giao diện người dùng khá dễ hiểu.

**1.2. Ưu, nhược điểm của công cụ** a) Ưu điểm

* Mã nguồn mở: Đây là một phần mềm nguồn mở, người dùng có thể tải được về miễn phí và sửa đổi, tùy chỉnh nó theo yêu cầu của mình. Là một ứng dụng Java thuần
* Dễ sử dụng: Người dùng có thể cài đặt và sử dụng một cách dễ dàng, chỉ cần tải xuống từ Internet, cài đặt và chạy
* Nền tảng độc lập: do phát triển trong môi trường java nên Jmeter có thể chạy trong bất kỳ hệ điều hành: window, linux, mac
* Báo cáo hiệu quả: kết quả kiểm tra có thể được xem bằng sử dụng đồ thị, dạng cây, dạng bảng…hoặc hỗ trợ cho báo cáo văn bản XML, HTML, Json
* Kiểm thử tối ưu: có thể làm bất kỳ loại thử nghiệm: load test, stress test, test functional, distributed test trong một công cụ
* Tính linh hoạt: người dùng có thể tùy chỉnh theo yêu cầu của mình và áp dụng thử nghiệm tự động cho jmeter (them plugin)
* Hỗ trợ đa giao thức: HTTP, FPT, SOAP, JDBC, JMS, LDAP,.. ngoài ra nó còn có thể sử dụng để kiểm tra hiệu suất của CSDL
* Record & Playback: đây là tính năng giúp ghi lại các hoạt động của người dùng trên trình duyệt đồng thời đưa ra mô phỏng nó trong một ứng dụng web sử dụng JMeter.
* Jmeter có thể tích hợp được với Bean Shell & Selenium nhằm thực hiện công việc kiểm thử tự động.

b) Nhược điểm

* Tiêu thụ bộ nhớ: do tool có thể mô phỏng tải nặng và báo cáo kết quả nhiều định dạng nên có thể tiêu tốn nhiều bộ nhớ
* Chỉ chạy trên ứng dụng web
* Thiếu hỗ trợ javascript: JMeter không phải là trình duyệt nên không thể chạy JavaScript trong ứng dụng web. Nó có hỗ trợ hạn chế để xử lý JavaScript hoặc Ajax, điều này có thể ảnh hưởng đến độ chính xác của mô phỏng.

**1.3. 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 và cung cấp các kết quả báo cáo hiệu suất dưới dạng biểu đồ trực quan, dễ hiểu

*Quy trình hoạt động của JMeter:*

User gửi request lên server > Request được gửi lên server với số lượng lớn > Server nhận, xử lý và phản hồi kết quả request > JMeter lưu tất cả các responses từ server > JMeter thu thập data để lấy thông tin thống kê rồi trả về server > Kết thúc chu trình làm việc > Lấy về báo cáo kết quả kiểm thử.

* **Load testing**: Mô hình hóa dự kiến sử dụng bởi nhiều người dùng truy cập một dịch vụ website trong cùng thời điểm. Kiểm thử này rất hữu ích để xác định khả năng hoạt động chịu tải của hệ thống trong các điều kiện tải bình thường và cao hơn điều kiện tải dự kiến.
* **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. xác định sự ổn định và sự mạnh mẽ của hệ thống Performance Test giúp kiểm tra hiệu suất của máy chủ trang web, cơ sở dữ liệu, mạng. Kiểm tra xem hệ thống hoạt động như thế nào khi quá tải và cách hệ thống phục hồi khi xảy ra lỗi.

**1.4. Các thành phần trong Jmeter**

Các thành phần của JMeter được gọi là Elements. Mỗi Element được thiết kế cho một mục đích cụ thể.

Hình dưới đây đưa ra một số Elements tiêu biểu trong JMeter.

**1.4.1. Thread Group**

Thread group là đại diện cho một nhóm người dùng và chứa các Thread khác. Mỗi Thread group sẽ mô phỏng những người dùng để thực hiện một trường hợp thử nghiệm cụ thể. Một số thông số cần chú ý trong Thread Group:

*Action to be the taken after Sampler error:*

- Continue: Bỏ qua lỗi, tiếp tục kiểm tra và chạy Sampler tiếp theo

- Start Next Loop: Bỏ qua lỗi, tiếp tục vòng lặp tiếp theo và tiếp tục bài test

- Stop Thread: Thoát Thread hiện tại

- Stop Test: Toàn bộ bài test được dừng lại khi kết thúc 1 sampler bất kỳ.Nó có nghĩa là các sampler đang chờ được xử lý cho đến khi hoàn thành.

- Stop Test Now: Toàn bộ bài test bị dừng lại. Mọi sampler hiện tại đều có thể bị gián đoạn.

*Thread Properties:*

– Number of Threads: Mỗi Thread đại diện cho một người dùng ảo, JMeter cho phép thay đổi số lượng người dùng không hạn chế để thực hiện các thử nghiệm.

– Ramp-Up Period: Thời gian để bắt đầu tất cả các Thread.

