Бригеда Валентина Домашнє завдання

**Урок 24\_Основи навантажувального тестування за допомогою Apache Jmeter**

**Перший рівень**

1. Створи набір з GET, POST, PUT, PATCH, DELETE запитів до [JSONPlaceholder](https://jsonplaceholder.typicode.com/), які надсилаються протягом 10 секунд у 3 ітерації.

2. До кожного з запитів застосуй 3 різних assert’и.

3. Результати виконання тестів мають бути отримані за такими Listener’ами:

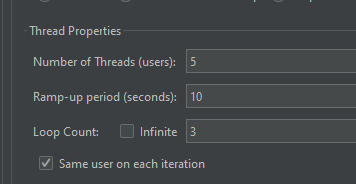
* View Results in Tree
* Summary Report.

4. Опиши висновки щодо результатів тестування (базуючись на репортерах) в окремому файлі.

Висновки щодо результатів тестування я виконала базуючись на репортерах View Results in Tree і Summary Report.

Навантажувальне тестування відбувалося з такими параметрами:

3 ітерації, 10 секунд, 15 юзерів



**Метод POST**: Response відрізняється від Response по документації, а саме: містить лише рядок з id, і не містить title, body userId як по документації

|  |  |
| --- | --- |
| Очікуваний Response по документації | Фактичний Response |
| E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_3.png | E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_4.png |

**Метод GET:** Response відрізняється від Response по документації, а саме: useId пишеться в 1-му, а не останньому рядку, як по документації

|  |  |
| --- | --- |
| Очікуваний Response по документації | Фактичний Response |
| E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_1.png | E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_2.png |

**Метод PUT:** Response відрізняється від Response по документації, а саме містить лише один рядок з id, і не містить title, body і userId.

|  |  |
| --- | --- |
| Очікуваний Response по документації | Фактичний Response |
| E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_7.png | E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_8.png |

**Метод PATCH**: Response відрізняється від Response по документації, а саме title містить не задане значеня ‘foo’, userId відображається в 1-му рядку, а не останньому

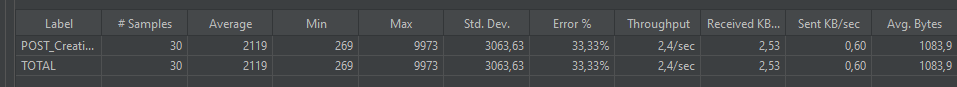
|  |  |
| --- | --- |
| Очікуваний Response по документації | Фактичний Response |
| E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_10.png | E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_9.png |

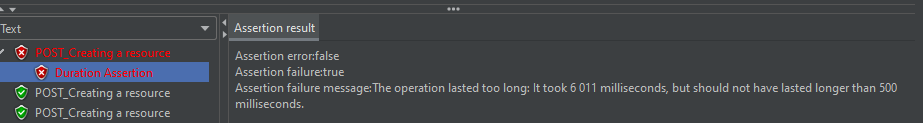
**Метод DELETE:** Response згідно з документацією.

|  |  |
| --- | --- |
| Очікуваний Response по документації | Фактичний Response |
| E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_13.png | E:\Роботка\#Фріланс\##QA ENGINEER\Beetroot Academy\ДЗ\Урок 24\Screenshot_11.png |

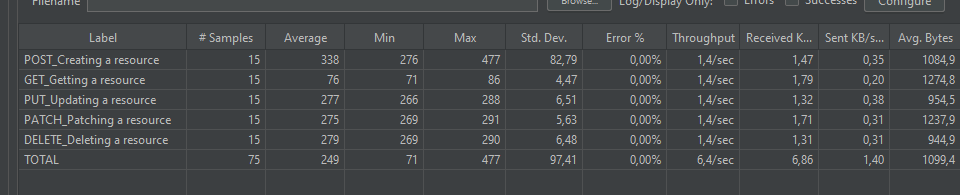
Метод POST видавав помилку змінного характеру (від 0% до 33% випадків) по Duration (уставка 500 мілісекунд), і запит POST відбувався довше, ніж 500 мілісекунд.

