SAMSUNG SW ACADEMY FOR YOUTH

對外秘

# 프로젝트 명세서

## 1학기 관통 PJT 결과에 대한

## 성능 테스트 수행

목차

對外秘

# 1. 프로젝트 개요

예를 들어 여러분들이 수강 신청 웹 사이트를 만들었다고 가정해 봅시다. 이 사이트는 수강신청 기간에 최소한 수천 명 이상의 접속을 견딜 수 있어야 합니다. 이런 경우 여러분들은 웹 사이트를 구축한 후에 성능 테스트를 통해 이런 상황을 견딜 수 있다는 것을 보장해야 할 것입니다.

여러분들이 1 학기 관통 프로젝트에 구축했던 과제물에 대해 여러분의 시스템 한계 성능은 어느 정도인지 측정해 보도록 합시다.

참고로 도메인별 일반적인 요구 성능 수준은 아래와 같습니다.

| NO | 도메인 | 최대 가용 접속자(서버당) | 예상 동시 접속자(서버당) |
|---|---|---|---|
| 1 | 게임서버(베틀그라운드 등) | 200,000 유저 | 10,000 유저 |
| 2 | 커뮤니티(페이스북 등) | 20,000 유저 | 1,000 유저 |
| 3 | 일반 웹 사이트 | 4,000 유저 | 200 유저 |

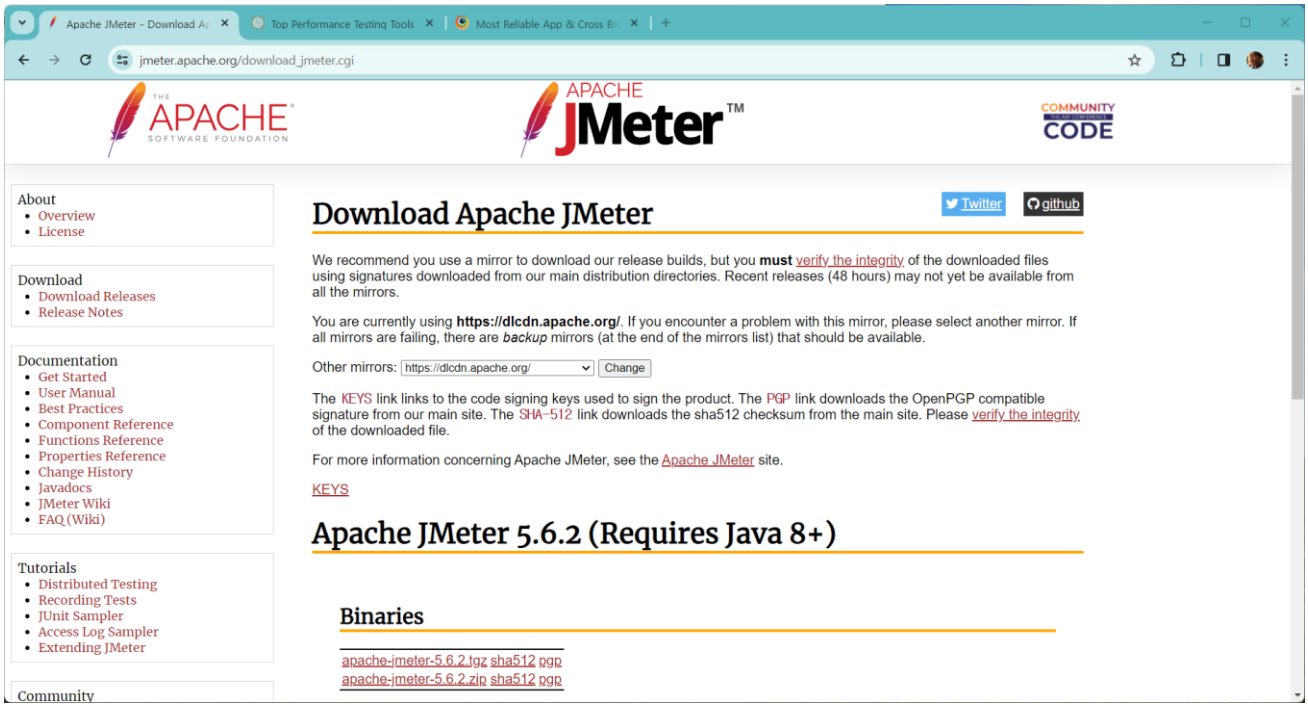
위 수치는 서버가 전문 IDC 에 분리되어 있는 상태에서 측정한 것이어서 환경이 다르면 한계 성능은 낮아질 수 있습니다.

먼저 측정대상인 여러분의 관통 프로젝트 서비스가 여러분의 PC 에 localhost 로 설치되어 있음을 전제로 하겠습니다.

# 2. 과제

여러분들이 공통 기간에 완성한 과제에 대해 성능 테스트를 수행하고 결과를 기록합니다.

성능테스트에 사용될 도구는 Jmeter(현 시점에서 5.6.2) 입니다.
(https://jmeter.apache.org/)




성능 테스트 도구는 상용 및 오픈소스 도구들이 많이 있으며, 오픈소스 계열에서는 Jmeter 가 언제나 부동의 1 위를 차지하고 있습니다.