– Loop Count: Số lần lặp lại những yêu cầu của người dùng. Ngoài ra, có thể tùy chọn chạy các Thread theo lịch biểu sẵn, xác định hành động sẽ thực hiện khi xảy ra lỗi...

Cách tạo Thread group: click chuột phải vào Test plan -> Add -> Threads (Users) -> Thread group.

Ví dụ: Number of Threads: 100, Ramp-Up Period: 100, Loop Count: 1. Tức là Jmeter sẽ giả lập thao tác cho 100 user thực hiện trong 100s, tức là mỗi user sẽ tiến hành thực hiện cách nhau 1s (100s/100) và lặp lại 1 lần.

Chú ý: 100 user + Loop count: 1 khác gì 10 user + Loop count 10.

Về tổng số request thì bằng nhau, Jmeter sẽ thực hiện 100 lượt test.

Tuy nhiên có sự khác nhau về thứ tự thực hiện của các user như sau:

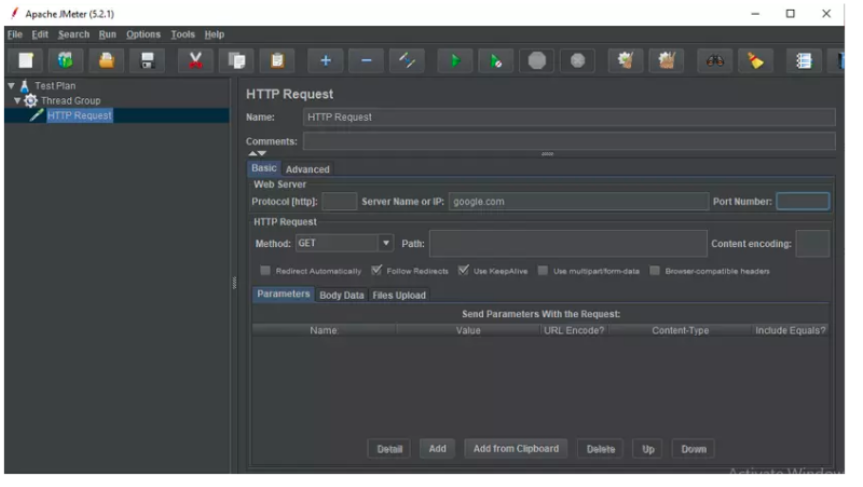
**1.4.2. Samplers** Samplers sẽ chỉ thị JMeter gửi request đến server và chờ đợi phản hồi. Để connect tới server, users cần tạo những request.

Cách tạo Sampler: Click chuột phải vào Thread Group -> Add -> Sampler -> chọn bất kỳ Sampler bạn muốn

Sampler bao gồm:

* FTP Request
* HTTP Request (can be used for SOAP or REST Web Service also)
* JDBC Request
* Java object request
* JMS request
* JUnit Test request
* LDAP Request
* Mail request
* OS Process request
* TCP request

Ví dụ khi sử dụng HTTP Request: Bạn muốn 100 người dùng sẽ tấn công [https://www.google.com](https://www.google.com/) đồng thời. Cách tạo: Thread Group -> Sampler -> HTTP Request.



**1.4.3. Listeners**

Tham khảo thêm các thông số phân tích báo cáo

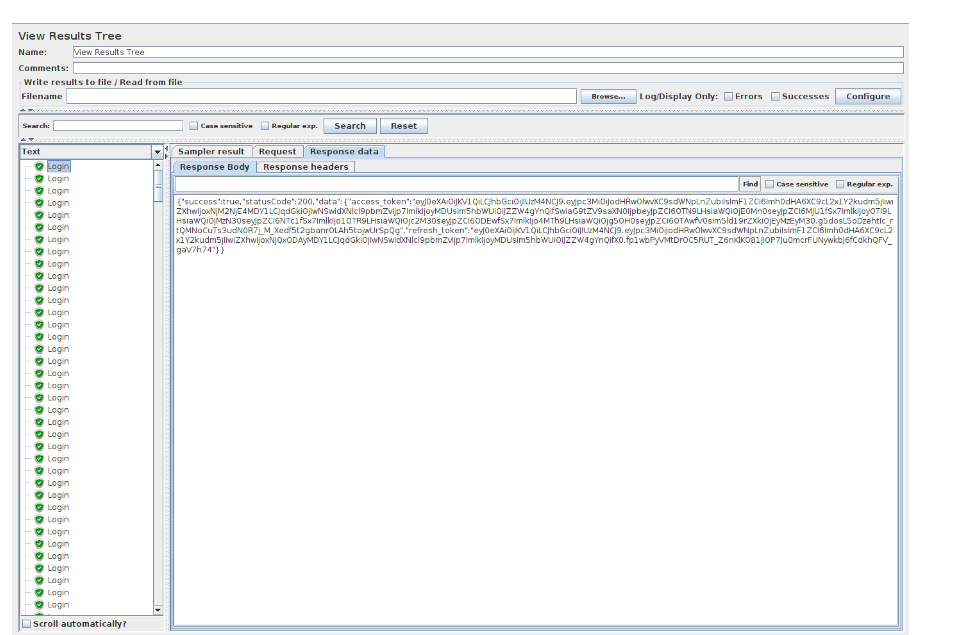
<https://www.testingvn.com/viewtopic.php?t=88057>

Sau khi tạo và gửi yêu cầu tới server, JMeter sẽ nhận kết quả trả về và hiển thị. Listeners cho phép xem các kết quả thu được từ việc chạy dưới dạng các loại khác nhau, như: bảng biểu, đồ thị, cây,… Listener sẽ cung cấp một cách trực quan những dữ liệu thu được.

