### **✅ Частина 1: Без cache(), лише фінальне .collect()**

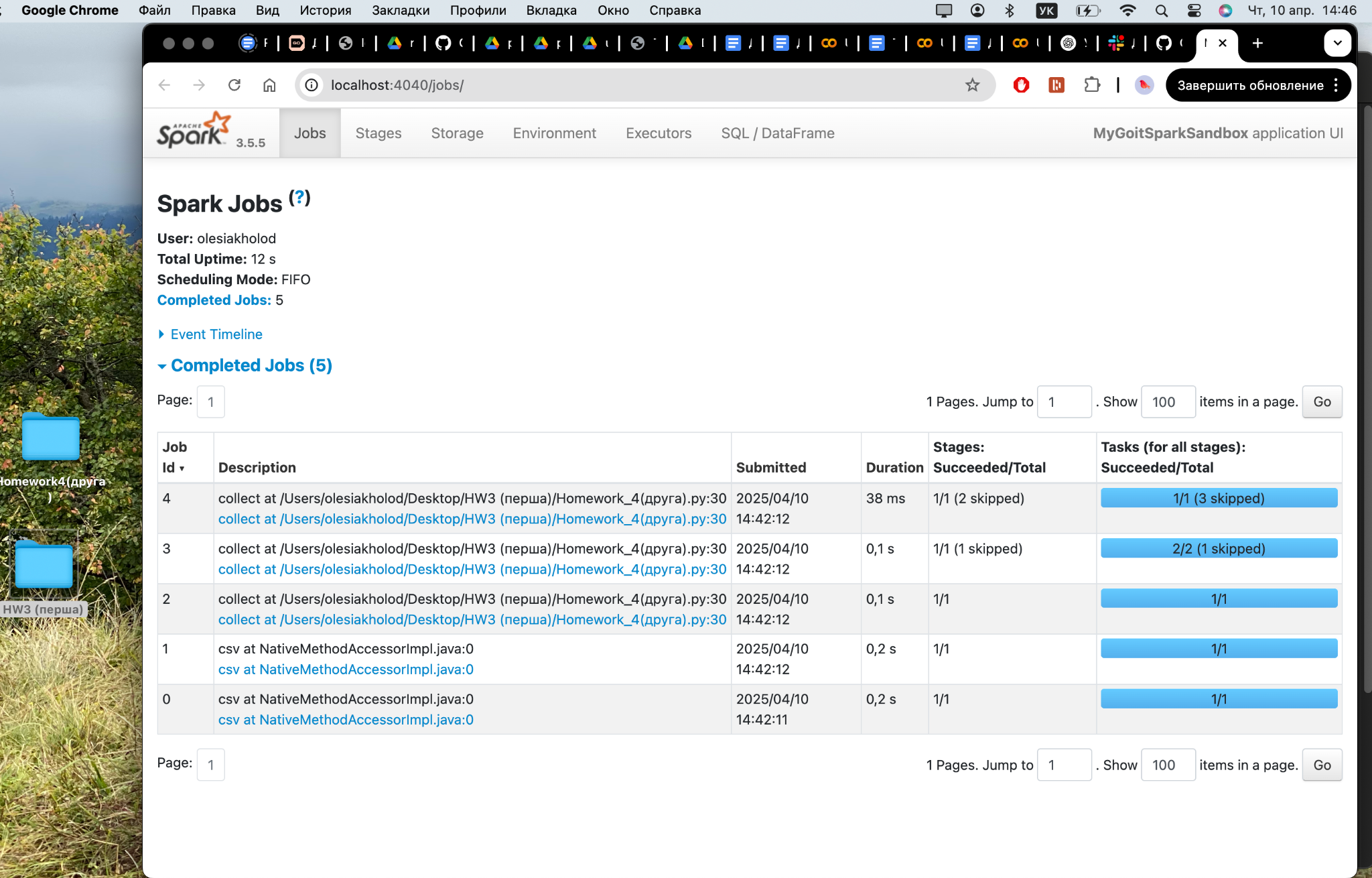
📌 **Кількість Jobs: 5**

#### **Що відбулося:**

1. Spark читає CSV (1 Job).
2. Далі — фільтрація, селекція, агрегація (групування та підрахунок), потім — ще одна фільтрація.
3. Всі ці трансформації виконуються **одним обчислювальним графом DAG**, оскільки виконання запускається тільки під час .collect() наприкінці.

#### **Чому 5:**

* 1 Job на читання даних.
* 1 Job на кожну стадію DAG, що включає repartition, where, select, groupBy, where, collect.
* Spark ділить DAG на кілька логічних Job’ів (разом — 5 Jobs), бо кожна з цих трансформацій може викликати окремий stage, залежно від шардінгу та shuffle.



