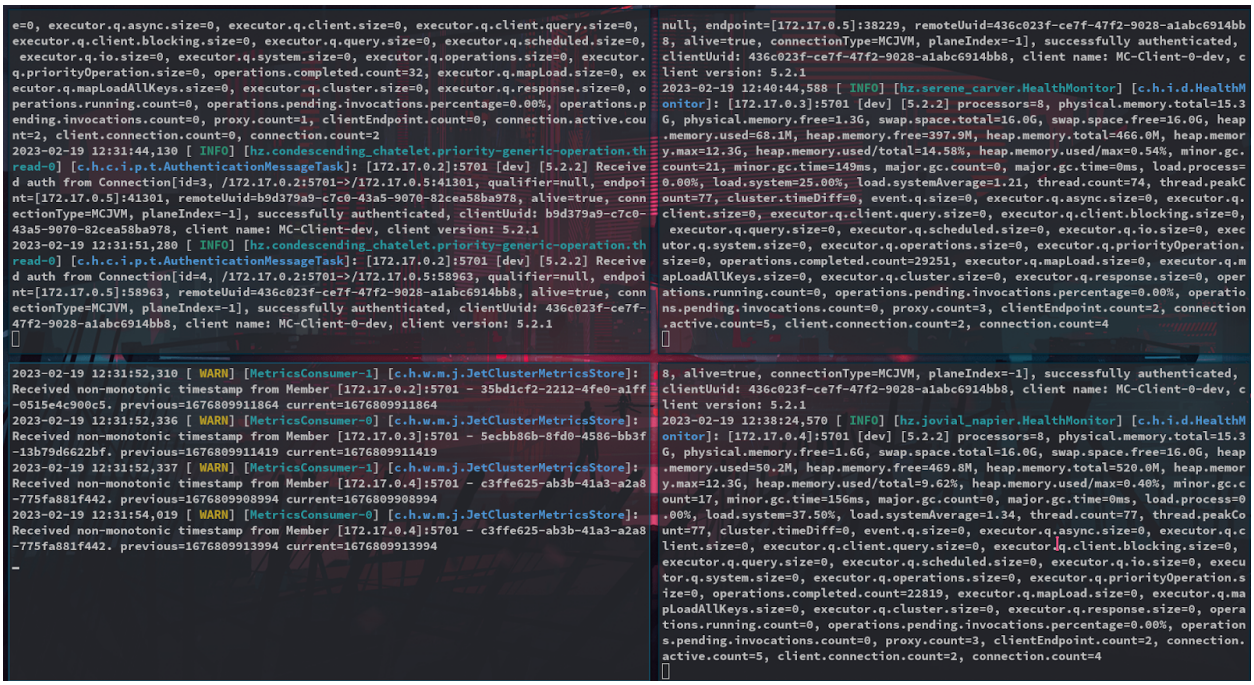
Lab 2

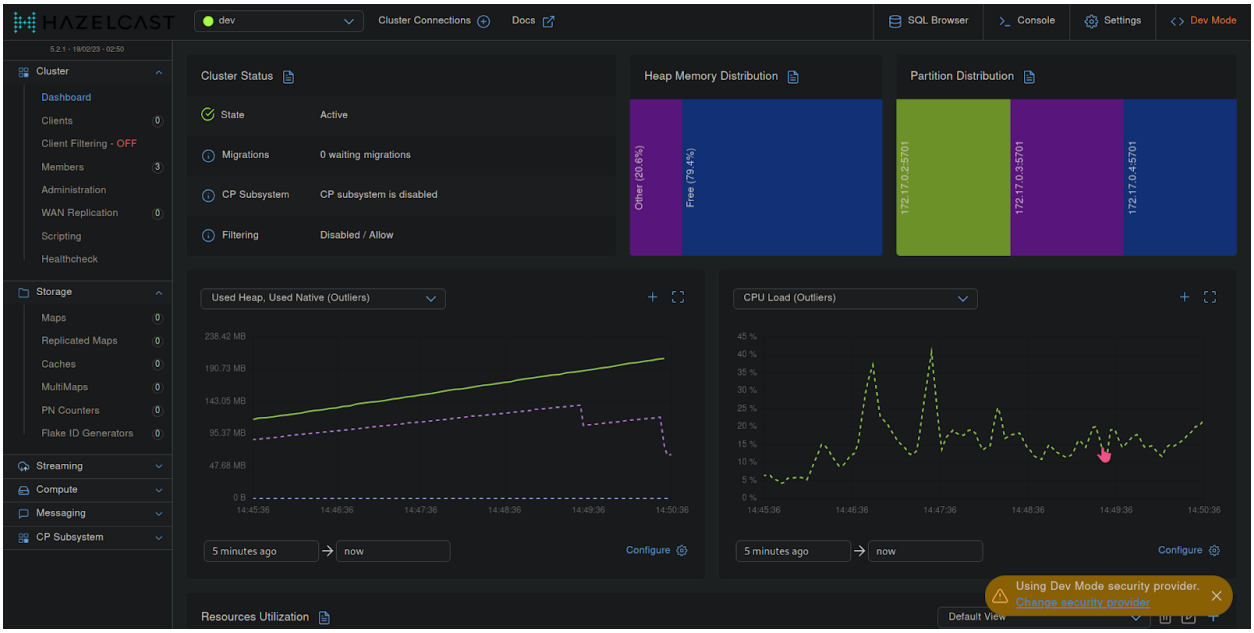
Виконав ***Остап Труш***

Github

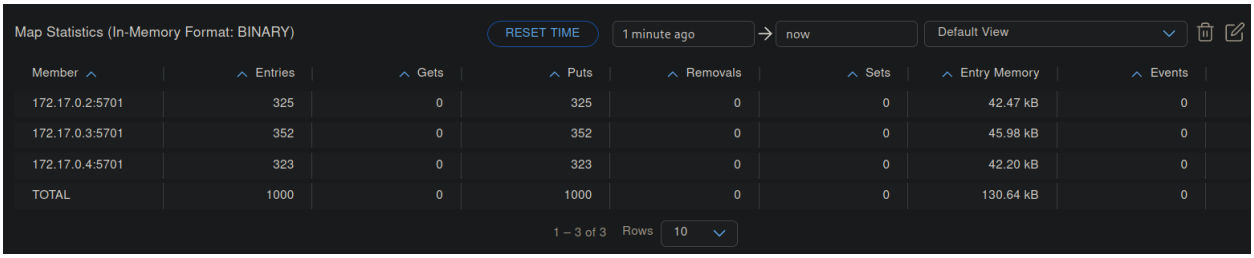
https://github.com/Adeon18/MicroServices/tree/hazelcast_basics