Cách tạo: Click chuột phải Thread group -> Add -> Listener -> Chọn dạng kết quả.

Chúng ta cùng tìm hiểu một số dạng phổ biến như sau:

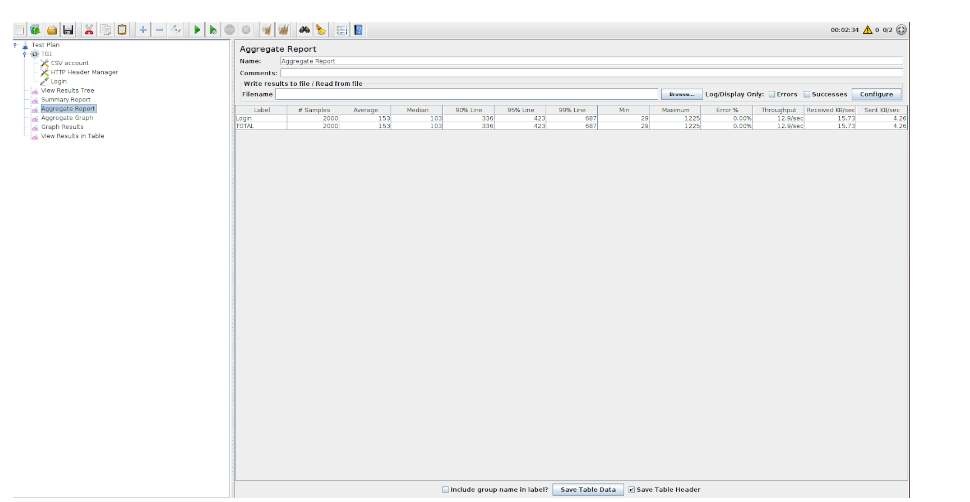
– View Results Tree: Cho phép theo dõi thông tin của dữ liệu mà server trả về cho mỗi người dùng dưới các dạng khác nhau.



- Aggregate Report:

**+ Label:** Hiển thị tên của từng requests có trong Test Plan của bạn.

Mặc định, tất cả những request bị trùng tên trong Test Plan, sẽ chỉ hiển thị 1 dòng duy nhất trong bảng này, cho dù nội dung của các request đó có khác nhau hay nằm khác Thread Group đi chăng nữa. Vì vậy, khi đặt tên cho các Request tránh để bị trùng.



“Include group name in the label?” mặc định là không tích chọn.

Nếu lựa chọn “Include group name in the label?” được check, thì những request sẽ được gán thêm tiền tố = tên của Thread Group chứa request đó.

* Samples: Tổng số lần run của request.

Công thức:

Samples = Number of Threads (users) \* Loop Count

Ví dụ 1: Thread Group có cấu hình

– Number of Threads (users): 10

– Loop Count: 3

Thì 1 HTTP Request của Thread Group này sẽ run 10 x 3 = 30 (lần)

—> # Samples: 30

Tuy nhiên, công thức trên sẽ không còn đúng trong 1 số trường hợp: đó là khi Request của bạn nằm bên dưới 1 Logic Controller nào đó, chẳng hạn như Logic Controller, such as Loop Controller, Once Only Controller, While Controller, v.v…

Ví dụ 2: Tiếp tục với ví dụ 1 ở trên, nhưng lần này thì hãy để HTTP Request vào 1 Logic Controller, là Loop Controller, và để giá trị Loop Count cho controller này là 2. Lúc này request của bạn sẽ run: 10 x 3 x 2 = 60 (lần).

—> # Samples: 60

* Average (millisecond): Thời gian phản hồi trung bình (Response Time) của request, tính cho đến lần run cuối cùng.

Ví dụ 3: Một Request A run tổng cộng 4 lần với các kết quả Response Time tương ứng là 101ms, 106ms, 153ms, và 128ms. Thì Response Time trung bình của Request A sẽ là 122ms.

* Min (millisecond): Response Time thấp nhất của request tính cho toàn bộ tất cả các lần run.

Trong ví dụ 3 ở trên thì Min = 101ms

* Max (millisecond): Response Time cao nhất của request tính cho toàn bộ tất cả các lần run.

Trong ví dụ 3 ở trên thì Max = 153ms

* Percentiles (millisecond):

Nói một cách đơn giản Percentiles sẽ là một con số x, và đi kèm theo 1 giá trị A. Nghĩa là sẽ có x% có giá trị thấp hơn giá trị A, còn lại (100-x)% sẽ có giá trị lớn hơn giá trị A.

