



Become a Quality Assurance Automation Engineer

Course 8 : Pengenalan Load Test dan Stress Test

•Pengenalan Load Test dan Stress Test

Modul Sekolah QA Cilsy

Hak Cipta © 2021 PT. Cilsy Fiolution Indonesia

Hak Cipta dilindungi Undang-Undang. Dilarang memperbanyak atau memindahkan sebagian atau seluruh isi buku ini dalam bentuk apapun, baik secara elektronis maupun mekanis, termasuk mecropy, merekam atau dengan sistem penyimpanan lainnya, tanpa izin tertulis dari Penulis dan Penerbit.

Penulis : Yunita Hutajulu, Amrianto Saragih
Editor : Muhammad Fakhri Abdillah, Iqbal Ilman Firdaus
Revisi Batch 9

Penerbit : **PT. Cilsy Fiolution Indonesia**
Web Site : <https://cilsyfiolution.com> , <https://sekolahqa.com>

Sanksi Pelanggaran Pasal 113 Undang-undang Nomor 28 Tahun 2014 tentang Hak Cipta

1. Setiap orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam pasal 9 ayat (1) huruf i untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan atau pidana denda paling banyak Rp100.000.000,00 (seratus juta rupiah).
2. Setiap orang yang dengan tanpa hak dan atau tanpa izin pencipta atau pemegang hak cipta melakukan pelanggaran hak ekonomi pencipta sebagaimana dimaksud dalam pasal 9 ayat (1) huruf c, huruf d, huruf f, dan atau huruf h, untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah)
3. Setiap orang yang dengan tanpa hak dan atau tanpa izin pencipta atau pemegang hak melakukan pelanggaran hak ekonomi pencipta sebagaimana dimaksud dalam pasal 9 ayat (1) huruf a, huruf b, huruf e, dan atau huruf g, untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan atau pidana denda paling banyak Rp1.000.000.000,00 (satu miliar rupiah)
4. Setiap orang yang memenuhi unsur sebagaimana dimaksud pada ayat (3) yang dilakukan dalam bentuk pembajakan, dipidana dengan pidana penjara paling lama 10 (sepuluh) tahun dan atau pidana denda paling banyak Rp4.000.000.000,00 (empat miliar rupiah)

Daftar Isi

| | |
|---|----|
| Daftar Isi | 3 |
| Performance Testing & Load Testing | 4 |
| Learning Outcomes | 4 |
| Outline Materi | 4 |
| Performance testing & Load testing introduction | 5 |
| Performance Testing | 5 |
| Load Testing | 5 |
| Apache JMeter | 5 |
| JMeter Introduction | 5 |
| Instalasi JMeter | 6 |
| Load Testing menggunakan JMeter | 7 |
| Membuat test plan | 7 |
| Add User Defined Variable | 7 |
| Add Thread Group | 8 |
| Add JMeter Element | 9 |
| Add Listener | 10 |
| Run Test | 10 |
| Integrasi JMeter dengan Selenium | 12 |
| Tugas | 19 |
| Referensi | 19 |

4.

Performance Testing & Load Testing

Learning Outcomes

Setelah selesai mempelajari bab ini, peserta mampu :

1. Memahami konsep performance test dan load test
2. Melakukan pengetesan menggunakan Apache JMeter
3. Membuat Script automasi JMeter

Outline Materi

1. Pengenalan Performance Testing & Load Testing
2. Apache JMeter
3. Load Testing Menggunakan jMeter
4. Integrasi jMeter dengan Selenium

4.1. Performance testing & Load testing introduction

4.1.1. Performance Testing

Performance testing adalah jenis pengujian untuk memastikan perangkat lunak akan bekerja dengan baik di bawah beban kerja yang diharapkan. Tujuan utamanya bukan untuk mencari bug, tapi untuk mengeliminasi performance bottleneck. Fokus dari Performance Testing, yaitu :

- Speed - menentukan apakah aplikasi merespon dengan cepat
- Scalability - menentukan apakah jumlah maksimum user load dapat ditangani
- Stability - menentukan apakah aplikasi stabil dengan berbagai beban

4.1.2. Load Testing

Load testing adalah teknik performance testing yang mana respon sistem diukur dalam berbagai load condition. Pengujian ini membantu menentukan bagaimana software berperilaku ketika beberapa user mengakses software secara bersamaan. Load testing diperlukan untuk membuat simulasi akses aplikasi web / website secara simultan. Cara ini lebih baik dibandingkan dengan harus mengundang sekian belas, atau puluh orang sekaligus untuk mengakses sebuah website.

4.2. Apache JMeter

4.2.1. JMeter Introduction

JMeter atau The **Apache JMeter™** adalah aplikasi open source berbasis Java yang dapat dipergunakan untuk performance test. Bagi seorang QA Engineer jMeter bisa digunakan untuk melakukan load/stress testing Web Application, FTP Application dan Database server test. Hal ini dapat digunakan untuk mensimulasikan beban berat pada server, sekelompok server, jaringan atau objek untuk menguji kekuatan atau untuk menganalisa kinerja secara keseluruhan di bawah jenis beban yang berbeda.

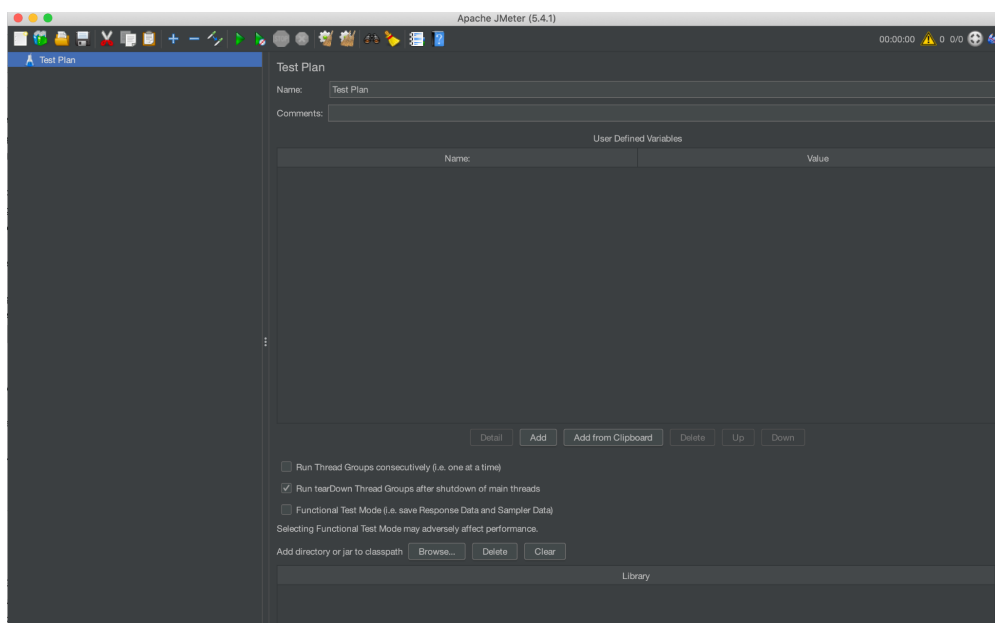
jMeter bisa dijalankan dengan 2 cara, dengan GUI atau non-GUI (Command line). Untuk beginner lebih baik saya sarankan menggunakan cara yang pertama. Mudah dan tanpa melakukan scripting tertentu. Tinggal membuat Test Plan, mengisi berapa thread & sample yang akan diujicobakan, running dan menganalisa hasil/report.