1. Підняв 3 ноди, як окремі застосування та запустив менеджмент центр(він тут зліва знизу). Під’єднався до інтерфесу менеджмент центру





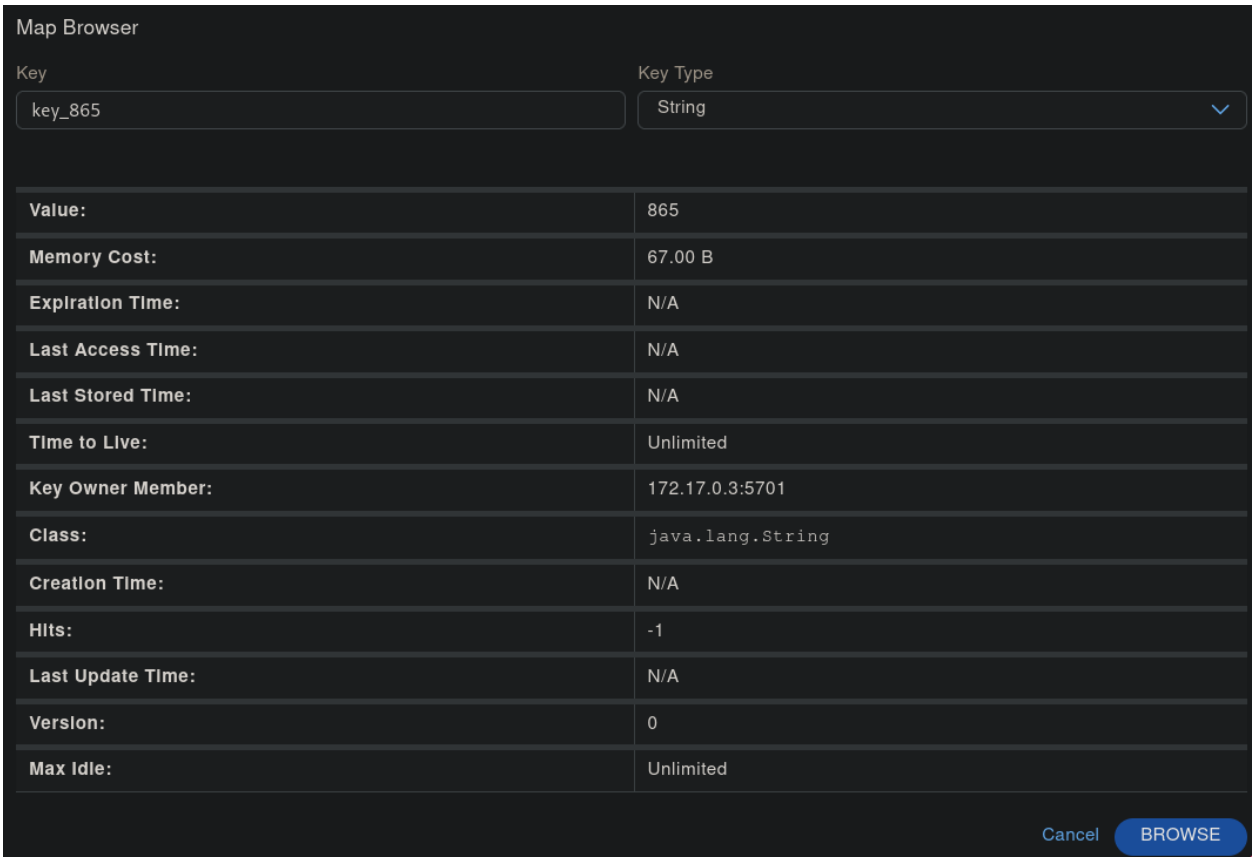
2. Запускаю код для запуску 1000 значень в “my-distributed-map”