Інші методи в задану тривалість відповіді у 500 мілісекунд вкладалися.

****

****

Додатково я створила Listener Summary Report для всієї Thread Group.



5. Створений тест-план та документ з описом результатів виконання репортерів додай в свій репозиторій в GitHub.

**Другий рівень**

1. Створи новий тест-план на основі тест-плану з попереднього рівня. В ньому:

* для кожного із запитів зроби стрес-тест використаного API;
* потрібно виявити такі мінімальні комбінації параметрів, за яких вебсервіс перестає витримувати навантаження.

Створила новий тест-план Bryheda\_Thread Group2 на основі попереднього тест-плану Bryheda\_Thread Group.

В рамках Stress Testing я протестувала не тільки поведінку системи при зміні навантаження (кількості користувачів, кількість ітерацій), а й швидкість виконання операцій, бо тривалість відповіді сервера (Час відгуку) впливає на продуктивність бізнес-процесу.

Значення для перевірки тривалості відповіді сервера Duration Assertion базувалися на прийнятому часі:

1. Between 100ms and 200 ms is considered good.
2. Between 200ms and 1 second is considered acceptable, but should be improved.
3. Above 1 second is too slow

Rump-up period (sec) – час, впродовж якого будуть нарощуватися кількість юзерів (Період нарощування). Дозволяє зробити плавний і прогнозований старт.

Стрес-тест створювався з поступовим збільшенням навантаження, параметри навантаження і помилки методів наведені в таблиці.

Таблиця

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # | Duration Assertion  ms | Number of users | Rump-up period (sec)  Період нарощування | Loop count | Error % | Причина Error |
|  | **Тестування часу відгуку при різних навантаженнях** | | | | | |
| 1 | 200 | 1 | 10 | 1 | POST 100%  PATCH 100%  PUT 100% | Duration |
| 2 | 200 | 10 | 5 | 1 | POST 100%  PATCH 100%  PUT 100% | Duration |
| 3 | 200 | 10 | 20 | 1 | POST 100%  PATCH 100%  PUT 100% | Duration |
| 4 | 200 | 10 | 100 | 1 | POST 100%  PATCH 100%  PUT 100% | Duration |
| 5 | 300 | 1 | 10 | 1 | POST 100%  PATCH 50%  PUT 50% | Duration |
| 6 | 400 | 1 | 10 | 1 | POST 100% | Duration |
| 7 | 400  500 POST | 10 | 10 | 1 | 0% |  |
| 8 | 400  500 POST | 20 | 10 | 1 | 0% |  |
| 9 | 400  500 POST | 20 | 10 | 2 | 0% |  |
|  | **Тестування помилок (часу відгуку і відповіді сервера) при різних навантаженнях** | | | | | |
| 10 | 400  500 POST | 20 | 10 | **10** | POST 0,5%  PATCH 0,5%  PUT 0,5% | Response 503  (для PUT, PATCH)  Duration  (для POST) |
| 11 | 400  500 POST | 20 | 5 | 2 | PUT 2.5% | Response 503 |
| 12 | 400  500 POST | 20 | 5 | 10 | POST 5,5%  PATCH 1%  DELETE 0.5% | Response 503  (для PATCH)  Duration  (для PATCH, POST, DELETE) |
| 13 | 400  500 POST | 30 | 10 | 2 | POST 3,3% | Duration |
| 14 | 400  500 POST | 40 | 10 | 2 | POST 3,75%  PATCH 1,25% | Duration |
| 15 |  | 40 | 20 | 2 | POST 16%  PATCH 2,5%  DELETE 2,5% | Duration |
|  | **Тестування відповіді сервера при збільшенні юзерів** | | | | | |
| 16 | 400  500 POST | 500 | 30 | 1 | POST 9,8%  PUT 0,6%  PATCH 0,6% | Response 503  (для PATCH)  Duration  (для PATCH, POST, PUT) |
| 17 | 400  500 POST | 1000 | 30 | 1 | POST 42,2%  GET 0,1%  PUT 3,2%  PATCH 3,1%  DELETE 8,6% | Response 503  (для PATCH)  Duration  (для PATCH, POST, PUT) |
| 18 | Прибрано Duration Assertion | 1000 | 30 | 1 | POST 0,2%  PUT 0,1% | Response 503 |
| 19 | Прибрано Duration Assertion | 2000 | 30 | 1 | POST 0,25%  PUT 0,15% | Response 503 |
| 20 | Прибрано Duration Assertion | 3000 | 30 | 1 | POST 0,07%  PUT 0,07%  PATCH 0,03% | Response 503 |
| 21 | Прибрано Duration Assertion | 5000 | 30 | 1 | POST 3,08%  PUT 0,12%  PATCH 0,12% | Response 503 |
| 22 | Прибрано Duration Assertion | 6000 | 30 | 1 | POST 42,8%  GET 19,72%  PUT 0,35%  PATCH 0,05%  DELETE 0,02% | Response 503 |
| 23 | 400  500 POST | 10 | 10 | 1 | 0% |  |