4.2.2. Instalasi JMeter

Berikut langkah - langkah Instalasi JMeter:

1. Download JMeter dari https://jmeter.apache.org/download_jmeter.cgi dan ekstrak file
2. Untuk menjalankan JMeter, double-click file jmeter.bat yang ada di folder bin. Sebelum itu, pastikan sudah menginstall JDK terlebih dahulu
3. Tunggu sampai jmeter GUI muncul seperti gambar dibawah ini.



4. Jika menggunakan Mac dapat mengikuti langkah-langkah instalasi dari sini <https://octoperf.com/blog/2017/10/26/how-to-install-jmeter-mac/>

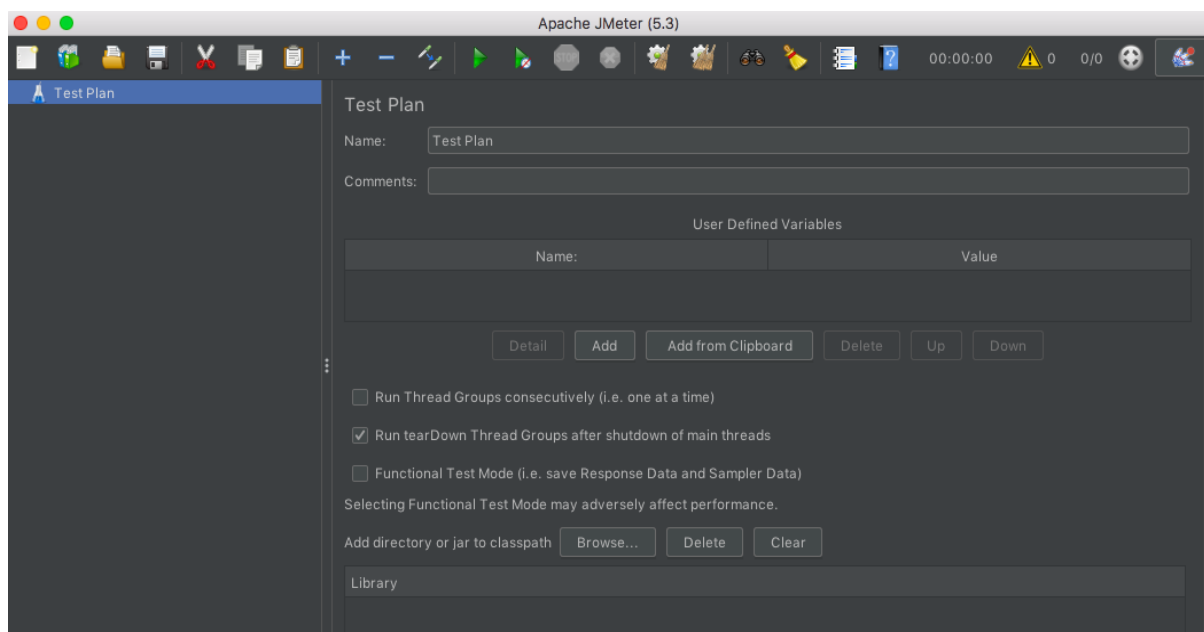
4.3. Load Testing menggunakan JMeter

Langkah - langkah pengujian menggunakan JMeter adalah sebagai berikut:

4.3.1. Membuat test plan

1. Create test plan

Setelah instalasi dan membuka file jmeter.bat, ubah nama test plan sesuai project yang diuji "Cilsy Project" lalu Save