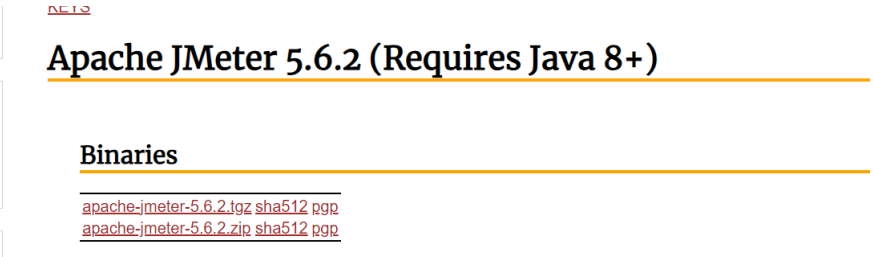
참고로 BrowserStack(https://www.browserstack.com/) 에서의 Top10 성능 테스트 도구 사용 랭킹은 다음과 같습니다.

| | |
|---|---|
| 1. | **Apache JMeter** - Open-source, supports various protocols, enables distributed testing, extensive reporting, and GUI/scripting customization. |
| | |
| 2. | **Gatling** - Open-source, efficient resource utilization, supports high concurrent loads, offers a DSL for script creation, real-time monitoring. |
| | |

| |
|---|
| 3. **BrowserStack App Performance** - For mobile app performance, offers real-device testing, network condition simulation, user flow analysis. |
| |
| 4. **LoadRunner** - Supports various applications and protocols, scalable for distributed testing, real-time monitoring, comprehensive analysis. |
| |
| 5. **BlazeMeter** - Cloud-based, supports JMeter, Gatling, Selenium WebDriver, scalable load generation, real-time monitoring, collaboration features. |
| |
| 6. **Locust** - Open-source, Python-based, supports scalable load generation, real-time web dashboard, extensible architecture. |
| |
| 7. **K6** - Open-source, uses JavaScript for scripts, supports high loads and scalability, real-time monitoring, performance checks. |
| |
| 8. **Apache Bench** - Command-line tool, simple interface, generates requests to a target URL, basic performance metrics. |
| |
| 9. **NeoLoad** - User-friendly interface, supports various technologies, live dashboards, scenario sharing. |
| |
| 10. **Tsung** - Open-source, supports massive virtual users, various protocols, real-time metrics, customization and integration options. |

Apache Jmeter 의 설치 과정은 다음과 같습니다.

- JMeter 다운로드




공식홈페이지에서 최신 버전의 zip 파일을 다운로드하여 적절한 위치에 압축을 풉니다.

실행은 ₩bin 디렉토리 밑에 jmeterw.cmd 파일을 실행시킵니다.

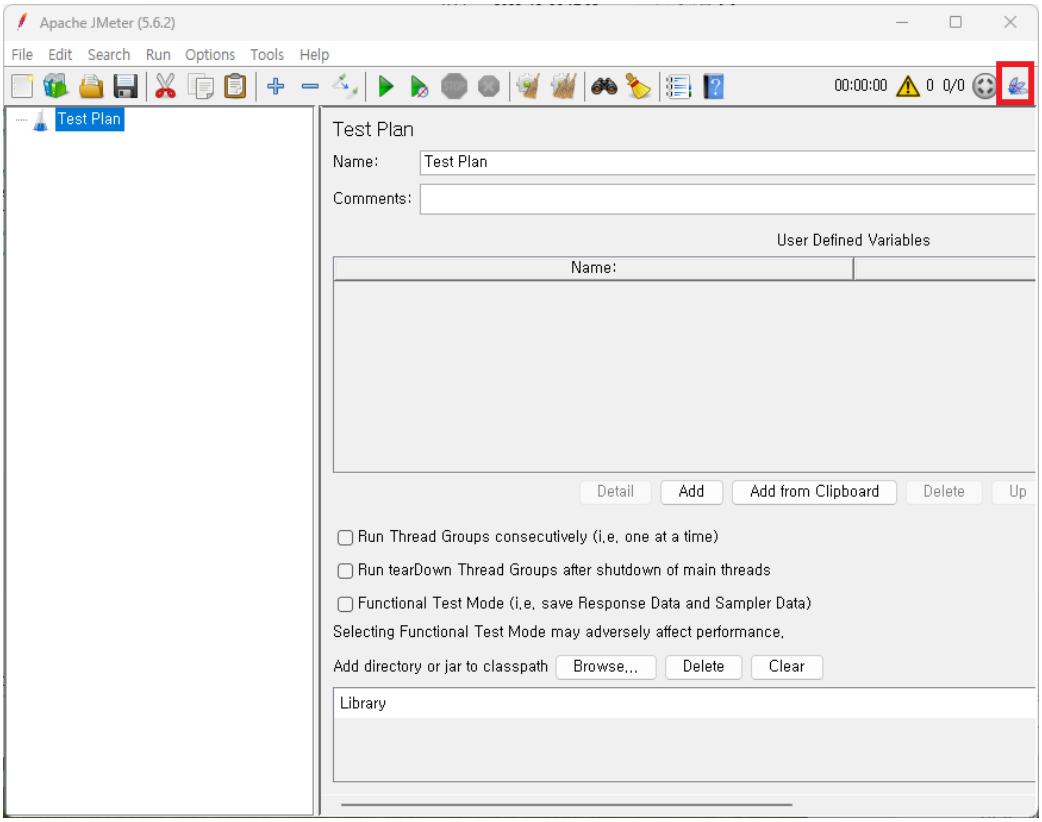
- Plugin Manager 설치

Jmeter 는 많은 플러그인을 지원하고 있습니다. 플러그인은 `lib\ext` 폴더 아래에 위치하고 있습니다.