Методи GET і DELETE вкладаються в тривалість відповіді сервера від 100 ms до 200 ms, при невеликих навантаженнях (кількість користувачів 1-10, кількість ітерацій 1) методи POST, PUT, PATCH не вкладаються в тривалість відповіді сервера від 100 ms до 200 ms - див. параметри навантаження в таблиці № 1-5.

Методи GET, PUT, PATCH, DELETE вкладаються до 400 ms для часу відгуку при навантаженнях зі значеннями - див. параметри навантаження в таблиці № 6-9.

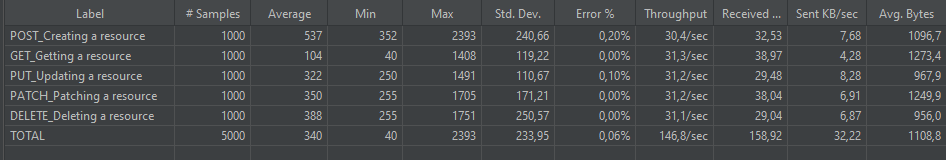
Метод POST вкладається в 500 ms - див. параметри навантаження в таблиці № 7-9.

При збільшенні навантаження (кількість користувачів 20, кількість ітерацій 2-10) помилки відповіді сервера 503 першими виникають в методі PATCH, потім PUT - див. параметри навантаження в таблиці №10-11.

При збільшенні кількості користувачів до 1000 перевищення тривалості відповіді сервера >400 ms спостерігається в усіх методах - див. параметри навантаження в таблиці №12-17.

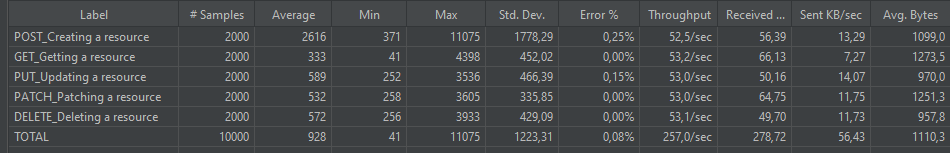
Для тестування при раптовому збільшенні навантаження (кількість юзерів 1000, 2000, 3000) я тимчасово прибрала Duration Assertion, щоби подивитися, скільки в % буде помилок відповіді сервера.

При навантаженні 1000 юзерів методи POST, PUT видають перші помилки відповідв сервера (таблиця п.18). Середній час відгуку для POST складає більше 0,5 сек, інші методи не перевищують середній час відгуку 0,4 сек.

