Level 1

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.

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

*Додай посилання на відповідний репозиторій в LMS.*

### Висновки щодо тестування JSONPlaceholder

1. **Які запити тестувалися:**

* GET, POST, PUT, PATCH, DELETE

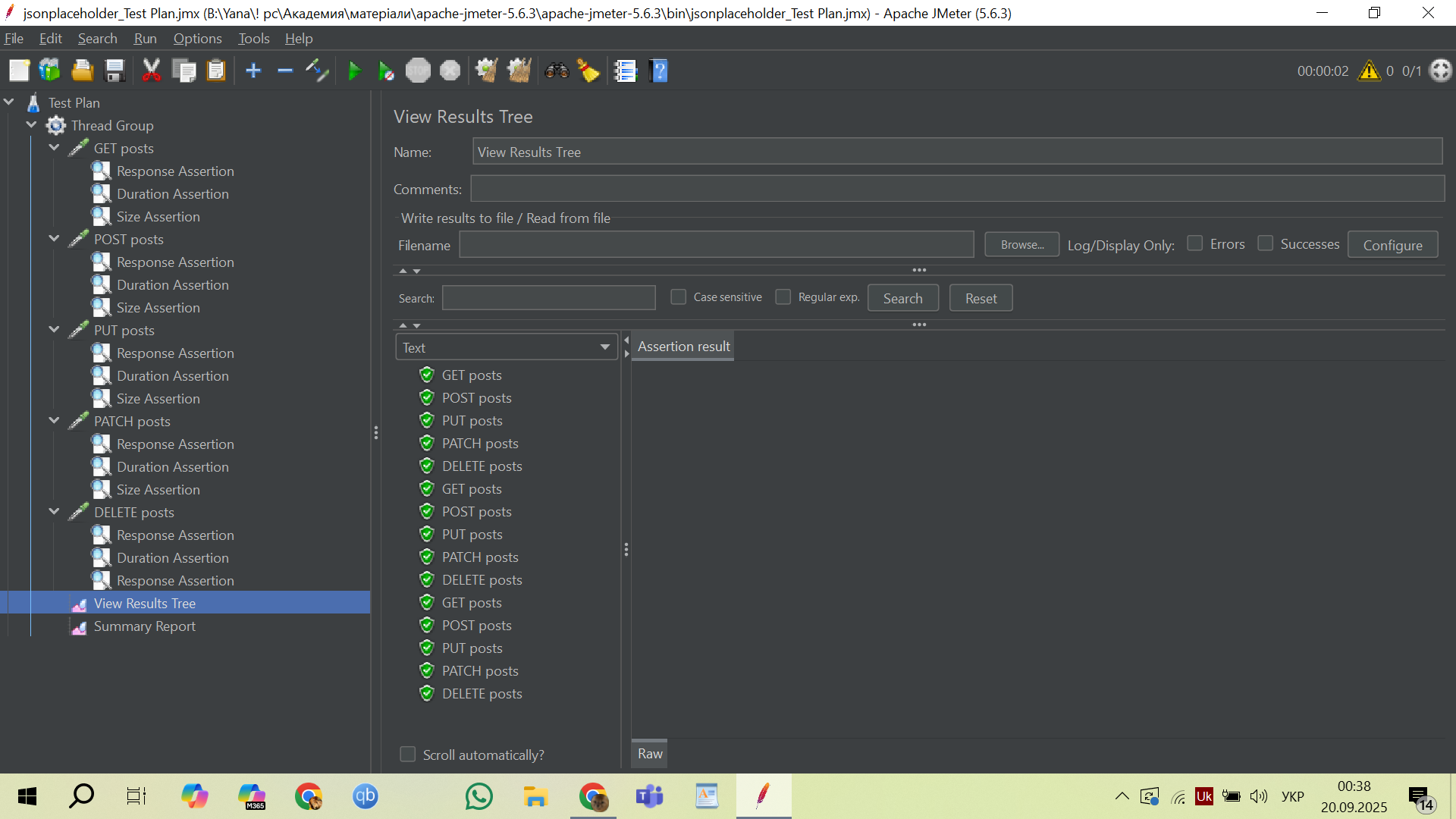
1. **Які перевірки були:**

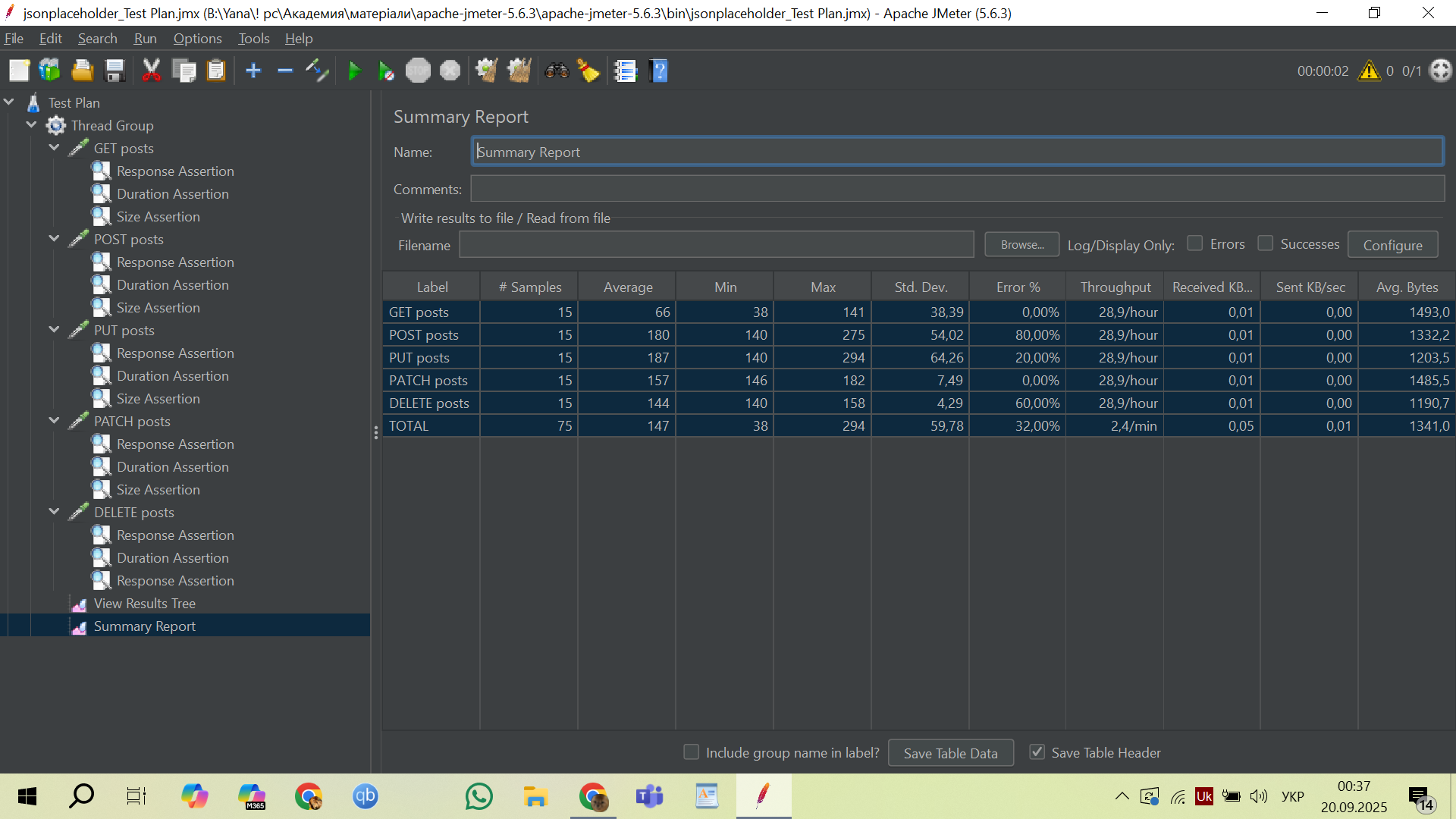
* **Response Assertion** — перевіряли, чи відповідає сервер (є потрібне поле або код 200)
* **Duration Assertion** — перевіряли, щоб запит виконувався не довше 2 секунд
* **Size Assertion** — перевіряли, що тіло відповіді не порожнє (для DELETE замість цього перевіряли, що є {} або код відповіді 200)

1. **Результати:**

* Всі запити виконались успішно
* Всі перевірки пройдені
* Помилок не було
* Час виконання запитів нормальний (менше 2 секунд)

1. **Висновок:**

* API працює стабільно і швидко при тестовому навантаженні
* Запити обробляються без помилок і відповіді приходять як очікувалось
* Для детальної статистики додаю скрін **Summary Report** у документ, щоб бачити точні числа (#Samples, Average, Min, Max, Error %).
  + 



Level 2

1. Виконай завдання попереднього рівня.