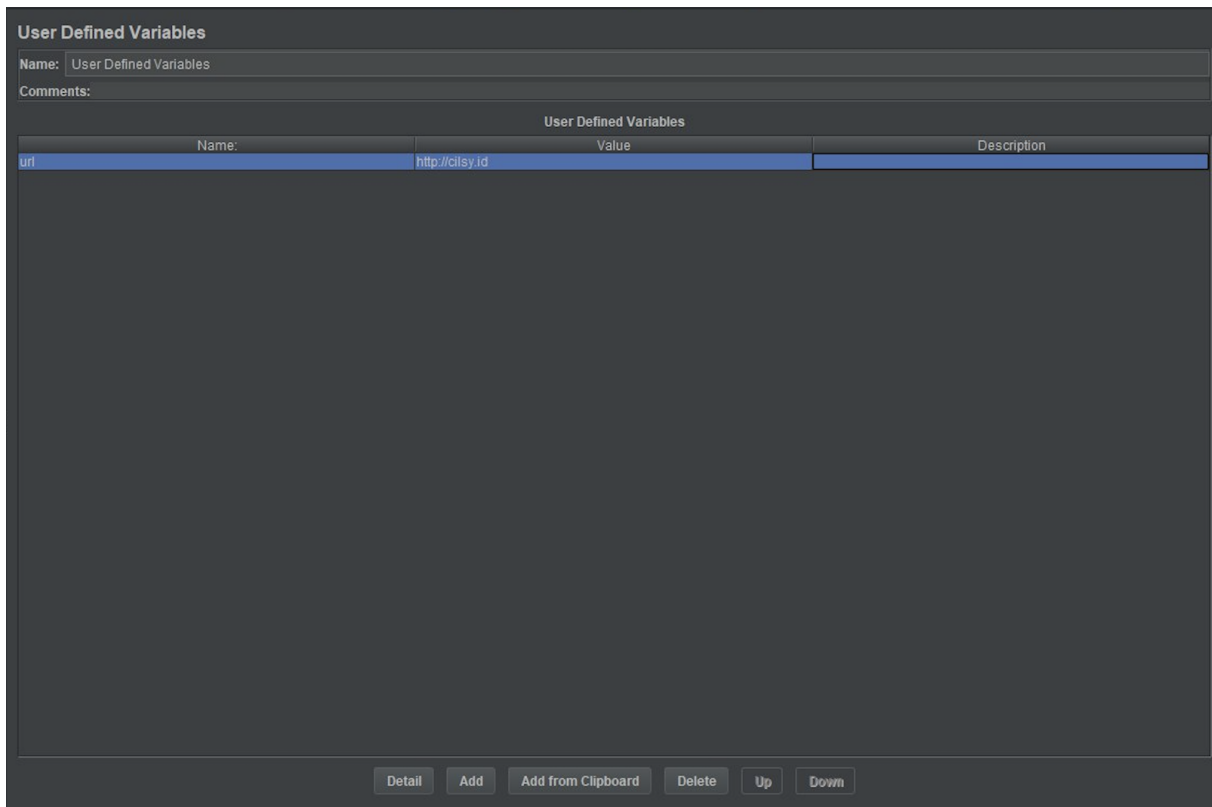


4.3.1.1. Add User Defined Variable

Di node ini kita akan menambahkan informasi global yang sering digunakan pada saat testing seperti informasi host dan port.

Untuk menambahkan node `User Defined Variables`,

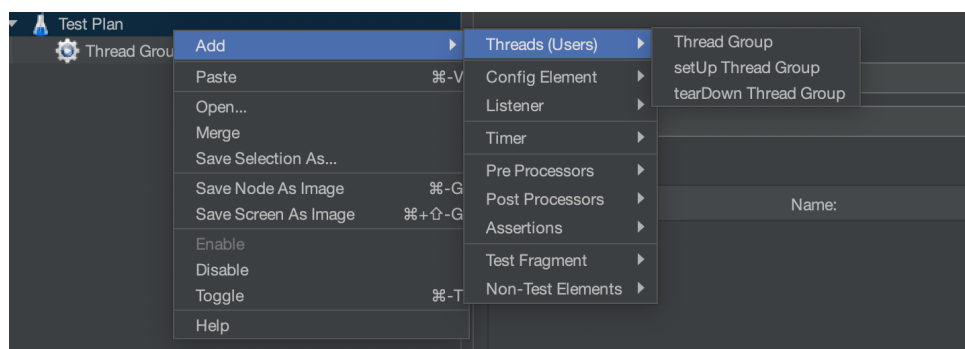
- a) Klik kanan node Test Plan (Cilsy Project) -> Add -> Config Element -> User Defined Variables.
- b) Kemudian lakukan pengaturan *User Defined Variables* seperti dibawah ini



4.3.1.2. Add Thread Group

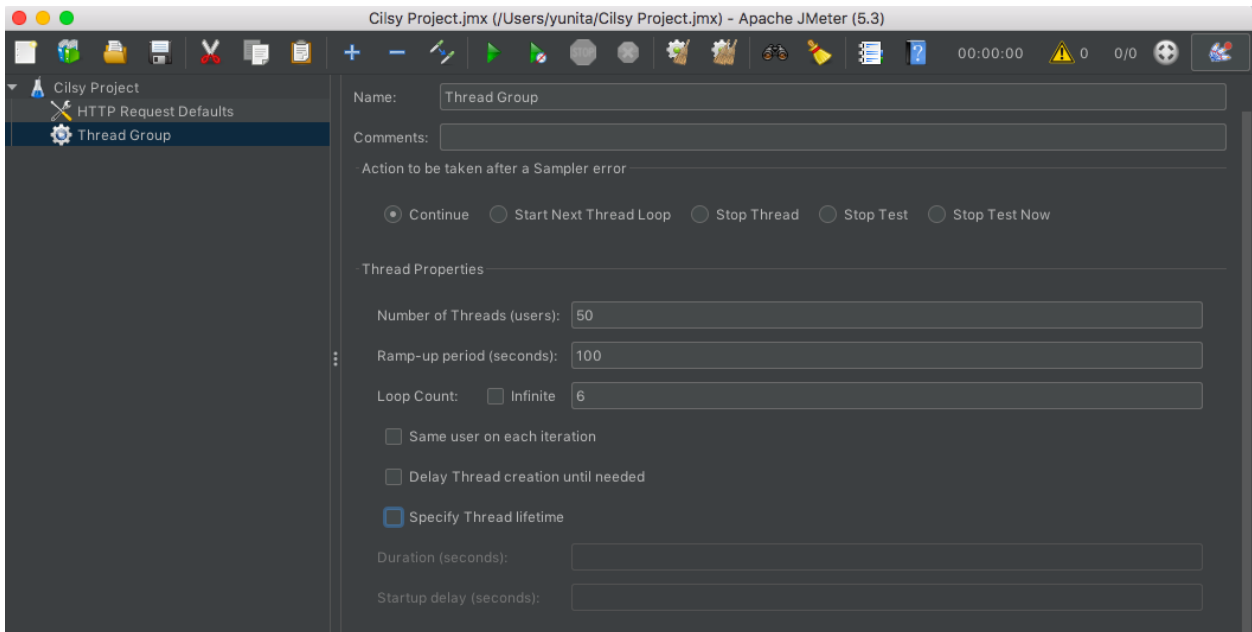
Yaitu menambahkan trafik/user visitor ke dalam komponen yang mau di test. Langkahnya :

1. Klik Kanan Test Plan
2. Add > Threads (Users) > Thread Group



3. Dalam kontrol panel Thread Group, Entri pada Thread Properties :
 - a. Number of threads (users) : isi berapa user/visitor yang akan mengakses web.

- b. Ramp-Up period (in seconds) : isi berapa waktu delay antara user satu dengan yang lainnya dalam mengakses web.
- c. Loop Count : waktu eksekusi, bertahap atau seterusnya.



Dari setting di atas, kita akan membuat skenario performance test seperti berikut :

- Jumlah user sebanyak 50 orang
- Setiap 2 detik (100/50), akan mengirimkan 6 request ke server.
- Total jumlah sample = 300 (50 x 6)