이의 관리를 용이하게 해 주는 것이 Jmeter Plugin Manager 입니다. 이를 설치합니다.

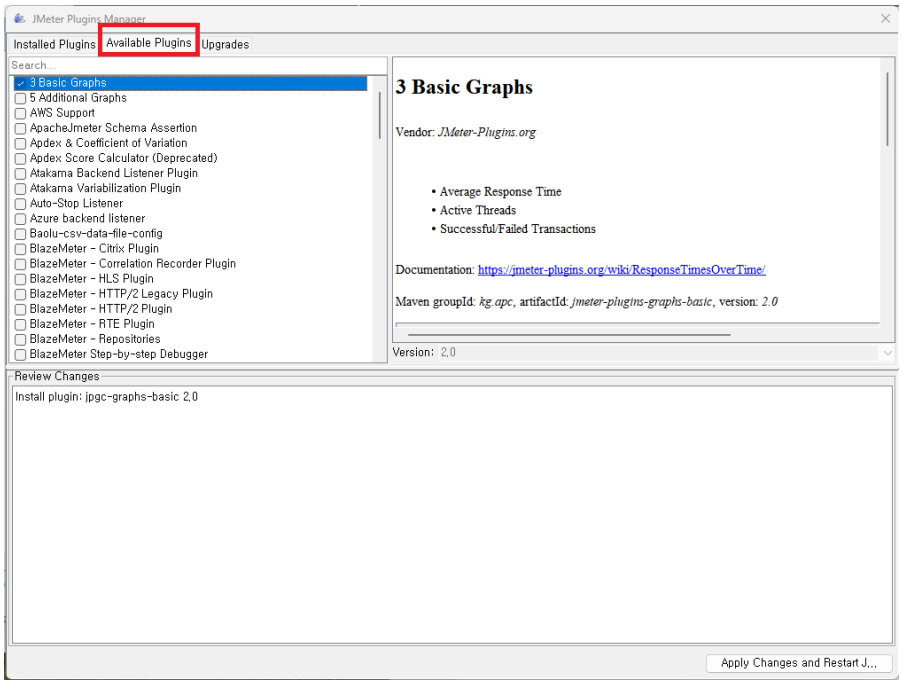
- https://jmeter-plugins.org/install/Install/ 에서 다운받아 `lib\ext` 폴더 아래에 옮겨놓고 jmeter 를 다시 시작합니다.

- 우측 상단 메뉴바에 아이콘이 생겼음을 확인할 수 있습니다.



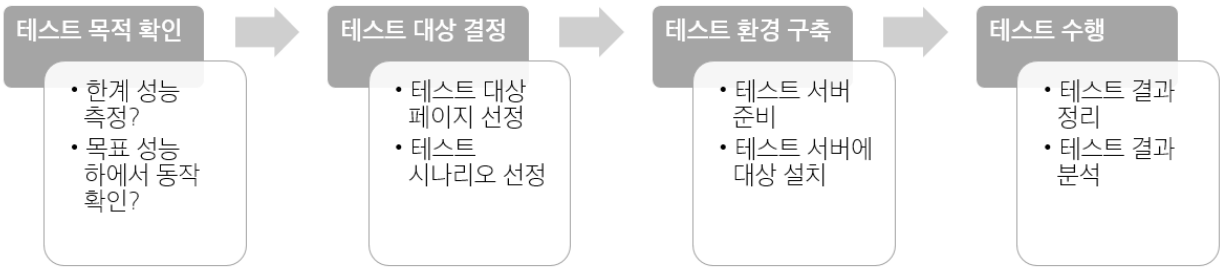

- 주요 Plugin 설치

플러그인 매니저를 실행하여 다음 플러그인을 선택하고 하단의 'ApplyChange….' 버튼을 클릭합니다.




- 3Basic Graphs : Active User, Thread, ResponseTime 의 그래프를 보여줍니다.
- Jpgc-Standard Set : 프로젝트 관리를 위한 도구들의 모음입니다.

이후 과정은 다음과 같습니다.




