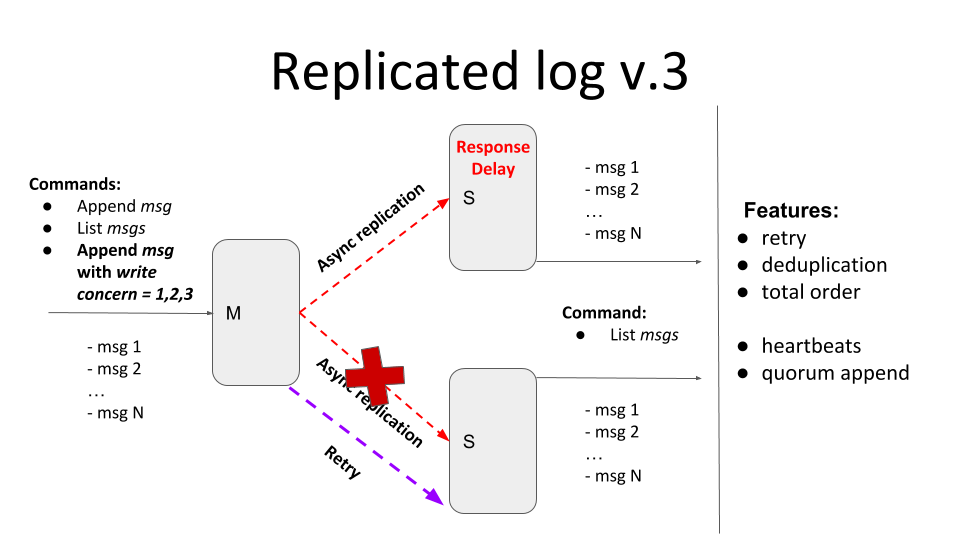
**Iteration 3.**

* **15 points**



The current iteration should provide tunable semi-synchronicity for replication with a *retry* mechanism that should deliver all messages *exactly-once* in total order.

Main features:

* If message delivery fails (due to connection, or internal server error, or secondary is unavailable) the delivery attempts should be repeated - *retry*
  + If one of the secondaries is down and *w=3*, the client should be blocked until the node becomes available. Clients running in parallel shouldn’t be blocked by the blocked one.
  + If *w>1* the client should be blocked until the message will be delivered to all secondaries required by the write concern level. Clients running in parallel shouldn’t be blocked by the blocked one.
  + All messages that secondaries have missed due to unavailability should be replicated after (re)joining the master
  + Retries can be implemented with an unlimited number of attempts but, possibly, with some “smart” delays logic
  + You can specify a *timeout* for the master in case if there is no response from the secondary
* All messages should be present exactly once in the secondary log - *deduplication*
  + To test deduplication you can generate some random internal server error response from the secondary after the message has been added to the log
* The order of messages should be the same in all nodes - *total order*
  + If secondary has received messages *[msg1, msg2, msg4]*, it shouldn’t display the message *‘msg4’* until the *‘msg3’* will bereceived
  + To test the total order, you can generate some random internal server error response from the secondaries

**У цій роботі реалізовано семі-синхронну реплікацію повідомлень між одним Master-вузлом і двома Secondary-вузлами. Ключові функції:**- Write-concern (w) для блокування відповіді клієнта до отримання потрібної кількості ACK.  
- Retry-механізм із нескінченними повторними спробами доставки до вторинок.  
- Дедуплікація у Secondary, щоб жодне повідомлення не з’явилося двічі.  
- Total-order через буферизацію «out-of-order» повідомлень до отримання пропущених.  
- Initial-sync у Secondary після запуску, щоб підтягнути всі пропущені записи із Master.

**1. master.py**

**Основні нові елементи в master.py:  
  
-** Retry‐механізм – якщо під час синхронної реплікації якийсь сервер недоступна або повернув помилку, цикл while True в sync\_replicate\_to\_secondary починає нескінченні спроби з паузою 2 сек. Поки останній сервер не «підніметься», master не відправить клієнту 200.  
  