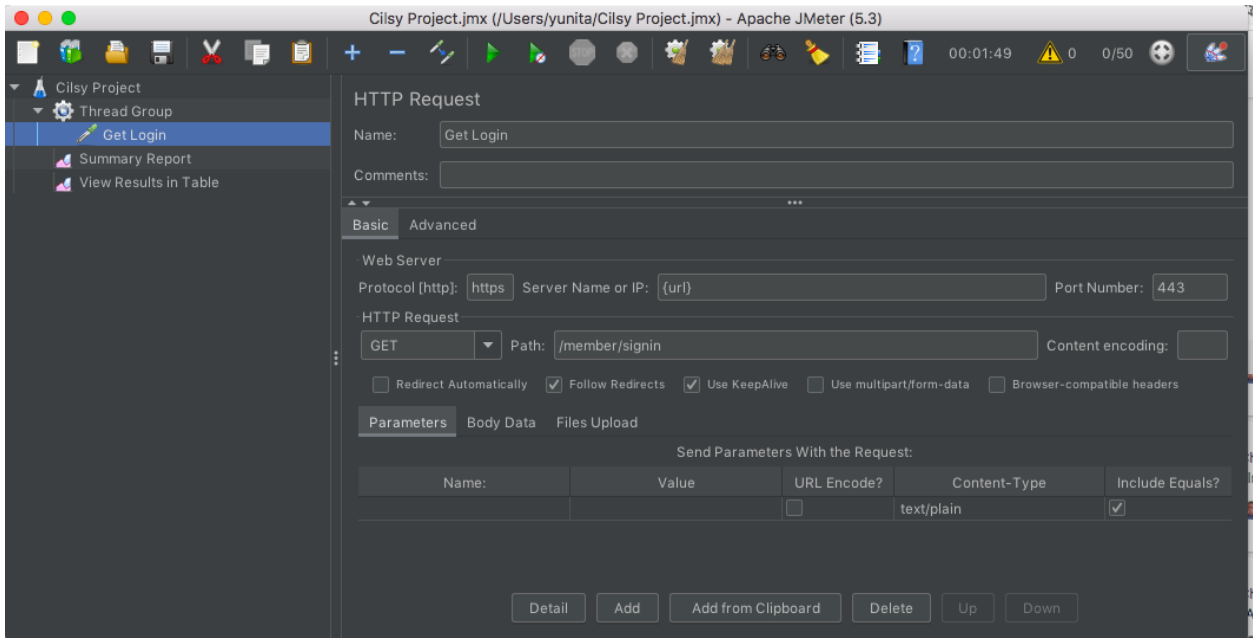
Skenario performance test ini bisa diganti-ganti nilainya sesuai kebutuhan.

4.3.1.3. Add JMeter Element

Yaitu menambahkan web server/IP Address yang akan dites. Caranya :

- A. Klik Kanan Threads Group
- B. Add > Config Element > HTTP Request
- C. Pada Web Server isi Server Name atau IP dan Portnya, atau gampangnya isi website/url yang akan dites. Karena sebelumnya kita sudah mendefinisikan nama url dan port di

node User Defined Variables, di node ini kita tinggal panggil variabel tersebut dengan format : `${NAMA_VARIABEL}`.



4.3.1.4. Add Listener

Menampilkan proses dan hasil test secara grafis atau bentuk tabel. Caranya :

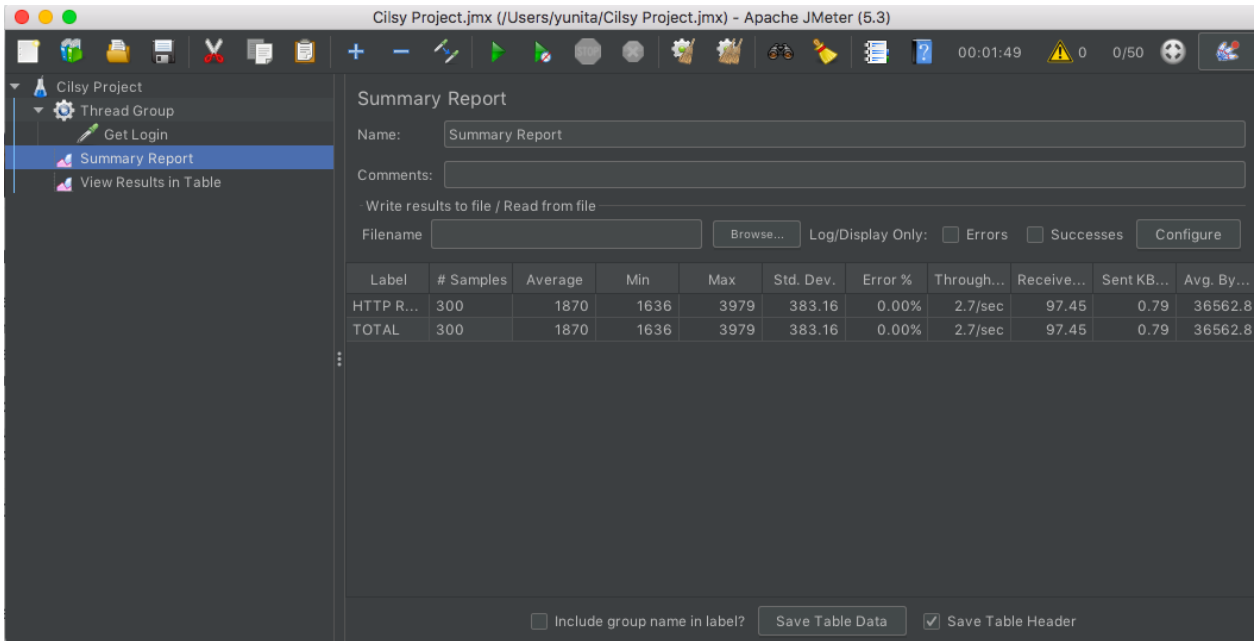
- Klik Kanan Test Plan
- Add > Listener > Summary Result
- Add > Listener > View Results in Table

4.3.2. Run Test

Untuk menjalankan Test secara otomatis. Caranya :

- Simpan terlebih dahulu Test Plan yang telah kita buat di File > Save (Ctrl + S).
- Klik Run atau Ctrl + R, jMeter akan mulai mensimulasi sejumlah user dalam mengakses web server yang telah ditentukan