- 테스트 목적 확인
  - 부하 테스트의 목적을 정의합니다. 목적에 따라 부하를 발생시키는 패턴이 달라집니다.
- 테스트 대상 결정

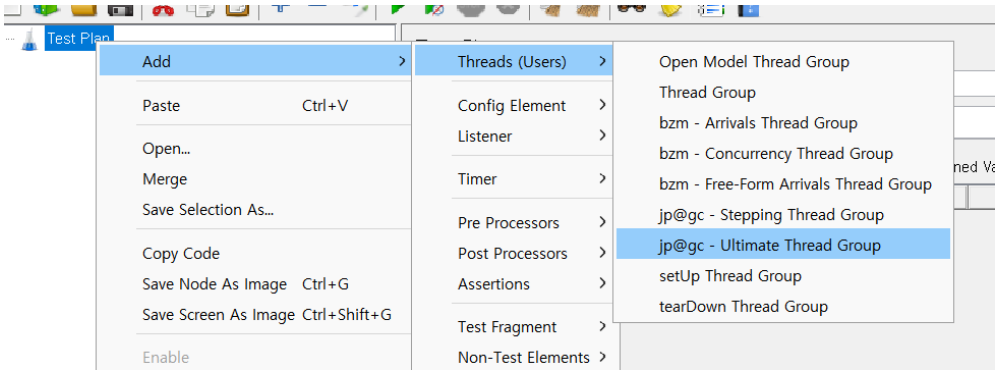
- ■ 가장 많이 사용될 것 같은 페이지들을 대상으로 선정합니다.(예: 로그인 후 랜딩 페이지 등)

- 테스트 환경 구축
  - ■ 테스트를 할 수 있는 서버(또는 PC)를 준비하고 여기에 테스트 대상의 서비스를 설치합니다.

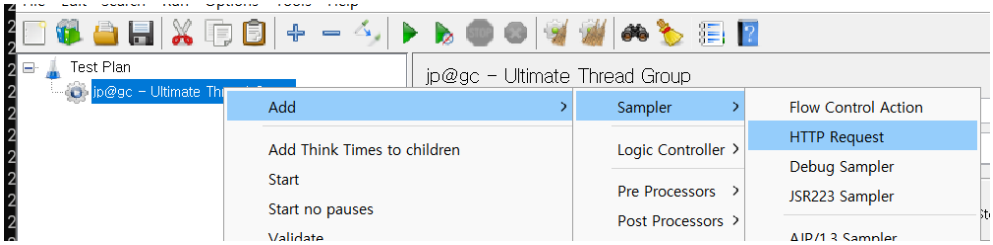
- 테스트 수행
  - ■ 테스트를 수행하고 결과를 분석합니다. 이 때 한계 부하를 예측합니다.

JMeter 테스트 계획 작성

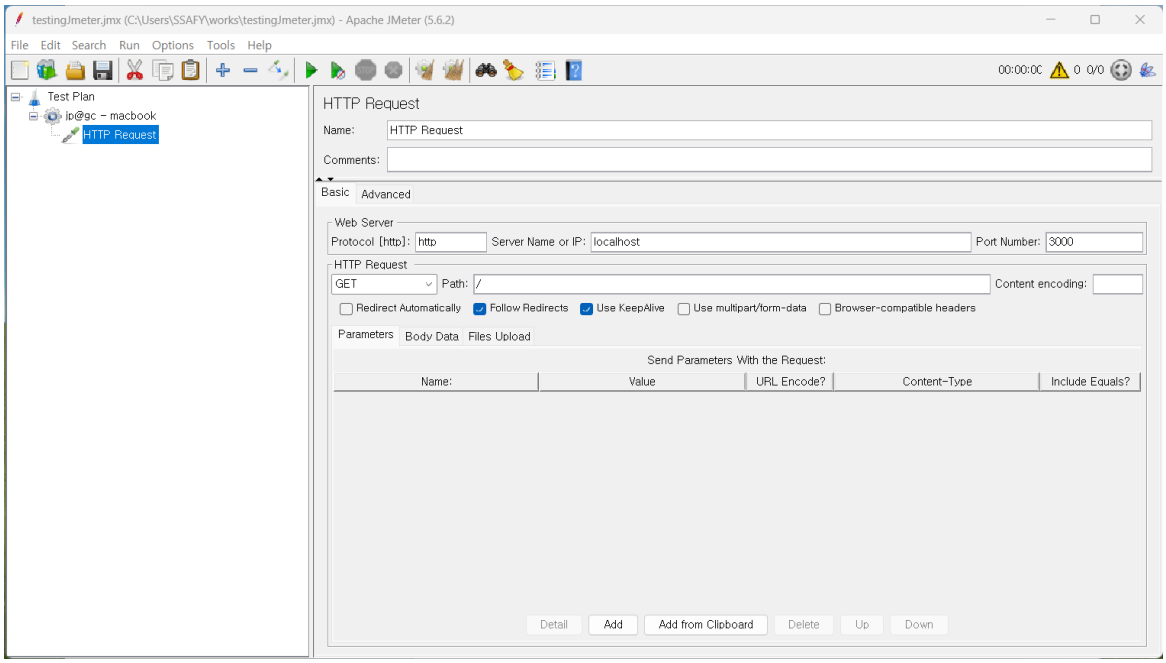
- Jmeter 를 시작하면 우측 내비게이션 영역에 'Test Plan' 에서 그림과 같이 'jp@gc - Ultimate Thread Group' 을 선택