- Метод /full\_messages – віддає увесь масив повідомлень із полями id, message, order. Це потрібно для механізму initial sync у вторинних вузлах.- Метод /clean – потрібен для коректного функціонування автоматичних тестів, цей виклик очистить messages.

**2. secondary.py**

- Initial Sync-- При старті сервіс робить GET /full\_messages до master.  
-- Отримує масив { id, message, order }, який сортує за order і додає в локальний список messages.  
-- Після цього expected\_order = len(messages) – всі нові POST /replicate будуть додаватися «з місця, де зупинилися».

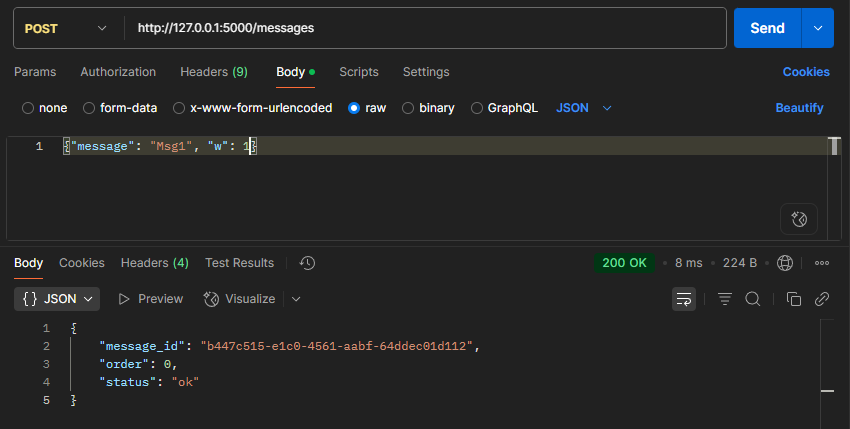
Self-check acceptance test:

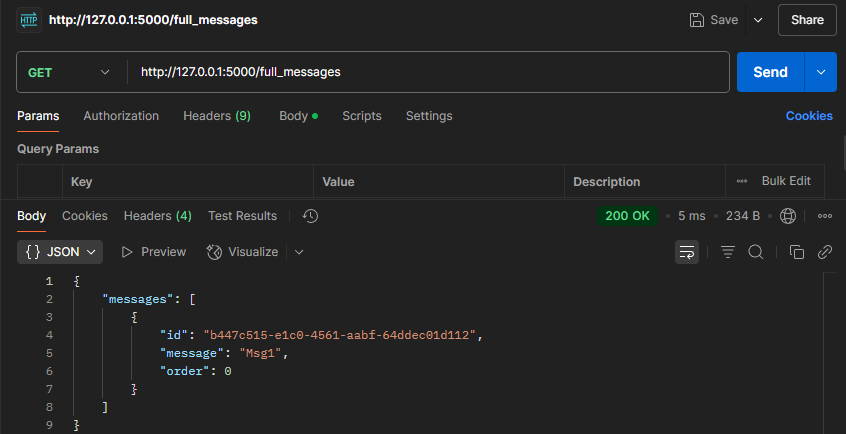
* Start M + S1
* send (Msg1, W=1) - Ok
* send (Msg2, W=2) - Ok
* send (Msg3, W=3) - Wait
* send (Msg4, W=1) - Ok
* Start S2
* Check messages on S2 - [Msg1, Msg2, Msg3, Msg4]