Berikut hasil dari menjalankan test



Summary Report

Name: Summary Report

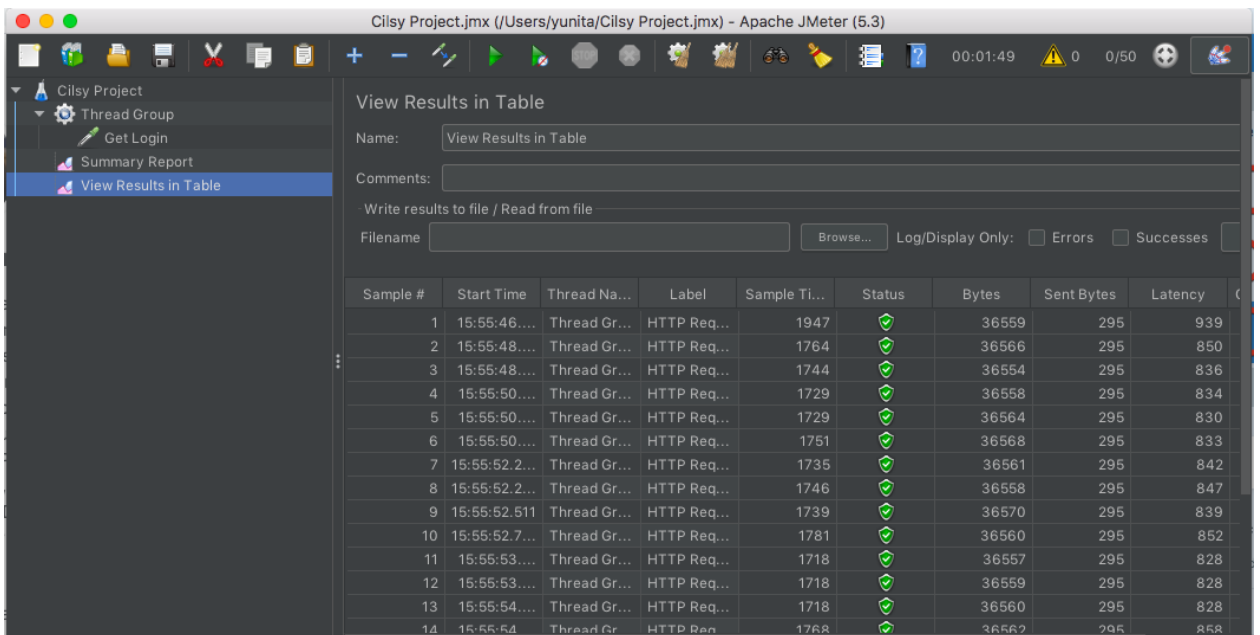
Comments:

Write results to file / Read from file

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

| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Through... | Receive... | Sent KB... | Avg. By... |
|-----------|-----------|---------|------|------|-----------|---------|------------|------------|------------|------------|
| HTTP R... | 300 | 1870 | 1636 | 3979 | 383.16 | 0.00% | 2.7/sec | 97.45 | 0.79 | 36562.8 |
| TOTAL | 300 | 1870 | 1636 | 3979 | 383.16 | 0.00% | 2.7/sec | 97.45 | 0.79 | 36562.8 |

☐ Include group name in label? ☒ Save Table Header



View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

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

| Sample # | Start Time | Thread Na... | Label | Sample Ti... | Status | Bytes | Sent Bytes | Latency |
|----------|-------------|--------------|-------------|--------------|--------|-------|------------|---------|
| 1 | 15:55:46... | Thread Gr... | HTTP Req... | 1947 | ✓ | 36559 | 295 | 939 |
| 2 | 15:55:48... | Thread Gr... | HTTP Req... | 1764 | ✓ | 36566 | 295 | 850 |
| 3 | 15:55:48... | Thread Gr... | HTTP Req... | 1744 | ✓ | 36554 | 295 | 836 |
| 4 | 15:55:50... | Thread Gr... | HTTP Req... | 1729 | ✓ | 36558 | 295 | 834 |
| 5 | 15:55:50... | Thread Gr... | HTTP Req... | 1729 | ✓ | 36564 | 295 | 830 |
| 6 | 15:55:50... | Thread Gr... | HTTP Req... | 1751 | ✓ | 36568 | 295 | 833 |
| 7 | 15:55:52... | Thread Gr... | HTTP Req... | 1735 | ✓ | 36561 | 295 | 842 |
| 8 | 15:55:52... | Thread Gr... | HTTP Req... | 1746 | ✓ | 36558 | 295 | 847 |
| 9 | 15:55:52... | Thread Gr... | HTTP Req... | 1739 | ✓ | 36570 | 295 | 839 |
| 10 | 15:55:52... | Thread Gr... | HTTP Req... | 1781 | ✓ | 36560 | 295 | 852 |
| 11 | 15:55:53... | Thread Gr... | HTTP Req... | 1718 | ✓ | 36557 | 295 | 828 |
| 12 | 15:55:53... | Thread Gr... | HTTP Req... | 1718 | ✓ | 36559 | 295 | 828 |
| 13 | 15:55:54... | Thread Gr... | HTTP Req... | 1718 | ✓ | 36560 | 295 | 828 |
| 14 | 15:55:54... | Thread Gr... | HTTP Req... | 1768 | ✓ | 36562 | 295 | 858 |

Hasil yang bisa dilihat melalui node View Results in Table kolom Status, dari node ini kelihatan apakah ada request service yang berstatus Warning. Jika ada mungkin ini bisa menjadi bahan acuan apakah service kita masih perlu di-improve lagi atau skenario performance test yang perlu diganti atau karena faktor lainnya seperti kondisi jaringan.

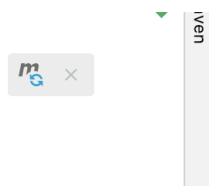
4.4. Integrasi JMeter dengan Selenium

JMeter juga dapat di integrasi dengan automation script selenium. Berikut adalah langkah-langkah pengujian menggunakan jmeter dengan selenium:

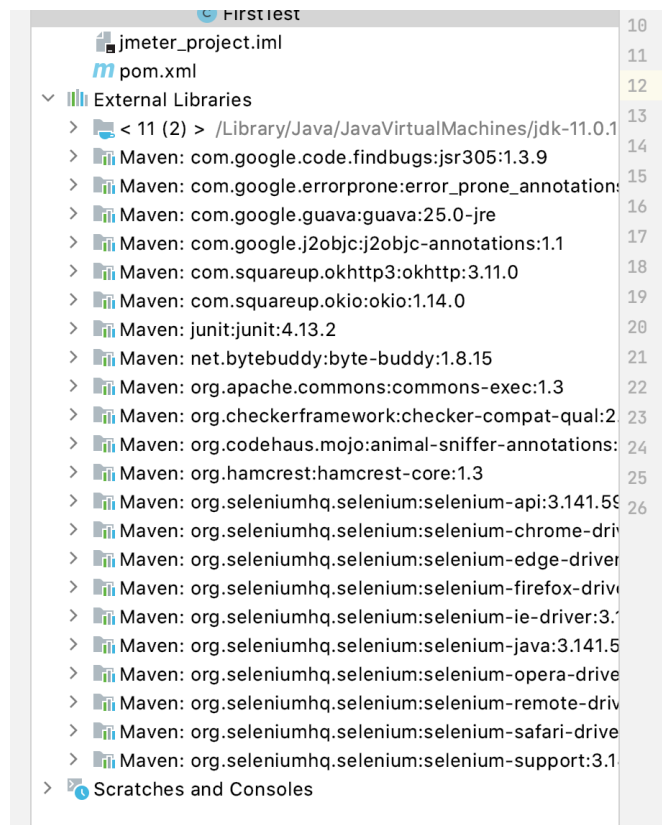
- Tools yang digunakan adalah IntelliJ IDEA
- Buat project baru menggunakan maven dengan nama jmeter_project.
- Tambahkan script berikut pada pom.xml tepat dibawah <version>.

```
<dependencies>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.141.59</version>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13.2</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

- Lalu klik button Load Maven Changes seperti gambar berikut.



- Setelah berhasil di load, akan muncul external libraries untuk junit dan selenium seperti gambar berikut.



- Create Class pada project. Klik kanan pada folder 'src' > New > Class. Beri nama class dengan 'FirstTest'
- Masukan script di bawah kedalam class