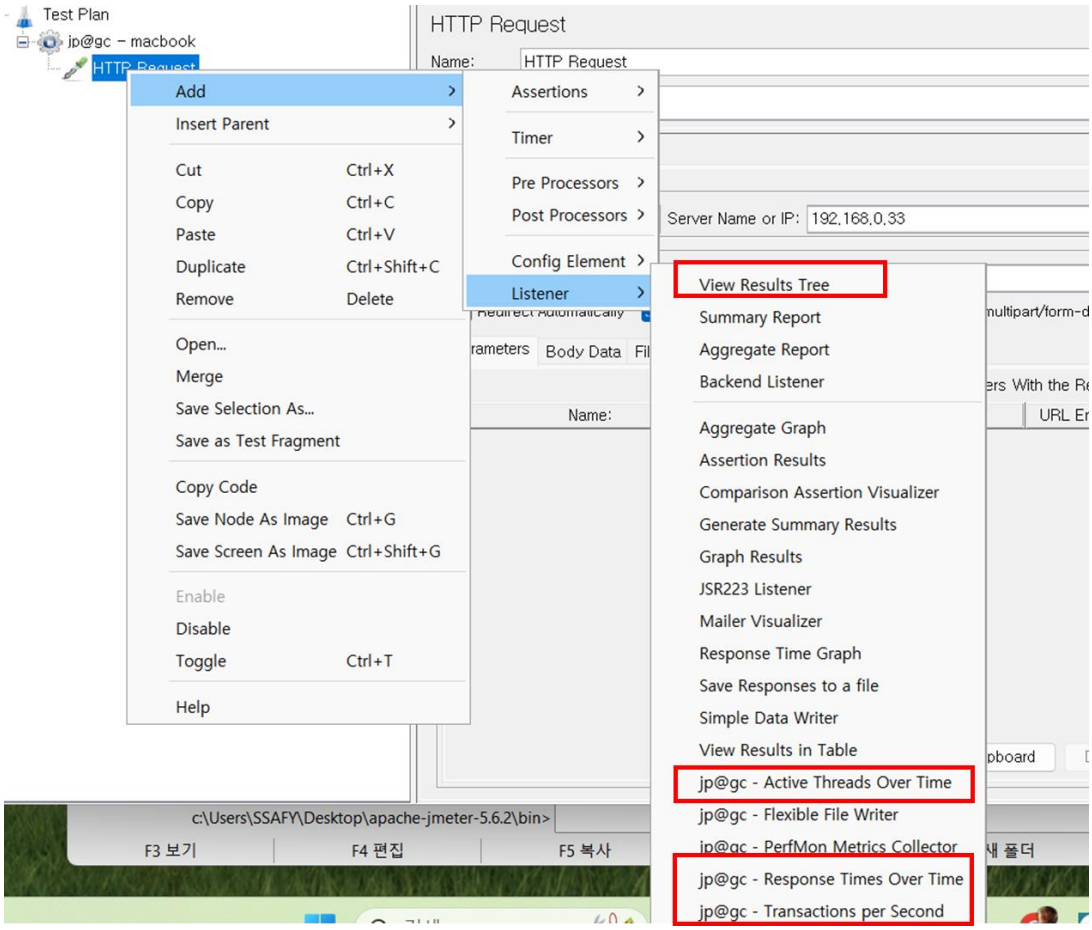

- 추가된 Ultimate Thread Group 에서 'HTTP Request' 를 다시 추가함.



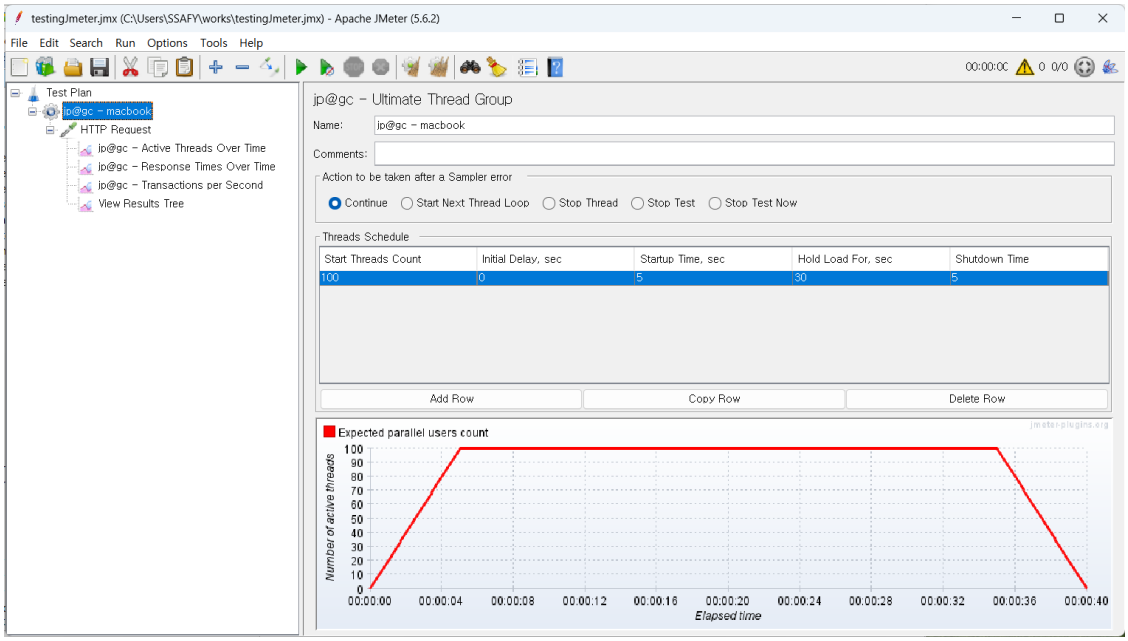
- HTTP Request 설정에서 테스트하려는 대상 서버 정보를 입력합니다. 여러분들이 PC 에 관통 프로젝트를 설치했다면 ServerName 부분은 localhost 가 될 것이고 IP 는 서비스가 사용하는 IP 를 입력합니다.