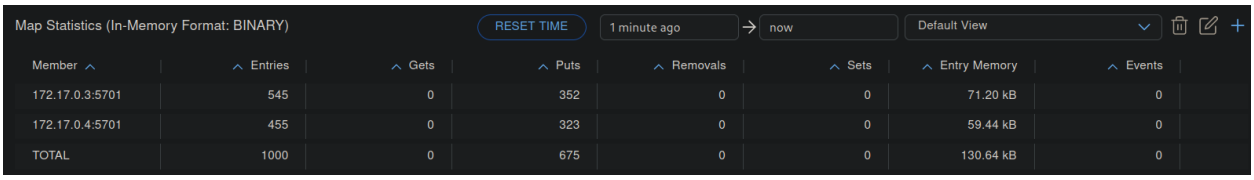
Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.17.0.2:5701	325	0	325	0	0	42.47 kB	0
172.17.0.3:5701	352	0	352	0	0	45.98 kB	0
172.17.0.4:5701	323	0	323	0	0	42.20 kB	0
TOTAL	1000	0	1000	0	0	130.64 kB	0

1 – 3 of 3 Rows 10

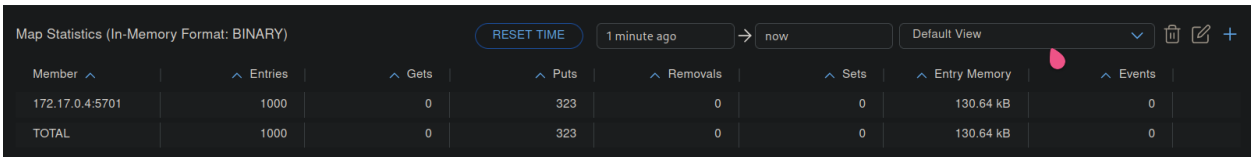
- Дані +- рівномірно розподілилися по трьох мапах, при пошуку ключів по мапі, отримую правильний результат.



- Зупиняю першу мапу - дані не втратилися. Браузинг по контентах мапи далі працює.



- Зупиняю другу мапу - дані не втратилися, всі дані зберігаються в одній ноді.



3. Продемонструйте роботу Distributed Map with locks

Я запускатиму багатопоточну програму написану на C++ і з 3 потоків одночасно писатиму в мапу. Програма є на гілці `hazelcast_basics` а саме `task2_datarace.cpp` там

можна вибрати яку опцію(з яким локінгом взяти), та заранити.