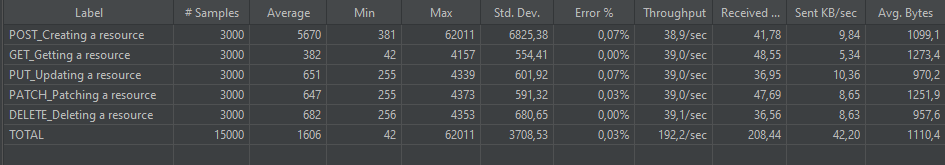


Навантаження в 2000 юзерів відсоток помилок для POST, PUT зростає (таблиця п.19)

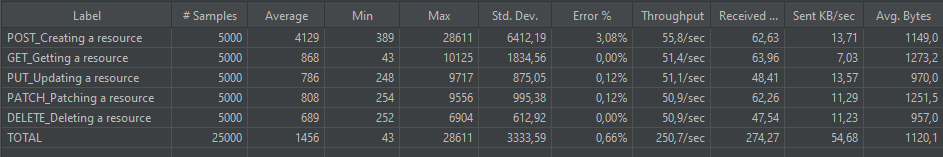
Середній час відгуку для POST складає більше 2,6 сек, інші методи не перевищують середній час відгуку 0,6 сек.



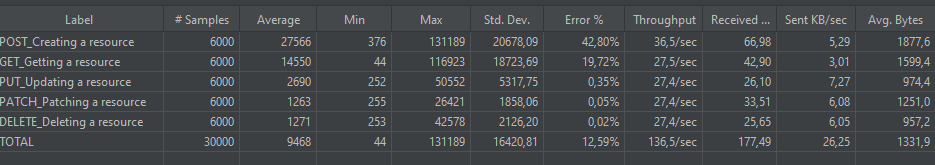
При навантаженні в 3000 юзерів зявилися перші помилки для PATCH (таблиця п.20)



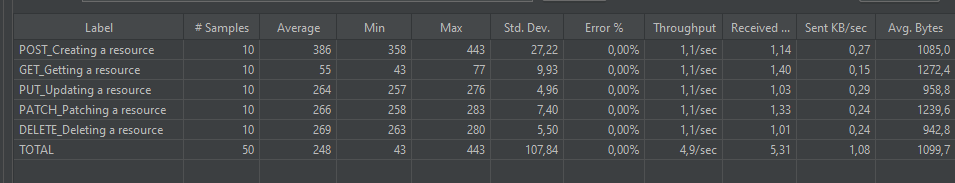
При навантаженні в 5000 юзерів помилки тільки в POST, PUT, PATCH (таблиця п.21)



Зміна навантаження до 6000 юзерів призводить до помилок у відповіді сервера в усіх методах (додалися помилки для GET і DELETE) (таблиця п.22)



Перевірка, чи повертається система після надмірного навантаження в 6000 юзерів до нормального режиму без помилок показала, що повертається, середній час відгуку для всіх методів не перевищує 400 мс. Кількість помилок відповіді сервера 0%.(таблиця п.23)



**Третій рівень**

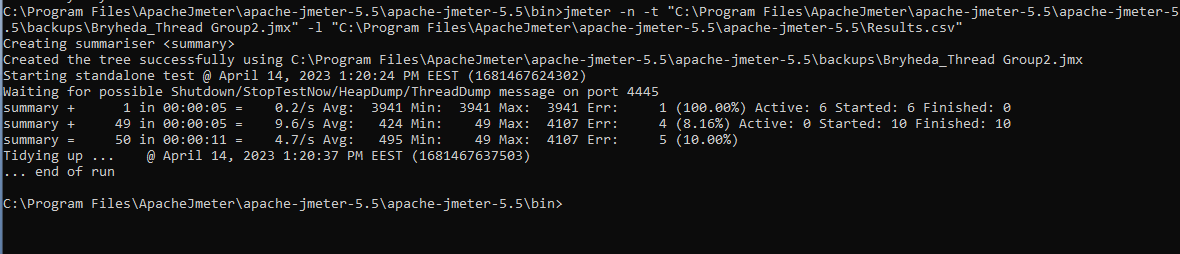
Створи команду для запуску вказаного тест-плану через Bash CLI. Запусти її та завантаж результати в Summary Report.