**✅ Частина 2: Додано .collect() посередині**

📌 **Кількість Jobs: 8**

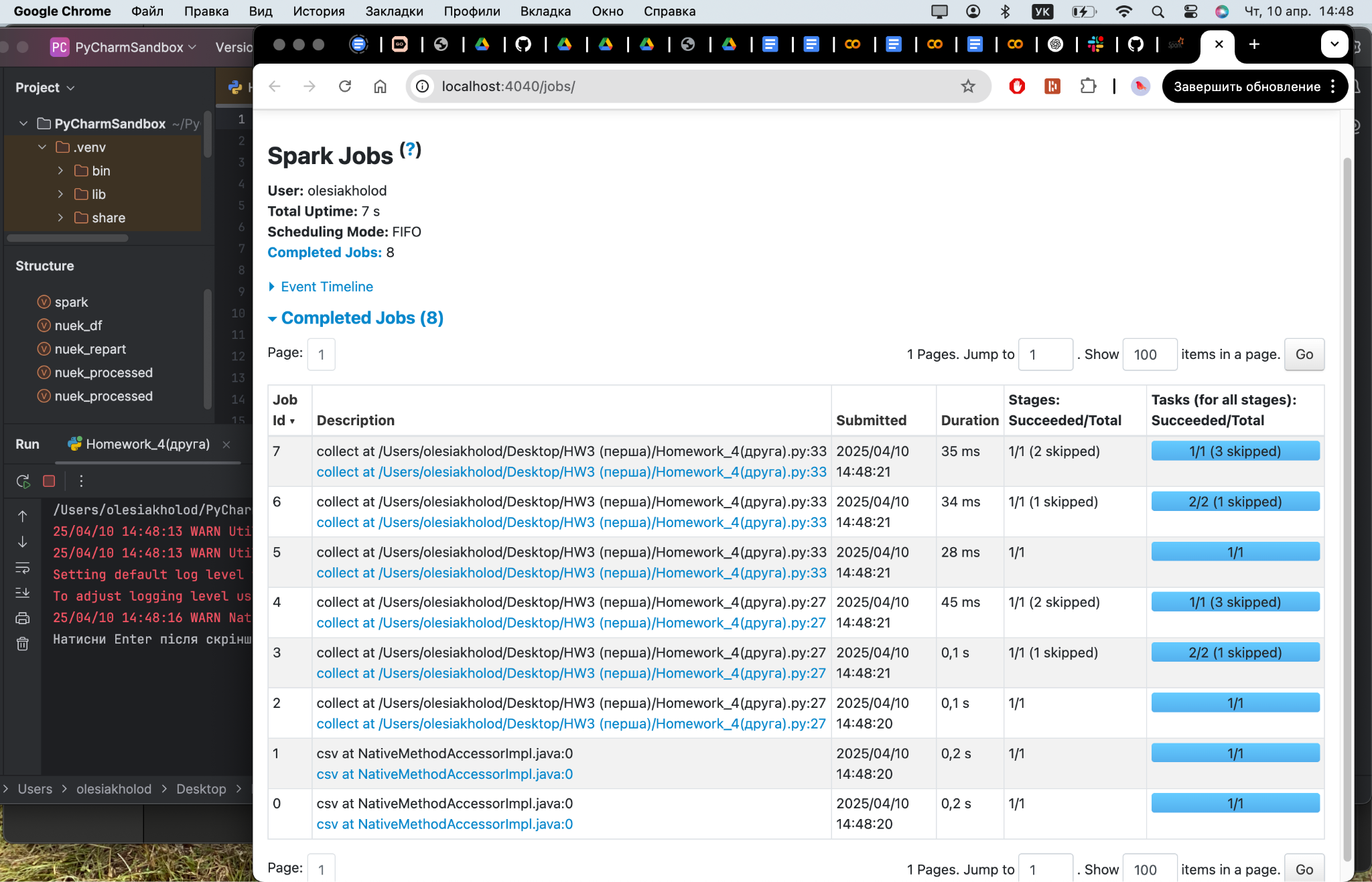
#### **Що змінилось:**

* Тепер .collect() виконується **до** фінального .where("count > 2"), тобто на ще не відфільтрованому DataFrame.

#### **Чому більше Job’ів:**

* Коли виконується .collect(), Spark **виконує всі попередні трансформації до цього моменту**.
* Це означає, що Spark запускає окремий DAG **двічі**:  
  1. Один — до .collect() (читання, фільтр, селект, групування, count).
  2. Інший — після .where("count > 2") і нового .collect().

➡️ Таким чином, з’являються **три нових Job’и**, які обчислюють ті самі дані ще раз, бо Spark не зберігає попередній результат у пам’яті.