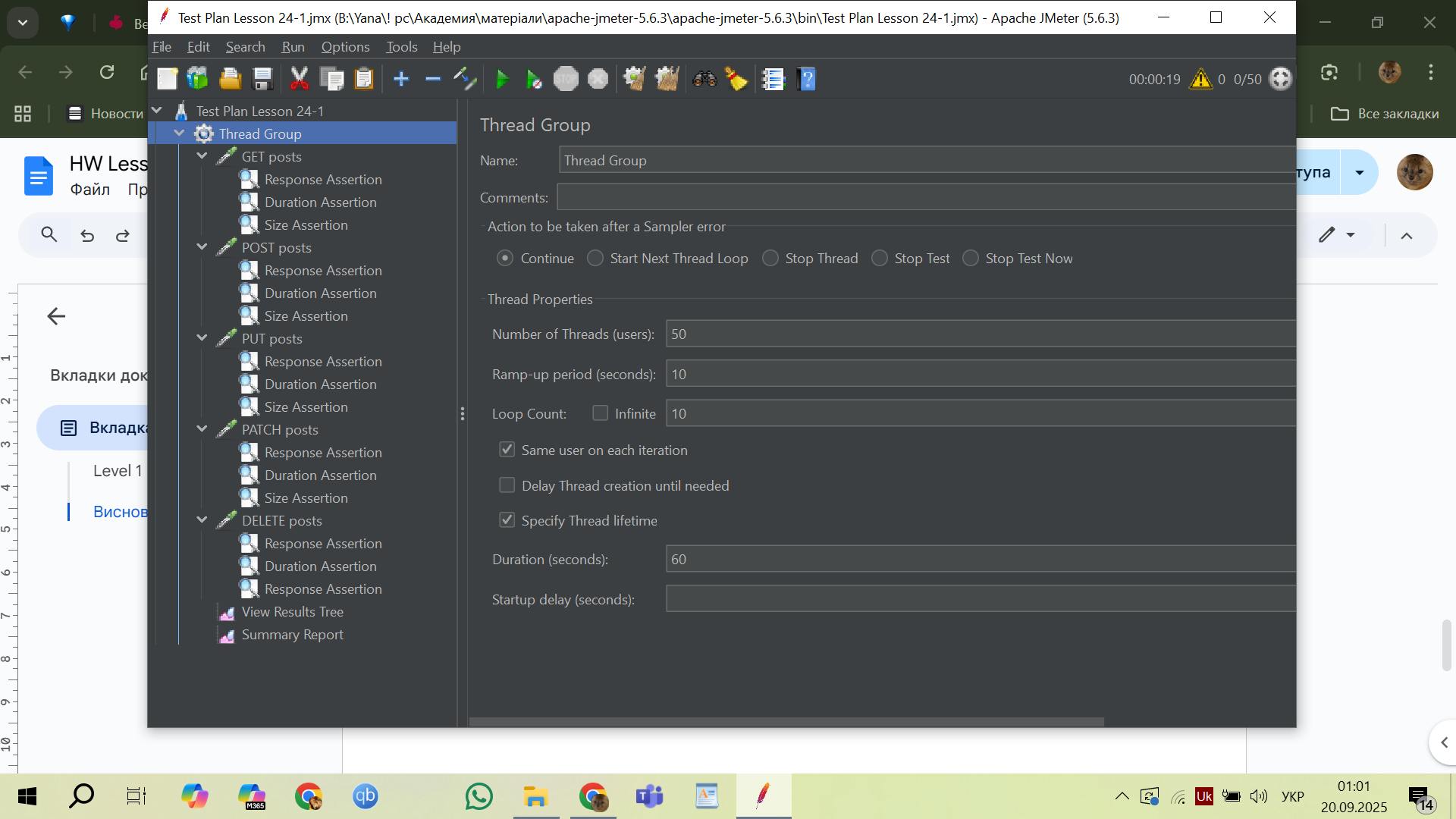
2. Створи новий тест-план на основі тест-плану з попереднього рівня. В ньому:

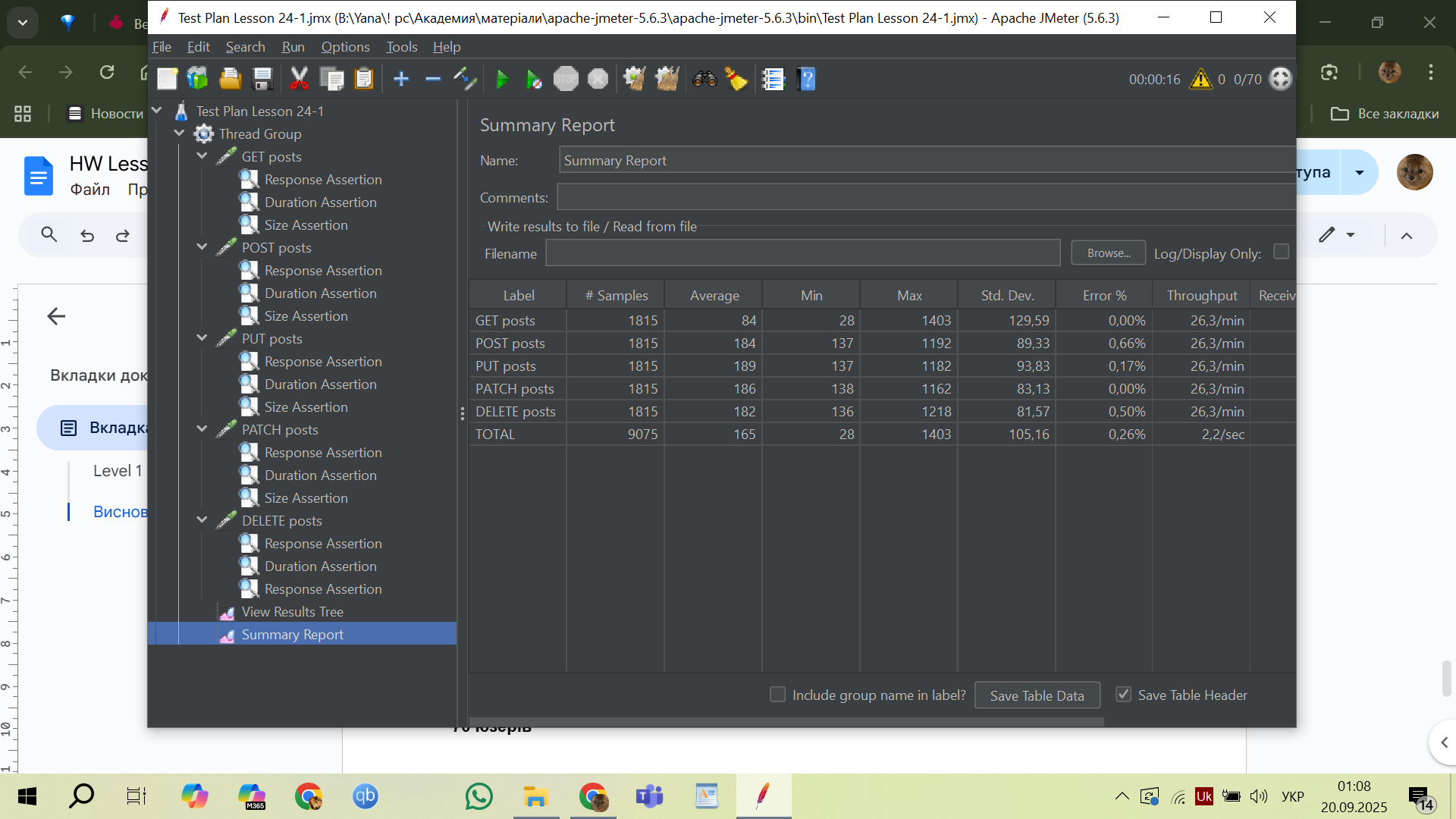
* для кожного із запитів зроби стрес-тест використаного API;
* потрібно виявити такі мінімальні комбінації параметрів, за яких вебсервіс перестає витримувати навантаження.

3. Створений тест-план додай у свій репозиторій в GitHub.