- 결과 리스너 설정 : 3 가지 그래프(Active Thread, Response Time, Transaction) 및 ViewResultTree 리스너를 추가합니다.



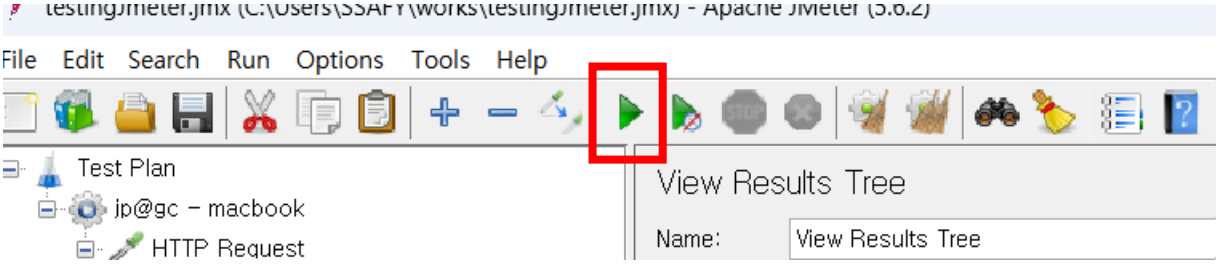

- 이제 jp@gc - ultimateThread로 다시 가서 부하 설계를 합니다. 우리가 할 것은 단순 부하 테스트 이기 때문에 아래와 같이 부하를 설계 합니다.



스케줄 설정값은 다음과 같이 입력합니다.

- Start Thread Count : 유저 쓰레드 수 입니다. 100 명으로 설정 합니다.
- Initial Delay : 사용자가 접속하기 전 까지의 지연 시간입니다. 일단 지연은 없다고 생각하고 0 으로 설정합니다.
- StartUp Time : 0 으로 설정하면 처음부터 100 명이 한꺼번에 접속하는 것입니다. 일단 5 초로 설정하여 5 초동안 사용자가 서서히 100 명까지 증가하는 걸로 하겠습니다.
- Hold Load For : 100 명의 접속이 완료되었으면 이 사용자들이 얼마나 접속을 유지하는지 입니다. 일단 30 초로 두겠습니다.
- Shutdown Time : 100 명의 접속을 해제하는데 걸리는 시간입니다. 0 이면 100 명이 한꺼번에 접속을 취소하는 의미입니다. 여기서는 5 초로 두어 5 초동안 사용자가 서서히 0 명으로 접속을 해제하는 것으로 하겠습니다.

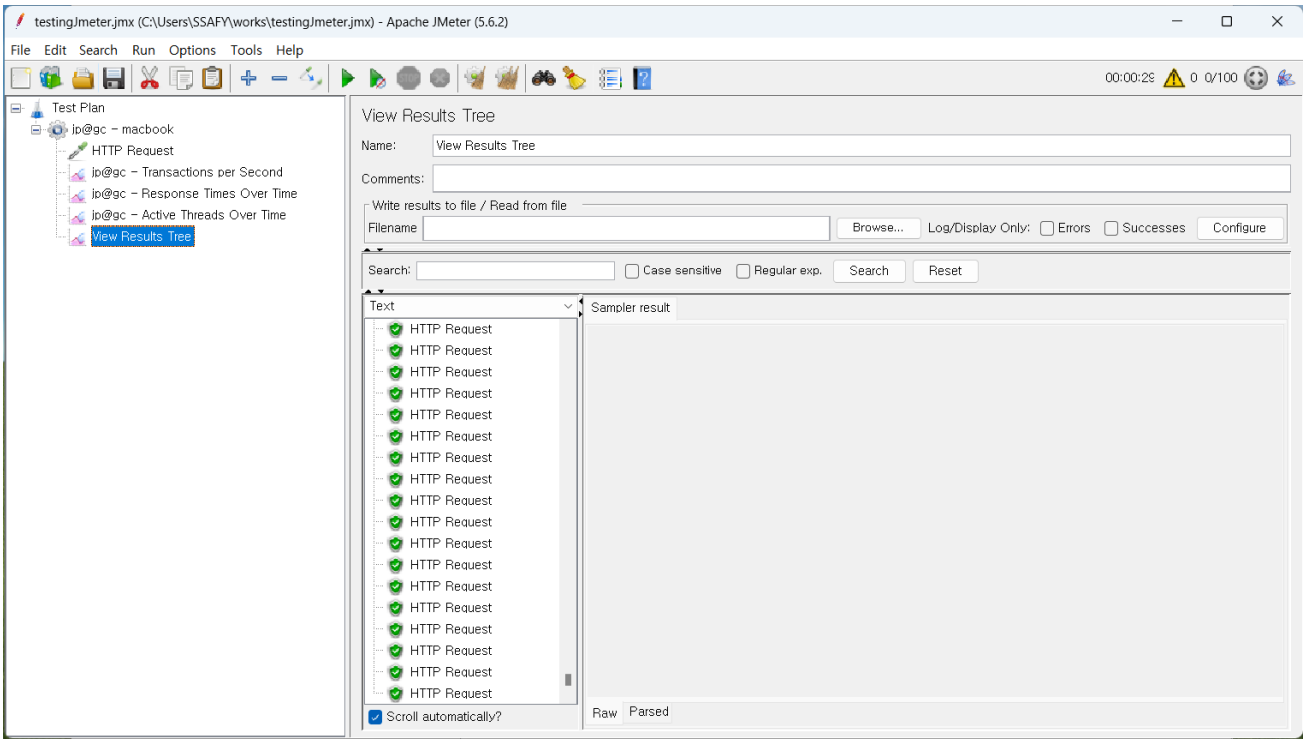
이제 설정이 완료가 되었습니다. 여러분들의 서비스를 PC 에서 띄운 후 서비스 랜딩 페이지 정상 구동 여부를 확인하시고 이상이 없으면 메뉴바에서 'play' 를 눌러 주시면 테스트가 시작됩니다.