Start M + S1

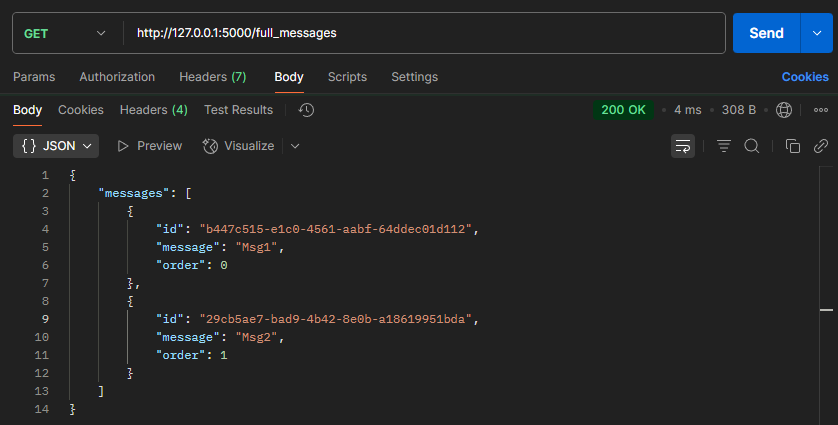
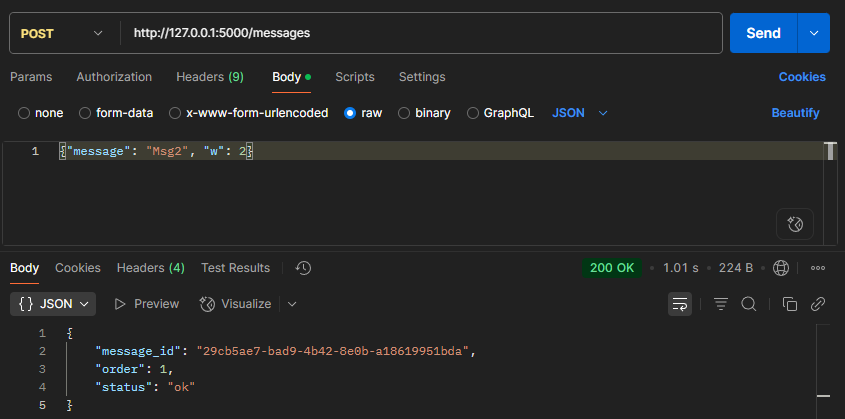