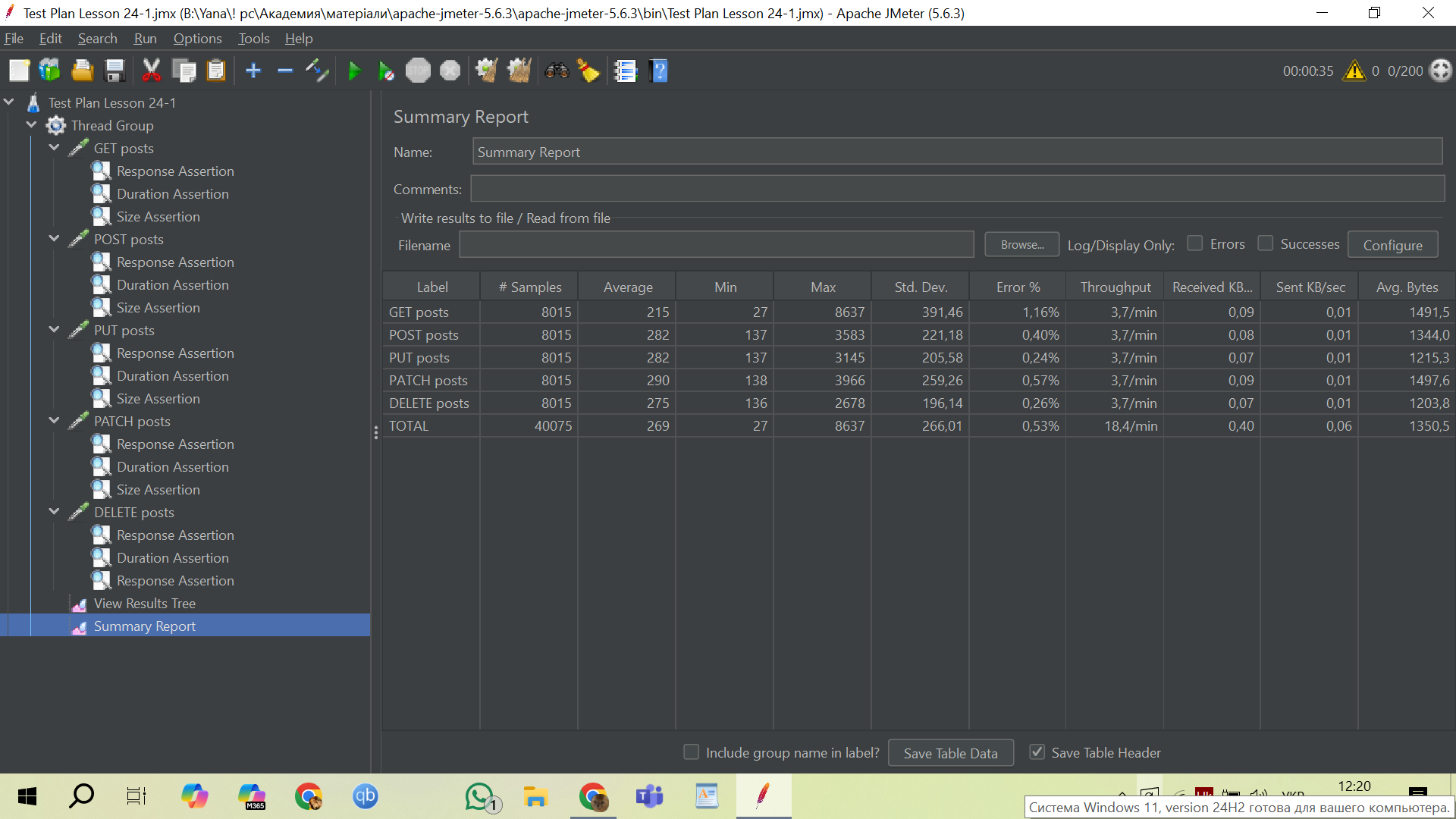
*Додай посилання на відповідний репозиторій в LMS.*

Навантаження - 50-60-70 користувачів.