3. **✅ Частина 3: Використано .cache()**

📌 **Кількість Jobs: 7**

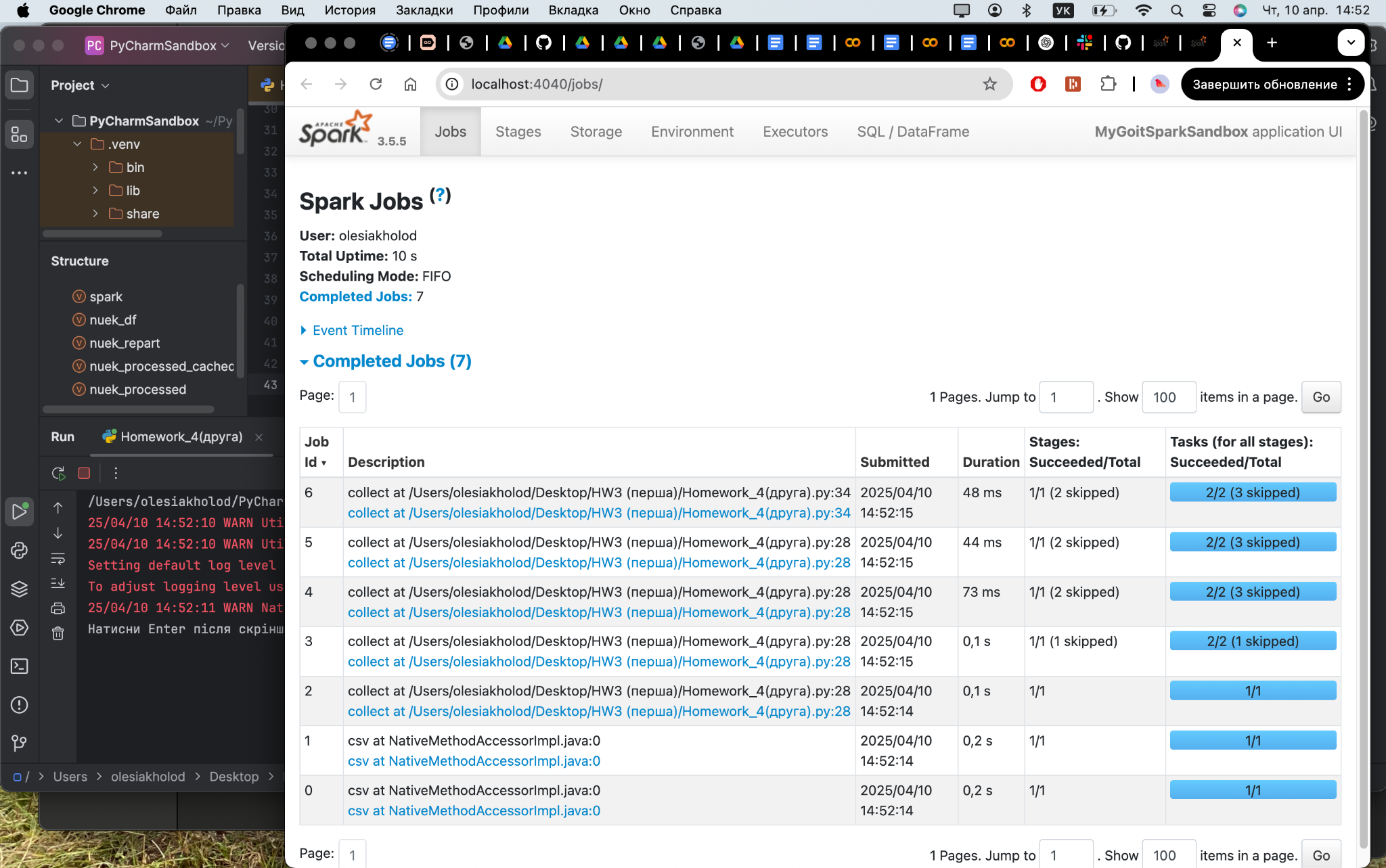
#### **Що відбулося:**

* .cache() зберігає результат першої частини (where, select, groupBy, count) у пам’яті.

#### **Чому менше Job’ів:**

* Коли викликається .collect() на nuek\_processed\_cached, Spark **вперше** обчислює і кешує результат.
* Потім, коли виконується nuek\_processed = nuek\_processed\_cached.where("count > 2"), Spark бере результат **з кешу**, не обчислює все знову.

➡️ Отже, Spark не виконує повний DAG двічі — і ми економимо Job.



* .collect() — це action, що змушує Spark виконати всі попередні трансформації.
* .cache() — дозволяє зберігати результат у пам’яті, щоб не перераховувати його знову.
* Тому .collect() без кешу → більше Job.
* .cache() перед .collect() → менше Job, бо використовується пам’ять.