Lấy 1 ví dụ đơn giản. Sau một bài kiểm tra ở lớp học, cô giáo nói 90th Percentile điểm số là 6. Nghĩa là 90% số điểm của lớp sẽ dưới 6 điểm, còn lại 10% sẽ cao hơn 6 điểm. Hay một ví dụ khác, sau khi làm bài đánh giá năng lực để pv vào 1 công ty, người ta thông báo cho bạn điểm số của bạn có Percentile là: 74%. Nghĩa là trong số tất cả những người đã làm bài test này, có 74% số người có điểm thấp hơn bạn, và 26% còn lại có điểm số cao hơn bạn.

* Median (millisecond): Nó gần giống với trung bình, nhưng ý nghĩa thì khác hoàn toàn. Median + một giá trị A, sẽ chia toàn bộ các giá trị của bạn thành 2 phần bằng nhau, một phần sẽ chứa những giá trị < A, phần còn lại sẽ chứa những giá trị > A. Median cũng được hiểu như là 50th Percentile. Quay lại Performance, thì Median sẽ chỉ ra, sẽ có 50% số request có response time nhỏ hơn giá trị (hiển thị trên table), và 50% số request còn lại có response time lớn hơn giá trị này

90% Line (90th Percentile) (millisecond):nghĩa là 90% số requests sẽ có response time nhỏ hơn giá trị hiển thị trong table, 10% số requests còn lại sẽ có response time lớn hơn giá trị hiển thị trong table 95% Line (90th Percentile) (millisecond):nghĩa là 95% số requests sẽ có response time nhỏ hơn giá trị hiển thị trong table, 5% số requests còn lại sẽ có response time lớn hơn giá trị hiển thị trong table 99% Line (90th Percentile) (millisecond):nghĩa là 99% số requests sẽ có response time nhỏ hơn giá trị hiển thị trong table, 1% số requests còn lại sẽ có response time lớn hơn giá trị hiển thị trong table 3 thông số percentile 90th, 95th và 99th là những thông số rất được hay sử dụng trong percentile, không chỉ Performance Testing mà còn trong những lĩnh vực khác nữa. Và những con số này hoàn toàn có thể cấu hình được trong JMeter thông qua file jmeter.properties, từ phiên bản 2.12. Mở file này từ folder /JMETER\_HOME/bin/

* Error %: % số lượng request bị fail, bị lỗi.

Ví dụ bạn run request A 100 lần và thấy có 15% errors, nghĩa là request A đã fail/error 15 lần (100\*15%)

* Throughput: Thông lượng. Con số này cho bạn biết được số lượng requests được hệ thống (server) xử lý trong 1 đơn vị thời gian, có thể là giây, phút, hoặc giờ.

Công thức:

Throughput = (Tổng số lượng requests) / (Tổng thời gian) \* (Đơn vị chuyển đổi)

Với:

– Tổng số lượng requests = Tổng số lần request này được run

– Tổng thời gian = (Thời gian bắt đầu chạy của request cuối cùng) + (Thời gian chạy/Response Time của request cuối cùng) – (Thời gian bắt đầu chạy của request đầu tiên)

– Đơn vị chuyển đổi: Mặc định nó sẽ tính theo millisecond, nên để đổi về second thì số này sẽ là 1000, hoặc 1000\*60 nếu bạn muốn chuyển về phút.

* KB/sec: Cũng là thông lượng, nhưng ko đo lường bằng số request, mà đo Kilobytes/second.

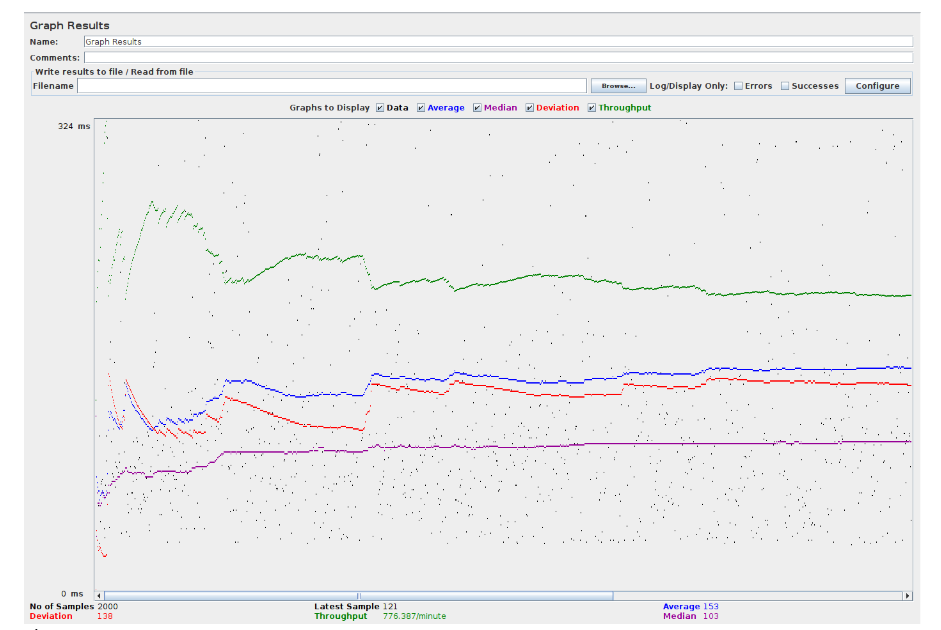
Công thức:

Throughput KB/sec = (Throughput \* Average Bytes) / 1024

Với Aggregate Report thì mình không thấy được thông số Average Bytes. Bạn có thể xem thông số này từ Summary Report.