- Без локінгу - бачимо результат далекий від 3000, а саме 1223.

```
Starting threads...
Thread 0 Started
Thread 1 Started
Thread 2 Started
Thread 0 Finished
Thread 1 Finished
Thread 2 Finished
Finished! Result = 1223
Done
```

- З песимістичним локінгом - бачимо що 3000 елементів попадають в мапу

```
Starting threads...
Thread 0 Started
Thread 1 Started
Thread 2 Started
Thread 0 Finished
Thread 1 Finished
Thread 2 Finished
Finished! Result = 3000
Done
```

- З оптимістичним локінгом - також всі 3000 елементів попадають у мапу.

```
Starting threads...
Thread 0 Started
Thread 1 Started
Thread 2 Started
Thread 0 Finished
Thread 1 Finished
Thread 2 Finished
Finished! Result = 3000
Done
```

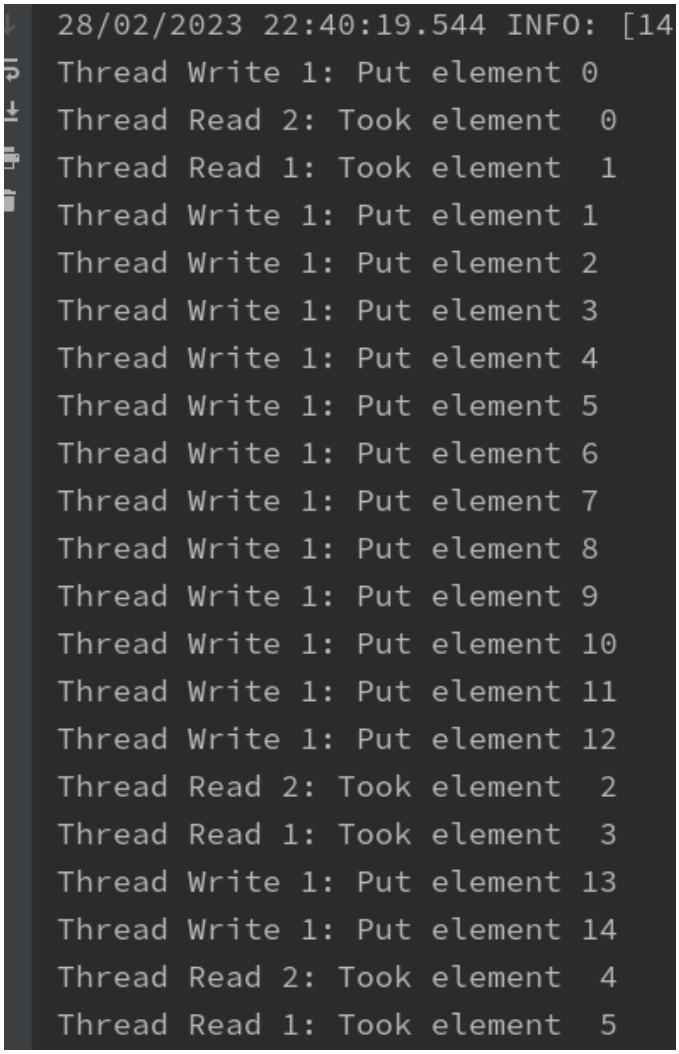
Документація по тому як це запустити та відтворити є в проєкті