```
package com.jmeter.project;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

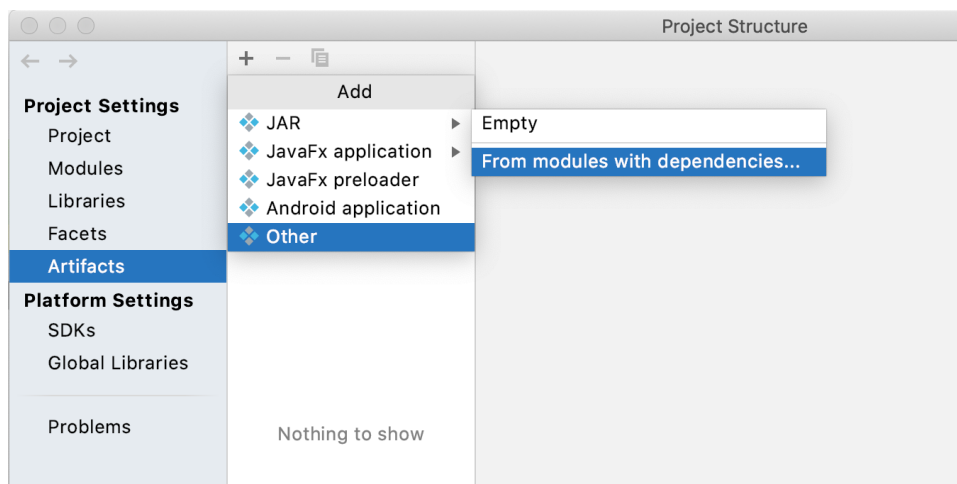
public class FirstTest {
    WebDriver webDriver = null;

    @Before
    public void beforeTest() {
        System.out.println("Before Test");
        webDriver = new ChromeDriver();
    }
}
```

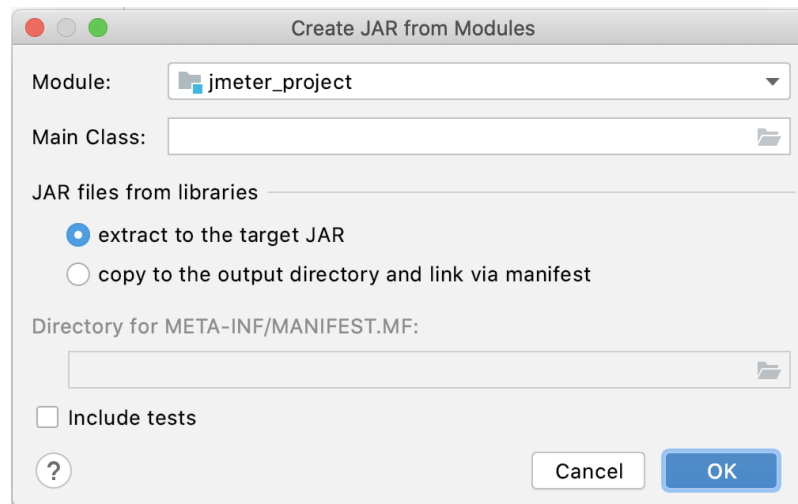
```
@Test
public void testJMeter() {
    System.out.println("Test");
    webDriver.get("https://www.cilsy.id");
}