Total: Trong report có 1 dòng cuối cùng đó là Total, nó sẽ tổng kết lại toàn bộ kết quả từ những request bên trên. Ngoại trừ # Samples, Throughput và KB/sec, nó sẽ được cộng lại theo đúng nghĩa “Total”. Còn các thông số còn lại đều được tính Total bằng cách lấy giá trị trung bình từ tất cả những request ở trên.

* Graph Results:



 Ở phía dưới của ảnh là số liệu thống kê đại diện trong các màu sắc:

Màu đen: Tổng số Samples đã gửi hiện tại

Màu xanh dương: Trung bình Samples đã gửi hiện tại

Màu đỏ: Thống kê độ chênh lệch hiện tại

Màu xanh lá : Tỷ lệ lưu lượng miêu tả số lượng người dùng yêu cầu máy chủ xử lý mỗi phút.

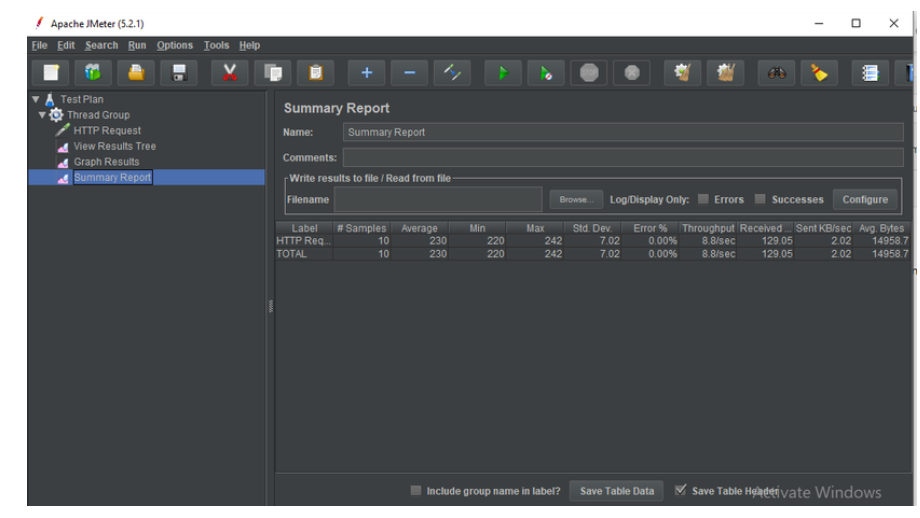
Để phân tích Performance của Web server bạn nên tập trung vào hai thông số: Throughput và Deviation. Trong đó:

* Throughput là thông số quan trọng nhất, nó miêu tả cho khả năng server có thể xử lý được độ tải lớn.
* Deviation được hiển thị màu đỏ nó cho thấy sự sai lệch hiện tại so với mức trung bình càng nhỏ càng tốt.

Phân tích dữ liệu của ví dụ trên ta thấy:

Trong kết quả test thông số Throughput của máy chủ là 776.387/phút. Điều đó có nghĩa là có thể xử lý 776.387 yêu cầu trên mỗi phút.

– Summary Report: Cung cấp báo cáo về các giá trị: thời gian phản hồi thấp nhất/cao nhất, số yêu cầu xảy ra lỗi, lưu lượng trung bình.



Phân tích báo cáo:

Hãy tập trung vào 2 thông số quan trọng nhất của mọi Performance Report:

Response Time: chỉ ra được việc xử lý request NHANH hay CHẬM. Và đương nhiên, Response Time thì phải càng THẤP càng tốt.

Throughput: chỉ ra được số lượng requests được server xử lý trong một đơn vị thời gian. Vậy thì, cùng một thời gian, càng xử lý được càng nhiều càng tốt. Nên với Throughput thì nó phải càng CAO càng tốt

Dựa vào đó, chúng ta có những trường hợp như sau:

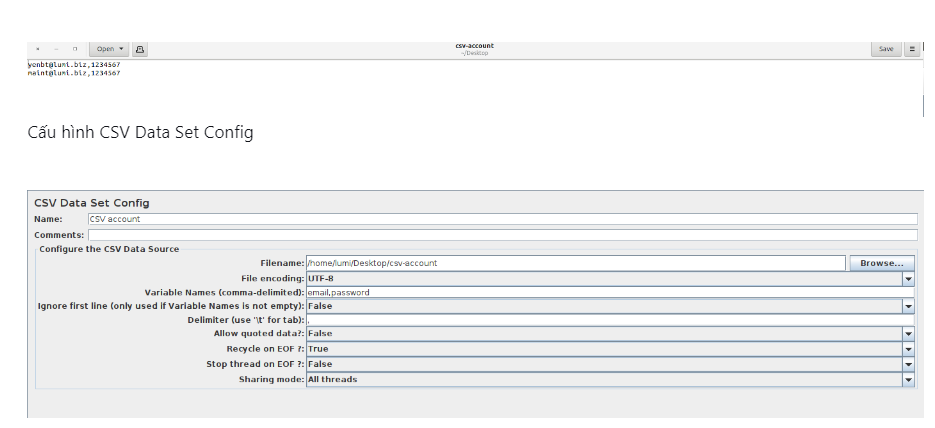
* Response Time: THẤP và Throughput: THẤP –> Trường hợp này sẽ không bao giờ xảy ra. Vì Response Time THẤP nghĩa là thời gian đáp ứng rất nhanh, nhưng Throughput THẤP lại chỉ ra rằng số request được xử lý rất ít. Noooo, chuyện này là vô lý
* Response Time: THẤP và Throughput: CAO –> Đây là một kết quả lý tưởng phải không nào các bạn? Thời gian xử lý thấp và số lượng request xử lý cùng đồng thời lại cao. Còn chần chờ gì nữa mà không tự tin báo cáo rằng Server đang rất tốt. Hãy xem xét khả năng mở rộng các tính năng, hoặc tăng thêm số lượng test để tìm xem giới hạn của server là bao nhiêu.
* Response Time: CAO và Throughput: THẤP –> Ngược lại với bên trên, đây là lúc mà Performance Test của bạn đã bị fail. Test chỉ ra rằng thời gian xử lý quá cao, và lượng request được xử lý lại rất thấp. Phải xem xét để improve về phía server side.
* Response Time: CAO và Throughput: CAO –> Khá nhạy cảm, vì bạn có thể thấy Throughput cao, tức là server đang làm việc rất tốt, vậy tại sao thời gian xử lý lại cũng cao (không tốt). Có thể vấn đề lúc này đế từ phía Client, hoặc cụ thể là đến từ JMeter, có thể đoạn script của bạn viết chưa được tối ưu, khiến quá trình nó xử lý mất nhiều thời gian chẳng hạn?

**1.4.4. Configuration**

Configuration dùng để thiết lập các giá trị mặc định và các biến để sử dụng sau này bởi các samplers. Hình dưới đây cho thấy một số configuration elements thường được dùng trong JMeter:

*CSV Data Set Config*

Trường hợp bạn muốn muốn kiểm thử đồng thời 100 người dùng đăng nhập bằng thông tin đăng nhập khác nhau. Khi đó bạn không cần ghi lại kịch bản 100 lần. Bạn có thể lưu trữ thông tin đăng nhập (tên người dùng, mật khẩu) vào một file text. Sử dụng ” CSV Data Set Config ” để đọc các tham số từ file.



**FileName:** Tên tệp chính xác (có đuôi .csv) chứa dữ liệu kiểm thử để thực thi (lưu ý chọn đúng đường dẫn)

**Variable Names:** danh sách tất cả các tên biến (được phân tách bằng dấu phẩy) theo cùng thứ tự như được mô tả trong file CSV. Giữ trường này trống và jmeter sẽ lấy hàng đầu tiên từ tệp csv làm tên biến cho mỗi cột.

**Delimiter:** được sử dụng để tách từng bản ghi trong tệp csv. Bạn cần đảm bảo xác định tên biến theo thứ tự chính xác khi bạn cung cấp giá trị trong file csv.

**Allow quoted data?:** Nếu được bật, thì các giá trị có thể được đặt trong ”- dấu ngoặc kép - cho phép các giá trị chứa dấu phân cách.

**Recycle on EOF?**: Nếu số lượng thread lớn hơn số lượng Test Data, bạn có muốn tiếp tục thực hiện kiểm thử bằng cách quay trở lại đọc từ đầu không?

**Stop thread on EOF?:** Nếu chọn “Set”, khi chạy đến EOF sẽ khiến cho thread bị dừng lại.

Sharing mode: Tại đây bạn có thể định nghĩa hành vi chia sẻ của file CSV. Mặc định sẽ chọn là "All threads".