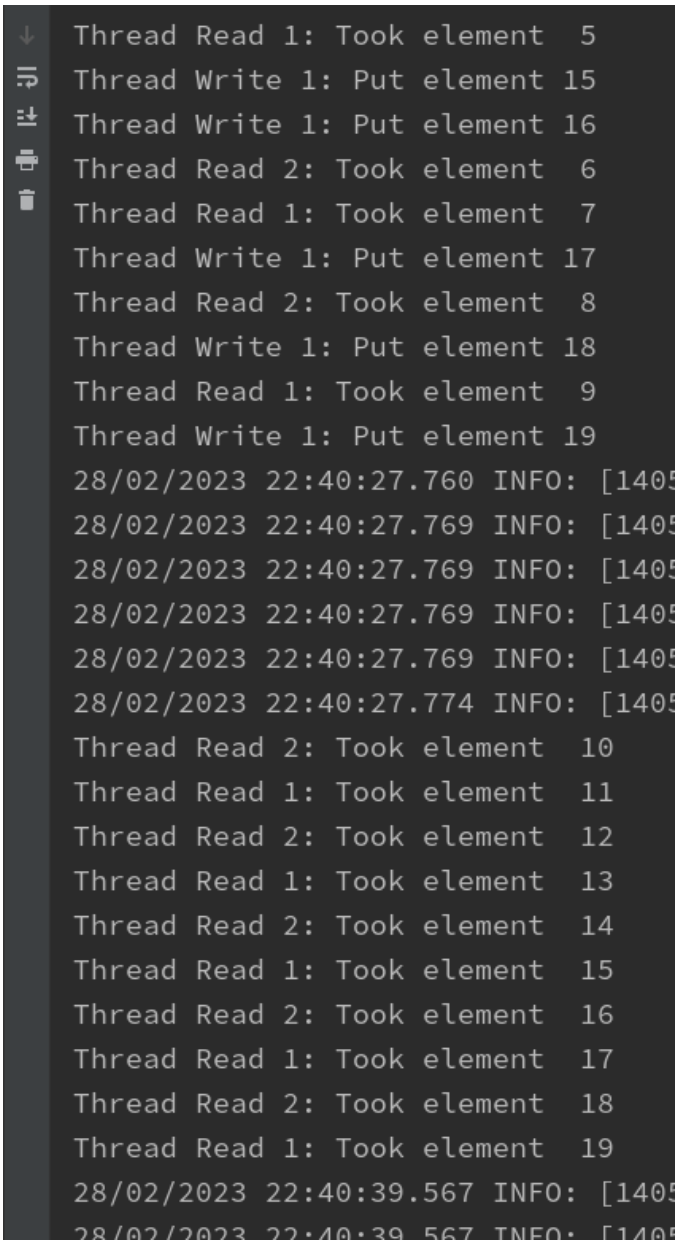
---

4. **Налаштуйте Bounded queue**

- з однієї ноди (клієнта) йде запис, а на двох інших читання

Один клієнт записує кожну `0.1` секунду(тотал записує 20 елементів), два клієнта читають кожних `2` секунди (читають по 10 елементів кожен), ліміт черги - `10`.

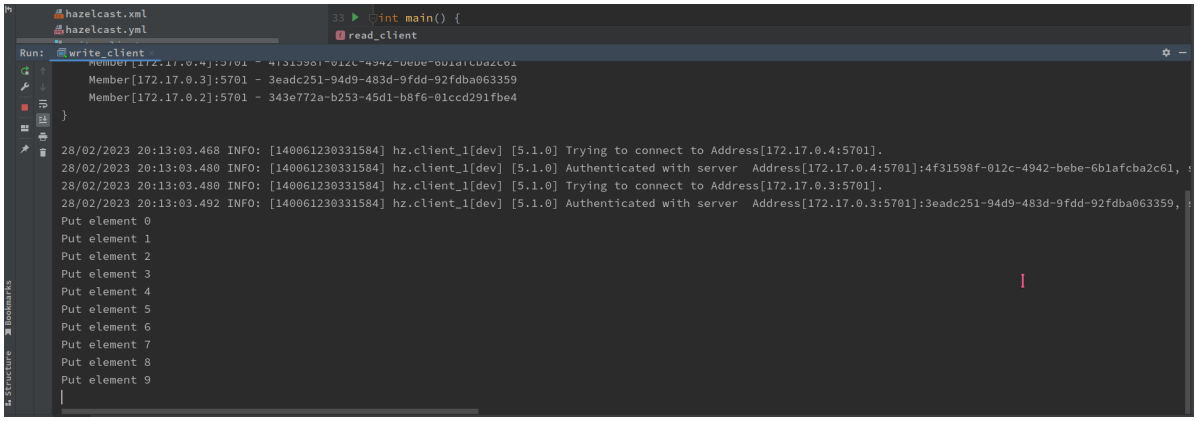




Можна побачити що елементи поклалися, ліміт черги досягнувся, і тоді вже тредами що читають витягуються елементи, як тільки витягнулися 2 елементи, з'явилося вільне місце, і туди той тред що пише зразу значення записав і забив чергу. Оскільки тред запису записує `20` значень, а треди читання читають по `10` жоден тред не лочиться

- перевірте яка буде поведінка на запис якщо відсутнє читання, і черга заповнена

При записі з одного потоку без читання в queue, при спробі запису 100 значень в чергу, ліміт якої `10` - мапа блокується, та при наступному колі до `put` потік лочиться та чекає допоки в черзі звільниться місце.



- як будуть вичитуватись значення з черги якщо є декілька читачів

  В чергу було запхано 10 елементів, і 2 потоки(клієнти) читання їх звідти читають одночасно(без зупинки на таймері). Результат наступний.