- 확인해야 할 사항

  ■ View Result Tree

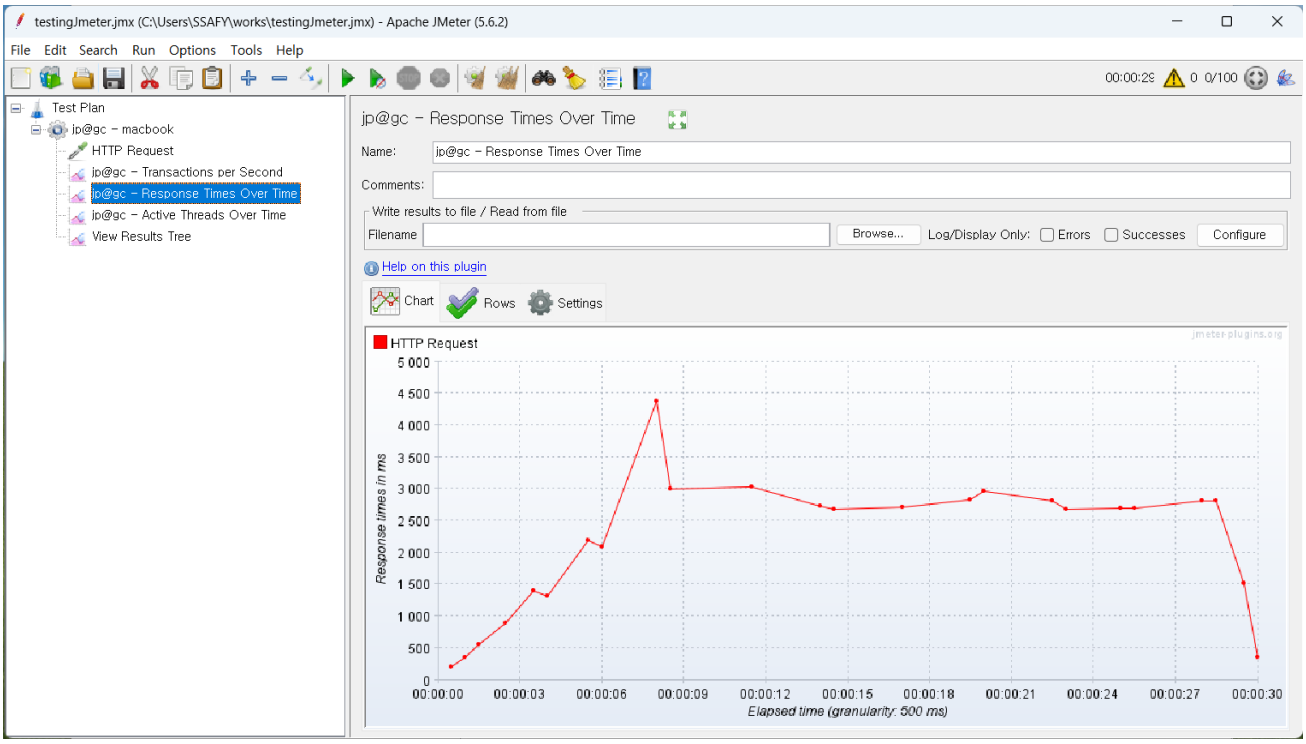
  - 테스트 수행 중 HTTP 오류가 발생했는지를 체크합니다. 테스트 수행 시작시에는 정상적이었다가 나중에 여러가지 문제로 서버가 터질 경우 빨간색으로 바뀝니다.
  - 테스트 수행 중 문제가 없을 경우에는 처음부터 끝까지 모두 정상으로 표시가 됩니다.
  - 테스트 수행 시작부터 오류가 뜨는 경우에는 서버 설정 등 서비스 자체의 문제이므로 서비스 설정을 다시 체크합니다.