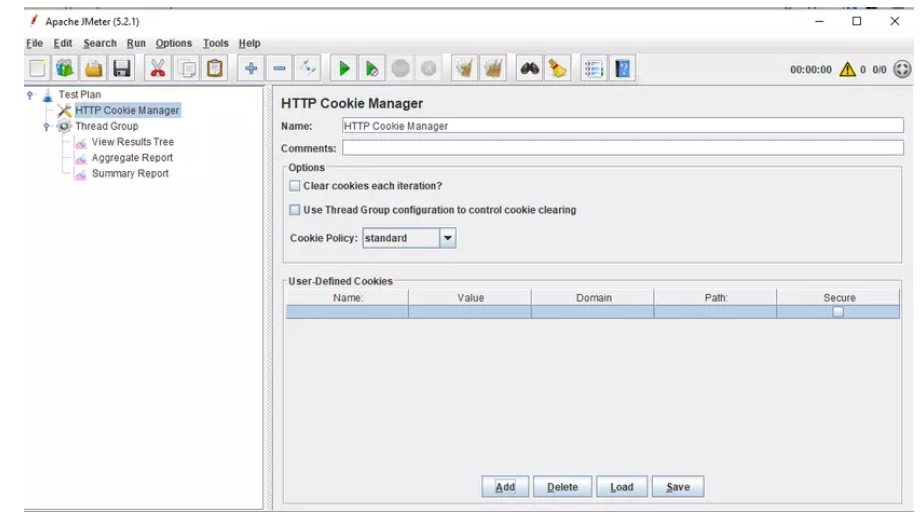
* All threads: Nếu trong script của bạn có nhiều hơn 1 element CSV Data Set Config cùng tham chiếu đến 1 file thì CSV Data Set Config kế tiếp sẽ tiếp tục đọc CSV File đã được mở từ CSV Data Set Config trước.
* Current Thread Group : Nếu trong script của bạn có nhiều hơn 1 element CSV Data Set Config cùng tham chiếu đến 1 file thì CSV Data Set Config kế tiếp sẽ mở lại csv file cho từng thread group
* Current Thread : Mỗi file csv được mở riêng biệt cho từng thread khi chọn option này.

Do đó, khi sử dụng nhiều CSV Data Set Config, hãy đảm bảo sử dụng kết hợp chính xác giữa Sharing Mode, Recycle on EOF & Stop Thread on EOF để nhận kết quả mong muốn.

HTTP Cookie Manager

Bạn đã bao giờ vào một trang web, ví dụ: <https://shopee.vn>, đăng nhập bằng người dùng và mật khẩu của bạn. Sau đó bạn tắt đi và mở lại thì thấy không cần đăng nhập nữa. Đã bao giờ bạn thắc mắc: Tại sao mình không phải đăng nhập nữa không? Lí do là trình duyệt của bạn sẽ sử dụng cookie làm dữ liệu người dùng để đăng nhập.

JMeter cung cấp trình quản lý HTTP Cookie Manager cũng có tính năng tương tự như trình duyệt web. Nếu bạn có yêu cầu HTTP Request có chứa cookie, Cookie Manager sẽ tự động lưu trữ cookie đó và sẽ sử dụng nó cho tất cả các yêu cầu trong tương lai tới trang web cụ thể đó. Đây là màn hình HTTP Cookie Manager:



HTTP request default

HTTP request default cho phép bạn đặt các giá trị mặc định mà bộ điều khiển HTTP Request của bạn sử dụng.

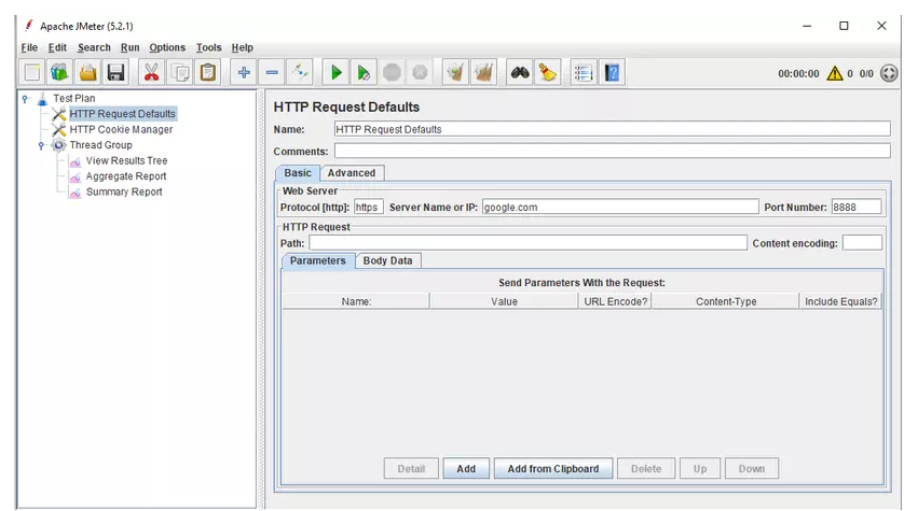
Ví dụ:

Bạn đang gửi 100 yêu cầu HTTP đến máy chủ [google.com](https://www.google.com/)

Bạn sẽ phải nhập thủ công tên máy chủ = [google.com](https://www.google.com/) cho tất cả 100 yêu cầu này

Thay vào đó, bạn có thể thêm một HTTP request defaults với trường “Tên máy chủ hoặc IP” = [google.com](https://www.google.com/)