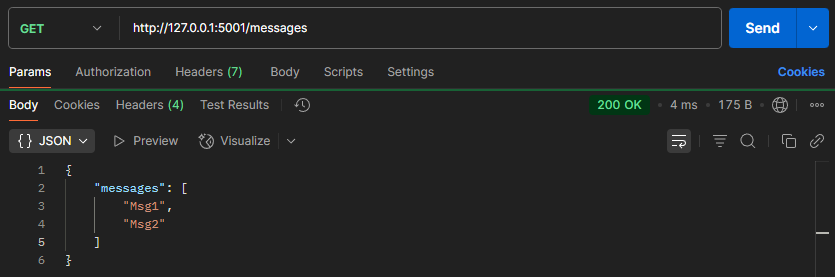
send (Msg1, W=1) — Ok



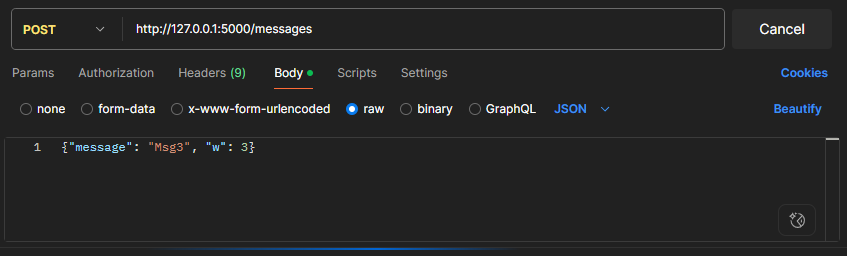


send (Msg2, W=2) — Ok

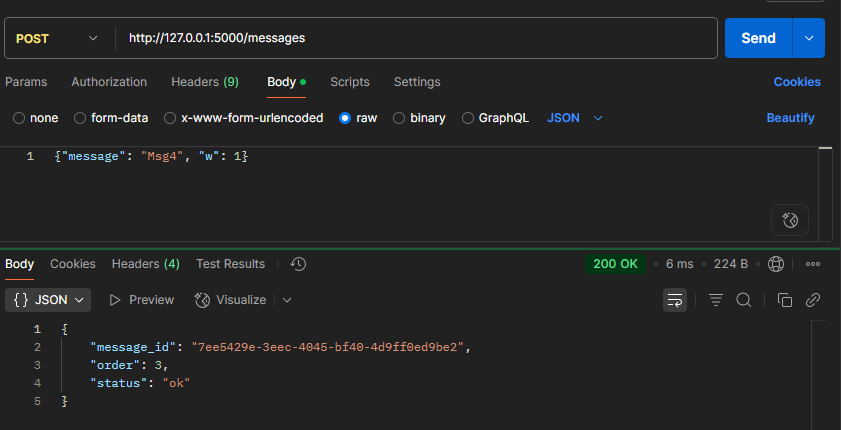


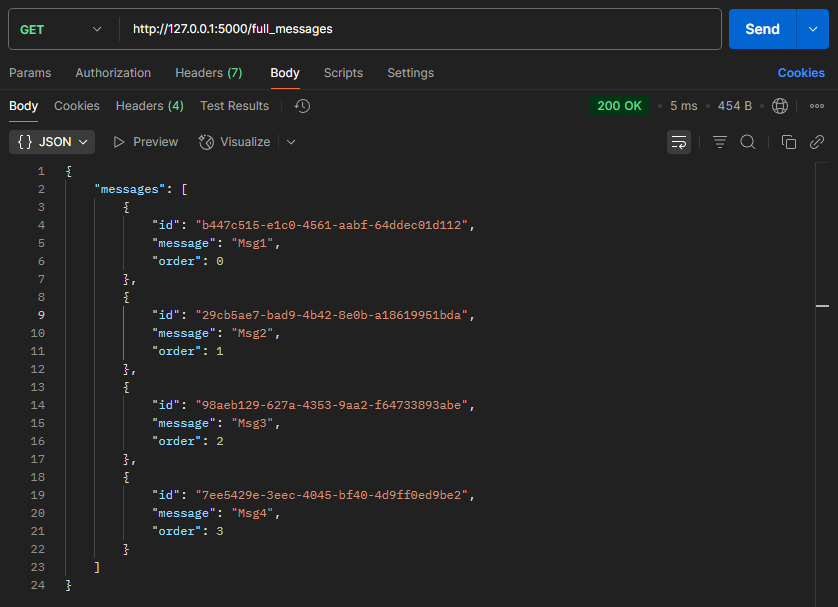


send (Msg3, W=3) - Wait



send (Msg4, W=1) - Ok

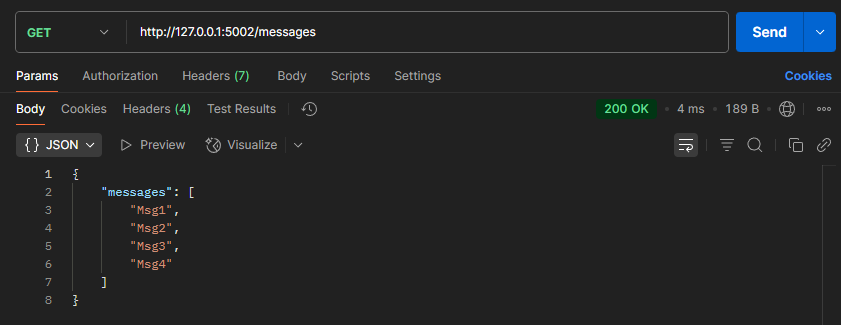




Start S2



Check messages on S2 - [Msg1, Msg2, Msg3, Msg4]



Перевіряю тепер автоматичними тестами, в них перевіряється в більшості все те саме, але автоматично.

Тести:  
1) перевірка що при w=3 master буде чекати запуску усіх secondary, перевірка доставки після запуску, перевірка що сервіс не блокується для інших запитів

2) перевірка що нода отримала всі повідомлення після запуску

3) перевірка на подвійне отримання

4) перевірка на правильний порядок при помилці в доставці