<View Result Tree의 정상적인 모습>

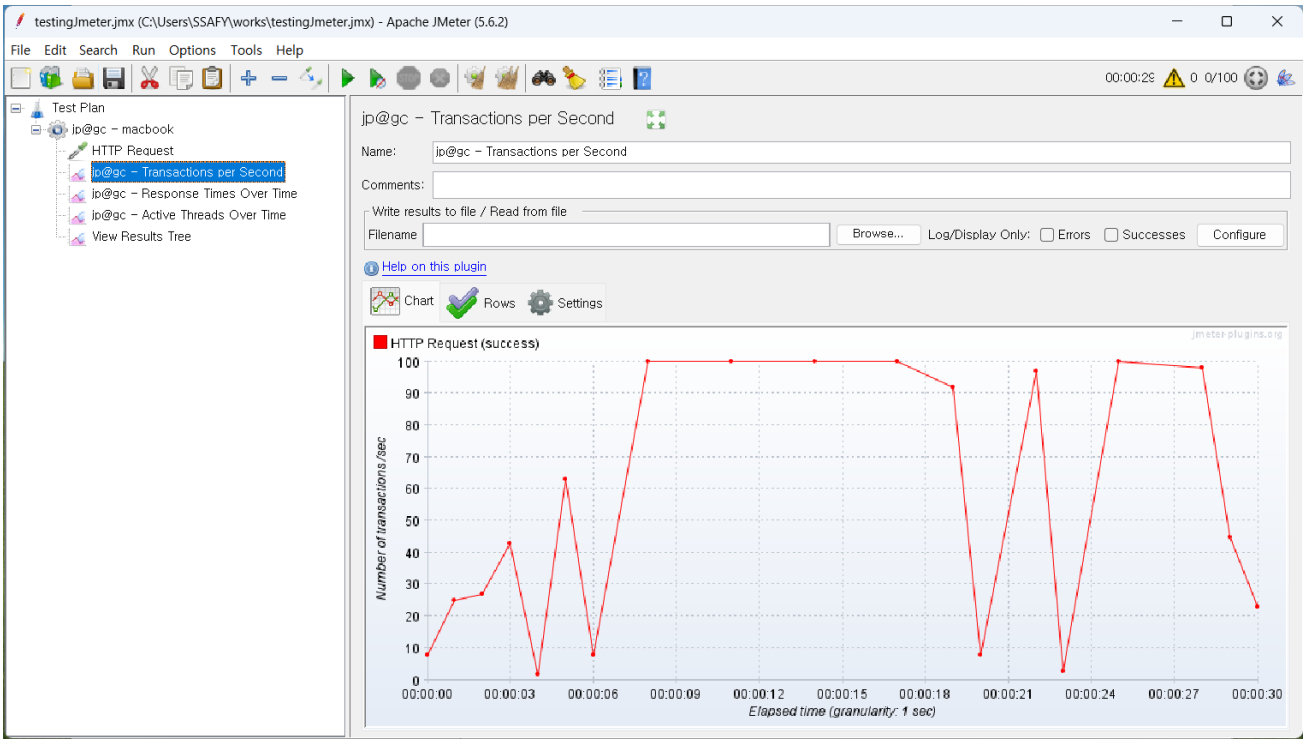
■ Response Time Over Time

◆ 응답 시간 체크입니다. 보통 500ms 아래로 들어와야 하며 이를 많이 넘어가는 경우에는 응답이 단순히 과부하로 느린 경우인지, 아니면 서비스가 터져서 정상작동이 안된것인지를 확인해야 합니다(View Result Tree와 같이 비교). 과부하로 느린경우이면 느리지만 서비스는 수행 가능한 상태입니다.




<Response Time 결과 그래프 : 서비스와 부하 발생을 같은 PC 에서 수행하여 전반적으로 RT 가 높음(페이지 속도가 느림)>

- Transaction per Second
  - 초당 트랜잭션의 수 입니다. 아래 그림에서는 100 이하로 유지되면서 종료되었음을 알 수 있습니다.




<transactrion per Second 그래프>

# 3. 산출물 제출

산출물은 아래 내용을 https://lab.ssafy.com/s10-study/self-project/ 의 "산출물 제출 가이드.docx" 를 참조하여 제출 바랍니다.

1. 측정 결과(그래프는 이미지 캡쳐)

- 본인의 PC 에서 1 학기 관통 프로젝트의 랜딩 페이지(서비스의 첫 화면)를 대상으로 테스트를 합니다.

- 측정은 Active User 를 100 부터 시작하여 100 씩 증가시켜 측정합니다. 예를 들어 아래와 같이 측정되었다면, 한계 부하는 400tps 입니다.

| Active User | 서비스 |
|---|---|

| | |
|---|---|
| 100 | 죽지 않음 |
| 200 | 죽지 않음 |
| 300 | 죽지 않음 |
| 400 | 서비스 안됨 |

- 

2. 한계부하(TPS)기술

- 만일 과부하로 인해 서비스 불능 상태가 된다면 지표 곡선은 다음과 같은 경향을 보일 것입니다.

  1. View Result Tree 의 결과가 갑자기 Fail 이 뜨기 시작한다.

  2. TPS 가 증가하다가 안정적으로 수렴하지 못하고 급격하게 우상향을 보인다.

  3. 응답속도는 갑자기 빨라지는데 이는 실패시에 웹 서버에서는 Fail 페이지를 내보내게 되며 이 페이지는 단순 html 페이지이기 때문에 반응속도가 빠르기 때문이다.

- 한계 부하 발생시 TPS 그래프를 캡쳐해 주세요

- 한계 부하 발생시 응답속도 그래프를 캡쳐해 주세요