@After
public void afterTest() {
    System.out.println("After test");
    webDriver.close();
}
}
```

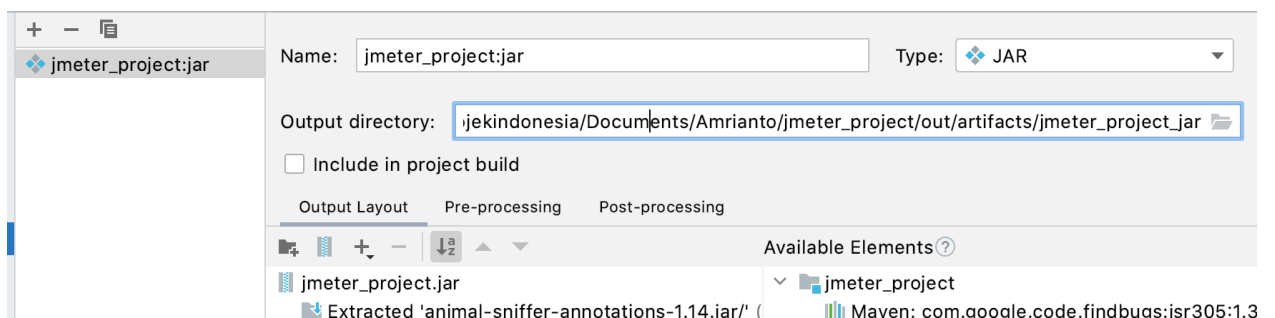
- Pastikan versi chrome adalah 91. Jika tidak, dapat mendownload versi tersebut pada link berikut:
 - Mac : <https://google-chrome.en.uptodown.com/mac/versions>
 - Windows : <https://google-chrome.en.uptodown.com/windows/versions>
- Kemudian Run script tersebut. Pastikan hasil test ialah passed (no error).
- Jika sudah passed, maka export project menjadi .jar file.
- Klik File > Project Structure... > Artifacts > klik button + > Klik JAR lalu klik From modules with dependencies...



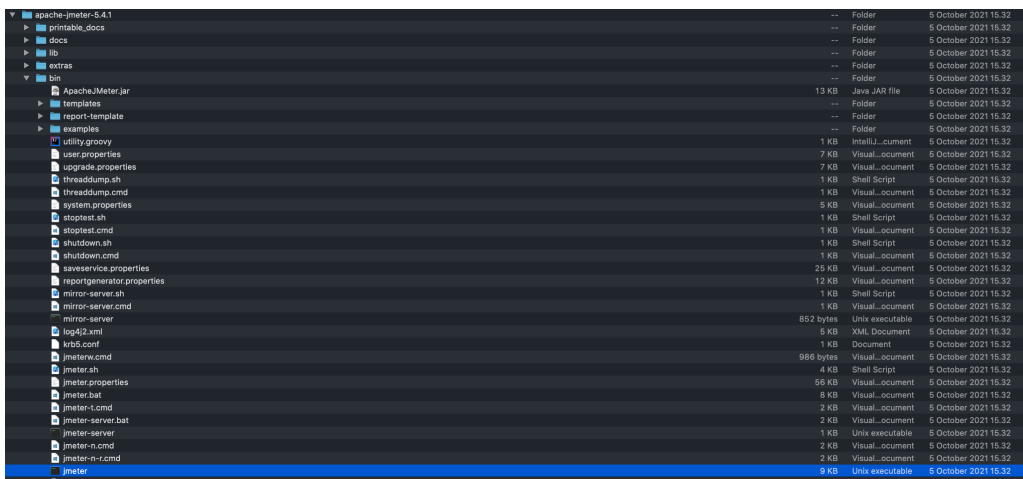
- Akan muncul tampilan seperti gambar berikut. Lalu klik button OK.

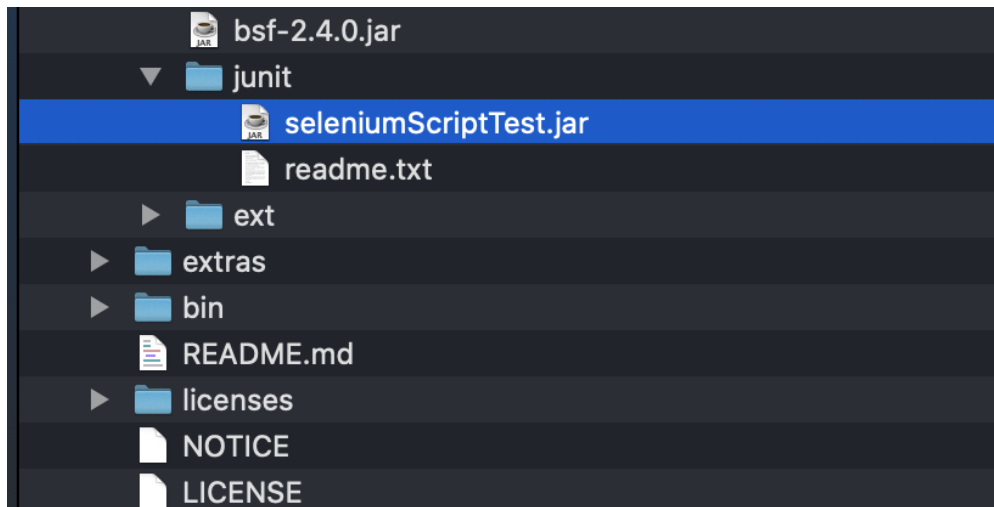


- JAR file telah berhasil dibuat, lalu copy JAR file tersebut melalui folder seperti di **Output directory**



- Copy dan paste file 'seleniumScript.jar' ke folder jmeter>lib>junit. Rename 'seleniumScript.jar' menjadi 'seleniumScriptTest.jar'





- Download file selenium JAR dari link berikut : <https://www.selenium.dev/downloads/>



Java

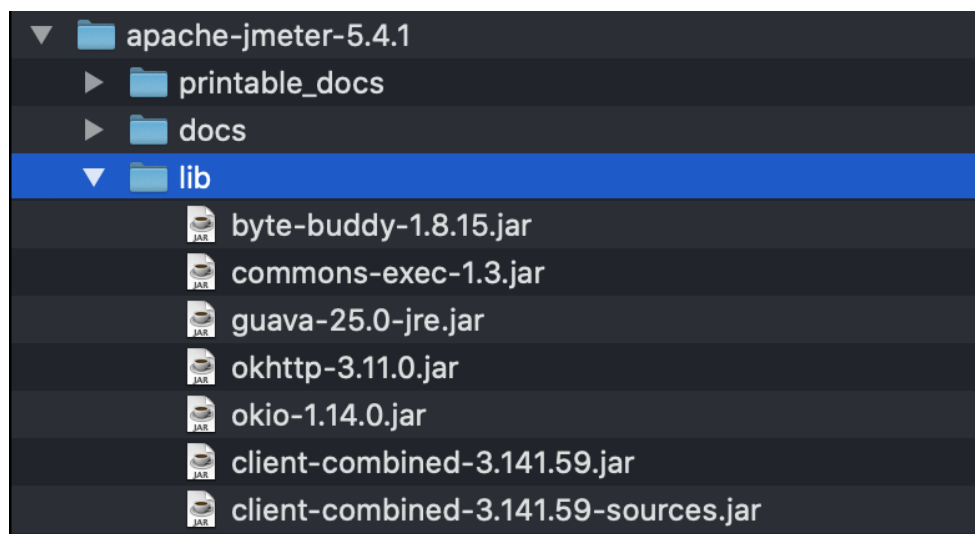
Stable: [3.141.59 \(November 14, 2018\)](#)

Release Candidate: [4.0.0-rc-3 \(October 09, 2021\)](#)

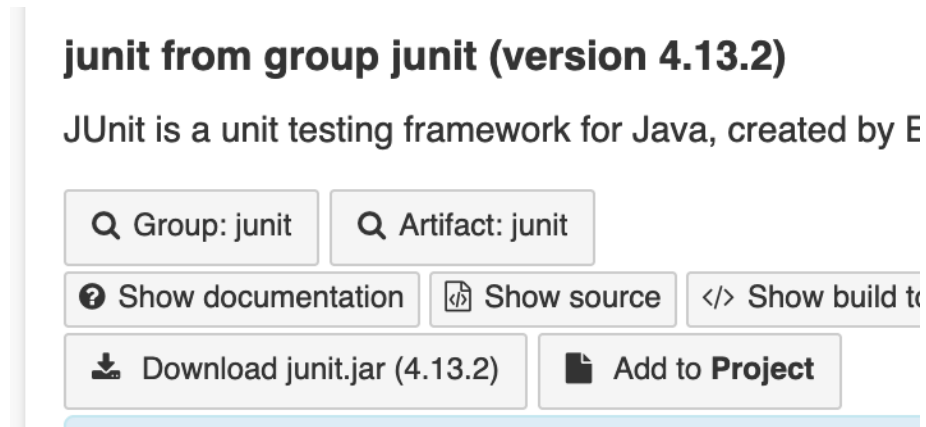
[Changelog](#)

[API Docs](#)

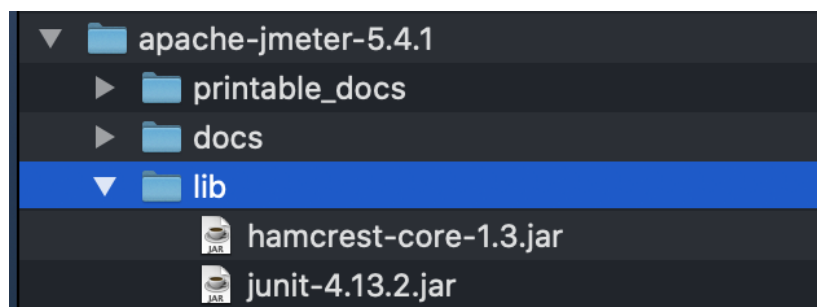
- Download untuk java. Lalu copy dan paste file-file berikut ke **jmeter/lib** dari selenium yang sudah di download sebelumnya.



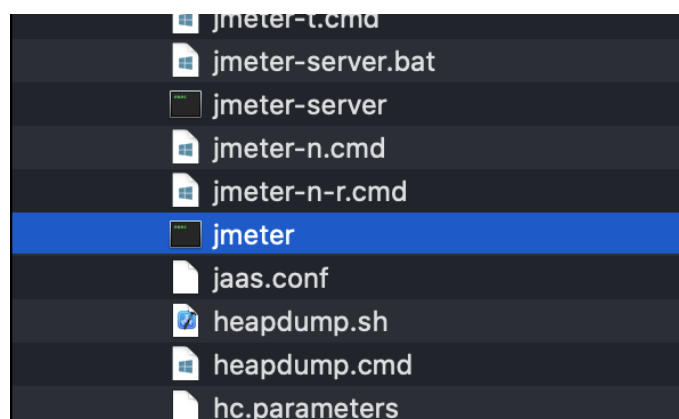