Консольний варіант може залучити більше ресурсів для тестування.

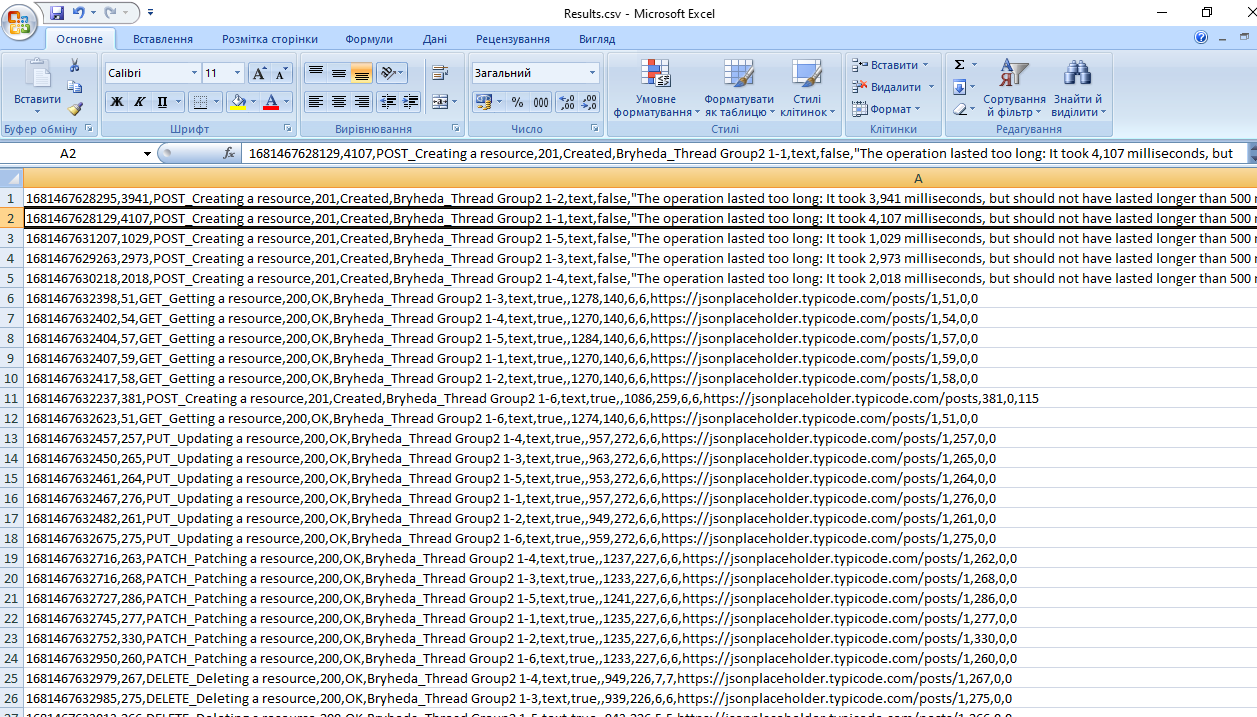
Запустимо тест-план без графічної оболонки jmeter в консолі

jmeter -n -t “C:\Program Files\ApacheJmeter\apache-jmeter-5.5\apache-jmeter-5.5\backups\Bryheda\_Thread Group2.jmx” -l “C:\Program Files\ApacheJmeter\apache-jmeter-5.5\apache-jmeter-5.5\Results.csv”

2. Створи документ зі скриншотом виконання команди в командному рядку та скриншотом Summary Report.



Файл з результатами тест-плану, який запускався з консолі



Створені тест-плани знаходяться у репозиторії в GitHub.

Посилання на репозиторії в LMS.