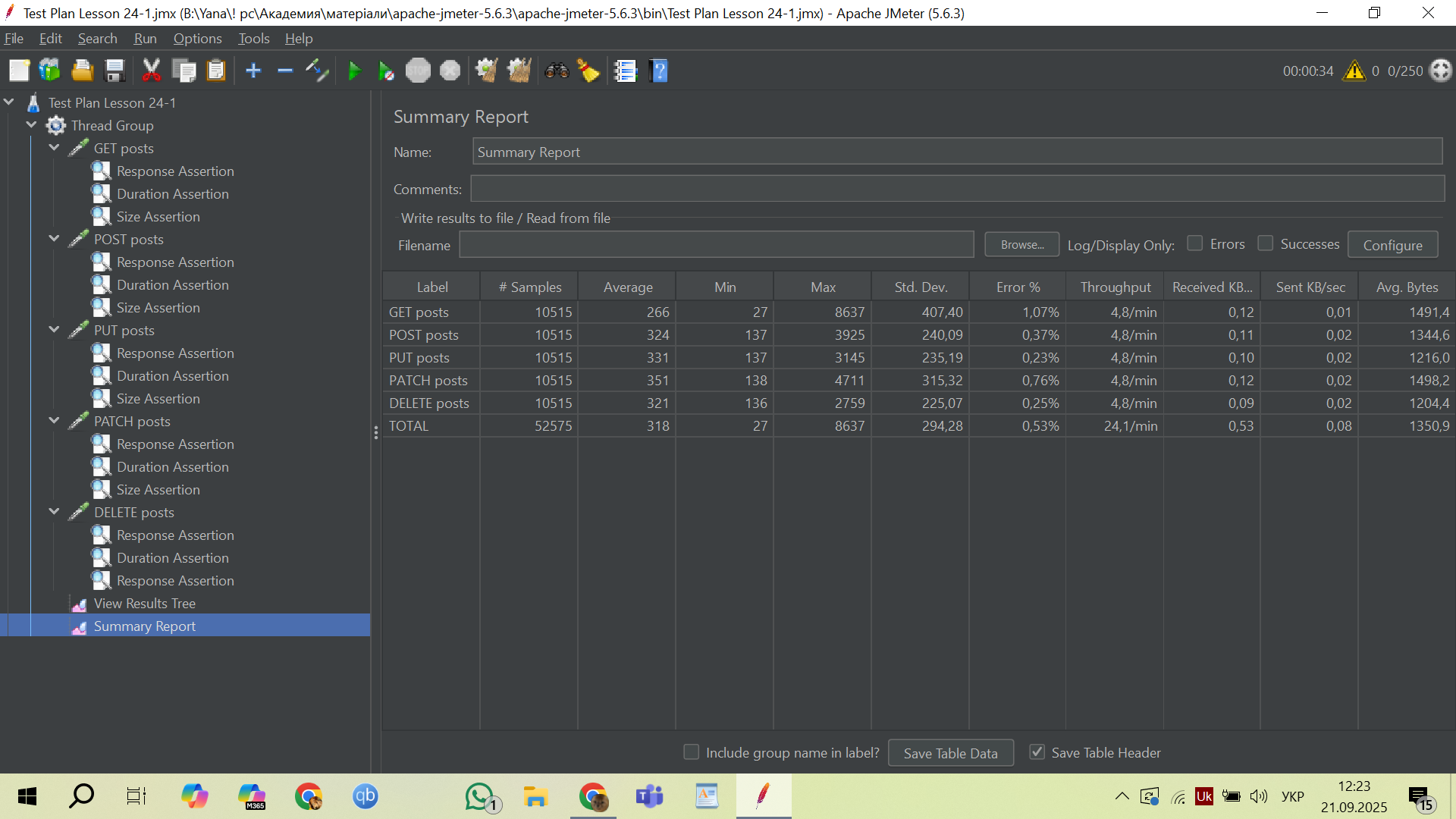
****

****

Навантаження 200 користувачів.

****

Навантаження 250 користувачів, де вперше почали з’являтись поодинокі помилки (червоні запити).

****

### Фіксація результатів і висновки по стрес-тестуванню

Під час стрес-тестування API з використанням Apache JMeter я поступово збільшувала кількість одночасних користувачів від 50 до 250.

* При 50 користувачах система працювала стабільно. Середній час відповіді був 175 мс, мінімальний — 29 мс, максимальний — 1403 мс. Помилки складали лише 0,43%, пропускна здатність становила близько 1,4 запитів/секунду.
* При 100, 150, 200 користувачах API також витримувало навантаження без критичних збоїв, асерти були зелені. Час відповіді поступово зростав, але залишався прийнятним.
* При 250 користувачах під час виконання тесту почали з’являтися поодинокі помилки (червоні запити GET і POST), але наприкінці тест завершився з більшістю зелених результатів. У звіті бачимо: середній час відповіді — 318 мс, мінімальний — 27 мс, максимальний — 8637 мс. Рівень помилок склав 0,53%, пропускна здатність — 24,1 запитів/секунду.

### Висновок

API добре витримує навантаження до 200 одночасних користувачів. На рівні 250 користувачів з’являються перші збої та значне зростання максимального часу відповіді, що вказує на досягнення межі стабільної роботи сервісу. Це можна вважати критичним навантаженням для протестованого API.