- Kemudian download file junit 4 JAR dari link berikut <https://jar-download.com/artifacts/junit/junit/4.13.2>



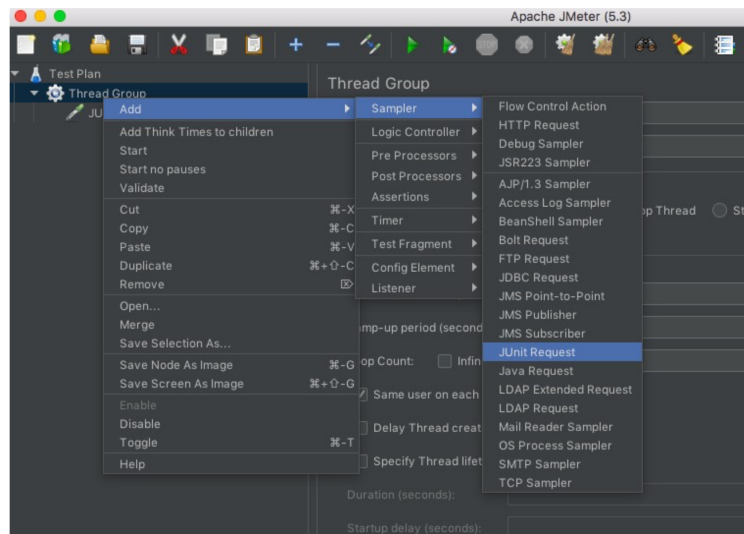
- Lalu klik button Download junit.jar (4.13.2). Setelah selesai download, copy dan paste ke folder jmeter>lib



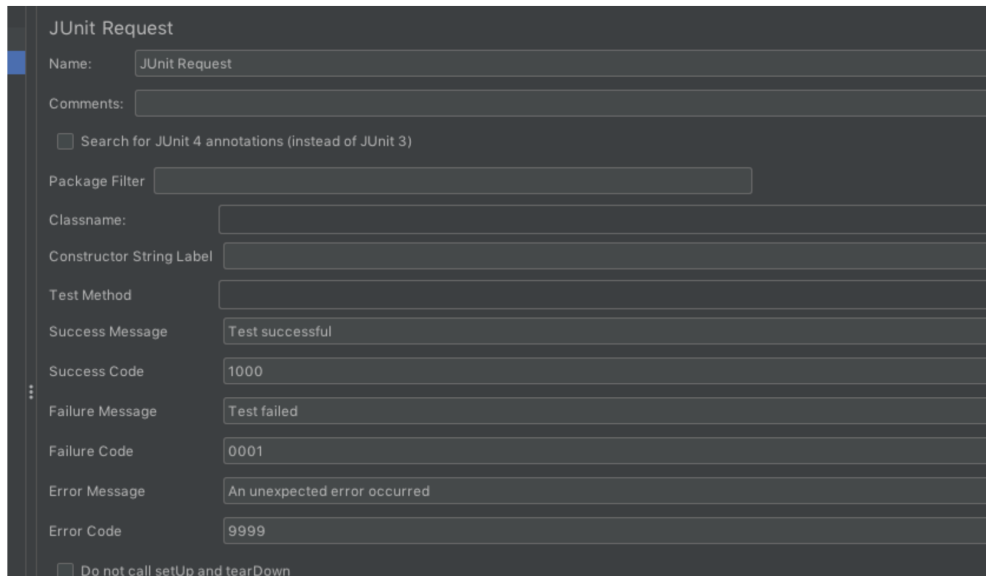
- Buka Jmeter dengan cara double klik file **apache-jmeter-5.4.1/bin/jmeter**.



- Create Test Plan seperti sebelumnya.
- Tambahkan Thread Group
- Tambahkan Junit Request. Klik kanan pada Thread Group > Sampler > JUnit Request



- Pada JUnit Request form, pilih classname sesuai nama class pada selenium “...FirstTest” dan pilih test method dengan ‘test1’
- Tambahkan seperti pada gambar dibawah



- Klik kanan thread group > Add > Listener > View Result Tree. Pada View Result Tree, kita dapat melihat hasil eksekusi test script selenium diatas.

4.5. Tugas

Temukan perbedaan dari hasil eksekusi (gunakan min 3 listener) pada dua kondisi berikut, yaitu:

- a) Jumlah user sebanyak 10000
- b) Jumlah user sebanyak 20

Masing-masing kondisi dengan test case login dan register pada website <https://tokokasur.com/> yang sama. Dan juga untuk register hanya sampai halaman verifikasi (tidak melakukan verifikasi). Hanya menggunakan JMeter dan tidak menggunakan selenium.

4.6. Referensi

<https://edusoftcenter.com/pengertian-jmeter-dan-contoh-performace-test-menggunakan-jmeter/>

<http://coding4ever.net/blog/2015/10/20/performance-test-menggunakan-jmeter/>

<https://www.ayies.com/stress-load-test-aplikasi-menggunakan-apache-jmeter/>