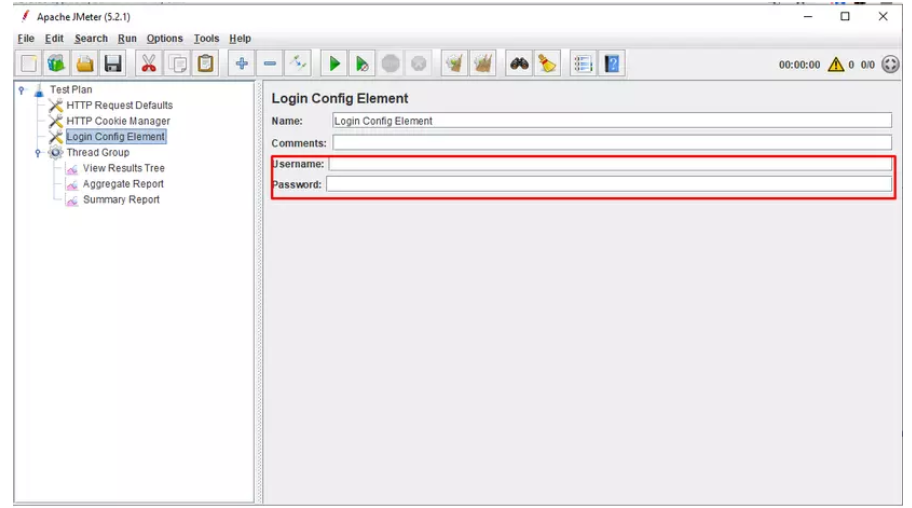
Không cần gõ 100 lần!



Login Config Element

Login Config Element cho phép bạn thêm hoặc ghi đè cài đặt tên người dùng và mật khẩu.

Ví dụ: bạn muốn mô phỏng một người dùng đăng nhập vào trang web www.facebook.com bằng người dùng và mật khẩu. Bạn có thể sử dụng Login Config Element để thêm cài đặt mật khẩu và người dùng này trong yêu cầu người dùng. Ở đây bạn sẽ thấy JMeter sẽ cung cấp Username và password để bạn điền



Login Config Element mô phỏng đăng nhập 1 người dùng, chỉ thích hợp cho tham số đăng nhập (người dùng và mật khẩu).

CSV Data Config mô phỏng đăng nhập nhiều người dùng, thích hợp cho số lượng lớn các tham số.

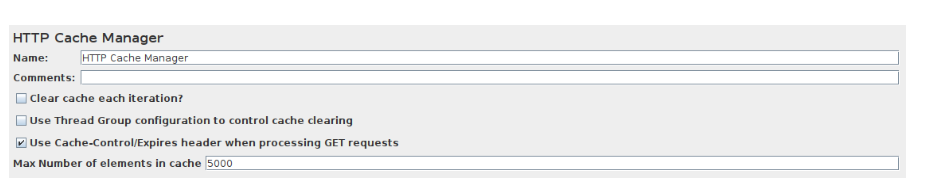
**1.4.5. Config Element**

**HTTP Cache Manager**

Jmeter sẽ không tải nội dung tĩnh cho đến khi nó được cấu hình rõ ràng để làm như vậy. Nếu bạn đã chọn tùy chọn "Retrieve All Embedded Resources", Jmeter sẽ tải tất cả các tệp tĩnh (như HTML, CSS, js, hình ảnh, v.v.) trong khi thực thi. Cache Manager sẽ lưu tất cả tệp tin tĩnh và không tải lại từ server cho đến khi được sửa lại.

Thêm HTTP Cache Manager bằng cách: Mở Jmeter và click chuột phải vào Test Plan -> Add -> Config Element -> HTTP Cache Manager

Thuộc tính của HTTP Cache Manager



**Clear Cache each Iteration** : Sau mỗi lần lặp lại, nội dung đã cache được xóa bỏ.

**Use Cache-Control/Expires header when processing GET requests** : Nếu tùy chọn này được chọn, Jmeter sẽ kiểm tra giá trị Cache-Control / Expires so với thời gian hiện tại. Nếu time stamp trong tương lai và yêu cầu là Get thì Sample sẽ trả về ngay lập tức mà không yêu cầu URL từ Server.

**Maximum Number of Elements in cache** : Jmeter lưu tất cả tài nguyên bộ đệm trong RAM. Theo mặc định, Cache Manager lưu trữ lên tới 5000 mục trong bộ nhớ cache cho mỗi người dùng ảo. Nếu bạn tăng giá trị này, Jmeter sẽ tiêu thụ nhiều bộ nhớ hơn cho phù hợp. Nó có thể dẫn đến ngoại lệ “OutOfMemory”. Để tránh lỗi như vậy, bạn nên chọn JVM-Xmx trong jmeter.bat \ sh.

HTTP Header Manager

Như đã biết, API đã đưa ra có yêu cầu Header là Content-Type: application/json (lưu ý rằng thông tin header các bạn cũng phải bám vào Requirements hoặc lấy thông tin từ hệ thống, mỗi hệ thống sẽ có 1 quy định về header riêng), bây giờ chúng ta sẽ thực hiện việc thiết lập Header trên JMeter.

Click vào HTTP Request Header đã tạo, click button Add và điền thông tin header vào.

